LPA/LPS

DIAGNOSTIC TEST

MD-11-DRLPH-A

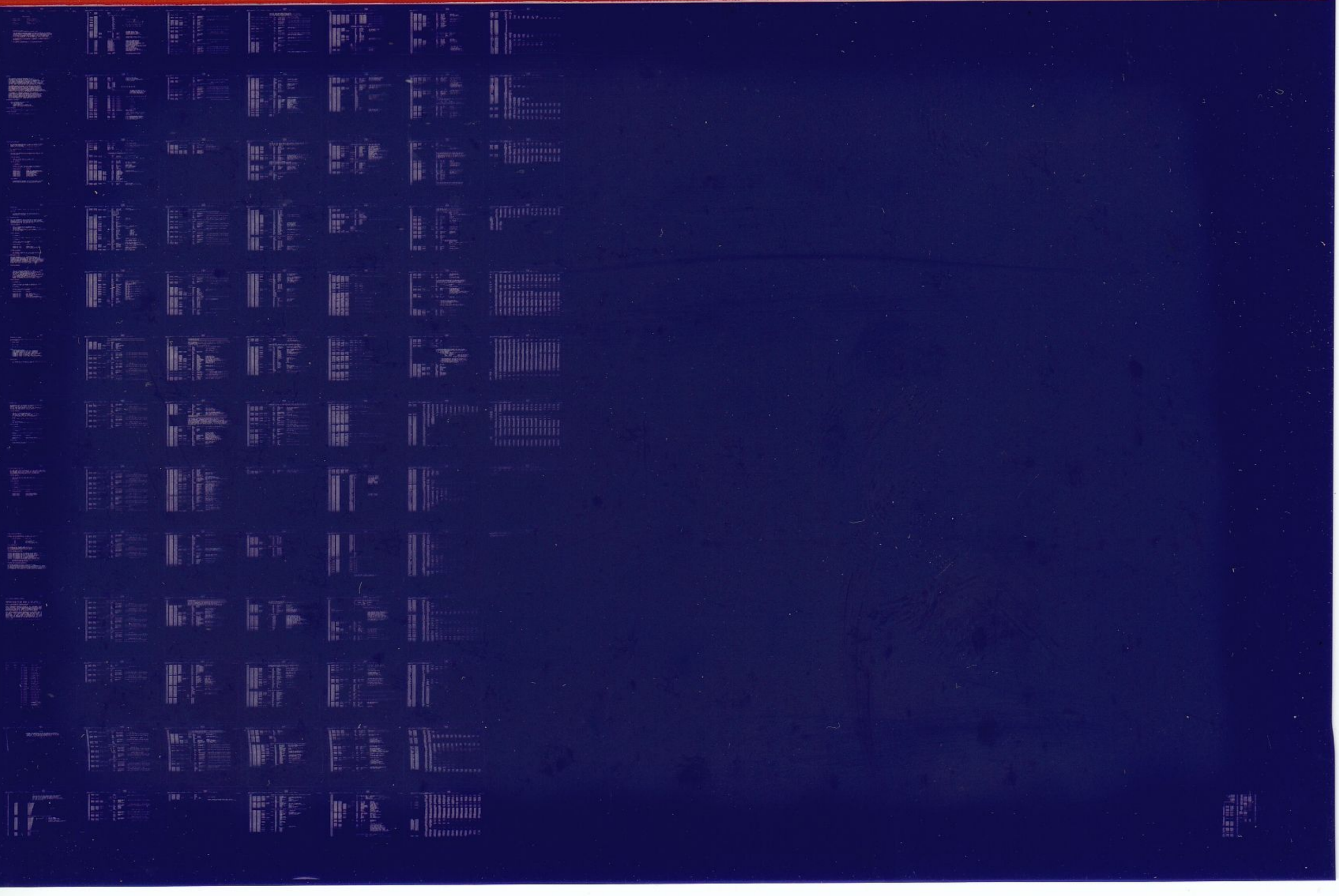
EP-DRLPH-A-DL

FICHE 1 OF 1

MAR 1978

digital

MADE IN USA



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DRLPH-A-D

PRODUCT NAME: LPA/LPS DIAGNOSTIC TEST

DATE: JANUARY 1978

MAINTAINER: DIAGNOSTIC GROUP

1. ABSTRACT

SEQ 0002

THIS DIAGNOSTIC TESTS AND EXERCISES THE "LPS11".
WHEN LOADED WILL TYPE OUT THE PROGRAM TITLE.
A MESSAGE IS THEN PRINTED GIVING THE LETTER DESIGNATORS
TO BE TYPED TO RUN ANY ONE OF THE FIVE (5) SEPERATE TESTS OF WHICH
THIS PROGRAM IS COMPRISED. THE PROGRAM THEN TYPES A 'CR .' AND
THEN WAITS IN A KEYBOARD MONITOR MODE FOR A LETTER TO BE TYPED.
ALTHOUGH THESE TESTS MAY BE RUN IN ANY ORDER IT IS IMPERATIVE
THAT THE 'LOGIC' TESTS ARE RUN FIRST AND PROVED FULLY OPERATIONAL.

THE PROGRAM IS SET UP TO GIVE THE OPERATOR AS MUCH CONTROL
OVER THE PROGRAM AS POSSIBLE VIA THE TELETYPE.TYPING A '↑C'
(OBTAINED VIA TYPING THE 'CNTR' AND 'C' KEYS SIMULTANEOUSLY)
WHILE RUNNING ANY TEST WILL ENABLE THE PROGRAM TO RETURN TO
THE KEYBOARD MONITOR AND AWAIT A NEW LETTER DESIGNATOR TO BE
TYPED. TYPING A '↑A' WHILE IN MONITOR MODE WILL ENABLE THE LETTER
DESIGNATORS TO BE RETYPED. IF RUNNING ON A NON-SWITCH REGISTER CPU,
TYPING A 'CTRL G' WILL ALLOW THE CHANGING OF A SOFTWARE SWITCH REGISTER.

THIS PROGRAM IS A MODIFIED VERSION OF "MD-11-DZLPC-C".
IT WAS MODIFIED TO ENABLE THE OPERATOR TO CHECK OUT THE LPS11
OPTION WHEN IT IS ON THE LPA11-KX I/O BUS. NO RECABLING IS
NEEDED. SOME TEST DONE IN THE ORIGINAL DIAGNOSTIC SUCH AS
ARBITRATION TEST, WERE DELETED AS THEY COULD NOT BE CHECKED.
IF THIS DIAGNOSTIC DOESN'T FIND A SUSPECTED PROBLEM, YOU MAY
HAVE TO RUN "DZLPC". YOU SHOULD RUN "MD-11-DRLPA" BEFORE
RUNNING THIS DIAGNOSTIC. PLEASE READ SECTION 15.

2. REQUIREMENTS (EQUIPMENT)

A. PDP-11 COMPUTER WITH 16K OF MEMORY
B. TELETYPE (OR EQUIVALENT)
C. LPS11 OPTION BOX WITH:
LPSAD12 SIMPLE A TO D CONTROL AND
LPSADNP DMA A TO D CONTROL AND/OR
LPSSH DUAL SAMPLE AND HOLD CONTROL.

3. LOADING PROCEDURE

A. USE STANDARD PROCEDURE FOR LOADING BINARY TAPES.

4. STARTING PROCEDURE

THE STARTING ADDRESS OF THIS DIAGNOSTIC IS 200

5. CONSOLE SWITCH SETTINGS

A. ALL SWITCHES SHOULD BE DOWN (0) WHEN THE PROGRAM IS STARTED.
B. REFER TO THE INDIVIDUAL TESt DESCRIPTIONS FOR APPLICABLE CONSOLE SWITCH SETTINGS
C. REFER TO 14. FOR SOFTWARE SWITCH REGISTER OPERATION.

* TYPE 'CARRIAGE RETURN' (CR) TO TERmINATE ALL INPUT DATA.

6. A TO D LOGIC TEST

A. THE LOGIC TEST CONSISTS OF 26 SUBTESTS WHICH CHECK OUT THE 'LPS A TO D' LOGIC. EACH TEST IS LOOPED '2048' TIMES TO tEST LOGIC RELIABILTY.

B. STARTING SEQUENCE

1. TYPE 'A' TO RUN THE A TO D LOGIC' TEST.
2. THE PROGRAM WILL THEN EXECUTE THE A TO d LOGIC TEST.

C. CONTROL SWITCHES

1. ↑C (CONTROL C)

TYPING A ↑C AT ANY TIME WILL ENABLE THE PROGRAM TO EXIT THE 'A TO D LOGIC' TEST AND RETURN TO THE MONITOR.

2.

CONSOLE SWITCH	FUNCTION
CONSOLE SW11=0	NORMAL RUN (2048 PASSES/TEST)
CONSOLE SW11=1	SUPPRESS SUBPROGRAM ITERATIONS
CONSOLE SW13=0	PRINT ERROR MESSAGE
CONSOLE SW13=1	INHIBIT ERROR MESSAGES
CONSOLE SW14=1	RUN SCOPE MODE
CONSOLE SW14=0	INHIBIT SCOPE MODE
CONSOLE SW15=0	CONTINUE AFTER ERROR
CONSOLE SW15=1	HALT ON ERROR

D. LOGIC ERRORS

ON ENCOUNTERING AN ERROR (DATA SWITCHES DOWN) THE ERROR ADDRESS AND THE CONTENTS OF THE A/D STATUS REGISTER AND A/D BUFFER AND DMA STATUS REGISTER ARE TYPED OUT.

E. RESTRICTIONS

DO 'NOT' CONNECT EXTERNAL SYNC FOR THE LOGIC TEST.

F. TEST TIME

IT TAKES APPROXIMATELY 5 SEC. TO RUN THE LOGIC TEST AND TYPE "END PASS". FURTHER ITERATIONS TAKE APPROXIMATELY 5 MINUTES.

7. A TO D CALIBRATION TEST

A. THE 'A/D CALIBRATION' TEST IS DESIGNED TO ACCEPT AN INPUT FROM THE TELETYPE TO INDICATE THE TYPE OF SYNC (EXTERNAL, INTERNAL OR LPS CLOCK TO BE USED AND THEN TAKES CONTINUOUS CONVERSIONS USING THE 'CH.' SELECTED VIA THE CONSOLE SWITCHES. THESE SETTINGS MAY BE CHANGED AT ANY TIME.

B. STARTING SEQUENCE

1. TYPE 'C' TO RUN THE A/D CALIBRATION TEST.
2. THE TEST HEADER PLUS A REQUEST FOR A SYNC TYPE WILL THEN BE TYPED.
3. TYPE IN THE DESIRED SYNC, 'I' FOR INTERNAL, 'E' FOR EXTERNAL 'C' FOR LPS CLOCK (IF INSTALLED) FOLLOWED BY 'CR'.
4. THE TEST WILL START.

C. CONTROL SWITCHES

1. ↑A (CONTROL A)

TYPING ↑A WILL ENABLE A NEW SYNC TYPE TO BE ENTERED.

2. ↑C (CONTROL C)

TYPING ↑C WILL CAUSE THE PROGRAM TO EXIT THE CALIBRATION TEST AND RETURN TO THE MONITOR.

3.

CONSOLE SWITCH	FUNCTION
CONSOLE SW '0-5'	CHANNEL SELECT
CONSOLE SW '10=0'	DISPLAY VALUE IN LED DISPLAY
CONSOLE SW '10=1'	PRINT CONVERSION VALUE

D. CALIBRATION ERRORS

THE PRINTOUT 'ERROR BIT SET' WILL OCCUR WHILE RUNNING WITH EXTERNAL SYNC IF THE SYNC FREQUENCY IS TO FAST.

8. A TO D REPEATIBILITY TEST

A. THIS TEST REQUESTS A CH.(S) AND A COUNT SPREAD OF '1-4' AND MODE OF OPERATION TO BE TYPED IN BY THE OPERATOR. A SERIES OF'512' CONVERSIONS ARE THEN TAKEN ON THE INPUT CH.(S) CONVERSIONS ARE THEN AVERAGED OUT AND IF THE COUNT SPREAD IS FOUND TO BE GREATER THAN REQUEST, THE RESULTS OF THE CONVERSIONS ARE TYPED OUT. A SINGLE CHANNEL OR A SERIES OF CHANNELS MAY BE TESTED VIA TYPING EITHER 'N(CR)' TO SELECT A SINGLE CHANNEL OR 'N,N(CR)' TO TEST A SERIES OF CHANNELS.

B. STARTING SEQUENCE

1. TYPE 'D' TO RUN THE 'REPEATIBILITY' TEST.
2. A REQUEST IS THEN MADE FOR CH.(S) TO BE TESTED AND COUNT SPREAD (RANGE IN WHICH ALL 512 COUNTS MUST FALL FOR THE CH. TO BE CONSIDERED ACCEPTABLE).
3. A REQUEST IS THEN MAKE FOR MODE OF OPERATION (NPR OR BR) TYPE 'I' FOR PROGRAM SAMPLING OR TYPE 'B' FOR BURST NPR OR 'D' FOR DUAL MODE SAMPLING IF INSTALLED.
4. IF THE CHANNEL IS FOUND TO BE WITHIN THE SELECTED COUNT SPREAD, THE PROGRAM WILL EITHER CONTINUE TO THE NEXT CHANNEL IF SELECTED OR RETEST THE CURRENT CHANNEL.

C. CONTROL SWITCHES

1. ↑A (CONTROL A)

TYPING A ↑A WHILE THE PROGRAM IS RUNNING WILL ENABLE A NEW CH.(S) AND COUNT SPREAD TO BE SELECTED.

2. ↑C (CONTROL)

TYPING ↑C WILL CAUSE THE PROGRAM TO EXIT THE 'REPEATIBILITY' TEST AND RETURN TO THE MONITOR.

3.

CONSOLE SWITCHES	FUNCTION
CONSOLE SW 10=0	PRINT ERRORS ONLY
CONSOLE SW 10=1	PRINT OUT ALL CONVERSIONS
CONSOLE SW 13=0	PRINT ERRORS
CONSOLE SW 13=1	INHIBIT ERROR PRINTOUTS
CONSOLE SW 15=0	DO NOT HALT UPON REPEATIBILITY REPORT
CONSOLE SW 15=1	HALT UPON REPEATIBILITY REPORT

D. REPEATIBILITY ERRORS

ON ENCOUNTERING AN ERROR (CONSOLE SWITCHES DOWN) THE ERROR DATA IS TYPED OUT.

1. ERROR FORMAT

```
CH.      HI       AV       LO
 A        B        C        D

LO  -5  -4  -3  -2  -1   0  +1  +2  +3  +4  +5  HI
E   F   G   H   I   J   K   L   M   N   O   P   Q
```

WHERE:

A=CHANNEL BEING TESTED
B=THE HIGHEST READING OF THE '512' CONVERSIONS
C=THE AVERAGE READING OF THE '512' CONVERSIONS
D=THE LOWEST READING OF THE '512' CONVERSIONS
E=NUMBER OF COUNTS 'OUT OF RANGE' LOWER THAN 5 COUNTS
F-J=NUMBER OF COUNTS IN EACH PART LOWER THAN AVERAGE.
K=NUMBER OF COUNTS AT AVERAGE OF THE '512'
L-P=NUMBER OF COUNTS IN EACH PART HIGHER THAN AVERAGE.
Q=NUMBER OF COUNTS 'OUT OF RANGE' HIGHER THAN 5 COUNTS

E. RESTRICTIONS

1. IF A SECOND CH. IS ENTERRED, IT MUST BE LARGER THAN THE FIRST CH. OR THE INPUT WILL NOT BE ACCEPTED.

9. A TO D RECOVERY TEST

SEQ 0007

A. THE "RECOVERY TEST" IS DESIGNED TO dETERMINE THE RECOVERY CAPABILITY OF THE 'LPS A TO D'. THE TEST REQUESTS FOR TWO (2) CH. INPUTS TO BE TYPED IN. THE TEST THEN TAKES A SERIES OF SIXTEEN (16) CONVERSIONS (8 ON EACH CH.) AND THEN TYPES OUT THE '8' CONVERSION VALUES IN THE ORDER THEY WERE TAKEN ON THE SECOND CH.

B. STARTING SEQUENCE

1. TYPE 'E' TO rUN THE RECOVERY TESt.
2. A REQUEST IS THEN MADE FOR THE CH.S TO BE TESTED.
3. TYPE 'N,N (CR)' WHERE 'N' IS ANY CH.
4. THE PROGRAM WILL THEN TAKE CONTINUOUS CONVERSIONS TYPING OUT THE CONVERSION VALUES FOR THE SECOND CH.

EXAMPLE:

```
CH. A   XXXXXX  XXXXXX  XXXXXX  XXXXXX  XXXXXX  XXXXXX  XXXXXX  XXXXXX
```

WHERE:

A=TO THE SECOND CH.
X=TO THE '8' CONVERSIONS TAKEN ON THAT CH.

C. CONTROL SWITCHES

1. ↑A (CONTROL A)

TYPING A '↑A' WILL ENABLE A NEW SET OF CH.S BE ENTERED.

2. ↑ C (CONTROL C)

TYPING A '↑C' WILL ENABLE THE PROGRAM TO rETURN TO tHE MONITOR.

3.

CONSOLE SWITCHES	FUNCTION
CONSOLE SW 10 = 0	PRINT RECOVERY REPORT
CONSOLE SW 10 = 1	INHIBIT PRINTING OF RECOVERY REPORT

D. RESTRICTIONS

DO NOT EXECUTE THIS TEST IF THE WIDE BAND JUMPERS HAVE BEEN REMOVED.

10. DUAL MODE DYNAMIC LOGIC TEST

SEQ 0008

A. THE 'DUAL MODE' TEST IS DESIGNED TO TEST THE DYNAMIC DUAL SAMPLE AND HOLD LOGIC OF THE LPS. A SAMPLE IS TAKEN FROM A CHANNEL PAIR (IE. CHANNEL 0 AND 10) NOT IN DUAL MODE, ANOTHER SAMPLE IS TAKEN FROM A CHANNEL PAIR USING 'DUAL MODE'. THE RESULTING CONVERSIONS ARE THEN COMPARED FOR VALUE AND SIMILARITY.

B. STARTING SEQUENCE

1. TYPE 'F' TO RUN THE 'DUAL MODE LOGIC TEST'.
2. THE PROGRAM WILL THEN EXECUTE THE DUAL MODE TEST ON CHANNEL PAIRS 0-3.

C. CONTROL SWITCHES

1. ↑C (CONTROL C)

TYPING ↑C WILL CAUSE THE PROGRAM TO EXIT TO THE MONITOR.

2. ↑A (CONTROL A)

TYPING ↑A WILL CAUSE THE RESTARTING OF THE DUAL MODE LOGIC TEST.

3.

CONSOLE SWITCHES	FUNCTION
CONSOLE SW13=0	PRINT ERROR MESSAGES
CONSOLE SW13=1	INHIBIT ERROR MESSAGES
CONSOLE SW15=0	CONTINUE AFTER ERROR
CONSOLE SW15=1	HALT ON ERROR

11. MANUAL STARTING ADDRESSES

EXTERNAL STARTING ADDRESSES OF THE BASIC TEST HAVE BEEN
PROVIDED FOR THE USER TO MANUAL START THESE TESTS.

STARTING ADDRESS	BASIC TEST
204	A TO D LOGIC TEST
210	DMA LOGIC TEST
214	DUAL SAMPLE LOGIC TEST
220	A TO D CALIBRATION TEST

12. MISC. INFORMATION

THE PROGRAM CAN BE CHAINED UNDER XXDP/ACT-11
IF THE PROGRAM WAS LOADED FROM ACT-11 OR DDP, THE A TO D LOGIC TEST
AND (IF INSTALLED) DMA LOGIC TEST WILL BE RUN.
THIS PROGRAM DOES NOT HAVE THE "APT" HOOKS.

13. OPERATOR VARIABLE LOCATIONS

LOCATION 1000 CONTAINS THE LPS STARTING DEVICE ADDRESS <170400>
LOCATION 1002 CONTAINS THE LPS STARTING VECTOR <340>
LOCATION 1004 CONTAINS THE LPS A TO D BR LEVEL <6><300>
LOCATION 1010 CONTAINS THE + OR - 37 COUNT SPREAD FOR THE DUAL SAMPLE TEST.
LOCATION 1012 CONTAINS THE TTY FILLER CHARACTER COUNT
LOCATION 1014 CONTAINS THE TTY FILLER CHARACTER
LOCATION 170 CONTAINS THE SOFTWARE SWITCH REGISTER VALUE
LOCATION 172 CONTAINS THE SOFTWARE DISPLAY REGISTER VALUE.

NOTE: IF LOCATIONS 1000 OR 1002 ARE CHANGED, THE TEST
MUST BE RE-INITILIZED AT 200.

14. SOFTWARE SWITCH REGISTER OPERATION

THE PROGRAM SUPPORTS NON-SWITCH REGISTER CPU TYPES.
THIS IS ACCOMPLISHED BY TYPING A "CTRL G". THE RESPONSE WILL REPORT
THE VALUE OF THE OLD SOFTWARE SWITCH REGISTER AND WAIT FOR A NEW VALUE.
THE OPERATOR NOW INPUTS THE NEW VALUE AND TERMINATES IT WITH A "CR".
IF THE OPERATOR TYPES A "CR" WITH NO INPUT, THE SWITCH REGISTER IS SET TO 0.
UPON TERMINATING, THE PROGRAM WILL RESUME RUNNING THE APPROPATE TEST.

15. 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:

