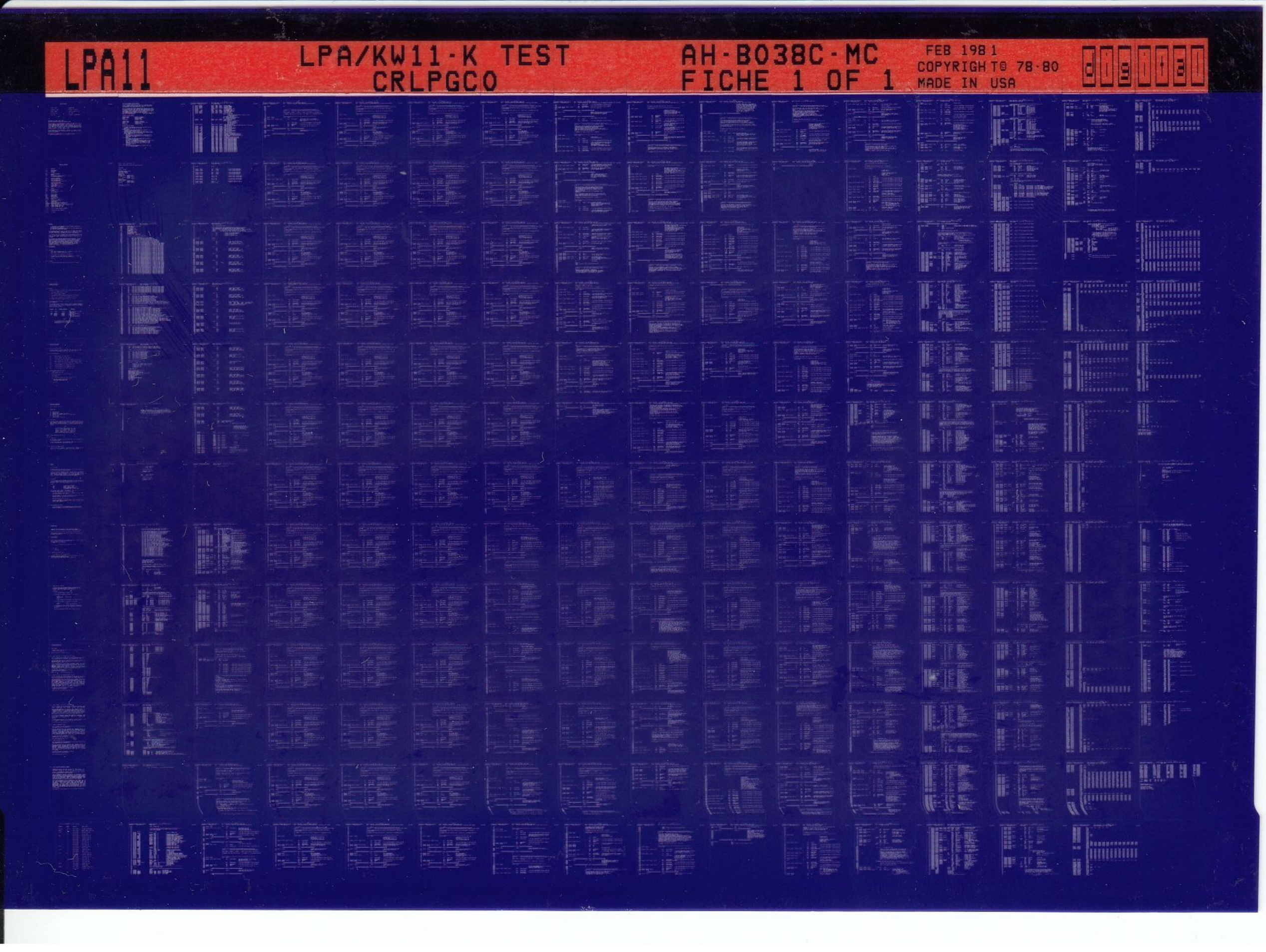
LPA11

LPA/KW11-K TEST
CRLPGCO

AH-B038C-MC
FICHE 1 OF 1

FEB 1981

MADE IN USA



Identification
****

Product Code:	AC-B037C-MC
Diagnostic Code:	MAINDEC-11-CRLPG-C
Product Name:	CRLPGC0 LPA/KW11K TEST
Date Revised:	DEC. 1980
Maintainer:	DIAGNOSTIC ENGINEERING

Table of Contents

1.0 ABSTRACT

"C" VERSION OF THE PROGRAM IS DUE TO A CHANGE IN THE MICRO-CODE IN THE LPA-11 (DMC-11).
"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-CRLPA" 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 (a 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)
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

For each pass through the program "END PASS" will be 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
****

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

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

The program is chainable under XXDP. However only one pass may be made (R RLPGC0/1).

8.0 MISCELLANEOUS
****

After a power failure occurs, program execution will NOT continue at the point where the power failure occured.
The program will be restarted.

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

Logic Test

90 SECONDS if 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 External 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 "LO STAT A HI" H + "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-KX DIAGNOSTIC KIT WILL INCLUDE:

OPTION	GROUP	DIAG. #	DIAG. TITLE
LPA11-KX	LEVEL 2	MD-11-CRLPA	LPA11-K SYSTEM EXER.
M8254	"B"	MD-11-CRLPM	M8254 (IPBM) FIELD DIAG.
AA11-K	A	MD-11-CRLPB	LPA/AA11-K DIAG.
	B	MD-11-DZAAC	AA11-K DIAG.
AR11	A	MD-11-CRLPC	LPA/AR11 DIAG. #1
	A	MD-11-CRLPD	LPA/AR11 DIAG. #2
	A	MD-11-CRLPE	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-DZDRG	DR11-K DIAG.
KW11-K	A	MD-11-CRLPG	LPA/KW11-K DIAG.
	B	MD-11-DZKWK	KW11-K DIAG.
LPS-11	A	MD-11-CRLPH	LPA/LPS11 DIAG. #1
	A	MD-11-CRLPI	LPA/LPS11 DIAG. #2
	A	MD-11-CRLPJ	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-CRLPL	LPA/DMC-11 DIAG. TST I
	B	MD-11-CRLPM	LPA/DMC-11 DIAG. TST II

THIS IS A HISTORY FILE OF CRLPG-C

THE "C" VERSION SOURCE ONLY HAS THE B TO C VERSION CHANGES
THE BIG CHANGE IS IN THE "CRLPA.MAC" FILE THAT HANDLES THE
NEW VERSION OF THE MICRO-CODE IN THE LPA-11

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.

LNKX11 V023 24-OCT-80 9:45 SEQ 0015

```
#CRLPGC.BIN/B:42000,CRLPGC.MAP=CRLPGC,CRLPX2/E


LOAD MAP

IDENT:  V1015B

TRANSFER ADDRESS: 000001
LOW LIMIT: 042000
HIGH LIMIT: 046000
**********
MODULE  LPA
SECTION ENTRY   ADDRESS SIZE
<. ABS.>        000000  000000
        DRLPX2  042000
<       >       042000  000000
**********
MODULE  DRLPX2
SECTION ENTRY   ADDRESS SIZE
<       >       042000  000000
<ABCODE>        042000  004000


 RUN-TIME:  1 SECONDS
 2K CORE USED
```

G 2

SEQ 0019

```
                .REM    [

                                CRLPAB.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
```

```
                        .TITLE  MMAST.MAC
                        .IDENT  /4.01/
                ;
                ; LPA11-K MICRO CODE
                ;
                ; CHARLES A. SAMUELSON
                ; NOVEMBER, 1977
                ;


                        .TITLE  DMAST.MAC
                        .IDENT  /4.01/
                ;
                ; LPA11-K MICRO CODE
                ;
                ; CHARLES A. SAMUELSON
                ; NOVEMBER, 1977
                ;
```

```
                                .REM    !

                              THIS IS A LIST OF TESTS DELETED FROM THIS DIAGNOSTIC.
                              THESE TEST COULD NOT BE DONE THROUGH THE LPA-11.

                              TEST THE LOW BYTE OPERATION OF CLOCK A'S STATUS REGISTER
                              TEST THE HIGH BYTE OPERATION OF A'S STATUS REGISTER
                              TEST THE LOW BYTE OPERATION OF B'S STATUS REGISTER
                              TEST THE HIGH BYTE OPERATION OF B'STATUS REGISTER
                              TEST THAT INIT CLEARS STATUS REGISTER A
                              TEST THAT INIT CLEARS BUFFER REGISTER A
                              TEST THAT INIT CLEARS STATUS REGISTER B
                              TEST THAT INIT CLEARS BUFFER REGISTER B
                              TEST THAT A'S COUNT REGISTER IS CLEARED BY INIT
                              TEST THAT CLOCK A WILL INTR. AND TO THE RIGHT VECTOR
                              TEST THAT CLOCK A WILL INTR. WHEN CPU PSW = CLK INTR LEV -1
                              TEST THAT CLOCK A WILL NOT INTR. WHEN CPU PSW = CLK INTR LEVEL
                              TEST THAT ST1 WILL CAUSE CLOCK A TO INTER.
                              TEST THAT CLOCK A OVERFLOW WILL CAUSE AN INTR.
                              TEST THAT A CLOCK A COUNTER BUFFER WILL CAUSES AN INTR.
                              TEST THAT CLOCK B WILL INTR. AND TO THE RIGHT VECTOR
                              TEST THAT CLOCK B WILL INTR. WHEN CPU PSW=CLK INTR LEV -1
                              TEST THAT CLOCK B WILL NOT INTR. WHEN CPU PSW=CLK INTR LEVEL
                              TEST THAT A CLOCKB OVERFLOW WILL CAUSE AN INTR.
                              TEST CLOCK A'S REPEATIBILITY AT 1MHZ RATE
                              TEST CLOCK A'S REPEATIBILITY AT 100KHZ RATE
                              TEST CLOCK A'S REPEATIBILITY AT 10KHZ RATE
                              TEST CLOCK A'S REPEATIBILITY AT 1KHZ RATE
                              TEST CLOCK A'S REPEATIBILITY AT 100HZ RATE
                              TEST CLOCK B'S REPEATIBILITY AT 1MHZ RATE
                              TEST CLOCK B'S REPEATIBILITY AT 100KHZ RATE
                              TEST CLOCK B'S REPEATIBILITY AT 10KHZ RATE
                              TEST CLOCK B'S REPEATIBILITY AT 1KHZ RATE
                              TEST CLOCK B'S REPEATIBILITY AT 100HZ RATE
                              TEST THAT "INIT" CLEARS B'S 100KHZ DIVIDE BY 10 CHIPS
                                      !


                              .TITLE  LPA-KW11K DIAGNOSTIC MD-11-CRLPG-C
(1)                           ;*COPYRIGHT (C) 1980
(1)                           ;*DIGITAL EQUIPMENT CORP.
(1)                           ;*MAYNARD, MASS. 01754
(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
                              .SBTTL  OPERATIONAL SWITCH SETTINGS
(1)                           ;*
(1)                           ;*      SWITCH                  USE
(1)                           ;*      ------          --------------------
(1)                           ;*        15            HALT ON ERROR
(1)                           ;*        14            LOOP ON TEST
(1)                           ;*        13            INHIBIT ERROR TYPEOUTS
```

```
                                        ;*        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  001726                          JMP     @#START         ;GO TO STARTING ADDRESS OF PROGRAM

000204  000137  002462                          JMP     @#RSTART        ;GO TO RESTART ADDRESS.
000210  000137  022450                          JMP     @#LS210         ;GO TO SPECIAL TEST #1.
000214  000137  022646                          JMP     @#LS214         ;GO TO SPECIAL TEST #2.
000220  000137  023062                          JMP     @#LS220         ;GO TO SPECIAL TEST #3.
000224  000137  023246                          JMP     @#LS224         ;GO TO SPECIAL TEST #4.
000230  000137  023454                          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
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
1221        170404                          ABASE=  170404
1222        000344                          AVECT1= 344
1223        000006                          APRIOR= 6
1265
1275                                .SBTTL  ACT11 HOOKS
 (1)
 (2)                                ;;******************************************************************
 (1)                                ;HOOKS REQUIRED BY ACT11
 (1)         000234                         $SVPC=.                 ;SAVE PC
 (1)         000046                         .=46
 (1)  000046 022414                         $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                         .=$SVPC                 ;; RESTORE PC
1276         001000                         .=1000
1277                                .SBTTL  APT PARAMETER BLOCK
 (1)
 (2)                                ;;******************************************************************
 (1)                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
 (2)                                ;;******************************************************************
 (1)         001000                         .$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 001200                 $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)
  1278
```

```
1279                                    .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     000377         $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                    $ERRPC: .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                    $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
 (1)  001170  177607  000377            $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
 (1)  001174     077                    $QUES:  .ASCII  /?/             ;;QUESTION MARK
 (1)  001175     015                    $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
 (1)  001176  000012                    $LF:    .ASCIZ  <12>            ;;LINE FEED
 (2)                                    ;;*****************************************************************
 (2)                                    .SBTTL  APT MAILBOX-ETABLE
 (2)
 (3)                                    ;;*****************************************************************
 (2)                                    .EVEN
 (2)  001200                            $MAIL:                          ;;APT MAILBOX
 (2)  001200  000000                    $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
 (2)  001202  000000                    $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
 (2)  001204  000000                    $TESTN: .WORD   ATESTN          ;;TEST NUMBER
 (2)  001206  000000                    $PASS:  .WORD   APASS           ;;PASS COUNT
```

```
(2)  001210  000000          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
(2)  001212  000000          $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
(2)  001214  000000          $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
(2)  001216  000000          $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2)  001220                  $ETABLE:                ;;APT ENVIRONMENT TABLE
(2)  001220      000         $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
(2)  001221      000         $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
(2)  001222  000000          $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2)  001224  000000          $USWR:  .WORD   AUSWR   ;;USER SWITCHES
(2)  001226  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)  001230      000         $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
(2)  001231      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)  001232  000000          $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)                          ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
(2)  001234      000         $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
(2)  001235      000         $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
(2)  001236  000000          $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2)  001240      000         $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
(2)  001241      000         $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
(2)  001242  000000          $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2)  001244      000         $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
(2)  001245      000         $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
(2)  001246  000000          $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2)  001250  000344          $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  001252  000000          $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  001254  170404          $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001256  000000          $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
(2)  001260  000000          $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
(2)  001262  000000          $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
(2)  001264  000000          $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
(2)  001266  000000          $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
(2)  001270  000000          $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
(2)  001272  000000          $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
(2)  001274  000000          $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
(2)  001276  000000          $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
(2)  001300  000000          $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
(2)  001302  000000          $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
(2)  001304  000000          $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
(2)  001306  000000          $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
(2)  001310  000000          $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
(2)  001312  000000          $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
(2)  001314  000000          $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
(2)  001316  000000          $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
(2)  001320  000000          $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
(2)  001322  000000          $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
(2)
```

```
(2)
(2)  001324                    $ETEND:
(2)
(3)
(3)  001324  170404            ASR:    170404          ;/CLOCK A STATUS REGISTER.
(3)  001326  170406            ABR:    170406          ;/CLOCK A BUFFER REGISTER.
(3)  001330  170430            ACR:    170430          ;/CLOCK A COUNT REGISTER.
(3)
(3)  001332  170432            BSR:    170432          ;/CLOCK B STATUS REGISTER.
(3)  001334  170434            BBR:    170434          ;/CLOCK B BUFFER REGISTER.
(3)  001336  170436            BCR:    170436          ;/CLOCK B COUNT REGISTER.
(3)
(3)  001340  000344            AVECT:  344             ;/CLOCK A INTR. VECTOR ADDR.
(3)  001342  000346            AVECP2: 346             ;/CLOCK A INTR. STATUS WORD.
(3)
(3)  001344  000364            BVECT:  364             ;/CLOCK B INTR. VECTOR ADDR.
(3)  001346  000366            BVECT2: 366             ;/CLOCK B INTR. STATUS WORD.
(3)
(3)  001350  000006            APRITY: 6               ;/PRIORITY LEVEL OF CLOCK A.
(3)  001352  000006            BPRITY: 6               ;/PRIORITY LEVEL OF CLOCK B.
(3)  001354  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)  001356                  $ERRTB:
  1284
  1285                          ;ITEM   1
  1286
  1287  001356  027166                 EM1                     ;CLOCK A SR FUNCTION ERROR
  1288  001360  030053                 DH1                     ;ERRPC   ASR     WAS     S/B
  1289  001362  030666                 DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
  1290  001364  031070                 DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
  1291
  1292                          :ITEM   2
  1293
  1294  001366  027221                 EM2                     ;CLOCKA SR DATA ERROR
  1295  001370  030053                 DH1                     ;ERRPC   ASR     WAS     S/B
  1296  001372  030666                 DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
  1297  001374  031070                 DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
  1298
  1299                          ;ITEM   3
  1300
  1301  001376  027250                 EM3                     ;CLOCKA BR DATA ERROR
  1302  001400  030111                 DH3                     ;ERRPC   ABR     WAS     S/B
  1303  001402  030700                 DT3                     ,$ERRPC,ABR,$BDDAT,$GDDAT
  1304  001404  031070                 DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
  1305
  1306                          ;ITEM   4
  1307
  1308  001406  027277                 EM4                     ;CLOCKA CR DATA ERROR
  1309  001410  030147                 DH4                     ;ERRPC   ACR     WAS     S/B
  1310  001412  030712                 DT4                     ;$ERRPC,ACR,$BDDAT,$GDDAT
  1311  001414  031070                 DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
  1312
  1313                          ;ITEM   5
  1314
  1315  001416  027326                 EM5                     ;CLOCK B SR DATA ERROR
  1316  001420  030205                 DH5                     ;ERRPC   BSR     WAS     S/B
  1317  001422  030724                 DT5                     ;$ERRPC,BSR,$BDDAT,$GDDAT
  1318  001424  031070                 DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
   (1)
  1319
```

```
1320                                    ;ITEM   6
1321
1322  001426  027355                            EM6                     ;CLOCK B BR DATA ERROR
1323  001430  030243                            DH6                     ;ERRPC   BBR     WAS     S/B
1324  001432  030736                            DT6                     ;$ERRPC,BBR,$BDDAT,$GDDAT
1325  001434  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1326
1327                                    ;ITEM   7
1328
1329  001436  027404                            EM7                     ;CLOCK B CR DATA ERROR
1330  001440  030301                            DH7                     ;ERRPC   BCR     WAS     S/B
1331  001442  030750                            DT7                     ;$ERRPC,BCR,$BDDAT,$GDDAT
1332  001444  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1333
1334                                    ;ITEM   10
1335
1336  001446  027433                            EM10                    ;DUAL ADDRESS ERROR
1337  001450  030337                            DH10                    ;ERROR    GOOD    BAD     GOOD    DATA READ FROM
1338                                                                    ;  PC     ADDR    ADDR    DATA    DUAL ADDRESS
1339  001452  030762                            DT10                    ;$ERRPC,$GDADR,$BDADR,$GDDAT,$BDDAT
1340  001454  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1341
1342                                    ;ITEM   11
1343
1344  001456  027460                            EM11                    ;CLOCK A COUNT ERROR
1345  001460  030147                            DH4                     ;ERRPC   ACR   WAS   S/B
1346  001462  030712                            DT4                     ;$ERRPC, ACR, $BDDAT, $GDDAT
1347  001464  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1348
1349                                    ;ITEM   12
1350
1351  001466  027507                            EM12                    ;CLOCK A COUNT FUNCTION ERROR
1352  001470  030476                            DH12                    ;ERRPC   ASR
1353  001472  030776                            DT12                    ;ERRPC, ASR
1354  001474  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1355
1356                                    ;ITEM   13
1357
1358  001476  031070                            DF0                     ;ERROR 13 DOES NOT EXSIST.
1359  001500  031070                            DF0                     ;IT WOULD BE BAD LUCK.
1360  001502  031070                            DF0
1361  001504  031070                            DF0
1362
1363                                    ;ITEM   14
1364
1365  001506  027547                            EM14                    ;CLOCK B COUNT FUNCTION ERROR
1366  001510  030515                            DH14                    ;ERRPC   BSR
1367  001512  031004                            DT14                    ;$ERRPC, BSR
1368  001514  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1369
```

```
1370                                    ;ITEM   15
1371
1372  001516  027607                            EM15                    ;CLOCK B COUNT ERROR
1373  001520  030301                            DH7                     ;ERRPC   CSR   WAS   S/B
1374  001522  030750                            DT7                     ;$ERRPC, BCR, $BDDAT, $GDDAT
1375  001524  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1376
1377                                    ;ITEM   16
1378
1379  001526  027636                            EM16                    ;CLOCK A INTERRUPT ERROR
1380  001530  030476                            DH12                    ;ERRPC   ASR
1381  001532  030776                            DT12                    ;$ERRPC, ASR
1382  001534  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1383
1384                                    ;ITEM   17
1385
1386  001536  027671                            EM17                    ;CLOCK B INTERRUPT ERROR
1387  001540  030515                            DH14                    ;ERRPC   BSR
1388  001542  031004                            DT14                    ;$ERRPC, BSR
1389  001544  031070                            DF0
1390
1391                                    ;ITEM   20
1392
1393  001546  027724                            EM20                    ;CLOCK A REPEATABILITY ERROR
1394  001550  030533                            DH20                    ;ERROR   ASR   2ND CNT 1ST CNT
1395  001552  030666                            DT1                     ;$ERRPC, ASR, $BDDAT, $GDDAT
1396  001554  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1397
1398                                    ;ITEM   21
1399
1400  001556  027460                            EM11                    ;CLOCK A COUNT ERROR
1401  001560  030147                            DH4                     ;ERROR   ASR   2ND CNT 1ST CNT
1402  001562  031012                            DT21                    ;$ERRPC, ASR, $BDDAT, $GDDAT
1403  001564  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1404
1405                                    ;ITEM   22
1406
1407  001566  027460                            EM11                    ;CLOCK A COUNT ERROR
1408  001570  030147                            DH4                     ;ERRPC   ASR   WAS   S/B
1409  001572  031024                            DT22                    ;$ERRPC, ACR, $BDDAT, $TMP0
1410  001574  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1411
1412                                    ;ITEM   23
1413
1414  001576  027763                            EM23                    ;CLOCK B REPEATABILITY ERROR
1415  001600  030575                            DH23                    ;ERROR   ASR   2NDCNT  1STCNT
1416  001602  030666                            DT1                     ;$ERRPC, ASR, $BDDAT, $GDDAT
1417  001604  031070                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1418
1419                                    ;ITEM   24
```

```
1420
1421  001606  027607                  EM15                    ;CLOCK B COUNT ERROR
1422  001610  030301                  DH7                     ;ERRPC   BCR   WAS   S/B
1423  001612  031036                  DT24                    ;$ERRPC, BCR, $GDDAT, $TMPO
1424  001614  031070                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1425
1426                          ;ITEM   25
1427
1428  001616  027607                  EM15                    ;CLOCK B COUNT ERROR
1429  001620  030301                  DH7                     ;ERRPC   BCR     WAS     S/B
1430  001622  031050                  DT25                    ;$ERRPC,BCR,$BDDAT,$TMPO
1431  001624  031070                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1432
1433                          ;ITEM   26
1434
1435  001626  030022                  EM26    ;CLOCK ADDRESSING ERRROR
1436  001630  030637                  DH26    ;ERRPC   CLOCK ADDR.
1437  001632  031062                  DT26    ;$ERRPC,$TMPO
1438  001634  031070                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
1439                                    ;
 (1)                                    ;ADDRESS OF KMC-11 OF LPA-11    THE ADDR FOR KMADO 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)  001636                    LPCI:
 (1)  001636  170460            KMAD0:  .WORD   170460          ;BASE KMC ADDR. MAY BE PATCHED BY USER.
 (1)
 (1)  001640                    LPMR:
 (1)  001640  170461            KMAD1:  .WORD   170460+1                ;>DO NOT        <;KMC-CSR ADDR
 (1)  001642                    LPCO:
 (1)  001642  170462            KMAD2:  .WORD   170460+2                ;>PATCH         <;
 (1)  001644                    LPSO:
 (1)  001644  170463            KMAD3:  .WORD   170460+3                ;>THIS AREA     <
 (1)  001646                    LPADL:
 (1)  001646  170464            KMAD4:  .WORD   170460+4                ;
 (1)  001650                    LPADH:
 (1)  001650  170465            KMAD5:  .WORD   170460+5                ;>DO NOT        <
 (1)  001652                    LPMS1:
 (1)  001652  170466            KMAD6:  .WORD   170460+6                ;>PATCH         <
 (1)  001654                    LPMS2:
 (1)  001654  170467            KMAD7:  .WORD   170460+7                ;>THIS AREA     <
 (1)
 (1)  001656  000344            VECTOR: .WORD   AVECT1&777      ;BASE VECTOR OF KMC
 (1)  001660  000350            VECTPS: .WORD   4+AVECT1&777    ;VECOTR ADDR.+2
 (1)
 (1)  001662  000005            VERSN:  .WORD   5               ;CURRENT VERSION NUMBER OF MICROCODE.
 (1)
 (1)  001664  000000            .DVLS:  .WORD   0               ;/DEVICE LIST OF I/O ADDR. DEFINED
 (1)  001666  000020                    .BLKW   16.             ;/BY INIT.
 (1)
```

```
                                H 3
1440                    .SBTTL  PROGRAM START
```

```
                                                                    1  3
 1443
 1444  001726                          START:
  (1)                                  .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)  001726  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  (1)  001732  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  (1)  001734  022706  001140                  CMP     #SWR,R6 ;;DONE?
  (1)  001740  001374                          BNE     .-6             ;;LOOP BACK IF NO
  (1)  001742  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  (1)                                  ;;INITIALIZE A FEW VECTORS
  (1)  001746  012737  025062  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)  001754  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  (1)  001762  012737  024302  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  001770  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  001776  012737  027102  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  002004  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  002012  012737  026662  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)  002020  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  (1)  002026  013737  022362  022354          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
  (1)  002034  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  (1)  002040  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
  (1)  002046  012737  002046  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (1)  002054  012737  002054  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
  (2)                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  (2)                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
  (2)  002062  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  (2)  002066  012737  002122  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  (2)  002074  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  002102  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  (2)  002110  022777  177777  177022          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  (2)  002116  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  002120  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
  (2)  002122  012716  002130          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  (2)  002126  000002                          RTI
  (2)  002130  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  (2)  002136  012737  000174  001142          MOV     #DISPREG,DISPLAY
  (2)  002144  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
  (1)
  (2)  002150  005037  001206                  CLR     $PASS           ;;CLEAR PASS COUNT
  (2)  002154  132737  000200  001221          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
  (2)  002162  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
  (2)  002164  012737  001222  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
  (2)  002172                          67$:
 1445  002172  122737  000001  001134          CMPB    #1,$AUTOB       ;IF RUNNING XXDP ETC.
 1446  002200  001426                          BEQ     10$
 1447
 1448  002202  104401  002210                  TYPE    ,69$            ;;TYPE ASCIZ STRING
  (1)  002206  000423                          BR      68$             ;;GET OVER THE ASCIZ
  (1)                                  ;;69$:  .ASCIZ  <15><12><12>#CRLPGC   LPA/KW11-K  DIAGNOSTIC#<15><12>
  (1)  002256                          68$:
 1449
 1450  002256  013737  001254  001324  10$:    MOV     $BASE,ASR
 1451  002264  012737  000001  001210          MOV     #1,$DEVCT
 1452  002272  005037  001206                  CLR     $PASS
 1453
```

```
1454
 (1)                                    ;
 (1)                                    ;THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
 (1)                                    ;
 (1)
 (1)  002276  010046                    MOV     R0,-(SP)
 (1)  002300  010146                    MOV     R1,-(SP)
 (1)  002302  013700  001636            MOV     KMAD0,R0        ;GET KMC-11 ADDRESS.
 (1)  002306  012701  001640            MOV     #KMAD1,R1       ;GET ADDR. OF ADDR. LIST.
 (1)
 (1)  002312  005200            70$:    INC     R0              ;UPDATE ADDR.
 (1)  002314  010021                    MOV     R0,(1)+         ;WRITE ADDR.
 (1)  002316  020127  001656            CMP     R1,#KMAD7+2     ;DONE ALL ADDRESSES?
 (1)  002322  001373                    BNE     70$             ;NO - DO NEXT ADDR.
 (1)  002324  005037  001664            CLR     .DVLS           ;CLR ADDR. LIST.
 (1)  002330  012601                    MOV     (SP)+,R1
 (1)  002332  012600                    MOV     (SP)+,R0
1455
1456  002334                    LOOP:
1457  002334  005000                    CLR     R0
1458  002336  005200            1$:     INC     R0              ;DELAY SOME TIME SO THAT FIRST RESET
1459  002340  001376                    BNE     1$              ;INSTR. WON'T CLOBBER TYPEOUT.
1460  002342  013700  001324            MOV     ASR,R0          ;NOW WE'RE GONNA FIX
1461  002346  062700  000002            ADD     #2,R0           ;ALL CLOCK ADDRESSES BASED ON ASR.
1462  002352  010037  001326            MOV     R0,ABR
1463  002356  062700  000022            ADD     #22,R0
1464  002362  010037  001330            MOV     R0,ACR
1465  002366  062700  000002            ADD     #2,R0
1466  002372  010037  001332            MOV     R0,BSR
1467  002376  062700  000002            ADD     #2,R0
1468  002402  010037  001334            MOV     R0,BBR
1469  002406  062700  000002            ADD     #2,R0
1470  002412  010037  001336            MOV     R0,BCR
1471
1472  002416  013700  001340            MOV     AVECT,R0        ;NOW FIX VECTOR ADDRESSES
1473  002422  062700  000002            ADD     #2,R0           ;BASED ON AVECT.
1474  002426  010037  001342            MOV     R0,AVECP2
1475  002432  062700  000016            ADD     #16,R0
1476  002436  010037  001344            MOV     R0,BVECT
1477  002442  062700  000002            ADD     #2,R0
1478  002446  010037  001346            MOV     R0,BVECT2
1479
1480  002452  013737  001350  001352    MOV     APRITY,BPRITY   ;FIX CLK B'S PRIORITY BASED ON A'S.
1481  002460  000402                    BR      SSTART
1482  002462  005037  001664    RSTART: CLR     .DVLS
1483  002466  012706  001100    SSTART: MOV     #STACK,SP
1484  002472  012746  000340            MOV     #340,-(SP)      ;SET PROCESSOR PRIORITY TO 7.
1485  002476  012746  002504            MOV     #1$,-(SP)
1486  002502  000002                    RTI
1487  002504                    1$:
1488
1489                            .SBTTL  *
1490                            .SBTTL  * PHASE 1 CLOCKS A+B BASIC LOGIC TESTS.
1491                            .SBTTL  *
```

```
1518
 (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 Z'' H AND WORK BACK
 (5)                                    ;*
 (4)                                    ;*****************************************************************
 (3)  002504  000240                    TST1:   NOP
 (1)  002506  112737  000001  001102    1$:     MOVB    #1,$TSTNM
 (1)  002514  112737  000001  001204            MOVB    #1,$TESTN
 (1)
 (1)
 (2)
 (2)                                    ;*      MOV     @ASR,$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)
1541
1542                                    ;;*****************************************************************
 (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 1N 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)  002602  000004                    TST2:   SCOPE
1543
1544                                                                ;WRITE INTO PRESET BUFFER.
1545                                                                ;SET $GDDAT FOR ERROR TYPEOUT.
1546                                                                ;READ THE PRESET BUFFER.
1547  002604  012737  001416  001124              MOV     #1416,$GDDAT
1548
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
1549
 (1)                                    ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
1550  002632  005737  001126                      TST     $BDDAT
1551  002636  001001                              BNE     1$              ;IF ANY DATA WAS READ BACK, WE WILL
1552                                                                      ;BR PAST THE ERROR CALL.
1553
1554
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

1555  002640  104003                              ERROR   3               ;UNABLE TO LOAD AND READ BACK
1556                                                                      ;FROM THE BUFFER REGISTER CLOCK A.
1557
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

1558  002642                           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'
                        ;*
                        ;*
                        ;;*****************************************************************
002642  000004         TST3:   SCOPE

002644  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.
002670  005737  001126         TST     $BDDAT
002674  001401                 BEQ     1$              ;SHOULD BE CLEAR - IF CLEAR - NEXT TEST.

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

002676  104003                 ERROR   3               ;CLEAR INTR. FAILED TO CLEAR CLOCK A'S
                                                        ;BUFFER REGISTER.

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

002700                  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" L
                        ;*       "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
                        ;*
                        ;*
                        ;;*****************************************************************
```

```
  (2)  002700  000004                    TST4:   SCOPE
1605
1606  002702  012737  001416  001124             MOV     #1416,$GDDAT    ;LOAD $GDDAT WITH S/B.
1607                                                                      ;LOAD STATUS REGISTER.
1608                                                                      ;READ BACK STATUS REGISTER
1609
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
1610
  (1)                                   ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
1611  002730  005737  001126                    TST     $BDDAT
1612  002734  001001                            BNE     1$              ;IF ANY BITS RETURNED - NO ERROR.
1613
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

1614  002736  104002                            ERROR   2               ;ERROR-UNABLE TO LOAD AND READ BACK
1615                                                                    ;STATUS REGISTER OF CLOCK A.
1616
1617
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

1618  002740                                 1$:                             ;CLEAR CLOCKA'S STATUS REG.
1619  002740  005037  001124                    CLR     $GDDAT
1620
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
1621
1629
1630                                    ;;**********************************************************************
  (3)                                   ;*TEST 5        *TEST THAT CLOCK B'S STATUS REGISTER CAN BE WROTE/READ
  (4)                                   ;*
  (4)                                   ;*NEW SIGNALS: ['BA03" H + "BA02" L + 'BA01" L]='LD STATB" L+'RD STAT B" L
  (5)                                   ;*
  (5)                                   ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEV ENABLE (1)" 2 OCCURANCES PER PASS
  (5)                                   ;*
  (4)                                   ;*
  (4)
  (3)                                   ;;**********************************************************************
  (2)  002754  000004                    TST5:   SCOPE
1631
1632  002756  012737  001016  001124             MOV     #1016,$GDDAT    ;USE 1016 AS PATTERN, PUT IN $GDDAT.
1633                                                                      ;LOAD B'S STAT REG.
1634                                                                      ;READ BACK THE STATUS REG.
1635
  (1)                                   ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
1636
  (1)                                   ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
1637  003004  005737  001126                    TST     $BDDAT
1638  003010  001001                            BNE     1$              ;IF ANY BITS CAME BACK, SUBTEST OK.
1639
1640
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

1641  003012  104005                            ERROR   5               ;ERROR-COULD NOT WRITE/READ BACK
1642                                                                    ;CLOCK B'S STATUS REGISTER.
1643
1644
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
1645  003014                        1$:
1646
1647
1654
1655                                ;;*****************************************************************
 (3)                                ;*TEST 6        *TEST THAT CLOCK B'S BUFFER REGISTER CAN BE WROTE/READ
 (4)                                ;*
 (4)                                ;*NEW SIGNALS: ["BA03" H + "BA02" L + "BA01" H]="LD BUFF B" L*"RD BUFF B" L
 (5)                                ;*
 (5)                                ;* PROBABLE SYNC POINT FOR THIS TEST:: "DEV ENABLE (1)" 2 OCCURANCES PER PASS
 (5)                                ;*
 (4)                                ;*
 (3)                                ;*****************************************************************
 (2)  003014  000004                TST6:   SCOPE
1656
1657  003016  012737  000370  001124        MOV     #370,$GDDAT     ;USE PATTERN "370", PUT IN $GDDAT.
1658                                                                  ;WRITE INTO CLOCK B'S BUFFER REGISTER.
1659                                                                  ;READ IT BACK.
1660
 (1)                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
1661
 (1)                                ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
1662  003044  005737  001126               TST     $BDDAT
1663  003050  001001                       BNE     1$              ;IF ANY BITS COME BACK-SUBTEST OK.
1664
1665
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

1666  003052  104006                       ERROR   6               ;ERROR-FAILED TO WRITE/READ CLOCKB'S
1667                                                                  ;BUFFER REGISTER.
1668
1669
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

1670  003054                        1$:
1671
1679
```

```
                        ;/#
                        ;;*****************************************************************
                        ;*TEST 7        *TEST THAT CLOCK A STATUS REGISTER BIT 15 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
                        ;*
                        ;*
                        ;;*****************************************************************
003054  000004        TST7:   SCOPE
                                                ;/CLEAR THE STATUS REGISTER.
                                                ;/SET BIT 15.
                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                 ;/READ THE STATUS REGISTER.
003056  012737  100000  001124          MOV     #BIT15,$GDDAT
                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                        ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
003104  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 15 AND ONLY BIT 15 SET?
003112  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
003114  104002                         ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
                                                ;/BIT 15 FAILED TO BIT SET.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
003116  000416                         BR      2$              ;/BR TO END SUBTEST.
003120                          1$:                             ;/TRY CLEARING BIT 15.
003120  005037  001124                 CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                ;/NOW READ IT BACK.
                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                        ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
003144  005737  001126                 TST     $BDDAT
003150  001401                         BEQ     2$              ;/IF ZERO-NO ERROR!
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
003152  104002                         ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
                                                ;/BIT 15 FAILED TO CLEAR.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
003154                          2$:
```

```
(2)                                     ;/#
(6)                                     ;;*****************************************************************
(5)                                     ;*TEST 10       *TEST THAT CLOCK A STATUS REGISTER BIT 14 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)  003154  000004                     TST10:  SCOPE
(2)                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                             ;/SET BIT 14.
(2)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                              ;/READ THE STATUS REGISTER.
(2)  003156  012737  040000  001124             MOV     #BIT14,$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)  003204  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 14 AND ONLY BIT 14 SET?
(2)  003212  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003214  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                     ;/BIT 14 FAILED TO BIT SET.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003216  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  003220                             1$:                             ;/TRY CLEARING BIT 14.
(2)  003220  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)  003244  005737  001126                     TST     $BDDAT
(2)  003250  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003252  104002                             ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                     ;/BIT 14 FAILED TO CLEAR.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003254                             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)  003254  000004                     TST11:  SCOPE
(2)                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                             ;/SET BIT 13.
(2)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                              ;/READ THE STATUS REGISTER.
(2)  003256  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)  003304  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 13 AND ONLY BIT 13 SET?
(2)  003312  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003314  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                             ;/BIT 13 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003316  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  003320                         1$:                                 ;/TRY CLEARING BIT 13.
(2)  003320  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)  003344  005737  001126                     TST     $BDDAT
(2)  003350  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003352  104002                             ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                             ;/BIT 13 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003354                         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)  003354  000004                     TST12:  SCOPE
(2)                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                             ;/SET BIT 9.
(2)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                              ;/READ THE STATUS REGISTER.
(2)  003356  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)  003404  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 9 AND ONLY BIT 9 SET?
(2)  003412  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003414  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                     ;/BIT 9 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003416  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  003420                             1$:                             ;/TRY CLEARING BIT 9.
(2)  003420  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)  003444  005737  001126                     TST     $BDDAT
(2)  003450  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003452  104002                             ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                     ;/BIT 9 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003454                             2$:
```

```
                                        ;/#
                                        ;;*****************************************************************
                                        ;*TEST 13       *TEST THAT CLOCK A STATUS REGISTER BIT 8 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
                                        ;*
                                        ;*
                                        ;;*****************************************************************
003454  000004                        TST13:  SCOPE
                                                                ;/CLEAR THE STATUS REGISTER.
                                                                ;/SET BIT 8.
                                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                                 ;/READ THE STATUS REGISTER.
003456  012737  000400  001124                MOV     #BIT8,$GDDAT

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

                                        ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
003504  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;/DID BIT 8 AND ONLY BIT 8 SET?
003512  001402                               BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003514  104002                               ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
                                                                ;/BIT 8 FAILED TO BIT SET.

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

003516  000416                               BR      2$              ;/BR TO END SUBTEST.
003520                          1$:                                 ;/TRY CLEARING BIT 8.
003520  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.
003544  005737  001126                       TST     $BDDAT
003550  001401                               BEQ     2$              ;/IF ZERO-NO ERROR!

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

003552  104002                               ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
                                                                ;/BIT 8 FAILED TO CLEAR.

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

003554                          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)  003554  000004                     TST14:  SCOPE
(2)                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                             ;/SET BIT 7.
(2)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                              ;/READ THE STATUS REGISTER.
(2)  003556  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)  003604  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 7 AND ONLY BIT 7 SET?
(2)  003612  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003614  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                     ;/BIT 7 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003616  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  003620                             1$:                             ;/TRY CLEARING BIT 7.
(2)  003620  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)  003644  005737  001126                     TST     $BDDAT
(2)  003650  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003652  104002                             ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                     ;/BIT 7 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003654                             2$:
```

```
                                        ;/#
                                ;;*****************************************************************
                                ;*TEST 15       *TEST THAT CLOCK A STATUS REGISTER BIT 6 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
                                ;*
                                ;*
                                ;;*****************************************************************
003654  000004                 TST15:  SCOPE
                                                        ;/CLEAR THE STATUS REGISTER.
                                                        ;/SET BIT 6.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE STATUS REGISTER.
003656  012737  000100  001124          MOV     #BIT6,$GDDAT
                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
003704  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
003712  001402                        BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003714  104002                        ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
                                                        ;/BIT 6 FAILED TO BIT SET.

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

003716  000416                        BR      2$              ;/BR TO END SUBTEST.
003720                          1$:                             ;/TRY CLEARING BIT 6.
003720  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.
003744  005737  001126                TST     $BDDAT
003750  001401                        BEQ     2$              ;/IF ZERO-NO ERROR!

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

003752  104002                        ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
                                                        ;/BIT 6 FAILED TO CLEAR.

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

003754                          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)  003754  000004                     TST16:  SCOPE
(2)                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                             ;/SET BIT 5.
(2)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                              ;/READ THE STATUS REGISTER.
(2)  003756  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)  004004  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
(2)  004012  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004014  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                     ;/BIT 5 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004016  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  004020                             1$:                             ;/TRY CLEARING BIT 5.
(2)  004020  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)  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 5 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004054                             2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 17       *TEST THAT CLOCK A STATUS REGISTER BIT 3 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
                                ;*
                                ;*
                                ;;**********************************************************************
(4)  004054  000004             TST17:  SCOPE
(2)                                                     ;/CLEAR THE STATUS REGISTER.
(2)                                                     ;/SET BIT 3.
(2)                                                     ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                      ;/READ THE STATUS REGISTER.
(2)  004056  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)  004104  023737  001124  001126   CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
(2)  004112  001402                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004114  104002                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                     ;/BIT 3 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004116  000416                  BR      2$              ;/BR TO END SUBTEST.
(2)  004120                     1$:                         ;/TRY CLEARING BIT 3.
(2)  004120  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)  004144  005737  001126          TST     $BDDAT
(2)  004150  001401                  BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004152  104002                  ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                     ;/BIT 3 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004154                     2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 20        *TEST THAT CLOCK A STATUS REGISTER BIT 2 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
                                ;*
                                ;*
                                ;;**********************************************************************
(4) 004154  000004              TST20:  SCOPE
(2)                                                     ;/CLEAR THE STATUS REGISTER.
(2)                                                     ;/SET BIT 2.
(2)                                                     ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                      ;/READ THE STATUS REGISTER.
(2) 004156  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) 004204  023737  001124  001126           CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
(2) 004212  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2) 004214  104002                          ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                             ;/BIT 2 FAILED TO BIT SET.
(3)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2) 004216  000416                          BR      2$              ;/BR TO END SUBTEST.
(2) 004220                      1$:                                 ;/TRY CLEARING BIT 2.
(2) 004220  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 $ DDAT.
(2) 004244  005737  001126                  TST     $BDDAT
(2) 004250  001401                          BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2) 004252  104002                          ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                             ;/BIT 2 FAILED TO CLEAR.
(3)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2) 004254                      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
                                ;*
                                ;*
                                ;;**********************************************************************
004254  000004                TST21:  SCOPE
                                                        ;/CLEAR THE STATUS REGISTER.
                                                        ;/SET BIT 1.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE STATUS REGISTER.
004256  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.
004304  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
004312  001402                        BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

004314  104002                        ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
                                                        ;/BIT 1 FAILED TO BIT SET.

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

004316  000416                        BR      2$              ;/BR TO FND SUBTEST.
004320                          1$:                           ;/TRY CLEARING BIT 1.
004320  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.
004344  005737  001126                TST     $BDDAT
004350  001401                        BEQ     2$              ;/IF ZERO-NO ERROR!

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

004352  104002                        ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
                                                        ;/BIT 1 FAILED TO CLEAR.

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

004354                          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)  004354  000004                         TST22:  SCOPE
(2)                                                                 ;/CLEAR THE STATUS REGISTER.
(2)                                                                 ;/SET BIT 0.
(2)                                                                 ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                  ;/READ THE STATUS REGISTER.
(2)  004356  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)  004404  023737  001124  001126                 CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
(2)  004412  001402                                 BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004414  104002                                 ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                 ;/BIT 0 FAILED TO BIT SET.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004416  000416                                 BR      2$              ;/BR TO END SUBTEST.
(2)  004420                                 1$:                             ;/TRY CLEARING BIT 0.
(2)  004420  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)  004444  005737  001126                         TST     $BDDAT
(2)  004450  001401                                 BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004452  104002                                 ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                 ;/BIT 0 FAILED TO CLEAR.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004454                                 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
                                ;*
                                ;*
                                ;;**********************************************************
004454  000004                 TST23:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 0.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
004456  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.
004504  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
004512  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
004514  104003                         ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                        ;/BIT 0 FAILED TO BIT SET.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
004516  000416                         BR      2$              ;/BR TO END SUBTEST.
004520                          1$:                            ;/TRY CLEARING BIT 0.
004520  005037  001124                 CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                        ;/NOW READ IT BACK.
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
004544  005737  001126                 TST     $BDDAT
004550  001401                         BEQ     2$              ;/IF ZERO-NO ERROR!
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
004552  104003                         ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                        ;/BIT 0 FAILED TO CLEAR.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
004554                          2$:
```

```
(3)                                     ;/#
(7)                                     ;;*****************************************************************
(6)                                     ;*TEST 24       *TEST THAT CLOCK A BUFFER REGISTER BIT 1 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)  004554  000004                     TST24:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 1.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  004556  012737  000002  001124             MOV     #BIT1,$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)  004604  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
(3)  004612  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  004614  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 1 FAILED TO BIT SET.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  004616  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  004620                             1$:                             ;/TRY CLEARING BIT 1.
(3)  004620  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)  004644  005737  001126                     TST     $BDDAT
(3)  004650  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  004652  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 1 FAILED TO CLEAR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  004654                             2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 25       *TEST THAT CLOCK A BUFFER REGISTER BIT 2 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
                                ;*
                                ;*
                                ;;**********************************************************************
004654  000004                 TST25:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 2.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
004656  012737  000004  001124          MOV     #BIT2,$GDDAT
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
004704  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
004712  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
004714  104003                          ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                        ;/BIT 2 FAILED TO BIT SET.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
004716  000416                          BR      2$              ;/BR TO END SUBTEST.
004720                          1$:                             ;/TRY CLEARING BIT 2.
004720  005037  001124                  CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                        ;/NOW READ IT BACK.
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
004744  005737  001126                  TST     $BDDAT
004750  001401                          BEQ     2$              ;/IF ZERO-NO ERROR!
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> FRROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
004752  104003                          ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                        ;/BIT 2 FAILED TO CLEAR.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
004754                          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
                                ;*
                                ;*
                                ;;**********************************************************************
004754  000004                TST26:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 3.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
004756  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.
005004  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
005012  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
005014  104003                          ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                                ;/BIT 3 FAILED TO BIT SET.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
005016  000416                          BR      2$              ;/BR TO END SUBTEST.
005020                          1$:                             ;/TRY CLEARING BIT 3.
005020  005037  001124                  CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
005044  005737  001126                  TST     $BDDAT
005050  001401                          BEQ     2$              ;/IF ZERO-NO ERROR!
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
005052  104003                          ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                                ;/BIT 3 FAILED TO CLEAR.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
005054                          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)  005054  000004                     TST27:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 4.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  005056  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)  005104  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
(3)  005112  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005114  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 4 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005116  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  005120                             1$:                             ;/TRY CLEARING BIT 4.
(3)  005120  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)  005144  005737  001126                     TST     $BDDAT
(3)  005150  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005152  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 4 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005154                             2$:
```

```
(3)                                     ;/#
(7)                                     ;;*****************************************************************
(6)                                     ;*TEST 30       *TEST THAT CLOCK A BUFFER REGISTER BIT 5 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)  005154  000004                     TST30:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 5.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  005156  012737  000040  001124             MOV     #BIT5,$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)  005204  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
(3)  005212  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005214  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 5 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

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

(3)  005216  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  005220                             1$:                             ;/TRY CLEARING BIT 5.
(3)  005220  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)  005244  005737  001126                     TST     $BDDAT
(3)  005250  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005252  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 5 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005254                             2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 31        *TEST THAT CLOCK A BUFFER REGISTER BIT 6 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
                                ;*
                                ;*
                                ;;**********************************************************************
005254  000004                TST31:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 6.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ .HE BUFFER REGISTER.
005256  012737  000100  001124          MOV     #BIT6,$GDDAT
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
005304  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
005312  001402                        BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

005314  104003                        ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                        ;/BIT 6 FAILED TO BIT SET.

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

005316  000416                        BR      2$              ;/BR TO END SUBTEST.
005320                          1$:
005320  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.
005344  005737  001126                TST     $BDDAT
005350  001401                        BEQ     2$              ;/IF ZERO-NO ERROR!

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

005352  104003                        ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                        ;/BIT 6 FAILED TO CLEAR.

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

005354                          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)  005354  000004                     TST32:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 7.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  005356  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)  005404  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 7 AND ONLY BIT 7 SET?
(3)  005412  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005414  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                             ;/BIT 7 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005416  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  005420                             1$:                             ;/TRY CLEARING BIT 7.
(3)  005420  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)  005444  005737  001126                     TST     $BDDAT
(3)  005450  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005452  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                             ;/BIT 7 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005454                             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)  005454  000004                     TST33:  SCOPE
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 8.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  005456  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)  005504  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 8 AND ONLY BIT 8 SET?
(3)  005512  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005514  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 8 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005516  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  005520                             1$:                             ;/TRY CLEARING BIT 8.
(3)  005520  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)  005544  005737  001126                     TST     $BDDAT
(3)  005550  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005552  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 8 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005554                             2$:
```

```
                                        ;/#
                                ;;*****************************************************************
                                ;*TEST 34       *TEST THAT CLOCK A BUFFER REGISTER BIT 9 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
                                ;*
                                ;*
                                ;;*****************************************************************
005554  000004                 TST34:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 9.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
005556  012737  001000  001124          MOV     #BIT9,$GDDAT

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

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
005604  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 9 AND ONLY BIT 9 SET?
005612  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

005614  104003                          ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                                ;/BIT 9 FAILED TO BIT SET.

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

005616  000416                          BR      2$              ;/BR TO END SUBTEST.
005620                          1$:                             ;/TRY CLEARING BIT 9.
005620  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.
005644  005737  001126                  TST     $BDDAT
005650  001401                          BEQ     2$              ;/IF ZERO-NO ERROR!

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

005652  104003                          ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                                ;/BIT 9 FAILED TO CLEAR.

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

005654                          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)  005654  000004                     TST35:  SCOPE
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 10.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  005656  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)  005704  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 10 AND ONLY BIT 10 SET?
(3)  005712  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005714  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 10 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005716  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  005720                          1$:                                ;/TRY CLEARING BIT 10.
(3)  005720  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)  005744  005737  001126                     TST     $BDDAT
(3)  005750  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005752  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 10 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005754                          2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 36        *TEST THAT CLOCK A BUFFER REGISTER BIT 11 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
                                ;*
                                ;*
                                ;**********************************************************************
(5) 005754  000004              TST36:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 11.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
(3) 005756  012737  004000  001124      MOV     #BIT11,$GDDAT
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3) 006004  023737  001124  001126      CMP     $GDDAT,$BDDAT   ;/DID BIT 11 AND ONLY BIT 11 SET?
(3) 006012  001402                      BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 006014  104003                      ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                             ;/BIT 11 FAILED TO BIT SET.
(4)
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 006016  000416                      BR      2$              ;/BR TO END SUBTEST.
(3) 006020                      1$:                             ;/TRY CLEARING BIT 11.
(3) 006020  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) 006044  005737  001126              TST     $BDDAT
(3) 006050  001401                      BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 006052  104003                      ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                             ;/BIT 11 FAILED TO CLEAR.
(4)
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 006054                      2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 37        *TEST THAT CLOCK A BUFFER REGISTER BIT 12 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
                                ;*
                                ;*
                                ;;**********************************************************************
006054  000004                TST37:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 12.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
006056  012737  010000  001124          MOV     #BIT12,$GDDAT
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
006104  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 12 AND ONLY BIT 12 SET?
006112  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

006114  104003                        ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                        ;/BIT 12 FAILED TO BIT SET.

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

006116  000416                        BR      2$              ;/BR TO END SUBTEST.
006120                          1$:                           ;/TRY CLEARING BIT 12.
006120  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.
006144  005737  001126                 TST     $BDDAT
006150  001401                         BEQ     2$              ;/IF ZERO-NO ERROR!

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

006152  104003                        ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                        ;/BIT 12 FAILED TO CLEAR.

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

006154                          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)  006154  000004            .        TST40:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 13.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  006156  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)  006204  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 13 AND ONLY BIT 13 SET?
(3)  006212  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006214  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 13 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006216  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006220                             1$:                             ;/TRY CLEARING BIT 13.
(3)  006220  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)  006244  005737  001126                     TST     $BDDAT
(3)  006250  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006252  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 13 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006254                             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)  006254  000004                     TST41:  SCOPE
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 14.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  006256  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)  006304  023737  001124  001126     .       CMP     $GDDAT,$BDDAT   ;/DID BIT 14 AND ONLY BIT 14 SET?
(3)  006312  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006314  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 14 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006316  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006320                             1$:                             ;/TRY CLEARING BIT 14.
(3)  006320  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)  006344  005737  001126                     TST     $BDDAT
(3)  006350  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006352  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 14 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006354                             2$:
```

```
(3)                                     ;/#
(7)                                     ;;*******************************************************************
(6)                                     ;*TEST 42       *TEST THAT CLOCK A BUFFER REGISTER BIT 15 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)  006354  000004                     TST42:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 15.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  006356  012737  100000  001124             MOV     #BIT15,$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)  006404  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 15 AND ONLY BIT 15 SET?
(3)  006412  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006414  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                             ;/BIT 15 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

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

(3)  006416  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006420                             1$:                             ;/TRY CLEARING BIT 15.
(3)  006420  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)  006444  005737  001126                     TST     $BDDAT
(3)  006450  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006452  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                             ;/BIT 15 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006454                             2$:
(1)
(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)  006454  000004                     TST43:  SCOPE
(3)                                                             ;/CLEAR THE STATUS REGISTER.
(3)                                                             ;/SET BIT 11.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE STATUS REGISTER.
(3)  006456  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)  006504  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 11 AND ONLY BIT 11 SET?
(3)  006512  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006514  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                             ;/BIT 11 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006516  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006520                             1$:                             ;/TRY CLEARING BIT 11.
(3)  006520  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)  006544  005737  001126                     TST     $BDDAT
(3)  006550  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006552  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                             ;/BIT 11 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006554                             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)  006554  000004                     TST44:  SCOPE
(3)                                                                 ;/CLEAR THE STATUS REGISTER.
(3)                                                                 ;/SET BIT 7.
(3)                                                                 ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                  ;/READ THE STATUS REGISTER.
(3)  006556  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)  006604  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 7 AND ONLY BIT 7 SET?
(3)  006612  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006614  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 7 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006616  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006620                             1$:                             ;/TRY CLEARING BIT 7.
(3)  006620  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)  006644  005737  001126                     TST     $BDDAT
(3)  006650  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006652  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 7 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006654                             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 REG'STER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' ? OCCURANCES PER PAS.
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;*****************************************************************
(5)  006654  000004                     TST45:  SCOPE
(3)                                                             ;/CLEAR THE STATUS REGISTER.
(3)                                                             ;/SET BIT 6.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE STATUS REGISTER.
(3)  006656  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)  006704  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
(3)  006712  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006714  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                             ;/BIT 6 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006716  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006720                             1$:                             ;/TRY CLEARING BIT 6.
(3)  006720  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)  006744  005737  001126                     TST     $BDDAT
(3)  006750  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006752  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                             ;/BIT 6 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006754                             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)  006754  000004                     TST46:  SCOPE
(3)                                                                     ;/CLEAR THE STATUS REGISTER.
(3)                                                                     ;/SET BIT 5.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE STATUS REGISTER.
(3)  006756  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)  007004  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
(3)  007012  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007014  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 5 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007016  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007020                          1$:                                ;/TRY CLEARING BIT 5.
(3)  007020  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)  007044  005737  001126                     TST     $BDDAT
(3)  007050  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007052  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 5 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007054                          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)  007054  000004                     TST47:  SCOPE
(3)                                                                     ;/CLEAR THE STATUS REGISTER.
(3)                                                                     ;/SET BIT 4.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE STATUS REGISTER.
(3)  007056  012737  000020  001124             MOV     #BIT4,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSP
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007104  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
(3)  007112  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             .;:$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007114  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTEF
(3)                                                                     ;/BIT 4 FAILED TO BIT SET.
(4)
             ;;:$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007116  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007120                             1$:                             ;/TRY CLEARING BIT 4.
(3)  007120  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)  007144  005737  001126                     TST     $BDDAT
(3)  007150  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;:$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007152  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 4 FAILED TO CLEAR.
(4)
             ;;:$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007154                             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)  007154  000004                     TST50:  SCOPE
(3)                                                                 ;/CLEAR THE STATUS REGISTER.
(3)                                                                 ;/SET BIT 3.
(3)                                                                 ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                  ;/READ THE STATUS REGISTER.
(3)  007156  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)  007204  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
(3)  007212  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007214  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 3 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007216  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007220                             1$:                             ;/TRY CLEARING BIT 3.
(3)  007220  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)  007244  005737  001126                     TST     $BDDAT
(3)  007250  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007252  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 3 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007254                             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)  007254  000004                     TST51:  SCOPE
(3)                                                             ;/CLEAR THE STATUS REGISTER.
(3)                                                             ;/SET BIT 2.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE STATUS REGISTER.
(3)  007256  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)  007304  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
(3)  007312  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007314  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 2 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007316  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007320                             1$:                             ;/TRY CLEARING BIT 2.
(3)  007320  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 DEVIC: REG BSR,PUT DATA IN $BDDAT.
(3)  007344  005737  001126                     TST     $BDDAT
(3)  007350  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007352  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 2 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007354                             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)  007354  000004                     TST52:  SCOPE
(3)                                                                     ;/CLEAR THE STATUS REGISTER.
(3)                                                                     ;/SET BIT 1.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE STATUS REGISTER.
(3)  007356  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)  007404  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
(3)  007412  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007414  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 1 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007416  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007420                             1$:                             ;/TRY CLEARING BIT 1.
(3)  007420  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)  007444  005737  001126                     TST     $BDDAT
(3)  007450  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007452  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 1 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007454                             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)  007454  000004                     TST53:  SCOPE
(3)                                                             ;/CLEAR THE STATUS REGISTER.
(3)                                                             ;/SET BIT 0.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE STATUS REGISTER.
(3)  007456  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)  007504  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
(3)  007512  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007514  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                             ;/BIT 0 FAILED TO BIT SET.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007516  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007520                          1$:                                ;/TRY CLEARING BIT 0.
(3)  007520  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)  007544  005737  001126                     TST     $BDDAT
(3)  007550  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007552  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                             ;/BIT 0 FAILED TO CLEAR.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007554                          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) 007554  000004              TST54:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 0.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
(3) 007556  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) 007604  023737  001124  001126      CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
(3) 007612  001402                      BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 007614  104006                      ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                             ;/BIT 0 FAILED TO BIT SET.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 007616  000416                      BR      2$              ;/BR TO END SUBTEST.
(3) 007620                      1$:                             ;/TRY CLEARING BIT 0.
(3) 007620  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) 007644  005737  001126              TST     $BDDAT
(3) 007650  001401                      BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 007652  104006                      ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                             ;/BIT 0 FAILED TO CLEAR.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 007654                      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
                                ;*
                                ;*
                                ;;**********:***********************************************************
007654  000004                 TST55:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 1.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
007656  012737  000002  001124          MOV     #BIT1,$GDDAT
                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
                                ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
007704  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
007712  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
007714  104006                          ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
                                                        ;/BIT 1 FAILED TO BIT SET.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
007716  000416                          BR      2$              ;/BR TO END SUBTEST.
007720                          1$:                             ;/TRY CLEARING BIT 1.
007720  005037  001124                  CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                        ;/NOW READ IT BACK.
                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
                                ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
007744  005737  001126                  TST     $BDDAT
007750  001401                          BEQ     2$              ;/IF ZERO-NO ERROR!
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
007752  104006                          ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
                                                        ;/BIT 1 FAILED TO CLEAR.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
007754                          2$:
```

```
(3)                                     ;/#
(7)                                     ;;*****************************************************
(6)                                     ;*TEST 56       *TEST THAT CLOCK B BUFFER REGISTER BIT 2 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)  007754  000004                     TST56:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 2.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  007756  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)  010004  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
(3)  010012  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010014  104006                             ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                             ;/BIT 2 FAILED TO BIT SET.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010016  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  010020                             1$:                             ;/TRY CLEARING BIT 2.
(3)  010020  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)  010044  005737  001126                     TST     $BDDAT
(3)  010050  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010052  104006                             ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                             ;/BIT 2 FAILED TO CLEAR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010054                             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)  010054  000004                     TST57:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 3.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  010056  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)  010104  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
(3)  010112  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010114  104006                             ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                             ;/BIT 3 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010116  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  010120                             1$:                             ;/TRY CLEARING BIT 3.
(3)  010120  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)  010144  005737  001126                     TST     $BDDAT
(3)  010150  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010152  104006                             ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                             ;/BIT 3 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010154                             2$:
```

```
                                        :/#
                                ;;**********************************************************************
                                ;*TEST 60        *TEST THAT CLOCK B BUFFER REGISTER BIT 4 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
                                ;*
                                ;*
                                ;;**********************************************************************
010154  000004                  TST60:  SCOPE
                                                        ;/CLEAR THE BUFFER REGISTER.
                                                        ;/SET BIT 4.
                                                        ;/SET FOR ERROR TYPEOUT S/B.
                                                         ;/READ THE BUFFER REGISTER.
010156  012737  000020  001124          MOV     #BIT4,$GDDAT
                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
                                ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
010204  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
010212  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
010214  104006                          ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
                                                                ;/BIT 4 FAILED TO BIT SET.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
010216  000416                          BR      2$              ;/BR TO END SUBTEST.
010220                          1$:                             ;/TRY CLEARING BIT 4.
010220  005037  001124                  CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.
                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
                                ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
010244  005737  001126                  TST     $BDDAT
010250  001401                          BEQ     2$              ;/IF ZERO-NO ERROR!
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
010252  104006                          ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
                                                                ;/BIT 4 FAILED TO CLEAR.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
010254                          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)  010254  000004                     TST61:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 5.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  010256  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)  010304  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 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 5 FAILED TO BIT SET.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010316  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  010320                             1$:                             ;/TRY CLEARING BIT 5.
(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 5 FAILED TO CLEAR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010354                             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)  010354  000004                     TST62:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 6.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  010356  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)  010404  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
(3)  010412  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010414  104006                             ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                             ;/BIT 6 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010416  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  010420                             1$:                             ;/TRY CLEARING BIT 6.
(3)  010420  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)  010444  005737  001126                     TST     $BDDAT
(3)  010450  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010452  104006                             ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                             ;/BIT 6 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010454                             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)  010454  000004                     TST63:  SCOPE
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 7.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  010456  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)  010504  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 7 AND ONLY BIT 7 SET?
(3)  010512  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010514  104006                             ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                             ;/BIT 7 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

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

(3)  010516  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  010520                             1$:                             ;/TRY CLEARING BIT 7.
(3)  010520  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)  010544  005737  001126                     TST     $BDDAT
(3)  010550  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010552  104006                             ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                             ;/BIT 7 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010554                             2$:
1779                                    .SBTTL  *
1780                                    .SBTTL  * PHASE 2 ADVANCED BASIC LOGIC TESTS
1781                                    .SBTTL  *
1782
1788
1789                                    ;;*****************************************************************
```

```
    (3)                         ;*TEST 64       *TEST THAT CLOCK A'S COUNT REGISTER IS CLEAR
    (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) 010554  000004          TST64:  SCOPE
1790
1791                                                        ;SELECT MODE 0.
1792                                                        ;CLEAR THE BUFFER REGISTER. BUFFER
1793                                                        ;REGISTER WILL BE TRANSFERRED TO
1794                                                        ;COUNT REGISTER SINCE THIS IS MODE 0.
1795 010556  005037  001124              CLR     $GDDAT
1796
    (1)                         ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
1797
    (1)                         ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
1798
1799 010602  005037  001124              CLR     $GDDAT          ;EXPECT TO READ BACK ALL 0'S.
1800                                                        ;READ THE COUNT REGISTER - IT SHOULD BE CLEAR.
1801
    (1)                         ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
1802 010616  005737  001126              TST     $BDDAT
1803 010622  001401                      BEQ     1$              ;BR IF YES TO NEXT TEST
1804
1805
                    ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

1806 010624  104004                      ERROR   4               ;ERROR - CLOCK A'S COUNTER REGISTER
1807                                                        ;NOT CLEAR.
1808
1809
                    ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

1810 010626                         1$:
1811
1812                                ;;*****************************************************************
    (3)                         ;*TEST 65       *TEST CLOCK A'S COUNT REGISTER WITH 125252 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) 010626  000004          TST65:  SCOPE
1813
1814                                                        ;SELECT MODE 0.
```

```
                                                        ;LOAD THE BUFFER REGISTER WITH
                                                        ;PATTERN 125252. IT WILL BE
                                                        ;TRANSFERRED TO THE COUNT REGISTER
                                                        ;SINCE THIS IS MODE 0.
010630  005037  001124              CLR     $GDDAT
                             ;*     MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
010644  012737  125252  001124      MOV     #125252,$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.
010672  023737  001124  001126      CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
                                                        ;COME THROUGH?
010700  001401                      BEQ     1$              ;BR IF YES TO NEXT TEST.

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

010702  104004                      ERROR   4               ;DATA ERROR CLOCK A   PATTERN "125252"
                                                        ;FAILED TO TRANSFER PROPERLY BETWEEN
                                                        ;BUFFER AND COUNT REGISTERS.

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

010704                      1$:

                            ;;*****************************************************************
                            ;*TEST 66       *TEST CLOCK A'S COUNT REGISTER WITH 052525 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
                            ;*
                            ;*
                            ;;*****************************************************************
010704  000004             TST66:  SCOPE

                                                        ;SELECT MODE 0.
010706  005037  001124              CLR     $GDDAT

                             ;*     MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                                        ;LOAD THE BUFFER REGISTER WITH
                                                        ;PATTERN 052525. IT WILL BE
                                                        ;TRANSFERRED TO THE COUNT REGISTER
                                                        ;SINCE THIS IS MODE 0.

010722  012737  052525  001124      MOV     #052525,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
```

```
1851                                                    ;NEED OF ERROR TYPEOUT.
1852                                                    ;READ THE COUNT REGISTER
1853
 (1)                            ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
1854
 (1)                            ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
1855
1856  010750  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
1857                                                                  ;COME THROUGH?
1858  010756  001401                          BEQ     1$              ;BR IF YES TO NEXT TEST.
1859
1860
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
1861  010760  104004                          ERROR   4               ;DATA ERROR CLOCK A   PATTERN "052525"
1862                                                                  ;FAILED TO TRANSFER PROPERLY BETWEEN
1863                                                                  ;BUFFER AND COUNT REGISTERS.
1864
1865
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

1866  010762                           1$:
1867
1873
1874                                    ;;**********************************************************************
 (3)                                    ;*TEST 67        *TEST THAT CLOCK B'S COUNT REGISTER IS CLEAR
 (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)  010762  000004                    TST67:  SCOPE
1875
1876                                                            ;SELECT MODE 0.
1877                                                            ;CLEAR THE BUFFER REGISTER. BUFFER
1878                                                            ;REGISTER WILL BE TRANSFERRED TO
1879                                                            ;COUNT REGISTER SINCE THIS IS MODE 0.
1880  010764  005037  001124                       CLR     $GDDAT
1881
 (1)                            ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
1882
 (1)                            ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
1883
1884  011010  005037  001124                       CLR     $GDDAT          ;EXPECT TO READ BACK ALL 0'S.
1885                                                                  ;READ THE COUNT REGISTER - IT SHOULD BE CLEAR.
1886
 (1)                            ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
1887  011024  005737  001126                       TST     $BDDAT
1888  011030  001401                               BEQ     1$              ;BR IF YES TO NEXT TEST
1889
1890
```

