NCV-11

DIAGNOSTIC
CZNCCA0

AH-E772A-MC

FICHE 1 OF 1

DEC 1978
digital
MADE IN USA

IDENTIFICATION

PRODUCT CODE: AC-E771A-MC

PRODUCT NAME: CZNCCA0 NCV11 DIAGNOSTIC TES

DATE: AUGUST 1978

MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 ABSTRACT

THE NCV11 DIAGNOSTIC PROGRAM IS A SERIES OF TESTS DESIGNED
TO TEST ALL LOGIC FUNCTIONS AND DATA PATHS ACCESSIBLE ON THE
M8026 AND M8036. THE M7952 DIAGNOSTIC SHOULD HAVE BEEN EXECUTED.
TOTAL PROGRAM CONTROL IS ACCOMPLISHED THRU THE CONSOLE TERMINAL
VIA KEYBOARD TEST SELECTION AND THE PROVISIONS OF SECTION 5.
THE PROGRAM CAN BE EXECUTED ON AN LSI-11 OR PDP-11 COMPUTER.

2.0 REQUIREMENTS

2.1 EQUIPMENT

1. PDP11 FAMILY OR LSI-11 FAMILY COMPUTER
2. I/O TERMINAL (IE: LA36)
3. NCV11 OPTION INCLUDING KWV11 REAL TIME CLOCK (M7952)
4. A017 SELF-TEST CONNECTOR (70-12894) (REQUIRED FOR SECTION 9.2 + 9.3)
5. H3060 JOYSTICK (OPTIONAL)
6. HARDWARE TESTER AND PROM BLASTER (OPTIONAL)

2.2 STORAGE

THE PROGRAM REQUIRES 16K OF MEMORY.
IF MEMORY MANAGEMENT OR ADDITIONAL MEMORY ARE SENSED, THE
PROGRAM WILL USE THOSE HARDWARE OPTIONS.

2.3 PRELIMINARY PROGRAMS

THE KWV11 DIAGNOSTIC (MD-11-DVKWA) SHOULD HAVE BEEN RUN.

3.0 LOADING PROCEDURE

NORMAL PROCEDURE FOR LOADING A BINARY PROGRAM INTO MEMORY
SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

1. MAKE SURE THE NCV11 DEVICE BUS ADDRESS, INTERRUPT VECTOR ON THE
   M8026 AGREE WITH THE DEFAULT VALUES DEFINED IN SECTION 7.1.
2. MAKE SURE THE NCV11 CLOCK BUS ADDRESS ON THE M7952 AGREE
   WITH THE DEFAULT VALUE DEFINED IN SECTION 7.1.
3. MAKE SURE THE NCV11 INTERRUPT PRIORITY LEVEL ON THE M8217
   AGREE WITH THE DEFAULT VALUE DEFINED IN SECTION 7.1.
4. INSURE THAT THE HALT SWITCH IS DISABLED (IF ANY).

LSI11:
5. TYPE THE STARTING ADDRESS OF 200 AND THE CHARACTER G.
6. THE PROGRAM WILL RESPOND BY TYPING THE PROGRAM TITLE.

LSI11:
7. THE PROGRAM WILL REQUEST SWITCH REGISTER VALUE.
8. THE PROGRAM WILL TYPE THE TEST SELECTION MENU FOR THE OPERATOR.
9. THE OPERATOR SELECT'S THE APPROPRIATE TEST BY TYPING THE TEST LETTER AND ''CR''

IF THE PROGRAM IS EXECUTED ON A CPU WITH A SWITCH REGISTER AND THE
OPERATOR SETS ALL SWITCHES TO A 1, THE SOFTWARE S.R. WILL BE USED.
IF THE HARDWARE SWITCH REGISTER IS USED, THE SOFTWARE S.R. HAS NO EFFECT.

4.1 PROGRAM START

SEQ 0004

200	STARTING ADDRESS OF THE PROGRAM
204	RESTART ADDRESS OF THE PROGRAM
210	STARTING ADDRESS FOR THE FACTORY OPTION CHECK-OUT AREA. (INFORMS THE PROGRAM THE TESTER AND BLASTER ARE CONNECTED)

5.0 SOFTWARE SWITCH REGISTER

5.1 LOGIC TEST OPTIONS

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS
SW11=1	004000	INHIBIT ITERATIONS
SW10=1	002000	BELL ON ERROR
SW09=1	001000	LOOP ON ERROR
SW08=1	0004XX	LOOP ON TEST IN SWR <7-0>

5.2 FIELD ADJUSTMENT LOOP

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ''INPUT VOLTAGE OUT OF RANGE'' ERROR
SW13=1	020000	INHIBIT ERROR OR CONVERSION TYPEOUT
SW09=1	001000	INHIBIT CONVERSIONS ON CAMERA #3 GAIN = 1
SW08=1	000400	INHIBIT CONVERSIONS ON CAMERA #2 GAIN - 1
SW07=1	000200	INHIBIT CONVERSIONS ON CAMERA #1 GAIN - 1
SW06=1	000100	INHIBIT CONVERSIONS ON CAMERA #0 GAIN = 1
SW04=1	000020	INHIBIT CONVERSIONS ON JOYSTICK
SW03=1	000010	INHIBIT CONVERSIONS ON CAMERA #3 GAIN = 2
SW02-1	000004	INHIBIT CONVERSIONS ON CAMERA #2 GAIN = 2
SW01=1	000002	INHIBIT CONVERSIONS ON CAMERA #1 GAIN = 2
SW00=1	000001	INHIBIT CONVERSIONS ON CAMERA #0 GAIN - 2