SEQ 0011

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

```
                        .REM    [

                                LPA.MAC

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

                                GOOD LUCK !

[
.GLOBL  DRLPX2
```

```
                                .REM    !

                                THIS IS A LIST OF TESTS DELETED FROM THIS DIAGNOSTIC.
                                THESE TESTS COULD NOT BE DONE THROUGH THE LPA-11
                                TEST THAT A TO D INTERRUPT AT LEVEL INDICATED-1
                                TEST THAT A TO D DOES NOT INTERRUPT AT LEVEL INDICATED
                                TEST THAT THE A TO D DOEs NOT INTERRUPT AT LEVEL INDICATED +1
                                ALL DMA TEST DELETED

                                !

100000                          BIT15=100000
040000                          BIT14=40000
020000                          BIT13=20000
010000                          BIT12=10000
004000                          BIT11=4000
002000                          BIT10=2000
001000                          BIT9=1000
000400                          BIT8=400
000200                          BIT7=200
000100                          BIT6=100
000040                          BIT5=40
000020                          BIT4=20
000010                          BIT3=10
000004                          BIT2=4
000002                          BIT1=2
000001                          BIT0=1

                                ;SWITCH REGISTER DEFINITIONS AND FUNcTIONS:

100000                          SW15=100000                     ;=1, HALT ON ERROR
040000                          SW14=40000                      ;=1, LOOP ON CURRENT TEST
020000                          SW13=20000                      ;=1, SUPPRESS ERROR TYPEOUT
010000                          SW12=10000
004000                          SW11=4000                       ;=1, SUPPRESS 'SUBPROGRAM' ITERATIONS
002000                          SW10=2000                       ;=1, FORCE TYPEOUT (REPEATIBILITY)
001000                          SW09=1000
000400                          SW08=400
000200                          SW07=200
000100                          SW06=100
000040                          SW05=40
000020                          SW04=20
000010                          SW03=10
000004                          SW02=4
000002                          SW01=2
000001                          SW00=1

                                ;REGISTER DEFiNITIONS

000000                          R0=%0
000001                          R1=%1
000002                          R2=%2
000003                          R3=%3
000004                          R4=%4
```

```
                    000005                  R5=%5
                    000006                  SP=%6
                    000007                  PC=%7

                                            ;LOAD TRAP CATCHER INTO LOC'S 0-1000
                    000000                          .=0
000000  000000                              HALT
000002  000000                              HALT
                    000024                          .=24
000024  011232                              PWRFAL                  ;POWER FAIL HANDLER
000026  000340                              340
                    000060                          .=60
000060  010130                              XTTYIN                  ;TELEPRINTER KEYBOARD ROUTINE
000062  000340                              340
                    000030                          .=30
000030  011334                              EMTSRV                  ;EMT TRAP, EMT DISPATCH SERVICE
000032  000340                              340
000034  012664                              LOGERR                  ;TRAP TRAP, LOGIC ERROR TRAP
000036  000340                              340
                    000046                          .=46
000046  002022                              LOGICAL
                    000052                          .=52
000052  000000                              0
                    000170                          .=170
000170  000000                  SOFTSW:     0                       ;SOFTWARE SWITCH VALUE
000172  000000                  SOFTDI:     0                       ;SOFTWARE DISPLAY VALUE
000174  000137  001546                      JMP     MONITR          ;PROGRAM 'RESTART' ADDRESS
000200  000137  001154                      JMP     INIT            ;INITIALIZATION ADDRESS
000204  000137  002040                      JMP     ADTEST          ;MANUAL START OF A TO D LOGIC
000210  000240                              NOP
000212  000240                              NOP
000214  000137  006772                      JMP     DSHTST          ;MANUAL START OF DUAL SAMPLE
000220  000137  005246                      JMP     MANCAL          ;MANUAL START OF THE A TO D CALIBRATION

                                            ;TRAP EQUIVALENCE TABLE:

                    104400                  ERROR=TRAP              ;LOGIC TEST ERROR ROUTINE
                    104000                  PRINT=EMT               ;MESSAGE PRINTER ROUTINE
                    104001                  DECOCT=EMT+1            ;OCTAL CONVERSIN ROUTINE
                    104002                  SCOPE=EMT+2             ;LOGIC TEST SCOPE SUBROUTINE (4000)
                    104003                  SCOPE1=EMT+3            ;LOGIC TEST SCOPE SUBROUTINE (10)
                    104004                  CMPUTE=EMT+4            ;A/D AVERAGING ROUTINE
                    104005                  CATORIZ=EMT+5           ;ROUTINE TO CALCULATE THE COUNT SPREAD
                    104006                  BINDEC=EMT+6            ;BINARY TO DECIMAL CONVERSION ROUTINE
                    104007                  SPACE=EMT+7             ;TYPE 'N' SPACES
                    104010                  PRTOCT=EMT+10           ;OCTAL PRINT ROUTINE
                    104011                  TTYIN=EMT+11            ;TELETYPE INPUT ROUTINE
                    104012                  PRTAVG=EMT+12           ;GAIN AVERAGE PRINT ROUTINE
                    104013                  TSTTKS=EMT+13           ;SUBROUTINE TO TEST FOR KEYBOARD FLAG
                    104014                  SCOPE0=EMT+14           ;SCOPE SUBROUTINE (1)

                    001000                          .=1000
001000  170400                  LPSADD:     170400                  ;LPS STARTING DEVICE ADDRESS
001002  000340                  LPSVCT:     340                     ;LPS STARTING DEVICE VECTOR
```

```
                                ADBRL:  300                     ;LPS A TO D BR LEVEL
                                PDPDLY: 1                       ;CHANGED BY THE PROGRAM
                                AMASK:  37                      ;+ OR - 37 COUNTS FOR DUAL SAMPLE LOGIC TEST
                                FILLS:  2                       ;NUMBER OF FILLER CHARS AFTER LF
                                FILCHR: 0                       ;FILLER CHARACTER

                                PSW:    177776
                                TKS:    177560
                                TKB:    177562
                                TPS:    177564
                                TPB:    177566
                                SWR:    177570          ;OR 170 IF NO SWR CPU TYPE
                                DISPLA: 177570          ;OR 172 IF NO SWR CPU TYPE
                                PASSCT: 0
                                ICOUNT: 4000
                                ;
                                ;ADDRESS OF KMC-11 OF LPA-11    THE ADDR FOR KMADO MAY BE
                                ;                               CHANGED BY THE USER TO rEFLECT
                                ;                               A DIFFERENT KMC-11 ADDR. THE
                                ;                               REST OF THE ADDRESSES WILL
                                ;                               BE CHANGED BY THE PROGRAM.
                                ;
001040                          LPCI:
001040  170460                 KMADO:  .WORD   170460          ;BASE KMC ADDR. MAY BE PATCHED BY USER.

001042                          LPMR:
001042  170461                 KMAD1:  .WORD   170460+1                ;>DO NOT        <;KMC-CSR ADDR
001044                          LPCO:
001044  170462                 KMAD2:  .WORD   170460+2                ;>PATCH         <;
001046                          LPSO:
001046  170463                 KMAD3:  .WORD   170460+3                ;>THIS AREA     <
001050                          LPADL:
001050  170464                 KMAD4:  .WORD   170460+4                ;
001052                          LPADH:
001052  170465                 KMAD5:  .WORD   170460+5                ;>DO NOT        <
001054                          LPMS1:
001054  170466                 KMAD6:  .WORD   170460+6                ;>PATCH         <
001056                          LPMS2:
001056  170467                 KMAD7:  .WORD   170460+7                ;>THIS AREA     <

001060  000300                 VECTOR: .WORD   AVECT1&777      ;BASE VECTOR OF KMC
001062  000304                 VECTPS: .WORD   4+AVECT1&777    ;VECOTR ADDR.+2

001064  000004                 VERSN:  .WORD   4               ;CURRENT VERSION NUMBER OF MICROCODE.

001066  000000                 .DVLS:  .WORD   0               ;/DEVICE LIST OF I/O ADDR. DEFINED
001070  000020                         .BLKW   16.             ;/BY INIT.

001130  170400                 ADCS:   170400                  ;A TO D STATUS/CONTROL REGISTER
001132  170401                 ADCS1:  170401                  ;A TO D STATUS REGISTER <HIGH BYTE>
001134  170402                 ADDBR:  170402                  ;A TO D CONVERTED VALUE <READ>
                                                               ;A TO D LED DISPLAY LIGHTS <WRITE>
001136  170404                 CSR:    170404                  ;CLOCK STATUS REGISTER
```

```
                                CSB:    170406                  ;CLOCK PRESET BUFFER
                                UADMR:  170436                  ;A TO D DIRECT MEMORY ACCESS ADDRESS UNUSED
                                ADINT:  340                     ;A TO D INTERRUPT VECTOR
                                ADINT1: 342                     ;
                                AVECT1= 300
                                $VECT1: .WORD   AVECT1
                                DMK2:   0                       ;TEMP STORAGE


                                ;THIS ROUTINE IS EXECUTED ON LOADING THE PROGRAM
                                ; OR RESTARTING AT LOCATION 174

                                        INIT:
                                        MOV     STACK,SP        ;INIT STACK POINTER=1000
                                        MOV     #340,@PSW
                                        ;
                                        ;THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
                                        ;

                                        MOV     R0,-(SP)
                                        MOV     R1,-(SP)
                                        MOV     KMAD0,R0        ;GET KMC-11 ADDRESS.
                                        MOV     #KMAD1,R1       ;GET ADDR. OF ADDR. LIST.

                                64$:    INC     R0              ;UPDATE ADDR.
                                        MOV     R0,(1)+         ;WRITE ADDR.
                                        CMP     R1,#KMAD7+2     ;DONE ALL ADDRESSES?
                                        BNE     64$             ;NO - DO NEXT ADDR.
                                        CLR     .DVLS           ;CLR ADDR. LIST.
                                        MOV     (SP)+,R1
                                        MOV     (SP)+,R0

                                        JSR     PC,$RESET       ;CLEAR THE WORLD
                                        MOV     LPSADD,ADCS     ;HOUSEKEEP THE ADDRESS AND VECTOR
                                        MOV     LPSADD,ADCS1
                                        MOV     LPSADD,ADDBR
                                        MOV     LPSADD,CSR
                                        MOV     LPSADD,CSB
                                        INC     ADCS1
                                        ADD     #2,ADDBR
                                        ADD     #4,CSR
                                        ADD     #6,CSB
                                        MOV     LPSVCT,ADINT
                                        MOV     LPSVCT,ADINT1
                                        ADD     #2,ADINT1

                                ;NOW TEST IF A SWITCH REGISTE EXISTS

                                        MOV     #1$,@#4         ;LOAD BUS TRAP
                                        TST     @SWR            ;TEST IF REAL SWITCH REGISTER
                                        BR      CPUTYP          ;BR IF NO TRAP
```

```
                                        1$:     CMP     (SP)+,(SP)+     ;POP STACK
                                                MOV     #170,$WR        ;CHANGE POINTER TO LOC. 170
                                                MOV     #172,DISPLA     ;       AND 172
                                        ;DETERMINE CPU TYPE
                                        ;       R0=3 IF 11/45
                                        ;       R0=2 IF 11/40
                                        ;       R0=1 IF 11/20
                                        ;       R0=0 IF 11/05
                                                PIRQ=177772
                                                PS=177776
                                        CPUTYP: MOV     #RTI,@#6
                                                MOV     #6,@#4
                                                MOV     #3,R0           ;LOAD R0
                                                SEC                     ;SET C BIT
                                                TST     @#PIRQ          ;TEST PIRQ
                                                SBC     R0
                                                SEC
                                                TSTB    @#PS+1
                                                SBC     R0
                                                CLR     @#177700
                                                CLR     @#6
                                                ASL     R0
                                                MOV     CPDLAY(R0),CPTIME       ;GET TIME DELAY
                                                MOV     R0,PDPDLY       ;LOAD CPU DELAY
                                                BNE     1$
                                                INC     PDPDLY          ;UPDATE IT
                                        ;TRAP CATCHER
                                        1$:     MOV     #242,R2                 ;LOAD R2
                                                MOV     #240,R1                 ;LOAD R1
                                        5$:     MOV     R2,(R1)+                ;LOAD .+2
                                                CLR     (R1)+                   ;LOAD HALT
                                                MOV     R1,R2                   ;LOAD R2
                                                TST     (R2)+                   ;BUMP R2
                                                CMP     R2,#1002                ;TEST FOR LAST
                                                BNE     5$                      ;BR UNTIL DONE
                                                TST     @#42
                                                BEQ     .+6
                                                JMP     INIT2
                                                MOV     #100,@TKS       ;ENABLE TTY INTERUPTS
                                                CLR     PRINT1
                                                PRINT                   ;CALL MESSAGE PRINTER VIA 'EMT'
                                                TITLE                   ;TYPE PROGRAM HEADER.
                                        INITA:  PRINT
                                                MES4                    ;PRINT THE TEST CALL LETTERS.
                                                BR      INIT2           ;GO AND AWAIT COMMAND.
                                        ;MONITOR SUBROUTINE. ENTER VIA '↑C' OR A RESTART AT LOCATION '200'.
                                        MONITR: JSR     PC,$RESET               ;INITIALIZE ON ENTRY
                                                MOV     STACK,SP        ;RESET STACK POINTER
                                                MOV     #340,@PSW
                                                MOV     #100,@TKS       ;ENABLE TTY INTERRUPTS
                                                PRINT                   ;CALL MESSAGE PRINTER
                                                CNTRLC                  ;TYPE '↑C'
                                        INIT2:  MOV     #PWRFAL,@#24    ;SET UP POWER FAIL
                                                MOV     #6,@#4          ;SET UP BUSS ERROR
```

```
300  001612  005037  000006                CLR     @#6
301  001616  013777  001146  177320        MOV     ADINT1,@ADINT
302  001624  005077  177316                CLR     @ADINT1
303  001630  012737  001540  014776        MOV     #INITA,AVECTR   ;SET UP '↑A' VECTOR ADDRESS.
304  001636  004537  013244                JSR     R5,LEDS
305  001642  000006                         6
306  001644  005737  000042                TST     @#42
307  001650  001402                        BEQ     .+6
308  001652  000137  002002                JMP     WHAT
309  001656  012777  000100  177134        MOV     #100,@TKS       ;ENABLE KEYBOARD INTERRUPT
310  001664  104000                        PRINT
311  001666  014022                        DOT                     ;PRINT '.' TO INDICATE MONITOR READY
312  001670  104011                        TTYIN                   ;WAIT FOR TTY ENTRY
313  001672  042737  000040  010546        BIC     #BIT5,INBUF     ;CLEAR LOWER CASE BIT
314  001700  122737  000101  010546        CMPB    #'A,INBUF       ;TEST FOR 'A'
315  001706  001002                        BNE     .+6             ;NOT 'A
316  001710  000137  002040                JMP     ADTEST          ;YES, RUN 'A TO D LOGIC' TEST
317  001714  122737  000103  010546        CMPB    #'C,INBUF       ;TEST FOR 'C'
318  001722  001002                        BNE     .+6             ;NOT 'C'
319  001724  000137  005204                JMP     CALBRT          ;YES, RUN 'A TO D CALIBRATION' TEST
320  001730  122737  000104  010546        CMPB    #'D,INBUF       ;TEST FOR 'D'
321  001736  001002                        BNE     .+6             ;NOT 'D'
322  001740  000137  005540                JMP     REPTST          ;YES, RUN 'A TO D REPEATABILITY' TEST
323  001744  122737  000105  010546        CMPB    #'E,INBUF       ;TEST FOR 'E'
324  001752  001002                        BNE     .+6             ;NOT 'E'
325  001754  000137  006356                JMP     RECVRY          ;YES RUN 'A TO D RECOVERY TEST
326  001760  122737  000106  010546        CMPB    #'F,INBUF       ;TEST FOR 'F'
327  001766  001002                        BNE     .+6             ;NOT 'F'
328  001770  000137  006772                JMP     DSHTST          ;YES, RUN 'DUAL MODE' TEST
329  001774  104000          INIT3:        PRINT                   ;ILLEGAL ENTRY
330  001776  014025                        QMARK                   ;TYPE '?'
331  002000  000676                        BR      INIT2           ;WAIT AGAIN
332  002002  005037  002036  WHAT:         CLR     NODMA           ;CLEAR DMA POINTER
333  002006  000137  002044                JMP     ADTST1          ;RUN A TO D TEST
334  002012                  WHATA:
335  002012  004737  021526  WHATB:        JSR     PC,$RESET
336  002016  013700  000042                MOV     @#42,R0
337  002022  004710          LOGICAL:      JSR     PC,(0)
338  002024  000240                        NOP
339  002026  000240                        NOP
340  002030  000240                        NOP
341  002032  000137  002002                JMP     WHAT
342  002036  000000          NODMA:        0
```

```
                                ;*************************************************************************
                                ;       A TO D LOGIC TEST
                                ;*************************************************************************

002040  104000                  ADTEST: PRINT
002042  014030                          MESS
002044  004737  021526          ADTST1: JSR     PC,$RESET
002050  005037  001034                  CLR     PASSCT
002054  012737  002162  013160  ADTST0: MOV     #ADT1+2,RETURN
002062  013777  001034  176742          MOV     PASSCT,@DISPLA
002070  013706  014774                  MOV     STACK,SP
002074  005077  176716                  CLR     @PSW
002100  052777  000100  176712          BIS     #BIT6,@TKS
002106  005037  001066                  CLR     .DVLS

                                ;TEST FOR NO BUSS ERRORS

                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS

                                ;*      MOV     $ZERO,@ADDBR    ;/ PUT DATA FROM $ZERO TO DEVICE REG ADDBR
002132  012737  000002  015072          MOV     #BIT1,$TMDAT

                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
002150  005737  020554                  TST     $AERR
002154  001401                          BEQ     .+4
002156  104400                          ERROR

                                ;DOES (BIT 1) SET AND CLEAR

002160  104003                  ADT1:   SCOPE1
002162  012737  000002  015072          MOV     #BIT1,$TMDAT            ;SET BIT 1

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

                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002210  022737  000002  015074          CMP     #BIT1,$BDDAT
002216  001401                          BEQ     .+4
002220  104400                          ERROR                           ;ERROR, ADCS NOT = 2

                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS

                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEViCE REG ADCS,PUT DATA IN $BDDAT.
002242  005737  015074                  TST     $BDDAT
002246  001401                          BEQ     .+4
002250  104400                          ERROR                           ;ERROR, ADCS NOT = 0

                                ;DOES (BIT 2) SET AND CLEAR

002252  104002                  ADT2:   SCOPE
002254  012737  000004  015072          MOV     #BIT2,$TMDAT            ;SET BIT 2

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

```
                                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002302  022737  000004  015074                  CMP     #BIT2,$BDDAT
002310  001401                                  BEQ     .+4
002312  104400                                  ERROR                           ;ERROR, ADCS NOT = 4

                                        ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS

                                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002334  005737  015074                  TST     $BDDAT
002340  001401                                  BEQ     .+4
002342  104400                                  ERROR                           ;ERROR, ADCS NOT = 0

                                        ;DOES BURST (BIT 3) SET AND CLEAR

002344  104002                          ADT3:   SCOPE
002346  012737  000010  015072                  MOV     #BIT3,$TMDAT            ;SET BIT 3

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

                                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002374  022737  000010  015074                  CMP     #BIT3,$BDDAT
002402  001401                                  BEQ     .+4
002404  104400                                  ERROR                           ;ERROR, ADCS NOT = 10

                                        ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS

                                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002426  005737  015074                  TST     $BDDAT
002432  001401                                  BEQ     .+4
002434  104400                                  ERROR                           ;ERROR, ADCS NOT = 0
```

```
                                        ;DOES SCHMITT TRIGGER (BIT 4) SET AND CLEAR
002436  104003                  ADT4:   SCOPE1
002440  012737  000020  015072          MOV     #BIT4,$TMDAT                    ;SET BIT 4
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002466  022737  000020  015072          CMP     #BIT4,$TMDAT
002474  001401                          BEQ     .+4
002476  104400                          ERROR                                   ;ERROR ADCS NOT = 20
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002520  005737  015074                  TST     $BDDAT
002524  001401                          BEQ     .+4
002526  104400                          ERROR                                   ;ERROR ADCS NOT = 0
                                        ;DOES CLOCK OVERFLOW (BIT 5) SET AND CLEAR
002530  104002                  ADT5:   SCOPE
002532  012737  000040  015072          MOV     #BIT5,$TMDAT                    ;SET BIT 5
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002560  022737  000040  015074          CMP     #BIT5,$BDDAT
002566  001401                          BEQ     .+4
002570  104400                          ERROR                                   ;ERROR ADCS NOT = 40
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002612  005737  015074                  TST     $BDDAT
002616  105037  015072                  CLRB    $TMDAT
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002642  005737  015074                  TST     $BDDAT
002646  001401                          BEQ     .+4
002650  104400                          ERROR                                   ;ERROR ADCS NOT = 0
                                        ;DOES INTERRUPT ENABLE (BIT 6) SET AND CLEAR
002652  104002                  ADT6:   SCOPE
002654  012737  000100  015072          MOV     #BIT6,$TMDAT                    ;SET BIT 6
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
```

```
                                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002702  022737  000100  015074                          CMP     #BIT6,$BDDAT
002710  001401                                          BEQ     .+4
002712  104400                                          ERROR                           ;ERROR ADCS NOT = 100

                                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS

                                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002734  005737  015074                                  TST     $BDDAT
002740  001401                                          BEQ     .+4
002742  104400                                          ERROR                           ;ERROR ADCS NOT = 100

                                                ;DOES MUX CHANNEL (BIT 8) SET AND CLEAR

002744  104002                                  ADT7:   SCOPE
002746  012737  000400  015072                          MOV     #BIT8,$TMDAT            ;SET BIT 8

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

                                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
002774  022737  000400  015074                          CMP     #BIT8,$BDDAT
003002  001401                                          BEQ     .+4
003004  104400                                          ERROR                           ;ERROR ADCS NOT = 400

                                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS

                                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003026  005737  015074                                  TST     $BDDAT
003032  001401                                          BEQ     .+4
003034  104400                                          ERROR                           ;ERROR ADCS NOT = 0
```

```
                                        ;DOES MUX CHANNEL (BIT 9) SET AND CLEAR
003036  104002                  ADT10:  SCOPE
003040  012737  001000  015072          MOV     #BIT9,$TMDAT            ;SET BIT 9
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003066  022737  001000  015074          CMP     #BIT9,$BDDAT
003074  001401                        BEQ     .+4
003076  104400                        ERROR                           ;ERROR ADCS NOT = 10000
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003120  005737  015074                  TST     $BDDAT
003124  001401                        BEQ     .+4
003126  104400                        ERROR                           ;ERROR ADCS NOT = 0
                                        ;DOES MUX CHANNEL (BIT 10) SET AND CLEAR
003130  104002                  ADT11:  SCOPE
003132  012737  002000  015072          MOV     #BIT10,$TMDAT           ;SET BIT 10
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003160  022737  002000  015074          CMP     #BIT10,$BDDAT
003166  001401                        BEQ     .+4
003170  104400                        ERROR                           ;ERROR ADCS NOT 2000
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003212  005737  015074                  TST     $BDDAT
003216  001401                        BEQ     .+4
003220  104400                        ERROR                           ;ERROR ADCS NOT = 0
                                        ;DOES MUX CHANNEL (BIT 11) SET AND CLEAR
003222  104002                  ADT12:  SCOPE
003224  012737  004000  015072          MOV     #BIT11,$TMDAT           ;SET BIT 11
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003252  022737  004000  015074          CMP     #BIT11,$BDDAT
003260  001401                        BEQ     .+4
003262  104400                        ERROR                           ;ERROR ADCST NOT = 4000
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
```

```
                                                TST     $BDDAT
                                                BEQ     .+4
                                                ERROR                           ;ERROR ADCST NOT = 0

                                        ;DOES MUX CHANNEL (BIT 12) SET AND CLEAR

                                        ADT13:  SCOPE
                                                MOV     #BIT12,$TMDAT

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

                                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
                                                CMP     #BIT12,$BDDAT
                                                BEQ     .+4
                                                ERROR                           ;ERROR ADCS NOT = 10000

                                        ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS

                                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
                                                TST     $BDDAT
                                                BEQ     .+4
                                                ERROR                           ;ERROR ADCS NOT = 0
```

```
                                        ;DOES MUX CHANNEL (BIT 13) SET AND CLEAR
                                        
003406  104002                  ADT14:  SCOPE
003410  012737  020000  015072          MOV     #BIT13,$TMDAT           ;SET BIT 13
                                        
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                        
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003436  022737  020000  015074          CMP     #BIT13,$BDDAT
003444  001401                          BEQ     .+4
003446  104400                          ERROR                           ;ERROR ADCS NOT = 20000
                                        
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                        
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003470  005737  015074                  TST     $BDDAT
003474  001401                          BEQ     .+4
003476  104400                          ERROR                           ;ERROR ADCS NOT = 0
                                        
                                        ;DOES DUAL MODE (BIT14) SET AND CLEAR
                                        
003500  104002                  ADT15:  SCOPE
003502  012737  040000  015072          MOV     #BIT14,$TMDAT           ;SET BIT 14
                                        
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                        
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003530  022737  040000  015074          CMP     #BIT14,$BDDAT
003536  001401                          BEQ     .+4
003540  104400                          ERROR                           ;ERROR, ADCS NOT 40000
                                        
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                        
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003562  005737  015074                  TST     $BDDAT
003566  001401                          BEQ     .+4
003570  104400                          ERROR                           ;ERROR, ADCS NOT = 0
                                        
                                        ;DOES ERROR (BIT 15) SET AND CLEAR
                                        
003572  104002                  ADT16:  SCOPE
003574  012737  100000  015072          MOV     #BIT15,$TMDAT
                                        
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                        
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003622  022737  100000  015074          CMP     #BIT15,$BDDAT
003630  001401                          BEQ     .+4
003632  104400                          ERROR                           ;ERROR , ERROR BIT FAILED TO SET
                                        
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                        
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
003654  005737  015074                  TST     $BDDAT
```

```
                                        BEQ     .+4
003660  001401
                                        ERROR                   ;ERROR, ERROR BIT FAILED TO CLEAR
003662  104400
                                ;DOES DONE (BIT 7) SET AND CLEAR
                        ADT20:  SCOPE1
003664  104003
                        ;*      MOV     @ADDBR,$BDDAT   ;/READ DEVICE REG ADDBR,PUT DATA IN $BDDAT.
                                MOV     $BDDAT,R0
003676  013700  015074
                                CLR     KSTOR4
003702  005037  015014
                                MOV     #BIT0,$TMDAT
003706  012737  000001  015072
                        ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
                        ADT20A: TSTB    $BDDAT
003734  105737  015074
                                BMI     ADT20B
003740  100404
                                INC     KSTOR4
003742  005237  015014
                                BNE     ADT20A
003746  001372
                                ERROR                           ;ERROR, ADCS NOT = 200
003750  104400
                        ADT20B:
003752
                        ;*      MOV     @ADDBR,$BDDAT   ;/READ DEVICE REG ADDBR,PUT DATA IN $BDDAT.
                                MOV     $BDDAT,R0
003762  013700  015074
                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
                                TSTB    $BDDAT
003776  105737  015074
                                BPL     .+4
004002  100001
                                ERROR                           ;ERROR, DONE FAILED TO CLEAR
004004  104400
                        ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
                                BIT     #BIT0,$BDDAT
004016  032737  000001  015074
                                BEQ     .+4
004024  001401
                                ERROR                           ;ERROR, AD TO FAILED TO cLEAR
004026  104400
```

```
                                        ;DOES ERROR (BIT 15) SET
                                        
004030  104003                  ADT21:  SCOPE1
                                        
                                ;*      MOV     @ADDBR,$BDDAT   ;/READ DEVICE REG ADDBR,PUT DATA IN $BDDAT.
004042  013700  015074                  MOV     $BDDAT,R0
004046  012737  000001  015072          MOV     #BIT0,$TMDAT
                                        
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
004064  105737  015074          1$:     TSTB    $BDDAT          ;WAIT FOR FLAG
004070  100375                          BPL     1$
004072  013737  011436  015032          MOV     CPTIME,DELAY1
004100  006237  015032                  ASR     DELAY1
004104  006237  015032                  ASR     DELAY1
004110  006237  015032                  ASR     DELAY1
004114  012737  000001  015072          MOV     #BIT0,$TMDAT
                                        
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
004132  005337  015032          2$:     DEC     DELAY1
004136  100375                          BPL     2$
                                        
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
004150  005737  015074                  TST     $BDDAT
004154  100401                          BMI     .+4
004156  104400                          ERROR                           ;ERROR ADCS NOT = 100000
                                        
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
004170  013737  011436  015032          MOV     CPTIME,DELAY1
004176  005337  015032          3$:     DEC     DELAY1
004202  100375                          BPL     3$
004204  000240                          NOP
004206  000240                          NOP
004210  000240                          NOP
                                        
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
004222  005737  015074                  TST     $BDDAT
004226  100001                          BPL     .+4
004230  104400                          ERROR                           ;ERROR ADCS NOT = 0
                                        
                                        ;TEST THAT NO EXTERNAL CONVERSIONS INPUT
                                        
004232  104003                  ADT22:  SCOPE1
                                        
                                ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
                                        
                                ;*      MOV     @ADDBR,$BDDAT   ;/READ DEVICE REG ADDBR,PUT DATA IN $BDDAT.
004254  013700  015074                  MOV     $BDDAT,R0
                                        
                                ;*      MOV     @ADCS,$TMDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $TMDAT.
004270  052737  000020  015072          BIS     #BIT4,$TMDAT
                                        
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
004306  005000                          CLR     R0
```

```
   730  004310  005200                          INC     R0
   731  004312  001376                          BNE     .-2
   732
   733                                  ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
   734  004324  105737  015074                   TSTB    $BDDAT
   735  004330  100001                          BPL     .+4
   736  004332  104400                          ERROR                           ;ERROR EXTERNAL CONVERSION
   737
   738                                  ;TEST THAT NO CLOCK OVERFLOWS START CONVERSIONS
   739
   740  004334  104003                 ADT23:   SCOPE1
   741
   742                                  ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
   743
   744                                  ;*      MOV     @ADDBR,$BDDAT   ;/READ DEVICE REG ADDBR,PUT DATA IN $BDDAT.
   745  004356  013700  015074                   MOV     $BDDAT,R0
   746
   747                                  ;*      MOV     @ADCS,$TMDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $TMDAT.
   748  004372  052737  000040  015072          BIS     #BIT5,$TMDAT
   749
   750                                  ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
   751  004410  005037  015014                   CLR     KSTOR4
   752  004414  005237  015014          1$:     INC     KSTOR4
   753  004420  001375                          BNE     1$
   754
   755                                  ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
   756  004432  105737  015074                   TSTB    $BDDAT
   757  004436  100001                          BPL     .+4
   758  004440  104400                          ERROR                           ;ERROR OVERFLOW STARTED A CONVERSION
```

```
                                        ;PRE INTERRUPT SETUP
                                
004442  042737  177437  001004        BIC     #177437,ADBRL   ;MASK TO BITS
004450  001001                        BNE     .+4
004452  000000                        HALT                    ;BR LEVEL INDICATED IS 0
004454  022737  000340  001004        CMP     #340,ADBRL      ;IS IT BR LEVEL 7
004462  001001                        BNE     .+4
004464  000000                        HALT                    ;BR LEVEL INDICATED IS 7

004466  013737  001004  015054        MOV     ADBRL,BRLEV1
004474  162737  000040  015054        SUB     #40,BRLEV1
004502  013737  001004  015056        MOV     ADBRL,BRLEV2
004510  013737  001004  015060        MOV     ADBRL,BRLEV3
004516  062737  000040  015060        ADD     #40,BRLEV3
```