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

1891  011032  104007                          ERROR   7               ;ERROR - CLOCK B'S COUNTER REGISTER
1892                                                                  ;NOT CLEAR.
1893
1894
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

1895  011034                          1$:
1896
1897
1898                                    ;;*****************************************************************
 (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)  011034  000004                    TST70:  SCOPE
1899
1900                                                                  ;SELECT MODE 0.
1901  011036  005037  001124                   CLR     $GDDAT
1902
 (1)                                ;*        MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
1903                                                                  ;LOAD THE BUFFER REGISTER WITH
1904                                                                  ;PATTERN 125. IT WILL BE
1905                                                                  ;TRANSFERRED TO THE COUNT REGISTER
1906                                                                  ;SINCE THIS IS MODE 0.
1907
1908  011052  012737  000125  001124          MOV     #125,$GDDAT     ;SET EXPECTED TO PATTERN IN CASE OF
1909
 (1)                                ;*        MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
1910                                                                  ;NEED OF ERROR TYPEOUT.
1911                                                                  ;READ THE COUNT REGISTER
1912
 (1)                                ;*        MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
1913
1914  011100  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
1915                                                                  ;COME THROUGH?
1916  011106  001401                          BEQ     1$              ;BR IF YES TO NEXT TEST.
1917
1918
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

1919  011110  104007                          ERROR   7               ;DATA ERROR CLOCK B - PATTERN "125"
1920                                                                  ;FAILED TO TRANSFER PROPERLY BETWEEN
1921                                                                  ;BUFFER AND COUNT REGISTERS.
1922
1923
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
 1924  011112                          1$:
 1925
 1926                                  ;;*****************************************************************
  (3)                                  ;*TEST 71       *TEST CLOCK B'S COUNT REGISTER WITH 252 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)  011112  000004                  TST71:  SCOPE
 1927
 1928                                                                ;SELECT MODE 0.
 1929  011114  005037  001124                  CLR     $GDDAT
 1930
  (1)                                  ;*       MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 1931                                                                 ;LOAD THE BUFFER REGISTER WITH
 1932                                                                 ;PATTERN 252. IT WILL BE
 1933                                                                 ;TRANSFERRED TO THE COUNT REGISTER
 1934                                                                 ;SINCE THIS IS MODE 0.
 1935
 1936  011130  012737  000252  001124          MOV     #252,$GDDAT     ;SET EXPECTED TO PATTERN IN CASE OF
 1937
  (1)                                  ;*       MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 1938                                                                 ;NEED OF ERROR TYPEOUT.
 1939                                                                 ;READ THE COUNT REGISTER
 1940
  (1)                                  ;*       MOV     @BCR,$BDDAI     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
 1941
 1942  011156  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
 1943                                                                 ;COME THROUGH?
 1944  011164  001401                          BEQ     1$              ;BR IF YES TO NEXT TEST.
 1945
 1946
                 ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
 1947  011166  104007                          ERROR   7               ;DATA ERROR CLOCK B - PATTERN ''252''
 1948                                                                 ;FAILED TO TRANSFER PROPERLY BETWEEN
 1949                                                                 ;BUFFER AND COUNT REGISTERS.
 1950
 1951
                 ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 1952  011170                          1$:
 1953
 1967
 1968                                  ;;*****************************************************************
  (3)                                  ;*TEST 72       *TEST THE SETTING OF MAINTENANCE STP1 IN CLOCK A BIT 15 TO SET
  (4)                                  ;*
  (4)                                  ;*NEW SIGNALS   GENERATION OF ''STP1'' BY 'LD STAT A'' H + 'BD 12'' H
  (5)                                  ;*
  (5)                                  ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 PER PASS
```

```
  (5)                                   ;*
  (4)                                   ;*
  (3)                                   ;;*****************************************************************
  (2)  011170  000004                   TST72:  SCOPE
1969
1970  011172  005037  001124                     CLR     $GDDAT
1971
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
1972
  (1)                                   ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
1973                                                                    ;MAKE SURE THE STATUS REGISTER IS CLEAR.
1974                                                                    ;SET MAINTENANCE STP1.
1975  011216  012737  010000  001124             MOV     #BIT12,$GDDAT
1976
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
1977
1978                                                                    ;DID BIT15 (STP1 FLAG) SET?
1979
  (1)                                   ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
1980  011244  005737  001126                     TST     $BDDAT
1981  011250  100401                           BMI     1$              ;BR IF YES - NEXT TEST
1982
1983
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

1984  011252  104001                           ERROR   1               ;ERROR - MAINTENANCE STP1 (BIT12)
1985                                                                    ;DID NOT SET BIT15 (STP1 FLAG) CLOCK A.
1986
1987
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

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

1988  011254                           1$:
1989
1997
1998                                    ;;*****************************************************************
  (3)                                   ;*TEST 73       *TEST THAT BIT00 IN CLOCK A STATUS REG. WILL SET WHEN BIT13 AND MAIN. ST
  (4)                                   ;*
  (4)                                   ;*NEW SIGNALS   "STP1" H + "ST1 ENB CNTR (1)" H DIRECT SETTING
  (4)                                   ;*              "ST1 FLAG"
  (5)                                   ;*
  (5)                                   ;* PROBABLE SYNC POINT FOR THIS TEST:: "DEVICE OUT" 2 PER PASS
  (5)                                   ;*
  (4)                                   ;*
  (3)                                   ;;*****************************************************************
  (2)  011254  000004                   TST73:  SCOPE
1999
2000                                                                    ;SET "ST1 ENB COUNTER" IN CLK A'S STATUS REG.
2001                                                                    ;GENERATE A MAINTENANCE ST1.
2002                                                                    ;DID BIT00 (ENABL CNTR A) SET?
2003  011256  012737  020000  001124             MOV     #BIT13,$GDDAT
2004
  (1)    .                              ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2005
  (1)                                   ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
2006  011304  052737  010000  001124             BIS     #BIT12,$GDDAT
2007
```

```
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2008
 (1)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
2009  011332  032737  000001  001126             BIT     #BIT00,$BDDAT
2010  011340  001001                             BNE     1$              ;BR IF YES - NEXT TEST.
2011
2012
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

2013  011342  104001                             ERROR   1               ;ERROR - BIT00 OF CLOCK A'S STATUS REGISTER
2014                                                                    ;FAILED TO SET WHEN BIT13 WAS SET
2015                                                                    ;AND A MAINTENANCE ST1 GENERATED.
2016
2017
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

2018  011344                                   1$:                     ;LEAVE SUBTEST WITH CLOCK CLEAR.
2019  011344  005037  001124                             CLR     $GDDAT
2020
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2021
2033
2034                                    ;;*****************************************************************
 (3)                                    ;*TEST 74       *TEST THAT CLOCK A WILL INCREMENT - 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)  011360  000004                    TST74:  SCOPE
2035
2036                                                                    ;CLEAR CLOCK A.
2037                                                                    ;CLEAR BUFFER REGISTER.
2038  011362  005037  001124                             CLR     $GDDAT
2039
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2040
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
2041                                                                    ;SELECT: MODE0! RATE "STP1" AND ENABLE
2042                                                                    ;CLOCK A TO COUNT.
2043                                                                    ;NOW GENERATE A MAINTENANCE STP1 - AT
2044                                                                    ;THIS TIME THE CLOCK SHOULD COUNT ONCE.
2045  011406  012737  000015  001124             MOV     #15,$GDDAT
2046
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2047
 (1)                                    ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
2048  011434  052737  010000  001124             BIS     #BIT12,$GDDAT
2049
```

```
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2050  011452  012737  000001  001124             MOV     #1,$GDDAT       ;FOR ERROR TYPEOUT (IF NEEDED) SET THE #1.
2051                                                                     ;READ THE COUNT REGISTER - SHOULD=1.
2052
 (1)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
2053  011470  023737  001126  001124             CMP     $BDDAT,$GDDAT   ;DID THE COUNTER COUNT ONCE?
2054  011476  001401                             BEQ     1$              ;IF YES - BR NEXT TEST.
2055
2056
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

2057  011500  104011                             ERROR   11              ;ERROR - COUNT A FAILED TO COUNT
2058                                                                     ;ONCE, WHEN ENABLED, MODE 0
2059                                                                     ;RATE "STP1" SEE ABOVE COMMENTS
2060                                                                     ;FOR COMPLETE DESCRIPTION AND LIST
2061                                                                     ;OF EVENTS.
2062
2063
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

2064  011502                        1$:

2065
2082
2083                                    ;;*********************************************************************
 (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)  011502  000004            TST75:  SCOPE
2084
2085  011504  005037  001124                     CLR     $GDDAT  ;START THE COUNTER FROM ZERO.
2086                                                                     ;NOTE: A VALUE OTHER THAN ZERO MAY BE
2087                                                                     ;PATCHED IN HERE IN ORDER THAT A COUNT
2088                                                                     ;MAY BE STARTED HIGHER.
2089  011510                        1$:                                  ;DISABLE CLOCK A.
2090  011510  005037  001354                     CLR     $TMDAT
2091
 (1)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
2092
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
2093                                                                     ;LOAD THE COUNTER BUFFER WITH VALUE.
2094                                                                     ;"1$" IS THE LOOP BACK POINT ON
```

```
2095                                                        ;"LOOP ON TEST" (SW14=1) FEATURE. NOWMAL
2096                                                        ;LOOP BACK POINT WILL BE "2$".
2097
2098  011534  012737  000015  001354          MOV     #15,$TMDAT
2099
 (1)                            ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
2100                                                        ;ENABLE COUNTER TO COUNT. SELECTED:
2101                                                        ;MODE 0, RATE "STP1"
2102  011552                    2$:
 (1)
 (1)                            ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
2103  011562  052737  010000  001354          BIS     #BIT12,$TMDAT   ;GENERATE A MAINTENANCE "STP1". THIS
2104
 (1)                            ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
2105                                                        ;ONE VALUE.
2106  011600  005237  001124          INC     $GDDAT          ;$GDDAT IS USED TO KEEP TRACK OF THE
2107                                                        ;VALUE THE CLOCK SHOULD COUNT TO.
2108
2109
 (1)                            ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
2110
2111  011614  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;DID COUNT OCCUR CORRECTLY?
2112  011622  001402                  BEQ     3$              ;IF YES - BR TO "3$".
2113
2114
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2115  011624  104011                  ERROR   11              ;ERROR - CLOCK A FAILED TO UPCOUNT
2116                                                        ;CORRECTLY USING MODE 0 - RATE "STP1".
2117
2118
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2119  011626  000403                  BR      4$
2120  011630  005737  001124          3$:     TST     $GDDAT          ;DID WE ACHIEVE OVERFLOW TO ZERO YET?
2121  011634  001410                  BEQ     5$              ;IF YES - BR NEXT TEST
2122
2123  011636  032777  040000  167274  4$:     BIT     #BIT14,@SWR     ;LOOP CURRENT COUNT?
2124  011644  001742                  BEQ     2$              ;BR IF NO TO NEXT COUNT UPDATE.
2125  011646  162737  000001  001124          SUB     #1,$GDDAT       ;IF YES - DECREMENT EXPECTED AND RELOAD.
2126  011654  000715                  BR      1$              ;GOTO RELOAD POINT.
2127
2128  011656                    5$:                           ;END SUBTEST
2129
2130                            .SBTTL  *
2131                            .SBTTL  * PHASE 3 CLOCK A COUNT FUNCTION TESTS
2132                            .SBTTL  *
2133
2143
2144                            ;;**************************************************************
 (3)                            ;*TEST 76        *TEST THAT CLOCK A OVERFLOW WILL OCCUR
 (4)                            ;*
 (4)                            ;*NOW WE'LL TRY AND GENERATE "A OVERFLOW L"
 (4)                            ;*WHICH SETS BDO5. WE'LL DO IT BY PRESETTING THE BUFFER
 (4)                            ;*AT 177777 AND GENERATE A MAINTENANCE STP1. WE ALREADY
 (4)                            ;*KNOW MAINTENANCE STP1'S WILL ADVANCE THE COUNTER.
```

```
                                        ;*ALSO SEE IF 'MODE'' FLG GETS SET.
                                        ;*
                                        ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
                                        ;*
                                        ;;*************************************************************
 011656  000004                         TST76:  SCOPE

                                                                        ;MAKE SURE CLOCK A CLEAR.
 011660  005037  001124                         CLR     $GDDAT

                                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 011674  012737  177777  001124                 MOV     #177777,$GDDAT  ;PRESET BUFFER TO ALL ONES.