5.3 CONTROL

1. THE SOFTWARE SWITCH REGISTER 'SWREG' (LOC. 176) CAN BE CHANGED BY USING THE ODT FACILITIES.
2. THE SOFTWARE SWITCH REGISTER CAN BE CHANGED UNDER PROGRAM CONTROL BY TYPING THE 'CONTROL & G' KEYS. THIS KEYBOARD OPERATION WILL PRINT THE CURRENT CONTENTS AND ACCEPT NEW OCTAL SWITCH REGISTER DATA TERMINATED WITH A CARRIAGE RETURN.

LSI-11:

3. ONCE THE ODT MODE HAS BEEN ENTERED BECAUSE OF AN ERROR CONDITION WITH BIT15 SET (HALT ON ERROR), STEP #2 ABOVE IS OF NO VALUE, SO RESORT TO STEP #1 TO ALTER THE SOFTWARE SWITCH RFGISTER IF DESIRED BEFORE TYPING 'P' (CONTINUE).
4. IF THE PROGRAM IS PERFORMING RESET INSTRUCTIONS, SEVERAL 'CONTROL & C' OR 'CONTROL & G' COMMANDS MAY BE NECESSARY TO BE ACKNOWLEDGE BY THE PROGRAM.
5. TESTING CAN BE ABORTED BY TYPING THE 'CONTROL & C' KEYS.

6.0 ERROR REPORTING

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE
COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF
THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE
ERROR PC OR FROM THE TEST ITSELF.

7.0 MISCELLANEOUS

7.1 NCV11 BUS ADDRESS MODIFICATION

MODIFY LOCATION '$BASE' (LOC. 1250) IF BASE NCV11 BUS ADDRESS IS NOT 772760.
MODIFY THE LOW NINE BITS OF LOCATION '$VECT1' (LOC. 1244) IF
BASE NCV11 INTERRUPT VECTOR IS NOT 370.
MODIFY THE HIGH THREE BITS OF LOCATION '$VECT1' (LOC. 1244) IF
THE INTERRUPT PRIORITY LEVEL IS NOT LEVEL 6 ON A UNIBUS CPU.
MODIFY LOCATION '$CDW1' (LOC. 1254) IF BASE NCV11 CLOCK BUS ADDRESS IS NOT 770420.

7.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP (REQUIRES 16K OR MORE).
IF RUNNING UNDER XXDP/ACT/APT, THE LOGIC AND DIFLIN TESTS WILL BE RUN.

7.3 POWER FAIL

A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT
WHICH TIME THE PROGRAM IS RESTARTED (ONLY ON SYSTEMS WITH
NON-VOLATILE MEMORY AND WITH APPROPRIATE HARDWARE).

7.4 MULTIPLE NCV11'S INTERFACE TESTING

THIS PROGRAM DOES NOT ''AUTO-SIZE'' THE NUMBER OF NCV11'S CONNECTED.
FOR EACH ADDITIONAL NCV11, THE OPERATOR MUST MODIFY LOCATIONS
$BASE, $VECT1 AND $CDW1 (REF. 7.1).

7.5 RESTRICTIONS

KWV11 REAL TIME CLOCK (M7952) MUST NOT BE CONNECTED TO M8C26 ''FAST-ON'' TABS.
UNLESS OTHERWISE STATED, NO EXTERNAL CONNECTIONS TO M8036 OR A017.

7.6 ADDRESS MAKER TABLE

SEQ 0006

THE M8036 CONTAINS LOGIC TO CONVERT A/D DATA INTO A RELATIVE ADDRESS. BELOW IS A TABLE WHICH SHOWS THE ADDRESS MAKER OUTPUT WITH THE "TEST CONTROL" MAINT. BIT SET. SETTING THIS BIT ENABLES THE STATES OF THE "GAIN, ZB ENABLE AND RESOLUTION" BITS SIMULATE THE CONVERTED A/D DATA.

RESOLUTION			ADDRESS MAKER OUTPUT BITS															
RES0	RES1	WORD*16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
1	1	1	B	G	Z	G	Z	G	1	1	1	G	Z	G	Z	G	1	1	1
1	1	0	0	B	G	Z	G	Z	G	1	1	G	Z	G	Z	G	1	1	0
1	0	1	0	0	B	G	Z	G	Z	G	1	0	G	Z	G	Z	G	1	0
1	0	0	0	0	0	B	G	Z	G	Z	G	1	G	Z	G	Z	G	1	0
0	1	1	0	0	0	0	B	G	Z	G	Z	G	0	G	Z	G	Z	G	0
0	1	0	0	0	0	0	0	B	G	Z	G	Z	G	G	Z	G	Z	G	0
0	0	1	0	0	0	0	0	0	B	G	Z	G	Z	G	G	Z	G	Z	G

0 = ALWAYS A 0
1 = ALWAYS A 1
G = GAIN FLOP STATE
Z = ZB ENABLE FLOP STATE
B = THE "AND" OF ZB ENABLE WITH ZB MAINT. INPUT

8.0 EXECUTION TIME

LOGIC: EXECUTION TIME RANGES FROM ABOUT 5 SECONDS WITH NO ITERATIONS
TO ABOUT 70 SECONDS WITH ITERATIONS ENABLED.
AN END PASS MESSAGE INDICATES ALL SUB-TESTS HAVE COMPLETED.
EACH ADDITIONAL 4K OF MEMORY WILL ADD ABOUT 10 SECONDS
TO THE LOGIC TEST RUNTIME.