```
                                        ;TEST THAT RESET CLEARS MUX BITS
                                        ;
004524  104003                  ADT27:  SCOPE1
004526  012737  037400  015072          MOV     #BIT13!BIT12!BIT11!BIT10!BIT9!BIT8,$TMDAT
                                        ;
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
004544  004737  021526                  JSR     PC,$RESET
                                        ;
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
004560  005737  015074                  TST     $BDDAT
004564  001401                          BEQ     .+4
004566  104400                          ERROR                   ;ERROR, RESET FAILED TO CLEAR MUX BITS
                                        ;
                                        ;TEST THAT RESET CLEARS ST, CLOCK OVERFLOW AND INTERRUPT ENABLE BITS
                                        ;
004570  104003                  ADT30:  SCOPE1
004572  012737  000160  015072          MOV     #BIT6!BIT5!BIT4,$TMDAT
                                        ;
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
004610  004737  021526                  JSR     PC,$RESET
                                        ;
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
004624  005737  015074                  TST     $BDDAT
004630  001401                          BEQ     .+4
004632  104400                          ERROR                   ;ERROR, RESET FAILED TO CLEAR ENABLES
                                        ;
                                        ;TEST THAT RESET CLEARS DONe AND ERROR BITS
                                        ;
004634  104003                  ADT31:  SCOPE1
004636  012737  000001  015072          MOV     #BIT0,$TMDAT
                                        ;
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
                                        ;
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
004664  105737  015074                  TSTB    $BDDAT
004670  100375                          BPL     .-4
004672  012737  100000  015072          MOV     #BIT15,$TMDAT   ;SET ERROR BIT
                                        ;
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
004710  004737  021526                  JSR     PC,$RESET
                                        ;
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
004724  005737  015074                  TST     $BDDAT
004730  001401                          BEQ     .+4
004732  104400                          ERROR                   ;ERROR, RESET FAILED TO CLEAR ERROR OR DONe FLAG
```

```
                                        ;TEST THAT RESET CLEARS BURST AND MODE BITS
                                        ;
004734  104003                  ADT32:  SCOPE1
004736  012737  000016  015072          MOV     #BIT3!BIT2!BIT1,$TMDAT  ;SET BITS 1-3
                                        ;
                                ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
004754  004737  021526                  JSR     PC,$RESET
                                        ;
                                ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
004770  005737  015074                  TST     $BDDAT
004774  001401                          BEQ     .+4
004776  104400                          ERROR                           ;ERROR, ADCS NOT = 0
                                        ;
                                        ;LED TEST
                                        ;
005000  104003                  TEST15: SCOPE1
005002  005737  000042                  TST     @#42
005006  001060                          BNE     TEST16
005010  005737  001034                  TST     PASSCT          ;TEST IF FIRST PASS
005014  001455                          BEQ     TEST16          ;BR IF YES
005016  005077  173774                  CLR     @PSW
005022  052777  000100  173770          BIS     #BIT6,@TKS
005030  005037  015046                  CLR     TEMP1
005034  013737  015046  015050  TST15A: MOV     TEMP1,TEMP2
005042  012737  000006  015052          MOV     #6,TEMP3
005050  013737  015050  015072  TST15B: MOV     TEMp2,$TMDAT
                                        ;
                                ;*      MOV     $TMDAT,@ADDBR   ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADDBR
005066  013737  001006  015012          MOV     PDPDLY,KSTOR3
005074  012700  000000                  MOV     #0,R0
005100  032777  004000  173722          BIT     #BIT11,@SWR     ;TEST IF NO INTERATIONS
005106  001020                          BNE     TEST16          ;BR IF NO INTERATIONS
005110  005300                  TST15C: DEC     R0
005112  001376                          BNE     TST15C
005114  005337  015012                  DEC     KSTOR3
005120  001373                          BNE     TST15C
005122  105237  015051                  INCB    TEMP2+1
005126  005337  015052                  DEC     TEMP3
005132  001346                          BNE     TST15B
005134  005237  015046                  INC     TEMP1
005140  022737  000040  015046          CMP     #40,TEMp1
005146  001332                          BNE     TST15A
                                        ;
005150  104014                  TEST16: SCOPE0                          ;LOGICAL END OF A TO d LOGIC TEST
005152  004737  021526                  JSR     PC,$RESET
005156  005737  000042                  TST     @#42
005162  001402                          BEQ     .+6
005164  000137  002012                  JMP     WHATA
005170  004737  013236                  JSR     PC,BELL         ;REPORT END OF PASS
005174  005237  001034                  INC     PASSCT
005200  000137  002054                  JMP     ADTST0
```

```
                                ;*****************************************************************************
                                ;CALIBRATION ROUTINE
                                ;*****************************************************************************

                                ;ROUTINE REQUESTS THE TYPE OF 'SYNC' TO BE USED ('I' INTERNAL OR 'E' EXTERNAL).
                                ;OR 'C' LPS CLOCK)
                                ;THE PROGRAM THEN TAKES CONTINUOUS CONVERSIONS USING DATA SW'S 5-0
                                ;TO SELECT THE CH.,
                                ;OR SW '10' TO PRINT THE CONVERSION VALUE.

005204  012737  005216  014776  CALBRT: MOV     #CALBT1,AVECTR  ;SET UP '↑A' RESTART ADDRESS
005212  104000                         PRINT
005214  014057                         MES7                    ;TYPE TEST HEADER
005216  104000                 CALBT1: PRINT
005220  014136                         MES10                   ;TEXT 'SYNC I OR E OR C'
005222  104011                 CALB1A: TTYIN                   ;WAIT FOR INPUT.
005224  042737  000040  010546         BIC     #BIT5,INBUF
005232  013737  010546  015000         MOV     INBUF,PROC      ;SAVE IT IN TEMP STORAGE
005240  104000                         PRINT
005242  014020                         CRLF
005244  000402                         BR      .+6
005246  005037  015000         MANCAL: CLR     PROC
005252  005037  015036                 CLR     USEDMA          ;CLEAR DMA FLAG
005256  005037  015040                 CLR     USECLK          ;CLEAR CLOCK FLAG
005262  005037  015034                 CLR     USEDSH          ;CLEAR DUAL MODE
005266  012737  000001  015004         MOV     #1,COUNT        ;SET UP FOR '1' CONVERSION
005274  117737  173530  014771 CALBT2: MOVB    @SWR,ADWRD2+1   ;GET CH. FROM THE SW REG.
005302  042737  140377  014770         BIC     #140377,ADWRD2  ;CLR UNWANTED BITS, A/D WORD COMPLETE
005310  017737  173514  015010         MOV     @SWR,KSTOR2     ;SAVE ORIGINAL SWITCH SETTING.
005316  022737  000105  015000         CMP     #105,PROC       ;TEST SYNC SELECT
005324  001004                         BNE     CALB2           ;BRANCH IF NOT 'E',
005326  052737  000020  014770         BIS     #20,ADWRD2      ;OTHERWISE ADD 'EXT' SYNC BIT TO A/D.
005334  000416                         BR      CALB2A
005336  122737  000103  015000 CALB2:  CMPB    #103,PROC
005344  001007                         BNE     CALB2C          ;BRANCH IF NOT 'C'
005346  052737  000040  014770         BIS     #40,ADWRD2      ;OTHERWISE ADD 'CLOCK' SYNC TO A/D
005354  012737  177777  015040         MOV     #-1,USECLK
005362  000403                         BR      CALB2A
005364  052737  000001  014770 CALB2C: BIS     #1,ADWRD2
005372  005737  015040         CALB2A: TST     USECLK          ;CHECK CLOCK FLAG
005376  001422                         BEQ     CALB2D          ;NOT LPS CLOCK

                                ;*      MOV     $ZERO,@CSR      ;/ PUT DATA FROM $ZERO TO DEVICE REG CSR
005410  012737  176000  015072 ;*      MOV     #-2000,$TMDAT           ;SET UP PRESET BUFFER

                                ;*      MOV     $TMDAT,@CSB     ;/ PUT DATA FROM $TMDAT TO DEVICE REG CSB
005426  012737  000403  015072 ;*      MOV     #403,$TMDAT     ;SET UP RATE, START CLOCK

                                ;*      MOV     $TMDAT,@CSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG CSR
005444  013706  014774 CALB2D: MOV     STACK,SP
005450  004737  010726         JSR     PC,ADCNVT       ;TAKE AND STORE THE CONVERSIONS
005454  012737  000001  015052         MOV     #1,TEMP3        ;SETUP TO PRINT '1' VALUE
005462  004537  013244                 JSR     R5,LEDS
005466  015214                         ADBUFF
```

```
                                BIT     #SW10,@SWR
                                BEQ     CALB2B
                                PRTAVG                  ;PRINT IT
                                PRINT
                                CRLF
                                BR      CALBT4          ;TEST FOR LOOP
                        CALB2B: MOV     ADBUFF,R0       ;SET UP A/D BUFFER.
                                MOV     R0,@DISPLA      ;LOAD DISPLAY REG. (11/45)
                                TSTTKS                  ;TEST FOR TTY FLAG
                        ARESET: NOP
                        CALBT4: TSTTKS                  ;TEST FOR KEYBOARD INTERRUPT
                                CMP     KSTOR2,@SWR     ;TEST IF SWITCH REGISTER HAS CHANGED
                                BEQ     CALB2A          ;BRANCH AND TAKE NEXT CONVERSION
                                BR      CALBT2          ;YES, COMPUTE NEW INPUT
                        ;*******************************************************************************
                        ;REPEATABILITY TEST
                        ;*******************************************************************************

                        ;THIS ROUTINE TO DESIGNED TO SHOW REPEATIBILITY BY TAKING A SERIES OF
                        ;'512' CONVERSIONS,  AVERAGING THEM AND THEN CATORIZING
                        ;THEM IN BINS FROM THE AVERAGE PLUS & MINUS 6 COUNTS.  THE ROUTINE
                        ;REQUESTS FOR A CHANNEL OR CHANNELS, A COUNT SPREAD AND A A/D MODE TO BE TYPED
                        ;IN VIA THE OPERATOR.  A CONTINUOUS SERIES OF CONVERSIONS ARE THEN TAKEN
                        ;AND COMPARED AGAINST THE INPUT COUNT SPREAD.  IF ALL '512' CONVERSIONS
                        ;ARE FOUND TO BE WITHIN THE SPREAD THE NEXT CH. IS EXERCISED OTHERWISE
                        ;THE COUNTS ARE TYPED OUT.  SETTING SWITCH '10' TO A '1' WILL FORCE A PRINTOUT
                        ;OF THE CH (S).

005540  012737  005552  014776  REPTST: MOV     #REPT1,AVECTR   ;SET UP CNTR 'A' VECTOR ADDRESS
005546  104000                          PRINT
005550  014256                          MES13                   ;TEXT 'REPEATIBILITY TEST'
005552  005037  015062          REPT1:  CLR     MESPRT
005556  005037  015036                  CLR     USEDMA
005562  005037  015034                  CLR     USEDSH
005566  005037  015042                  CLR     OPS1
005572  104000                          PRINT
005574  014306                          MES14                   ;REQUEST CHANNEL (S)
005576  104011                          TTYIN                   ;WAIT FOR INPUT
005600  104001                          DECOCT                  ;CONVERT TO OCTAL
005602  013737  010716  015006          MOV     BCDTAB,KSTOR1   ;SAVE AS INTIAL CH.
005610  013737  015006  015010          MOV     KSTOR1,KSTOR2   ;ALSO SAVE AS 2ND CH. ENTRY
005616  005737  010720                  TST     BCDTAB+2        ;TEST FOR SECOND ENTRY
005622  001407                          BEQ     REPT2           ;BRANCH IF NO SECOND ENTRY
005624  023737  010720  015006          CMP     BCDTAB+2,KSTOR1 ;COMPARE ENTRY 1 TO ENTRY 2
005632  100747                          BMI     REPT1           ;BRANCH AND RESTART IF ILLEGAL
005634  013737  010720  015010          MOV     BCDTAB+2,KSTOR2 ;OTHERWISE SAVE AS SECOND CH.
005642  104000                  REPT2:  PRINT
005644  014346                          MES16                   ;TEXT 'COUNT SPREAD ?'
005646  104011                          TTYIN                   ;WAIT FOR ENTRY
005650  104001                          DECOCT                  ;DECODE TO OCTAL
005652  013737  010716  015012          MOV     BCDTAB,KSTOR3   ;SAVE IT
005660  104000                          PRINT
005662  014365                          MES18
005664  104011                          TTYIN                   ;REQUEST MODE
```

```
  980  005666  122737  000102  010546          CMPB    #'B,INBUF       ;TEST FOR B
  981  005674  001003                          BNE     REPT2B          ;NOT
  982  005676  005137  015036                  COM     USEDMA
  983  005702  000411                          BR      REPT2A
  984  005704  122737  000104  010546  REPT2B: CMPB    #'D,INBUF       ;TEST FOR DUAL MODE
  985  005712  001005                          BNE     REPT2A          ;BRANCH IF NOT
  986  005714  005137  015034                  COM     USEDSH          ;SET DUAL MODE
  987  005720  052737  000100  015006          BIS     #BIT6,KSTOR1    ; BIT
  988  005726  013706  014774          REPT2A: MOV     STACK,SP
  989  005732  013737  015006  015014          MOV     KSTOR1,KSTOR4   ;SAVE STARTING CH.
  990  005740  104013          REPT3:  TSTTKS          ;TEST FOR KEYBOARD FLAG
  991  005742  012737  001000  015004          MOV     #1000,COUNT     ;SET FOR '512' CONVERSIONS
  992  005750  113737  015014  014771          MOVB    KSTOR4,ADWRD2+1 ;MOV SELECTED CH. TO HIGH BYTE OF ADWORD
  993  005756  042737  100377  014770          BIC     #100377,ADWRD2  ;MASK
  994  005764  052737  000001  014770          BIS     #1,ADWRD2
  995  005772  004737  010726                  JSR     PC,ADCNVT       ;TAKE THE CONVERSIONS
  996  005776  104004                          CMPUTE                  ;AVERAGE & COMPUTE DISTRIBUTION
  997  006000  104005                          CATORIZ
  998  006002  013777  015114  173022          MOV     AVRAGE,@DISPLA  ;LOAD DISPLAY REG. (11/45)
  999  006010  032777  002000  173012          BIT     #SW10,@SWR      ;TEST DATA SW10
 1000  006016  001047                          BNE     REPT4           ;IF SET, FORCE TYPE OUT
 1001  006020  032777  020000  173002  TSTCT4: BIT     #SW13,@SWR      ;TEST FOR INHIBIT TYPEOUT
 1002  006026  001135                          BNE     REPT7           ;BRANCH IF SW SET
 1003  006030  022737  000004  015012          CMP     #4,KSTOR3       ;WAS 4 TYPED
 1004  006036  001005                          BNE     TSTCT3          ;NO, TEST FOR '3'
 1005  006040  022737  001000  015170          CMP     #1000,XSPRD4    ;TOTAL COUNTS WITHIN 4 COUNTS
 1006  006046  001033                          BNE     REPT4           ;BRANCH IF NO.
 1007  006050  000524                          BR      REPT7           ;YES, TEST NEXT CH.
 1008  006052  022737  000003  015012  TSTCT3: CMP     #3,KSTOR3       ;COUNT = TO 3
 1009  006060  001005                          BNE     TSTCT2          ;NO TEST COUNT 2
 1010  006062  022737  001000  015166          CMP     #1000,XSPRD3
 1011  006070  001022                          BNE     REPT4           ;BRANCH IF COUNT NOT WITHIN 3
 1012  006072  000513                          BR      REPT7           ;YES, TEST NEXT CH.
 1013
 1014  006074  022737  000002  015012  TSTCT2: CMP     #2,KSTOR3       ;COUNT =TO 2
 1015  006102  001005                          BNE     TSTCT1          ;NO, TEST COUNT 1
 1016  006104  022737  001000  015164          CMP     #1000,XSPRD2
 1017  006112  001011                          BNE     REPT4           ;BRANCH IF NOT WITHIN 2
 1018  006114  000502                          BR      REPT7           ;YES, TEST NEXT CH.
 1019  006116  022737  000001  015012  TSTCT1: CMP     #1,KSTOR3       ;COUNT = TO 1
 1020  006124  001004                          BNE     REPT4           ;NO, REPORT EVEN IF NOT '0'
 1021  006126  022737  001000  015162          CMP     #1000,XSPRD1
 1022  006134  001472                          BEQ     REPT7           ;BRANCH IF TOTAL WITHIN 1 COUNT
 1023  006136  104000          REPT4:  PRINT
 1024  006140  014020                          CRLF
 1025  006142  005737  015062                  TST     MESPRT          ;TEST IF HEADER HAS BEEN TYPED
 1026  006146  001002                          BNE     REPT5           ;BRANCH IF YES
 1027  006150  104000                          PRINT
 1028  006152  014444                          MES19                   ;TEXT 'CH. HIGH AVG. LOW'
 1029  006154  104013          REPT5:  TSTTKS          ;TEST FOR KEYBOARD INTERRUPT
 1030  006156  104000                          PRINT
 1031  006160  014020                          CRLF                    ;CARRIAGE RETURN, LINE FEED
 1032  006162  013737  015014  006354          MOV     KSTOR4,REPT8A   ;MOV. CH.
 1033  006170  042737  177700  006354          BIC     #177700,REPT8A
```

```
1034  006176  005737  015034            TST     USEDSH          ;TEST FOR DUAL MODE
1035  006202  001403                    BEQ     REPT8           ;BRANCH IF NOT
1036  006204  062737  000010  006354    ADD     #10,REPT8A
1037  006212  104010            REPT8:  PRTOCT
1038  006214  006354                    REPT8A
1039  006216  104007                    SPACE
1040  006220  104010                    PRTOCT                  ;PRINT LOW VALUE
1041  006222  015070                    LOW
1042  006224  104007                    SPACE
1043  006226  104010                    PRTOCT                  ;PRINT AVERAGE VALUE
1044  006230  015114                    AVRAGE
1045  006232  104007                    SPACE
1046  006234  104010                    PRTOCT                  ;PRINT HIGH VALUE
1047  006236  015066                    HIGH
1048  006240  005737  015062            TST     MESPRT
1049  006244  001002                    BNE     REPT6
1050  006246  104000                    PRINT
1051  006250  014477                    MES20                   ;PRINT 'COUNT SPREAD' HEADER
1052  006252  052737  000007  015062  REPT6:  BIS     #7,MESPRT       ;INHIBIT OTHER HEADERS
1053  006260  022737  001000  015144          CMP     #1000,AVGCNT    ;TEST IF ALL COUNTS WERE AT AVG.
1054  006266  000240                    NOP                     ;<BEQ REPT7> BRANCH TO NEXT CH. IF YES.
1055  006270  104000                    PRINT
1056  006272  014020                    CRLF
1057  006274  012704  015130            MOV     #ORLOW,R4
1058  006300  012402            REPT6A: MOV     (R4)+,R2
1059  006302  104006                    BINDEC                  ;TYPE OUT COUNT SPREAD
1060  006304  022704  015162            CMP     #XSPRD1,R4      ;TEST FOR DONE
1061  006310  001373                    BNE     REPT6A          ;BRANCH IF NO AND TYPE NEXT COUNT
1062  006312  005777  172512            TST     @SWR
1063  006316  100001                    BPL     REPT7
1064  006320  000000                    HALT                    ;REPEATIBILITY ERROR
1065  006322  013704  015014    REPT7:  MOV     KSTOR4,R4
1066  006326  042704  177700            BIC     #177700,R4
1067  006332  023704  015010            CMP     KSTOR2,R4       ;TESTED ALL CH.(S)?
1068  006336  001404                    BEQ     REPT7A
1069  006340  005237  015014            INC     KSTOR4          ;TEST NEXT CHANNEL
1070  006344  000137  005740            JMP     REPT3           ;
1071  006350  000137  005726    REPT7A: JMP     REPT2A
1072
1073  006354  000000    REPT8A: 0
```

```
1074                                    ;*************************************************************************
1075                                    ;RECOVERY TEST
1076                                    ;*************************************************************************
1077                                    ;THIS TEST IS DESIGNED TO TEST RECOVERY OF THE A/D CONVERTER VIA ACCEPT-
1078                                    ;ING TWO (2) CHANNEL FROM THE TELETYPE AND THEN TAKE A
1079                                    ;SERIES OF EIGHT CONVERSIONS ON EACH CHANNEL AND TYPING OUT THE CON-
1080                                    ;VERSION VALUES OF THE 2ND CHANNEL IN THE ORDER THEY WERE TAKEN.
1081  006356  012737  006400  014776  RECVRY: MOV     #RECVY1,AVECTR  ;SET UP THE '↑A' RETURN ADDRESS
1082  006364  005037  015036                 CLR     USEDMA
1083  006370  005037  015034                 CLR     USEDSH
1084  006374  104000                          PRINT
1085  006376  014106                          MES8                    ;TEXT 'RECOVERY TEST'
1086  006400  104000                  RECVY1: PRINT                   ;REQUEST CHANNELS
1087  006402  014306                          MES14
1088  006404  104011                          TTYIN                   ;WAIT FOR INPUT
1089  006406  104001                          DECOCT                  ;CONVERT TO OCTAL
1090  006410  013737  010716  015006          MOV     BCDTAB,KSTOR1   ;SAVE 1ST CH.
1091  006416  013737  010720  015010          MOV     BCDTAB+2,KSTOR2 ;SAVE 2ND CH.
1092  006424  013706  014774          RECVY2: MOV     STACK,SP
1093  006430  104013                          TSTTKS                  ;CHECK FOR KEYBOARD FLAG
1094  006432  012737  000020  006744          MOV     #16.,10$        ;LOAD COUNT
1095  006440  005037  006746                  CLR     11$             ;CLEAR POINTER
1096  006444  012737  000010  015052          MOV     #10,TEMP3       ;SET UP TO PRINT 10
1097  006452  012737  000010  015004          MOV     #10,COUNT       ;SETUP TO TAKE '8' CONVERSIONS ON 1ST CH.
1098  006460  113737  015006  014771  1$:     MOVB    KSTOR1,ADWRD2+1 ;LOAD 1ST CH.
1099  006466  042737  140377  014770          BIC     #140377,ADWRD2
1100  006474  052737  000001  014770          BIS     #1,ADWRD2
1101  006502  005237  015030                  INC     DELAY           ;SET FLAG
1102  006506  004737  010726                  JSR     PC,ADCNVT       ;TAKE THE CONVERSIONS
1103  006512  113737  015010  014771          MOVB    KSTOR2,ADWRD2+1 ;SET UP 2ND CH.
1104  006520  042737  140377  014770          BIC     #140377,ADWRD2
1105  006526  052737  000001  014770          BIS     #1,ADWRD2
1106  006534  005237  015030                  INC     DELAY           ;SET FLAG
1107  006540  004737  010726                  JSR     PC,ADCNVT       ;TAKE 2ND SERIES OF CONVERSIONS
```

```
1108   006544   013701   006746                MOV     11$,R1          ;LOAD A POINTER
1109   006550   012700   015214                MOV     #ADBUFF,R0      ;LOAD POINTER
1110   006554   012061   015256                MOV     (R0)+,ADTB0(R1)
1111   006560   012061   015320                MOV     (R0)+,ADTB1(R1)
1112   006564   012061   015362                MOV     (R0)+,ADTB2(R1)
1113   006570   012061   015424                MOV     (R0)+,ADTB3(R1)
1114   006574   012061   015466                MOV     (R0)+,ADTB4(R1)
1115   006600   012061   015530                MOV     (R0)+,ADTB5(R1)
1116   006604   012061   015572                MOV     (R0)+,ADTB6(R1)
1117   006610   012061   015634                MOV     (R0)+,ADTB7(R1)
1118   006614   062737   000002   006746       ADD     #2,11$          ;UPDATE POINTER
1119   006622   005337   006744                DEC     10$
1120   006626   001314                        BNE     1$
1121
1122   006630   012700   000010                MOV     #8.,R0          ;LOAD COUNT
1123   006634   012702   015214                MOV     #ADBUFF,R2      ;LOAD POINTER
1124   006640   012701   000000                MOV     #0,R1
1125   006644   012737   000017   015046  2$:  MOV     #15.,TEMP1      ;LOAD TEMP
1126   006652   012737   000004   012160       MOV     #4,CMPCNT       ;LOAD AVRG. COUNT
1127   006660   016104   006752                MOV     LADTB(R1),R4    ;GET POINTER
1128   006664   012746   000340                MOV     #340,-(SP)      ;PUSH
1129   006670   004737   012014                JSR     PC,CMPTEA       ;CONVERT AND AVERAGE
1130   006674   013722   015114                MOV     AVRAGE,(R2)+    ;SAVE AVERAGE
1131   006700   005721                        TST     (R1)+           ;UPDATE POINTER
1132   006702   005300                        DEC     R0
1133   006704   001357                        BNE     2$              ;BR IF NOT DONE
1134
1135   006706   032777   002000   172114       BIT     #SW10,@SWR      ;TES1 BIT 10
1136   006714   001243                        BNE     RECVY2
1137   006716   104000                        PRINT
1138   006720   014131                        MES9                    ;TEXT 'CH.'
1139   006722   104013                        TSTTKS                  ;CHECK AGAIN FOR KEYBOARD FLAG
1140   006724   013737   015010   006354       MOV     KSTOR2,REPT8A
1141   006732   104010                        PRTOCT
1142   006734   006354                        REPT8A
1143   006736   104007                        SPACE
1144   006740   104012                        PRTAVG                  ;PRINT VALUES OF 2ND CH.
1145   006742   000630                        BR RECVY2               ;DO IT AGAIN
1146   006744   000000              10$:      0
1147   006746   000000              11$:      0
1148   006750   000000              12$:      0
1149
1150   006752   015256              LADTB:    ADTB0
1151   006754   015320                        ADTB1
1152   006756   015362                        ADTB2
1153   006760   015424                        ADTB3
1154   006762   015466                        ADTB4
1155   006764   015530                        ADTB5
1156   006766   015572                        ADTB6
1157   006770   015634                        ADTB7
```