                                        ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                                                        ;SELECT MODE 0; RATE STP4; GO.
                                                                        ;GENERATE A MAINTENANCE STP1.
 011712  012737  000015  001124                 MOV     #15,$GDDAT

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

                                        ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
 011740  052737  010000  001124                 BIS     #BIT12,$GDDAT

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

 011756                                 1$:                             ;DID OVERFLOW BIT SET?

                                        ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
 011766  032737  000040  001126                 BIT     #BIT05,$BDDAT
 011774  001002                                 BNE     2$              ;BR IF YES TO ''2$''.

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

 011776  104012                                 ERROR   12              ;ERROR BIT05 OF CSR CLOCK A FAILED
                                                                        ;TO SET ON OVERFLOW.

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

 012000  000411                                 BR      3$

 012002                                 2$:                             ;DID BIT07 - 'MODE'' FLG SET?

                                        ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
 012012  032737  000200  001126                 BIT     #BIT07,$BDDAT
                                                                        ;IT SHOULD SET WHEN BIT05 SETS.
 012020  001001                                 BNE     3$              ;BR IF YES TO 3$.

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

 012022  104012                                 ERROR   12              ;OVERFLOW FAILED TO SET CLOCK A'S
                                                                        ;CSR BIT07 'MODE'' FLG. IT
                                                                        ;HAD; HOWEVER SET BIT05 ''OVERFLOW''
                                                                        ;FLAG.

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

```
                                3$:
                                ;;*****************************************************************
                                ;*TEST 77       *TEST IN CLOCK A THAT OVERFLOW IN MODE 0 CAUSE CLEARING OF "ENB CNTR" F/
                                ;*
                                ;*WE'RE GOING TO SEE IF "A RELOAD" H AND "MODE 0" H CLEAR
                                ;*"ENB CNTR A" F/F.  "A RELOAD" GENERATED ON OVERFLOW AND
                                ;*"MODE 0" GENERATED BY "MODE A0 (0)" AND "MODE A0 (1)"
                                ;*ALSO SEE IF COUNTER REGISTER GETS RELOADED
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD BUFF A"
                                ;*
                                ;*
                                ;;*****************************************************************
(2)  012024  000004             TST77:  SCOPE
2191
2192                                                            ;MAKE SURE CLOCK A IS CLEAR.
2193  012026  005037  001124           CLR     $GDDAT
2194
(1)                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2195                                                            ;PRESET BUFFER TO ALL ONES.
2196                                                            ;SELECT MODE 0, RATE "STP4"; GO.
2197                                                            ;GENERATE A MAINTENANCE STP1.
2198                                                            ;THIS SHOULD CAUSE AN OVERFLOW
2199                                                            ;OVERFLOW WILL GENERATE "A RELOAD" H
2200                                                            ;COMBINE THIS WITH MODE 0 AND WE
2201  012042  012737  177777  001124           MOV     #177777,$GDDAT
2202
(1)                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
2203  012060  012737  000015  001124           MOV     #15,$GDDAT
2204
(1)                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2205
(1)                             ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
2206  012106  052737  010000  001124           BIS     #BIT12,$GDDAT
2207
(1)                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2208                                                            ;SHOULD CLEAR "ENB CNTR A" F/F.
2209
2210
(1)                             ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
2211  012134  032737  000001  001354           BIT     #BIT00,$TMDAT   ;DID BIT00 "ENB CNTR A" F/F CLEAR?
2212  012142  001402                           BEQ     1$              ;BR IF YES - NEXT TEST.
2213
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$
2214  012144  104012                           ERROR   12              ;"ENB CNTR A" F/F NOT CLEARED
2215                                                                  ;ON OVERFLOW.
2216
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$
2217  012146  000411                           BR      2$
2218
2219  012150                        1$:
(1)
(1)                             ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
2220  012160  022737  177777  001126           CMP     #177777,$BDDAT  ;DID COUNTER GET RELOADED AFTER
```

```
2221
2222  012166  001401                          BEQ     2$              ;THE OVERFLOW OCCURRED?
2223                                                                  ;BR IF YES - NEXT TEST.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
2224  012170  104012                          ERROR   12              ;ERROR CLOCK A - COUNTER DID NOT GET RELOAD FROM
2225                                                                  ;BUFFER AFTER OVERFLOW. "A RELOAD" H
2226                                                                  ;DID GET GENERATED ON WE WOULD
2227                                                                  ;HAVE GOT PREVIOUS ERROR; THEREFORE PROBLEM
2228                                                                  ;MUST EXIST WITH COMBINING "A RELOAD" H
2229                                                                  ;WITH "MODE A1 (0)" H TO GENERATE A
2230                                                                  ;COUNTER LOAD.
2231
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
2232  012172                          2$:
2280
```

```
                                        ;/#

                                ;;*****************************************************************
                                ;*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"
                                ;*
                                ;*
                                ;;*****************************************************************
012172  000004                  TST100: SCOPE

                                                        ;/MAKE SURE CLOCK IS CLEAR.
                                                        ;/CLEAR THE BUFFER.
                                                        ;/SELECT: MODE 0, RATE 1MHZ ; GO.
012174  005037  001124                  CLR     $GDDAT

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

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
012220  012737  000003  001124          MOV     #1!2,$GDDAT

                                :*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
012236  005000                          CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
012240  005200                  1$:     INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
012242  001376                          BNE     1$
                                                        ;/DID COUNTER INCREMENT AT ALL?

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
012254  005737  001126                  TST     $BDDAT
012260  001011                          BNE     2$              ;/IF YES - BR NEXT TEST.

                                                        ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
012272  032737  000040  001126          BIT     #BIT05,$BDDAT
                                                        ;/AT HIGH RATE - SO WE'LL SEE IF "OVERFLOW"
                                                        ;/F/F HAD SET.
012300  001001                          BNE     2$              ;/BR IF YES NEXT TEST.

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

012302  104012                          ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
                                                        ;/COUNT RATE: 1MHZ.

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

012304                          2$:
```

```
                                                 ;/#
 (1)
 (5)                                  ;;*************************************************************
 (4)                                  ;*TEST 101      *TEST THE ABILITY OF CLOCK A TO COUNT AT 100KHZ RATE PART 1
 (5)                                  ;*
 (5)                                  ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
 (5)                                  ;*IN RATE: 100KHZ               PART 1
 (6)                                  ;*
 (6)                                  ;* PROBABLE SYNC POINT FOR THIS TEST:: ''LD BUFF A''
 (6)                                  ;*
 (5)                                  ;*
 (4)                                  ;;*************************************************************
 (3)  012304  000004                  TST101: SCOPE
 (1)
 (1)                                                                  ;/MAKE SURE CLOCK IS CLEAR.
 (1)                                                                  ;/CLEAR THE BUFFER.
 (1)                                                                  ;/SELECT: MODE 0, RATE 100KHZ ; GO.
 (1)  012306  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 $GDDAT TO DEVICE REG ABR
 (1)  012332  012737  000005  001124          MOV     #1!4,$GDDAT
 (2)
 (2)                                  ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (1)  012350  005000                          CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
 (1)  012352  005200                  1$:     INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
 (1)  012354  001376                          BNE     1$
 (1)                                                                  ;/DID COUNTER INCREMENT AT ALL?
 (2)
 (2)                                  ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 (1)  012366  005737  001126                  TST     $BDDAT
 (1)  012372  001011                          BNE     2$              ;/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 $BDDAT.
 (1)  012404  032737  000040  001126          BIT     #BIT05,$BDDAT
 (1)                                                                  ;/AT HIGH RATE - SO WE'LL SEE IF ''OVERFLOW''
 (1)                                                                  ;/F/F HAD SET.
 (1)  012412  001001                          BNE     2$              ;/BR IF YES NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  012414  104012                          ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
 (1)                                                                  ;/COUNT RATE: 100KHZ.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  012416                          2$:
```

```
                                        ;/#

                                ;;*****************************************************************
                                ;*TEST 102      *TEST THE ABILITY OF CLOCK A TO COUNT AT 10KHZ RATE PART 1
                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
                                ;*IN RATE: 10KHZ                 PART 1
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD BUFF A"
                                ;*
                                ;*
                                ;;*****************************************************************
012416  000004                TST102: SCOPE

                                                        ;/MAKE SURE CLOCK IS CLEAR.
                                                        ;/CLEAR THE BUFFER.
                                                        ;/SELECT: MODE 0, RATE 10KHZ ; GO.
012420  005037  001124                  CLR     $GDDAT
                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
012444  012737  000007  001124          MOV     #1!6,$GDDAT
                                :*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
012462  005000                          CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
012464  005200                1$:       INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
012466  001376                          BNE     1$
                                                        ;/DID COUNTER INCREMENT AT ALL?
                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
012500  005737  001126                  TST     $BDDAT
012504  001011                          BNE     2$              ;/IF YES - BR NEXT TEST.

                                                        ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.
                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
012516  032737  000040  001126          BIT     #BIT05,$BDDAT
                                                        ;/AT HIGH RATE - SO WE'LL SEE IF "OVERFLOW"
                                                        ;/F/F HAD SET.
012524  001001                          BNE     2$              ;/BR IF YES NEXT TEST.

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

012526  104012                          ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
                                                        ;/COUNT RATE: 10KHZ.

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

012530                          2$:
```

```
                                        ;/#

                                ;;*****************************************************************
                                ;*TEST 103      *TEST THE ABILITY OF CLOCK A TO COUNT AT 1KHZ RATE PART 1
                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
                                ;*IN RATE: 1KHZ          PART 1
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD BUFF A"
                                ;*
                                ;*
                                ;;*****************************************************************
012530  000004                TST103: SCOPE

                                                        ;/MAKE SURE CLOCK IS CLEAR.
                                                        ;/CLEAR THE BUFFER.
                                                        ;/SELECT: MODE 0, RATE 1KHZ ; GO.
012532  005037  001124                  CLR     $GDDAT

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

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
012556  012737  000011  001124          MOV     #1!10,$GDDAT

                                :*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
012574  005000                  1$:     CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
012576  005200                          INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
012600  001376                          BNE     1$
                                                        ;/DID COUNTER INCREMENT AT ALL?

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
012612  005737  001126                  TST     $BDDAT
012616  001011                          BNE     2$              ;/IF YES - BR NEXT TEST.

                                                        ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
012630  032737  000040  001126          BIT     #BIT05,$BDDAT
                                                        ;/AT HIGH RATE - SO WE'LL SEE IF "OVERFLOW"
                                                        ;/F/F HAD SET.
012636  001001                          BNE     2$              ;/BR IF YES NEXT TEST.

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

012640  104012                          ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
                                                        ;/COUNT RATE: 1KHZ.

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

012642                          2$:
```

```
                                        ;/#

                                ;;*****************************************************************
                                ;*TEST 104       *TEST THE ABILITY OF CLOCK A TO COUNT AT 100HZ RATE PART 1
                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
                                ;*IN RATE: 100HZ                 PART 1
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A'
                                ;*
                                ;*
                                ;;*****************************************************************
(3)  012642  000004             TST104: SCOPE
(1)
(1)                                                             ;/MAKE SURE CLOCK IS CLEAR.
(1)                                                             ;/CLEAR THE BUFFER.
(1)                                                             ;/SELECT: MODE 0, RATE 100HZ ; GO.
(1)  012644  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 $GDDAT TO DEVICE REG ABR
(1)  012670  012737  000013  001124     MOV     #1!12,$GDDAT
(2)
(2)                             :*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  012706  005000                     CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
(1)  012710  005200             1$:     INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
(1)  012712  001376                     BNE     1$
(1)                                                             ;/DID COUNTER INCREMENT AT ALL?
(2)
(2)                             ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)  012724  005737  001126             TST     $BDDAT
(1)  012730  001011                     BNE     2$              ;/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 $BDDAT.
(1)  012742  032737  000040  001126     BIT     #BIT05,$BDDAT
(1)                                                             ;/AT HIGH RATE - SO WE'LL SEE IF ''OVERFLOW''
(1)                                                             ;/F/F HAD SET.
(1)  012750  001001                     BNE     2$              ;/BR IF YES NEXT TEST.
(2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
(1)  012752  104012                     ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
(1)                                                             ;/COUNT RATE: 100HZ.
(2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  012754                     2$:
```

```
                                        ;/#

                                ;;*****************************************************************
                                ;*TEST 105      *TEST THE ABILITY OF CLOCK A TO COUNT AT LINE-FREQ RATE PART 1
                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
                                ;*IN RATE: LINE-FREQ            PART 1
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A'
                                ;*
                                ;*
                                ;;*****************************************************************
012754  000004                  TST105: SCOPE

                                                        ;/MAKE SURE CLOCK IS CLEAR.
                                                        ;/CLEAR THE BUFFER.
                                                        ;/SELECT: MODE 0, RATE LINE-FREQ ; GO.
012756  005037  001124                  CLR     $GDDAT

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

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
013002  012737  000017  001124          MOV     #1!16,$GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
013020  005000                          CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
013022  005200                  1$:     INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
013024  001376                          BNE     1$
                                                        ;/DID COUNTER INCREMENT AT ALL?

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
013036  005737  001126                  TST     $BDDAT
013042  001011                          BNE     2$              ;/IF YES - BR NEXT TEST.

                                                        ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
013054  032737  000040  001126          BIT     #BIT05,$BDDAT
                                                        ;/AT HIGH RATE - SO WE'LL SEE IF ''OVERFLOW''
                                                        ;/F/F HAD SET.
013062  001001                          BNE     2$              ;/BR IF YES NEXT TEST.

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

013064  104012                          ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
                                                        ;/COUNT RATE: LINE-FREQ.

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

013066                  2$:


                                ;;*****************************************************************
                                ;*TEST 106      *TEST THAT CLOCK A DOESN'T COUNT WHEN NO RATE IS SELECTED
                                ;*
                                ;*WE KNOW THAT CLOCK A COUNTS AT ALL
```

```
                                        ;*ITS RATES, SO LETS BE SURE IT DOESNT
                                        ;*COUNT WHEN NO RATE IS SELECTED. ON
                                        ;*ERROR SUSPECT DUAL RATE SELECTION.
                                        ;*
                                        ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD BUFF A"
                                        ;*
                                        ;;*****************************************************************
013066  000004                          TST106: SCOPE

                                                                        ;CLEAR CLOCK A.
                                                                        ;ZERO ITS BUFFER.
                                                                        ;TELL THE CLOCK TO GO!
013070  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
013114  005237  001124                          INC     $GDDAT

                                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                                                        ;NO RATE SELECTED.
013130  005000                                  CLR     R0
013132  005200                          1$:     INC     R0
013134  001376                                  BNE     1$
                                                                        ;ANY COUNT OCCUR?

                                        ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
013146  005737  001126                          TST     $BDDAT
013152  001010                                  BNE     2$              ;IF COUNTER HAS SOMETHING IN IT REPORT ERROR


                                        ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
013164  032737  000040  001126                  BIT     #BIT05,$BDDAT   ;IF THE OVERFLOW BIT SET THEN IT MUST
                                                                        ;HAVE COUNTED.
013172  001401                                  BEQ     3$


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

013174  104012                          2$:     ERROR   12              ;AH HA, ERROR CLOCK A COUNTED WHEN
                                                                        ;ENABLED-BUT-NO-RATE SELECTED
                                                                        ;BETTER FIND OUT HOW CAUSED SIGNAL
                                                                        ;"CLOCK A" L.

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

013176                                  3$:



                                        ;;*****************************************************************
                                        ;*TEST 107      *TEST THAT CLOCK A'S COUNT REG ISN'T LOADED WHEN CLOCK A IS ENABLED
                                        ;*
                                        ;*IN THIS TEST WE'LL FIND OUT IF F/F "ENB CNTR A" WHEN SET,
                                        ;*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''
                                ;*
                                ;;*****************************************************************
013176  000004                TST107: 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.
013210  005737  001126                  TST     $BDDAT
013214  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
013240  005237  001124                  INC     $GDDAT          ;SET THE ENABLE F/F. THIS

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                                                ;FROM BEING LOADED WHEN THE
                                                                ;BUFFER IS LOADED.
013254  012737  177777  001124          MOV     #177777,$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.
013302  005737  001126                  TST     $BDDAT
013306  001401                          BEQ     1$              ;BR IF NO - NEXT TEST.

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

013310  104012                          ERROR   12              ;ERROR CLOCK A BUFFER TO COUNT REG TRANSFER
                                                                ;OCCURRED EVEN THOUGH THE ''ENB CNTR A'' F/F
                                                                ;WAS SET.

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

013312                          1$:


                                ;;*****************************************************************
                                ;*TEST 110       *TEST THAT CLOCK A IN MODE 1 DOES NOT CLEAR ENABLE ON OVERFLOW
                                ;*
                                ;*NOW WE'RE GOING TO SEE IF ''A OVERFLOW'' H WHEN GENERATED
                                ;*BY CAUSING AN OVERFLOW DOESN'T CLEAR THE ''ENB CNTR A'' F/F
                                ;*WHEN IN MODE 1. IF F/F GETS CLEARED SUSPECT SIGNAL ''MODE 0'' H.
                                ;*
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
```

```
 (5)                                    ;*
 (3)                                    ;;*****************************************************************
 (2)  013312  000004                    TST110: SCOPE
2380
2381                                                        ;MAKE SURE CLOCK A IS CLEAR.
2382                                                        ;SET BUFFER + COUNT REG. TO -1 FROM OVERFLOW
2383                                                        ;SET: MODE 1; RATE STP1; GO.
2384                                                        ;CAUSE A MAINTENANCE STP4. CLOCK 1
2385                                                        ;SHOULD OVERFLOW - BUT THIS OVERFLOW SHOULD
2386                                                        ;NOT CLEAR "ENB CNTR A" F/F.
2387                                                        ;DID BIT00, "ENB CNTR A" F/F GET CLEARED?
2388  013314  005037  001124                       CLR     $GDDAT
2389
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2390  013330  012737  177777  001124              MOV     #177777,$GDDAT
2391
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
2392  013346  012737  000415  001124              MOV     #415,$GDDAT
2393
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2394
 (1)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
2395  013374  052737  010000  001354              BIS     #BIT12,$TMDAT
2396
 (1)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
2397
 (1)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
2398  013422  032737  000001  001126              BIT     #BIT00,$BDDAT
2399  013430  001001                             BNE     1$              ;BR IF NO TO NEXT TEST.
2400
                                       ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
2401  013432  104012                             ERROR   12              ;ERROR MODE 1 OPERATION "ENB CNTR A" F/F
2402                                                        ;WAS CLEARED ON OVERFLOW.
2403
                                       ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
2404  013434                           1$:
2405
2417
2418                                   ;;*****************************************************************
 (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 TIME 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)  013434  000004                   TST111: SCOPE
2419
```

```
2420                                                    ;MAKE SURE CLOCK A IS CLEAR.
2421                                                    ;SET BUFFER + COUNT REG TO -1 FROM OVERFLOW.
2422                                                    ;SET: MODE 2; RATE STP1; GO.
2423                                                    ;CAUSE A MAINTENANCE STP1 - CLOCK ONCE.
2424                                                    ;THIS SHOULD CAUSE AN OVERFLOW AND LEAVE
2425                                                    ;THE COUNT REGISTER CLEAR AS A
2426                                                    ;BUFFER TO COUNT REG. SHOULDN'T OCCUR.
2427                                                    ;IS THE COUNT REG. CLEAR?
2428  013436  005037  001124          CLR     $GDDAT
2429
 (1)                              ;*     MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2430  013452  012737  177777  001124      MOV     #177777,$GDDAT
2431
 (1)                              ;*     MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
2432  013470  012737  001015  001124      MOV     #1015,$GDDAT
2433
 (1)                              ;*     MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2434
 (1)                              ;*     MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
2435  013516  052737  010000  001124      BIS     #BIT12,$GDDAT
2436
 (1)                              ;*     MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2437
 (1)                              ;*     MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
2438  013544  005737  001126          TST     $BDDAT
2439  013550  001401                  BEQ     1$              ;BR IF YES TO NEXT TEST.
2440
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
2441  013552  104012                  ERROR   12              ;ERROR THE CONTENTS OF THE BUFFER REG.
2442                                                    ;GOT TRANSFERRED TO THE COUNT REG.
2443                                                    ;(CLOCK A) ON OVERFLOW DOING A
2444                                                    ;MODE 2 OPERATION.
2445
2446                                                    ;THERE'S AN OUTSIDE CHANCE THAT THE
2447                                                    ;COUNT REG. NEVER GOES TO ZERO ON
2448                                                    ;AN OVERFLOW - THIS IS THE FIRST TIME
2449                                                    ;THAT WE WERE ABLE TO LOOK AT IT ON
2450                                                    ;OVERFLOW BECAUSE MODES 0 + 1 CAUSED
2451                                                    ;THAT AUTOMATIC BUFFER TO COUNT REG.
2452
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
2453  013554                    1$:
2454
2466
2467                                    ;;*****************************************************************
 (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 POINT FOR THIS TEST:: "RD STAT B"
 (5)                                ;*
 (4)                                ;*
 (3)                                ;;*****************************************************************
 (2)  013554  000004                TST112: SCOPE
2468
2469                                                                ;GENERATE A SYNC PULSE
2470
 (1)                                ;*      MOV     @BSR,$BDDAT    ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
2471  013566  005737  001126                TST     $BDDAT
2472                                                                ;MAKE SURE CLOCK A'S STAT REG IS CLEAR.
2473  013572  005037  001124                CLR     $GDDAT
2474
 (1)                                ;*      MOV     $GDDAT,@ASR    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2475                                                                ;SET MODE 2.
2476                                                                ;GENERATE MAINTENANCE ST2.
2477                                                                ;THE COMBO OF MODE 2 + ST2 SHOULD
2478                                                                ;GET "CNTR TO BUFF" H WHICH SHOULD
2479                                                                ;SET "MODE FLG" F/F.
2480  013606  012737  001000  001124        MOV     #1000,$GDDAT
2481
 (1)                                ;*      MOV     $GDDAT,@ASR    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2482
 (1)                                ;*      MOV     @ASR,$GDDAT    ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
2483  013634  052737  002000  001124        BIS     #BIT10,$GDDAT
2484
 (1)                                ;*      MOV     $GDDAT,@ASR    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2485                                                                ;DID IT SET?
2486
 (1)                                ;*      MOV     @ASR,$BDDAT    ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
2487  013662  032737  000200  001126        BIT     #BIT07,$BDDAT
2488  013670  001001                      BNE     1$             ;IF YES-BR TO NEXT TEST.
2489
                   ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$
2490  013672  104012                      ERROR   12             ;ERROR - MODE 2 + ST2 DID NOT SET MODE FL.
2491
                   ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$
2492  013674                           1$:
2493
2548
 (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)  013674  000004                     TST113: SCOPE
(1)
(1)                                                                 ;GENERATE A SYNC PULSE
(2)
(2)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(1)  013706  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)  013712  005037  001124                     CLR     $GDDAT
(2)
(2)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  013726  012737  052525  001124             MOV     #052525,$GDDAT
(2)
(2)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(1)  013744  012737  001001  001124             MOV     #1001,$GDDAT
(2)
(2)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  013762  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 $GDDAT.
(1)  014006  052737  002000  001124             BIS     #BIT10,$GDDAT
(2)
(2)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  014024  012737  052525  001124             MOV     #052525,$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)  014042  023737  001126  001124             CMP     $BDDAT,$GDDAT   ;WAS THE TRANSFER SUCCESSFUL?
(1)  014050  001401                             BEQ     1$              ;IF YES THEN BR TO NEXT TEST.
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

(1)  014052  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)  014054                         1$:
(1)          000011                             P=P+1
2549
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 114      *TEST THAT PATTERN 125252 CAN BE XFERRED BETWEEN A'S COUNT-BUFFER REGS
(5)                                     ;*
```

```
                        ;*NOW WE'LL SHOT THE WORKS - WE KNOW FROM THE PREVIOUS TEST WE
                        ;*CAN GENERATE "CNTR TO BUFF" H FROM MODE 2 + MAINTENANCE ST2, NOW
                        ;*WE WELL TRY AND GENERATE A TRANSFER BETWEEN THE COUNTER AND BUFFER
                        ;*USING A CB PAT PATTERN.
                        ;*IF NO DATA PATTERN GETS TRANSFERRED, SUSPECT SIG "LD BUFFER"
                        ;*TO BE STUCK LOW AT THE MUX INPUT FOR THE BUFFER OR
                        ;*"CNTR TO BUFF" H NOT GETTING THROUGH TO THE LOAD INPUTS OF THE
                        ;*BUFFER REGISTER.
                        ;*IF JUST ONE OR A FEW BITS GETS MESSED UP ON THE XFERR-
                        ;*SUSPECT THE RESPECTIVE MUX OR ETCH BETWEEN THE COUNT REG
                        ;*AND MUX. GOOD LUCK.
                        ;*
                        ;* PROBABLE SYNC POINT FOR THIS TEST:: "RD STAT B"
                        ;*
                        ;*
                        ;;*****************************************************************
014054  000004          TST114: SCOPE

                                                        ;GENERATE A SYNC PULSE

                        ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
014066  005737  001126          TST     $BDDAT
                                                        ;MAKE SURE CLOCK A IS CLEAR.
                                                        ;PUT PATTERN 125252 INTO BUFFER REG.
                                                        ;IT SHOULD GET XFERRED TO COUNT REG.
                                                        ;SELECT: MODE 2, ENABLE.
                                                        ;NOW GENERATE A MAINTENANCE ST2.
014072  005037  001124          CLR     $GDDAT

                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
014106  012737  125252  001124  MOV     #125252,$GDDAT

                        ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
014124  012737  001001  001124  MOV     #1001,$GDDAT

                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA  ROM $GDDAT TO DEVICE REG ASR
014142  005037  001124          CLR     $GDDAT

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

                        ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
014166  052737  002000  001124  BIS     #BIT10,$GDDAT

                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
014204  012737  125252  001124  MOV     #125252,$GDDAT  ;RECORD $GDDAT (PATTERN) IN CASE WE
                                                        ;NEED TO TYPE OUT AN ERROR.
                                                        ;NOW READ BACK THE BUFFER REG.

                        ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
014222  023737  001126  001124  CMP     $BDDAT,$GDDAT   ;WAS THE TRANSFER SUCCESSFUL?
014230  001401                  BEQ     1$              ;IF YES THEN BR TO NEXT TEST.

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

014232  104012                  ERROR   12              ;ERROR FAILED TO XFERR 125252 PATTERN
                                                        ;CORRECTLY FROM COUNT TO BUFFER REG.
```

```
(1)                                                     ;SEE INIT. COMMENT AS TO WHY
(1)                                                     ;IT MIGHT HAVE GONE SOUR.
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1) 014234                          1$:
(1)         000012                          P=P+1
2550
2557
2593
 (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) 014234  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) 014236  005037  001124                 CLR     $GDDAT
 (2)
 (2)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (1) 014252  012737  177777  001124         MOV     #177777,$GDDAT
 (2)
 (2)                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
 (1) 014270  012737  000200  001124         MOV     #200,$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 $GDDAT.
 (1) 014316  052737  002000  001124         BIS     #BIT10,$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) 014344  005737  001126                 TST     $BDDAT
 (1) 014350  001001                         BNE     1$              ;BR IF NO - NEXT TEST.
 (2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

 (1) 014352  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) 014354                         1$:
2594
```

```
                        ;;*****************************************************************
                        ;*TEST 116      *TEST THAT A'S COUNT REGISTER ISN'T CLEARED IN MODE 2 WHEN STP2 IS GENER
                        ;*
                        ;*THIS TEST IS DESIGNED TO MAKE SURE THAT WE DON'T CLEAR THE COUNT
                        ;*REGISTER IN MODE 2 WHEN AN STP2 IS GENERATED.
                        ;*
                        ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD BUFF A"
                        ;*
                        ;*
                        ;*****************************************************************
014354  000004          TST116: SCOPE

                                                        ;MAKE SURE CLOCK A IS CLEAR.
                                                        ;LOAD ALL ONES INTO BUFFER COUNT REGS.
                                                        ;SET MODE 2.
                                                        ;GENERATE A MAINTENANCE STP2.
                                                        ;SEE IF IT CLEARED COUNT REG.
014356  005037  001124          CLR     $GDDAT
                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
014372  012737  177777  001124  MOV     #177777,$GDDAT
                        ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
014410  012737  001000  001124  MOV     #1000,$GDDAT
                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                        ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
014436  052737  002000  001124  BIS     #BIT10,$GDDAT
                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                        ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
014464  005737  001126          TST     $BDDAT
014470  001001                  BNE     1$              ;BR IF NO - NEXT TEST.

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

014472  104012                  ERROR   12              ;ERROR CLOCK A MODE 2 + STP2 CLEARED COUNT REG.
                                                        ;TO SCOPE FIND OUT WHAT IS GENERATING
                                                        ;SIGNAL ON CLR INPUT OF COUNT REG.

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

014474                  1$:


                        ;;*****************************************************************
                        ;*TEST 117      *TEST THAT MODE 3 + "STP2" CLEARS A'S COUNT REGISTER.
                        ;*
                        ;*IN THIS TEST WE'LL DO A MODE 3 FOR THE FIRST TIME.
                        ;*MODE 3 + "STP2" SHOULD CLEAR THE COUNT REGISTER.
                        ;*WE KNOW FROM A PREVIOUS TEST THAT "INIT" L WAS ABLE TO
                        ;*CLEAR THE REG. AND ALSO THAT WE CAN GENERATE AN "STP2" H.
                        ;*'MODE A0 (1)" H + "MODE A1 (1)" H + "STP2" H = CLEARING SIGNAL.
                        ;*
```

```
 (5)                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A'
 (5)                                ;*
 (4)                                ;*
 (3)                                ;;*****************************************************************
 (2)  014474  000004                TST117: SCOPE
2608
2609                                                                ;CLEAR CLOCK A + SELECT MODE 3.
2610                                                                ;LOAD ALL ONE'S TO BUFFER + COUNTER REGS.
2611                                                                ;GENERATE A MAINTENANCE "STP2".
2612                                                                ;THIS SHOULD CLEAR THE COUNT REG.
2613
2614                                                                ;IS COUNT REG. CLEAR?
2615  014476  012737  001400  001124          MOV     #1400,$GDDAT
2616
 (1)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2617  014514  012737  177777  001124          MOV     #177777,$GDDAT
2618
 (1)                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
2619
 (1)                                ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
2620  014542  052737  002000  001124          BIS     #BIT10,$GDDAT
2621
 (1)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2622
 (1)                                :*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
2623  014570  005737  001126                  TST     $BDDAT
2624  014574  001401                          BEQ     1$              ;BR IF YES - NEXT TEST.
2625
                   ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2626  014576  104012                    ERROR   12                    ;ERROR-CLOCK A-COUNT REGISTER FAILED TO
2627                                                                ;CLEAR IN MODE 3 ON "STP2" PULSE.
2628
                   ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2629  014600                    1$:
2630
2650
2651                               ;;*****************************************************************
 (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)  014600  000004                TST120: SCOPE
2652
2653                                                        ;MAKE SURE CLOCK B IS CLEAR.
2654                                                        ;MAKE SURE CLOCK A IS CLEAR.
2655                                                        ;LOAD BUFFER + COUNT REGS.
2656                                                        ;SET AUTO INCREMENT MODE
2657                                                        ;ENABLE COUNTER, MODE 0, RATE STP1
2658                                                        ;ZAP! A MAINTENANCE STP1 HAS BEEN MADE.
2659                                                        ;STOP HERE AND LOOK WHAT JUST HAPPENED.
2660                                                        ;1. STP1 CLOCKED THE COUNTER (SET 177777).
2661                                                        ;2. COUNTER OVERFLOWED - "A OVERFLOW" L
2662                                                        ;3. MODE 0 + "A OVERFLOW" L + "A AUTO INC (1)" H
2663                                                        ;   DOWN COUNTED THE BUFFER.
2664                                                        ;4. "A OVERFLOW" L GOES THROUGH ONE SHOT DELAY
2665                                                        ;   TO GIVE "A RELOAD" H
2666                                                        ;5. "A RELOAD" H CAUSES A BUFFER TO COUNTER
2667                                                        ;   RELOAD.
2668  014602  005037  001124                CLR     $GDDAT
2669
 (1)                                ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FPOM $GDDAT TO DEVICE REG BSR
2670
 (1)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2671  014626  012737  177777  001124        MOV     #177777,$GDDAT
2672
 (1)                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
2673  014644  012737  000020  001124        MOV     #20,$GDDAT
2674
 (1)                                ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
2675  014662  012737  000015  001124        MOV     #15,$GDDAT
2676
 (1)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2677
 (1)                                ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
2678  014710  052737  010000  001124        BIS     #BIT12,$GDDAT
2679
 (1)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2680
2681                                        ;PART 1 DID BUFFER GET DECREMENTED?
2682
2683  014726  012737  177776  001124        MOV     #177776,$GDDAT  ;SET FOR ERROR TYPEOUT IF ANY.
2684                                                                ;READ THE RESULTS OF THE BUFFER.
2685
 (1)                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
2686  014744  023727  001126  177776        CMP     $BDDAT,#177776  ;DID BUFFER DECREMENT TO 177776?
2687  014752  001402                       BEQ     1$              ;BR IF YES TO PART 2
2688
                   ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
2689  014754  104012                       ERROR   12              ;ERROR-CLOCK A. AFTER AN OVERFLOW
2690                                                                ;WITH AUTO INC ENABLED MODE 0, THE
2691                                                                ;BUFFER REGISTER FAILED TO DOWN COUNT
2692                                                                ;SEE ABOVE COMMENTS FOR SEQUENCE
```

```
2693                                                    ;OF EVENTS.
2694  014756  000411                    BR      2$      ;IF ERROR ONLY LOOP ON PART 1.
2695
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2696  014760                        1$:   ;PART 2 DID NEW COUNT GET TRANSFERRED TO COUNT REGISTER?
2697
2698                                                    ;READ THE COUNT REGISTER
2699
 (1)                                  ;*    MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
2700  014770  023727  001126  177776        CMP     $BDDAT,#177776  ;DID NEW VALUE OF THE BUFFER
2701                                                    ;REGISTER GET PROPERLY LOADED INTO
2702                                                    ;THE COUNT REGISTER?
2703  014776  001401                    BEQ     2$      ;BR IF YES - NEXT TEST.
2704
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2705  015000  104012                    ERROR   12      ;ERROR CLOCK A. NEW CONTENTS OF
2706                                                    ;BUFFER REGISTER FAILED TO BE PROPERLY
2707                                                    ;LOADED INTO COUNT REGISTER AFTER
2708                                                    ;AUTO DECREMENT.
2709                                                    ;SEE ABOVE COMMENTS FOR SEQUENCE OF EVENTS.
2710
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2711  015002                        2$:                     ;CLEAR AUTO INC OPTION.
2712  015002  005037  001124               CLR     $GDDAT
2713
 (1)                                  ;*    MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
2723
2724                                                    ;;*****************************************************************
 (3)                                                    ;*TEST 121      *TEST THAT CLOCK A'S 1 MHZ CLK CAN BE DISABLED
 (4)                                                    ;*
 (4)                                                    ;*FOR THIS TEST, WE'LL TRY DISABLING THE 1 MHZ CLOCK. TO DO THIS,
 (4)                                                    ;*WE'LL SET THE "DISABLE OSC 1 MHZ", BIT 11 IN CLOCK B SR. THEN
 (4)                                                    ;*COUNTED OR OVERFLOWED, IF SO ERROR.
 (4)                                                    ;*THE UNKNOWN THING HERE IS BIT 11 SETTING THE DISABLE F/F THAT
 (4)                                                    ;*GATES WITH THE 1 MHZ FREG TO PRODUCE "A 1 MHZ CLK" L
 (5)                                                    ;*
 (5)                                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD STAT B"
 (5)                                                    ;*
 (3)                                                    ;;*****************************************************************
 (2)  015016  000004                    TST121: SCOPE
2725
2726                                                                ;CLEAR CLOCK A.
2727                                                                ;SET THE "DISABLE OSC 1 MHZ" F/F.
2728                                                                ;CLEAR THE BUFFER + COUNT REGS.
2729                                                                ;START CLOCK: RATE 1 MHZ, MODE 0, GO.
2730  015020  005037  001124                       CLR     $GDDAT
2731
 (1)                                  ;*    MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2732  015034  012737  004000  001124        MOV     #BIT11,$GDDAT
2733
 (1)                                  ;*    MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
2734  015052  005037  001124               CLR     $GDDAT
2735
```

```
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
2736  015066  012737  000003  001124             MOV     #3,$GDDAT
2737
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2738  015104  005000                           CLR     R0              ;SHORT DELAY. WE ALLOW THIS DELAY TO
2739  015106  105200                   1$:     INCB    R0              ;OCCUR. IF THE 1 MHZ CLOCK IS DISABLED
2740  015110  100376                           BPL     1$              ;NO CLOCKING OF CLOCK A WILL OCCUR.
2741                                                                    ;SEE SOMETHING IN THE COUNT REGISTERS
2742
 (1)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
2743  015122  005737  001126                   TST     $BDDAT          ;DOES COUNT REG HAVE ANYTHING IN IT?
2744  015126  001007                           BNE     2$              ;YES - REPORT ERROR!
2745
 (1)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
2746  015140  105737  001126                   TSTB    $BDDAT
2747  015144  100001                           BPL     3$              ;NO - BR NEXT TEST - NO ERROR.
2748
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2749  015146  104012                   2$:     ERROR   12              ;ERROR - UNABLE TO DISABLE 1 MHZ CLK
2750                                                                    ;USING "DISABLE OSC 1 MHZ" F/F
2751
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2752                                                                    ;CLEAR CLOCK A.
2753  015150  005037  001124           3$:     CLR     $GDDAT
2754
 (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"
                                ;*
                                ;*
                                ;:*****************************************************************
015164  000004                  TST122: SCOPE

                                                                ;CLEAR CLOCK B AND DISABLE 1MHZ OSC.
015166  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.
015204  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 $GDDAT.
015230  052737  000043  001124          BIS     #BIT5!BIT1!BIT0,$GDDAT

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

                        ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
015256  052737  004000  001124          BIS     #BIT11,$GDDAT

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

                        ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
015304  005737  001126          TST     $BDDAT
015310  001001                  BNE     1$              ;BR IF YES TO NEXT TEST.

                        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
015312  104014                  ERROR   14              ;ERROR - CLOCK B. FAILED TO ADVANCE
                                                        ;COUNT REGISTER.
```

```
2800                                                    ;TO SCOPE: CHECK LOAD + CLEAR INPUTS TO
2801                                                    ;COUNT REG TO SEE IF NOT BEING HELD LOW.
2802                                                    ;IF OK SEE IF WE'RE GETTING ANY "CLOCK B" L
2803                                                    ;PULSES - WE SHOULD GET ONE EACH LOOP OF
2804                                                    ;THIS SUBTEST. "CLOCK B" L COMES FROM
2805                                                    ;THE B CLOCK TIMING SECTION. THE RATE IS
2806                                                    ;1 MHZ. "B 1 MHZ" L IS GENERATED BY
2807                                                    ;"A 1 MHZ" H + "FEED B TO A (1)" H.
2808                                                    ;THE "A 1 MHZ" IS GENERATED BY "MAINT
2809                                                    ;SIMUL 1 MHZ" IN CLOCK A.
2810
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2811  015314                        1$:
2812
2826
2827                                ;;*****************************************************************
 (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)  015314  000004                TST123: SCOPE
2828
2829  015316  005037  001124                CLR     $GDDAT  ;START THE COUNTER FROM ZERO.
2830                                                          ;NOTE: A VALUE OTHER THAN ZERO MAY BE
2831                                                          ;PATCHED IN HERE IN ORDER THAT A COUNT
2832                                                          ;MAY BE STARTED HIGHER.
2833  015322  005037  001354          1$:   CLR     $TMDAT
2834
 (1)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
2835                                                          ;CLEAR CLOCK B, DISABLE 1 MHZ CLOCK A.
2836  015336  012737  004000  001354        MOV     #BIT11,$TMDAT
2837
 (1)                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
2838                                                          ;LOAD VALUE INTO COUNT + BUFFER REGS.
2839                                                          ;"1$" IS THE LOOP BACK POINT ON
2840                                                          ;"LOOP ON TEST" (SW14=1) FEATURE. NORMAL
2841                                                          ;LOOP BACK POINT WILL BE "2$".
2842
2843
 (1)                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
2844                                                          ;SELECT: "DISABLE OSC 1 MHZ"; "FEED B TO A";
2845                                                          ;RATE 1 MHZ.
2846                                                          ;GO. ENABL MUST BE SET AFTER "FEED B TO A"
2847  015364  012737  004042  001354        MOV     #4042,$TMDAT
```

```
2848
 (1)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
2849  015402  005237  001354                    INC     $TMDAT
2850
 (1)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
2851
2852  015416                           2$:                             ;GENERATE A CLOCK PULSE.
2853                                                                    ;SHOULD CAUSE THE CLOCK TO
2854                                                                    ;INCREMENT ONCE.
2855
 (1)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
2856  015426  052737  004000  001354              BIS     #BIT11,$TMDAT
2857
 (1)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
2858  015444  005237  001124                    INC     $GDDAT          ;$GDDAT IS USED TO KEEP TRACK OF THE
2859                                                                    ;VALUE THE CLOCK SHOULD COUNT TO.
2860
2861
 (1)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
2862
2863  015460  123737  001126  001124              CMPB    $BDDAT,$GDDAT   ;DID THE COUNT OCCUR CORRECTLY?
2864  015466  001402                            BEQ     3$              ;IF YES - BR "3$".
2865
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2866  015470  104015                            ERROR   15              ;ERROR CLOCK B FAILED TO UPCOUNT
2867                                                                    ;CORRECTLY - SEE COMMENTS AT SUB-TEST
2868                                                                    ;HEADING FOR MORE DETAILS.
2869
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

2870  015472  000403                                    BR      4$
2871
2872  015474  105737  001124            3$:     TSTB    $GDDAT          ;DID WE ACHIEVE OVERFLOW YET?
2873  015500  001410                            BEQ     5$              ;
2874
2875  015502  032777  040000  163430  4$:     BIT     #BIT14,@SWR     ;LOOP ON CURRENT COUNT?
2876  015510  001742                            BEQ     2$              ;BR IF NO TO NEXT COUNT UPDATE.
2877  015512  162737  000001  001124            SUB     #1,$GDDAT       ;IF YES DECREMENT EXPECTED AND RELOAD.
2878  015520  000700                            BR      1$              ;GO TO RELOAD POINT
2879
2880  015522                           5$:                             ;END SUBTEST.
2881
2892
2893                                    ;;*****************************************************************
 (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)  015522  000004                   TST124: SCOPE
2894
2895                                                                ;CLEAR CLOCK A.
2896                                                                ;CLEAR CLOCK B.
2897                                                                ;SET COUNT AND BUFFER REGS.
2898                                                                ;SELECT "DISABLE OSC 1 MHZ"; "FEED B TO A";
2899                                                                ;RATE 1 MHZ.
2900                                                                ;GO. ENABL MUST BE SET AFTER "FEED B TO A"
2901                                                                ;GENERATE A CLOCK PULSE.
2902
2903                                                                ;DID OVERFLOW FLAG SET?
2904  015524  005037  001124                CLR     $GDDAT
2905
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2906
  (1)                                   ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
2907  015550  012737  000377  001124          MOV     #377,$GDDAT
2908
  (1)                                   ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
2909  015566  012737  004042  001124          MOV     #4042,$GDDAT
2910
  (1)                                   ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
2911  015604  005237  001124                INC     $GDDAT
2912
  (1)                                   ;*      MOV     $GDDAT.@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
2913
  (1)                                   ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
2914  015630  052737  004000  001124          BIS     #BIT11,$GDDAT
2915
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
2916
  (1)                                   ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
2917  015656  105737  001126                TSTB    $BDDAT
2918  015662  100402                       BMI     1$              ;BR IF YES TO "1$".
2919
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$
2920  015664  104014                       ERROR   14              ;ERROR CLOCK B "B OVERFL FLAG" (CSR BIT7)
2921                                                                ;FAILED TO SET ON OVERFLOW.
2922
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$
2923  015666  000411                       BR      2$
2924  015670                        1$:                             ;WAS COUNTER RELOAD AFTER OVERFLOW
2925
  (1)                                   ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
2926  015700  022737  000377  001126          CMP     #377,$BDDAT
2927
2928  015706  001401                       BEQ     2$              ;BR IF YES TO NEXT TEST.
2929
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$
2930  015710  104014                       ERROR   14              ;ERROR CLOCK B COUNT REG. FAILED
2931                                                                ;TO BE RELOADED AFTER OVERFLOW.
2932
```