DIFFERENTIAL LINEARITY: EXECUTION TIME IS ABOUT 90 SECONDS.

9.0 PROGRAM TEST DESCRIPTIONS

SEQ 0007

THE PROGRAM CONSISTS OF FOUR MAIN SECTIONS PLUS THREE AUXILIARY AND
THREE FACTORY OPTION CHECK-OUT SECTIONS.
THE OPERATOR SELECTS EACH SECTION VIA THE CONSOLE KEYBOARD.
BELOW IS A BRIEF DESCRIPTION OF EACH SECTION AND THE KEYBOARD CHARACTER.

MAIN SECTIONS

9.1 L LOGIC TEST (M8026 + M8036)

<NO INTERVENTION> <SETUP ONLY IF SA=210>

THE TEST CONTAINS A SERIES OF INDEPENDENT SUB-TESTS DESIGNED
TO TEST LOGIC FUNCTIONS AND DATA PATHS OF THE NCV11 GAMMA CAMERA INTERFACE.
A COMPLETE LIST OF TESTS IS AVAILABLE IN THE TABLE OF CONTENTS
AT THE BEGINNING OF THE LISTING. THE COMMENT FIELD WITHIN
EACH SUB-TEST TITLE CAN BE BENEFICIAL TO UNDERSTANDING THE LOGIC TESTED.
ADDITIONAL LOGIC TESTS OF THE A017 WILL BE EXECUTED IF THE
PROGRAM WAS STARTED AT LOC. 210.

9.2 D (M) DIFFERENTIAL LINEARITY (A017)

<REQUIRES SELF-TEST CONNECTOR IN J1 ON THE A017> <SETUP INTERVENTION ONLY>
<A017 SWITCH SET TO 'MAINT.' POSITION>

A FINITE VARIATION OF INPUT VOLTAGE EQUALS A GIVEN CONVERTED
VALUE OR STATE. THE TEST IS TO VERIFY THE WIDTH OF EACH POSSIBLE
STATE. AN INPUT VOLTAGE IS RAMPED AND THE NCV11 SAMPLES THE RESULT.
THE NCV11 INCREMENTS THE RESPECTIVE MEMORY LOCATION CORRESPONDING
TO THE INPUT VALUE. THE PROGRAM THEN DETERMINES IF ANY STATE HAS
BEEN SKIPPED, IS EXCESSIVELY WIDE OR NARROW. WHEN 'M' IS USED TO
SELECT THE SECTION, THE PROGRAM OUTPUT AN EXPANDED REPORT OF
THE 254. STATES. TIGHTER ERROR TOLERANCES ARE USED IF RUNNING
IN THE OPTION CHECKOUT AREA (SA 210).

9.3 F FINAL ACCEPTANCE

<REQUIRES SELF-TEST CONNECTOR IN J1 ON THE A017> <SETUP INTERVENTION ONLY>
<A017 SWITCH SET TO 'MAINT.' POSITION>

THE SUB-SECTION RUNS SECTION 'L' AND 'D' TO VERIFY PROPER OPERATION.

9.4 A ADJUSTMENT OF A017 AT FIELD SITE

SEQ 0008

<MANUAL INTERVENTION> <SETUP INTERVENTION>
<A017 SWITCH SET TO "OPER." POSITION>

COARSE ADJUSTMENT OT THE A017 CAN BE PERFORMED USING THIS SUB-SECTION.
THE OPERATOR MAY SELECT THE CAMERA AND GAIN TO BE SAMPLED USING THE
SWITCH REGISTER. A SERIES OF CONVERSIONS ARE TAKEN ON EACH CAMERA
AND THE OPERATOR IS INFORMED OF THE RESULTS.

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON "INPUT VOLTAGE OUT OF RANGE" ERROR
SW13=1	020000	INHIBIT ERROR OR CONVERTED VALUE TYPE-OUT
SW09=1	001000	INHIBIT CONVERSIONS ON CAMERA #3 GAIN - 1
SW08=1	000400	INHIBIT CONVERSIONS ON CAMERA #2 GAIN - 1
SW07=1	000200	INHIBIT CONVERSIONS ON CAMERA #1 GAIN 1
SW06=1	000100	INHIBIT CONVERSIONS ON CAMERA #0 GAIN 1
SW04=1	000020	INHIBIT CONVERSIONS ON JOYSTICK
SW03=1	000010	INHIBIT CONVERSIONS ON CAMERA #3 GAIN 2
SW02=1	000004	INHIBIT CONVERSIONS ON CAMERA #2 GAIN = 2
SW01=1	000002	INHIBIT CONVERSIONS ON CAMERA #1 GAIN 2
SW00=1	000001	INHIBIT CONVERSIONS ON CAMERA #0 GAIN 2

AUXILIARY SECTION

9.5 O OTHER TERMINAL ADDRESS

IN SOME CUSTOMER SITES, THE CONSOLE TERMINAL MAY NOT BE LOCATED
WITHIN EASY ACCESS TO THE NCV11. THIS SECTION ALLOWS THE OPERATOR
TO CHANGE THE PROGRAMS CONTROL TO ANOTHER TERMINAL.