```
1158
1159                                    ;*****************************************************************
1160                                    ;       DUAL SAMPLE AND HOLD TEST       <CH 0-3> AND <CH 10-13> ONLY
1161                                    ;*****************************************************************
1162
1163  006772  012737  007000  014776  DSHTST: MOV     #DSHTO,AVECTR   ;SET UP '↑A' RETURN
1164  007000  104000                  DSHTO:  PRINT                   ;TYPE HEADER
1165  007002  014321                          MES15
1166  007004  005037  001034          DSHT1:  CLR     PASSCT          ;CLEAR PASS COUNT
1167  007010  005037  015024          DSHT2:  CLR     STCHAN          ;CLEAR STARTING CHANNEL
1168  007014  004537  013244          DSHT3:  JSR     R5,LEDS
1169  007020  015024                          STCHAN
1170  007022  000240                          NOP                     ;RESET'''
1171  007024  052777  000100  171766          BIS     #BIT6,@TKS      ;ENABLE INTERRUPT
1172  007032  013706  014774                  MOV     STACK,SP        ;LOAD THE STACK POINTER
1173  007036  013777  001034  171766          MOV     PASSCT,@DISPLA  ;LOAD PASS COUNT
1174  007044  113737  015024  015072          MOVB    STCHAN,$TMDAT   ;LOAD CHANNEL <0-3>
1175  007052  005077  171740                  CLR     @PSW
1176
1177                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1178  007066  052737  000001  015072          BIS     #1,$TMDAT
1179
1180                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1181  007104                          1$:
1182
1183                                    ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
1184  007114  105737  015074                  TSTB    $BDDAT
1185  007120  100371                          BPL     1$
1186
1187                                    ;*      MOV     @ADDBR,AVERM4   ;/READ DEVICE REG ADDBR,PUT DATA IN AVERM4.
1188
1189                                    ;*      MOV     @ADCS,$TMDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $TMDAT.
1190  007142  152737  000010  015073          BISB    #10,$TMDAT+1
1191
1192                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1193
1194                                    ;*      MOV     @ADCS,$TMDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $TMdAT.
1195  007170  052737  000001  015072          BIS     #1,$TMDAT
1196
1197                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1198  007206                          2$:
1199
1200                                    ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
1201  007216  105737  015074                  TSTB    $BDDAT
1202  007222  100371                          BPL     2$
1203
1204                                    ;*      MOV     @ADDBR,AVERM3   ;/READ DEVICE REG ADDBR,PUT DATA IN AVERM3.
1205
1206  007234  004537  007634                  JSR     R5,COMPR        ;COMPARE TWO NUMBERS
1207  007240  015104                          AVERM4
1208  007242  015106                          AVERM3
1209  007244  001014                          BNE     DSHT6           ;BRANCH IF NOT EQUAL <DIFFERENT INPUT>
1210  007246  032777  020000  171554          BIT     #BIT13,@SWR
1211  007254  001004                          BNE     DSHT4
```

```
1212  007256  104000                        PRINT
1213  007260  014020                        CRLF
1214  007262  104000                        PRINT
1215  007264  014605                        MES22                   ;PRINT MESSAGE
1216  007266  005777  171536      DSHT4:    TST     @SWR
1217  007272  100001                        BPL     DSHT6
1218  007274  000000                        HALT                    ;ERROR, CHANNEL PAIRS HAVE SAME INPUT VOLTAGE
1219                                                                ;THE FIRST CHANNEL NUMBER OF THE PAIR IS IN THE LEDS
1220
```

```
1221
1222                                    ;NOW TEST THAT DUAL SAMPLE REALLY WORKS
1223                                    ;SET DUAL MODE AND CONVERT A CHANNEL THE FIRST CONVERSION
1224                                    ;SHOULD NOT EQUAL THE SECOND CONVERSION
1225                                    ;IF IT DOEs THE DUAL SAMPLE AND HOLD IS NOT WORKING CORRECTLY
1226
1227  007276  113737  015024  015073   DSHT6:  MOVB    STCHAN,$TMDAT+1 ;LOAD 1ST MUX CHANNEL
1228
1229                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1230  007314  052737  040000  015072           BIS     #BIT14,$TMDAT
1231
1232                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1233
1234                                    ;*      MOV     @ADCS,$TMDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $TMDAT.
1235  007342  052737  000001  015072           BIS     #1,$TMDAT
1236
1237                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1238  007360                            1$:
1239
1240                                    ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
1241  007370  105737  015074                   TSTB    $BDDAT          ;WAIT FOR DONE
1242  007374  100371                           BPL     1$
1243
1244                                    ;*      MOV     @ADCS,$TMDAT    ;/READ DEViCE REG ADCS,PUT DATA IN $TMDAT.
1245  007406  005237  015072                   INC     $TMDAT
1246
1247                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1248
1249                                    ;*      MOV     @ADDBR,AVERM2   ;/READ DEVICE REG ADDBR,PUT DATA IN AVERM2.
1250                                                                    ;SAVE 1ST CHANNEL CONVERSION
1251  007432                            2$:
1252
1253                                    ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
1254  007442  105737  015074                   TSTB    $BDDAT          ;WAIT FOR DONe
1255  007446  100371                           BPL     2$
1256
1257                                    ;*      MOV     @ADDBR,AVERM1   ;/READ DEVICE REG ADDBR,PUT DATA IN AVERM1.
1258
1259  007460  004537  007634                   JSR     R5,COMPR
1260  007464  015110                           AVERM2
1261  007466  015112                           AVERM1
1262  007470  001015                           BNE     DSHT7           ;BRANCH IF NOT EQUAL
1263  007472  032777  020000  171330           BIT     #BIT13,@SWR
1264  007500  001004                           BNE     DSHT6A
1265  007502  104000                           PRINT
1266  007504  014020                           CRLF
1267  007506  104000                           PRINT
1268  007510  014663                           MES23           ;PRINT ERROR
1269  007512  005777  171312           DSHT6A: TST     @SWR
1270  007516  100014                           BPL     DSHT7A
1271  007520  000000                           HALT            ;ERROR, FIRST CONVERSION ON CHANNEL X (STCHAN) WAS EQUAL
1272  007522  000412                           BR      DSHT7A  ;TO THE SECOND CONVERSION IN DUAL MODE
1273
```

```
1274
1275  007524  004537  007634          DSHT7:  JSR     R5,COMPR
1276  007530  015104                          AVERM4                  ;COMPARE TWO NUMBERS
1277  007532  015110                          AVERM2                  ;TEST IF NOT EQUAL
1278  007534  001021                          BNE     DSHT8
1279  007536  004537  007634                  JSR     R5,COMPR        ;COMPARE TWO SAMPLES
1280  007542  015106                          AVERM3
1281  007544  015112                          AVERM1
1282  007546  001014                          BNE     DSHT8           ;BRANCH IF NOT EQUAL
1283  007550  005237  015024          DSHT7A: INC     STCHAN          ;UPDATE CHANNEL
1284  007554  022737  000004  015024        CMP     #4,STCHAN       ;FINISHED ?
1285  007562  001402                          BEQ     1$              ;YES
1286  007564  000137  007014                  JMP     DSHT3
1287  007570  005037  015024          1$:     CLR     STCHAN          ;CLEAR CHANNEL
1288  007574  104013                          TSTTKS                  ;TEST KEYBOARD
1289  007576  000414                          BR      DSHT8B
1290
1291  007600  032777  020000  171222  DSHT8:  BIT     #BIT13,@SWR
1292  007606  001004                          BNE     DSHT8A
1293  007610  104000                          PRINT
1294  007612  014020                          CRLF
1295  007614  104000                          PRINT
1296  007616  014742                          MES24
1297  007620  005777  171204          DSHT8A: TST     @SWR
1298  007624  100001                          BPL     DSHT8B
1299  007626  000000                          HALT                    ;ERROR, PREVIOUS CONVERSTION ON CHANNEL NOT EQUAL
1300  007630  000137  007550          DSHT8B: JMP     DSHT7A          ; CURRENT CONVERSION
1301
1302  007634  013537  007724          COMPR:  MOV     @(R5)+,CMPRA    ;SAVE 1ST NUMBER
1303  007640  013537  007726                  MOV     @(R5)+,CMPRB    ;SAVE 2ND NUMBER
1304  007644  013737  007726  007730          MOV     CMPRB,CMPRC     ;SAVE 2ND AGAIN
1305  007652  163737  007724  007730          SUB     CMPRA,CMPRC     ;SUBTRACT 2ND FROM 1ST
1306  007660  001413                          BEQ     CMPR2           ;BRANCH IF EQUAL
1307  007662  100005                          BPL     CMPR1           ;BRANCH IF + <A-B=+>
1308  007664  063737  001010  007730          ADD     AMASK,CMPRC     ;ADD SPREAD
1309  007672  100006                          BPL     CMPR2           ;BRANCH IF <A=B+MASK>
1310  007674  000410                          BR      CMPR3           ;BRANCH IF
1311  007676  163737  001010  007730  CMPR1:  SUB     AMASK,CMPRC     ;SUBTRACT AMASK
1312  007704  003401                          BLE     CMPR2           ;BRANCH IF EQUAL *
1313  007706  000403                          BR      CMPR3           ;BRANCH IF NOT-EQUAL *
1314
1315  007710  005037  007730          CMPR2:  CLR     CMPRC           ;A AND B ARE WITHIN AMASK
1316  007714  000205                          RTS     R5
1317
1318  007716  005137  007730          CMPR3:  COM     CMPRC           ;A AND B ARE NOT WITHIN AMASK
1319  007722  000205                          RTS     R5
1320
1321  007724  000000          CMPRA:  0
1322  007726  000000          CMPRB:  0
1323  007730  000000          CMPRC:  0
```

```
                                        ;ROUTINE TO LOOP THUR A SINGLE LOGIC SUBTEST.  ENTERED FROM THE 'MONITOR'
                                        ;VIA TYPING 'TNN' WHERE 'NN' IS EQUATED TO THE 'PC' OF A SUBTEST.
                                        ;NOTE THAT 'SW11' MUST BE '0' (DOWN) TO RUN THIS TEST.

007732  012701  010546          TESTX:  MOV     #INBUF,R1
007736  005037  015006                  CLR     KSTOR1
007742  042711  177770          TSTA:   BIC     #177770,(R1)    ;MASK TO OCTAL
007746  062137  015006                  ADD     (R1)+,KSTOR1    ;ADD TO LAST RESULT
007752  005337  015002                  DEC     CHRCNT
007756  001407                          BEQ     TSTXB
007760  006337  015006                  ASL     KSTOR1
007764  006337  015006                  ASL     KSTOR1
007770  006337  015006                  ASL     KSTOR1
007774  000762                          BR      TSTA

007776  022737  005204  015006  TSTXB:  CMP     #CALBRT,KSTOR1  ;IS NO. WITHIN LIMITS OF THE LOGIC TEST
010004  100002                          BPL     .+6             ;CONTINUE IF YES
010006  000137  001774                  JMP     INIT3           ;OTHERWISE RETURN TO MONITOR
010012  062737  000002  015006          ADD     #2,KSTOR1       ;ADD '2' TO POINT TO INSTRUCTION AFTER SCOPE
010020  005037  013156          XLOOP:  CLR     SCOPEF          ;KEEP COUNT AT ZERO
010024  012737  010020  013160          MOV     #XLOOP,RETURN   ;LOAD SCOPE LOOP RETURN POINTER
010032  000177  004750                  JMP     @KSTOR1         ;JUMP TO TEST

                                        ;SUBROUTINE TO ISSUE N SPACES
                                        ;N IS ONE PLUS VALUE CONTAINED IN SPACEX
                                        ;SPACEX IS CLEARED WITHIN THE SUBROUTINE, SO THAT A CALL ON
                                        ;SPACE WITHOUT LOADING SPACEX ISSUES ONLY ONE SPACE

010036  105777  170762          XSPACE: TSTB    @TPS            ;WAIT FOR TTY READY
010042  100375                          BPL     .-4
010044  012777  000240  170754          MOV     #240,@TPB       ;OUTPUT A SPACE
010052  005337  010066                  DEC     SPACEX          ;DECREMENT COUNT
010056  003367                          BGT     XSPACE          ;LOOP IF NOT DONE
010060  005037  010066                  CLR     SPACEX          ;RESET COUNT TO ZERO
010064  000002                          RTI                     ;RETURN
010066  000000          SPACEX: 0

                                        ;SUBROUTINE TO TYPE OUT AVERAGES FOR THE RECOVERY TEST

010070  012737  015214  010100  XPRTAV: MOV     #ADBUFF,AVGTAB
010076  104010          XPTA1:  PRTOCT
010100  015214          AVGTAB: ADBUFF
010102  062737  000002  010100          ADD     #2,AVGTAB
010110  012737  000002  010066          MOV     #2,SPACEX
010116  104007                          SPACE
010120  005337  015052                  DEC     TEMP3
010124  001364                          BNE     XPTA1
010126  000002                          RTI
```

```
1372                                        ;KEYBOARD SERVICE ROUTINE
1373
1374  010130  010046                XTTYIN: MOV     R0,-(SP)
1375  010132  010146                        MOV     R1,-(SP)
1376  010134  010246                        MOV     R2,-(SP)
1377  010136  010346                        MOV     R3,-(SP)
1378  010140  010446                        MOV     R4,-(SP)
1379  010142  010546                        MOV     R5,-(SP)
1380  010144  012704  010546       NEWIN:  MOV     #INBUF,R4       ;SETUP CHARACTER BUFFER
1381  010150  042777  000100  170642        BIC     #BIT6,@TKS
1382  010156  005037  015002                CLR     CHRCNT          ;CLEAR CHARACTER COUNTER
1383  010162  005037  010546                CLR     INBUF
1384  010166  005037  010550                CLR     INBUF+2
1385  010172  005037  010552                CLR     INBUF+4
1386  010176  005037  010554                CLR     INBUF+6
1387  010202  005037  010556                CLR     INBUF+10
1388  010206  005037  010560                CLR     INBUF+12
1389  010212  005037  010562                CLR     INBUF+14
1390  010216  105777  170576       INPUTA: TSTB    @TKS            ;CHARACTER READY?
1391  010222  100375                        BPL     INPUTA          ;NO, WAIT IT OUT
1392  010224  017701  170572                MOV     @TKB,R1         ;SAVE CHARACTER
1393  010230  042701  177600                BIC     #177600,R1      ;STRIP PARITY BIT
1394  010234  120127  000060                CMPB    R1,#60          ;IS IT A SPECIAL CHARACTER
1395  010240  100420                        BMI     SPCHR           ;YES, TEST IT
1396  010242  122701  000173                CMPB    #173,R1
1397  010246  100415                        BMI     SPCHR
1398  010250  010124       INPUTB: MOV     R1,(R4)+        ;SAVE CHARACTER
1399  010252  005237  015002                INC     CHRCNT          ;INCREMENT THE CHARACTER COUNT.
1400  010256  022737  000006  015002        CMP     #6,CHRCNT
1401  010264  100524                        BMI     SPCHR5
1402  010266  105777  170532       OUTPTA: TSTB    @TPS            ;ECHO CHARACTER
1403  010272  100375                        BPL     OUTPTA
1404  010274  110177  170526                MOVB    R1,@TPB
1405  010300  000746                        BR      INPUTA          ;WAIT FOR NEXT CHARACTER
1406
1407                                        ;SUBROUTINE TO TEST FOR SPECIAL CHARACTERS :'↑A','↑C','↑G','CR',',' OR 'RUBOUT'
1408
1409  010302  122701  000001       SPCHR:  CMPB    #1,R1           ;CHAR.='↑A'
1410  010306  001016                        BNE     SPCHR1          ;NO, NOT '↑A'
1411  010310  104000                        PRINT                   ;ECHO '↑A'
1412  010312  013766                        CNTRLA                  ;RESTORE 'SP'
1413  010314  012605                        MOV     (SP)+,R5
1414  010316  012604                        MOV     (SP)+,R4
1415  010320  012603                        MOV     (SP)+,R3
1416  010322  012602                        MOV     (SP)+,R2
1417  010324  012601                        MOV     (SP)+,R1
1418  010326  012600                        MOV     (SP)+,R0
1419  010330  022626                        CMP     (SP)+,(SP)+
1420  010332  052777  000100  170460        BIS     #BIT6,@TKS      ;ENABLE KEYBOARD INTR.
1421  010340  000177  004432                JMP     @AVECTR         ;YES, EXIT VIA '↑A' VECTOR ADDRESS.
1422  010344  122701  000003       SPCHR1: CMPB    #3,R1           ;CHAR.='↑C'
1423  010350  001002                        BNE     1$              ;NO,NOT '↑C'
1424  010352  000137  001546                JMP     MONITR          ;YES, EXIT TO MONITOR
1425  010356  122701  000177       1$:     CMPB    #177,R1         ;CHAR. = 'RUBOUT'
```

```
1426  010362  001011                        BNE     SPCHR3          ;NO TRY ANOTHER CHAR
1427  010364  005737  015002                TST     CHRCNT          ;IS RUBOUT LEGAL?
1428  010370  001712                        BEQ     INPUTA          ;NO, IGNORE IT
1429  010372  005337  015002                DEC     CHRCNT
1430  010376  012701  000134                MOV     #134,R1         ;TYPE '\' TO INDICATE RUBOUUT
1431  010402  005744                        TST     -(R4)           ;POP OFF LAST CHARACTER
1432  010404  000730                        BR      OUTPTA          ;WAIT FOR NEXT CHARACTER
1433  010406  122701  000054        SPCHR3: CMPB    #54,R1          ;TEST FOR ','
1434  010412  001716                        BEQ     INPUTB          ;LEGAL CHAR., SAVE IT
1435  010414  122701  000015        SPCHR4: CMPB    #15,R1          ;=TO 'CARRIAGE RETURN' TO tERMINATE?
1436  010420  001014                        BNE     1$              ;NO, CONTINUE
1437  010422  104000                        PRINT                   ;YES, TYPE 'CR-LF'
1438  010424  014020                        CRLF
1439  010426  012605                4$:     MOV     (SP)+,R5
1440  010430  012604                        MOV     (SP)+,R4
1441  010432  012603                        MOV     (SP)+,R3
1442  010434  012602                        MOV     (SP)+,R2
1443  010436  012601                        MOV     (SP)+,R1
1444  010440  012600                        MOV     (SP)+,R0
1445  010442  052777  000100  170350        BIS     #BIT6,@TKS      ;ENABLE KEYBOARD INTR.
1446  010450  000002                        RTI                     ;EXIT
1447  010452  122701  000007        1$:     CMPB    #7,R1           ;TEST IF CTRL G
1448  010456  001027                        BNE     SPCHR5          ;BR IF NOT
1449  010460  104000                        PRINT
1450  010462  013771                        CNTRLG
1451  010464  104010                        PRTOCT                  ;PRINT OLD VALUE
1452  010466  000170                        SOFTSW
1453  010470  104000                        PRINT
1454  010472  014007                        NEWSWR                  ;ASK FOR NEW VALUE
1455  010474  104011                        TTYIN
1456  010476  005001                        CLR     R1
1457  010500  012700  010546                MOV     #INBUF,R0
1458  010504  005710                2$:     TST     (R0)            ;TEST FOR TERM
1459  010506  001410                        BEQ     3$              ;BR IF TERM
1460  010510  012002                        MOV     (R0)+,R2        ;GET CHAR
1461  010512  042702  177770                BIC     #177770,R2      ;MASK OUT
1462  010516  006301                        ASL     R1
1463  010520  006301                        ASL     R1
1464  010522  006301                        ASL     R1
1465  010524  060201                        ADD     R2,R1
1466  010526  000766                        BR      2$
1467  010530  010137  000170        3$:     MOV     R1,SOFTSW       ;SAVE NEW SWITCH VALUE
1468  010534  000734                        BR      4$
```

```
                                SPCHR5: PRINT                   ;OTHERWISE TYPE '?'
                                        QMARK
                                        JMP     XTTYIN          ; WAIT FOR NEW ENTRY
                                INBUF:  0                       ;CHARACTER STORAGE BUFFER

                                        .=.+14
                                ;SUBROUTINE WILL CONVERT 'N' WORDS (SEPARATED VIA COMMA'S)
                                ;WHICH WERE STORED IN A TABLE VIA 'TTYIN' TO OCTAL AND STORE THEM.

                                BCDBIN: MOV     #INBUF,R4       ;SETUP ASCII STORAGE TABLE
                                        MOV     #BCDTAB,R3      ;TABLE FOR STORAGE OF CONVERTED WORDS
                                        CLR     BCDTAB+2
                                BCDBN1: CLR     R1              ;REG. TO STORE RUNNING TOTAL
                                        CLR     R2              ;TEMP. STORAGE FOR 'R1'
                                BCDBN2: TST     CHRCNT          ;END OF DATA?
                                        BLE     BCDEND          ;YES, EXIT
                                        DEC     CHRCNT          ;DECREMENT CHARACTER COUNTER
                                        CMPB    #54,(R4)        ;IS CHARACTER = TO ','?
                                        BEQ     BCDEND          ;YES, DECODE NEW WORD
                                        CMPB    (R4),#60
                                        BLT     BCDERR          ;TEST FOR LEGAL NO.
                                        CMP     (R4),#67
                                        BGT     BCDERR
                                        BIC     #177770,(R4)    ;STRIP NO.
                                        MOV     (R4)+,R0        ;SAVE NO. IN R0.
                                        MOV     R1,R2           ;SAVE CURRENT TOTAL
                                        ASL     R1              ;NX2
                                        ASL     R1              ;NX4
                                        ASL     R1              ;NX8
                                        ADD     R0,R1           ;N+NEW NO.
                                        BR      BCDBN2
                                BCDEND: CMP     R1,#77
                                        BGT     BCDERR
                                        TST     (R4)+           ;UPDATE BUFFER
                                        MOV     R1,(R3)+        ;SAVE CONVERTED VALUE & SETUP TO SAVE NEXT
                                        TST     CHRCNT          ;FINISHED?
                                        BNE     BCDBN1          ;NO, CONVERT NEXT WORD
                                        RTI                     ;YES, EXIT
                                BCDERR: PRINT
                                        QMARK                   ;TYPE '?',
                                        TTYIN                   ;TO BE TYPED ON QUESTIONABLE ENTRIES.
                                        JMP     BCDBIN
                                BCDTAB: 0                       ;OCTAL STORAGE TABLE
                                        0
                                        0
                                        0
```

```
1516                                    ;SUBROUTINE TO TAKE 'N' CONVERSIONS AND STORE THEM IN AN A/D BUFFER. ROUTINE
1517                                    ;IS ENTERED WITH 'N' IN COUNT AND THE CH TO BE CONVERTED IN 'ADWORD'.
1518
1519  010726  005077  170064            ADCNVT: CLR     @PSW
1520  010732  013737  015004  015046            MOV     COUNT,TEMP1     ;SET UP # OF CONVERSIONS TO BE TAKEN
1521  010740  012704  015214                    MOV     #ADBUFF,R4      ;LOAD BUFFER ADDRESS
1522  010744  005037  015026                    CLR     FIRST           ;CLEAR TEMP
1523  010750  005737  015030                    TST     DELAY           ;TEST DELAY
1524  010754  001042                            BNE     6$              ;IF SET, SKIP
1525
1526                                    ;*      MOV     @ADCS,$TMDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $TMDAT.
1527  010766  113737  015073  001152            MOVB    $TMDAT+1,DMK2
1528  010774  123737  014771  001152            CMPB    ADWRD2+1,DMK2   ;TEST
1529  011002  001415                            BEQ     1$              ;BR IF SAME
1530  011004  113737  014771  015073            MOVB    ADWRD2+1,$TMDAT+1 ;LOAD MUX CHANNELS
1531
1532                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1533  011022  013737  011436  015032            MOV     CPTIME,DELAY1
1534  011030  005337  015032            2$:     DEC     DELAY1
1535  011034  001375                            BNE     2$              ;DELAY
1536  011036  005037  015030            1$:     CLR     DELAY
1537  011042  113737  014770  015072    4$:     MOVB    ADWRD2,$TMDAT   ;LOAD CONTROL BYTE
1538
1539                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1540  011060  000407                            BR      5$
1541  011062  013737  014770  015072    6$:     MOV     ADWRD2,$TMDAT   ;LOAD CONTROL WORD
1542
1543                                    ;*      MOV     $TMDAT,@ADCS    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ADCS
1544                                                                    ;LOAD CH. & START CONVERT IF NOT 'EXT'
1545  011100                            5$:
1546
1547                                    ;*      MOV     @ADCS,$BDDAT    ;/READ DEVICE REG ADCS,PUT DATA IN $BDDAT.
1548  011110  105737  015074                    TSTB    $BDDAT
1549  011114  100371                            BPL     5$              ;WAIT FOR DONE
1550  011116  005737  015074                    TST     $BDDAT
1551  011122  100016                            BPL     11$             ;BRANCH IT NOT SET
1552
1553                                    ;*      MOV     $ZERO,@ADCS     ;/ PUT DATA FROM $ZERO TO DEVICE REG ADCS
1554  011134  037727  167670  020000            BIT     @SWR,#BIT13
1555  011142  001002                            BNE     10$
1556  011144  104000                            PRINT
1557  011146  014236                            MES12
1558  011150  005777  167654            10$:    TST     @SWR
1559  011154  100001                            BPL     11$
1560  011156  000000                            HALT                    ;ERROR BIT SET
1561
1562  011160  005737  015034            11$:    TST     USEDSH
1563  011164  001410                            BEQ     12$             ;BRANCH IF NOT DUAL MODE
1564  011166  005137  015026                    COM     FIRST
1565  011172  001405                            BEQ     12$
1566
1567                                    ;*      MOV     @ADDBR,BRLEV1   ;/READ DEVICE REG ADDBR,PUT DATA IN BRLEV1.
1568  011204  000716                            BR      4$
1569  011206                            12$:
```

```
1570
1571                                    ;*      MOV     @ADDBR,$TMDAT   ;/READ DEVICE REG ADDBR,PUT DATA IN $TMDAT.
1572  011216  013724  015072                    MOV     $TMDAT,(4)+
1573  011222  005337  015046                    DEC     TEMP1           ;DECREMENT COUNTER
1574  011226  003305                            BGT     4$              ;IF NOT '0' TAKE NEXT CONVERSION
1575  011230  000207                            RTS     PC
1576
```

```
1577
1578                                        ;POWER FAIL HANDLER
1579
1580  011232  010046                PWRFAL: MOV     R0,-(SP)
1581  011234  010146                        MOV     R1,-(SP)
1582  011236  010246                        MOV     R2,-(SP)
1583  011240  010346                        MOV     R3,-(SP)
1584  011242  010446                        MOV     R4,-(SP)
1585  011244  010546                        MOV     R5,-(SP)
1586  011246  013746  000024               MOV     24,-(SP)
1587  011252  010637  015000               MOV     SP,PROC
1588  011256  012737  011266  000024       MOV     #PWRUP,24
1589  011264  000000                       HALT
1590
1591                                        ;POWER UP HANDLER
1592
1593  011266  012777  000340  167522 PWRUP: MOV     #340,@PSW
1594  011274  013706  015000               MOV     PROC,SP
1595  011300  012637  000024               MOV     (SP)+,24
1596  011304  012605                       MOV     (SP)+,R5
1597  011306  012604                       MOV     (SP)+,R4
1598  011310  012603                       MOV     (SP)+,R3
1599  011312  012602                       MOV     (SP)+,R2
1600  011314  012601                       MOV     (SP)+,R1
1601  011316  012600                       MOV     (SP)+,R0
1602  011320  004737  021526               JSR     PC,$RESET
1603  011324  104000                       PRINT
1604  011326  014565                       MES21
1605  011330  000137  001546               JMP     MONITR
```

```
                                        ;EMT DISPATCH SERVICE ROUTINE
                                        ;ARGUMENT OF EMT IS EXTRACTED AND USED AS OFFSET TO OBTAIN POINTER
                                        ;TO THE SELECTED SUBROUTINE.