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

 2933  015712                          2$:
 2934
 2942
 2943                                   ;;*****************************************************************
  (3)                                   ;*TEST 125      *TEST THE INIT. ABILITY OF CLOCK B'S COUNT REG.
  (4)                                   ;*
  (4)                                   ;*ALL WE'RE GOING TO DO HERE IS TEST THE INITIALIZE INPUTS
  (4)                                   ;*TO THE 74193 ICS THAT FORM THE COUNT REGISTER.
  (5)                                   ;*
  (5)                                   ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD STAT A"
  (5)                                   ;*
  (4)                                   ;*
  (3)                                   ;*****************************************************************
  (2)  015712  000004                   TST125: SCOPE
 2944
 2945                                                                ;CLEAR CLOCK A.
 2946                                                                ;CLEAR CLOCK B.
 2947                                                                ;LOAD ALL ONES TO BUFFER + COUNT REGS.
 2948  015714  005037  001124                 CLR     $GDDAT
 2949
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 2950
  (1)                                   ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 2951  015740  012737  000377  001124           MOV     #377,$GDDAT
 2952
  (1)                                   ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 2953
 2954  015756  004737  032534                 JSR     PC,$RESET                       ;DO SYSTEM INIT - GENERATES "INIT" H.
 2955
 2956  015762  005037  001124                 CLR     $GDDAT          ;FIX $GDDAT FOR ERROR TYPEOUT, IF NEEDED.
 2957                                                                  ;READ B'S COUNT REGISTER, INIT
 2958
  (1)                                   ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
 2959  015776  005737  001126                 TST     $BDDAT
 2960                                                                  ;SHOULD HAVE MADE IT AL ZEROS.
 2961  016002  001401                         BEQ     1$              ;BR IF YES - NEXT TEST.
 2962
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 2963  016004  104007                         ERROR   7               ;ERROR CLOCK B - INIT FAILED TO CLEAR
 2964                                                                  ;COUNT REG.
 2965
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 2966  016006                          1$:
```

```
                        ;;*****************************************************************
                        ;*TEST 126      *TEST THAT CLOCK B DOESN'T COUNT WHEN NO RATE IS SELECTED
                        ;;*****************************************************************
016006  000004          TST126: SCOPE

                                                        ;CLEAR CLOCK A.
                                                        ;CLEAR CLOCK B.
                                                        ;ZEROS TO BUFFER + COUNT REGS.
                                                        ;ENABLE COUNTER TO COUNT - RATE - 0
                                                        ;(NO RATE).
016010  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
                        ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
016044  005237  001124          INC     $GDDAT
                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
016060  005000                  CLR     R0              ;DELAY FOR ANY COUNT THAT
016062  105200          1$:     INCB    R0              ;COULD FALSELY OCCUR.
016064  001376                  BNE     1$              ;THIS DELAY APPROX. 369 MS ON A
                                                        ;PDP 11/20.

                                                        ;DID ANY COUNT OCCUR?

                        ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
016076  005737  001126          TST     $BDDAT
016102  001401                  BEQ     2$              ;IF NO BR TO NEXT TEST.

                        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
016104  104014                  ERROR   14              ;ERROR - CLOCK B COUNTED WHEN ENABLED BUT
                                                        ;NO RATE SELECTED. BETTER FIND OUT
                                                        ;WHATS GENERATING ''CLOCK B'' L PULSES.

                        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
016106                  2$:
```

```
3048                                    ;/#
 (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 COUNT
 (5)                                    ;*IN RATE: 1MHZ PART1
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT A'
 (6)                                    ;*
 (5)                                    ;*
 (4)                                    ;;*****************************************************************
 (3)  016106  000004                    TST127: SCOPE
 (1)
 (1)                                                                    ;/CLEAR CLOCK A.
 (1)  016110  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 $GDDAT TO DEVICE REG BBR
 (1)  016144  012737  000003  001124            MOV     #1!2,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  016162  005000                    1$:     CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  016164  005200                            INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  016166  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)  016200  005737  001126                    TST     $BDDAT
 (1)  016204  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  016216  105737  001126                    TSTB    $BDDAT
 (1)                                                                    ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                                                    ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  016222  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
 (1)  016224  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                                                    ;/AT 1MHZ RATE.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  016226                            2$:
```

```
3049                                    ;/#
 (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)  016226  000004                    TST130: SCOPE
 (1)
 (1)                                                                    ;/CLEAR CLOCK A.
 (1)  016230  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 $GDDAT TO DEVICE REG BBR
 (1)  016264  012737  000005  001124            MOV     #1!4,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  016302  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  016304  005200                    1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  016306  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)  016320  005737  001126                    TST     $BDDAT
 (1)  016324  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  016336  105737  001126                    TSTB    $BDDAT
 (1)                                                                    ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                                                    ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  016342  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
 (1)  016344  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                                                    ;/AT 100KHZ RATE.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  016346                            2$:
```

```
3050
 (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)  016346  000004                    TST131: SCOPE
 (1)
 (1)                                                                ;/CLEAR CLOCK A.
 (1)  016350  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 10 DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 (1)  016404  012737  000007  001124            MOV     #1!6,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  016422  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  016424  005200                    1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  016426  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)  016440  005737  001126                    TST     $BDDAT
 (1)  016444  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  016456  105737  001126                    TSTB    $BDDAT
 (1)                                                                ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                                                ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  016462  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  016464  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                                                ;/AT 10KHZ RATE.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  016466                            2$:
```

```
3051                                    ;/#
 (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)  016466  000004                    TST132: SCOPE
 (1)
 (1)                                                                    ;/CLEAR CLOCK A.
 (1)  016470  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 $GDDAT TO DEVICE REG BBR
 (1)  016524  012737  000011  001124            MOV     #1!10,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  016542  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  016544  005200                    1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  016546  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)  016560  005737  001126                    TST     $BDDAT
 (1)  016564  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  016576  105737  001126                    TSTB    $BDDAT
 (1)                                                                    ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                                                    ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  016602  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
 (1)  016604  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                                                    ;/AT 1KHZ RATE.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  016606                            2$:
```

```
3052
 (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)  016606  000004                    TST133: SCOPE
 (1)
 (1)                                                                    ;/CLEAR CLOCK A.
 (1)  016610  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 $GDDAT TO DEVICE REG BBR
 (1)  016644  012737  000011  001124            MOV     #1!11,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  016662  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  016664  005200                    1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  016666  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)  016700  005737  001126                    TST     $BDDAT
 (1)  016704  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  016716  105737  001126                    TSTB    $BDDAT
 (1)                                                                    ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                                                    ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  016722  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  016724  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                                                    ;/AT 100HZ RATE.
 (2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  016726                            2$:
```

```
3053                                    ;/#
 (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)  016726  000004                    TST134: SCOPE
 (1)
 (1)                                                            ;/CLEAR CLOCK A.
 (1)  016730  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 $GDDAT TO DEVICE REG BBR
 (1)  016764  012737  000017  001124            MOV     #1!16,$GDDAT
 (2)
 (2)  ---  -  -  -  --  -;*-  -- MOV-    $GDDAT,@BSR     ;/-PUT DATA EROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  017002  005000                    1$:     CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  017004  005200                            INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  017006  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)  017020  005737  001126                    TST     $BDDAT
 (1)  017024  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  017036  105737  001126                    TSTB    $BDDAT
 (1)                                                            ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                                            ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  017042  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017044  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                                            ;/AT LINE-FREQ RATE.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017046                            2$:
3054
3063
```

```
                                ;;*****************************************************************
                                ;*TEST 135      *TEST THE "FEED B TO A" 24 BIT COUNTER FEATURE OF CLOCKS A + B
                                ;*
                                ;*WE'RE GOING TO TEST CLOCKS A+B AS A 24 BIT COUNTER; THAT IS;
                                ;*WE'RE GOINT TO TAKE THE OVERFLOW FROM CLOCK B AND FEED IT INTO
                                ;*CLOCK A.
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF B'
                                ;*
                                ;*
                                ;;*****************************************************************
017046  000004                  TST135: SCOPE
                                                        ;CLEAR CLOCK A.
                                                        ;CLEAR CLOCK B.
                                                        ;CLEAR A'S BUFFER + COUNT REGISTERS.
                                                        ;PRESET B'S BUFFER + COUNT TO -1 FROM
                                                        ;OVERFLOW.
                                                        ;SELECT: "DISABLE OSC 1 MHZ"; RATE 1 MHZ;
                                                        ;"FEED B TO A"
                                                        ;SET ENABL. MUST BE SET AFTER "FEED B TO A".
                                                        ;ENABLE CLOCK A; MODE 0; RATE 0.
                                                        ;SIMULATE A 1 MHZ PULSE - THIS PULSE
                                                        ;WILL CLOCK CLOCK B'S COUNTER
                                                        ;REGISTER. AN OVERFLOW WILL
                                                        ;OCCUR - THAT OVERFLOW SHOULD
                                                        ;CLOCK CLOCK A'S COUNT REGISTER.
                                                        ;DID CLOCK A'S COUNT REG GET CLOCKED?
017050  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
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
017104  012737  000377  001124          MOV     #377,$GDDAT
                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
017122  012737  004042  001124          MOV     #4042,$GDDAT
                                ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
017140  005237  001124                  INC     $GDDAT
                                ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
017154  012737  000001  001124          MOV     #1,$GDDAT
                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                ;*      MOV     @ASR,$GDDAT     :/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
017202  052737  004000  001124          BIS     #BIT11,$GDDAT
                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
017230  005737  001126                  TST     $BDDAT
```

```
3099  017234  001001                          BNE     1$              ;IF YES THEN BR NEXT TEST.
3100
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
3101  017236  104014                          ERROR   14              ;ERROR - UNABLE TO CLOCK CLOCK A'S
3102                                                                  ;COUNT REGISTER WITH THE OVERFLOW
3103                                                                  ;FROM CLOCK B.
3104                                                                  ;THE MOST LOGICAL PLACE TO START
3105                                                                  ;WOULD BE !A CLOCK TIMING''.
3106                                                                  ;''FEED B TO A (1)'' H + ''B OVERFLOW'' H
3107                                                                  ;SHOULD BE COMING TOGETHER GOING
3108                                                                  ;INTO THE MUX - BEGING SELECTED AND
3109                                                                  ;COMING OUT OF THE MUX TO BECOME
3110                                                                  ;''CLOCK A'' L.
3111
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

3112  017240                            1$:
3113                                    .SBTTL  *
3114                                    .SBTTL  * PHASE 6 CLOCK A+B ADVANCE TESTING
3115                                    .SBTTL  *
3116
```

```
                                        ;/#
                                ;;*****************************************************************
                                ;*TEST 136      *TEST THAT THE TRAILING EDGE OF STP1 WILL INCR. COUNTER
                                ;*
                                ;*IN THIS TEST WE'LL SEE IF WE CATCH THE FIRST COUNT
                                ;*AFTER STP1 COMES IN AND SETS THE ENABLE F/F ('ST1 ENB COUNTER'
                                ;*SET).
                                ;*WHAT WE SHOULD SEE IS THE LEADING EDGE OF ST1 COME
                                ;*IN AND SET THE ENB F/F AND THE TRAILING EDGE TRIGGER
                                ;*A 'CLOCK A' PULSE TO INCREMENT THE COUNTER.
                                ;*WE KNOW FROM A PREVIOUS TEST THAT AN STP1 WILL
                                ;*COME IN AND SET 'ENABL CNTR A' F/F AND THAT THE
                                ;*COUNTER WILL INCREMENT, SO WHATS HAPPENING IS WE'RE ACCUALLY
                                ;*LOOKING AT THE TRAILING EDGE OF STP1 TO SEE IF ITS DOING
                                ;*THE INCREMENTING
                                ;*
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD BUFF A"
                                ;*
                                ;;*****************************************************************
017240  000004                  TST136: SCOPE

                                                        ;CLR CLK A, SET 'ST1 ENB COUNTER'.;RATE:STP1.
                                                        ;CLR COUNT + BUFFER REGS.
                                                        ;GENERATE A MAINTENANCE ST1.
                                                        ;THE LEADING EDGE SHOULD CAUSE
                                                        ;'.ENABL CNTR A' F/F TO SET (CSR BIT 00).
                                                        ;THE TRAILING EDGE SHOULD CLOCK
                                                        ;THE COUNTER ONCE.
017242  012737  000001  001354          MOV     #BIT0,$TMDAT

                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
017260  005037  001124                  CLR     $GDDAT

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

                                ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
017304  052737  010000  001124          BIS     #BIT12,$GDDAT

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

017322  012737  000001  001124          MOV     #1,$GDDAT       ;SET S/B FOR ERROR TYPEOUT IF NEEDED.
                                                                ;READ THE COUNT REGISTER.

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
017340  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;DID THE COUNT REG COUNT ONCE?
017346  001401                          BEQ     1$              ;BR IF YES TO NEXT TEST.

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

017350  104012                          ERROR   12              ;ST1 FAILED TO COUNT
                                                                ;CLOCK A'S COUNT REG. AFTER SETTING
                                                                ;'ENABL CNTR A' - SEE TEST HEADING
```

```
3164                                                            ;COMMENTS
3165
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
3166
3167  017352                        1$:
3168
3257
```

```
                                        ;/#

                                ;;*****************************************************************
                                ;*TEST 137      *TEST CLOCK A'S 100KHZ DIVIDER
                                ;*
                                ;+IN THIS TEST WE'LL SEE IF THE 100KHZ DIVIDER WILL DIVIDE 1MHZ
                                ;+BY 10 TO GIVE US A 100KHZ CLK L PULSE.
                                ;+TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
                                ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 100KHZ
                                ;+PULSES.
                                ;*THEN WE'LL GENERATE 9 MORE 1MHZ PULSES AND MAKE
                                ;*SURE THAT WE DON'T GET ANOTHER 100KHZ PULSE.
                                ;*
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: ''LD BUFF A''
                                ;*
                                ;;*****************************************************************
017352  000004                  TST137: SCOPE

                                                        ;/CLEAR CLOCK B.
                                                        ;/CLEAR CLOCK A.
                                                        ;/CLEAR A'S BUFFER + COUNT REGS.
                                                        ;/DISABLE THE 1MHZ OSC.
                                                        ;/ENABLE CNTR, RATE: 100KHZ ;MODE.

017354  005037  001354                  CLR     $TMDAT

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

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

                                ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
017410  012737  004000  001354          MOV     #BIT11,$TMDAT

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

                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
017436  052737  000405  001354          BIS     #401!4,$TMDAT

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

                                                        ;/SET TO GENERATE ON 1MHZ PULSES
017454  012700  177766                  MOV     #-10.,R0

017460                          1$:                             ;/GENERATE 1 1MHZ PULSE
                                                                ;/HAS COUNTER ADVANCED ANY?

                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
017470  052737  004000  001354          BIS     #BIT11,$TMDAT

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

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
017516  005737  001126                  TST     $BDDAT
017522  001002                          BNE     10$             ;/IF SO EXIT THIS LOOP.
                                                                ;/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)  017524  005200                     INC     R0      ;/DONE 10. 1MHZ PULSES?
(1)  017526  001354                     BNE     1$      ;/IF NOT - DO ANOTHER.
(1)
(1)  017530  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)  017546  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  017554  001402                     BEQ     2$              ;/IF YES - NEXT CHECK.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  017556  104012                     ERROR   12              ;/ERROR - CLOCK A - 100KHZ - PULSE
(1)                                                     ;/NOT GENERATED WHEN 10 1MHZ PULSES
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  017560  000430                     BR      4$
(1)  017562  012700  000011          2$: MOV     #9.,R0  ;/GET THE NUMBER OF '1 MHZ' H PULSES
(1)
(1)  017566                          3$:                ;/GENERATE 9 1MHZ PULSES
(2)
(2)                                 ;*  MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  017576  052737  004000  001354      BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  017614  005300                     DEC     R0              ;/INORDER TO CHECK TO SEE THAT
(1)  017616  001363                     BNE     3$              ;/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)  017630  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 100KHZ PULSE GENERATED?
(1)  017636  001401                     BEQ     4$              ;/NO-GO TO NEXT TEST!
(1)
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  017640  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)  017642                          4$:
```

```
3259                                    ;/#
 (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)  017642  000004                    TST140: SCOPE
 (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)  017644  005037  001354                    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 $TMDAT TO DEVICE REG ABR
 (1)  017700  012737  004000  001354            MOV     #BIT11,$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 $TMDAT.
 (1)  017726  052737  000407  001354            BIS     #401!6,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (1)
 (1)                                                            ;/SET TO GENERATE ON 1MHZ PULSES
 (1)  017744  012700  177634                    MOV     #-100.,R0
 (1)
 (1)  017750                            1$:                             ;/GENERATE 1 1MHZ PULSE
 (1)                                                            ;/HAS COUNTER ADVANCED ANY?
 (2)
 (2)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  017760  052737  004000  001354            BIS     #BIT11,$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)  020006  005737  001126                    TST     $BDDAT
 (1)  020012  001002                            BNE     10$             ;/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)  020014  005200                          INC     R0         ;/DONE 100. 1MHZ PULSES?
(1)  020016  001354                          BNE     1$         ;/IF NOT - DO ANOTHER.
(1)
(1)  020020  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)  020036  023737  001126  001124           CMP     $BDDAT,$GDDAT ;/DID THE COUNTER ADVANCE ONCE?
(1)  020044  001402                          BEQ     2$         ;/IF YES - NEXT CHECK.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020046  104012                          ERROR   12         ;/ERROR - CLOCK A - 10KHZ - PULSE
(1)                                                             ;/NOT GENERATED WHEN 10 100KHZ PULSES
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020050  000430                          BR      4$
(1)  020052  012700  000143          2$:     MOV     #99.,R0 ;/GET THE NUMBER OF '1 MHZ' H PULSES
(1)
(1)  020056                          3$:                     ;/GENERATE 9 100KHZ PULSES
(2)
(2)                                  ;*      MOV     @ASR,$TMDAT ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  020066  052737  004000  001354           BIS     #BIT11,$TMDAT
(2)
(2)                                  ;*      MOV     $TMDAT,@ASR ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  020104  005300                          DEC     R0         ;/INORDER TO CHECK TO SEE THAT
(1)  020106  001363                          BNE     3$         ;/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)  020120  023737  001126  001124           CMP     $BDDAT,$GDDAT ;/WAS ANOTHER 10KHZ PULSE GENERATED?
(1)  020126  001401                          BEQ     4$         ;/NO-GO TO NEXT TEST!
(1)
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020130  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)  020132                          4$:
```

```
                                        ;/#

                                ;;*****************************************************************
                                ;*TEST 141       *TEST CLOCK A'S 1KHZ DIVIDER
                                ;*
                                ;+IN THIS TEST WE'LL SEE IF THE 1KHZ DIVIDER WILL DIVIDE 10KHZ
                                ;+BY 10 TO GIVE US A 1KHZ CLK L PULSE.
                                ;+TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
                                ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 1KHZ
                                ;*PULSES.
                                ;*THEN WE'LL GENERATE 9 MORE 10KHZ PULSES AND MAKE
                                ;*SURE THAT WE DON'T GET ANOTHER 1KHZ PULSE.
                                ;*
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A'
                                ;*
                                ;;*****************************************************************
020132  000004                TST141: SCOPE

                                                        ;/CLEAR CLOCK B.
                                                        ;/CLEAR CLOCK A.
                                                        ;/CLEAR A'S BUFFER + COUNT REGS.
                                                        ;/DISABLE THE 1MHZ OSC.
                                                        ;/ENABLE CNTR, RATE: 1KHZ ;MODE.

020134  005037  001354                CLR     $TMDAT

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

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

                                ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
020170  012737  004000  001354        MOV     #BIT11,$TMDAT

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

                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
020216  052737  000411  001354        BIS     #401!10,$TMDAT

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

                                                        ;/SET TO GENERATE ON 1MHZ PULSES
020234  012700  176030                MOV     #-1000.,R0

020240                          1$:                     ;/GENERATE 1 1MHZ PULSE
                                                        ;/HAS COUNTER ADVANCED ANY?

                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
020250  052737  004000  001354        BIS     #BIT11,$TMDAT

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

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
020276  005737  001126                TST     $BDDAT
020302  001002                        BNE     10$             ;/IF SO EXIT THIS LOOP.
                                                        ;/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)  020304  005200                      INC     R0         ;/DONE 1000. 1MHZ PULSES?
(1)  020306  001354                      BNE     1$         ;/IF NOT - DO ANOTHER.
(1)
(1)  020310  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)  020326  023737  001126  001124       CMP     $BDDAT,$GDDAT  ;/DID THE COUNTER ADVANCE ONCE?
(1)  020334  001402                      BEQ     2$         ;/IF YES - NEXT CHECK.
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020336  104012                      ERROR   12         ;/ERROR - CLOCK A - 1KHZ - PULSE
(1)                                                         ;/NOT GENERATED WHEN 10 10KHZ PULSES
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020340  000430                      BR      4$
(1)  020342  012700  001747          2$: MOV     #999.,R0   ;/GET THE NUMBER OF '1 MHZ' H PULSES
(1)
(1)  020346                          3$:                    ;/GENERATE 9 10KHZ PULSES
(2)
(2)                                  ;*     MOV     @ASR,$TMDAT  ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  020356  052737  004000  001354       BIS     #BIT11,$TMDAT
(2)
(2)                                  ;*     MOV     $TMDAT,@ASR  ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  020374  005300                      DEC     R0         ;/INORDER TO CHECK TO SEE THAT
(1)  020376  001363                      BNE     3$         ;/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)  020410  023737  001126  001124       CMP     $BDDAT,$GDDAT  ;/WAS ANOTHER 1KHZ PULSE GENERATED?
(1)  020416  001401                      BEQ     4$         ;/NO-GO TO NEXT TEST!
(1)
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020420  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)  020422                          4$:
```

```
                        ;/#

                        ;;*****************************************************************
                        ;*TEST 142      *TEST CLOCK A'S 100HZ DIVIDER
                        ;*
                        ;*IN THIS TEST WE'LL SEE IF THE 100HZ DIVIDER WILL DIVIDE 1KHZ
                        ;*BY 10 TO GIVE US A 100HZ CLK L PULSE.
                        ;*TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
                        ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 100HZ
                        ;*PULSES.
                        ;*THEN WE'LL GENERATE 9 MORE 1KHZ PULSES AND MAKE
                        ;*SURE THAT WE DON'T GET ANOTHER 100HZ PULSE.
                        ;*
                        ;*
                        ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A'
                        ;*
                        ;;*****************************************************************
(3)  020422  000004     TST142: SCOPE
(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)  020424  005037  001354             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 $TMDAT TO DEVICE REG ABR
(1)  020460  012737  004000  001354         MOV     #BIT11,$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 $TMDAT.
(1)  020506  052737  000413  001354         BIS     #401!12,$TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)
(1)                                                     ;/SET TO GENERATE ON 1MHZ PULSES
(1)  020524  012700  154360             MOV     #-10000.,R0
(1)
(1)  020530             1$:                             ;/GENERATE 1 1MHZ PULSE
(1)                                                     ;/HAS COUNTER ADVANCED ANY?
(2)
(2)                             ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  020540  052737  004000  001354         BIS     #BIT11,$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)  020566  005737  001126             TST     $BDDAT
(1)  020572  001002             BNE     10$             ;/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)  020574  005200                     INC     R0      ;/DONE 10000. 1MHZ PULSES?
(1)  020576  001354                     BNE     1$      ;/IF NOT - DO ANOTHER.
(1)
(1)  020600  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)  020616  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  020624  001402                     BEQ     2$              ;/IF YES - NEXT CHECK.
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020626  104012                     ERROR   12              ;/ERROR - CLOCK A - 100HZ - PULSE
(1)                                                     ;/NOT GENERATED WHEN 10 1KHZ PULSES
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020630  000430                     BR      4$
(1)  020632  012700  023417          2$: MOV     #9999.,R0       ;/GET THE NUMBER OF '1 MHZ' H PULSES
(1)
(1)  020636                          3$:                        ;/GENERATE 9 1KHZ PULSES
(2)
(2)                             ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  020646  052737  004000  001354      BIS     #BIT11,$TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  020664  005300                     DEC     R0              ;/INORDER TO CHECK TO SEE THAT
(1)  020666  001363                     BNE     3$              ;/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)  020700  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 100HZ PULSE GENERATED?
(1)  020706  001401                     BEQ     4$              ;/NO-GO TO NEXT TEST!
(1)
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020710  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)  020712                          4$:
3262
3361
3362
(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)  020712  000004                 TST143: SCOPE
(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)  020714  005037  001354                 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 $TMDAT TO DEVICE REG BBR
(1)  020750  012737  004040  001354       MOV     #BIT11!BIT5,$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 $TMDAT.
(1)  020776  052737  000004  001354       BIS     #4,$TMDAT
(2)
(2)                                 ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  021014  005237  001354                 INC     $TMDAT
(2)
(2)                                 ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  021030  012700  177766                 MOV     #-10.,R0        ;/SET TO GENERATE 10. 1MHZ PULSES
(1)
(1)  021034                         1$:                             ;/GENERATE 1 1MHZ PULSE
(1)                                                                 ;/HAS THE COUNTER ADVANCED?
(2)
(2)                                 ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021044  052737  004000  001354       BIS     #BIT11,$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)  021072  005737  001126                 TST     $BDDAT
(1)  021076  001002                         BNE     10$             ;/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)  021100  005200                          INC     R0              ;/DONE 10. 1MHZ PULSES?
(1)  021102  001354                          BNE     1$              ;/IF NOT - DO ANOTHER.
(1)
(1)  021104  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)  021122  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  021130  001402                          BEQ     2$              ;/IF YES - NEXT CHECK
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021132  104015                          ERROR   15              ;/ERROR - CLOCK B - 100KHZ - PULSE
(1)                                                                  ;/NOT GENERATED WHEN 10 1MHZ PULSE
(1)                                                                  ;/WERE GENERATED.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021134  000430                          BR      4$
(1)
(1)  021136  012700  177767          2$:     MOV     #-9.,R0 ;/GET THE NUMBER OF "1 MHZ" H PULSES
(1)                                                                  ;/NEED TO GIVE 9 1MHZ PULSES
(1)  021142                           3$:
(2)
(2)                                  ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021152  052737  004000  001354          BIS     #BIT11,$TMDAT
(2)
(2)                                  ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  021170  005200                          INC     R0              ;/IN ORDER TO CHECK TO SEE THAT
(1)  021172  001363                          BNE     3$              ;/WE DON'T GENERATE ANOTHER 100KHZ PULSE
(1)
(2)
(2)                                  ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  021204  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 100KHZ PULSE GENERATED?
(1)  021212  001401                          BEQ     4$              ;/NO - GO TO NEXT CHECK.
(1)
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021214  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)  021216                           4$:
3363
 (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)  021216  000004             TST144: SCOPE
(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)  021220  005037  001354             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 $TMDAT TO DEVICE REG BBR
(1)  021254  012737  004040  001354       MOV     #BIT11!BIT5,$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 $TMDAT.
(1)  021302  052737  000006  001354       BIS     #6,$TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  021320  005237  001354             INC     $TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  021334  012700  177634             MOV     #-100.,R0       ;/SET TO GENERATE 100. 1MHZ PULSES
(1)
(1)  021340                     1$:                             ;/GENERATE 1 1MHZ PULSE
(1)                                                             ;/HAS THE COUNTER ADVANCED?
(2)
(2)                             ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021350  052737  004000  001354       BIS     #BIT11,$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)  021376  005737  001126             TST     $BDDAT
(1)  021402  001002             10$     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)  021404  005200                     INC     R0              ;/DONE 100. 1MHZ PULSES?
(1)  021406  001354                     BNE     1$              ;/IF NOT - DO ANOTHER.
```