9.6 H HELP THE OPERATOR

A LIST OF THE COMMANDS WILL BE TYPED TO REFRESH THE OPERATOR'S MEMORY.

9.7 S SOFTWARE SWITCH REGISTER

ENABLES THE OPERATOR TO CHANGE THE SOFTWARE SWITCH REGISTER WHEN
NOT RUNNING ANOTHER SUB-SECTION.

FACTORY OPTION CHECK-OUT SECTION

9.8 I INITIAL ADJUSTMENT OF A017

<MANUAL INTERVENTION> <SETUP INTERVENTION> <HARDWARE TESTER>
<A017 SWITCH SET TO "OPER." POSITION>

A KNOWN GOOD DIGITAL TO ANALOG (D/A) SUPPLIES A REFERENCE INPUT
VOLTAGE. THE OPERATOR IS INFORMED WHICH ADJUSTMENT TO PERFORM.
THE PURPOSE OF THE TEST IS TO VERIFY THAT THE ADJUSTMENT TO THE PRE-AMP
SECTION CAN BE PREFORMED. THE TEST DOES NOT ATTEMPT TO CALIBRATE
THE PRE-AMP, BUT ONLY TO FUNCTIONALLY VERIFY THE ADJUSTMENTS.

9.9 B BLASTING THE LINEARITY CORRECTION PROM

<MANUAL INTERVENTION> <SETUP INTERVENTION> <HARDWARE TESTER>
<A017 SWITCH SET TO "MAINT." POSITION>

THE TEST IS DESIGNED TO CREATE THE LINEARITY CORRECTION PROM (LCP). THE TESTER HARDWARE CONTAINS RAM STORAGE TO EMULATE THE LCP. THE PROGRAM WILL RUN DIFFERENTIAL LINEARITY. THE PROGRAM WILL THEN USE THE RESULTS TO DETERMINE THE CORRECT VALUE TO BE PLACED IN THE LCP. THE PROGRAM WILL THEN INFORM THE PROM BLASTER TO GENERATE A CORRECTED LINEARITY PROM FOR THE A017 MODULE. UPON COMPLETION OF GENERATION AND VERIFICATION STAGE, THE OPERATOR IS INSTRUCTED TO REMOVE THE EMULATOR CABLE FROM THE A017 AND INSERT THE NEWLY CREATED LCP. THE PROGRAM WILL RE-EXECUTE DIFFERENTIAL LINEARITY.

9.10 C CONTROL PROGRAM PROM

<MANUAL INTERVENTION> <SETUP INTERVENTION> <HARDWARE TESTER>

THE ROUTINE WILL CREATE THE M8036 CONTROL PROM. THE OPERATOR MAY CREATE SEVERAL COPIES BY REPEATELY EXECUTING THIS SECTION. THE PROGRAM WILL INFORM THE OPERATOR THE STEPS TO PERFORM.

LOCATION	OCTAL	BINARY
00	215	10001101
01	244	10100100
02	116	01001110
03	144	01100100
04	215	10001101
05	244	10100100
06	116	01001110
07	144	01100100
10	215	10001101
11	244	10100100
12	116	01001110
13	144	01100100
14	215	10001101
15	244	10100100
16	116	01001110
17	144	01100100
20	116	01001110
21	144	01100100
22	216	10001110
23	244	10100100