011334  011646              EMTSRV: MOV     (SP),-(SP)      ;GET PC FOR TO RETURN
011336  162716  000002              SUB     #2,(SP)         ;PC OF EMT
011342  017616  000000              MOV     @(SP),(SP)      ;GET EMT
011346  005716                      TST     (SP)            ;IS EMT VALID?
011350  001001                      BNE     EMTOK
011352  000000                      HALT                    ;INVALID EMT
011354  006316              EMTOK:  ASL     (SP)            ;MULTIPLY EMT ARG BY '2'
011356  042716  177001              BIC     #177001,(SP)    ;CLEAR UNWANTED BITS
011362  062716  011374              ADD     #EMTTAB,(SP)    ;POINTER TO SUBROUTINE ADDRESS
011366  017616  000000              MOV     @(SP),(SP)      ;SUBROUTINE ADDRESS
011372  000136                      JMP     @(SP)+          ;GO TO SUBROUTINE

                                        ;EMT DISPATCH TABLE

011374  011440              EMTTAB: TYPMES                  ;MESSAGE PRINT ROUTINE
011376  010564                      BCDBIN                  ;BINARY CONVERSION ROUTINE
011400  013066                      SCOPEC                  ;LOGIC TEST SCOPE ROUTINE
011402  013162                      SCOPEH                  ;LOGIC TEST SCOPE LOOP (10)
011404  011774                      CMPTE                   ;SUBROUTINE TO COMPUTE THE AVG
011406  012162                      CATORZ                  ;SUBROUTINE TO COMPUTE 'COUNT SPREAD'
011410  011576                      DECPRT                  ;SUBROUTINE TO CONVERT OCT TO DEC + PRINT
011412  010036                      XSPACE                  ;SUBROUTINE TO TYPE SPACES
011414  012522                      OCTPRT                  ;OCTAL PRINT ROUTINE
011416  010130                      XTTYIN                  ;TELEPRINTER SERVICE ROUTINE
011420  010070                      XPRTAV                  ;SUBROUTINE TO PRINT OUT THE GAIN AVERAGES
011422  012652                      TKSFLG                  ;SUBROUTINE TO TEST FOR KEYBOARD FLAG
011424  013222                      SCOPEI                  ;SCOPE ROUTINE

011426  002000              CPDLAY: 2000                    ;PDP-11/05
011430  002400                      2400                    ;PDP-11/20
011432  003500                      3500                    ;PDP-11/40
011434  006000                      6000                    ;PDP-11/45

011436  002000              CPTIME: 2000
```

```
1646                                    ;MESSAGE PRINT ROUTINE, ENTERED VIA EMT DISPATCH HANDLER.
1647                                    ;ROUTINE PICKS UP CONTENTS OF THE 'PC' AND USES THIS AS
1648                                    ;THE ADDRESS OF MESSAGE TO BE TYPED.
1649
1650  011440  005077  167352           TYPMES: CLR     @PSW            ;ENABLE KEYBOARD INTR.
1651  011444  010537  012650                   MOV     R5,SAV5
1652  011450  017605  000000                   MOV     @(SP),R5        ;GET THE MESSAGE ADDRESS FROM START
1653  011454  062716  000002                   ADD     #2,(SP)         ;SET UP STACK TO EXIT
1654  011460  105777  167340           TYPERA: TSTB    @TPS
1655  011464  100375                           BPL     TYPERA          ;WAIT FOR TTY DONE
1656  011466  122715  000100                   CMPB    #100,(R5)       ;TEST FOR '@'
1657  011472  001003                           BNE     TYPER1          ;BRANCH IF NO EQUAL
1658  011474  013705  012650                   MOV     SAV5,R5
1659  011500  000002                           RTI                     ;OTHERWISE EXIT
1660  011502  122715  000045           TYPER1: CMPB    #45,(R5)        ;TEST FOR '%'
1661  011506  001403                           BEQ     TYPECL          ;IF = TYPE 'CR-LF'
1662  011510  112577  167312           TYPER2: MOVB    (R5)+,@TPB      ;OUTPUT CHAR.
1663  011514  000761                           BR      TYPERA
1664  011516  012777  000015  167302   TYPECL: MOV     #15,@TPB        ;TYPE 'CR'
1665  011524  105777  167274                   TSTB    @TPS
1666  011530  100375                           BPL     .-4
1667  011532  012777  000012  167266           MOV     #12,@TPB
1668  011540  013737  001012  011574           MOV     FILLS,2$        ;LOAD FILL COUNT
1669  011546  105777  167252           1$:     TSTB    @TPS
1670  011552  100375                           BPL     1$
1671  011554  113777  001014  167244           MOVB    FILCHR,@TPB     ;FILL CHAR
1672  011562  005337  011574                   DEC     2$
1673  011566  100367                           BPL     1$
1674  011570  105725                           TSTB    (R5)+           ;INCREMENT BUFFER
1675  011572  000732                           BR      TYPERA
1676  011574  000002           2$:     2
1677
1678                                    ;PRINT DECIMAL VALUE IN R2
1679
1680  011576  005077  167214           DECPRT: CLR     @PSW
1681  011602  012737  177774  011756           MOV     #-4,DIGCNT
1682  011610  012737  011764  011762           MOV     #DECPNT+2,DECPNT
1683  011616  012737  000240  011760           MOV     #240,ZERO
1684  011624  012737  177777  011754   TYPT1:  MOV     #-1,DIGIT
1685  011632  005237  011754           TYPT2:  INC     DIGIT
1686  011636  167702  000120                   SUB     @DECPNT,%2
1687  011642  100373                           BPL     TYPT2
1688  011644  067702  000112                   ADD     @DECPNT,%2
1689  011650  004737  011674                   JSR     PC,DECOUT
1690  011654  005237  011756                   INC     DIGCNT
1691  011660  001001                           BNE     TYPT3
1692  011662  000002                           RTI
1693  011664  062737  000002  011762   TYPT3:  ADD     #2,DECPNT
1694  011672  000754                           BR      TYPT1
1695  011674  005737  011754           DECOUT: TST     DIGIT
1696  011700  001010                           BNE     DEC1
1697  011702  022737  177777  011756           CMP     #-1,DIGCNT
1698  011710  001404                           BEQ     DEC1
1699  011712  013737  011760  011754           MOV     ZERO,DIGIT
```

```
 1700  011720  000406                          BR      DEC2
 1701  011722  012737  000260  011760  DEC1:   MOV     #260,ZERO
 1702  011730  052737  000260  011754          BIS     #260,DIGIT
 1703  011736  105777  167062          DEC2:   TSTB    @TPS
 1704  011742  100375                          BPL     .-4
 1705  011744  013777  011754  167054          MOV     DIGIT,@TPB
 1706  011752  000207                          RTS     7
 1707
 1708  011754  000000                  DIGIT:  0
 1709  011756  000000                  DIGCNT: 0
 1710  011760  000240                  ZERO:   240
 1711  011762  011764                  DECPNT: .+2
 1712  011764  001750                          1000.
 1713  011766  000144                          100.
 1714  011770  000012                          10.
 1715  011772  000001                          1.
 1716
 1717                                  ;COMPUTE THE RESULTS OF 512 CONVERSIONS AS HIGH, LOW AND AVERAGE
 1718
 1719  011774  012737  000777  015046  CMPTE:  MOV     #777,TEMP1      ;SET UP TO COMPARE '511' NUMBERS
 1720  012002  012704  015214                  MOV     #ADBUFF,R4      ;SET UP DATA BUFFER ADDRESS
 1721  012006  012737  000011  012160          MOV     #11,CMPCNT      ;LOAD AVRG. COUNT
 1722  012014  005037  015064          CMPTEA: CLR     HIORDV
 1723  012020  012437  015114                  MOV     (R4)+,AVRAGE    ;STORE 1ST VALUE AS AVERAGE
 1724  012024  013737  015114  015066          MOV     AVRAGE,HIGH     ;HIGH
 1725  012032  013737  015114  015070          MOV     AVRAGE,LOW      ;& LOW
 1726  012040  012437  015050          GETDAT: MOV     (R4)+,TEMP2
 1727  012044  023737  015050  015066          CMP     TEMP2,HIGH      ;IS NEW NO. GREATER THAN OLD NO.
 1728  012052  003403                          BLE     TSLO            ;BRANCH IF NOT GREATER
 1729  012054  013737  015050  015066          MOV     TEMP2,HIGH      ;OTHERWISE SAVE AS NEW HIGH
 1730  012062  023737  015050  015070  TSLO:   CMP     TEMP2,LOW
 1731  012070  003003                          BGT     TAGA
 1732  012072  013737  015050  015070          MOV     TEMP2,LOW       ;OTHERWISE SAVE AS NEW LOW
 1733  012100  063737  015050  015114  TAGA:   ADD     TEMP2,AVRAGE    ;ADD LOW ORDER
 1734  012106  005537  015064                  ADC     HIORDV          ;ADD CARRY TO HI ORDER
 1735  012112  005337  015046                  DEC     TEMp1
 1736  012116  001350                          BNE     GETDAT          ;512 OR 16 ADDITIONS?
 1737  012120  013737  012160  015046          MOV     CMPCNT,TEMP1    ;YES, DIVIDE/512 OR 16
 1738  012126  006237  015064          AVGDAT: ASR     HIORDV
 1739  012132  006037  015114                  ROR     AVRAGE          ;SHIFT CARRY BIT INTO LO ORDER
 1740  012136  005337  015046                  DEC     TEMp1
 1741  012142  001371                          BNE     AVGDAT          ;DONE?
 1742  012144  005537  015114                  ADC     AVRAGE          ;YES, ADD REMAINDER TO LO ORDER
 1743  012150  004537  013244                  JSR     5,LEDS
 1744  012154  015114                          AVRAGE
 1745  012156  000002                          RTI
 1746
 1747  012160  000011                  CMPCNT: 11                      ;11 OR 4
```

```
1748                                        ;SUBROUTINE TO CALCULATE THE PLUS & MINUS 5 COUNT LIMITS FROM AN AVERAGE
1749
1750  012162  012737  000005  015046  CATORZ: MOV     #5,TEMP1
1751  012170  013737  015114  015050          MOV     AVRAGE,TEMP2    ;MOV AVER. TO WORK AREA
1752  012176  012703  015116                  MOV     #AVERP1,R3      ;SETUP DISTRIBUTION TABLE (POS.)
1753  012202  005237  015050          FILE1:  INC     TEMP2           ;A=A+1
1754  012206  013723  015050                  MOV     TEMP2,(R3)+     ;SAVE A+1
1755  012212  005337  015046                  DEC     TEMP1           ;SAVED '5' COUNTS?
1756  012216  001371                          BNE     FILE1           ;BRANCH IF NO
1757                                        ;SET UP TABLE OF AVG. -1 TO -5
1758  012220  012737  000005  015046          MOV     #5,TEMP1
1759  012226  013737  015114  015050          MOV     AVRAGE,TEMP2    ;MOV AVG. TO WORK AREA.
1760  012234  012703  015114                  MOV     #AVRAGE,R3      ;SET UP DISTRIBUTION TABLE NEG.
1761  012240  005337  015050          FILE2:  DEC     TEMP2           ;A=1-1
1762  012244  013743  015050                  MOV     TEMP2,-(R3)     ;SAVE 'A-1'
1763  012250  005337  015046                  DEC     TEMP1           ;SAVED '5' COUNTS?
1764  012254  001371                          BNE     FILE2           ;BRANCH IF NO
1765
1766                                        ;CATEGORIZE THE COUNT SPREAD AS '+6 & -6' COUNTS FROM THE AVERAGE
1767
1768  012256  012703  015130                  MOV     #ORLOW,R3       ;CLEAR COUNTS
1769  012262  005023          CATR1:  CLR     (R3)+
1770  012264  022703  015162                  CMP     #ORHIGH+2,R3    ;FINISHED?
1771  012270  001374                          BNE     CATR1           ;NO, CLEAR NEXT COUNTER
1772  012272  012737  001001  015046          MOV     #1001,TEMP1     ;COMPARE '512' COUNTS
1773  012300  012700  015214                  MOV     #ADBUFF,R0      ;SET UP A/D BUFFER
1774  012304  005337  015046          CATR2:  DEC     TEMP1
1775  012310  001437                          BEQ     CATR5           ;EXIT IF '0'
1776  012312  012037  015050                  MOV     (R0)+,TEMP2
1777  012316  023737  015126  015050          CMP     AVERP5,TEMP2
1778  012324  100423                          BMI     OVRHI
1779  012326  023737  015050  015102          CMP     TEMP2,AVERM5
1780  012334  100422                          BMI     OVRLO
1781  012336  005001                          CLR     R1
1782  012340  012702  015102                  MOV     #AVERM5,R2
1783  012344  022237  015050          CATR3:  CMP     (R2)+,TEMP2
1784  012350  001405                          BEQ     CATR4
1785  012352  005201                          INC     R1
1786  012354  022701  000013                  CMP     #13,R1
1787  012360  001371                          BNE     CATR3
1788  012362  000000                          HALT                    ;FATAL ERROR
1789  012364  006301          CATR4:  ASL     R1                      ;MULTIPLY 'OFFSET' X2
1790  012366  005261  015132                  INC     MINUS5(R1)
1791  012372  000744                          BR      CATR2
1792  012374  005237  015160          OVRHI:  INC     ORHIGH
1793  012400  000741                          BR      CATR2
1794  012402  005237  015130          OVRLO:  INC     ORLOW
1795  012406  000736                          BR      CATR2
```

```
1796
1797                                    ;ADD THE COUNTS AND SAVE TOTAL IN SPREADS OF '1-4'
1798
1799  012410  013737  015144  015162    CATR5:  MOV     AVGCNT,XSPRD1
1800  012416  063737  015146  015162            ADD     PLUS1,XSPRD1
1801  012424  063737  015142  015162            ADD     MINUS1,XSPRD1   ;=TO NO. COUNTS AT SPREAD OF '1'
1802  012432  013737  015162  015164            MOV     XSPRD1,XSPRD2
1803  012440  063737  015150  015164            ADD     PLUS2,XSPRD2
1804  012446  063737  015140  015164            ADD     MINUS2,XSPRD2   ;=TO NO. COUNTS AT SPREAD OF '2'
1805  012454  013737  015164  015166            MOV     XSPRD2,XSPRD3
1806  012462  063737  015152  015166            ADD     PLUS3,XSPRD3
1807  012470  063737  015136  015166            ADD     MINUS3,XSPRD3   ;=TO NO. COUNTS AT SPREAD OF '3'
1808  012476  013737  015166  015170            MOV     XSPRD3,XSPRD4
1809  012504  063737  015154  015170            ADD     PLUS4,XSPRD4
1810  012512  063737  015134  015170            ADD     MINUS4,XSPRD4   ;=TO NO. COUNTS AT SPREAD OF '4'
1811  012520  000002                            RTI                     ;EXIT
1812
1813                                    ;SUBROUTINE TO TYPEOUT A '6' DIGIT OCTAL NO. THE 'PC' CONTAINS
1814                                    ;THE ADDRESS OF 'WORD' TO BE TYPED
1815
1816  012522  010537  012650    OCTPRT: MOV     R5,SAV5
1817  012526  017605  000000            MOV     @(SP),R5        ;THE ADDRESS OF WORD TO BE TYPED
1818  012532  062716  000002            ADD     #2,(SP)         ;SET UP STACK TO EXIT
1819  012536  012737  000006  012642    MOV     #6,10$          ;LOAD COUNTER
1820  012544  012737  000376  012646    MOV     #376,MASK       ;MASK FOR FIRST BIT
1821  012552  000401                    BR      .+4
1822  012554  006115            1$:     ROL     (R5)
1823  012556  006115                    ROL     (R5)
1824  012560  006115                    ROL     (R5)
1825  012562  111537  012644            MOVB    (R5),11$
1826  012566  143737  012646  012644    BICB    MASK,11$
1827  012574  052737  000260  012644    BIS     #260,11$
1828  012602  132777  000200  166214    BITB    #200,@TPS
1829  012610  100374                    BPL     .-6             ;WAIT FOR PRINTER READY
1830  012612  113777  012644  166206    MOVB    11$,@TPB        ;PRINT CHAR.
1831  012620  012737  000370  012646    MOV     #370,MASK       ;MASK FOR NEXT '5' DIGITS
1832  012626  005337  012642            DEC     10$
1833  012632  001350                    BNE     1$
1834  012634  013705  012650            MOV     SAV5,R5
1835  012640  000002                    RTI
1836  012642  000000            10$:    0
1837  012644  000000            11$:    0
1838  012646  000376            MASK:   376
1839  012650  000000            SAV5:   0
1840
1841                                    ;SUBROUTINE TO TEST FOR THE KEYBOARD FLAG BEING SET
1842
1843  012652  105777  166142    TKSFLG: TSTB    @TKS            ;FLAG SET?
1844  012656  100001                    BPL     .+4             ;NO, EXIT
1845  012660  104011                    TTYIN                   ;YES,INQUIRE
1846
1847
1848  012662  000002                    RTI
1849                                    ;ENTERED WITH SYSTEM TRAP CALL (ERROR)
```

```
 1850  012664  104013                LOGERR: TSTTKS                  ;TEST FOR KEYBOARD INTERRUPT
 1851  012666  037727  166136  020000        BIT     @SWR,#20000     ;TEST FOR INHIBIT PRINT OUT
 1852  012674  001067                        BNE     CK              ;INHIBIT,CHECK FOR HALT
 1853  012676  012737  000002  000006        MOV     #RTI,@#6        ;SET UP FOR BUSS ERROR
 1854  012704  011637  015012                MOV     (SP),KSTOR3     ;PC OF FAILING ROUTINE
 1855  012710  162737  000002  015012        SUB     #2,KSTOR3
 1856  012716  004537  013244                JSR     5,LEDS
 1857  012722  015012                        KSTOR3
 1858
 1859                              ;*        MOV     @ADCS,KSTOR4    ;/READ DEVICE REG ADCS,PUT DATA IN KSTOR4.
 1860
 1861                              ;*        MOV     @ADDBR,KSTOR5   ;/READ DEVICE REG ADDBR,PUT DATA IN KSTOR5.
 1862  012744  005037  015022                CLR     KSTR12          ;SAVE ZERO IN KSTR12
 1863  012750  013737  015044  015020        MOV     TEMP,KSTR11     ;SAVE TEMP
 1864  012756  012737  000000  000006        MOV     #0,@#6          ;RESET BUSS ERROR
 1865  012764  005737  014772                TST     PRINT1
 1866  012770  001006                        BNE     LGERR1
 1867  012772  104000                        PRINT
 1868  012774  014020                        CRLF
 1869  012776  104000                        PRINT
 1870  013000  013453                        MES1
 1871  013002  005237  014772                INC     PRINT1
 1872  013006  104000                LGERR1: PRINT                   ;OUTPUT CARRIAGE RETURN AND LINE FEED
 1873  013010  014020                        CRLF
 1874  013012  104010                        PRTOCT                  ;PRINT FAILING PC+2
 1875  013014  015012                        KSTOR3
 1876  013016  104007                        SPACE                   ;OUTPUT A SPACE
 1877  013020  104010                        PRTOCT                  ;PRINT
 1878  013022  015014                        KSTOR4                  ;CONTENTS OF A/D STATUS REG.
 1879  013024  104007                        SPACE
 1880  013026  104010                        PRTOCT
 1881  013030  015016                        KSTOR5
 1882  013032  104007                        SPACE
 1883  013034  104010                        PRTOCT
 1884  013036  015022                        KSTR12
 1885  013040  104007                        SPACE
 1886  013042  104010                        PRTOCT
 1887  013044  015020                        KSTR11
 1888  013046  105777  165752                TSTB    @TPS
 1889  013052  100375                        BPL     .-4
 1890  013054  005777  165750        CK:     TST     @SWR            ;CHECK SR FOR HALT SWITCH
 1891  013060  100001                        BPL     .+4             ;BRANCH IF NOT SET
 1892  013062  000000                        HALT                    ;HALT ON ERROR UP
 1893  013064  000002                        RTI                     ;RETURN TO MAIN LINE
```

```
1894
1895                                    ;SCOPE AND/OR ITERATION LOOP FOR SOME TEST 4000 TIMES
1896
1897  013066  104013                    SCOPEC: TSTTKS                  ;TEST FOR KEYBOARD INIT
1898  013070  032777  040000  165732            BIT     #40000,@SWR     ;TEST SR FOR SCOPE
1899  013076  001015                            BNE     SCOPEB          ;YES, SCOPE
1900  013100  032777  004000  165722            BIT     #4000,@SWR      ;NO-TEST FOR ITERATION
1901  013106  001016                            BNE     SCOPEG          ;INHIBIT ITERATION
1902  013110  005737  001034                    TST     PASSCT          ;TEST IF FIRST PASS
1903  013114  001413                            BEQ     SCOPEG          ;BR IF FIRST PASS -- QUCIK PASS
1904  013116  023737  013156  001036            CMP     SCOPEF,ICOUNT   ;COMPARE CURRENT COUNT TO MAX NUMBER
1905  013124  001407                    SCOPEJ: BEQ     SCOPEG          ;EXIT-DONE
1906  013126  005237  013156                    INC     SCOPEF          ;INCREMENT COUNT
1907  013132  022606                    SCOPEB: CMP     (6)+,SP         ;REPOSITION STACK
1908  013134  012677  165656                    MOV     (6)+,@PSW       ;RESTORE PREVIOUS PROCESSOR STATUS
1909  013140  000177  000014                    JMP     @RETURN         ;REPEAT TEST
1910  013144  005037  013156            SCOPEG: CLR     SCOPEF          ;CLEAR COUNT
1911  013150  011637  013160                    MOV     @SP,RETURN      ;SAVE SCOPE RETURN POINTER
1912  013154  000002                            RTI                     ;RETURN INLINE-NEXT TEST
1913  013156  000000            SCOPEF: 0                               ;COUNT LOCATION FOR ITERATION LOOP
1914  013160  002162            RETURN: ADT1+2                          ;ADDRESS OF LAST TEST
1915
1916                                    ;SCOPE AND/OR INTERATION LOOP FOR SOME TESTS 10 TIMES
1917
1918  013162  104013                    SCOPEH: TSTTKS
1919  013164  032777  040000  165636            BIT     #40000,@SWR
1920  013172  001357                            BNE     SCOPEB
1921  013174  032777  004000  165626            BIT     #4000,@SWR
1922  013202  001360                            BNE     SCOPEG
1923  013204  005737  001034                    TST     PASSCT          ;TEST IF FIRST PASS
1924  013210  001755                            BEQ     SCOPEG          ;BR IF YES
1925  013212  023727  013156  000010            CMP     SCOPEF,#10
1926  013220  000741                            BR      SCOPEJ
1927
1928                                    ;SCOPE LOOP FOR SOME TEST 1 TIME
1929
1930  013222  104013                    SCOPEI: TSTTKS                  ;TEST KEYBOARD
1931  013224  032777  040000  165576            BIT     #40000,@SWR
1932  013232  001337                            BNE     SCOPEB
1933  013234  000743                            BR      SCOPEG
```

```
1934
1935  013236  104000                    BELL:   PRINT
1936  013240  013354                            ENDPAS
1937  013242  000207                            RTS     PC              ;EXIT
1938
1939                                    ;LOAD THE LPS DISPLAY LIGHTS
1940
1941  013244  012737  000006  013344    LEDS:   MOV     #6,CNTLED
1942  013252  005037  013352                    CLR     LEDSV3
1943  013256  013537  013346                    MOV     @(5)+,LEDSV1
1944  013262  013737  013346  013350    LEDSA:  MOV     LEDSV1,LEDSV2
1945  013270  042737  177770  013350            BIC     #177770,LEDSV2
1946  013276  113737  013352  013351            MOVB    LEDSV3,LEDSV2+1
1947
1948                                    ;*      MOV     LEDSV2,@ADDBR   ;/ PUT DATA FROM LEDSV2 TO DEVICE REG ADDBR
1949  013314  006237  013346                    ASR     LEDSV1
1950  013320  006237  013346                    ASR     LEDSV1
1951  013324  006237  013346                    ASR     LEDSV1
1952  013330  005237  013352                    INC     LEDSV3
1953  013334  005337  013344                    DEC     CNTLED
1954  013340  001350                            BNE     LEDSA
1955  013342  000205                            RTS     5
1956
1957  013344  000006                    CNTLED: 6
1958  013346  000000                    LEDSV1: 0
1959  013350  000000                    LEDSV2: 0
1960  013352  000000                    LEDSV3: 0
1961
1962  013354      007       007         ENDPAS: .BYTE   7,7
1963  013356  042445  042116  050040            .ASCII  '%END PASS   @'
1964  013364  051501  020123  020040
1965  013372      100
```

```
                                        ;MESSAGES
013373      000                         .BYTE
013374  022445  046045  051520  TITLE:  .ASCII  '%%%LPS DIAGNOSTIC TEST I, (MAINDEC-11-DRLPH-A)@'
013402  042040  040511  047107
013410  051517  044524  020103
013416  042524  052123  044440
013424  020054  046450  044501
013432  042116  041505  030455
013440  026461  051104  050114
013446  026510  024501      100
013453      040  050040  020103  MES1:   .ASCII  "  PC   "
013460  020040
013462  042101  052123  052101          .ASCII  "ADSTAT "
013470      040
013471      101  041104  043125          .ASCII  "ADBUFF "
013476  020106
013500  040440  042104  040515          .ASCII  " ADDMA "
013506      040
013507      040  042524  050115          .ASCII  " TEMP@"
013514      100
013515      045  054524  042520  MES4:   .ASCII  "%TYpE LETTER ' ' TO RUN DESIRED TEST:%"
013522  046040  052105  042524
013530  020122  020047  020047
013536  047524  051040  047125
013544  042040  051505  051111
013552  042105  052040  051505
013560  035124      045
013563      047  023501  040475          .ASCII  "'A'=A TO d LOGIC TEST%"
013570  052040  020117  020104
013576  047514  044507  020103
013604  042524  052123      045
013611      047  023503  040475          .ASCII  "'C'=A TO D CALIBRATION%"
013616  052040  020117  020104
013624  040503  044514  051102
013632  052101  047511  022516
013640  042047  036447  020101          .ASCII  "'D'=A TO D REPEATABILITY%"
013646  047524  042040  051040
013654  050105  040505  040524
013662  044502  044514  054524
013670      045
013671      047  023505  040475          .ASCII  "'E'=A TO D RECOVERY%"
013676  052040  020117  020104
013704  042522  047503  042526
013712  054522      045
013715      047  023506  040475          .ASCII  "'F'=A TO D DUAl MODE DYNAMIC LOGIC %@"
013722  052040  020117  020104
013730  052504  046101  046440
013736  042117  020105  054504
013744  040516  044515  020103
013752  047514  044507  020103
```