```
(1)
(1)  021410  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)  021426  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  021434  001402                          BEQ     2$              ;/IF YES - NEXT CHECK
(2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021436  104015                          ERROR   15              ;/ERROR - CLOCK B - 10KHZ - PULSE
(1)                                           ;/NOT GENERATED WHEN 10 100KHZ PULSE
(1)                                           ;/WERE GENERATED.
(2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021440  000430                          BR      4$
(1)
(1)  021442  012700  177635          2$:     MOV     #-99.,R0        ;/GET THE NUMBER OF ''1 MHZ'' H PULSES
(1)                                           ;/NEED TO GIVE 9 100KHZ PULSES
(1)  021446                           3$:
(2)
(2)                                   ·*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021456  052737  004000  001354          BIS     #BIT11,$TMDAT
(2)
(2)                                   ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  021474  005200                          INC     R0              ;/IN ORDER TO CHECK TO SEE THAT
(1)  021476  001363                          BNE     3$              ;/WE DON'T GENERATE ANOTHER 10KHZ PULSE
(1)
(2)
(2)                                   ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  021510  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 10KHZ PULSE GENERATED?
(1)  021516  001401                          BEQ     4$              ;/NO - GO TO NEXT CHECK.
(1)
(2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021520  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)  021522                           4$:
3364
 (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)  021522  000004                     TST145: SCOPE
(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)  021524  005037  001354                     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 $TMDAT TO DEVICE REG BBR
(1)  021560  012737  004040  001354             MOV     #BIT11!BIT5,$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 $TMDAT.
(1)  021606  052737  000010  001354             BIS     #10,$TMDAT
(2)
(2)                                     ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FRC' $TMDAT TO DEVICE REG BSR
(1)  021624  005237  001354                     INC     $TMDAT
(2)
(2)                                     ;*      MO,     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  021640  012700  176030                     MOV     #-1000.,R0      ;/SET TO GENERATE 1000. 1MHZ PULSES
(1)
(1)  021644                             1$:                             ;/GENERATE 1 1MHZ PULSE
(1)                                                                     ;/HAS THE COUNTER ADVANCED?
(2)
(2)                                     ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021654  052737  004000  001354             BIS     #BIT11,$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)  021702  005737  001126                     TST     $BDDAT
(1)  021706  001002                             BNE     10$             ;/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)  021710  005200                             INC     R0              ;/DONE 1000. 1MHZ PULSES?
(1)  021712  001354                             BNE     1$              ;/IF NOT - DO ANOTHER.
(1)
(1)  021714  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)  021732  023737  001126  001124        CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  021740  001402                        BEQ     2$              ;/IF YES - NEXT CHECK
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021742  104015                        ERROR   15              ;/ERROR - CLOCK B - 1KHZ - PULSE
(1)                                                         ;/NOT GENEPATED WHEN 10 10KHZ PULSE
(1)                                                         ;/WERE GENERATED.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021744  000430                        BR      4$
(1)
(1)  021746  012700  176031        2$:     MOV     #-999.,R0       ;/GET THE NUMBER OF ''1 MHZ'' H PULSES
(1)                                                         ;/NEED TO GIVE 9 10KHZ PULSES
(1)  021752                        3$:
(2)
(2)                             ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021762  052737  004000  001354        BIS     #BIT11,$TMDAT
(2)
(2)                             :*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  022000  005200                        INC     R0              ;/IN ORDER TO CHECK TO SEE THAT
(1)  022002  001363                        BNE     3$              ;/WE DON'T GENERATE ANOTHER 1KHZ PULSE
(1)
(2)
(2)                             ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  022014  023737  001126  001124        CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 1KHZ PULSE GENERATED?
(1)  022022  001401                        BEQ     4$              ;/NO - GO TO NEXT CHECK.
(1)
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022024  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)  022026                        4$:
3365
 (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)  022026  000004                     TST146: SCOPE
(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)  022030  005037  001354                     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 $TMDAT TO DEVICE REG BBR
(1)  022064  012737  004040  001354             MOV     #BIT11!BIT5,$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 $TMDAT.
(1)  022112  052737  000012  001354             BIS     #12,$TMDAT
(2)
(2)                                     ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  022130  005237  001354                     INC     $TMDAT
(2)
(2)                                     ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  022144  012700  154360                     MOV     #-10000.,R0     ;/SET TO GENERATE 10000. 1MHZ PULSES
(1)
(1)  022150                             1$:                             ;/GENERATE 1 1MHZ PULSE
(1)                                                                     ;/HAS THE COUNTER ADVANCED?
(2)
(2)                                     ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  022160  052737  004000  001354             BIS     #BIT11,$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)  022206  005737  001126                     TST     $BDDAT
(1)  022212  001002                             BNE     10$             ;/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)  022214  005200                             INC     R0              ;/DONE 10000. 1MHZ PULSES?
(1)  022216  001354                             BNE     1$              ;/IF NOT - DO ANOTHER.
(1)
(1)  022220  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)  022236  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  022244  001402                          BEQ     2$              ;/IF YES - NEXT CHECK
(2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022246  104015                          ERROR   15              ;/ERROR - CLOCK B - 100HZ - PULSE
(1)                                                                  ;/NOT GENERATED WHEN 10 1KHZ PULSE
(1)                                                                  ;/WERE GENERATED.
(2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022250  000430                          BR      4$
(1)
(1)  022252  012700  154361          2$:     MOV     #-9999.,R0      ;/GET THE NUMBER OF "1 MHZ" H PULSES
(1)                                                                  ;/NEED TO GIVE 9 1KHZ PULSES
(1)  022256                          3$:
(2)
(2)                                  ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  022266  052737  004000  001354          BIS     #BIT11,$TMDAT
(2)
(2)                                  ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  022304  005200                          INC     R0              ;/IN ORDER TO CHECK TO SEE THAT
(1)  022306  001363                          BNE     3$              ;/WE DON'T GENERATE ANOTHER 100HZ PULSE
(1)
(2)
(2)                                  ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  022320  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 100HZ PULSE GENERATED?
(1)  022326  001401                          BEQ     4$              ;/NO - GO TO NEXT CHECK.
(1)
(2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022330  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)  022332                          4$:
3366
3392
3393                                         .SBTTL
3394
3395                                 .SBTTL  END OF PASS ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*INCREMENT THE PASS NUMBER ($PASS)
(1)                                  ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
(1)                                  ;*IF THERES A MONITOR GO TO IT
(1)                                  ;*IF THERE ISN'T JUMP TO LOOP
(1)
(1)  022332                          $EOP:
(2)  022332  000240                          NOP
(1)  022334  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
(1)  022340  005237  001206                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
```

```
   (1)  022344  042737  100000  001206          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
   (1)  022352  005327                          DEC     (PC)+           ;;LOOP?
   (1)  022354  000001                  $EOPCT: .WORD   1
   (1)  022356  003022                          BGT     $DOAGN          ;;YES
   (1)  022360  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
   (1)  022362  000001                  $ENDCT: .WORD   1
   (1)  022364  022354                          $EOPCT
   (1)  022366  104401  022433                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
   (2)  022372  013746  001206                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
   (2)  022376  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
   (1)  022400  104401  022430                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
   (1)  022404  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
   (1)  022410  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
   (1)  022412  000005                          RESET                   ;;CLEAR THE WORLD
   (1)  022414  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
   (1)  022416  000240                          NOP                     ;;SAVE ROOM
   (1)  022420  000240                          NOP                     ;;FOR
   (1)  022422  000240                          NOP                     ;;ACT11
   (1)  022424                          $DOAGN:
   (1)  022424  000137                          JMP     @(PC)+          ;;RETURN
   (1)  022426  002334                  $RTNAD: .WORD   LOOP
   (1)  022430     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
   (1)  022433     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
   (1)  022440  050040  051501  020123
   (1)  022446  000043
3396                                            .SBTTL  *
3397                                            .SBTTL  * SPECIAL I/O SIGNAL TESTS
3398                                            .SBTTL  *
3399
3415
3433
3434                            .SBTTL  *       "STP2 OUT" TO "SCHMITT TRIG 1 IN" TESTS
3435                                    ;*
3436                                    ;* THIS IS A SPECIAL SECTION DEVOTED FOR TESTING AND
3437                                    ;* PROVIDING SCOPE LOOP CAPABILITIES FOR "STP2 OUT" L
3438                                    ;* AND "SCHMITT TRIG 1" IN.
3439                                    ;*
3440                                    ;* WHEN YOU LOAD AND START AT LOCATION 210, PROGRAM
3441                                    ;* CONTROL IS TRANSFERRED HERE. "STP2 OUT" L PULSES ARE
3442                                    ;* GENERATED BY "LO STAT A HI" H + "BD10" H (MAIN. STP2).
3443                                    ;* PIN V ("STP2 OUT") IS WIRED TO PIN LL (SCHMITT TRIG1) FOR
3444                                    ;* THIS TEST. "STP2 OUT" PULSES ARE RECEIVED AS "SCHMITT TRIG 1"
3445                                    ;* PULSES WHICH SET CLOCK A'S STATUS REGISTER BIT 15.
3446                                    ;* IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING TECHNIC.
3447                                    ;* AND ERROR SWITCH REGISTER OPTIONS ARE USED.
3448                                    ;* AN "*" IS TYPED AFTER EACH 65,324 LOOPS THROUGH THE
3449                                    ;* TEST. SW13=1 WILL INHIBIT THIS FEATURE.
3450                            ;*
 (1)                            ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD STAT B"
 (1)                            ;*
3451                                    ;*
3452                                    ;* YOU MUST WIRE PINS V AND LL OF J1 TOGETHER AND INSTALL JUMPER W3
3453                                    ;*
3454                                    ;* LOGIC TEST (L + S 200) SHOULD BE RUN FIRST.
3455                                    ;*
3456
```

```
 3457  022450  005037  001664          LS210:  CLR     .DVLS
 3458
  (1)  022454  005037  001104          1$:     CLR     $ICNT           ;/CLEAR ITERATION COUNT
  (1)  022460  012737  177704  001206          MOV     #-60.,$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)  022466                          2$:                             ;/CLEAR CLOCK B.
  (1)  022466  005037  001354                  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)
 3459
 3460                                                                  ;GENERATE AN "STP2 OUT" PULSE
 3461                                                                  ;AT THIS POINT YOU SHOULD
 3462                                                                  ;SEE AN OUTPUT AT PIN V.
 3463                                                                  ;PIN V SHOULD BE WIRED TO
 3464                                                                  ;PIN LL FOR "SCHMITT TRIG 1" IN.
 3465
 3466
  (1)                                 ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 3467
  (1)                                 ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 3468  022532  052737  002000  001354          BIS     #BIT10,$TMDAT   ;GENERATE STP2
 3469
  (1)                                 ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 3470  022550  000240                          NOP
 3471  022552  000240                          NOP
 3472
  (1)                                 ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 3473  022564  032737  100000  001354          BIT     #BIT15,$TMDAT   ;IS ST1 FLAG SET <BIT 15>
 3474  022572  001001                          BNE     3$              ;BR IF YES TO 3$:
 3475
 3476
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
 3477
 3478  022574  104000                          ERROR                   ;ERROR "SCHMITT TRIG 1" IN NOT
 3479                                                                  ;RECEIVED. HAVE YOU WIRED IT RIGHT?
 3480
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
 3481  022576                          3$:
 3482
  (1)  022576  032777  020000  156334          BIT     #BIT13,@SWR     ;/INHIBIT "*" TYPEOUT?
  (1)  022604  001330                          BNE     2$              ;/YES - IGNORE ANY UPDATES.
  (1)
  (1)  022606  005237  001104                  INC     $ICNT           ;/UPDATE COUNT.
  (1)  022612  001325                          BNE     2$              ;/IF NOT DONE 65,32' TIMES,
  (1)                                                                  ;/DO IT AGAIN.
  (1)
  (2)  022614  104401  022622                  TYPE    ,65$            ;;TYPE ASCIZ STRING
```

```
    (2)  022620  000401                    BR      64$             ;;GET OVER THE ASCIZ
    (2)                                 ;;65$:  .ASCIZ  #*#
    (2)  022624                         64$:
    (1)
    (1)  022624  005237  001206            INC     $PASS           ;/DONE 60 PASSES?
    (1)  022630  100716                    BMI     2$              ;/NO - NO NEED FOR CR,LF.
    (2)  022632  104401  022640            TYPE    ,67$            ;;TYPE ASCIZ STRING
    (2)  022636  000402                    BR      66$             ;;GET OVER THE ASCIZ
    (2)                                 ;;67$:  .ASCIZ  <15><12>##
    (2)  022644                         66$:
    (1)  022644  000703                    BR      1$
    (1)
  3483
  3484                                  .SBTTL  ;*      "STP1 OUT" TO "SCHMITT TRIG 2" H TESTS
  3485                                          ;*
  3486                                          ;* THIS IS A SPECIAL TEST SECTION DEVOTED FOR TESTING AND
  3487                                          ;* PROVIDING SCOPE LOOP CAPABILITIES FOR "STP1 OUT" AND
  3488                                          ;* "SCHMITT TRIG2" IN.
  3489                                          ;*
  3490                                          ;* WHEN YOU LOAD AND START AT LOCATION 214, PROGRAM
  3491                                          ;* CONTROL IS TRANSFERRED HERE. "STP1 OUT" L PULSES ARE
  3492                                          ;* GENERATED BY "LD STAT A HI" + "BD12" H (MAIN ST1).
  3493                                          ;* PIN DD ("STP1 OUT") IS WIRED TO PIN BB ("SCHMITT
  3494                                          ;* TRIG 2") FOR THIS TEST. "STP1 OUT" PULSES ARE RECEIVED AS
  3495                                          ;* "SCHMITT TRIG 2" PULSES WHICH WILL CLEAR CLOCK A'S
  3496                                          ;* COUNT REGISTER IF MODE 3 IS SELECTED.
  3497                                          ;* IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING TECHNIC.
  3498                                          ;* AND ERROR SWITCH REGISTER OPTIONS ARE USED.
  3499                                          ;* AN "*" IS TYPED AFTER EACH 65,324 LOOPS THROUGH
  3500                                          ;* THE TEST. SW13=1 WILL INHIBIT THIS FEATURE.
  3501                                          ;*
  3502                                  ;*
    (1)                                 ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD STAT B"
    (1)                                 ;*
  3503                                          ;* YOU MUST WIRE PINS DD AND BB OF J1 TOGETHER AND INSTALL JUMPER W4.
  3504                                          ;*
  3505                                          ;* LOGIC TESTS (L + S AT 200) SHOULD BE RUN FIRST.
  3506                                          ;*
  3507
  3508  022646  005037  001664         LS214:  CLR     .DVLS
  3509
    (1)  022652  005037  001104         1$:     CLR     $ICNT           ;/CLEAR ITERATION COUNT
    (1)  022656  012737  177704  001206         MOV     #-60.,$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)  022664                         2$:                             ;/CLEAR CLOCK B.
    (1)  022664  005037  001354                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)
  3510                                                                  ;LOAD ALL ONES TO CLK A'S BUFFER + COUNT REG.
```

```
3511                                                    ;SELECT MODE 3.
3512                                                    ;GENERATE A "STP1 OUT" PULSE.
3513                                                    ;AT THIS POINT WE SHOULD SEE AN
3514                                                    ;OUTPUT AT PIN DD.  PIN DD SHOULD
3515                                                    ;BE WIRED TO PIN BB FOR
3516                                                    ;"SCHMITT TRIG 2" IN. THIS SHOULD
3517                                                    ;CAUSE AN "STP2" WHICH WILL CLEAR
3518                                                    ;CLOCK A'S COUNT REGISTER.
3519  022710  012737  177777  001354          MOV     #-1,$TMDAT
3520
 (1)                              ;*       MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
3521  022726  012737  001400  001354          MOV     #BIT8!BIT9,$TMDAT
3522
 (1)                              ;*       MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
3523
 (1)                              ;*       MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
3524  022754  052737  010000  001354          BIS     #BIT12,$TMDAT
3525
 (1)                              ;*       MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
3526                                                    ;WAS THE COUNT REGISTER CLEARED?
3527
 (1)                              ;*       MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
3528  023002  005737  001126                  TST     $BDDAT
3529
3530  023006  001401                          BEQ     3$              ;IF YES BR 3$.
3531
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

3532  023010  104000                          ERROR                   ;ERROR "SCHMITT TRIG 2" IN NOT
3533                                                                  ;RECEIVED. HAVE YOU WIRED IT RIGHT?
3534
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

3535  023012                          3$:
3536
 (1)  023012  032777  020000  156120          BIT     #BIT13,@SWR     ;/INHIBIT "*" TYPEOUT?
 (1)  023020  001321                          BNE     2$              ;/YES - IGNORE ANY UPDATES.
 (1)
 (1)  023022  005237  001104                  INC     $ICNT           ;/UPDATE COUNT.
 (1)  023026  001316                          BNE     2$              ;/IF NOT DONE 65,324 TIMES,
 (1)                                                                  ;/DO IT AGAIN.
 (1)
 (2)  023030  104401  023036                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 (2)  023034  000401                          BR      64$             ;;GET OVER THE ASCIZ
 (2)                              ;;65$:  .ASCIZ  #*#
 (2)  023040                      64$:
 (1)
 (1)  023040  005237  001206                  INC     $PASS           ;/DONE 60 PASSES?
 (1)  023044  100707                          BMI     2$              ;/NO - NO NEED FOR CR,LF.
 (2)  023046  104401  023054                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 (2)  023052  000402                          BR      66$             ;;GET OVER THE ASCIZ
 (2)                              ;;67$:  .ASCIZ  <15><12>##
 (2)  023060                      66$:
 (1)  023060  000674                          BR      1$
 (1)
3537
```

```
                                .SBTTL  *       "SCHMITT TRIG 3" IN, "ST3 OUT" TESTS
                                        ;*
                                        ;*THIS IS A SPECIAL SECTION DEVOTED FOR TESTING AND
                                        ;*PROVIDING SCOPE LOOP 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 TRIG 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 TECHNIC.
                                        ;*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 DD TO T OF J1 TOGETHER, AS WELL AS
                                        ;*PINS L TO BB OF J1 TOGETHER AND INSTALL JUMPERS W4 AND W5.
                                        ;*
                                        ;*TESTS LS210 AND LS214 SHOULD BE RUN FIRST.
                                        ;*
023062  005037  001664          LS220:  CLR     .DVLS
023066  005037  001104          1$:     CLR     $ICNT           ;/CLEAR ITERATION COUNT
023072  012737  177704  001206          MOV     #-60.,$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.
023100                          2$:                             ;/CLEAR CLOCK B.
023100  005037  001354                  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 A "STP2 OUT" PULSE.
023124  005037  001354                  CLR     $TMDAT
                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
023140  052737  011000  001354          BIS     #BIT12!BIT9,$TMDAT
                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
                                                                ;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
```

```
3576                                                    ;TRIG2 IN") SHOULD GIVE "ST2" WHICH
3577                                                    ;SETS CLOCK A'S CSR BIT 7.
3578
3579                                                    ;DID BIT 7 SET?
3580
 (1)                            ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
3581  023166  105737  001126              TSTB    $BDDAT
3582  023172  100401                      BMI     3$              ;IF YES, BR TO 3$.
3583
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
3584  023174  104000                      ERROR                   ;ERROR "SCHMITT TRIG 2" NOT RECEIVED
3585                                                             ;HAVE YOU WIRED IT RIGHT?
3586
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
3587  023176                        3$:
3588
 (1)  023176  032777  020000  155734        BIT     #BIT13,@SWR     ;/INHIBIT "*" TYPEOUT?
 (1)  023204  001335                        BNE     2$              ;/YES - IGNORE ANY UPDATES.
 (1)
 (1)  023206  005237  001104                INC     $ICNT           ;/UPDATE COUNT.
 (1)  023212  001332                        BNE     2$              ;/IF NOT DONE 65,324 TIMES,
 (1)                                                             ;/DO IT AGAIN.
 (1)
 (2)  023214  104401  023222                TYPE    ,65$            ;;TYPE ASCIZ STRING
 (2)  023220  000401                        BR      64$             ;;GET OVER THE ASCIZ
 (2)                            ;;65$:  .ASCIZ  #*#
 (2)  023224                    64$:
 (1)
 (1)  023224  005237  001206                INC     $PASS           ;/DONE 60 PASSES?
 (1)  023230  100723                        BMI     2$              ;/NO - NO NEED FOR CR,LF.
 (2)  023232  104401  023240                TYPE    ,67$            ;;TYPE ASCIZ STRING
 (2)  023236  000402                        BR      66$             ;;GET OVER THE ASCIZ
 (2)                            ;;67$:  .ASCIZ  <15><12>##
 (2)  023244                    66$:
 (1)  023244  000710                        BR      1$
 (1)
3589
3590                            .SBTTL  *       "A EVENT OUT" TEST
3591                                    ;*
3592                                    ;*THIS IS A SPECIAL SECTION DEVOTED FOR TESTING AND
3593                                    ;*PROVIDING SCOPE LOOP CAPABILITIES FOR "A EVENT OUT".
3594                                    ;*
3595                                    ;*WHEN YOU LOAD AND START AT LOCATION 224, PROGRAM
3596                                    ;*CONTROL IS TRANSFERRED HERE. "A EVENT OUT" PULSES ARE
3597                                    ;*GENERATED BY CLOCK A OVERFLOWS. PIN VV
3598                                    ;*("A EVENT OUT") IS WIRED TO PIN BB ("SCHMITT TRIG 2")
3599                                    ;*"SCHMITT TRIG 2" PULSES WILL SET CLOCK A'S CSR BIT 7.
3600                                    ;*IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING
3601                                    ;*AND ERROR SWITCH REGISTER OPTIONS ARE USED.
3602                                    ;*AN "*" IS TYPED AFTER EACH 65,324 LOOPS THROUGH THE TEST.
3603                                    ;*SW13=1 WILL INHIBIT THIS FEATURE.
3604                                    ;*
3605                            ;*
 (1)                            ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD STAT B"
```

```
 (1)                                    ;*
3606                                    ;*YOU MUST WIRE PINS VV AND BB OF J1 TOGETHER AND INSTALL JUMPER W4.
3607                                    ;*
3608                                    ;*TEST LS210 SHOULD BE RUN FIRST.
3609                                    ;*
3610
3611  023246  005037  001664      LS224:  CLR     .DVLS
3612
 (1)  023252  005037  001104      1$:     CLR     $ICNT           ;/CLEAR ITERATION COUNT
 (1)  023256  012737  177704  001206      MOV     #-60.,$PASS     ;/SET PASS COUNT.
 (1)                                                            ;/NOTE: PASS COUNT USED ONLY
 (1)                                                            ;/      TO DETECT 60 P'SSES SO
 (1)                                                            ;/      IT CAN GENERATE A CRLF.
 (1)                                                            ;/      AFTER CRLF IT WILL BE ZEROED.
 (1)                                                            ;/CLEAR CLOCK A.
 (1)  023264                            2$:                     ;/CLEAR CLOCK B.
 (1)  023264  005037  001354                CLR     $TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (1)
3613                                                            ;PRELOAD CLOCK A'S BUFFER + COUNT REGS.
3614  023310  012737  177777  001354      MOV     #-1,$TMDAT
3615
 (1)                                    ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
3616
3617                                                            ;RATE: 1MHZ, ENABLE COUNTER
3618  023326  012737  001003  001354      MOV     #1003,$TMDAT
3619
 (1)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
3620
3621  023344                            3$:                     ;HAS AN OVERFLOW OCCURRED?
3622
 (1)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
3623  023354  032737  000040  001354      BIT     #BIT05,$TMDAT
3624  023362  001770                      BEQ     3$              ;NO - THEN WAIT FOR IT.
3625                                                            ;OVERFLOW SHOULD GO OUT AS
3626                                                            ;"A EVENT OUT". YOU SHOULD SEE
3627                                                            ;AN OUTPUT AT PIN VV.
3628                                                            ;THIS IN TURN IS BROUGHT
3629                                                            ;IN AS "SCHMITT TRIG 2" IN TO
3630                                                            ;SET CLOCK A'S CSR BIT 7.
3631
3632
 (1)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
3633  023374  105737  001126                TSTB    $BDDAT          ;DID BIT 7 SET
3634  023400  100401                      BMI     4$              ;BR IF YES TO 4$
3635
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

3636  023402  104000                      ERROR                   ;ERROR "A EVENT OUT" PULSE
3637                                                            ;NOT DETECTED. HAVE YOU
3638                                                            ;WIRED IT RIGHT?
3639
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
3640  023404                                4$:
3641
 (1)  023404  032777  020000  155526              BIT     #BIT13,@SWR     ;/INHIBIT "*" TYPEOUT?
 (1)  023412  001324                              BNE     2$              ;/YES - IGNORE ANY UPDATES.
 (1)
 (1)  023414  005237  001104                      INC     $ICNT           ;/UPDATE COUNT.
 (1)  023420  001321                              BNE     2$              ;/IF NOT DONE 65,324 TIMES,
 (1)                                                                      ;/DO IT AGAIN.
 (1)
 (2)  023422  104401  023430                      TYPE    ,65$            ;;TYPE ASCIZ STRING
 (2)  023426  000401                              BR      64$             ;;GET OVER THE ASCIZ
 (2)                                       ;;65$: .ASCIZ  #*#
 (2)  023432                               64$:
 (1)
 (1)  023432  005237  001206                      INC     $PASS           ;/DONE 60 PASSES?
 (1)  023436  100712                              BMI     2$              ;/NO - NO NEED FOR CR,LF.
 (2)  023440  104401  023446                      TYPE    ,67$            ;;TYPE ASCIZ STRING
 (2)  023444  000402                              BR      66$             ;;GET OVER THE ASCIZ
 (2)                                       ;;67$: .ASCIZ  <15><12>##
 (2)  023452                               66$:
 (1)  023452  000677                              BR      1$
 (1)
3642
3643                                       .SBTTL *       "B EVENT OUT" TEST
3644                                              ;*
3645                                              ;*THIS IS A SPECIAL SECTION DEVOTED FOR TESTING AND
3646                                              ;*PROVIDING SCOPE LOOP CAPABILITIES FOR "B EVENT OUT".
3647                                              ;*
3648                                              ;*WHEN YOU LOAD AND START AT LOCATION 230, PROGRAM
3649                                              ;*CONTROL IS TRANSFERRED HERE. "B EVENT OUT" PULSES ARE
3650                                              ;*GENERATED BY CLOCK B OVERFLOWS. PIN TT
3651                                              ;*("B EVENT OUT") IS WIRED TO PIN BB ("SCHMITT TRIG 2")
3652                                              ;*"SCHMITT TRIG 2" PULSES WILL SET CLOCK A'S CSR BIT 7.
3653                                              ;*IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING TECHNIQ.
3654                                              ;*AND ERROR SWITCH REGISTER OPTIONS ARE USED.
3655                                              ;*AN "*" IS TYPED AFTER EACH 65,324 LOOPS THROUGH THE TEST.
3656                                              ;*SW13=1 WILL INHIBIT THIS FEATURE.
3657                                              ;*
3658                                       ;*
 (1)                                       ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD BUFF B"
 (1)                                       ;*
3659                                              ;*YOU MUST WIRE PINS TT AND BB OF J1 TOGETHER AND INSTALL JUMPER W4.
3660                                              ;*
3661                                              ;*TEST LS210 SHOULD BE RUN FIRST.
3662                                              ;*
3663
3664  023454  005037  001664             LS230: CLR     .DVLS
3665
 (1)  023460  005037  001104             1$:    CLR     $ICNT           ;/CLEAR ITERATION COUNT
 (1)  023464  012737  177704  001206             MOV     #-60.,$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)  023472                          2$:                                ;/CLEAR CLOCK B.
  (1)  023472  005037  001354                 CLR     $TMDAT
  (2)
  (2)                                  ;*     MOV     $TMDAT,@ASR        ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
  (2)
  (2)                                  ;*     MOV     $TMDAT,@BSR        ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
  (1)
3666                                                                       ;PRELOAD CLOCK B'S BUFFER + COUNT REGS.
3667  023516  012737  177777  001354         MOV     #-1,$TMDAT
3668
  (1)                                  ;*     MOV     $TMDAT,@BBR        ;/ PUT DATA FROM $TMDAT TO DEVICE REG BBR
3669  023534  012737  001000  001354         MOV     #1000,$TMDAT
3670
  (1)                                  ;*     MOV     $TMDAT,@ASR        ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
3671
3672                                                                       ;RATE: 1MHZ, ENABLE COUNTER
3673
3674                                                                       ;HAS AN OVERFLOW OCCURRED?
3675  023552  012737  000003  001354         MOV     #3,$TMDAT
3676
  (1)                                  ;*     MOV     $TMDAT,@BSR        ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
3677  023570                          3$:
  (1)
  (1)                                  ;*     MOV     @BSR,$TMDAT        ;/READ DEVICE REG BSR,PUT DATA IN $TMDAT.
3678  023600  032737  000200  001354         BIT     #BIT07,$TMDAT
3679  023606  001770                        BEQ     3$                     ;NO -THEN WAIT FOR IT.
3680                                                                       ;OVERFLOW SHOULD GO OUT AS
3681                                                                       ;'B EVENT OUT''. YOU SHOULD SEE
3682                                                                       ;AN OUTPUT AT PIN TT.
3683                                                                       ;THIS IN TURN IS BROUGHT
3684                                                                       ;IN AS ''SCHMITT TRIG 2'' IN TO
3685                                                                       ;SET CLOCK A'S CSR BIT 7.
3686
3687
  (1)                                  ;*     MOV     @ASR,$BDDAT        ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
3688  023620  105737  001126                 TSTB    $BDDAT             ;DID BIT 7 SET
3689  023624  100401                         BMI     4$                 ;BR IF YES TO 4$
3690
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
3691  023626  104000                         ERROR                      ;ERROR 'B EVENT OUT'' PULSE
3692                                                                       ;NOT DETECTED. HAVE YOU
3693                                                                       ;WIRED IT RIGHT?
3694
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
3695  023630                          4$:
3696
  (1)  023630  032777  020000  155302         BIT     #BIT13,@SWR        ;/INHIBIT ''*'' TYPEOUT?
  (1)  023636  001315                        BNE     2$                 ;/YES - IGNORE ANY UPDATES.
  (1)
  (1)  023640  005237  001104                INC     $ICNT              ;/UPDATE COUNT.
  (1)  023644  001312                        BNE     2$                 ;/IF NOT DONE 65,324 TIMES,
  (1)                                                                      ;/DO IT AGAIN.
  (1)
  (2)  023646  104401  023654                TYPE    ,65$               ;;TYPE ASCIZ STRING
```

```
   (2)  023652  000401                          BR      64$             ;;GET OVER THE ASCIZ
   (2)                                ;;65$:    .ASCIZ  #*#
   (2)  023656                        64$:
   (1)
   (1)  023656  005237  001206                  INC     $PASS           ;/DONE 60 PASSES?
   (1)  023662  100703                          BMI     2$              ;/NO - NO NEED FOR CR,LF.
   (2)  023664  104401  023672                  TYPE    ,67$            ;;TYPE ASCIZ STRING
   (2)  023670  000402                          BR      66$             ;;GET OVER THE ASCIZ
   (2)                                ;;67$:    .ASCIZ  <15><12>##
   (2)  023676                        66$:
   (1)  023676  000670                          BR      1$
   (1)
  3697
  3698                                ;*ROUTINE TO HANDLE TRAPS TO LOC 4, 10 AND .
  3699                                ;*INTERRUPTS TO WRONG VECTORS.
  3700                                ;*.+2, IOTT(TRAPS) WERE PUT IN LOCATIONS 4-1000
  3701
  3702
  3703  023700                        IOTRD:
  3704  023700  011637  024050                  MOV     (R6),2$         ;GET WHERE WE TRAPPED TO.
  3705  023704  162737  000004  024050          SUB     #4,2$           ;=WHERE R6 RETURN 10-4
  3706  023712  104401  023720                  TYPE    ,65$            ;;TYPE ASCIZ STRING
   (1)  023716  000412                          BR      64$             ;;GET OVER THE ASCIZ
   (1)                                ;;65$:    .ASCIZ  <15><12>#ILLEGAL TRAP TO: #
   (1)  023744                        64$:
  3707
  3708  023744  013746  024050                  MOV     2$,-(SP)        ;;SAVE 2$ FOR TYPEOUT
   (1)  023750  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
  3709
  3710  023752  104401  023760                  TYPE    ,67$            ;;TYPE ASCIZ STRING
   (1)  023756  000407                          BR      66$             ;;GET OVER THE ASCIZ
   (1)                                ;;67$:    .ASCIZ  # FROM LOC.: #
   (1)  023776                        66$:
  3711
  3712  023776  062706  000004                  ADD     #4,R6           ;POINT TO WHERE WE TRAPPED FROM.
  3713
  3714  024002  011637  024052                  MOV     (R6),3$         ;PICK UP LOC
  3715  024006  162737  000002  024052          SUB     #2,3$           ;FROM REAL ADDR.
  3716  024014  013746  024052                  MOV     3$,-(SP)        ;;SAVE 3$ FOR TYPEOUT
   (1)  024020  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
  3717
  3718  024022  023727  024050  000004          CMP     2$,#4           ;DID WE TRAP TO LOC 4?
  3719  024030  001405                          BEQ     1$              ;IF SO - DON'T RETURN!
  3720  024032  023727  024050  000010          CMP     2$,#10          ;DID WE TRAP TO LOC. 10?
  3721  024040  001401                          BEQ     1$              ;ID SO - DON'T RETURN!
  3722  024042  000002                          RTI                     ;TRY RETURNING.
  3723
  3724
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