10.0 LISTING

TEST	DESCRIPTION
T1	VERIFY THE 8 NCV11 AND 2 NCV11 CLOCK BUS ADDRESSES RESPOND
T2	FLOAT A 1 ACROSS 10 BITS OF THE COMMAND/STATUS REG.
T3	VERIFY THAT ''INIT'' CLEARS THE CSR REGISTER
T4	VERIFY THAT ''CLEAR ALL'' CLEARS THE CSR REGISTER
T5	VERIFY LOW BYTE OPERATION OF THE ''CSR'' REGISTER
T6	FLOAT A 1 ACROSS 4 BITS OF THE SPECIAL FUNCTION REGISTER
T7	VERIFY THAT CLEARING HIGH BYTE OF SFR DOES NOT CLEAR LOW BYTE
T10	VERIFY THAT ''INIT'' CLEARS THE SFR REGISTER
T11	VERIFY THAT ''CLEAR ALL'' CLEARS THE SFR REGISTER
T12	FLOAT A 1 ACROSS THE WORD COUNT REGISTER
T13	FLOAT A 1 ACROSS THE BUS ADDRESS REGISTER
T14	FLOAT A 1 ACROSS THE OFFSET REGISTER
T15	VERIFY NO DUAL REGISTER SELECTION
T16	VERIFY ''CLR ALL'' CLEARS THE EXTENDED OFFSET BITS
T17	TEST THE ''ACTIVE'' FLOP CAN SET AND CLEAR
T20	VERIFY Z INPUTS CAUSE THE LOW 8 BITS OF 32 BIT COUNTER TO CHANGEIN MATRIX MODE
T21	VERIFY Z INPUTS CAUSE THE LOW 16 BITS OF 32 BIT COUNTER TO CHANGE
T22	VERIFY Z INPUTS CAUSE THE LOW 24 BITS OF 32 BIT COUNTER TO CHANGE
T23	VERIFY Z INPUTS CAUSE THE HIGH 8 BITS OF THE 32 BIT COUNTER TO CHANGE
T24	VERIFY Z INPUTS DO NOT CAUSE THE LOW 8 BITS OF 32 BIT COUNTER TO CHANGE IN LIST MODE
T25	TEST THAT ''WCA OVFL'' SETS Z/WC OVERFLOW FLOP AND ''CLR ALL'' CLEARS IT
T26	TEST THAT ''WCA OVFL'' SETS Z/WC OVERFLOW FLOP AND ''CLR WC OVFL'' CLEARS IT
T27	TEST THAT ''WCA OVFL'' GENERATES AN INTERRUPT
T30	VERIFY ''WCA OVFL'' CLEARS ''ACTIVE''
T31	VERIFY JOYSTICK DONE FLOP SETS
T32	VERIFY THAT ''RESET'' INSTRUCTION CLEARS THE JOY READY FLOP
T33	JOYSTICK DATA PATH = GAIN =0 ZB ENABLE =0 RES. = 000
T34	JOYSTICK DATA PATH = GAIN =1
T35	JOYSTICK DATA PATH = ZB ENABLE =1
T36	JOYSTICK DATA PATH RES. = 001
T37	JOYSTICK DATA PATH RES. = 010
T40	JOYSTICK DATA PATH RES. = 100

```
                        .LIST   ME
                        .NLIST  MD,MC,CND
                                .ENABL  ABS,AMA
167400                          $SWR=167400
000001                          $TN 1
                                .MCALL  .$CATCH,.$CMTAG,.$TYPE,.$EOP,.EQUAT,.HEADER,.$ERROR,.$TYPOCT
                                .MCALL  .$PARM,.$POWER,.$READ,.$SAVE,.$SCOPE,.SETUP,.$SPACE,.$SWDOC,.$TYPE,.$TRA
                                .MCALL  .SWRHI,.$ERRTYP,.$TYPDS,.$TYPBIN,SOFTSWR,.$CKSWR
                                .MCALL .$RDOCT,.$SIZE,.KT11,.$DIV,.$MULT,.$SAVE,.$SB2D,.$DB2D
                                .MCALL  .$ACT11,.$APTHDR,.$APTBLS,.$APTYPE
                        .TITLE  CZNCCA  NCV11   DIAGNOSTIC
                        ;*COPYRIGHT (C) 1978
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY RAYMOND SHOOP
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-D7QAC-C3), JAN 19, 1977.
                        ;*
                        .SBTTL  BASIC DEFINITIONS

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

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

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

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
```

```
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7

                        ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
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
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

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

```
                                .EQUIV  BIT07,BIT7
                                .EQUIV  BIT06,BIT6
                                .EQUIV  BIT05,BIT5
                                .EQUIV  BIT04,BIT4
                                .EQUIV  BIT03,BIT3
                                .EQUIV  BIT02,BIT2
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

                                ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
000004                          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                          TBITVEC=14              ;;''T'' BIT
000014                          TRTVEC= 14              ;;TRACE TRAP
000014                          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                          PWRVEC= 24              ;;POWER FAIL
000030                          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                          TRAPVEC=34              ;;''TRAP'' TRAP
000060                          TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                          TPVEC=  64              ;;TTY PRINTER VECTOR
000240                          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                                .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                                ;*KT11 VECTOR ADDRESS

000250                          MMVEC=  250

                                ;*KT11 STATUS REGISTER ADDRESSES

177572                          SR0=    177572
177574                          SR1=    177574
177576                          SR2=    177576
172516                          SR3=    172516

                                ;*USER ''I'' PAGE DESCRIPTOR REGISTERS

177600                          UIPDR0= 177600
177602                          UIPDR1= 177602
177604                          UIPDR2= 177604
177606                          UIPDR3= 177606
177610                          UIPDR4= 177610
177612                          UIPDR5= 177612
177614                          UIPDR6= 177614
177616                          UIPDR7= 177616

                                ;*USER 'D'' PAGE DESCRIPTOR REGISTORS

177620                          UDPDR0= 177620
177622                          UDPDR1= 177622
177624                          UDPDR2= 177624
177626                          UDPDR3= 177626
177630                          UDPDR4= 177630
177632                          UDPDR5= 177632
```

```
                        UDPDR6= 177634
                        UDPDR7= 177636

                        ;*USER ''I'' PAGE ADDRESS REGISTERS

                        UIPAR0= 177640
                        UIPAR1= 177642
                        UIPAR2= 177644
                        UIPAR3= 177646
                        UIPAR4= 177650
                        UIPAR5= 177652
                        UIPAR6= 177654
                        UIPAR7= 177656

                        ;*USER ''D'' PAGE ADDRESS REGISTERS

                        UDPAR0= 177660
                        UDPAR1= 177662
                        UDPAR2= 177664
                        UDPAR3= 177666
                        UDPAR4= 177670
                        UDPAR5= 177672
                        UDPAR6= 177674
                        UDPAR7= 177676

                        ;*SUPERVISOR ''I'' PAGE DESCRIPTOR REGISTERS

                        SIPDR0= 172200
                        SIPDR1= 172202
                        SIPDR2= 172204
                        SIPDR3= 172206
                        SIPDR4= 172210
                        SIPDR5= 172212
                        SIPDR6= 172214
                        SIPDR7= 172216