```
2020  013760  040045
2021
2022  013762  041536  040045          CNTRLC: .ASCII  '↑C%@'
2023
2024  013766  040536     100          CNTRLA: .ASCII  '↑A@'
2025  013771     136  022507  046117  CNTRLG: .ASCII  '↑G%OLD SWR = @'
2026  013776  020104  053523  020122
2027  014004  020075     100
2028  014007     040  047040  053505  NEWSWR: .ASCII  '  NEW = @'
2029  014014  036440  040040
2030
2031  014020  040045                  CRLF:   .ASCII  '%@'
2032
2033  014022  027045     100          DOT:    .ASCII  '%.@'
2034
2035  014025     077  040040          QMARK:  .ASCII  '? @'
2036
2037  014030  021045  020101  047524  MES5:   .ASCII  '%"A TO D LOGIC TEST"% @'
2038  014036  042040  046040  043517
2039  014044  041511  052040  051505
2040  014052  021124  020045     100
2041
2042  014057     045  040442  052040  MES7:   .ASCII  '%"A TO D CALIBRATION"%@'
2043  014064  020117  020104  040503
2044  014072  044514  051102  052101
2045  014100  047511  021116  040045
2046  014106  021045  020101  047524  MES8:   .ASCII  '%"A TO D RECOVERY"@'
2047  014114  042040  051040  041505
2048  014122  053117  051105  021131
2049  014130     100
2050
2051  014131     045  044103  040040  MES9:   .ASCII  '%CH @'
2052
2053  014136  023445  023505  052130  MES10:  .ASCII  "%'E'XT. OR 'I'NT. OR 'C'LOCK. SYNC? @"
2054  014144  020056  051117  023440
2055  014152  023511  052116  020056
2056  014160  051117  023440  023503
2057  014166  047514  045503  020056
2058  014174  054523  041516  020077
2059  014202     100
2060
2061
2062  014203     045  040442  052040  MES11:  .ASCII  '%"A TO D DMA LOGIC TEST"% @'
2063  014210  020117  020104  046504
2064  014216  020101  047514  044507
2065  014224  020103  042524  052123
2066  014232  022442  040040
2067  014236  051105  047522  020122  MES12:  .ASCII  'ERROR BIT SET!%@'
2068  014244  044502  020124  042523
2069  014252  020524  040045
2070
2071  014256  021045  020101  047524  MES13:  .ASCII  '%"A TO D REPEATIBILITY"@'
2072  014264  042040  051040  050105
2073  014272  040505  044524  044502
```

```
2074  014300  044514  054524  040042
2075
2076  014306  022445  044103  024056  MES14:  .ASCII  '%%CH.(S)? @'
2077  014314  024523  020077     100
2078  014321     045  040442  052040  MES15:  .ASCII  '%"A TO D DUAL MODE"%@'
2079  014326  020117  020104  052504
2080  014334  046101  046440  042117
2081  014342  021105  040045
2082
2083  014346  047503  047125  020124  MES16:  .ASCII  'COUNT SPREAD? @'
2084  014354  050123  042522  042101
2085  014362  020077     100
2086  014365     045  044447  047047  MES18:  .ASCII  "%'I'NTERNAL OR 'B'URST OR 'D'UAL SAMPLE MODE? @"
2087  014372  042524  047122  046101
2088  014400  047440  020122  041047
2089  014406  052447  051522  020124
2090  014414  051117  023440  023504
2091  014422  040525  020114  040523
2092  014430  050115  042514  046440
2093  014436  042117  037505  040040
2094  014444  020045  041440  027110  MES19:  .ASCII  '%  CH.    LO     AV     HI@'
2095  014452  020040  020040  047514
2096  014460  020040  020040  040440
2097  014466  020126  020040  020040
2098  014474  044510     100
2099
2100  014477     045  020040  047514  MES20:  .ASCII  '%  LO  -5  -4  -3  -2  -1  AV  +1  +2  +3  +4  +5  HI@'
2101  014504  020040  032455  020040
2102  014512  032055  020040  031455
2103  014520  020040  031055  020040
2104  014526  030455  020040  053101
2105  014534  020040  030453  020040
2106  014542  031053  020040  031453
2107  014550  020040  032053  020040
2108  014556  032453  020040  044510
2109  014564     100
2110
2111  014565     045  047520  042527  MES21:  .ASCII  '%POWER FAILURE @'
2112  014572  020122  040506  046111
2113  014600  051125  020105     100
2114  014605     111  050116  052125  MES22:  .ASCII  'INPUT CHANNEL PAIR HAVE THE SAME INPUT VALUE%@'
2115  014612  041440  040510  047116
2116  014620  046105  050040  044501
2117  014626  020122  040510  042526
2118  014634  052040  042510  051440
2119  014642  046501  020105  047111
2120  014650  052520  020124  040526
2121  014656  052514  022505     100
2122  014663     105  051122  051117  MES23:  .ASCIZ  'ERROR - FIRST CONVERSION EQUAL TO THE SECOND%@'
2123  014670  026440  043040  051111
2124  014676  052123  041440  047117
2125  014704  042526  051522  047511
2126  014712  020116  050505  040525
2127  014720  020114  047524  052040
```

```
2128  014726  042510  051440  041505
2129  014734  047117  022504  000100
2130  014742  051461  020124  047503  MES24:  .ASCIZ  '1ST CONV. NOT = 2ND%@'
2131  014750  053116  020056  047516
2132  014756  020124  020075  047062
2133  014764  022504  000100
2134                                          .EVEN
2135                                  ;ADDRESS AND CONSTANTS TABLE
2136  014770  000000                  ADWRD2: 0                       ;LOW BYTE OF 'ADWORD'
2137  014772  000000                  PRINT1: 0
2138  014774  001000                  STACK:  1000                    ;INITIAL SP. ADDRESS
2139  014776  001540                  AVECTR: INITA                   ;'↑A' VECTOR ADDRESS
2140  015000  000000                  PROC:   0                       ;TEMP STORAGE FOR 'PSW'
2141  015002  000000                  CHRCNT: 0                       ;TEMP STORAGE
2142  015004  000000                  COUNT:  0                       ;TEMP STORAGE
2143  015006  000000                  KSTOR1: 0                       ;PERMANENT STORAGE
2144  015010  000000                  KSTOR2: 0                       ;PERMANENT STORAGE
2145  015012  000000                  KSTOR3: 0                       ;PERMANENT STORAGE
2146  015014  000000                  KSTOR4: 0                       ;PERMANENT STORAGE
2147  015016  000000                  KSTOR5: 0
2148  015020  000000                  KSTR11: 0
2149  015022  000000                  KSTR12: 0
2150  015024  000000                  STCHAN: 0
2151  015026  000000                  FIRST:  0
2152  015030  000000                  DELAY:  0
2153  015032  000000                  DELAY1: 0
2154  015034  000000                  USEDSH: 0
2155  015036  000000                  USEDMA: 0
2156  015040  000000                  USECLK: 0
2157  015042  000000                  OPS1:   0
2158  015044  000000                  TEMP:   0
2159  015046  000000                  TEMP1:  0                       ;TEMPORARY STORAGE
2160  015050  000000                  TEMP2:  0                       ;TEMPORARY STORAGE
2161  015052  000000                  TEMP3:  0                       ;TEMPORARY STORAGE
2162  015054  000000                  BRLEV1: 0
2163  015056  000000                  BRLEV2: 0
2164  015060  000000                  BRLEV3: 0
2165  015062  000000                  MESPRT: 0
2166  015064  000000                  HIORDV: 0
2167  015066  000000                  HIGH:   0
2168  015070  000000                  LOW:    0
2169  015072  000000                  $TMDAT: 0
2170  015074  000000                  $BDDAT: 0
2171  015076  000000                  $GDDAT: 0
2172  015100  000000                  $ZERO:  0
2173  015102  000000                  AVERM5: 0
2174  015104  000000                  AVERM4: 0
2175  015106  000000                  AVERM3: 0
2176  015110  000000                  AVERM2: 0
2177  015112  000000                  AVERM1: 0
2178  015114  000000                  AVRAGE: 0
2179  015116  000000                  AVERP1: 0
2180  015120  000000                  AVERP2: 0
2181  015122  000000                  AVERP3: 0
```

```
2182  015124  000000          AVERP4: 0
2183  015126  000000          AVERP5: 0
2184  015130  000000          ORLOW:  0
2185  015132  000000          MINUS5: 0
2186  015134  000000          MINUS4: 0
2187  015136  000000          MINUS3: 0
2188  015140  000000          MINUS2: 0
2189  015142  000000          MINUS1: 0
2190  015144  000000          AVGCNT: 0
2191  015146  000000          PLUS1:  0
2192  015150  000000          PLUS2:  0
2193  015152  000000          PLUS3:  0
2194  015154  000000          PLUS4:  0
2195  015156  000000          PLUS5:  0
2196  015160  000000          ORHIGH: 0
2197  015162  000000          XSPRD1: 0
2198  015164  000000          XSPRD2: 0
2199  015166  000000          XSPRD3: 0
2200  015170  000000          XSPRD4: 0
2201  015172  000000          DIAA:   0
2202  015174  000000          DIAB:   0
2203  015176  000000          DIAC:   0
2204  015200  015202          DIAD:   .+2
2205  015202  023420                  10000.
2206  015204  001750                  1000.
2207  015206  000144                  100.
2208  015210  000012                  10.
2209  015212  000001                  1.
2210
2211  015214  000000          ADBUFF: 0
2212
2213          015256                  .=.+40
2214  015256  000000          ADTB0:  0
2215          015320                  .=.+40
2216  015320  000000          ADTB1:  0
2217          015362                  .=.+40
2218  015362  000000          ADTB2:  0
2219          015424                  .=.+40
2220  015424  000000          ADTB3:  0
2221          015466                  .=.+40
2222  015466  000000          ADTB4:  0
2223          015530                  .=.+40
2224  015530  000000          ADTB5:  0
2225          015572                  .=.+40
2226  015572  000000          ADTB6:  0
2227          015634                  .=.+40
2228  015634  000000          ADTB7:  0
2229          015676                  .=.+40
2230  015676  001000          .BLKW   512.
2231
2232                                  ;*
2233                                  ;*THIS SUB CODE IS USED TO INITIALIZE THE LPA-11
2234                                  ;*FIRST WE WILL LOAD MICROCODE INTO KMC-11
2235                                  ;*NEXT WE WILL INIT BOTH UPROCESSORS
```

```
                                        ;*THEN WE WILL LOAD DEVICE TABLE IN SLAVE UP.
                                        ;*THE ORDER OF LOAD IS DETERMINED BY THE USER.
                                        ;*
                                        ;*      CALL=   JSR     R5,$LPAI
                                        ;*              .WORD   0               ;ADDR. OF DEVICE ADDRESS.
                                        ;* ROUTINES REQUIRED:   .LOADLP
                                        ;* PROGRAMS REQUIRED:   DRLPX2
                                        ;*
                                        ;*              ;RETURNS WITH $AERR=1 IF SLAVE
                                        ;*              ;MICRO SAYS AN ADDR. DOES NOT EXSIST. IN THE LIST.
                                        ;*
017676                          $LPAI:
017676  013746  000004                  MOV     4,-(SP)

017702  000413                          BR      31$             ;FIELD DOES NOT HAVE A BUS SWITCH TO
                                                                ;WORRY ABOUT,SO WE WILL UNCONDITIONALLY
                                                                ;BRANCH ARROUD THE NEXT CODE THAT
                                                                ;WORKS BASED ON A BUS SWITCH.
                                                                ;CODE LEFT IN HERE FOR IN HOUSE
                                                                ;PERSONAL WHO MAY PATCH THIS BRANCH
                                                                ;INSTRUCTION TO A <NOP> OCTAL <240>
                                                                ;IN ORDER TO rUN PROGRAM WITH A SWITCH.

                                                                ;NOTE THIS "SWITCH" IS A PIECE OF INHOUSE
                                                                ;TEST EQUIPMENT ONLY IT CONNECTS
                                                                ;THE UNIBUS TO THE I/O BUS FOR
                                                                ;CERTAIN TESTING.
017704  012737  017730  000004            MOV     #30$,4
017712  005237  170000                    INC     170000
017716  104000                          PRINT
017720  017724                          64$
017722  000401                          BR      65$
017724                          64$:    ;ASCII  <7>##
017726                          65$:
017726  000401                          BR      31$
017730  022626                  30$:    CMP     (SP)+,(SP)+
017732  012637  000004            31$:    MOV     (SP)+,4         ;ALL THIS JUNK MUST BE REMOVED!!
017736  005037  020554                  CLR     $AERR
017742  004537  020556                  JSR     R5,$LOAD        ;LOAD MICRO-CODE.
017746  000000G                         .WORD   DRLPX2          ;FILE "DRLPX2.OBJ"

017750  052777  040000  161062          BIS     #BIT14,@KMADO   ;ISSUE KMC+DMC INIT.

017756                          1$:
                                                                ;"HANGS" HERE THEN KMC-11 ERROR.
017756  010146                          MOV     R1,-(SP)
017760  005001                          CLR     R1
017762  005201                  2$:     INC     R1              ;STALL FOR DMC-UP
017764  001376                          BNE     2$
017766  012777  104000  161044          MOV     #BIT15!BIT11,@KMADO     ;SET RUN, AND ENABLE ARBITRATION.
017774  105201                  25$:    INCB    R1
017776  001376                          BNE     25$
```

```
 2290  020000  032777  000040  161032          BIT     #BIT5,@KMAD0    ;SLAVE READY? (READING IPBM SR)
 2291  020006  001401                          BEQ     3$
 2292                                                                  ;FATAL LPA-11 ERROR SLAVE NOT READY.
 2293  020010  104400                          ERROR
 2294
 2295  020012  012777  000004  161024  3$:     MOV     #4,@KMAD2       ;READ FAST PATH
 2296  020020                          4$:
 2297  020020  004537  021466                  JSR     R5, $TOuT       ;-TOUT-CHECK FOR TIMEOUT
 2298
 2299  020024  104400                          ERROR                   ;/TIME-OUT ERROR
 2300                                                                  ;/WE FAILED TO COMPLETE
 2301                                                                  ;/CURRENT OPERATION.
 2302                                                                  ;/CONTINUES IN THIS LOOP
 2303                                                                  ;/WOULD MAKE US "HANG" HERE
 2304
 2305  020026  000774                          BR              4$
 2306
 2307                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
 2308  020030  122777  000377  161006          CMPB    #377,@KMAD2     ;WAIT TILL KMC DONe COMMAND.
 2309  020036  001370                          BNE     4$
 2310  020040  122777  000377  161002          CMPB    #377,@KMAD4     ;IF FAST PATH=377 THEN ERROR.
 2311  020046  001001                          BNE     35$
 2312  020050  104400                          ERROR                   ;IPBM ERROR (SLAVE SIDE)
 2313                                                                  ;YOU MUST RUN IPBM DIAGNOSTIC.
 2314
 2315  020052  122777  000004  160770  35$:     CMPB    #4,@KMAD4       ;IS THIS THE CORRECT VERSION OF MICRO-CODE?
 2316  020060  001543                          BEQ     5$              ;YES-CONTINUE.
 2317  020062  005227  177777                  INC     #-1
 2318  020066  001140                          BNE     5$
 2319  020070  005227  177777                  INC     #-1
 2320  020074  001135                          BNE     5$
 2321  020076  104000                          PRINT
 2322  020100  020104                          66$
 2323  020102  000440                          BR      67$
 2324  020104                          66$:     ;ASCII  <200>"W A R N I N G  THIS PROGRAM WAS DESIGNED TO rUN WITH VERSION 4"
 2325  020204                          67$:
 2326  020204  104000                          PRINT
 2327  020206  020212                          68$
 2328  020210  000430                          BR      69$
 2329  020212                          68$:     ;ASCII  <200>"MICRO-CODE. ANOTHER VERSION CODE WAS DETECTED."
 2330  020272                          69$:
 2331  020272  104000                          PRINT
 2332  020274  020300                          70$
 2333  020276  000434                          BR      71$
 2334  020300                          70$:     ;ASCII  <200>"THIS MAY OR MAYNOT CAUSE FALSE ERROR TO BE REPORTED."<200><200>
 2335  020370                          71$:
 2336
 2337  020370  112737  177777  020522  5$:      MOVB    #0-1,11$        ;DAC CODE FOR SLAVE.
 2338  020376  012501                          MOV     (5)+,R1         ;GET NEXT DEVICE ADDR.
 2339  020400  021127  000000          6$:      CMP     (R1),#0         ;TERM REACHED?
 2340  020404  001444                          BEQ     10$
 2341  020406  105237  020522                  INCB    11$
 2342  020412  113777  020522  160430          MOVB    11$,@KMAD4      ;FIFO DATA
 2343  020420  004737  020524                  JSR     PC,20$          ;ISSUE SEND
```

```
2344  020424  112177  160420          MOVB    (R1)+,@KMAD4    ;SEND LOW BYTE OF DEVICE ADDR TO sLAVE.
2345  020430  004737  020524          JSR     PC,20$          ;ISSUE SEND
2346  020434  112177  160410          MOVB    (R1)+,@KMAD4    ;SEND HIGH BYTE OF DEVICE ADDR. TO SLAVE.
2347  020440  004737  020524          JSR     PC,20$
2348
2349  020444  032777  000002  160366  7$:     BIT     #BIT1,@KMAD0    ;WAIT FOR FIFO DATA
2350  020452  001374                  BNE     7$              ;=1 NO DATA. =0 DATA.
2351  020454  112777  000002  160362  MOVB    #2,@KMAD2       ;READ FIFO.
2352
2353  020462                  8$:
2354  020462  004537  021466          JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
2355
2356  020466  104400                  ERROR                   ;/TIME-OUT ERROR
2357                                                          ;/WE FAILED TO COMPLETE
2358                                                          ;/CURRENT OPERATION.
2359                                                          ;/CONTINUES IN THIS LOOP
2360                                                          ;/WOULD MAKE US "HANG" HERE
2361
2362  020470  000774                  BR              8$
2363
2364                                                          ;/RETURNS HERE-FROM-TIMED OUT.
2365  020472  122777  000377  160344  CMPB    #377,@KMAD2     ;WAIT FOR READ.
2366  020500  001370                  BNE     8$
2367  020502  105777  160342          TSTB    @KMAD4          ;WAS A ZERO RETURNED?
2368  020506  001734                  BEQ     6$              ;YES GET NEXT ADDR.
2369                                                          ;SLAVE WILL RETURN CODE 0 IF
2370  020510  005237  020554          INC     $AERR           ;DEV PRESENT,   ELSE
2371                                                          ;EXIT $AERR=1 IF SLAVE GIVES ERROR.
2372  020514  005041                  CLR     -(1)            ;GET RID OF REFERENCE TO BAD ADDR.
2373  020516  012601          10$:    MOV     (SP)+,R1
2374  020520  000205                  RTS     R5              ;RETURN ALL ADDR. CHECKED.
2375
2376  020522  000000          11$:    .WORD   0               ;HOLDS DAC CODE PLUS OFFSET
2377                                                          ;TO SLAVES ADDR. TABLE.
2378
2379  020524  112777  000003  160312  20$:    MOVB    #3,@KMAD2       ;ISSUE FIFO WRITE
2380  020532                  21$:
2381  020532  004537  021466          JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
2382
2383  020536  104400                  ERROR                   ;/TIME-OUT ERROR
2384                                                          ;/WE FAILED TO cOMPLETE
2385                                                          ;/CURRENT OPERATION.
2386                                                          ;/CONTINUES IN THIS LOOP
2387                                                          ;/WOULD MAKE US "HANG" HERE
2388
2389  020540  000774                  BR              21$
2390
2391                                                          ;/RETURNS HERE-FROM-TIMED OUT.
2392  020542  122777  000377  160274  CMPB    #377,@KMAD2     ;KMC CODE WILL RETURN A "377"
2393  020550  001370                  BNE     21$             ;WHEN DONE COMMAND.
2394  020552  000207                  RTS     PC
2395
2396  020554  000000          $AERR:  .WORD   0               ;=0 IF ADDR. LIST OK,=1 IF BAD.
2397
```

```
                                        ;*
                                        ;*THIS SUB CODE USED TO LOAD MICRO-CODE INTO LPA-11.
                                        ;*      CALL =  JSR     R5,$LOAD
                                        ;*              .WORD   XX              ;ADDR. OF MICRO CODE.
                                        ;*                      ;RETURNS HERE
                                        ;*      NOTE:   MICRO CODE FILE MUST END IN -1 DATA.
                                        ;*

020556  010446          $LOAD:  MOV     R4,-(SP)        ;SAVE R4.
020560  010046                  MOV     R0,-(SP)        ;SAVE R0.
020562  012500          1$:     MOV     (5)+,R0         ;GET PROG. ADDR.
020564  005077  160250          CLR     @KMAD0          ;CLEAR CSR
020570  005077  160254          CLR     @KMAD4          ;CLEAR CRAM ADDR.
020574  052777  002000  160236  2$:     BIS     #2000,@KMAD0    ;SELECT CRAM.
020602  012077  160246          MOV     (0)+,@KMAD6     ;WRITE DATA.
020606  052777  020000  160224          BIS     #20000,@KMAD0   ;SET CRAM WRITE
020614  005077  160220          CLR     @KMAD0          ;DISABLE CRAM.
020620  005277  160224          INC     @KMAD4          ;UPDATE CRAM ADDR.
020624  021027  177777          CMP     (0),#-1         ;ALL DONE?
020630  001361                  BNE     2$              ;NO LOOP.
020632  005077  160212          CLR     @KMAD4          ;CLEAR CRAM ADDR.
020636  016500  177776          MOV     -2(5),R0        ;GET MICRO CODE ADDR.

020642  052777  002000  160170  3$:     BIS     #2000,@KMAD0    ;SELECT CRAM
020650  022077  160200          CMP     (R0)+,@KMAD6    ;DATA OK?
020654  001013                  BNE     5$              ;NO - REPORT AN ERROR.
020656  021027  177777          CMP     (0),#-1         ;ALL DONe?
020662  001405                  BEQ     4$              ;YES - EXIT
020664  005077  160150          CLR     @KMAD0          ;NO - DESELECT CRAM.
020670  005277  160154          INC     @KMAD4          ;UPDATE CRAM ADDR.
020674  000762                  BR      3$

020676  012600          4$:     MOV     (SP)+,R0        ;RESTORE R0
020700  012604                  MOV     (SP)+,R4        ;RESTORE R4
020702  000205                  RTS     R5              ;EXIT

020704                  5$:                             ;COME HERE ON LOAD ERROR
020704  005745                  TST     -(5)
020706  105204                  INCB    R4              ;UPDATE ERROR COUNTER.
020710  100324                  BPL     1$              ;IF NOT TOO MANY, TRY AGAIN.
020712  000000                  HALT                    ;MICRO CODE LOAD ERROR.
                                                        ;KMC-11 FAULT. YOU COULD TRY
020714  000722                  BR      1$              ;TO PRESS CONTINUE TO GIVE IT
                                                        ;ANOTHER CHANCE, BUT I DOUBT
                                                        ;THAT THAT WOULD WORK. SINCE I'VE
                                                        ;ALREADY GIVEN IT 177 (OCTAL) CHANCES.
                                                        ;TRY RUNNING THE KMC-11 DIAGNOSTIC.