  3725  024044  000000                1$:       HALT                    ;WE STOPPED HERE BECAUSE WE TRAPPED
  3726  024046  000776                          BR      1$              ;TO LOC 4 OR LOC 10.  THIS IS A
  3727                                                                  ;FATAL CONDITION THAT WE CAN NOT
  3728                                                                  ;RECOVER FROM.
  3729
  3730
```

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

3731  024050  000000                  2$:     .WORD   0               ;USED BY IOTRP TO STORE WHERE WE TRAPPED TO.
3732  024052  000000                  3$:     .WORD   0               ;USED BY IOTRP TO STORE WHERE WE TRAPPED FROM.
3733                                          .SBTTL
3734                                          .SBTTL  *SYSMAC ROUTINES
3735                                          .SBTTL
3736
3737                                  .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)  024054  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 (1)  024060  116637  000001  024277          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 (1)  024066  112637  024301                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 (1)  024072  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 (1)  024076  000406                          BR      $TYPON
 (1)  024100  112737  000001  024277  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 (1)  024106  112737  000006  024301          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 (1)  024114  112737  000005  024276  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
 (1)  024122  010346                          MOV     R3,-(SP)        ;;SAVE R3
 (1)  024124  010446                          MOV     R4,-(SP)        ;;SAVE R4
 (1)  024126  010546                          MOV     R5,-(SP)        ;;SAVE R5
 (1)  024130  113704  024301                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 (1)  024134  005404                          NEG     R4
 (1)  024136  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 (1)  024142  110437  024300                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
 (1)  024146  113704  024277                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
 (1)  024152  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
 (1)  024156  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
 (1)  024160  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
 (1)  024162  000404                          BR      3$              ;;GO DO MSB
 (1)  024164  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
 (1)  024166  006105                          ROL     R5
 (1)  024170  006105                          ROL     R5
```

```
(1)  024172  010503                    MOV     R5,R3
(1)  024174  006103                3$:  ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  024176  105337  024300            DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  024202  100016                    BPL     7$              ;;BR IF NO
(1)  024204  042703  177770            BIC     #177770,R3      ;;GET RID OF JUNK
(1)  024210  001002                    BNE     4$              ;;TEST FOR 0
(1)  024212  005704                    TST     R4              ;;SUPPRESS THIS 0?
(1)  024214  001403                    BEQ     5$              ;;BR IF YES
(1)  024216  005204                4$:  INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  024220  052703  000060            BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  024224  052703  000040        5$: BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  024230  110337  024274            MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  024234  104401  024274            TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  024240  105337  024276        7$: DECB    $OCNT           ;;COUNT BY 1
(1)  024244  003347                    BGT     2$              ;;BR IF MORE TO DO
(1)  024246  002402                    BLT     6$              ;;BR IF DONE
(1)  024250  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  024252  000744                    BR      2$              ;;GO DO THE LAST DIGIT
(1)  024254  012605                6$: MOV     (SP)+,R5        ;;RESTORE R5
(1)  024256  012604                    MOV     (SP)+,R4        ;;RESTORE R4
(1)  024260  012603                    MOV     (SP)+,R3        ;;RESTORE R3
(1)  024262  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  024270  012616                    MOV     (SP)+,(SP)
(1)  024272  000002                    RTI                     ;;RETURN
(1)  024274     000                8$: .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  024275     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  024276     000                $OCNT:  .BYTE   0           ;;OCTAL DIGIT COUNTER
(1)  024277     000                $OFILL: .BYTE   0           ;;ZERO FILL SWITCH
(1)  024300  000000                $OMODE: .WORD   0           ;;NUMBER OF DIGITS TO TYPE
3738                               .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)  024302                        $ERROR:
(1)  024302  104407                    CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  024304  105237  001103        7$: INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  024310  001775                    BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  024312  013777  001102  154622    MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  024320  032777  002000  154612    BIT     #BIT10,@SWR     ;;BELL ON ERROR?
(1)  024326  001402                    BEQ     1$              ;;NO - SKIP
(1)  024330  104401  001170            TYPE    ,$BELL          ;;RING BELL
(1)  024334  005237  001112        1$: INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(1)  024340  011637  001116            MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  024344  162737  000002  001116    SUB     #2,$ERRPC
(1)  024352  117737  154540  001114    MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  024360  032777  020000  154552    BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
```

```
(1)  024366  001004                          BNE     20$             ;;SKIP TYPEOUTS
(1)  024370  004737  024502                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  024374  104401  001175                  TYPE    ,$CRLF
(1)  024400                          20$:
(1)  024400  122737  000001  001220          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  024406  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
(1)  024410  113737  001114  024422          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  024416  004737  026432                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(1)  024422     000                  21$:    .BYTE   0
(1)  024423     000                          .BYTE   0
(1)  024424  000777                  22$:    BR      22$             ;;APT ERROR LOOP
(1)  024426  005777  154506          2$:     TST     @SWR            ;;HALT ON ERROR
(1)  024432  100002                          BPL     3$              ;;SKIP IF CONTINUE
(1)  024434  000000                          HALT                    ;;HALT ON ERROR!
(1)  024436  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  024440  032777  001000  154472  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(1)  024446  001402                          BEQ     4$              ;;BR IF NO
(1)  024450  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  024454  005737  001166          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  024460  001402                          BEQ     5$              ;;BR IF NONE
(1)  024462  013716  001166                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  024466                          5$:
(1)  024466  022737  022414  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
(1)  024474  001001                          BNE     6$              ;;BRANCH IF NO
(1)  024476  000000                          HALT                    ;;YES
(1)  024500                          6$:
(1)  024500  000002                          RTI                     ;;RETURN
3739                                 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
(2)                                  ;;**************************************************************
(1)                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($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)  024502                          $ERRTYP:
(1)  024502  104401  001175                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  024506  010046                          MOV     R0,-(SP)        ;;SAVE R0
(1)  024510  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  024512  153700  001114                  BISB    @#$ITEMB,R0
(1)  024516  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                  ;;TYPE THE PC OF THE ERROR
(2)  024520  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                  ;;ERROR ADDRESS
(2)  024524  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  024526  000426                          BR      6$              ;;GET OUT
(1)  024530  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  024532  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
(1)  024534  006300                          ASL     R0
(1)  024536  006300                          ASL     R0
(1)  024540  062700  001356                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  024544  012037  024554                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  024550  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  024552  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  024554  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  024556  104401  001175                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  024562  012037  024572          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
```

```
(1)  024566  001404                    BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  024570  104401                    TYPE                    ;;TYPE THE "DATA HEADER"
(1)  024572  000000                4$:  .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(1)  024574  104401  001175            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  024600  011000                5$:  MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
(1)  024602  001004                    BNE     7$              ;;GO TYPE THE DATA
(1)  024604  012600                6$:  MOV     (SP)+,R0        ;;RESTORE R0
(1)  024606  104401  001175            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  024612  000207                    RTS     PC              ;;RETURN
(1)  024614                        7$:
(2)  024614  013046                    MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
(2)  024616  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  024620  005710                    TST     (R0)            ;;IS THERE ANOTHER NUMBER?
(1)  024622  001770                    BEQ     6$              ;;BR IF NO
(1)  024624  104401  024632            TYPE    ,8$             ;;TYPE TWO(2) SPACES
(1)  024630  000771                    BR      7$              ;;LOOP
(1)  024632  020040      000       8$:  .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          024636                    .EVEN
3740                               .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)  024636                        $TYPDS:
(3)  024636  010046                    MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  024640  010146                    MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  024642  010246                    MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  024644  010346                    MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  024646  010546                    MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  024650  012746  020200            MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  024654  016605  000020            MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  024660  100004                    BPL     1$              ;;BR IF INPUT IS POS.
(1)  024662  005405                    NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  024664  112766  000055  000001    MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  024672  005000                1$:  CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  024674  012703  025052            MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  024700  112723  000040            MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  024704  005002                2$:  CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  024706  016001  025042            MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  024712  160105                3$:  SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  024714  002402                    BLT     4$              ;;BR IF DONE
(1)  024716  005202                    INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  024720  000774                    BR      3$
(1)  024722  060105                4$:  ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  024724  005702                    TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  024726  001002                    BNE     5$              ;;FALL THROUGH IF 0
(1)  024730  105716                    TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  024732  100407                    BMI     7$              ;;BR IF YES
(1)  024734  106316                5$:  ASLB    (SP)            ;;MSD?
```

```
    (1)  024736  103003                        BCC     6$                ;;BR IF NO
    (1)  024740  116663  000001  177777        MOVB    1(SP),-1(R3)      ;;YES--SET THE SIGN
    (1)  024746  052702  000060        6$:     BIS     #'0,R2            ;;MAKE THE BCD DIGIT ASCII
    (1)  024752  052702  000040        7$:     BIS     #' ,R2            ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
    (1)  024756  110223                        MOVB    R2,(R3)+          ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
    (1)  024760  005720                        TST     (R0)+             ;;JUST INCREMENTING
    (1)  024762  020027  000010                CMP     R0,#10            ;;CHECK THE TABLE INDEX
    (1)  024766  002746                        BLT     2$                ;;GO DO THE NEXT DIGIT
    (1)  024770  003002                        BGT     8$                ;;GO TO EXIT
    (1)  024772  010502                        MOV     R5,R2             ;;GET THE LSD
    (1)  024774  000764                        BR      6$                ;;GO CHANGE TO ASCII
    (1)  024776  105726        8$:             TSTB    (SP)+             ;;WAS THE LSD THE FIRST NON-ZERO?
    (1)  025000  100003                        BPL     9$                ;;BR IF NO
    (1)  025002  116663  177777  177776        MOVB    -1(SP),-2(R3)     ;;YES--SET THE SIGN FOR TYPING
    (1)  025010  105013        9$:             CLRB    (R3)              ;;SET THE TERMINATOR
    (3)  025012  012605                        MOV     (SP)+,R5          ;;POP STACK INTO R5
    (3)  025014  012603                        MOV     (SP)+,R3          ;;POP STACK INTO R3
    (3)  025016  012602                        MOV     (SP)+,R2          ;;POP STACK INTO R2
    (3)  025020  012601                        MOV     (SP)+,R1          ;;POP STACK INTO R1
    (3)  025022  012600                        MOV     (SP)+,R0          ;;POP STACK INTO R0
    (1)  025024  104401  025052                TYPE    ,$DBLK            ;;NOW TYPE THE NUMBER
    (1)  025030  016666  000002  000004        MOV     2(SP),4(SP)       ;;ADJUST THE STACK
    (1)  025036  012616                        MOV     (SP)+,(SP)
    (1)  025040  000002                        RTI                       ;;RETURN TO USER
    (1)  025042  023420        $DTBL:  10000.
    (1)  025044  001750                        1000.
    (1)  025046  000144                        100.
    (1)  025050  000012                        10.
    (1)  025052  000004        $DBLK:  .BLKW   4
 3741                          .SBTTL  SCOPE HANDLER ROUTINE
    (1)
    (2)                        ;;**********************************************************************
    (1)                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
    (1)                        ;*AND LOAD THE TEST NUMBER($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)                        ;*SW09=1        LOOP ON ERROR
    (1)                        ;*SW08=1        LOOP ON TEST IN SWR<7:0>
    (1)                        ;*CALL
    (1)                        ;*      SCOPE           ;;SCOPE=IOT
    (1)
    (1)  025062                $SCOPE:
    (1)  025062  104407                        CKSWR                     ;;TEST  FOR CHANGE IN SOFT-SWR
    (1)  025064  032777  040000  154046  1$:     BIT     #BIT14,@SWR       ;;LOOP ON PRESENT TEST?
    (1)  025072  001062                        BNE     $OVER             ;;YES IF SW14=1
    (1)                        ;#####START OF CODE FOR THE XOR TESTER#####
    (1)  025074  000416        $XTSTR: BR      6$                        ;;IF RUNNING ON THE "XOR" TESTER CHANGE
    (1)                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
    (1)  025076  013746  000004                MOV     @#ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
    (1)  025102  012737  025122  000004        MOV     #5$,@#ERRVEC      ;;SET FOR TIMEOUT
    (1)  025110  005737  177060                TST     @#177060          ;;TIME OUT ON XOR?
    (1)  025114  012637  000004                MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
    (1)  025120  000431                        BR      $SVLAD            ;;GO TO THE NEXT TEST
    (1)  025122  022626        5$:             CMP     (SP)+,(SP)+       ;;CLEAR THE STACK AFTER A TIME OUT
    (1)  025124  012637  000004                MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
```

```
   (1)  025130  000417                          BR      7$              ;;LOOP ON THE PRESENT TEST
   (1)  025132                                6$:;#####END OF CODE FOR THE XOR TESTER#####
   (1)  025132  032777  000400  154000          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
   (1)  025140  001404                          BEQ     2$              ;;BR IF NO
   (1)  025142  127737  153772  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
   (1)  025150  001433                          BEQ     $OVER           ;;BR IF YES
   (1)  025152  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
   (1)  025156  001412                          BEQ     $SVLAD          ;;BR IF NO
   (1)  025160  032777  001000  153752          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
   (1)  025166  001404                          BEQ     4$              ;;BR IF NO
   (1)  025170  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
   (1)  025176  000420                          BR      $OVER
   (1)  025200  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
   (1)  025204  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
   (1)  025210  113737  001102  001204          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
   (1)  025216  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
   (1)  025222  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
   (1)  025226  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
   (1)  025232  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
   (1)  025240  013777  001102  153674  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
   (1)  025246  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
   (1)  025252  000002                          RTI                     ;;FIXES PS
  3742                                  .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)  025254  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
   (1)  025256  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
   (3)  025264  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   (3)  025266  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   (3)  025270  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   (1)  025272  104411                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
   (1)  025274  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
   (1)  025276  005001                          CLR     R1              ;;CLEAR DATA WORD
   (1)  025300  005002                          CLR     R2
   (1)  025302  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
   (1)  025304  001412                          BEQ     3$              ;;IF ZERO GET OUT
   (1)  025306  006301                          ASL     R1              ;;*2
   (1)  025310  006102                          ROL     R2
   (1)  025312  006301                          ASL     R1              ;;*4
   (1)  025314  006102                          ROL     R2
   (1)  025316  006301                          ASL     R1              ;;*8
   (1)  025320  006102                          ROL     R2
   (1)  025322  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
   (1)  025326  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
   (1)  025330  000764                          BR      2$              ;;LOOP
   (1)  025332  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
   (1)  025334  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
   (1)  025340  010237  025354                  MOV     R2,$HIOCT
   (3)  025344  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
(3)  025346  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  025350  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  025352  000002                        RTI                     ;;RETURN
(1)  025354  000000                $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
3743                               .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)  025356  022737  000176  001140 $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
(1)  025364  001074                        BNE     15$             ;;BRANCH IF NO
(1)  025366  105777  153552                TSTB    @$TKS           ;;CHAR THERE?
(1)  025372  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
(1)  025374  117746  153546                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
(1)  025400  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
(1)  025404  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
(1)  025410  001062                        BNE     15$             ;;NO, RETURN TO USER
(1)  025412  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
(1)  025420  001456                        BEQ     15$             ;;BRANCH IF YES
(1)
(1)  025422  104401  026103                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
(1)  025426  104401  026110        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
(2)  025432  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
(2)  025436  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  025440  104401  026121                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
(1)  025444  005046                19$:    CLR     -(SP)           ;;CLEAR COUNTER
(1)  025446  005046                        CLR     -(SP)           ;;THE NEW SWR
(1)  025450  105777  153470        7$:     TSTB    @$TKS           ;;CHAR THERE?
(1)  025454  100375                        BPL     7$              ;;IF NOT TRY AGAIN
(1)
(1)  025456  117746  153464                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
(1)  025462  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)
(1)
(1)
(1)  025466  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
(1)  025472  001005                        BNE     10$             ;;BRANCH IF NOT
(1)  025474  104401  026076                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
(1)  025500  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  025504  000757                        BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  025506  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
(1)  025512  001022                        BNE     16$             ;;BRANCH IF NO
(1)  025514  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  025520  001403                        BEQ     11$             ;;BRANCH IF YES
(1)  025522  016677  000002  153410        MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  025530  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
(1)  025534  104401  001175        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  025540  123727  001135  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)  025546  001003                        BNE     15$             ;;BRANCH IF NOT
```

```
(1)  025550  012777  000100  153366          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  025556  000002                  15$:     RTI                     ;;RETURN
(1)  025560  004737  026344          16$:     JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  025564  021627  000060                   CMP     (SP),#60        ;;CHAR < 0?
(1)  025570  002420                           BLT     18$             ;;BRANCH IF YES
(1)  025572  021627  000067                   CMP     (SP),#67        ;;CHAR > 7?
(1)  025576  003015                           BGT     18$             ;;BRANCH IF YES
(1)  025600  042726  000060                   BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  025604  005766  000002                   TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  025610  001403                           BEQ     17$             ;;BRANCH IF YES
(1)  025612  006316                           ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  025614  006316                           ASL     (SP)            ;;   CHAR OVER TO MAKE
(1)  025616  006316                           ASL     (SP)            ;;   ROOM FOR NEW ONE.
(1)  025620  005266  000002          17$:     INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  025624  056616  177776                   BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  025630  000707                           BR      7$              ;;GET THE NEXT ONE
(1)  025632  104401  001174          18$:     TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  025636  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)  025640  011646                 $RDCHR:  MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  025642  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  025650  105777  153270          1$:      TSTB    @$TKS           ;;WAIT FOR
(1)  025654  100375                           BPL     1$              ;;A CHARACTER
(1)  025656  117766  153264  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  025664  042766  17[illegible]  000004   BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  025672  0266[illegible]  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  025700  00[illegible]                    BNE     3$              ;;BRANCH IF NO
(1)  025702  1[illegible]            2$:      TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  025706  [illegible]                      BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  025710                                   MOVB    @$TKB,-(SP)     ;;GET CHARACTER
(1)  025714  [illegible]                      BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  025720  0[illegible]                     CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  025724  001[illegible]                   BNE     2$              ;;IF NOT DISCARD IT
(1)  025726  000750                           BR      1$              ;;YES, RESUME
(1)  025730  026627  000004  000140  3$:      CMP     4(SP),#140      ;;IS IT UPPER CASE?
(1)  025736  002407                           BLT     4$              ;;BRANCH IF YES
(1)  025740  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
(1)  025746  003003                           BGT     4$              ;;BRANCH IF YES
(1)  025750  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
(1)  025756  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)  025760  010346                    $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  025762  012703  026066            1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  025766  022703  026076            2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
(1)  025772  101405                            BLOS    4$              ;;BR IF YES
(1)  025774  104410                            RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  025776  112613                            MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  026000  122713  000177            10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  026004  001003                            BNE     3$              ;;SKIP IF NOT
(1)  026006  104401  001174            4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  026012  000763                            BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  026014  111337  026064            3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  026020  104401  026064                    TYPE    ,9$
(1)  026024  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  026030  001356                            BNE     2$              ;;LOOP IF NOT RETURN
(1)  026032  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  026036  104401  001176                    TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  026042  012603                            MOV     (SP)+,R3        ;;RESTORE R3
(1)  026044  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  026046  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  026054  012766  026066  000004            MOV     #$TTYIN,4(SP)
(1)  026062  000002                            RTI                     ;;RETURN
(1)  026064     000            9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  026065     000                    .BYTE   0               ;;TERMINATOR
(1)  026066  000010            $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
(1)  026076  052536  005015  000  $CNTLU: .ASCIZ  /^U/<15><12>  ;;CONTROL "U"
(1)  026103     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>  ;;CONTROL "G"
(1)  026110  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  026116  020075     000
(1)  026121     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
(1)  026126  036440  000040
3744                                .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)  026132  105737  001157            $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  026136  100002                            BPL     1$              ;;BR IF YES
(1)  026140  000000                            HALT                    ;;HALT HERE IF NO TERMINAL
(1)  026142  000430                            BR      3$              ;;LEAVE
(1)  026144  010046                    1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  026146  017600  000002                    MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  026152  122737  000001  001220            CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  026160  001011                            BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
```

```
(1)  026162  132737  000100  001221          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(1)  026170  001405                           BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  026172  010037  026202                   MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  026176  004737  026422                   JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(1)  026202  000000                  61$:     .WORD   0               ;;MESSAGE ADDRESS
(1)  026204  132737  000040  001221  62$:     BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  026212  001003                           BNE     60$             ;;YES,SKIP TYPE OUT
(1)  026214  112046                  2$:      MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  026216  001005                           BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  026220  005726                           TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  026222  012600                  60$:     MOV     (SP)+,R0        ;;RESTORE R0
(1)  026224  062716  000002          3$:      ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  026230  000002                           RTI                     ;;RETURN
(1)  026232  122716  000011          4$:      CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  026236  001430                           BEQ     8$
(1)  026240  122716  000200                   CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  026244  001006                           BNE     5$
(1)  026246  005726                           TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  026250  104401                           TYPE                    ;;TYPE A CR AND LF
(1)  026252  001175                           $CRLF
(1)  026254  105037  026410                   CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  026260  000755                           BR      2$              ;;GET NEXT CHARACTER
(1)  026262  004737  026344          5$:      JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  026266  123726  001156          6$:      CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  026272  001350                           BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  026274  013746  001154                   MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                           ;;AND THE NULL CHAR.
(1)  026300  105366  000001          7$:      DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  026304  002770                           BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  026306  004737  026344                   JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  026312  105337  026410                   DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  026316  000770                           BR      7$              ;;LOOP
(1)
(1)                                   ;HORIZONTAL TAB PROCESSOR
(1)
(1)  026320  112716  000040          8$:      MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  026324  004737  026344          9$:      JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  026330  132737  000007  026410          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  026336  001372                           BNE     9$              ;;TAB STOP
(1)  026340  005726                           TST     (SP)+           ;;POP SPACE OFF STACK
(1)  026342  000724                           BR      2$              ;;GET NEXT CHARACTER
(1)  026344  105777  152600          $TYPEC:  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  026350  100375                           BPL     $TYPEC
(1)  026352  116677  000002  152572          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  026360  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  026366  001003                           BNE     1$              ;;BRANCH IF NO
(1)  026370  105037  026410                   CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  026374  000406                           BR      $TYPEX          ;;EXIT
(1)  026376  122766  000012  000002  1$:      CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  026404  001402                           BEQ     $TYPEX          ;;BRANCH IF YES
(1)  026406  105227                           INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  026410  000000                  $CHARCNT:.WORD   0               ;;CHARACTER COUNT STORAGE
(1)  026412  000207                  $TYPEX:  RTS     PC
(1)
3745                                  .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
```

```
    (2)                                 ;;*****************************************************************
    (1)  026414  112737  000001  026660 $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
    (1)  026422  112737  000001  026656 $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
    (1)  026430  000403                        BR      $ATYC
    (1)  026432  112737  000001  026660 $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
    (1)  026440                        $ATYC:
    (3)  026440  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
    (3)  026442  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
    (1)  026444  105737  026656                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
    (1)  026450  001450                        BEQ     5$              ;;IF NOT:  BR
    (1)  026452  122737  000001  001220        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
    (1)  026460  001031                        BNE     3$              ;;IF NOT:  BR
    (1)  026462  132737  000100  001221        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
    (1)  026470  001425                        BEQ     3$              ;;IF NOT:  BR
    (1)  026472  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
    (1)  026476  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
    (1)  026504  005737  001200        1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
    (1)  026510  001375                        BNE     1$              ;;IF NOT:  WAIT
    (1)  026512  010037  001214                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
    (1)  026516  105720                2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
    (1)  026520  001376                        BNE     2$
    (1)  026522  163700  001214                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
    (1)  026526  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
    (1)  026530  010037  001216                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
    (1)  026534  012737  000004  001200        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
    (1)  026542  000413                        BR      5$
    (1)  026544  017637  000004  026570 3$:    MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
    (1)  026552  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
    (3)  026560  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
    (1)  026564  004737  026132                JSR     PC,$TYPE        ;;CALL TYPE MACRO
    (1)  026570  000000                4$:     .WORD   0
    (1)  026572                        5$:
    (1)  026572  105737  026660        10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
    (1)  026576  001416                        BEQ     12$             ;;IF NOT:  BR
    (1)  026600  005737  001220                TST     $ENV            ;;RUNNING UNDER APT?
    (1)  026604  001413                        BEQ     12$             ;;IF NOT:  BR
    (1)  026606  005737  001200        11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
    (1)  026612  001375                        BNE     11$             ;;IF NOT:  WAIT
    (1)  026614  017637  000004  001202        MOV     @4(SP),$FATAL   ;;GET ERROR #
    (1)  026622  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
    (1)  026630  005237  001200                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
    (1)  026634  105037  026660        12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
    (1)  026640  105037  026657                CLRB    $LFLG           ;;CLEAR LOG FLAG
    (1)  026644  105037  026656                CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
    (3)  026650  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
    (3)  026652  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
    (1)  026654  000207                        RTS     PC              ;;RETURN
    (1)  026656     000        $MFLG:  .BYTE   0               ;;MESSG. FLAG
    (1)  026657     000        $LFLG:  .BYTE   0               ;;LOG FLAG
    (1)  026660     000        $FFLG:  .BYTE   0               ;;FATAL FLAG
    (1)          026662                .EVEN
    (1)          000200        APTSIZE=200
    (1)          000001        APTENV=001
    (1)          000100        APTSPOOL=100
    (1)          000040        APTCSUP=040
  3746                          .SBTTL  POWER DOWN AND UP ROUTINES
```

```
(1)
(2)                                     ;;*****************************************************************
(1)                                     ;POWER DOWN ROUTINE
(1)  026662  012737  027026  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1)  026670  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
(3)  026676  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  026700  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  026702  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  026704  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  026706  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  026710  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(3)  026712  017746  152222                 MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
(1)  026716  010637  027032                 MOV     SP,$SAVR6       ;;SAVE SP
(1)  026722  012737  026734  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(1)  026730  000000                         HALT
(1)  026732  000776                         BR      .-2             ;;HANG UP
(1)
(2)                                     ;;*****************************************************************
(1)                                     ;POWER UP ROUTINE
(1)  026734  012737  027026  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1)  026742  013706  027032                 MOV     $SAVR6,SP       ;;GET SP
(1)  026746  005037  027032                 CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
(1)  026752  005237  027032          1$:    INC     $SAVR6          ;;WAIT FOR THE INC
(1)  026756  001375                         BNE     1$              ;;OF  WORD
(3)  026760  012677  152154                 MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
(3)  026764  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  026766  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  026770  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  026772  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  026774  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  026776  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  027000  012737  026662  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)  027006  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
(1)  027014  104401                         TYPE                    ;;REPORT THE POWER FAILURE
(1)  027016  027034                $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
(1)  027020  012716                         MOV     (PC)+,(SP)      ;;RESTART AT START
(1)  027022  001726                $PWRAD: .WORD   START           ;;RESTART ADDRESS
(1)  027024  000002                         RTI
(1)  027026  000000                $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
(1)  027030  000776                         BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(1)  027032  000000                $SAVR6: 0                       ;;PUT THE SP HERE
3747 027034      015      012       PWRMSG: .BYTE   15,12
3748 027036  042522  052123  051101         .ASCII  /RESTARTING AFTER A POWER FAILURE/
     027044  044524  043516  040440
     027052  052106  051105  040440
     027060  050040  053517  051105
     027066  043040  044501  052514
     027074  042522
3749 027076      015      012      012       .BYTE   15,12,12,0
     027101      000
3750                                         .EVEN
3751                                .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)  027102  010046                     $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
(1)  027104  016600  000002                     MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  027110  005740                             TST     -(R0)           ;;BACKUP BY 2
(1)  027112  111000                             MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  027114  006300                             ASL     R0              ;;POSITION FOR INDEXING
(1)  027116  016000  027136                     MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  027122  000200                             RTS     R0              ;;GO TO ROUTINE
(1)
(1)
(1)                                     ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(1)
(1)  027124  011646                     $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
(1)  027126  016666  000004  000002             MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
(1)  027134  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)  027136  027124                     $TRPAD: .WORD   $TRAP2
(3)  027140  026132                             $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
(3)  027142  024100                             $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)  027144  024054                             $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)  027146  024114                             $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)  027150  024636                             $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
(1)
(3)  027152  025426                             $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
(1)
(3)  027154  025356                             $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
(3)  027156  025640                             $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
(3)  027160  025760                             $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
(3)  027162  025254                             $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
3752 027164  023700                             IOTRD   ;;CALL=IOTT     TRAP+13(104413)
3753 027166  005015  046103  041517  EM1:      .ASCIZ  <15><12>/CLOCKA SR FUNCTION ERROR/
     027174  040513  051440  020122
     027202  052506  041516  044524
     027210  047117  042440  051122
     027216  051117    000
3754 027221     015  041412  047514  EM2:      .ASCIZ  <15><12>/CLOCKA SR DATA ERROR/
     027226  045503  020101  051123
     027234  042040  052101  020101
     027242  051105  047522  000122
3755 027250  005015  046103  041517  EM3:      .ASCIZ  <15><12>/CLOCKA BR DATA ERROR/
     027256  040513  041040  020122
     027264  040504  040524  042440
     027272  051122  051117    000
3756 027277     015  041412  047514  EM4:      .ASCIZ  <15><12>/CLOCKA CR DATA ERROR/
     027304  045503  020101  051103
     027312  042040  052101  020101
     027320  051105  047522  000122
```

```
 3757  027326  005015  046103  041517  EM5:    .ASCIZ  <15><12>/CLOCKB SR DATA ERROR/
       027334  041113  051440  020122
       027342  040504  040524  042440
       027350  051122  051117     000
 3758  027355     015  041412  047514  EM6:    .ASCIZ  <15><12>/CLOCKB BR DATA ERROR/
       027362  045503  020102  051102
       027370  042040  052101  020101
       027376  051105  047522  000122
 3759  027404  005015  046103  041517  EM7:    .ASCIZ  <15><12>/CLOCKB CR DATA ERROR/
       027412  041113  041440  020122
       027420  040504  040524  042440
       027426  051122  051117     000
 3760  027433     015  042012  040525  EM10:   .ASCIZ  <15><12>/DUAL ADDRESS ERROR/
       027440  020114  042101  051104
       027446  051505  020123  051105
       027454  047522  000122
 3761  027460  005015  046103  041517  EM11:   .ASCIZ  <15><12>#CLOCK A COUNT ERROR #
       027466  020113  020101  047503
       027474  047125  020124  051105
       027502  047522  020122     000
 3762  027507     015  041412  047514  EM12:   .ASCIZ  <15><12>#CLOCK A COUNT FUNCTION ERROR #
       027514  045503  040440  041440
       027522  052517  052116  043040
       027530  047125  052103  047511
       027536  020116  051105  047522
       027544  020122     000
 3763  027547     015  041412  047514  EM14:   .ASCIZ  <15><12>#CLOCK B COUNT FUNCTION ERROR #
       027554  045503  041040  041440
       027562  052517  052116  043040
       027570  047125  052103  047511
       027576  020116  051105  047522
       027604  020122     000
 3764  027607     015  041412  047514  EM15:   .ASCIZ  <15><12>#CLOCK B COUNT ERROR #
       027614  045503  041040  041440
       027622  052517  052116  042440
       027630  051122  051117  000040
 3765  027636  005015  046103  041517  EM16:   .ASCIZ  <15><12>#CLOCK A INTERRUPT ERROR #
       027644  020113  020101  047111
       027652  042524  051122  050125
       027660  020124  051105  047522
       027666  020122     000
 3766  027671     015  041412  047514  EM17:   .ASCIZ  <15><12>#CLOCK B INTERRUPT ERROR #
       027676  045503  041040  044440
       027704  052116  051105  052522
       027712  052120  042440  051122
       027720  051117  000040
 3767  027724  005015  046103  041517  EM20:   .ASCIZ  <15><12>#CLOCK A REPEATABILITY ERROR #
       027732  020113  020101  042522
       027740  042520  052101  041101
       027746  046111  052111  020131
       027754  051105  047522  020122
       027762     000
 3768  027763     015  041412  047514  EM23:   .ASCIZ  <15><12>#CLOCK B REPEATABILITY ERROR #
       027770  045503  041040  051040
       027776  050105  040505  040524
       030004  044502  044514  054524
```

```
                030012  042440  051122  051117
                030020  000040
  3769          030022  005015  046103  041517  EM26:   .ASCIZ  <15><12>#CLOCK ADDRESSING ERROR#
                030030  020113  042101  051104
                030036  051505  044523  043516
                030044  042440  051122  051117
                030052     000
  3770
  3771          030053     015  042412  051122  DH1:    .ASCIZ  <15><12>#ERRPC   ASR     WAS     S/B#
                030060  041520  020040  040440
                030066  051123  020040  020040
                030074  053440  051501  020040
                030102  020040  051440  041057
                030110     000
  3772          030111     015  042412  051122  DH3:    .ASCIZ  <15><12>#ERRPC   ABR     WAS     S/B#
                030116  041520  020040  040440
                030124  051102  020040  020040
                030132  053440  051501  020040
                030140  020040  051440  041057
                030146     000
  3773          030147     015  042412  051122  DH4:    .ASCIZ  <15><12>#ERRPC   ACR     WAS     S/B#
                030154  041520  020040  040440
                030162  051103  020040  020040
                030170  053440  051501  020040
                030176  020040  051440  041057
                030204     000
  3774          030205     015  042412  051122  DH5:    .ASCIZ  <15><12>#ERRPC   BSR     WAS     S/B#
                030212  041520  020040  041040
                030220  051123  020040  020040
                030226  053440  051501  020040
                030234  020040  051440  041057
                030242     000
  3775          030243     015  042412  051122  DH6:    .ASCIZ  <15><12>#ERRPC   BBR     WAS     S/B#
                030250  041520  020040  041040
                030256  051102  020040  020040
                030264  053440  051501  020040
                030272  020040  051440  041057
                030300     000
  3776          030301     015  042412  051122  DH7:    .ASCIZ  <15><12>#ERRPC   BCR     WAS     S/B#
                030306  041520  020040  041040
                030314  051103  020040  020040
                030322  053440  051501  020040
                030330  020040  051440  041057
                030336     000
  3777          030337     015  042412  051122  DH10:   .ASCII  <15><12>/ERROR   GOOD    BAD     GOOD    DATA READ FROM/
                030344  051117  020040  043440
                030352  047517  020104  020040
                030360  041040  042101  020040
                030366  020040  043440  047517
                030374  020104  020040  042040
                030402  052101  020101  042522
                030410  042101  043040  047522
                030416     115
  3778          030417     015  020012  050040          .ASCIZ  <15><12>/  PC    ADDR    ADDR    DATA    DUAL ADDRESS/
                030424  020103  020040  040440
                030432  042104  020122  020040
```

```
                030440  040440  042104  020122
                030446  020040  042040  052101
                030454  020101  020040  042040
                030462  040525  020114  042101
                030470  051104  051505  000123
3779            030476  005015  051105  050122  DH12:   .ASCIZ  <15><12>#ERRPC   ASR #
                030504  020103  020040  051501
                030512  020122     000
3780            030515     015  042412  051122  DH14:   .ASCIZ  <15><12>#ERRPC   BSR#
                030522  041520  020040  041040
                030530  051123     000
3781            030533     015  042412  051122  DH20:   .ASCIZ  <15><12>#ERRPC   ASR     2NDCNT  1STCNT #
                030540  041520  020040  040440
                030546  051123  020040  020040
                030554  031040  042116  047103
                030562  020124  030440  052123
                030570  047103  020124     000
3782            030575     015  042412  051122  DH23:   .ASCIZ  <15><12>#ERRPC   BSR     2NDCNT  1STCNT #
                030602  041520  020040  041040
                030610  051123  020040  020040
                030616  031040  042116  047103
                030624  020124  030440  052123
                030632  047103  020124     000
3783            030637     015  042412  051122  DH26:   .ASCIZ  <15><12>#ERRPC   CLOCK ADDR.#
                030644  041520  020040  041440
                030652  047514  045503  040440
                030660  042104  027122     000
3784
3785                    030666                  .EVEN
3786
3787            030666  001116  001324  001126  DT1:    .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,0
                030674  001124  000000
3788            030700  001116  001326  001126  DT3:    .WORD   $ERRPC,ABR,$BDDAT,$GDDAT,0
                030706  001124  000000
3789            030712  001116  001330  001126  DT4:    .WORD   $ERRPC,ACR,$BDDAT,$GDDAT,0
                030720  001124  000000
3790            030724  001116  001332  001126  DT5:    .WORD   $ERRPC,BSR,$BDDAT,$GDDAT,0
                030732  001124  000000
3791            030736  001116  001334  001126  DT6:    .WORD   $ERRPC,BBR,$BDDAT,$GDDAT,0
                030744  001124  000000
3792            030750  001116  001336  001126  DT7:    .WORD   $ERRPC,BCR,$BDDAT,$GDDAT,0
                030756  001124  000000
3793            030762  001116  001120  001122  DT10:   .WORD   $ERRPC,$GDADR,$BDADR,$GDDAT,$BDDAT,0
                030770  001124  001126  000000
3794            030776  001116  001324  000000  DT12:   .WORD   $ERRPC,ASR,0
3795            031004  001116  001332  000000  DT14:   .WORD   $ERRPC,BSR,0
3796            031012  001116  001330  001124  DT21:   .WORD   $ERRPC,ACR,$GDDAT,$TMP0,0
                031020  001164  000000
3797            031024  001116  001330  001126  DT22:   .WORD   $ERRPC,ACR,$BDDAT,$TMP0,0
                031032  001164  000000
3798            031036  001116  001336  001124  DT24:   .WORD   $ERRPC,BCR,$GDDAT,$TMP0,0
                031044  001164  000000
3799            031050  001116  001336  001126  DT25:   .WORD   $ERRPC,BCR,$BDDAT,$TMP0,0
                031056  001164  000000
3800            031062  001116  001164  000000  DT26:   .WORD   $ERRPC,$TMP0,0
3801
```

```
3802  031C70  000000  000000          DF0:    .WORD   0,0
3803
3804
3805
 (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)  031074                          $LPAI:
 (2)  031074  013746  000004                  MOV     4,-(SP)
 (2)
 (2)  031100  000413                          BR      31$             ;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)  031102  012737  031126  000004          MOV     #30$,4
 (2)  031110  005237  170000                  INC     170000
 (3)  031114  104401  031122                  TYPE    ,65$             ;;TYPE ASCIZ STRING
 (3)  031120  000401                          BR      64$              ;;GET OVER THE ASCIZ
 (3)                                  ;;65$:  .ASCIZ  <7>##
 (3)  031124                          64$:
 (2)  031124  000401                          BR      31$
 (2)  031126  022626                  30$:    CMP     (SP)+,(SP)+
 (2)  031130  012637  000004          31$:    MOV     (SP)+,4          ;ALL THIS JUNK MUST BE REMOVED!!
 (2)  031134  005037  031562                  CLR     $AERR
 (2)  031140  004537  031564                  JSR     R5,$LOAD         ;LOAD MICRO-CODE.
 (2)  031144  000000G                         .WORD   DRLPX2           ;FILE "DRLPX2.OBJ"
 (2)
 (2)  031146  052777  040000  150462          BIS     #BIT14,@KMADO    ;ISSUE KMC+DMC INIT.
 (2)
 (2)  031154                          1$:
 (2)                                                                  ;"HANGS" HERE THEN KMC-11 ERROR.
 (2)  031154  010146                          MOV     R1,-(SP)
 (2)  031156  005001                          CLR     R1
 (2)  031160  005201                  2$:     INC     R1               ;STALL FOR DMC-UP
```

```
(2)  031162  001376                        BNE     2$
(2)  031164  012777  104000  150444        MOV     #BIT15!BIT11,@KMAD0     ;SET RUN, AND ENABLE ARBITRATION.
(2)  031172  105201                25$:     INCB    R1
(2)  031174  001376                        BNE     25$
(2)
(2)  031176  032777  000040  150432        BIT     #BIT5,@KMAD0    ;SLAVE READY? (READING IPBM SR)
(2)  031204  001401                        BEQ     3$
(2)                                                                 ;FATAL LPA-11 ERROR SLAVE NOT READY.
(2)  031206  104000                        ERROR
(2)
(2)  031210  012777  000004  150424  3$:   MOV     #4,@KMAD2       ;READ FAST PATH
(2)  031216                        4$:
(3)  031216  004537  032474                JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  031222  104000                        ERROR                   ;/TIME-OUT ERROR
(3)                                                                 ;/WE FAILED TO COMPLETE
(3)                                                                 ;/CURRENT OPERATION.
(3)                                                                 ;/CONTINUES IN THIS LOOP
(3)                                                                 ;/WOULD MAKE US ''HANG'' HERE
(3)
(3)  031224  000774                        BR              4$
(3)
(3)                                                                 ;/RETURNS HERE-FROM-TIMED OUT.
(2)  031226  122777  000377  150406        CMPB    #377,@KMAD2     ;WAIT TILL KMC DONE COMMAND.
(2)  031234  001370                        BNE     4$
(2)  031236  122777  000377  150402        CMPB    #377,@KMAD4     ;IF FAST PATH=377 THEN ERROR.
(2)  031244  001001                        BNE     35$
(2)  031246  104000                        ERROR                   ;IPBM ERROR (SLAVE SIDE)
(2)                                                                 ;YOU MUST RUN IPBM DIAGNOSTIC.
(2)
(2)  031250  117737  150372  031530  35$:   MOVB    @KMAD4,11$      ;GET THE VERSION NUMBER FROM DMC-11
(2)  031256  005227  177777                INC     #-1
(2)  031262  001045                        BNE     5$
(2)  031264  005227  177777                INC     #-1
(2)  031270  001042                        BNE     5$
(3)  031272  104401  031300                TYPE    ,67$            ;;TYPE ASCIZ STRING
(3)  031276  000426                        BR      66$             ;;GET OVER THE ASCIZ
(3)                                ;;67$:  .ASCIZ  <200>'M8200-YC (DMC) MICROCODE VERSION NUMBER = ''
(3)  031354                        66$:
(2)  031354  013746  031530                MOV     11$,-(SP)
(2)  031360  104403                        TYPOS
(2)  031362     002     000                .BYTE   2,0
(3)  031364  104401  031372                TYPE    ,69$            ;;TYPE ASCIZ STRING
(3)  031370  000402                        BR      68$             ;;GET OVER THE ASCIZ
(3)                                ;;69$:  .ASCIZ  <200>''  ''
(3)  031376                        68$:
(2)
(2)  031376  112737  177777  031530  5$:    MOVB    #0-1,11$        ;DAC CODE FOR SLAVE.
(2)  031404  012501                        MOV     (5)+,R1         ;GET NEXT DEVICE ADDR.
(2)  031406  021127  000000        6$:     CMP     (R1),#0         ;TERM REACHED?
(2)  031412  001444                        BEQ     10$
(2)  031414  105237  031530                INCB    11$
(2)  031420  113777  031530  150220        MOVB    11$,@KMAD4      ;FIFO DATA
(2)  031426  004737  031532                JSR     PC,20$          ;ISSUE SEND
(2)  031432  112177  150210                MOVB    (R1)+,@KMAD4    ;SEND LOW BYTE OF DEVICE ADDR TO SLAVE.
(2)  031436  004737  031532                JSR     PC,20$          ;ISSUE SEND
```

```
(2)  031442  112177  150200              MOVB    (R1)+,@KMAD4    ;SEND HIGH BYTE OF DEVICE ADDR. TO SLAVE.
(2)  031446  004737  031532              JSR     PC,20$
(2)
(2)  031452  032777  000002  150156  7$: BIT     #BIT1,@KMAD0    ;WAIT FOR FIFO DATA
(2)  031460  001374                      BNE     7$              ;=1 NO DATA. =0 DATA.
(2)  031462  112777  000002  150152      MOVB    #2,@KMAD2       ;READ FIFO.
(2)
(2)  031470                         8$:
(3)  031470  004537  032474              JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  031474  104000                      ERROR                   ;/TIME-OUT ERROR
(3)                                                              ;/WE FAILED TO COMPLETE
(3)                                                              ;/CURRENT OPERATION.
(3)                                                              ;/CONTINUES IN THIS LOOP
(3)                                                              ;/WOULD MAKE US "HANG" HERE
(3)
(3)  031476  000774                      BR              8$
(3)
(3)                                                              ;/RETURNS HERE-FROM-TIMED OUT.
(2)  031500  122777  000377  150134      CMPB    #377,@KMAD2     ;WAIT FOR READ.
(2)  031506  001370                      BNE     8$
(2)  031510  105777  150132              TSTB    @KMAD4          ;WAS A ZERO RETURNED?
(2)  031514  001734                      BEQ     6$              ;YES GET NEXT ADDR.
(2)                                                              ;SLAVE WILL RETURN CODE 0 IF
(2)  031516  005237  031562              INC     $AERR           ;DEV PRESENT.   ELSE
(2)                                                              ;EXIT $AERR=1 IF SLAVE GIVES ERROR.
(2)  031522  005041                      CLR     -(1)            ;GET RID OF REFERENCE TO BAD ADDR.
(2)  031524  012601                 10$: MOV     (SP)+,R1
(2)  031526  000205                      RTS     R5              ;RETURN ALL ADDR. CHECKED.
(2)
(2)  031530  000000                 11$: .WORD   0               ;HOLDS DAC CODE PLUS OFFSET
(2)                                                              ;TO SLAVES ADDR. TABLE.
(2)
(2)  031532  112777  000003  150102  20$: MOVB    #3,@KMAD2       ;ISSUE FIFO WRITE
(2)  031540                         21$:
(3)  031540  004537  032474              JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  031544  104000                      ERROR                   ;/TIME-OUT ERROR
(3)                                                              ;/WE FAILED TO COMPLETE
(3)                                                              ;/CURRENT OPERATION.
(3)                                                              ;/CONTINUES IN THIS LOOP
(3)                                                              ;/WOULD MAKE US "HANG" HERE
(3)
(3)  031546  000774                      BR              21$
(3)
(3)                                                              ;/RETURNS HERE-FROM-TIMED OUT.
(2)  031550  122777  000377  150064      CMPB    #377,@KMAD2     ;KMC CODE WILL RETURN A "377"
(2)  031556  001370                      BNE     21$             ;WHEN DONE COMMAND.
(2)  031560  000207                      RTS     PC
(2)
(2)  031562  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)  031564  010446                 $LOAD:  MOV     R4,-(SP)        ;SAVE R4.
(2)  031566  010046                         MOV     R0,-(SP)        ;SAVE R0.
(2)  031570  012500                 1$:     MOV     (5)+,R0         ;GET PROG. ADDR.
(2)  031572  005077  150040                 CLR     @KMAD0          ;CLEAR CSR
(2)  031576  005077  150044                 CLR     @KMAD4          ;CLEAR CRAM ADDR.
(2)  031602  052777  002000  150026  2$:     BIS     #2000,@KMAD0    ;SELECT CRAM.
(2)  031610  012077  150036                 MOV     (0)+,@KMAD6     ;WRITE DATA.
(2)  031614  052777  020000  150014         BIS     #20000,@KMAD0   ;SET CRAM WRITE
(2)  031622  005077  150010                 CLR     @KMAD0          ;DISABLE CRAM.
(2)  031626  005277  150014                 INC     @KMAD4          ;UPDATE CRAM ADDR.
(2)  031632  021027  177777                 CMP     (0),#-1         ;ALL DONE?
(2)  031636  001361                         BNE     2$              ;NO LOOP.
(2)  031640  005077  150002                 CLR     @KMAD4          ;CLEAR CRAM ADDR.
(2)  031644  016500  177776                 MOV     -2(5),R0        ;GET MICRO CODE ADDR.
(2)
(2)  031650  052777  002000  147760  3$:     BIS     #2000,@KMAD0    ;SELECT CRAM
(2)  031656  022077  147770                 CMP     (R0)+,@KMAD6    ;DATA OK?
(2)  031662  001013                         BNE     5$              ;NO - REPORT AN ERROR.
(2)  031664  021027  177777                 CMP     (0),#-1         ;ALL DONE?
(2)  031670  001405                         BEQ     4$              ;YES - EXIT
(2)  031672  005077  147740                 CLR     @KMAD0          ;NO - DESELECT CRAM.
(2)  031676  005277  147744                 INC     @KMAD4          ;UPDATE CRAM ADDR.
(2)  031702  000762                         BR      3$
(2)
(2)  031704  012600                 4$:     MOV     (SP)+,R0        ;RESTORE R0
(2)  031706  012604                         MOV     (SP)+,R4        ;RESTORE R4
(2)  031710  000205                         RTS     R5              ;EXIT
(2)
(2)  031712                         5$:                             ;COME HERE ON LOAD ERROR
(2)  031712  005745                         TST     -(5)
(2)  031714  105204                         INCB    R4              ;UPDATE ERROR COUNTER.
(2)  031716  100324                         BPL     1$              ;IF NOT TOO MANY, TRY AGAIN.
(2)  031720  000000                         HALT                    ;MICRO CODE LOAD ERROR.
(2)                                                                 ;KMC-11 FAULT. YOU COULD TRY
(2)  031722  000722                         BR      1$              ;TO PRESS CONTINUE TO GIVE IT
(2)                                                                 ;ANOTHER CHANCE, BUT I DOUBT
(2)                                                                 ;THAI 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)  031724  010046                 $TLKW:  MOV     R0,-(SP)        ;SAVE R0
(2)  031726  012500                         MOV     (5)+,R0         ;GET DEVICE OFFSET
(2)  031730  052700  000340                 BIS     #340,R0         ;ADD WRITE CODE.
(2)  031734  004737  032206                 JSR     PC,$LPW         ;WAIT FOR FAST PATH READY
```

```
(2)  031740  010037  032032                MOV     R0,W1
(2)  031744  010077  147676                MOV     R0,@KMAD4
(2)  031750  112777  000005  147664        MOVB    #5,@KMAD2       ;ISSUE FAST PATH WRITE
(2)  031756  004737  032206                JSR     PC,$LPW         ;WAIT FOR RDY
(2)  031762  011537  032034                MOV     (5),W2
(2)  031766  112577  147654                MOVB    (5)+,@KMAD4     ;WRITE LOW BYTE DATA.
(2)
(2)  031772  112777  000005  147642        MOVB    #5,@KMAD2       ;FP WRITE
(2)  032000  004737  032206                JSR     PC,$LPW
(2)  032004  111537  032036                MOVB    (5),W3
(2)  032010  112577  147632                MOVB    (5)+,@KMAD4     ;WRITE HIGH BYTE
(2)  032014  112777  000005  147620        MOVB    #5,@KMAD2
(2)  032022  004737  032206                JSR     PC,$LPW
(2)  032026  012600                        MOV     (SP)+,R0
(2)  032030  000205                        RTS     R5              ;EXIT DONE.
(2)  032032  000000                W1:     0
(2)  032034  000000                W2:     0
(2)  032036  000000                W3:     0
(2)
(2)                                        ;*
(2)                                        ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
(2)                                        ;*
(2)                                        ;*      CALL =  JSR     R5,$TLKR
(2)                                        ;*              .WORD   0               ;OFFSET OF DEVICE
(2)                                        ;*              ;RETURNS HERE
(2)                                        ;*DATA IN WORD $DATR
(2)                                        ;*
(2)
(2)  032040  010046                $TLKR:  MOV     R0,-(SP)        ;SAVE R0
(2)  032042  012500                        MOV     (5)+,R0         ;GET OFFSET
(2)  032044  052700  000300                BIS     #300,R0         ;ADD READ CODE
(2)  032050  004737  032206                JSR     PC,$LPW         ;WAIT TILL READY
(2)  032054  110077  147566                MOVB    R0,@KMAD4
(2)  032060  112777  000005  147554        MOVB    #5,@KMAD2       ;ISSUE WRITE FP
(2)  032066  004737  032206                JSR     PC,$LPW
(2)  032072  010037  032202                MOV     R0,RD1
(2)  032076                        1$:
(3)  032076  004537  032474                JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  032102  104000                        ERROR                   ;/TIME-OUT ERROR
(3)                                                                ;/WE FAILED TO COMPLETE
(3)                                                                ;/CURRENT OPERATION.
(3)                                                                ;/CONTINUES IN THIS LOOP
(3)                                                                ;/WOULD MAKE US "HANG" HERE
(3)
(3)  032104  000774                        BR              1$
(3)
(3)                                                                ;/RETURNS HERE-FROM-TIMED OUT.
(2)  032106  032777  000040  147522        BIT     #BIT5,@KMAD0    ;FAST PATH GOT DATA?
(2)  032114  001370                        BNE     1$
(2)  032116  112777  000004  147516        MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
(2)  032124  004737  032206                JSR     PC,$LPW
(2)  032130  117737  147512  032204        MOVB    @KMAD4,$DATR    ;GET LOW BYTE
(2)  032136                        2$:
(3)  032136  004537  032474                JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
```

```
(3)  032142  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)  032144  000774                        BR          2$
(3)
(3)                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
(2)  032146  032777  000040  147462        BIT     #BIT5,@KMAD0    ;FAST PATH READY?
(2)  032154  001370                        BNE     2$
(2)  032156  112777  000004  147456        MOVB    #4,@KMAD2       ;!SSUE FAST PATH READ
(2)  032164  004737  032206                JSR     PC,$LPW
(2)  032170  117737  147452  032205        MOVB    @KMAD4,$DATR+1  ;SAVE HIGH BYTE
(2)  032176  012600                        MOV     (SP)+,R0
(2)  032200  000205                        RTS     R5
(2)  032202  000000                RD1:    0
(2)  032204  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)  032206  010146                $LPW:   MOV     R1,-(SP)        ;SAVE R1
(2)  032210  005001                        CLR     R1
(2)  032212  122777  000377  147422  1$:    CMPB    #377,@KMAD2     ;FINISHED INSTRUCTION?
(2)  032220  001403                        BEQ     2$
(2)  032222  005201                        INC     R1              ;TIME OUT?
(2)  032224  001372                        BNE     1$
(2)  032226  000411                        BR      10$
(2)
(2)  032230  032777  000020  147400  2$:    BIT     #BIT4,@KMAD0    ;FAST PATH READ?
(2)  032236  001403                        BEQ     3$
(2)  032240  005201                        INC     R1              ;NO - TIME OUT?
(2)  032242  001372                        BNE     2$
(2)  032244  000402                        BR      10$             ;YES - REPORT AN ERROR
(2)
(2)  032246  012601                3$:     MOV     (SP)+,R1        ;RESTORE R1
(2)  032250  000207                        RTS     PC              ;EXIT
(2)
(2)  032252                        10$:
(3)  032252  104401  032260                TYPE    ,65$            ;;TYPE ASCIZ STRING
(3)  032256  000407                        BR      64$             ;;GET OVER THE ASCIZ
(3)                                ;;65$:  .ASCIZ  <200>#LPA-11 FAULT#
(3)  032276                        64$:
(2)
(2)  032276  000000                11$:    HALT                    ;LPA-11 FAULT RUN LPA-11
(2)  032300  000776                        BR      11$             ;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
(                                           ;* THAT ADDRESS.
(2)                                         ;* WHEN THE ADDR. IS KNOWN BY THE LPA, DO THE OUTPUT BY
(2)                                         ;* $TLKW
(2)                                         ;*
(2)
(2)  032302  010046          $OUTLP: MOV     R0,-(SP)        ;SAVE R0
(2)  032304  010146                  MOV     R1,-(SP)        ;SAVE R1
(2)
(2)  032306  012700  001664          MOV     #.DVLS,R0       ;PROGRAM DEFINED LIST.
(2)  032312  005001                  CLR     R1
(2)  032314  005710          1$:     TST     (0)             ;TERMINATOR REACHED?
(2)  032316  001421                  BEQ     10$             ;YES NEXT STEP.
(2)  032320  027520  000000          CMP     @(5),(0)+       ;MATCH WITH ADDR IN LIST?
(2)  032324  001402                  BEQ     2$
(2)  032326  005201                  INC     R1
(2)  032330  000771                  BR      1$
(2)
(2)  032332  010137  032350  2$:     MOV     R1,3$           ;SAVE OFFSET, DEVICE KNOWN.
(2)  032336  005725                  TST     (5)+
(2)  032340  013537  032352          MOV     @(5)+,4$        ;GET DATA TO BE WRITTEN
(2)  032344  004537  031724          JSR     R5,$TLKW        ;DO WRITE
(2)  032350  000000          3$:     .WORD   0               ;DEVICE OFFSET
(2)  032352  000000          4$:     .WORD   0               ;DATA TO BE WRITTEN.
(2)  032354  012601                  MOV     (SP)+,R1
(2)  032356  012600                  MOV     (SP)+,R0
(2)  032360  000205                  RTS     R5
(2)  032362  017520  000000  10$:     MOV     @(5),(0)+       ;SAVE ADDR.
(2)  032366  005010                  CLR     (0)
(2)  032370  004537  031074          JSR     R5,$LPAI
(2)  032374  001664                  .WORD   .DVLS
(2)  032376  000755                  BR      2$
(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)  032400  010046          $INLP:  MOV     R0,-(SP)        ;SAVE R0
(2)  032402  010146                  MOV     R1,-(SP)        ;SAVE R1
(2)
```

```
(2)  032404  012700  001664          MOV     #.DVLS,R0       ;PROG DEFINED ADDR. LIST.
(2)  032410  005001                  CLR     R1
(2)  032412  005710          1$:     TST     (0)             ;EOL REACHED?
(2)  032414  001420                  BEQ     10$             ;YES - DEFINE NEW ADDR.
(2)
(2)  032416  027520  000000          CMP     @(5),(0)+       ;ADDR. MATCH?
(2)  032422  001402                  BEQ     2$
(2)  032424  005201                  INC     R1
(2)  032426  000771                  BR      1$
(2)
(2)  032430  010137  032442  2$:     MOV     R1,3$           ;SAVE LIST OFFSET
(2)  032434  005725                  TST     (5)+
(2)  032436  004537  032040          JSR     R5,$TLKR        ;GO READ DEVICE
(2)          032442          $OFS=.
(2)  032442  000000          3$:     .WORD   0               ;OFFSET OF DEVICE
(2)
(2)  032444  013735  032204          MOV     $DATR,@(5)+     ;STORE DATA.
(2)  032450  012601                  MOV     (SP)+,R1        ;RESTORE R1
(2)  032452  012600                  MOV     (SP)+,R0        ;RESTORE R2
(2)  032454  000205                  RTS     R5              ;EXIT
(2)
(2)  032456  017520  000000  10$:     MOV     @(5),(0)+
(2)  032462  005010                  CLR     (0)
(2)  032464  004537  031074          JSR     R5,$LPAI
(2)  032470  001664                  .WORD   .DVLS
(2)  032472  000756                  BR      2$
(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)  032474  020537  032530  $TOUT:  CMP     R5,$SAD         ;SAME ADDR?
(2)  032500  001405                  BEQ     1$
(2)  032502  010537  032530          MOV     R5,$SAD         ;NO-SAVE THIS ADDR.
(2)  032506  005037  032532          CLR     $CNT            ;CLR CNT AT ADDR.
(2)  032512  000403                  BR      2$
(2)  032514  005237  032532  1$:     INC     $CNT            ;OVERFLOW?
(2)  032520  100402                  BMI     3$              ;YES-ERROR RETURN
(2)  032522  062705  000004  2$:     ADD     #4,R5           ;NO-NON ERROR RETURN
(2)  032526  000205          3$:     RTS     R5              ;RETURN.
(2)
(2)  032530  000000          $SAD:   .WORD   0               ;CONTAINS LOOP ADDR.
(2)  032532  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)  032534  000005                 $RESET:     RESET                   ;RESET THE WORLD.
(3)
(3)                                 ;*          MOV     @2$,1$  ;/READ DEVICE REG 2$,PUT DATA IN 1$.
(2)  032546  005737  031562                     TST     $AERR           ;IF NO ERROR,LOOP
(2)  032552  001004                             BNE     10$             ;THERE WAS AN ERROR.
(2)  032554  062737  000002  032570             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)  032562  000764                             BR      $RESET
(2)  032564                         10$:
(2)  032564  000207                             RTS     PC
(2)  032566  000000                 1$:         .WORD   0               ;JUNK LOC.
(2)  032570  160000                 2$:         .WORD   160000          ;DUMB ADDR. FORCES INIT OF DMC/KMC.
(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)  032572                         SDELAY:
(2)  032572  005737  032654                     TST     RTCCSR          ;CLOCK PRESENT?
(2)  032576  100016                             BPL     10$
(2)  032600  012737  000002  032644             MOV     #2,TIME
(2)  032606  052777  000115  000040             BIS     #115,@RTCCSR    ;START CLOCK
(2)  032614  005037  177776                     CLR     PS
(2)  032620  005737  032644         1$:         TST     TIME
(2)  032624  001375                             BNE     1$
(2)  032626  005077  000022                     CLR     @RTCCSR         ;STOP CLOCK
(2)
(2)  032632  000207                             RTS PC
(2)  032634  105237  032644         10$:        INCB    TIME
(2)  032640  001375                             BNE     10$
(2)  032642  000207                             RTS     PC
(2)
(2)  032644  000000                 TIME:       .WORD   0
(2)
(2)  032646  005337  032644         CLKINT:     DEC     TIME
(2)  032652  000002                             RTI
(2)  032654  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)  032656                                     $UTK:
(2)  032656  005037  001664                             CLR     .DVLS
(2)  032662                                     21$:
(3)  032662  104401  032670                             TYPE    ,65$            ;;TYPE ASCIZ STRING
(3)  032666  000405                             ;;65$:  BR      64$             ;;GET OVER THE ASCIZ
(3)                                             64$:    .ASCIZ  <200>#E OR D?#
(3)  032702                                     1$:
(2)  032702  105777  146236                             TSTB    @$TKS
(2)  032706  100375                             2$:     BPL     1$
(2)  032710  117737  146232  033032                     MOVB    @$TKB,20$       ;GET INPUT
(2)  032716  104401  033032                             TYPE,   20$             ;ECHO, NEXT MESSAGE.
(2)  032722  142737  000240  033032                     BICB    #240,20$        ;STRIP PARITY, LC
(2)  032730  104412                                     RDOCT                   ;GET ADDR.
(2)  032732  012637  033030                             MOV     (SP)+,14$
(2)  032736  123727  033032  000104                     CMPB    20$,#'D         ;DEPOSIT?
(2)  032744  001411                                     BEQ     10$
(2)
(2)  032746  004537  032400                             JSR     R5,$INLP        ;GET DATA
(2)  032752  033030                             2$:     .WORD   14$
(2)  032754  032766                                     .WORD   5$
(2)
(3)  032756  013746  032766                             MOV     5$,-(SP)        ;;SAVE 5$ FOR TYPEOUT
(3)  032762  104402                                     TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(2)  032764  000736                                     BR      21$             ;LOOP.
(2)  032766  000000                             5$:     .WORD   0
(2)
(2)  032770                                     10$:
(3)  032770  104401  032776                             TYPE    ,67$            ;;TYPE ASCIZ STRING
(3)  032774  000404                             ;;67$:  BR      66$             ;;GET OVER THE ASCIZ
(3)                                             66$:    .ASCIZ  <200>#DATA= #
(3)  033006                                     11$:
(2)  033006  104412                                     RDOCT
(2)  033010  012637  033026                             MOV     (SP)+,13$
(2)
(2)  033014  004537  032302                             JSR     R5,$OUTLP       ;OUTPUT ROUTINE.
(2)  033020  033030                             12$:    .WORD   14$             ;DEVICE ADDR.
(2)  033022  033026                                     .WORD   13$             ;DATA
(2)  033024  000716                                     BR      21$
(2)
(2)  033026  000000                             13$:    .WORD   0
(2)  033030  000000                             14$:    .WORD   0
(2)  033032  100001  042504  044526             20$:    .ASCIZ  <1><200>#DEVICE ADDR= #
(2)  033040  042503  040440  042104
(2)  033046  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)  033052  012537  033062     $PUTS:  MOV     (5)+,1$                 ;GET ADDR OF ADDR. OF A/D
(2)  033056  004537  032400             JSR     R5,$INLP
(2)  033062  000000             1$:     .WORD   0
(2)  033064  033160                     .WORD   10$
(2)  033066  113777  032442  146556     MOVB    $OFS,@KMAD6
(2)  033074  113777  032442  146552     MOVB    $OFS,@KMAD7
(2)  033102  013737  033062  033122     MOV     1$,2$
(2)  033110  062737  000002  033122     ADD     #2,2$
(2)  033116  004537  032400             JSR     R5,$INLP
(2)  033122  000000             2$:     .WORD   0
(2)  033124  033160                     .WORD   10$
(2)  033126  113777  032442  146510     MOVB    $OFS,@KMAD3
(2)  033134  152777  000340  146510     BISB    #340,@KMAD6
(2)  033142  152777  000300  146504     BISB    #300,@KMAD7
(2)  033150  152777  000300  146466     BISB    #300,@KMAD5
(2)  033156  000205                     RTS     R5
(2)  033160  000000             10$:    .WORD   0
(2)
3806
3807            042000                   .=42000
3808                                     ;HERE RESIDES THE MICRO-CODE USED BY THE PROGRAM
3809            043000                   .=43000
3810                                     ;FREE LOCATION AFTER THE MICRO-CODE
3811
3812            000001                   .END
```

```
ABASE = 170404          1221#   1279
ABR     001326          1279#   1462*   1518    1548    1549    1576    1577    1778    1797    1825    1853    2040    2092
                        2150    2202    2281    2282    2283    2284    2285    2286    2304    2350    2356    2391    2431
                        2548    2549    2593    2594    2618    2672    2685    2735    3084    3148    3258    3259    3260
                        3261    3520    3615    3788
ACDW1 = 000000          1279
ACDW2 = 00000C          1279
ACPUOP= 000000          1279
ACR     001330          1279#   1464*   1518    1801    1826    1854    2052    2109    2219    2281    2282    2283    2284
                        2285    2286    2312    2360    2437    2593    2594    2622    2699    2742    2987    3096    3155
                        3258    3259    3260    3261    3527    3789    3796    3797
ADDW0 = 000000          1279
ADDW1 = 000000          1279
ADDW10= 000000          1279
ADDW11= 000000          1279
ADDW12= 000000          1279
ADDW13= 000000          1279
ADDW14= 000000          1279
ADDW15= 000000          1279
ADDW2 = 000000          1279
ADDW3 = 000000          1279
ADDW4 = 000000          1279
ADDW5 = 000000          1279
ADDW6 = 000000          1279
ADDW7 = 000000          1279
ADDW8 = 000000          1279
ADDW9 = 000000          1279
ADEVCT= 000000          1279
ADEVM = 000000          1279
AENV  = 000000          1279
AENVM = 000000          1279
AFATAL= 000000          1279
AMADR1= 000000          1279
AMADR2= 000000          1279
AMADR3= 000000          1279
AMADR4= 000000          1279
AMAMS1= 000000          1279
AMAMS2= 000000          1279
AMAMS3= 000000          1279
AMAMS4= 000000          1279
AMSGAD= 000000          1279
AMSGLG= 000000          1279
AMSGTY= 000000          1279
AMTYP1= 000000          1279
AMTYP2= 000000          1279
AMTYP3= 000000          1279
AMTYP4= 000000          1279
APASS = 000000          1279
APRIOR= 000006          1223#   1279
APRITY  001350          1279#   1480
APTCSU= 000040          3744    3745#
APTENV= 000001          3738    3744    3745#
APTSIZ= 000200          1444    3745#
APTSPO= 000100          3744    3745#
ASR     001324          1279#   1450*   1460    1518    1609    1610    1620    1778    1796    1820    1844    1971    1972
                        1976    1979    2004    2005    2007    2008    2020    2039    2046    2047    2049    2091    2099
```

```
                    2102  2104  2148  2154  2155  2157  2160  2170  2194  2204  2205  2207  2210
                    2281  2282  2283  2284  2285  2286  2303  2306  2316  2349  2352  2389  2393
                    2394  2396  2397  2429  2433  2434  2436  2474  2481  2482  2484  2486  2548
                    2549  2593  2594  2616  2619  2621  2670  2676  2677  2679  2731  2737  2745
                    2754  2791  2793  2834  2855  2857  2905  2913  2915  2949  2976  2980  3048
                    3049  3050  3051  3052  3053  3082  3092  3093  3095  3146  3149  3151  3258
                    3259  3260  3261  3362  3363  3364  3365  3458  3466  3469  3472  3509  3522
                    3523  3525  3564  3567  3569  3580  3612  3619  3622  3632  3665  3670  3687
                    3787  3794
ASWREG= 000000      1279
ATESTN= 000000      1279
AUNIT = 000000      1279
AUSWR = 000000      1279
AVECP2  001342      1279#  1474*
AVECT   001340      1279#  1472
AVECT1= 000344      1222#  1279  1439
AVECT2= 000000      1279
BBR     001334      1279#  1468*  1518  1660  1661  1778  1882  1909  1937  2784  2843  2908  2952
                    2978   3048   3049  3050  3051  3052  3053  3086  3362  3363  3364  3365  3668
                    3791
BCR     001336      1279#  1470*  1518  1886  1912  1940  2794  2861  2925  2958  3048  3049  3050
                    3051   3052   3053  3362  3363  3364  3365  3792  3798  3799
BIT0  = 000001      1220#  1778  2789  3145
BIT00 = 000001      1220#  2009  2211  2398
BIT01 = 000002      1220#
BIT02 = 000004      1220#
BIT03 = 000010      1220#
BIT04 = 000020      1220#
BIT05 = 000040      1220#  2161  2281  2282  2283  2284  2285  2286  2317  3623
BIT06 = 000100      1220#
BIT07 = 000200      1220#  2171  2487  3678
BIT08 = 000400      1220#  3741
BIT09 = 001000      1220#  3738  3741
BIT1  = 000002      1220#  1778  2789  3805
BIT10 = 002000      1220#  1778  2483  2548  2549  2593  2594  2620  3468  3738
BIT11 = 004000      1220#  1778  2732  2779  2792  2836  2856  2914  3094  3258  3259  3260  3261
                    3362   3363  3364  3365  3805
BIT12 = 010000      1220#  1778  1975  2006  2048  2103  2156  2206  2395  2435  2678  3150  3524
                    3568
BIT13 = 020000      1220#  1778  2003  3482  3536  3588  3641  3696  3738
BIT14 = 040000      1220#  1778  2123  2875  3741  3805
BIT15 = 100000      1220#  1778  3473  3805
BIT2  = 000004      1220#  1778
BIT3  = 000010      1220#  1778
BIT4  = 000020      1220#  1778  3805
BIT5  = 000040      1220#  1778  2789  3362  3363  3364  3365  3805
BIT6  = 000100      1220#  1778
BIT7  = 000200      1220#  1778
BIT8  = 000400      1220#  1778  3521
BIT9  = 001000      1220#  1778  3521  3568
BPRITY  001352      1279#  1480*
BPTVEC= 000014      1220#
BSR     001332      1279#  1466*  1518  1635  1636  1778  1881  1902  1930  2346  2470  2548  2549
                    2669   2674   2713  2733  2780  2788  2790  2837  2848  2850  2906  2910  2912
                    2916   2950   2977  3048  3049  3050  3051  3052  3053  3083  3088  3090  3258
                    3259   3260   3261  3362  3363  3364  3365  3458  3467  3509  3564  3612  3665
```

```
                    3676    3677    3790    3795
BVECT   001344      1279#   1476*
BVECT2  001346      1279#   1478*
CKSWR = 104407      3738    3741    3751#
CLKINT  032646      3805#
CR    = 000015      1220#   3744
CRLF  = 000200      1220#   3744
DDISP = 177570      1220#   1279    1444
DF0     031070      1290    1297    1304    1311    1318    1325    1332    1340    1347    1354    1358    1359    1360
                    1361    1368    1375    1382    1389    1396    1403    1410    1417    1424    1431    1438    3802#
DH1     030053      1288    1295    3771#
DH10    030337      1337    3777#
DH12    030476      1352    1380    3779#
DH14    030515      1366    1387    3780#
DH20    030533      1394    3781#
DH23    030575      1415    3782#
DH26    030637      1436    3783#
DH3     030111      1302    3772#
DH4     030147      1309    1345    1401    1408    3773#
DH5     030205      1316    3774#
DH6     030243      1323    3775#
DH7     030301      1330    1373    1422    1429    3776#
DISPLA  001142      1279#   1444*   3738*   3741*
DISPRE  000174      1209#   1444
DRLPX2= ****** G      52#   3805
DSWR  = 177570      1220#   1279    1444
DT1     030666      1289    1296    1395    1416    3787#
DT10    030762      1339    3793#
DT12    030776      1353    1381    3794#
DT14    031004      1367    1388    3795#
DT21    031012      1402    3796#
DT22    031024      1409    3797#
DT24    031036      1423    3798#
DT25    031050      1430    3799#
DT26    031062      1437    3800#
DT3     030700      1303    3788#
DT4     030712      1310    1346    3789#
DT5     030724      1317    3790#
DT6     030736      1324    3791#
DT7     030750      1331    1374    3792#
EMTVEC= 000030      1220#   1444*
EM1     027166      1287    3753#
EM10    027433      1336    3760#
EM11    027460      1344    1400    1407    3761#
EM12    027507      1351    3762#
EM14    027547      1365    3763#
EM15    027607      1372    1421    1428    3764#
EM16    027636      1379    3765#
EM17    027671      1386    3766#
EM2     027221      1294    3754#
EM20    027724      1393    3767#
EM23    027763      1414    3768#
EM26    030022      1435    3769#
EM3     027250      1301    3755#
EM4     027277      1308    3756#
EM5     027326      1315    3757#
```

```
EM6      027355            1322    3758#
EM7      027404            1329    3759#
ERRVEC=  000004            1220#   1444*   3741*
GNS   =  ****** U          1209    1448    3482    3536    3588    3641    3696    3706    3710    3751    3752    3805
GTSWR =  104406            3751#
HT    =  000011            1220#   3744
IOTRD    023700            3703#   3752
IOTT  =  104413            3752#
IOTVEC=  000020            1220#   1444*
KMAD0    001636            1439#   1454    3805*
KMAD1    001640            1439#   1454
KMAD2    001642            1439#   3805*
KMAD3    001644            1439#   3805*
KMAD4    001646            1439#   3805*
KMAD5    001650            1439#
KMAD6    001652            1439#   3805*
KMAD7    001654            1439#   1454    3805*
LF    =  000012            1220#   3744
LOOP     002334            1456#   3395
LPADH    001650            1439#
LPADL    001646            1439#
LPCI     001636            1439#
LPCO     001642            1439#
LPMR     001640            1439#
LPMS1    001652            1439#
LPMS2    001654            1439#
LPSO     001644            1439#
LS210    022450            1214    3457#
LS214    022646            1215    3508#
LS220    023062            1216    3563#
LS224    023246            1217    3611#
LS230    023454            1218    3664#
P     =  000012            1778#   2548#   2549#
PIRQ  =  177772            1220#
PIRQVE=  000240            1220#
PR0   =  000000            1220#
PR1   =  000040            1220#
PR2   =  000100            1220#
PR3   =  000140            1220#
PR4   =  000200            1220#
PR5   =  000240            1220#
PR6   =  000300            1220#
PR7   =  000340            1220#
PS    =  177776            1220#   3805*
PSW   =  177776            1220#
PWRMSG   027034            3746    3747#
PWRVEC=  000024            1220#   1444*   3746*
RDCHR =  104410            3743    3751#
RDLIN =  104411            3742    3751#
RDOCT =  104412            3751#   3805
RD1      032202            3805#*
RESVEC=  000010            1220#
RSTART   002462            1213    1482#
RTCCSR   032654            3805#*
SDELAY   032572            3805#
SSTART   002466            1481    1483#
```

```
STACK = 001100           1220#   1444    1483
START   001726           1211    1444#   3746
STKLMT= 177774           1220#
SWR     001140           1279#   1444*   2123    2875    3482    3536    3588    3641    3696    3738    3741    3743*   3746*
SWREG   000176           1209#   1444    3743
SW0   = 000001           1220#
SW00  = 000001           1220#
SW01  = 000002           1220#
SW02  = 000004           1220#
SW03  = 000010           1220#
SW04  = 000020           1220#
SW05  = 000040           1220#
SW06  = 000100           1220#
SW07  = 000200           1220#
SW08  = 000400           1220#
SW09  = 001000           1220#
SW1   = 000002           1220#
SW10  = 002000           1220#
SW11  = 004000           1220#
SW12  = 010000           1220#
SW13  = 020000           1220#
SW14  = 040000           1220#
SW15  = 100000           1220#
SW2   = 000004           1220#
SW3   = 000010           1220#
SW4   = 000020           1220#
SW5   = 000040           1220#
SW6   = 000100           1220#
SW7   = 000200           1220#
SW8   = 000400           1220#
SW9   = 001000           1220#
TBITVE= 000014           1220#
TIME    032644           3805#*
TKVEC = 000060           1220#
TPVEC = 000064           1220#
TRAPVE= 000034           1220#   1444*
TRTVEC= 000014           1220#
TST1    002504           1518#
TST10   003154           1778#
TST100  012172           2281#
TST101  012304           2282#
TST102  012416           2283#
TST103  012530           2284#
TST104  012642           2285#
TST105  012754           2286#
TST106  013066           2297#
TST107  013176           2339#
TST11   003254           1778#
TST110  013312           2379#
TST111  013434           2418#
TST112  013554           2467#
TST113  013674           2548#
TST114  014054           2549#
TST115  014234           2593#
TST116  014354           2594#
TST117  014474           2607#
```