                        ;*SUPERVISOR ''D'' PAGE DESCRIPTOR REGISTERS

                        SDPDR0= 172220
                        SDPDR1= 172222
                        SDPDR2= 172224
                        SDPDR3= 172226
                        SDPDR4= 172230
                        SDPDR5= 172232
                        SDPDR6= 172234
                        SDPDR7= 172236

                        ;*SUPERVISOR ''I'' PAGE ADDRESS REGISTERS

                        SIPAR0= 172240
                        SIPAR1= 172242
                        SIPAR2= 172244
                        SIPAR3= 172246
                        SIPAR4= 172250
```

```
                        SIPAR5= 172252
                        SIPAR6= 172254
                        SIPAR7= 172256

                        ;*SUPERVISOR 'D'' PAGE ADDRESS REGISTERS

                        SDPAR0= 172260
                        SDPAR1= 172262
                        SDPAR2= 172264
                        SDPAR3= 172266
                        SDPAR4= 172270
                        SDPAR5= 172272
                        SDPAR6= 172274
                        SDPAR7= 172276

                        ;*KERNEL ''I'' PAGE DESCRIPTOR REGISTERS

                        KIPDR0= 172300
                        KIPDR1= 172302
                        KIPDR2= 172304
                        KIPDR3= 172306
                        KIPDR4= 172310
                        KIPDR5= 172312
                        KIPDR6= 172314
                        KIPDR7= 172316

                        ;*KERNEL 'D'' PAGE DESCRIPTOR REGISTERS

                        KDPDR0= 172320
                        KDPDR1= 172322
                        KDPDR2= 172324
                        KDPDR3= 172326
                        KDPDR4= 172330
                        KDPDR5= 172332
                        KDPDR6= 172334
                        KDPDR7= 172336

                        ;*KERNEL ''I'' PAGE ADDRESS REGISTERS

                        KIPAR0= 172340
                        KIPAR1= 172342
                        KIPAR2= 172344
                        KIPAR3= 172346
                        KIPAR4= 172350
                        KIPAR5= 172352
                        KIPAR6= 172354
                        KIPAR7= 172356

                        ;*KERNEL 'D'' PAGE ADDRESS REGISTERS

                        KDPAR0= 172360
                        KDPAR1= 172362
                        KDPAR2= 172364
                        KDPAR3= 172366
```

```
                        KDPAR4= 172370
                        KDPAR5= 172372
                        KDPAR6= 172374
                        KDPAR7= 172376


                                ABASE=172760            ;NCv11 BASE ADDRESS
                                AVECT1=140370           ;BR LEVEL 6 VECTOR TO 370
                                ACDW1=170420            ;NCV11 CLOCK ADDRESS
```

```
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                .SBTTL  TRAP CATCHER

            000000                      .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
            000174                      .=174
  000174    000000            DISPREG:  .WORD  0              ;;SOFTWARE DISPLAY REGISTER
  000176    000000            SWREG:    .WORD  0              ;;SOFTWARE SWITCH REGISTER
                                .SBTTL  STARTING ADDRESS(ES)
  000200    000137  002032            JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
  000204    000137  002012            JMP     RTRT            ;RESTART ADDRESS
  000210    000137  002020            JMP     TESTER          ;STARTING ADDRESS WHEN IN THE OPTION AREA

            000100                      .-100
  000100    000104  000200  000002           104,200,2                ;B EVENT SAFE GUARD
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
000106                                  $SVPC=.                 ;SAVE PC
000046                                  .-46
000046  030164                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000106                                  .=$SVPC                 ;; RESTORE PC
001000                                  .=1000
                                .SBTTL  APT PARAMETER BLOCK

                                ;;******************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;******************************************************************
001000                                  .$X=.   ;;SAVE CURRENT LOCATION
000024                                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
000044                                  .-44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
001000                                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;******************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000030                  $TSTM:  .WORD   30      ;;RUN TIM OF LONGEST TEST
001006  000010                  $PASTM: .WORD   10      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000030                  $UNITM: .WORD   30      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
001157  000             $TPFLG: .BYTE   0               ;;''TERMINAL AVAILABLE'' FLAG (BIT<07>=0-YES)
001160  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001162  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001164  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001170  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**********************************************************************
                                .EVEN
001174                  $MAIL:                  ;;APT MAILBOX
001174  000000          $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
001176  000000          $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
001200  000000          $TESTN: .WORD   ATESTN  ;;TEST NUMBER
001202  000000          $PASS:  .WORD   APASS   ;;PASS COUNT
001204  000000          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
```

```
                                        $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
                                        $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
                                        $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
                                        $ETABLE:                ;;APT ENVIRONMENT TABLE
                                        $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
                                        $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
                                        $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
                                        $USWR:  .WORD   AUSWR   ;;USER SWITCHES
                                        $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                        ;*                      BITS 15-11=CPU TYPE
                                        ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                        ;*                              11/70=06,PDQ=07,Q=10
                                        ;*                      BIT 10=REAL TIME CLOCK
                                        ;*                      BIT  9=FLOATING POINT PROCESSOR
                                        ;*                      BIT  8=MEMORY MANAGEMENT
                                        $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
                                        $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                        ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                        ;*                              900 NSEC CORE=001
                                        ;*                              300 NSEC BIPOLAR=002
                                        ;*                              500 NSEC MOS=003
                                        $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                        ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE'' ABOVE
                                        $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
                                        $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
                                        $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
                                        $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
                                        $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
                                        $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
                                        $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
                                        $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
                                        $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
                                        $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
                                        $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
                                        $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
                                        $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
                                        $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
                                        $ETEND:
                                        .MEXIT
```