                                        ;*THIS ROUTINE ISSUES A WRITE COMMAND TO THE LPA-11
                                        ;*
                                        ;*      CALL =  JSR     R5,$TLKW
                                        ;*              .WORD   0               ;OFFSET OF DEVICE ADDR.
```

```
2452                                                    ;*              .WORD   0               ;DATA TO BE WRITTEN
2453                                                    ;*
2454  020716  010046                          $TLKW:    MOV     R0,-(SP)        ;SAVE R0
2455  020720  012500                                    MOV     (5)+,R0         ;GET DEVICE OFFSET
2456  020722  052700  000340                           BIS     #340,R0         ;ADD WRITE CODE.
2457  020726  004737  021200                           JSR     PC,$LPW         ;WAIT FOR FAST PATH READY
2458  020732  010037  021024                           MOV     R0,W1
2459  020736  010077  160106                           MOV     R0,@KMAD4
2460  020742  112777  000005  160074                   MOVB    #5,@KMAD2       ;ISSUE FAST PATH WRITE
2461  020750  004737  021200                           JSR     PC,$LPW         ;WAIT FOR RDY
2462  020754  011537  021026                           MOV     (5),W2
2463  020760  112577  160064                           MOVB    (5)+,@KMAD4     ;WRITE LOW BYTE DATA.
2464
2465  020764  112777  000005  160052                   MOVB    #5,@KMAD2       ;FP WRITE
2466  020772  004737  021200                           JSR     PC,$LPW
2467  020776  111537  021030                           MOVB    (5),W3
2468  021002  112577  160042                           MOVB    (5)+,@KMAD4     ;WRITE HIGH BYTE
2469  021006  112777  000005  160030                   MOVB    #5,@KMAD2
2470  021014  004737  021200                           JSR     PC,$LPw
2471  021020  012600                                    MOV     (SP)+,R0
2472  021022  000205                                    RTS     R5              ;EXIT DONE.
2473  021024  000000                          W1:       0
2474  021026  000000                          W2:       0
2475  021030  000000                          W3:       0
2476
2477                                                    ;*
2478                                                    ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
2479                                                    ;*
2480                                                    ;*      CALL =  JSR     R5,$TLKR
2481                                                    ;*              .WORD   0               ;OFFSET OF DEViCE
2482                                                    ;*              ;RETURNS HERE
2483                                                    ;*DATA IN WORD $DATR
2484                                                    ;*
2485
2486  021032  010046                          $TLKR:    MOV     R0,-(SP)        ;SAVE R0
2487  021034  012500                                    MOV     (5)+,R0         ;GET OFFSET
2488  021036  052700  000300                           BIS     #300,R0         ;ADD READ CODE
2489  021042  004737  021200                           JSR     PC,$LPW         ;WAIT TILL READY
2490  021046  110077  157776                           MOVB    R0,@KMAD4
2491  021052  112777  000005  157764                   MOVB    #5,@KMAD2       ;ISSUE WRITE FP
2492  021060  004737  021200                           JSR     PC,$LPW
2493  021064  010037  021174                           MOV     R0,RD1
2494  021070                                  1$:
2495  021070  004537  021466                           JSR     R5, $TOUt       ;-TOUT-CHECK FOR TIMEOUT
2496
2497  021074  104400                                    ERROR                   ;/TIME-OUT ERROR
2498                                                                            ;/WE FAILED TO COMPLETE
2499                                                                            ;/CURRENT OPERATION.
2500                                                                            ;/CONTINUES IN THIS LOOP
2501                                                                            ;/WOULD MAKE US "HANG" HERE
2502
2503  021076  000774                                    BR              1$
2504
2505                                                                            ;/RETURNS HERE-FROM-TIMED OUT.
```

```
2506  021100  032777  000040  157732          BIT     #BIT5,@KMADO    ;FAST PATH GOT DATA?
2507  021106  001370                          BNE     1$
2508  021110  112777  000004  157726          MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
2509  021116  004737  021200                  JSR     PC,$LPW
2510  021122  117737  157722  021176          MOVB    @KMAD4,$DATR    ;GET LOW BYTE
2511  021130                          2$:
2512  021130  004537  021466                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
2513
2514  021134  104400                          ERROR                   ;/TIME-OUT ERROR
2515                                                                  ;/WE FAILED TO COMPLETE
2516                                                                  ;/CURRENT OPERATION.
2517                                                                  ;/CONTINUES IN THIS LOOP
2518                                                                  ;/WOULD MAKE US "HANG" HERE
2519
2520  021136  000774                          BR              2$
2521
2522                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
2523  021140  032777  000040  157672          BIT     #BIT5,@KMADO    ;FAST PATH READY?
2524  021146  001370                          BNE     2$
2525  021150  112777  000004  157666          MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
2526  021156  004737  021200                  JSR     PC,$LPW
2527  021162  117737  157662  021177          MOVB    @KMAD4,$DATR+1  ;SAVE HIGH BYTE
2528  021170  012600                          MOV     (SP)+,R0
2529  021172  000205                          RTS     R5
2530  021174  000000                  RD1:    0
2531  021176  000000                  $DATR:  .WORD   0
2532                                          ;
2533                                          ;THIS ROUTINE WAITS FOR KMC-CODE TO BECOME READY AS WELL
2534                                          ;AS FAST PATH TO BE READ.
2535                                          ;
2536                                          ;       CALL =  JSR     PC,$LPw
2537                                          ;
2538                                          ;IT WILL TIME OUT IF TOO MUCH TIME IS TAKEN BY
2539                                          ;THE MICRO-PROCESSORS AND REPORT AN ERROR, THEN HALT.
2540                                          ;
2541
2542  021200  010146                  $LPW:   MOV     R1,-(SP)        ;SAVE R1
2543  021202  005001                          CLR     R1
2544  021204  122777  000377  157632  1$:     CMPB    #377,@KMAD2     ;FINISHED INSTRUCTION?
2545  021212  001403                          BEQ     2$
2546  021214  005201                          INC     R1              ;TIME OUT?
2547  021216  001372                          BNE     1$
2548  021220  000411                          BR      10$
2549
2550  021222  032777  000020  157610  2$:     BIT     #BIT4,@KMADO    ;FAST PATH READ?
2551  021230  001403                          BEQ     3$
2552  021232  005201                          INC     R1              ;NO - TIME OUT?
2553  021234  001372                          BNE     2$
2554  021236  000402                          BR      10$             ;YES - REPORT AN ERROR
2555
2556  021240  012601                  3$:     MOV     (SP)+,R1        ;RESTORE R1
2557  021242  000207                          RTS     PC              ;EXIT
2558
2559  021244                          10$:
```

```
2560  021244  104000                    PRINT
2561  021246  021252                    64$
2562  021250  000407                    BR      65$
2563  021252                    64$:    ;ASCII  <200>#LPA-11 FAULT#
2564  021270                    65$:
2565
2566  021270  000000            11$:    HALT                    ;LPA-11 FAULT RUN LPA-11
2567  021272  000776                    BR      11$             ;DIAGNOSTICS.
2568
2569
2570
2571                                    ;*
2572                                    ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE TO
2573                                    ;*A DEVICE ADDRESS ON THE I/O BUSS FOR WRITE ONLY.
2574                                    ;*
2575                                    ;* FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN USED
2576                                    ;* BEFORE. IF NOT WE HAVE TO INITIALIZE THE LPA WITH
2577                                    ;* THAT ADDRESS.
2578                                    ;* WHEN THE ADDR. IS KNOWN BY THE LPA, DO THE OUTPUT BY
2579                                    ;* $TLKW
2580                                    ;*
2581
2582  021274  010046            $OUTLP: MOV     R0,-(SP)        ;SAVE R0
2583  021276  010146                    MOV     R1,-(SP)        ;SAVE R1
2584
2585  021300  012700  001066            MOV     #.DVLS,R0       ;PROGRAM DEFINED LIST.
2586  021304  005001                    CLR     R1
2587  021306  005710            1$:     TST     (0)             ;TERMINATOR REACHED?
2588  021310  001421                    BEQ     10$             ;YES NEXT STEP.
2589  021312  027520  000000            CMP     @(5),(0)+       ;MATCH WITH ADDR IN LIST?
2590  021316  001402                    BEQ     2$
2591  021320  005201                    INC     R1
2592  021322  000771                    BR      1$
2593
2594  021324  010137  021342    2$:     MOV     R1,3$           ;SAVE OFFSET, DEVICE KNOWN.
2595  021330  005725                    TST     (5)+
2596  021332  013537  021344            MOV     @(5)+,4$        ;GET DATA TO BE WRITTEN
2597  021336  004537  020716            JSR     R5,$TLKW        ;DO WRITE
2598  021342  000000            3$:     .WORD   0               ;DEVICE OFFSET
2599  021344  000000            4$:     .WORD   0               ;DATA TO BE WRITTEN.
2600  021346  012601                    MOV     (SP)+,R1
2601  021350  012600                    MOV     (SP)+,R0
2602  021352  000205                    RTS     R5
2603  021354  017520  000000    10$:    MOV     @(5),(0)+       ;SAVE ADDR.
2604  021360  005010                    CLR     (0)
2605  021362  004537  017676            JSR     R5,$LPAI
2606  021366  001066                    .WORD   .DVLS
2607  021370  000755                    BR      2$
2608
2609                                    ;*
2610                                    ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE
2611                                    ;*TO A DEVICE ADDR. ON THE I/O BUSS FOR READ ONLY.
2612                                    ;*
2613                                    ;*FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN
```

```
                                        ;*USED BEFORE. IF NOT, WE HAVE TO INITIALIZE THE LPA
                                        ;*WITH THE NEW ADDR.
                                        ;*WHEN THE ADDR IS KNOWN WE CAN DO OUTPUT THROUGH
                                        ;*$TLKr
                                        ;*      CALL THROUGH    MOVEI   DATA,ADDR.
                                        ;*              WHICH EQUALS:
                                        ;*                      JSR     R5,$INLP
                                        ;*                      .WORD   XX      ADDR OF DEVICE
                                        ;*                      .WORD   YY      ADDR TO STORE READ DATA.

021372  010046          $INLP:  MOV     R0,-(SP)        ;SAVE R0
021374  010146                  MOV     R1,-(SP)        ;SAVE R1

021376  012700  001066          MOV     #.DVLs,R0       ;PROG DEFINED ADDR. LIST.
021402  005001                  CLR     R1
021404  005710          1$:     TST     (0)             ;EOL REACHED?
021406  001420                  BEQ     10$             ;YES - DEFINE NEW ADDR.

021410  027520  000000          CMP     @(5),(0)+       ;ADDR. MATCH?
021414  001402                  BEQ     2$
021416  005201                  INC     R1
021420  000771                  BR      1$

021422  010137  021434  2$:     MOV     R1,3$           ;SAVE LIST OFFSET
021426  005725                  TST     (5)+
021430  004537  021032          JSR     R5,$TLKR        ;GO READ DEVICE
                021434  $OFS=.
021434  000000          3$:     .WORD   0               ;OFFSET OF DEVICE

021436  013735  021176          MOV     $DATR,@(5)+     ;STORE DATA.
021442  012601                  MOV     (SP)+,R1        ;RESTORE R1
021444  012600                  MOV     (SP)+,R0        ;RESTORE R2
021446  000205                  RTS     R5              ;EXIT

021450  017520  000000  10$:     MOV     @(5),(0)+
021454  005010                  CLR     (0)
021456  004537  017676          JSR     R5,$LPAI
021462  001066                  .WORD   .DVLS
021464  000756                  BR      2$
                                ;*
                                ;*$TOUt ROUTINE USED TO WATCH IF
                                ;*      WE'RE IN A LOOP TOO-LONG
                                ;*      CALL=   JSR R5, $TOUT
                                ;*              ERROR X     ;RETURNS HERE ON TIMEOUT
                                ;*              BR
                                ;*              ;RETURNS HERE NO ERROR
                                ;*

021466  020537  021522  $TOuT:  CMP     R5,$SAD         ;SAME ADDR?
021472  001405                  BEQ     1$
021474  010537  021522          MOV     R5,$SAD         ;NO-SAVE THIS ADDR.
021500  005037  021524          CLR     $CNT            ;CLR CNT AT ADDR.
021504  000403                  BR      2$
021506  005237  021524  1$:     INC     $CNT            ;OVERFLOW?
```

```
2668  021512  100402                    BMI     3$              ;YES-ERROR RETURN
2669  021514  062705  000004    2$:     ADD     #4,R5           ;NO-NON ERROR RETURN
2670  021520  000205              3$:     RTS     R5              ;RETURN.
2671
2672  021522  000000              $SAD:   .WORD   0               ;CONTAINS LOOP ADDR.
2673  021524  000000              $CNT:   .WORD   0               ;# OF TIMES AT ADDR.
2674
2675                                        ;*
2676                                        ;* THIS ROUTINE REPLACES WHAT THE USER WOULD ORDINARILY
2677                                        ;*USE FOR A RESET.  FIRST WE DO A RESET INSTRUCTION.
2678                                        ;*THEN WE CLR ".DVLST" WHICH FORCES US TO RESET BOTH THE
2679                                        ;*KMC AND DMC AS SOON AS A DEVICE IS REFERENCED.
2680                                        ;*
2681                                        ;*      CALL=JSR        PC,$RESET       ;REPLACES "RESET INSTRUCTION
2682                                        ;*              ;RETURNS HERE.
2683                                        ;*
2684  021526  000005              $RESET: RESET                   ;RESET THE WORLD.
2685
2686                              ;*        MOV     @2$,1$  ;/READ DEVICE REG 2$,PUT DATA IN 1$.
2687  021540  005737  020554              TST     $AERR           ;IF NO ERROR,LOOP
2688  021544  001004                      BNE     10$             ;THERE WAS AN ERROR.
2689  021546  062737  000002  021562      ADD     #2,2$           ;UPDATE DEVICE ADDR.
2690                                                              ;YOU SEE ,WE HAVE TO PROTECT OUR SELF!
2691                                                              ;IF 2$ CONTAINED A VALID ADDR,WE
2692                                                              ;MUST KEEP TRYING UNTIL WE GENERATE
2693                                                              ;AN INVALID ADDR.
2694  021554  000764                      BR      $RESET
2695  021556                      10$:
2696  021556  000207                      RTS     PC
2697  021560  000000              1$:     .WORD   0               ;JUNK LOC.
2698  021562  160000              2$:     .WORD   160000          ;DUMB ADDR. FORCES INIT OF DMC/KMC.
2699
2700
2701                                        ;
2702                                        ;SDELAY- ROUTINE TO GIVE A MINOR DELAY.
2703                                        ;        IS NOT TIME DEPENDENT CODE SENCE
2704                                        ;        NOT USED TO GET SPECIFIC TIME BUT
2705                                        ;        JUST A LITTLE DELAY.
2706                                        ;
2707                                        ;        THAT IS UNLESS A REAL TIME CLOCK IS PRESENT!
2708                                        ;        THEN WE'LL GENERATE A TIME BETWEEN 16MS TO 32 MS
2709                                        ;
2710                                        ;
2711                                        ;        CALL=   JSR PC,  SDELAY
2712                                        ;
2713  021564                      SDELAY:
2714  021564  005737  021646              TST     RTCCSR          ;CLOCK PRESENT?
2715  021570  100016                      BPL     10$
2716  021572  012737  000002  021636      MOV     #2,TIME
2717  021600  052777  000115  000040      BIS     #115,@RTCCSR    ;START CLOCK
2718  021606  005037  177776              CLR     PS
2719  021612  005737  021636      1$:     TST     TIME
2720  021616  001375                      BNE     1$
2721  021620  005077  000022              CLR     @RTCCSR         ;STOP CLOCK
```

```
                                        RTS PC
021624  000207                          INCB    TIME
021626  105237  021636          10$:    BNE     10$
021632  001375                          RTS     PC
021634  000207
                                TIME:   .WORD   0
021636  000000
                                CLKINT: DEC     TIME
021640  005337  021636                  RTI
021644  000002
                                RTCCSR: .WORD   0               ;CLOCK CSR IF USED.
021646  000000
                                        ;
                                        ;THIS ROUTINE LOOKS THROUGH CURENT .DVLS FOR A/D ADDR.
                                        ;IF UNFOUND,GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
                                        ;TO SET UP THE USER PROGRAM TO LINK TO FILE "DRLPX2" FOR
                                        ;SAMPLE TAKEING PURPOSES.
                                        ;       TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
                                        ;       A/D CSR IN BSEL 4,AND 5.
                                        ;       (2) HE MUST CALL THIS ROUTINE:
                                        ;               JSR     R5,$PUTS        ;CALL SET UP ROUTINE.
                                        ;               .WORD   ADCSR           ;ADDR. OF A/D CSR.
                                        ;               ;RETURNS HERE ;KMC BSEL 3,6,7 PERMINENTLY SET UP
                                        ;                               ;(UNTILL ONE DOES A RESET)
                                        ;
                                        ;       (3)THE USER MUST PUT CODE 006 INTO KMC REG 2 TO
                                        ;          START CONVERSION  CAUTION*DO WITH MOVB INSTR.!
                                        ;       (4)MONITOR KMC REG 2 FOR CODE 377 (DRLPX2 IS DONE)
                                        ;       (5)READ KMC REG 4,5 FOR A/D RESULT.
                                        ;       (6) TO TAKE MORE SAMPLES,SIMPLY PUT A/D CSR INTO
                                        ;          BSEL 4,5 AND CODE 6 INTO BSEL 2.
                                        ;
021650  012537  021660          $PUTS:  MOV     (5)+,1$                 ;GET ADDR OF ADDR. OF A/D
021654  004537  021372                  JSR     R5,$INLP
021660  000000                  1$:     .WORD   0
021662  021756                          .WORD   10$
021664  113777  021434  157162          MOVB    $OFS,@KMAD6
021672  113777  021434  157156          MOVB    $OFS,@KMAD7
021700  013737  021660  021720          MOV     1$,2$
021706  062737  000002  021720          ADD     #2,2$
021714  004537  021372                  JSR     R5,$INLP
021720  000000                  2$:     .WORD   0
021722  021756                          .WORD   10$
021724  113777  021434  157114          MOVB    $OFS,@KMAD3
021732  152777  000340  157114          BISB    #340,@KMAD6
021740  152777  000300  157110          BISB    #300,@KMAD7
021746  152777  000300  157072          BISB    #300,@KMAD3
021754  000205                          RTS     R5
021756  000000                  10$:    .WORD   0

000001                                  .END
```

```
ADBRL   001004          138#    761*    764     768     770     771
ADBUFF  015214          925     932     1109    1123    1363    1365    1521    1720    1773    2211#
ADCNVT  010726          922     995     1102    1107    1519#
ADCS    001130          187#    228*    362     367     377     379     384     386     396     398     403     405     415
                        417     422     424     435     437     442     444     454     456     462     464     468     470
                        480     482     487     489     499     501     506     508     518     520     525     527     537
                        539     544     546     556     558     563     565     575     577     582     584     594     596
                        601     603     613     615     620     622     632     634     639     641     655     657     668
                        673     686     695     699     704     712     721     726     729     734     743     748     751
                        756     780     783     793     796     806     808     813     816     826     829     1178    1181
                        1184    1190    1193    1195    1198    1201    1230    1233    1235    1238    1241    1245    1248
                        1254    1527    1533    1540    1544    1548    1554    1860
ADCS1   001132          188#    229*    233*
ADDBR   001134          189#    230*    234*    364     650     665     682     723     745     849     1188    1205    1250
                        1258    1568    1572    1862    1949
ADINT   001144          196#    237*    301*
ADINT1  001146          197#    238*    239*    301     302*
ADTB0   015256          1110*   1150    2214#
ADTB1   015320          1111*   1151    2216#
ADTB2   015362          1112*   1152    2218#
ADTB3   015424          1113*   1153    2220#
ADTB4   015466          1114*   1154    2222#
ADTB5   015530          1115*   1155    2224#
ADTB6   015572          1116*   1156    2226#
ADTB7   015634          1117*   1157    2228#
ADTEST  002040          112     316     347#
ADTST0  002054          351#    871
ADTST1  002044          333     349#
ADT1    002160          351     373#    1914
ADT10   003036          514#
ADT11   003130          533#
ADT12   003222          552#
ADT13   003314          571#
ADT14   003406          590#
ADT15   003500          609#
ADT16   003572          628#
ADT2    002252          392#
ADT20   003664          647#
ADT20A  003734          657#    660
ADT20B  003752          658     662#
ADT21   004030          679#
ADT22   004232          718#
ADT23   004334          740#
ADT27   004524          776#
ADT3    002344          411#
ADT30   004570          789#
ADT31   004634          802#
ADT32   004734          822#
ADT4    002436          431#
ADT5    002530          450#
ADT6    002652          476#
ADT7    002744          495#
ADWRD2  014770          898*    899*    903*    907*    910*    992*    993*    994*    1098*   1099*   1100*   1103*   1104*
                        1105*   1528    1530    1537    1541    2136#
```

```
AMASK   001010            140#  1308  1311
ARESET  005522            935#
AVECTR  014776            303*   882*   954*  1081*  1163*  1421  2139#
AVECT1= 000300            179    180    198#   199
AVERM1  015112           1258   1261   1281  2177#
AVERM2  015110           1250   1260   1277  2176#
AVERM3  015106           1205   1208   1280  2175#
AVERM4  015104           1188   1207   1276  2174#
AVERM5  015102           1779   1782  2173#
AVERP1  015116           1752  2179#
AVERP2  015120           2180#
AVERP3  015122           2181#
AVERP4  015124           2182#
AVERP5  015126           1777  2183#
AVGCNT  015144           1053   1799  2190#
AVGDAT  012126           1738#  1741
AVGTAB  010100           1363*  1365*  1366*
AVRAGE  015114            998   1044   1130  1723*  1724  1725  1733*  1739*  1742*  1744  1751  1759  1760
                         2178#
BCDBIN  010564           1479#  1511  1627
BCDBN1  010600           1482#  1506
BCDBN2  010604           1484#  1500
BCDEND  010662           1485   1488  1501#
BCDERR  010704           1490   1492  1502  1508#
BCDTAB  010716            965    967   969   971   976  1090  1091  1480  1481*  1512#
BELL    013236            869   1935#
BINDEC= 104006            127#  1059
BIT0    = 000001           56#   652   673   683   692   803
BIT1    = 000002           55#   364   374   379   823  2349
BIT10   = 002000           46#   534   539   777
BIT11   = 004000           45#   553   558   777   851  2286
BIT12   = 010000           44#   572   577   777
BIT13   = 020000           43#   591   596   777  1210  1263  1291  1554
BIT14   = 040000           42#   610   615  1230  2278
BIT15   = 100000           41#   629   634   810  2286
BIT2    = 000004           54#   393   398   823
BIT3    = 000010           53#   412   417   823
BIT4    = 000020           52#   432   437   726   790  2550
BIT5    = 000040           51#   313   451   456   748   790   888  2290  2506  2523
BIT6    = 000100           50#   355   477   482   790   842   987  1171  1381  1420  1445
BIT7    = 000200           49#
BIT8    = 000400           48#   496   501   777
BIT9    = 001000           47#   515   520   777
BRLEV1  015054           768*   769*  1568  2162#
BRLEV2  015056           770*  2163#
BRLEV3  015060           771*   772*  2164#
CALBRT  005204            319   882#  1339
CALBT1  005216            882   885#
CALBT2  005274            898#   939
CALBT4  005524            931   936#
CALB1A  005222            887#
CALB2   005336            902   905#
CALB2A  005372            904   909   911#   938
CALB2B  005510            927   932#
```

```
CALB2C  005364            906     910#
CALB2D  005444            912     921#
CATORI= 104005            126#    997
CATORZ  012162           1631    1750#
CATR1   012262           1769#   1771
CATR2   012304           1774#   1791    1793    1795
CATR3   012344           1783#   1787
CATR4   012364           1784    1789#
CATR5   012410           1775    1799#
CHRCNT  015002           1332*   1382*   1399*   1400    1427    1429*   1484    1486*   1505    2141#
CK      013054           1852    1890#
CLKINT  021640           2730#
CMPCNT  012160           1126*   1721*   1737    1747#
CMPRA   007724           1302*   1305    1321#
CMPRB   007726           1303*   1304    1322#
CMPRC   007730           1304*   1305*   1308*   1311*   1315*   1318*   1323#
CMPRI   007676           1307    1311#
CMPR2   007710           1306    1309    1312    1315#
CMPR3   007716           1310    1313    1318#
CMPTE   011774           1630    1719#
CMPTEA  012014           1129    1722#
CMPUTE= 104004            125#    996
CNTLED  013344           1941*   1953*   1957#
CNTRLA  013766           1412    2024#
CNTRLC  013762            297    2022#
CNTRLG  013771           1450    2025#
COMPR   007634           1206    1259    1275    1279    1302#
COUNT   015004            897*    991*   1097*   1520    2142#
CPDLAY  011426            268    1640#
CPTIME  011436            268*    688     704    1533    1645#
CPUTYP  001370            245     256#
CRLF    014020            891     930    1024    1031    1056    1213    1266    1294    1438    1868    1873    2031#
CSB     001140            192#    232*    236*    918
CSR     001136            191#    231*    235*    915     921
DECOCT= 104001            122#    964     975    1089
DECOUT  011674           1689    1695#
DECPNT  011762           1682*   1686    1688    1693*   1711#
DECPRT  011576           1632    1680#
DEC1    011722           1696    1698    1701#
DEC2    011736           1700    1703#
DELAY   015030           1101*   1106*   1523    1536*   2152#
DELAY1  015032            688*    689*    690*    691*    695*    704*    705*   1533*   1534*   2153#
DIAA    015172           2201#
DIAB    015174           2202#
DIAC    015176           2203#
DIAD    015200           2204#
DIGCNT  011756           1681*   1690*   1697    1709#
DIGIT   011754           1684*   1685*   1695    1699*   1702*   1705    1708#
DISPLA  001032            150#    248*    352*    933*    998*   1173*
DMK2    001152            200#   1527*   1528
DOT     014022            311    2033#
DRLPX2= ******  G          14#    2276
DSHTST  006772            115     328    1163#
DSHTO   007000           1163    1164#
```

```
DSHT1   007004          1166#
DSHT2   007010          1167#
DSHT3   007014          1168#   1286
DSHT4   007266          1211    1216#
DSHT6   007276          1209    1217    1227#
DSHT6A  007512          1264    1269#
DSHT7   007524          1262    1275#
DSHT7A  007550          1270    1272    1283#   1300
DSHT8   007600          1278    1282    1291#
DSHT8A  007620          1292    1297#
DSHT8B  007630          1289    1298    1300#
EMTOK   011354          1616    1618#
EMTSRV  011334            99    1612#
EMTTAB  011374          1620    1626#
ENDPAS  013354          1936    1962#
ERROR = 104400           120#    369     381     388     400     407     419     426     439     446     458     472     484
                         491     503     510     522     529     541     548     560     567     579     586     598     605
                         617     624     636     643     661     670     675     701     714     736     758     785     798
                         818     831    2293    2299    2312    2356    2383    2497    2514
FILCHR  001014           142#   1671
FILE1   012202          1753#   1756
FILE2   012240          1761#   1764
FILLS   001012           141#   1668
FIRST   015026          1522*   1564*   2151#
GETDAT  012040          1726#   1736
HIGH    015066          1047    1724*   1727    1729*   2167#
HIORDV  015064          1722*   1734*   1738*   2166#
ICOUNT  001036           152#   1904
INBUF   010546           313*    314     317     320     323     326     888*    889     980     984    1328    1380    1383*
                        1384*   1385*   1386*   1387*   1388*   1389*   1457    1472#   1479
INIT    001154           111     207#
INITA   001540           288#    303    2139
INIT2   001576           283     290     298#    331
INIT3   001774           329#   1341
INPUTA  010216          1390#   1391    1405    1428
INPUTB  010250          1398#   1434
KMAD0   001040           162#    216    2278*   2286*   2290    2349    2409*   2411*   2413*   2414*   2421*   2426*   2506
                        2523    2550
KMAD1   001042           165#    217
KMAD2   001044           167#   2295*   2308    2351*   2365    2379*   2392    2460*   2465*   2469*   2491*   2508*   2525*
                        2544
kMAD3   001046           169#   2765*   2768*
KMAD4   001050           171#   2310    2315    2342*   2344*   2346*   2367    2410*   2415*   2418*   2427*   2459*   2463*
                        2468*   2490*   2510    2527
KMAD5   001052           173#
KMAD6   001054           175#   2412*   2422    2758*   2766*
KMAD7   001056           177#    221    2759*   2767*
KSTOR1  015006           965*    966     969     987*    989    1090*   1098    1329*   1331*   1334*   1335*   1336*   1339
                        1342*   1345    2143#
KSTOR2  015010           900*    937     966*    971*   1067    1091*   1103    1140    2144#
KSTOR3  015012           849*    855*    976*   1003    1008    1014    1019    1854*   1855*   1857    1875    2145#
KSTOR4  015014           651*    659*    751*    752*    989*    992    1032    1065    1069*   1860    1878    2146#
KSTOR5  015016          1862    1881    2147#
KSTR11  015020          1863*   1887    2148#
```