TST12	003354	1778#
TST120	014600	2651#
TST121	015016	2724#
TST122	015164	2776#
TST123	015314	2827#
TST124	015522	2893#
TST125	015712	2943#
TST126	016006	2968#
TST127	016106	3048#
TST13	003454	1778#
TST130	016226	3049#
TST131	016346	3050#
TST132	016466	3051#
TST133	016606	3052#
TST134	016726	3053#
TST135	017046	3064#
TST136	017240	3136#
TST137	017352	3258#
TST14	003554	1778#
TST140	017642	3259#
TST141	020132	3260#
TST142	020422	3261#
TST143	020712	3362#
TST144	021216	3363#
TST145	021522	3364#
TST146	022026	3365#
TST15	003654	1778#
TST16	003754	1778#
TST17	004054	1778#
TST2	002602	1542#
TST20	004154	1778#
TST21	004254	1778#
TST22	004354	1778#
TST23	004454	1778#
TST24	004554	1778#
TST25	004654	1778#
TST26	004754	1778#
TST27	005054	1778#
TST3	002642	1571#
TST30	005154	1778#
TST31	005254	1778#
TST32	005354	1778#
TST33	005454	1778#
TST34	005554	1778#
TST35	005654	1778#
TST36	005754	1778#
TST37	006054	1778#
TST4	002700	1604#
TST40	006154	1778#
TST41	006254	1778#
TST42	006354	1778#
TST43	006454	1778#
TST44	006554	1778#
TST45	006654	1778#
TST46	006754	1778#
TST47	007054	1778#

```
TST5    002754          1630#
TST50   007154          1778#
TST51   007254          1778#
TST52   007354          1778#
TST53   007454          1778#
TST54   007554          1778#
TST55   007654          1778#
TST56   007754          1778#
TST57   010054          1778#
TST6    003014          1655#
TST60   010154          1778#
TST61   010254          1778#
TST62   010354          1778#
TST63   010454          1778#
TST64   010554          1789#
TST65   010626          1812#
TST66   010704          1840#
TST67   010762          1874#
TST7    003054          1778#
TST70   011034          1898#
TST71   011112          1926#
TST72   011170          1968#
TST73   011254          1998#
TST74   011360          2034#
TST75   011502          2083#
TST76   011656          2144#
TST77   012024          2190#
TYPDS = 104405          3395    3751#
TYPE  = 104401          1448    3395    3482    3536    3588    3641    3696    3706    3710    3737    3738    3739    3740
                        3743    3744    3746    3751#   3805
TYPOC = 104402          3708    3716    3739    3743    3751#   3805
TYPON = 104404          3751#
TYPOS = 104403          3751#   3805
VECTOR  001656          1439#
VECTPS  001660          1439#
VERSN   001662          1439#
W1      032032          3805#*
W2      032034          3805#*
W3      032036          3805#*
$AERR   031562          3805#*
$APTHD  001000          1277#
$ASTAT= ****** U        3745
$ATYC   026440          3745#
$ATY1   026414          3745#
$ATY3   026422          3744    3745#
$ATY4   026432          3738    3745#
$AUTOB  001134          1279#   1445    3743
$BASE   001254          1279#   1450
$BDADR  001122          1279#   3793
$BDDAT  001126          1279#   1518    1549    1550    1577    1578    1610    1611    1636    1637    1661    1662    1778
                        1801    1802    1826    1828    1854    1856    1886    1887    1912    1914    1940    1942    1979
                        1980    2008    2009    2052    2053    2109    2111    2160    2161    2170    2171    2219    2220
                        2281    2282    2283    2284    2285    2286    2312    2313    2316    2317    2346    2347    2360
                        2361    2397    2398    2437    2438    2470    2471    2486    2487    2548    2549    2593    2594
                        2622    2623    2685    2686    2699    2700    2742    2743    2745    2746    2794    2795    2861
                        2863    2916    2917    2925    2926    2958    2959    2987    2988    3048    3049    3050    3051
```

```
                        3052    3053    3096    3097    3155    3156    3258    3259    3260    3261    3362    3363    3364
                        3365    3527    3528    3580    3581    3632    3633    3687    3688    3787    3788    3789    3790
                        3791    3792    3793    3797    3799
$BELL   001170          1279#   3738
$CDW1   001260          1279#
$CDW2   001262          1279#
$CHARC  026410          3744#*
$CKSWR  025356          3743#   3751
$CMTAG  001100          1279#   1444
$CM1  = 000001          1279#
$CM2  = 000002          1279#
$CM3  = 000001          1279#
$CM4  = 000001          1279#
$CNT    032532          3805#*
$CNTLG  026103          3743#
$CNTLU  026076          3743#
$CPUOP  001226          1279#
$CRLF   001175          1279#   3738    3739    3743    3744
$DATR   032204          3805#*
$DBLK   025052          3740#
$DDW0   001264          1279#
$DDW1   001266          1279#
$DDW10  001310          1279#
$DDW11  001312          1279#   -
$DDW12  001314          1279#
$DDW13  001316          1279#
$DDW14  001320          1279#
$DDW15  001322          1279#
$DDW2   001270          1279#
$DDW3   001272          1279#
$DDW4   001274          1279#
$DDW5   001276          1279#
$DDW6   001300          1279#
$DDW7   001302          1279#
$DDW8   001304          1279#
$DDW9   001306          1279#
$DEVCT  001210          1279#   1451*
$DEVM   001256          1279#
$DOAGN  022424          3395#
$DTBL   025042          3740#
$ENDAD  022414          1275    3395#   3738
$ENDCT  022362          1444    3395#
$ENDMG  022433          3395#
$ENULL  022430          3395#
$ENV    001220          1279#   3738    3744    3745
$ENVM   001221          1279#   1444    3744    3745
$EOP    022332          3395#
$EOPCT  022354          1444*   3395#
$ERFLG  001103          1279#   3738*   3741*
$ERMAX  001115          1279#   1444*   3741*
$ERROR  024302          1444    3738#
$ERRPC  001116          1279#   3738*   3739    3787    3788    3789    3790    3791    3792    3793    3794    3795    3796
                        3797    3798    3799    3800
$ERRTB  001356          1279#   3739
$ERRTY  024502          3738    3739#
$ERTTL  001112          1279#   3738*
```

```
$ESCAP  001166  1279#  1444*  3738  3741*
$ETABL  001220  1279#
$ETEND  001324  1277  1279#
$FATAL  001202  1279#  3745*
$FFLG   026660  3745#*
$FILLC  001156  1279#  3744
$FILLS  001155  1279#  3744
$GDADR  001120  1279#  3793
$GDDAT  001124  1279#  1547*  1548  1573*  1576  1606*  1609  1619*  1620  1632*  1635  1657*  1660
                1778*  1795*  1796  1797  1799*  1819*  1820  1822*  1825  1828  1843*  1844  1850*
                1853  1856  1880*  1881  1882  1884*  1901*  1902  1908*  1909  1914  1929*  1930
                1936*  1937  1942  1970*  1971  1972  1975*  1976  2003*  2004  2005  2006*  2007
                2019*  2020  2038*  2039  2040  2045*  2046  2047  2048*  2049  2050*  2053  2085*
                2092  2106*  2111  2120  2125*  2147*  2148  2149*  2150  2153*  2154  2155  2156*
                2157  2193*  2194  2201*  2202  2203*  2204  2205  2206*  2207  2281*  2282*  2283*
                2284*  2285*  2286*  2302*  2303  2304  2305*  2306  2348*  2349  2350  2351*  2352
                2355*  2356  2388*  2389  2390*  2391  2392*  2393  2428*  2429  2430*  2431  2432*
                2433  2434  2435*  2436  2473*  2474  2480*  2481  2482  2483*  2484  2548*  2549*
                2593*  2594*  2615*  2616  2617*  2618  2619  2620*  2621  2668*  2669  2670  2671*
                2672  2673*  2674  2675*  2676  2677  2678*  2679  2683*  2712*  2713  2730*  2731
                2732*  2733  2734*  2735  2736*  2737  2753*  2754  2779*  2780  2783*  2784  2788
                2789*  2790  2791  2792*  2793  2829*  2843  2858*  2863  2872  2877*  2904*  2905
                2906  2907*  2908  2909*  2910  2911*  2912  2913  2914*  2915  2948*  2949  2950
                2951*  2952  2956*  2975*  2976  2977  2978  2979*  2980  3048*  3049*  3050*  3051*
                3052*  3053*  3081*  3082  3083  3084  3085*  3086  3087*  3088  3089*  3090  3091*
                3092  3093  3094*  3095  3147*  3148  3149  3150*  3151  3153*  3156  3258*  3259*
                3260*  3261*  3362*  3363*  3364*  3365*  3787  3788  3789  3790  3791  3792  3793
                3796  3798
$GET42  022404  3395#
$GTSWR  025426  3743#  3751
$HD  =  000001  1207
$HIBTS  001000  1277#
$HIOLT  025354  3742#*
$ICNT   001104  1279#  3458*  3482*  3509*  3536*  3564*  3588*  3612*  3641*  3665*  3696*
$ILLUP  027026  3746#
$INLP   032400  1518  1549  1577  1610  1636  1661  1778  1801  1826  1854  1886  1912  1940
                1972  1979  2005  2008  2047  2052  2102  2109  2155  2160  2170  2205  2210
                2219  2281  2282  2283  2284  2285  2286  2312  2316  2346  2360  2394  2397
                2434  2437  2470  2482  2486  2548  2549  2593  2594  2619  2622  2677  2685
                2699  2742  2745  2788  2791  2794  2855  2861  2913  2916  2925  2958  2987
                3048  3049  3050  3051  3052  3053  3093  3096  3149  3155  3258  3259  3260
                3261  3362  3363  3364  3365  3472  3523  3527  3567  3580  3622  3632  3677
                3687  3805#
$INTAG  001135  1279#  3743
$ITEMB  001114  1279#  3738*  3739
$LF     001176  1279#  3738  3743  3744
$LFLG   026657  3745#*
$LOAD   031564  3805#
$LPADR  001106  1279#  1444*  3741*
$LPAI   031074  3805#
$LPERR  001110  1279#  1444*  3738  3741*
$LPW    032206  3805#
$MADR1  001232  1279#
$MADR2  001236  1279#
$MADR3  001242  1279#
$MADR4  001246  1279#
```

```
$MAIL   001200          1277    1279#   1444    3738    3741    3744
$MAMS1  001230          1279#
$MAMS2  001234          1279#
$MAMS3  001 40          1279#
$MAMS4  001244          1279#
$MBADR  001002          1277#
$MFLG   026656          3745#*
$MNEW   026121          3743#
$MSGAD  001214          1279#   3745*
$MSGLG  001216          1279#   3745*
$MSGTY  001200          1279#   3745*
$MSWR   026110          3743#
$MTYP1  001231          1279#
$MTYP2  001235          1279#
$MTYP3  001241          1279#
$MTYP4  001245          1279#
$NULL   001154          1279#   3744
$NWTST= 000001          1518#   1542#   1571#   1604#   1630#   1655#   1778#   1789#   1812#   1840#   1874#   1898#   1926#
                        1968#   1998#   2034#   2083#   2144#   2190#   2281#   2282#   2283#   2284#   2285#   2286#   2297#
                        2339#   2379#   2418#   2467#   2548#   2549#   2593#   2594#   2607#   2651#   2724#   2776#   2827#
                        2893#   2943#   2968#   3048#   3049#   3050#   3051#   3052#   3053#   3064#   3136#   3258#   3259#
                        3260#   3261#   3362#   3363#   3364#   3365#
$OCNT   024276          3737#*
$OFS  = 032442          3805#
$OMODE  024300          3737#*
$OUTLP  032302          1548    1576    1609    1620    1635    1660    1778    1796    1797    1820    1825    1844    1853
                        1881    1882    1902    1909    1930    1937    1971    1976    2004    2007    2020    2039    2040
                        2046    2049    2091    2092    2099    2104    2148    2150    2154    2157    2194    2202    2204
                        2207    2281    2282    2283    2284    2285    2286    2303    2304    2306    2349    2350    2352
                        2356    2389    2391    2393    2396    2429    2431    2433    2436    2474    2481    2484    2548
                        2549    2593    2594    2616    2618    2621    2669    2670    2672    2674    2676    2679    2713
                        2731    2733    2735    2737    2754    2780    2784    2790    2793    2834    2837    2843    2848
                        2850    2857    2905    2906    2908    2910    2912    2915    2949    2950    2952    2976    2977
                        2978    2980    3048    3049    3050    3051    3052    3053    3082    3083    3084    3086    3088
                        3090    3092    3095    3146    3148    3151    3258    3259    3260    3261    3362    3363    3364
                        3365    3458    3466    3467    3469    3509    3520    3522    3525    3564    3569    3612    3615
                        3619    3665    3668    3670    3676    3805#
$OVER   025240          3741#
$PASS   001206          1279#   1444*   1452*   3395*   3458*   3482*   3509*   3536*   3564*   3588*   3612*   3641*   3665*
                        3696*
$PASTM  001006          1277#
$PUTS   033052          3805#
$PWRAD  027022          3746#
$PWRDN  026662          1444    3746#
$PWRMG  027016          3746#
$PWRUP  026734          3746#
$QUES   001174          1279#   3738    3743    3744
$RDCHR  025640          3743#   3751
$RDDEC= ****** U        3751
$RDLIN  025760          3743#   3751
$RDOCT  025254          3742#   3751
$RDSZ = 000010          3743#
$REGAD  001160          1279#
$REGO   001162          1279#
$RESET  032534          2954    3805#
$RTNAD  022426          3395#
```

```
$R2A   = ****** U        3751
$SAD     032530          3805#*
$SAVRE= ****** U         3751
$SAVR6   027032          3746#*
$SCOPE   025062          1444    3741#
$SETUP= 000137           1442#   1444    3395    3738    3741    3743
$STUP  = 177777          1442#
$SVLAD   025204          3741#
$SVPC  = 000234          1275#
$SWR   = 163400          1161#   1207    1208    1279    1444    1518    1542    1571    1604    1630    1655    1778    1789
                         1812    1840    1874    1898    1926    1968    1998    2034    2083    2144    2190    2281    2282
                         2283    2284    2285    2286    2297    2339    2379    2418    2467    2548    2549    2593    2594
                         2607    2651    2724    2776    2827    2893    2943    2968    3048    3049    3050    3051    3052
                         3053    3064    3136    3258    3259    3260    3261    3362    3363    3364    3365    3395    3738
                         3741    3746
$SWREG   001222          1279#   1444
$SWRMK= 000000           1208    3741
$TESTN   001204          1279#   1518*   3741*
$TKB     001146          1279#   3743    3805
$TKS     001144          1279#   3743*   3805
$TLKR    032040          3805#
$TLKW    031724          3805#
$TMDAT   001354          1279#   2090*   2091    2098*   2099    2102    2103*   2104    2210    2211    2394    2395*   2396
                         2833*   2834    2836*   2837    2847*   2848    2849*   2850    2855    2856*   2857    3145*   3146
                         3258*   3259*   3260*   3261*   3362*   3363*   3364*   3365*   3458*   3466    3467    3468*   3469
                         3472    3473    3509*   3519*   3520    3521*   3522    3523    3524*   3525    3564*   3566*   3567
                         3568*   3569    3612*   3614*   3615    3618*   3619    3622    3623    3665*   3667*   3668    3669*
                         3670    3675*   3676    3677    3678
$TMPO    001164          1279#   3796    3797    3798    3799    3800
$TN    = 000147          1207#   1518#   1542#   1571#   1604#   1630#   1655#   1778#   1789#   1812#   1840#   1874#   1898#
                         1926#   1968#   1998#   2034#   2083#   2144#   2190#   2281#   2282#   2283#   2284#   2285#   2286#
                         2297#   2339#   2379#   2418#   2467#   2548#   2549#   2593#   2594#   2607#   2651#   2724#   2776#
                         2827#   2893#   2943#   2968#   3048#   3049#   3050#   3051#   3052#   3053#   3064#   3136#   3258#
                         3259#   3260#   3261#   3362#   3363#   3364#   3365#
$TOUT    032474          3805#
$TPB     001152          1279#   3744*
$TPFLG   001157          1279#   3744
$TPS     001150          1279#   3744
$TRAP    027102          1444    3751#
$TRAP2   027124          3751#
$TRP   = 000014          3751#   3752#
$TRPAD   027136          3751#
$TSTM    001004          1277#
$TSTNM   001102          1279#   1518*   3395*   3738    3741*
$TTYIN   026066          3743#
$TYPBN= ****** U         3751
$TYPDS   024636          3740#   3751
$TYPE    026132          3744#   3745    3751
$TYPEC   026344          3743    3744#
$TYPEX   026412          3744#
$TYPOC   024100          3737#   3751
$TYPON   024114          3737#   3751
$TYPOS   024054          3737#   3751
$UNIT    001212          1279#
$UNITM   001010          1277#
$USWR    001224          1279#
```

```
$UTK    032656          3805#
$VECT1  001250          1279#
$VECT2  001252          1279#
$XTSTR  025074          3741#
$$GET4= 000000          3395#
$OFILL  024277          3737#*
$40CAT= ****** U        3738    3741
.     = 043000          1209#   1210#   1275#   1276#   1277#   1279#   1439#   1444    1448#   3395    3482#   3536#   3588#
                        3641#   3696#   3710#   3738    3739#   3740#   3741    3743#   3744    3745#   3746    3785#   3805#
                        3807#   3809#
.DVLS   001664          1439#   1454*   1482*   3457*   3508*   3563*   3611*   3664*   3805*
.$ASTA= ****** U        3745
.$X   = 001000          1277#
```

```
ADTST   1493#  1518
AREPT   3263#
BREPT   3367#
BRTM    2997#  3048  3049  3050  3051  3052  3053
CADT    3169#  3258  3259  3260  3261
CADTB   3288#  3362  3363  3364  3365
CBC     1778#
CBTC    2558#  2593  2594
CBT25   2494#  2548  2549
COMMEN  1220#  1554  1557  1581  1584  1613  1617  1640  1644  1665  1669  1778  1805  1809  1832
        1837   1860  1865  1890  1894  1918  1923  1946  1951  1983  1987  2012  2017  2056  2063
        2114   2118  2163  2166  2174  2179  2213  2216  2223  2231  2281  2282  2283  2284  2285
        2286   2321  2326  2363  2367  2400  2403  2440  2452  2489  2491  2548  2549  2593  2594
        2625   2628  2688  2695  2704  2710  2748  2751  2797  2810  2865  2869  2919  2922  2929
        2932   2962  2965  2990  2994  3048  3049  3050  3051  3052  3053  3100  3111  3159  3165
        3258   3259  3260  3261  3362  3363  3364  3365  3476  3480  3531  3534  3583  3586  3635
        3639   3690  3694  3724  3730
CSRDTA  1680#  1778
DFC     1280#  1290  1297  1304  1311  1318  1325  1332  1340  1347  1354  1368  1375  1382  1396
        1403   1410  1417  1424  1431  1438
ECALL   1717#  1778
ECB     1254#  1554  1557  1581  1584  1613  1617  1640  1644  1665  1669  1778  1805  1809  1832
        1837   1860  1865  1890  1894  1918  1923  1946  1951  1983  1987  2012  2017  2056  2063
        2114   2118  2163  2166  2174  2179  2213  2216  2223  2231  2281  2282  2283  2284  2285
        2286   2321  2326  2363  2367  2400  2403  2440  2452  2489  2491  2548  2549  2593  2594
        2625   2628  2688  2695  2704  2710  2748  2751  2797  2810  2865  2869  2919  2922  2929
        2932   2962  2965  2990  2994  3048  3049  3050  3051  3052  3053  3100  3111  3159  3165
        3258   3259  3260  3261  3362  3363  3364  3365  3476  3480  3531  3534  3583  3586  3635
        3639   3690  3694  3724  3730
ENDCOM  1220#  1554  1557  1581  1584  1613  1617  1640  1644  1665  1669  1778  1805  1809  1832
        1837   1860  1865  1890  1894  1918  1923  1946  1951  1983  1987  2012  2017  2056  2063
        2114   2118  2163  2166  2174  2179  2213  2216  2223  2231  2281  2282  2283  2284  2285
        2286   2321  2326  2363  2367  2400  2403  2440  2452  2489  2491  2548  2549  2593  2594
        2625   2628  2688  2695  2704  2710  2748  2751  2797  2810  2865  2869  2919  2922  2929
        2932   2962  2965  2990  2994  3048  3049  3050  3051  3052  3053  3100  3111  3159  3165
        3258   3259  3260  3261  3362  3363  3364  3365  3476  3480  3531  3534  3583  3586  3635
        3639   3690  3694  3701  3724  3730
ERROR   1220#  1555  1582  1614  1641  1666  1778  1806  1833  1861  1891  1919  1947  1984  2013
        2057   2115  2164  2175  2214  2224  2281  2282  2283  2284  2285  2286  2322  2364  2401
        2441   2490  2548  2549  2593  2594  2626  2689  2705  2749  2798  2866  2920  2930  2963
        2991   3048  3049  3050  3051  3052  3053  3101  3161  3258  3259  3260  3261  3362  3363
        3364   3365  3478  3532  3584  3636  3691  3805
ESCAPE  1220#
GETPRI  1220#
GETSWR  1220#
MOVEI    170#  1518  1549  1577  1610  1636  1661  1778  1801  1826  1854  1886  1912  1940  1972
        1979   2005  2008  2047  2052  2102  2109  2155  2160  2170  2205  2210  2219  2281  2282
        2283   2284  2285  2286  2312  2316  2346  2360  2394  2397  2434  2437  2470  2482  2486
        2548   2549  2593  2594  2619  2622  2677  2685  2699  2742  2745  2788  2791  2794  2855
        2861   2913  2916  2925  2958  2987  3048  3049  3050  3051  3052  3053  3093  3096  3149
        3155   3258  3259  3260  3261  3362  3363  3364  3365  3472  3523  3527  3567  3580  3622
        3632   3677  3687  3805
MOVEM    157#  1548  1576  1609  1620  1635  1660  1778  1796  1797  1820  1825  1844  1853  1881
        1882   1902  1909  1930  1937  1971  1976  2004  2007  2020  2039  2040  2046  2049  2091
        2092   2099  2104  2148  2150  2154  2157  2194  2202  2204  2207  2281  2282  2283  2284
        2285   2286  2303  2304  2306  2349  2350  2352  2356  2389  2391  2393  2396  2429  2431
```