```
                                        .SBTTL  ERROR POINTER TABLE

                                        ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                        ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                        ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                        ;*NOTE 1:       IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                        ;*NOTE 2:       EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                        ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                        ;*      DH              ;;POINTS TO THE DATA HEADER
                                        ;*      DT              ;;POINTS TO THE DATA
                                        ;*      DF              ;;POINTS TO THE DATA FORMAT


001256                                  $ERRTB:
001256  030220  032500  032730          E.11,DH2,DT2,DF0        ;M8026  NCV11 TIMEOUT
001264  033002
001266  030260  032500  032730          EM.',DH2,DT2,DF0        ;M8026  COMMAND-STATUS REGISTER ERROR
001274  033002
001276  030324  032500  032730          EM3 DH2,DT2,DF0         ;M8026  SPECIAL FUNCTION REGISTER ERROR
001304  033002
001306  0303?2  032500  032730          EM4, )H2,DT2,DF0        ;M8026  WORD COUNT REGISTER ERROR
001314  033002
001316  030432  032500  032730          EM5,I H2,DT2,DF0        ;M8026  BUS ADDRESS REGISTER ERROR
001324  033002
001326  030473  032500  032730          EM6, )H2,DT2,DF0        ;M8026  OFFSET REGISTER ERROR
001334  033002
001336  030527  032500  032730          EM7 DH2,DT2,DF0         ;M8026  DUAL REGISTER SELECTION ERROR
001344  033002
001346  030573  032500  032730          EM'0,DH2,DT2,.F0        ;M8026-M8036    LOW 16 BIT Z COUNT ERROR
001354  033002
001356  030640  032500  032730          :M11,DH2,DT2,D J        ;M8026-M8036    HIGH 16 BIT Z COUNT ERROR
001364  033002
001366  030706  032500  032730          EM12,DH2,DT2,DF0        ;M8026  Z COUNT STATUS ERROR
001374  033002
001376  030741  032500  032730          EM13,DH2,DT2,DF0        ;M8026  Z COUNT INTERRUPT ERROR
001404  033002
001406  030777  032500  032730          EM14,DH2,DT2,DF0        ;M8036  JOYSTICK STATUS ERROR
001414  033002
001416  031033  032500  032730          EM15,DH2,DT2,DF0        ;M8036  JOYSTICK DATA ERROR
001424  033002
001426  031065  032500  032730          EM16,DH2,DT2,DF0        ;M8036  DATA INCREMENT ERROR
001434  033002
001436  031120  032500  03273J          EM17,DH2,DT2,DF0        ;M8036  DATA DECREMENT ERROR
001444  033002
001446  031153  032500  032730          EM20,DH2,DT2,DF0        ;M8026-M8036    MATRIX MODE ADDRESS MAKER DATA ERROR
001454  033002
001456  031234  032500  03' 730          EM21,DH2,DT2,DF0        ;M8026  LIST MODE ADDRESS MAKER DATA ERROR
001464  033002
001466  031305  032500  0: 2730          EM22,DH2,DT2,DF0        ;M8026  LIST MODE TRANSFER BUS ADDRESS ERROR
001474  033002
001476  031360  032500  0 :2730          EM23,DH2,DT2,DF0        ;M8026  LIST MODE TRANSFER WORD COUNT ERROR
001504  033002
001506  031432  032500  0 2730          EM24,DH2,DT2,DF0        ;M8026  LIST MODE TRANSFER OFFSET ERROR
```