```
KSTR12  015022            1862*   1884    2149#
LADTB   006752            1127    1150#
LEDS    013244             304     924    1168    1743    1856    1941#
LEDSA   013262            1944#   1954
LEDSV1  013346            1943*   1944    1949*   1950*   1951*   1958#
LEDSV2  013350            1944*   1945*   1946*   1949    1959#
LEDSV3  013352            1942*   1946    1952*   1960#
LGERR1  013006            1866    1872#
LOGERR  012664             101    1850#
LOGICA  002022             104     337#
LOW     015070            1041    1725*   1730    1732*   2168#
LPADH   001052             172#
LPADL   001050             170#
LPCI    001040             161#
LPCO    001044             166#
LPMR    001042             164#
LPMS1   001054             174#
LPMS2   001056             176#
LPSADD  001000             136#    228     229     230     231     232
LPSO    001046             168#
LPSVCT  001002             137#    237     238
MANCAL  005246             116     893#
MASK    012646            1820*   1826    1831*   1838#
MESPRT  015062             957*   1025    1048    1052*   2165#
MES1    013453            1870    1978#
MES10   014136             886    2053#
MES11   014203            2062#
MES12   014236            1557    2067#
MES13   014256             956    2071#
MES14   014306             962    1087    2076#
MES15   014321            1165    2078#
MES16   014346             973    2083#
MES18   014365             978    2086#
MES19   014444            1028    2094#
MES20   014477            1051    2100#
MES21   014565            1604    2111#
MES22   014605            1215    2114#
MES23   014663            1268    2122#
MES24   014742            1296    2130#
MES4    013515             289    1990#
MES5    014030             348    2037#
MES7    014057             884    2042#
MES8    014106            1085    2046#
MES9    014131            1138    2051#
MINUS1  015142            1801    2189#
MINUS2  015140            1804    2188#
MINUS3  015136            1807    2187#
MINUS4  015134            1810    2186#
MINUS5  015132            1790*   2185#
MONITR  001546             110     292#    1424    1605
NEWIN   010144            1380#
NEWSWR  014007            1454    2028#
NODMA   002036             332*    342#
OCTPRT  012522            1634    1816#
```

```
OPS1     015042            960*   2157#
ORHIGH   015160           1770    1792*   2196#
ORLOW    015130           1057    1768    1794*   2184#
OUTPTA   010266           1402#   1403    1432
OVRHI    012374           1778    1792#
OVRLO    012402           1780    1794#
PASSCT   001034            151#    350*    352     839     870*   1166*   1173    1902    1923
PC     =%000007             86#    227*    292*    335*    337*    349*    780*    793*    813*    826*    865*    869*    922*
                           995*   1102*   1107*   1129*   1575*   1602*   1689*   1937*   2343*   2345*   2347*   2394*   2457*
                          2461*   2466*   2470*   2489*   2492*   2509*   2526*   2557*   2696*   2723*   2726*
PDPDLY   001006            139#    269*    271*    849
PIRQ   = 177772            254#    260
PLUS1    015146           1800    2191#
PLUS2    015150           1803    2192#
PLUS3    015152           1806    2193#
PLUS4    015154           1809    2194#
PLUS5    015156           2195#
PRINT  = 104000            121#    286     288     296     310     329     347     883     885     890     929     955     961
                           972     977    1023    1027    1030    1050    1055    1084    1086    1137    1164    1212    1214
                          1265    1267    1293    1295    1411    1437    1449    1453    1469    1508    1556    1603    1867
                          1869    1872    1935    2266    2321    2326    2331    2560
PRINT1   014772            285*   1865    1871*   2137#
PROC     015000            889*    893*    901     905    1587*   1594    2140#
PRTAVG=  104012            131#    928    1144
PRTOCT=  104010            129#   1037    1040    1043    1046    1141    1364    1451    1874    1877    1880    1883    1886
PS     = 177776            255#    263    2718*
PSW      001016            144#    209*    294*    354*    841*   1175*   1519*   1593*   1650*   1680*   1908*
PWRFAL   011232             93     298    1580#
PWRUP    011266           1588    1593#
QMARK    014025            330    1470    1509    2035#
RD1      021174           2493*   2530#
RECVRY   006356            325    1081#
RECVY1   006400           1081    1086#
RECVY2   006424           1092#   1136    1145
REPTST   005540            322     954#
REPT1    005552            954     957#    970
REPT2    005642            968     972#
REPT2A   005726            983     985     988#    1071
REPT2B   005704            981     984#
REPT3    005740            990#   1070
REPT4    006136           1000    1006    1011    1017    1020    1023#
REPT5    006154           1026    1029#
REPT6    006252           1049    1052#
REPT6A   006300           1058#   1061
REPT7    006322           1002    1007    1012    1018    1022    1063    1065#
REPT7A   006350           1068    1071#
REPT8    006212           1035    1037#
REPT8A   006354           1032*   1033*   1036*   1038    1073#   1140*   1142
RETURN   013160            351*   1344*   1909    1911*   1914#
RTCCSR   021646           2714    2717*   2721*   2732#
RO     =%000000             79#     214     216*    219*    220     225*    258*    261*    264*    267*    268     269     336*
                           650*    665*    682*    723*    729*    730*    745*    850*    853*    932*    933    1109*   1110
                          1111    1112    1113    1114    1115    1116    1117    1122*   1132*   1374    1418*   1444*   1457*
                          1458    1460    1494*   1499    1580    1601*   1773*   1776    2407    2408*   2419*   2422    2430*
```

```
                2454    2455*   2456*   2458    2459    2471*   2486    2487*   2488*   2490    2493    2528*   2582
                2585*   2601*   2624    2627*   2645*
R1      =%000001  80#     215     217*    221     224*    274*    275*    276*    277     1108*   1110*   1111*   1112*
                1113*   1114*   1115*   1116*   1117*   1124*   1127    1131    1328*   1330*   1331    1375    1392*
                1393*   1394    1396    1398    1404    1409    1417*   1422    1425    1430*   1433    1435    1443*
                1447    1456*   1462*   1463*   1464*   1465*   1467    1482*   1495    1496*   1497*   1498*   1499*
                1501    1504    1581    1600*   1781*   1785*   1786    1789*   1790*   2282    2283*   2284*   2287*
                2338*   2339    2344    2346    2373*   2542    2543*   2546*   2552*   2556*   2583    2586*   2591*
                2594    2600*   2625    2628*   2634*   2637    2644*
R2      =%000002  81#     273*    275     277*    278     279     1058*   1123*   1130*   1376    1416*   1442*   1460*
                1461*   1465    1483*   1495*   1582    1599*   1782*   1783
R3      =%000003  82#     1377    1415*   1441*   1480*   1504*   1583    1598*   1752*   1754*   1760*   1762*   1768*
                1769*   1770
R4      =%000004  83#     1057*   1058    1060    1065*   1066*   1067    1127*   1378    1380*   1398*   1414*   1431
                1440*   1479*   1487    1489    1491    1493*   1494    1503    1521*   1584    1597*   1720*   1723
                1726    2406    2431*   2436*
R5      =%000005  84#     304*    362*    364*    367*    377*    379*    384*    386*    396*    398*    403*    405*
                415*    417*    422*    424*    435*    437*    442*    444*    454*    456*    462*    464*    468*
                470*    480*    482*    487*    489*    499*    501*    506*    508*    518*    520*    525*    527*
                537*    539*    544*    546*    556*    558*    563*    565*    575*    577*    582*    584*    594*
                596*    601*    603*    613*    615*    620*    622*    632*    634*    639*    641*    650*    655*
                657*    665*    668*    673*    682*    686*    695*    699*    704*    712*    721*    723*    726*
                729*    734*    743*    745*    748*    751*    756*    780*    783*    793*    796*    806*    808*
                813*    816*    826*    829*    849*    915*    918*    921*    924*    1168*   1178*   1181*   1184*
                1188*   1190*   1193*   1195*   1198*   1201*   1205*   1206*   1230*   1233*   1235*   1238*   1241*
                1245*   1248*   1250*   1254*   1258*   1259*   1275*   1279*   1302    1303    1316*   1319*   1379
                1413*   1439*   1527*   1533*   1540*   1544*   1548*   1554*   1568*   1572*   1585    1596*   1651
                1652*   1656    1658*   1660    1662    1674    1816    1817*   1822*   1823*   1824*   1825    1834*
                1860*   1862*   1949*   2275*   2297*   2354*   2374*   2381*   2432*   2472*   2495*   2512*   2529*
                2597*   2602*   2605*   2639*   2646*   2650*   2662    2664    2669*   2670*   2687*   2755*   2762*
                2769*
SAV5    012650  1651*   1658    1816*   1834    1839#
SCOPE   = 104002  123#     392     411     450     476     495     514     533     552     571     590     609     628
SCOPEB  013132  1899    1907#   1920    1932
SCOPEC  013066  1628    1897#
SCOPEF  013156  1343*   1904    1906*   1910*   1913#   1925
SCOPEG  013144  1901    1903    1905    1910#   1922    1924    1933
SCOPEH  013162  1629    1918#
SCOPEI  013222  1638    1930#
SCOPEJ  013124  1905#   1926
SCOPE0= 104014  133#     864
SCOPE1= 104003  124#     373     431     647     679     718     740     776     789     802     822     836
SDELAY  021564  2713#
SOFTDI  000172  109#
SOFTSW  000170  108#     1452    1467*
SP      =%000006  85#     208*    214*    215*    224     225     246     293*    353*    921*    988*    1092*   1128*
                1172*   1374*   1375*   1376*   1377*   1378*   1379*   1413    1414    1415    1416    1417    1418
                1419    1439    1440    1441    1442    1443    1444    1580*   1581*   1582*   1583*   1584*   1585*
                1586*   1587    1594*   1595    1596    1597    1598    1599    1600    1601    1612*   1613*   1614*
                1615    1618*   1619*   1620*   1621*   1622    1652    1653*   1817    1818*   1854    1907    1911
                2249*   2272    2273    2282*   2373    2406*   2407*   2430    2431    2454*   2471    2486*   2528
                2542*   2556    2582*   2583*   2600    2601    2624*   2625*   2644    2645
SPACE   = 104007  128#     1039    1042    1045    1143    1368    1876    1879    1882    1885
SPACEX  010066  1355*   1357*   1359#   1367*
```

```
SPCHR   010302          1395    1397    1409#
SPCHR1  010344          1410    1422#
SPCHR3  010406          1426    1433#
SPCHR4  010414          1435#
SPCHR5  010536          1401    1448    1469#
STACK   014774           208     293     353     921     988    1092    1172    2138#
STCHAN  015024          1167*   1169    1174    1227   1263*   1284   1287*   2150#
SWR     001030           149#    244     247*    851     898     900     926     937     999    1001    1062    1135    1210
                        1216    1263    1269    1291    1297    1554    1558    1851    1890    1898    1900    1919    1921
                        1931
SW00  = 000001            75#
SW01  = 000002            74#
SW02  = 000004            73#
SW03  = 000010            72#
SW04  = 000020            71#
SW05  = 000040            70#
SW06  = 000100            69#
SW07  = 000200            68#
SW08  = 000400            67#
SW09  = 001000            66#
SW10  = 002000            65#     926     999    1135
SW11  = 004000            64#
SW12  = 010000            63#
SW13  = 020000            62#    1001
SW14  = 040000            61#
SW15  = 100000            60#
TAGA    012100          1731    1733#
TEMP    015044          1863    2158#
TEMp1   015046           843*    844     860*    861    1125*   1520*   1573*   1719*   1735*   1737*   1740*   1750*   1755*
                        1758*   1763*   1772*   1774*   2159#
TEMP2   015050           844*    846     857*   1726*   1727    1729    1730    1732    1733    1751*   1753*   1754    1759*
                        1761*   1762    1776*   1777    1779    1783    2160#
TEMP3   015052           845*    858*    923*   1096*   1369*   2161#
TESTX   007732          1328#
TESt15  005000           836#
TEST16  005150           838     840     852     864#
TIME    021636          2716*   2719    2724*   2728#   2730*
TITLE   013374           287    1969#
TKB     001022           146#   1392
TKS     001020           145#    284#    295*    309*    355*    842*   1171*   1381*   1390    1420*   1445*   1843
TKSFLG  012652          1637    1843#
TPB     001026           148#   1354*   1404*   1662*   1664*   1667*   1671*   1705*   1830*
TPS     001024           147#   1352    1402    1654    1665    1669    1703    1828    1888
TSLO    012062          1728    1730#
TSTA    007742          1330#   1337
TSTCT1  006116          1015    1019#
TSTCT2  006074          1009    1014#
TSTCT3  006052          1004    1008#
TSTCT4  006020          1001#
TSTTKS= 104013           132#    934     936     990    1029    1093    1139    1288    1850    1897    1918    1930
TSTXB   007776          1333    1339#
TST15A  005034           844#    862
TST15B  005050           846#    859
TST15C  005110           853#    854     856
```

```
TTYIN = 104011    130#   312    887    963    974    979   1088   1455   1510   1845
TYPECL  011516   1661   1664#
TYPERA  011460   1654#  1655   1663   1675
TYPER1  011502   1657   1660#
TYPER2  011510   1662#
TYPMES  011440   1626   1650#
TYPT1   011624   1684#  1694
TYPT2   011632   1685#  1687
TYPT3   011664   1691   1693#
UADMR   001142    194#
USECLK  015040    895*   908*   911   2156#
USEDMA  015036    894*   958*   982*  1082*  2155#
USEDSW  015034    896*   959*   986*  1034   1083*  1562   2154#
VECTOR  001060    179#
VECTPS  001062    180#
VERSN   001064    182#
WHAT    002002    308    332#   341
WHATA   002012    334#   868
WHATB   002012    335#
W1      021024   2458*  2473#
W2      021026   2462*  2474#
W3      021030   2467*  2475#
XLOOP   010020   1343#  1344
XPRTAV  010070   1363#  1636
XPTA1   010076   1364#  1370
XSPACE  010036   1352#  1356   1633
XSPRD1  015162   1021   1060   1799*  1800*  1801*  1802   2197#
XSPRD2  015164   1016   1802*  1803*  1804*  1805   2198#
XSPRD3  015166   1010   1805*  1806*  1807*  1808   2199#
XSPRD4  015170   1005   1808*  1809*  1810*  2200#
XTTYIN  010130     96   1374#  1471   1635
ZERO    011760   1683*  1699   1701*  1710#
$AERR   020554    367   2274*  2370*  2396#  2687
$BDDAT  015074    379    386    398    405    417    424    437    444    456    464    470    482    489
                  501    508    520    527    539    546    558    565    577    584    596    603    615
                  622    634    641    650    657    665    668    673    682    686    699    712    723
                  734    745    756    783    796    808    816    829   1184   1201   1241   1254   1548
                 1550   2170#
$CNT    021524   2665*  2667*  2673#
$DATR   021176   2510*  2527*  2531#  2643
$GDDAT  015076   2171#
$INLP   021372    379    386    398    405    417    424    437    444    456    464    470    482    489
                  501    508    520    527    539    546    558    565    577    584    596    603    615
                  622    634    641    650    657    665    668    673    682    699    712    723    726
                  734    745    748    756    783    796    808    816    829   1184   1188   1190   1195
                 1201   1205   1235   1241   1245   1250   1254   1258   1527   1548   1568   1572   1860
                 1862   2624#  2687   2755   2762
$LOAD   020556   2275   2406#
$LPAI   017676   2248#  2605   2650
$LPW    021200   2457   2461   2466   2470   2489   2492   2509   2526   2542#
$OFS  = 021434   2640#  2758   2759   2765
$OUTLP  021274    362    364    367    377    384    396    403    415    422    435    442    454    462
                  468    480    487    499    506    518    525    537    544    556    563    575    582
                  594    601    613    620    632    639    655    686    695    704    721    729    743
```

```
                    751     780     793     806     813     826     849     915     918     921    1178    1181    1193
                   1198    1230    1233    1238    1248    1533    1540    1544    1554    1949    2582#
                   2754#
$PUTS   021650
$RESET  021526      227     292     335     349     780     793     813     826     865    1602    2684#   2694
$SAD    021522     2662    2664*   2672#
$TLKR   021032     2486#   2639
$TLKW   020716     2454#   2597
$TMDAT  015072      364#    367     374#    377     393#    396     412#    415     432#    435     437     451#    454
                    465#    468     477#    480     496#    499     515#    518     534#    537     553#    556     572#
                    575     591#    594     610#    613     629#    632     652#    655     683#    686     692#    695
                    726#    729     748#    751     777#    780     790#    793     803#    806     810#    813     823#
                    826     846#    849     915#    918#    921    1174#   1178#   1181    1190#   1193    1195#   1198
                   1227#   1230#   1233    1235#   1238    1245#   1248    1527    1530#   1533    1537#   1540    1541#
                   1544    1572    2169#
$TOuT   021466     2297    2354    2381    2495    2512    2662#
$VECT1  001150      199#
$ZERO   015100      362     364     384     403     422     442     462     487     506     525     544     563     582
                    601     620     639     704     721     743     915    1554    2172#
.       = 021760     89#     92#     95#     98#    103#    105#    107#    135#    185#    282     307     315     318
                    321     324     327     368     380     387     399     406     418     425     438     445     457
                    471     483     490     502     509     521     528     540     547     559     566     578     585
                    597     604     616     623     635     642     669     674     700     713     731     735     757
                    762     765     784     797     809     817     830     867     892    1340    1353    1475#   1666
                   1704    1711    1821    1829    1844    1889    1891    2204    2213#   2215#   2217#   2219#   2221#
                   2223#   2225#   2227#   2229#   2230#   2640
.DVLs   001066      184#    223*    356*    2585    2606    2627    2651
```

```
MOVEI    20#   377   384   396   403   415   422   435   442   454   462   468   480   487   499
        506   518   525   537   544   556   563   575   582   594   601   613   620   632   639
        648   655   662   666   671   680   697   710   721   724   732   743   746   754   781
        794   806   814   827  1181  1186  1188  1193  1198  1203  1233  1238  1243  1248  1251
       1256  1525  1545  1566  1569  1858  1860  2685
MOVEM    19#   360   362   365   375   382   394   401   413   420   433   440   452   460   466
        478   485   497   504   516   523   535   542   554   561   573   580   592   599   611
        618   630   637   653   684   693   702   719   727   741   749   778   791   804   811
        824   847   913   916   919  1176  1179  1191  1196  1228  1231  1236  1246  1531  1538
       1542  1552  1947
TOUT     187#  2296  2353  2380  2494  2511
TYPTXT   202#  2266  2321  2326  2331  2559
$CAL.     30#
$DMAST    30#
$DMDT     30#
$MMAST    30#
.KMADR    17#   153
.KSIS     22#   210
.LOADL    25#  2398
.LPAIN    22#  2232
.PUTCS    24#  2734
.RESET    22#  2674
.UTK      29#
.$INLP    28#  2609
.$MMAC    18#
.$OUtL    27#  2570
.$TLKW    26#  2447
.$TOUT   187#  2653
```

```
        2566
INC      219   233   271   659   730   752   860   870  1069  1101  1106  1245  1283  1399  1685
        1690  1753  1785  1790  1792  1794  1871  1906  1952  2265  2284  2317  2319  2370  2415
        2427  2546  2552  2591  2634  2667
INCB     857  2287  2341  2436  2724
JMP      110   111   112   115   116   283   308   316   319   322   325   328   333   341   868
         871  1070  1071  1286  1300  1341  1345  1421  1424  1471  1511  1605  1622  1909
JSR      227   292   304   335   337   349   362   364   367   377   379   384   386   396   398
         403   405   415   417   422   424   435   437   442   444   454   456   462   464   468
         470   480   482   487   489   499   501   506   508   518   520   525   527   537   539
         544   546   556   558   563   565   575   577   582   584   594   596   601   603   613
         615   620   622   632   634   639   641   650   655   657   665   668   673   682   686
         695   699   704   712   721   723   726   729   734   743   745   748   751   756   780
         783   793   796   806   808   813   816   826   829   849   865   869   915   918   921
         922   924   995  1102  1107  1129  1168  1178  1181  1184  1188  1190  1193  1195  1198
        1201  1205  1206  1230  1233  1235  1238  1241  1245  1248  1250  1254  1258  1259  1275
        1279  1527  1533  1540  1544  1548  1554  1568  1572  1602  1689  1743  1856  1860  1862
        1949  2275  2297  2343  2345  2347  2354  2381  2457  2461  2466  2470  2489  2492  2495
        2509  2512  2526  2597  2605  2639  2650  2687  2755  2762
MOV      208   209   214   215   216   217   220   224   225   228   229   230   231   232   237
         238   243   247   248   256   257   258   268   269   273   274   275   277   284   293
         294   295   298   299   301   303   309   336   351   352   353   364   374   393   412
         432   451   477   496   515   534   553   572   591   610   629   650   652   665   682
         683   688   692   704   723   745   768   770   771   777   790   803   810   823   844
         845   846   849   850   882   889   897   900   908   915   918   921   923   932   933
         954   965   966   971   976   988   989   991   998  1032  1057  1058  1065  1081  1090
        1091  1092  1094  1096  1097  1108  1109  1110  1111  1112  1113  1114  1115  1116  1117
        1122  1123  1124  1125  1126  1127  1128  1130  1140  1163  1172  1173  1302  1303  1304
        1328  1344  1354  1363  1367  1374  1375  1376  1377  1378  1379  1380  1392  1398  1413
        1414  1415  1416  1417  1418  1430  1439  1440  1441  1442  1443  1444  1457  1460  1467
        1479  1480  1494  1495  1504  1520  1521  1533  1541  1572  1580  1581  1582  1583  1584
        1585  1586  1587  1588  1593  1594  1595  1596  1597  1598  1599  1600  1601  1612  1614
        1621  1651  1652  1658  1664  1667  1668  1681  1682  1683  1684  1699  1701  1705  1719
        1720  1721  1723  1724  1725  1726  1729  1732  1737  1750  1751  1752  1754  1758  1759
        1760  1762  1768  1772  1773  1776  1782  1799  1802  1805  1808  1816  1817  1819  1820
        1831  1834  1853  1854  1863  1864  1908  1911  1941  1943  1944  2249  2264  2273  2282
        2286  2295  2338  2373  2406  2407  2408  2412  2419  2430  2431  2454  2455  2458  2459
        2462  2471  2486  2487  2493  2528  2542  2556  2582  2583  2585  2594  2596  2600  2601
        2603  2624  2625  2627  2637  2643  2644  2645  2648  2664  2716  2754  2760
MOVB     898   992  1098  1103  1174  1227  1404  1527  1530  1537  1662  1671  1825  1830  1946
        2337  2342  2344  2346  2351  2379  2460  2463  2465  2467  2468  2469  2490  2491  2508
        2510  2525  2527  2758  2759  2765
NOP      113   114   338   339   340   707   708   709   935  1054  1170
RESET   2684
ROL     1822  1823  1824
ROR     1739
RTI      256  1358  1371  1446  1507  1659  1692  1745  1811  1835  1848  1853  1893  1912  2731
RTS     1316  1319  1575  1706  1937  1955  2374  2394  2432  2472  2529  2557  2602  2646  2670
        2696  2723  2726  2769
SBC      261   264
SEC      259   262
SUB      769  1305  1311  1613  1686  1855
TRAP     120
TST      244   260   278   281   306   367   386   405   424   444   464   470   489   508   527
```

```
                546   565   584   603   622   641   699   712   783   796   816   829   837   839   866
                911   967  1025  1034  1048  1062  1131  1216  1269  1297  1427  1431  1458  1484  1503
               1505  1523  1550  1558  1562  1615  1695  1865  1890  1902  1923  2435  2587  2595  2629
               2638  2687  2714  2719
TSTB            263   657   668   686   734   756   808  1184  1201  1241  1254  1352  1390  1402  1548
               1654  1665  1669  1674  1703  1843  1988  2367
.ASCII         1963  1969  1978  1980  1982  1984  1967  1990  1997  2001  2005  2010  2014  2022  2024
               2025  2028  2031  2033  2035  2037  2042  2046  2051  2053  2062  2067  2071  2076  2078
               2083  2086  2094  2100  2111  2114  2270  2325  2330  2335  2564
.ASCIZ         2122  2130
.ASECT           14
.BLKW           185  2230
.BYTE          1962  1968
.ENABL           30
.END           2772
.ENDC           182   226  2300  2307  2357  2364  2384  2391  2498  2505  2515  2522  2734
.EVEN          2134  2270  2325  2330  2335  2564
.GLOBL           14
.IF             179   226  2299  2305  2356  2362  2383  2389  2497  2503  2514  2520  2734
.IFF           2299  2305  2356  2362  2383  2389  2497  2503  2514  2520
.LIST            14    30    92   362   364   367   377   379   384   386   396   398   403   405   415
                417   422   424   435   437   442   444   454   456   462   464   468   470   480   482
                487   489   499   501   506   508   518   520   525   527   537   539   544   546   556
                558   563   565   575   577   582   584   594   596   601   603   613   615   620   622
                632   634   639   641   650   655   657   665   668   673   682   686   695   699   704
                712   721   723   726   729   734   743   745   748   751   756   780   783   793   796
                806   808   813   816   826   829   849   915   918   921  1178  1181  1184  1188  1190
               1193  1195  1198  1201  1205  1230  1233  1235  1238  1241  1245  1248  1250  1254  1258
               1527  1533  1540  1544  1548  1554  1568  1572  1860  1862  1949  2270  2325  2330  2335
               2564  2687
.MACRO           17    18    19    20    22    24    25    26    27    28    29    30   187   202
.NLIST           14    30    92   362   364   367   377   379   384   386   396   398   403   405   415
                417   422   424   435   437   442   444   454   456   462   464   468   470   480   482
                487   489   499   501   506   508   518   520   525   527   537   539   544   546   556
                558   563   565   575   577   582   584   594   596   601   603   613   615   620   622
                632   634   639   641   650   655   657   665   668   673   682   686   695   699   704
                712   721   723   726   729   734   743   745   748   751   756   780   783   793   796
                806   808   813   816   826   829   849   915   918   921  1178  1181  1184  1188  1190
               1193  1195  1198  1201  1205  1230  1233  1235  1238  1241  1245  1248  1250  1254  1258
               1527  1533  1540  1544  1548  1554  1568  1572  1860  1862  1949  2270  2325  2330  2335
               2564  2687
.PSECT           14
.REM              1    14    30
.REPT            92
.TITLE           30
.WORD           162   165   167   169   171   173   175   177   179   180   182   184   199   362   364
                367   377   379   384   386   396   398   403   405   415   417   422   424   435   437
                442   444   454   456   462   464   468   470   480   482   487   489   499   501   506
                508   518   520   525   527   537   539   544   546   556   558   563   565   575   577
                582   584   594   596   601   603   613   615   620   622   632   634   639   641   650
                655   657   665   668   673   682   686   695   699   704   712   721   723   726   729
                734   743   745   748   751   756   780   783   793   796   806   808   813   816   826
                829   849   915   918   921  1178  1181  1184  1188  1190  1193  1195  1198  1201  1205
               1230  1233  1235  1238  1241  1245  1248  1250  1254  1258  1527  1533  1540  1544  1548
```

1554 1568 1572 1860 1862 1949 2276 2376 2396 2531 2598 2599 2606 2641 2651
2672 2673 2687 2697 2698 2728 2732 2756 2757 2763 2764 2770

000000

ERRORS DETECTED: 0

```
*DRLPH,DRLPH/SOL/CRF=DRLPA.MAC,DRLPH
RUN-TIME:  7 10 1 SECONDS
CORE USED:  21K
```