	2433	2436	2474	2481	2484	2548	2549	2593	2594	2616	2618	2621	2669	2670	2672
	2674	2676	2679	2713	2731	2733	2735	2737	2754	2780	2784	2790	2793	2834	2837
	2843	2848	2850	2857	2905	2906	2908	2910	2912	2915	2949	2950	2952	2976	2977
	2978	2980	3048	3049	3050	3051	3052	3053	3082	3083	3084	3086	3088	3090	3092
	3095	3146	3148	3151	3258	3259	3260	3261	3362	3363	3364	3365	3458	3466	3467
	3469	3509	3520	3522	3525	3564	3569	3612	3615	3619	3665	3668	3670	3676	
MSY	1249#	1518	1542	1571	1604	1630	1655	1778	1789	1812	1840	1874	1898	1926	1968
	1998	2034	2083	2144	2190	2281	2282	2283	2284	2285	2286	2297	2339	2379	2418
	2467	2548	2549	2593	2594	2607	2651	2724	2776	2827	2893	2943	3048	3049	3050
	3051	3052	3053	3064	3136	3258	3259	3260	3261	3362	3363	3364	3365	3450	3502
	3556	3605	3658												
MTAGS	1224#	1279													
MULT	1220#														
NEWTST	1220#	1518	1542	1571	1604	1630	1655	1778	1789	1812	1840	1874	1898	1926	1968
	1998	2034	2083	2144	2190	2281	2282	2283	2284	2285	2286	2297	2339	2379	2418
	2467	2548	2549	2593	2594	2607	2651	2724	2776	2827	2893	2943	2968	3048	3049
	3050	3051	3052	3053	3064	3136	3258	3259	3260	3261	3362	3363	3364	3365	
POP	1220#	3740	3742	3745	3746										
POPSP2	1246#														
PUSH	1220#	3740	3742	3745	3746										
REPORT	1220#														
RTM	2233#	2281	2282	2283	2284	2285	2286								
SCOPE	1220#	1542	1571	1604	1630	1655	1778	1789	1812	1840	1874	1898	1926	1968	1998
	2034	2083	2144	2190	2281	2282	2283	2284	2285	2286	2297	2339	2379	2418	2467
	2548	2549	2593	2594	2607	2651	2724	2776	2827	2893	2943	2968	3048	3049	3050
	3051	3052	3053	3064	3136	3258	3259	3260	3261	3362	3363	3364	3365		
SETPRI	1220#														
SETTRA	3751#	3752													
SETUP	1220#	1444													
SKIP	1220#														
SLASH	1220#														
SPACE	1220#														
STARS	1220#	1275	1277	1279	1518	1542	1571	1604	1630	1655	1778	1789	1812	1840	1874
	1898	1926	1968	1998	2034	2083	2144	2190	2281	2282	2283	2284	2285	2286	2297
	2339	2379	2418	2467	2548	2549	2593	2594	2607	2651	2724	2776	2827	2893	2943
	2968	3048	3049	3050	3051	3052	3053	3064	3136	3258	3259	3260	3261	3362	3363
	3364	3365	3395	3737	3738	3739	3740	3741	3742	3743	3744	3745	3746	3751	
STM1	3400#	3458	3509	3564	3612	3665									
STM2	3416#	3482	3536	3588	3641	3696									
SWRSU	1220#	1444#													
TOUT	1439#	3805													
TRMTRP	3751#														
TYPBIN	1220#														
TYPDEC	1220#	3395													
TYPNAM	1220#														
TYPNUM	1220#														
TYPOCS	1157#	1220#													
TYPOCT	1220#	3708	3716	3739	3743	3805									
TYPTXT	1220#	1448	3482	3536	3588	3641	3696	3706	3710	3805					
UPDATE	1266#														
ZP23	1783#	1789	1812	1840											
ZP24	1868#	1874	1898	1926											
ZP25	1954#														
ZP26	1961#	1968													
ZP27	1990#	1998													
ZP28	2022#	2034													

```
ZP29     2066#  2083
ZP3E     2281#  2282#  2283#  2284#  2285#  2286#
ZP3EA    2288#  2297
ZP30     2134#  2144
ZP31     2181#  2190
ZP310    2596#  2607
ZP311    2631#  2651
ZP312    2714#  2724
ZP33     2329#  2339
ZP34     2370#  2379
ZP35     2406#  2418
ZP36     2455#  2467
ZP37     2548#  2549#
ZP38     2551#
ZP39     2593#  2594#
ZP42     2760#  2776
ZP43     2813#  2827
ZP44     2882#  2893
ZP45     2935#  2943
ZP46E    3048#  3049#  3050#  3051#  3052#  3053#
ZP47     3055#  3064
ZP6E     3117#  3136
ZP61E    3258#  3259#  3260#  3261#  3362   3363   3364   3365
ZZ1      1672#  1778   1789   1812   1840   1874   1898   1926
Z1       1518#
Z2       1519#  1542
Z3       1560#  1571
Z4       1587#  1604
Z6       1622#  1630
Z7       1648#  1655
$CAL.     748#  3805
$DMAST    914#
$DMD1    1048#
$MMAST    772#
$SCMRE   1279#
$SCMTM   1279#
$SESCA   1220#
$SNEWT   1220#  1518   1542   1571   1604   1630   1655   1778   1789   1812   1840   1874   1898   1926   1968
         1998   2034   2083   2144   2190   2281   2282   2283   2284   2285   2286   2297   2339   2379   2418
         2467   2548   2549   2593   2594   2607   2651   2724   2776   2827   2893   2943   2968   3048   3049
         3050   3051   3052   3053   3064   3136   3258   3259   3260   3261   3362   3363   3364   3365
$SSET    3751#  3752
$SSETM   1444#
$SSKIP   1220#
.EQUAT   1157#  1220
.HEADE   1156#  1207
.KMADR     55#  1439
.KSIS     184#  1454
.LOADL    459#  3805
.LPAIN    209#  3805
.PUTCS    418#  3805
.RESET    329#  3805
.SETTR   1156#
.SETUP   1156#  1442
.SWRHI   1157#  1208
.SWRLO   1208#
```

```
.TRMTR   1156#
.UTK      699#   3805
.$ACT1   1159#   1275
.$APTB   1159#   1279#
.$APTH   1159#   1277
.$APTY   1159#   3745
.$CATC   1157#   1209
.$CMTA   1157#   1279
.$EOP    1158#   3395
.$ERRO   1158#   3738
.$ERRT   1158#   3739
.$INLP    652#   3805
.$MMAC    141#
.$OUTL    610#   3805
.$POWE   1157#   3746
.$RDOC   1156#   1159#   3742
.$READ   1158#   3743
.$SCOP   1158#   3741
.$TLKW    511#   3805
.$TOUT   1439#   3805
.$TRAP   1156#   3751
.$TYPD   1156#   1158#   3740
.$TYPE   1158#   3744
.$TYPO   1157#   3737


. ABS.  043000      000     CON   RW    ABS   LCL   I
        000000      001     CON   RW    REL   LCL   I


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 CRLPGC,CRLPGC/CRF=CRLPAB.MAC,CRLPGC.P11
 RUN-TIME: 42 36 1 SECONDS
 RUN-TIME RATIO: 390/80=4.8
 CORE USED:  42K  (83 PAGES)
```

```
                                ;THIS FILE IS THE SAME AS "CRLPX0.P11" EXCEPT IT IS LOADED INTO 65000
                                ;IT IS ALSO THE SAME AS "DRLPX2.P11" EXCEPT NAME CHANGE "CRLPX2.P11"


                                        .LIST   MC,BIN,BEX,MEB
                                        .NLIST  MD,CND,ME

           177777                       ADDRESS=-1
                                ;       MACRO DEFFINITIONS FOR M8200 AND M8204 MICRO-PROCESSOR
                                ;       INSTRUCTION SET.
                                ;       TO BE USED WITH RSX MACRO-11 ASSEMBLER
                                ;
                                ;       26-MAY-1976
000000'                                 $BEGIN
000000                                  $LOC    42000
                                        .GLOBL  DRLPX2
                                        .ENABL  GBL


                                        ;*
                                        ;*MICRO CODE FOR KMC-11
```

```
                                        ;*THIS CODE WILL BE DOWN LOADED INTO BOTH
                                        ;*KMC-11'S. THE CODE RUNS ASYNCRONOUS TO THE PDP-11 CODE
                                        ;*WE SYNC THROUGH COMMANDS PASSED VIA THE OUT*/IBUS* REGS.
                                        ;*


042000                          DRLPX2: ;JUMP TABLE USED FOR COMMANDS
042000                                  BR      STARTU          ;GOTO START
(2) 042000  100407                      .WORD   .$$$.
042002                                  BR      CMNOP           ;NOP=1
(2) 042002  100420                      .WORD   .$$$.
042004                                  BR      RDSILO          ;=2 READ SILO PUT IN BSEL4
(2) 042004  100430                      .WORD   .$$$.
042006                                  BR      WRSILO          ;=3 READ BSEL4 PUT IN SILO.
(2) 042006  100432                      .WORD   .$$$.
042010                                  BR      RDCMND          ;=4 READ FAST PATH PUT IN BSEL4
(2) 042010  100434                      .WORD   .$$$.
042012                                  BR      WRCMND          ;=5 READ BSEL4, PUT IN FAST PATH.
(2) 042012  100436                      .WORD   .$$$.
042014                                  BR      SAMP            ;=6 TAKE AN A/D SAMPLE
(2) 042014  100440                      .WORD   .$$$.

                                        ;START OF U CODED


042016                          STARTU:
042016                                  MOVE    # 0,BREG
(3) 042016  000400                      .WORD   .$$$.
042020                                  MOVE    BREG,OUT1 <0>   ;CLEAR UNIBUS CSRS
(3) 042020  061220                      .WORD   .$$$.
042022                                  MOVE    BREG,OUT1 <2>
(3) 042022  061222                      .WORD   .$$$.
042024                                  MOVE    BREG,OUT1 <3>
(3) 042024  061223                      .WORD   .$$$.
042026                                  MOVE    BREG,OUT1 <4>
(3) 042026  061224                      .WORD   .$$$.
042030                                  MOVE    BREG,OUT1 <5>
(3) 042030  061225                      .WORD   .$$$.
042032                                  MOVE    BREG,OUT1 <6>
(3) 042032  061226                      .WORD   .$$$.
042034                                  MOVE    BREG,OUT1 <7>
(3) 042034  061227                      .WORD   .$$$.
042036                                  MOVE    BREG,SPAD <6>
(3) 042036  063226                      .WORD   .$$$.

042040                          CMNOP:  MOVE    INP0 <12>,OUT1 <0> ;READ STATUS
(3) 042040  021240                      .WORD   .$$$.
042042                                  MOVE    # 377,BREG
(3) 042042  000777                      .WORD   .$$$.
042044                                  MOVE    BREG,OUT1 <2>   ;INDICATE READY FOR COMMAND.
(3) 042044  061222                      .WORD   .$$$.

042046                          LOOP:   MOVE    INP0 <12>,OUT1 <0> ;READ STATUS
(3) 042046  021240                      .WORD   .$$$.

042050                                  MOVE    INP1 <2>,SPAD <0> ;READ COMMAND REG.
```

```
(3)  042050  123040                        .WORD   .$$$.
496  042052                               BZ      LOOP            ;NO COMMAND THEN LOOP
(2)  042052  101423                       .WORD   .$$$.
497
498  042054                               MOVE    INP1 <2>,SPAD <0> ;RE-READ COMMAND.
(3)  042054  123040                       .WORD   .$$$.
499
500  042056                               BR      SPAD <0>        ;BR BASED ON CMND.
(2)  042056  160600                       .WORD   .$$$.
501                                                               ;NO-USER PROTECTION OFFERED.
502                                                               ;IF YOU ENTER WRONG CODE -
503                                                               ;YOU LOSE.
504
505                                       ;
506                                       ;ROUTINE TO READ THE SILO, PUT IN
507                                       ;*BUS REG 4
508                                       ;CMD=2
509                                       ;
510  042060                       RDSILO: MOVE    INP0 <10>,OUT1 <4> ;READ SILO.
(3)  042060  021204                       .WORD   .$$$.
511                                                               ;WRITE *BUS
512  042062                               BR      CMNOP           ;RETURN.
(2)  042062  100420                       .WORD   .$$$.
513
514                                       ;
515                                       ;ROUTINE TO WRITE SILO, READ DATA FROM
516                                       ;*BUS REG 4
517                                       ;CMD=3
518                                       ;
519
520  042064                       WRSILO: MOVE    INP1 <4>,OUT0 <10> ;READ DATA IN *BUS
(3)  042064  122110                       .WORD   .$$$.
521                                                               ;WRITE SILO.
522  042066                               BR      CMNOP
(2)  042066  100420                       .WORD   .$$$.
523
524                                       ;
525                                       ;ROUTINE TO READ FAST PATH (CMND) REG.
526                                       ;PUT IN *BUS REG 4
527                                       ;CMD=4
528                                       ;
529
530  042070                       RDCMND: MOVE    INP0 <11>,OUT1 <4> ;READ FAST PATH
(3)  042070  021224                       .WORD   .$$$.
531                                                               ;WRITE *BUS.
532  042072                               BR      CMNOP           ;RETURN
(2)  042072  100420                       .WORD   .$$$.
533
534                                       ;
535                                       ;ROUTINE TO WRITE FAST PATH (CMND) REG.
536                                       ;TAKE DATA FROM *BUS REG 4.
537                                       ;CMD=5
538                                       ;
539
540  042074                       WRCMND: MOVE    INP1 <4>,OUT0 <11> ;READ DATA IN *BUS
(3)  042074  122111                       .WORD   .$$$.
```

```
                                                        ;WRITE INTO FAST PATH.
042076                                  BR      CMNOP   ;RETURN.
(2) 042076  100420                      .WORD   .$$$.

                                        ;
                                        ;THIS ROUTINE TAKES AN A/D SAMPLE.
                                        ;CALL= CMND 6 IN BSEL2
                                        ;THESE REGS. MUST BE SET UP IN ADVANCE.
                                        ;BSEL 3 MUST CONTAIN READ CODE FOR A/D BUFFER.
                                        ;BSEL 4,5 MUST CONTAIN  A/D CSR SETTING.
                                        ;BSEL 6 MUST CONTAIN WRITE CODE FOR A/D CSR
                                        ;BSEL 7 MUST CONTAIN READ CODE FOR A/D CSR
                                        ; BSEL 3,6,7 WILL REMAIN  UNEFFECTED.
                                        ; BSEL 4,5 WILL CONTAIN A/D SAMPLE.
                                        ;BSEL2 WILL CONTAIN CODE 377 WHEN DONE.

042100                                  WTMC    SAMP
(4) 042100  020640                      .WORD   .$$$.
(3) 042102  103040                      .WORD   .$$$.
042104                                  MOVE    INP1 <6>,OUTO <11>      ;SEND A/D WRITE CODE.
(3) 042104  122151                      .WORD   .$$$.
042106                                  WTMC    SAMP1
(4) 042106  020640                      .WORD   .$$$.
(3) 042110  103043                      .WORD   .$$$.
042112                                  MOVE    INP1 <4>,OUTO <11>      ;SEND LOW BYTE CSR INFO.
(3) 042112  122111                      .WORD   .$$$.
042114                                  WTMC    SAMP2
(4) 042114  020640                      .WORD   .$$$.
(3) 042116  103046                      .WORD   .$$$.
042120                                  MOVE    INP1 <5>,OUTO <11>      ;SEND HIGH BYTE CSR INFO.
(3) 042120  122131                      .WORD   .$$$.
042122                                  WTMC    SLOOP
(4) 042122  020640                      .WORD   .$$$.
(3) 042124  103051                      .WORD   .$$$.
042126                                  MOVE    INP1 <7>,OUTO <11>      ;SEND READ CODE TO GET A/D CSR.
(3) 042126  122171                      .WORD   .$$$.
042130                                  WTMC    SAMP3
(4) 042130  020640                      .WORD   .$$$.
(3) 042132  103054                      .WORD   .$$$.
042134                                  WTMM    SLOOP1
(4) 042134  020640                      .WORD   .$$$.
(4) 042136  061620                      .WORD   .$$$.
(3) 042140  103056                      .WORD   .$$$.
042142                                  MOVE    INP0 <11>,BREG
(3) 042142  020620                      .WORD   .$$$.
042144                                  MOVE    BREG,SPAD <0>
(3) 042144  063220                      .WORD   .$$$.
042146                                  WTMM    SLOOP2
(4) 042146  020640                      .WORD   .$$$.
(4) 042150  061620                      .WORD   .$$$.
(3) 042152  103063                      .WORD   .$$$.
042154                                  MOVE    INP0 <11>,BREG
(3) 042154  020620                      .WORD   .$$$.
042156                                  BB7     CMNOP           ;ABORT IF A/D BIT 15=1
```

```
    (2)  042156  103420                        .WORD   .$$$.
    582  042160                                MOVE    SPAD <0>,BREG
    (3)  042160  060600                        .WORD   .$$$.
    583  042162                                BB7     LOPE
    (2)  042162  103473                        .WORD   .$$$.
    584  042164                                BR      SLOOP                   ;IF A/D NOT DONE,EXIT.
    (2)  042164  100451                        .WORD   .$$$.
    585  042166                          LOPE: MOVE    INP1 <3>,OUT0 <11>      ;ISSUE READ A/B BUFFER.
    (3)  042166  122071                        .WORD   .$$$.
    586  042170                                WTMM    SLOOP3
    (4)  042170  020640                        .WORD   .$$$.
    (4)  042172  061620                        .WORD   .$$$.
    (3)  042174  103074                        .WORD   .$$$.
    587  042176                                MOVE    INP0 <11>,OUT1 <4>
    (3)  042176  021224                        .WORD   .$$$.
    588  042200                                WTMM    SLOOP4
    (4)  042200  020640                        .WORD   .$$$.
    (4)  042202  061620                        .WORD   .$$$.
    (3)  042204  103100                        .WORD   .$$$.
    589  042206                                MOVE    INP0 <11>,OUT1 <5>
    (3)  042206  021225                        .WORD   .$$$.
    590  042210                                BR      CMNOP
    (2)  042210  100420                        .WORD   .$$$.
    591  042212  177777                        .WORD   -1
    592          000001                        .END
```

```
ADDRES= 177777          SAMP    042100          .ADDWC= 000020          .DMEM = 002400          .SELB = 000220
CLK   = 000020          SAMP1   042106          .AND  = 000260          .DNOP = 000000          .SIMM = 000000
CMNOP   042040          SAMP2   042114          .BB0  = 002000          .DOUT0= 002000          .SIN0 = 020000
DRLPX2  042000 G        SAMP3   042130          .BB1  = 002400          .DOUT1= 001000          .SIN1 = 120000
LOOP    042046          SLOOP   042122          .BB4  = 003000          .DSPAD= 003000          .SMEM = 040000
LOPE    042166          SLOOP1  042134          .BB7  = 003400          .DSPBR= 003400          .SUB  = 000340
MARHLD= 000000          SLOOP2  042146          .BC   = 001000          .DO   = 000400          .SUBWC= 000040
MARINC= 014000          SLOOP3  042170          .BR   = 000400          .FO   = 000020          .SUB2C= 000360
MARLD = 010000          SLOOP4  042200          .BSBRG= 160000          .INC  = 000060          .SO   = 020000
MARLDX= 004000          STARTU  042016          .BSIMM= 100000          .LORN = 000240          .XOR  = 000320
PAGE0 = 000000          WRCMND  042074          .BSMEM= 140000          .MINUS= 000360          .$$$. = 100420
PAGE1 = 001000          WRSILO  042064          .BZ   = 001400          .MO   = 004000          ..LOC = 042040
PAGE2 = 002000          $$$SER= 000001          .CO   = 000400          .OR   = 000300          .2A   = 000120
PAGE3 = 003000          .     = 042214          .DBR  = 000400          .PLUS = 000000          .2AWC = 000140
RDCMND  042070          .ADC  = 000100          .DBRSH= 001400          .SBREG= 060000
RDSILO  042060          .ADD  = 000000          .DEC  = 000160          .SELA = 000200

. ABS.  042214    000       OVR   RW    ABS   LCL   D
        000000    001       CON   RW    ABS   LCL   I
ABCODE  004000    002       CON   RW    REL   LCL   I


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 CRLPX2,CRLPX2=CRLPX2
 RUN-TIME: 3 3 0 SECONDS
 RUN-TIME RATIO: 49/7=6.5
 CORE USED:  42K  (83 PAGES)
```