```
488  001514  033002
489  001516  031500  032500  032730          EM25,DH2,DT2,DF0        ;M8026  TIMEOUT STATUS ERROR
490  001524  033002
491  001526  031533  032500  032730          EM26,DH2,DT2,DF0        ;M8026  TIMEOUT INTERRUPT ERROR
492  001534  033002
493  001536  031571  032500  032730          EM27,DH2,DT2,DF0        ;M8026  SET 'EVENT' OR 'TIME' DATA ERROR
494  001544  033002
495  001546  031640  032500  032730          EM30,DH2,DT2,DF0        ;M8026  CELL INCREMENT DATA ERROR
496  001554  033002
497  001556  031700  032500  032730          EM31,DH2,DT2,DF0        ;M8026  CELL OVERFLOW STATUS ERROR
498  001564  033002
499  001566  031741  032500  032730          EM32,DH2,DT2,DF0        ;M8026  CELL OVERFLOW INTERRUPT ERROR
500  001574  033002
501  001576  032005  032524  032742          EM33,DH3,DT3,DF0        ;M8026  MATRIX MODE ADDRESS MUX ERROR
502  001604  033002
503  001606  032051  032500  032730          EM34,DH2,DT2,DF0        ;M8026  'TESTX' FUNCTION ERROR
504  001614  033002
505  001616  032106  032557  032756          EM35,DH4,DT4,DF0        ;M8026  DATA ERROR WHEN TRANSFERING TO EXTENDED MEMORY
506  001624  033002
507  001626  032173  032500  032730          EM36,DH2,DT2,DF0        ;M8026  LIST MODE TRANSFER STATUS ERROR
508  001634  033002
509  001636  032241  032524  032742          EM37,DH3,DT3,DF0        ;M8026  LIST MODE TRANSFER DATA ERROR
510  001644  033002
511  001646  032305  032500  032730          EM40,DH2,DT2,DF0        ;JUMPER-M8026-M7952     'EVENT' OR 'TIME' MARK ERROR
512  001654  033002
513  001656  032365  032465  032774          EM41,DH1,DT5,DF0        ;M7952  CLOCK TIMEOUT
514  001664  033002
515  001666  032425  032465  032722          EM42,DH1,DT1,DF0        ;M8217  INTERRUPT LEVEL ERROR
516  001674  033002
517                                  ;DL11 BLASTER COMM. ADDRESSES (THE OPERATOR MUST CHANGE IF DIFFERENT)
518  001676  176510                  DLICSR: 176510          ;INPUT STATUS REG.
519  001700  176512                  DLIBD:  176512          ;INPUT DATA
520  001702  176514                  DLOCSR: 176514          ;OUTPUT STATUS
521  001704  176516                  DLODB:  176516          ;OUTPUT DATA
522
523                                  ;VSV01 ADDRESSES (THE OPERATOR MUST CHANGE IF DIFFERENT)
524  001706  172600                  VTCHAR: 172600          ;CHAR. STATUS
525  001710  172602                  VTYPOS: 172602
526  001712  172603                  VTXPOS: 172603
527  001714  172604                  VTCXY:  172604
528  001716  172620                  VTCSR:  172620          ;MAP STATUS
529  001720  172622                  VTMAP:  172622
530  001722  172624                  VTPX:   172624
531  001724  172630                  VTINT:  172630
532
533                                  ;TESTER ADDRESSES (THE OPERATOR MUST CHANGE IF DIFFERENT)
534  001726  176416                  DACSR:  176416          ;KNOWN GOOD D/A STATUS
535  001730  176420                  DAC0:   176420
536  001732  176422                  DAC1:   176422
537  001734  176424                  DAC2:   176424
538  001736  176426                  DAC3:   176426
```

```
                                ;KWV11 PROGRAM GENERATED ADDRESSES (THE OPER. ONLY HAS TO CHANGE '$CDW1')

001740  170420                  KWCSR:  ACDW1           ;KWV11 STATUS REGISTER
001742  170421                  KWCSR1: ACDW1+1         ;KWV11 STATUS REG. HIGH BYTE
001744  170422                  KWPSR:  ACDW1+2         ;KWV11 PRESET REGISTER

                                ;NCV11 PROGRAM GENERATER ADDRESSES (THE OPER. ONLY HAS TO CHANGE '$BASE')

001746  172760                  CSR:    ABASE
001750  172762                  OFF:    ABASE+2
001752  172764                  WCR:    ABASE+4
001754  172766                  BAR:    ABASE+6
001756  172770                  SFR:    ABASE+10
001760  172772                  ADM:    ABASE+12
001762  172774                  JOY:    ABASE+14
001764  172776                  BAR1:   ABASE+16
001766  172761                  CSRHB:  ABASE+1
001770  172771                  SFRHB:  ABASE+11

                                ;NCV11 PROGRAM GENERATED VECTORS (THE OPER. ONLY HAS TO CHANGE LOW 9 BITS OF '$VECT1')

001772  140370                  VECTA0: AVECT1
001774  140372                  VECTA1: AVECT1+2
001776  140374                  VECTB0: AVECT1+4
002000  140376                  VECTB1: AVECT1+6

                                ;NCV11 PROGRAM GENERATED BR LEVEL (THE OPER. ONLY HAS TO CHANGE TOP 3 BITS OF '$VECT1')

002002  000300                  BRLEV:  300             ;BR LEVEL 6

002004  000000                  $TEMP:  0
002006  000001                  CPUDL0: 1
002010  000020                  CPUDL1: 20

        040000                          CLRWCO=BIT14
        004000                          CLRALL=BIT11
        000400                          ENDDMA=BIT8
        000010                          TSTDMA=BIT3
        000004                          TSTCON=BIT2
        000002                          TFSTZ=BIT1
        000001                          REDJOY=BIT0
        000000                          MORTST 0        ;DEFINE TO RESTORE TEST CODE

```

```
                                RTRT:   INC     AGAN            ;SET RESTART FLAG
                                        BR      RTRTA
                                TESTER: INC     WFMODE          ;SET FLAG INDICATING OPTION AREA MODE
                                        CLR     AGAN
                                        BR      RTRTA
                                BEGIN:  CLR     AGAN
                                        CLR     WFMODE          ;CLEAR FLAG INDIC. OPTION TEST AREA MODE
                                RTRTA:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
                                        MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
                                        CLR     (R6)+           ;;CLEAR MEMORY LOCATION
                                        CMP     #SWR,R6 ;;DONE?
                                        BNE     .-6             ;;LOOP BACK IF NO
                                        MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
                                        MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
                                        MOV     #340,@#IOTVEC+2 ;;LEVEL 7
                                        MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
                                        MOV     #340,@#EMTVEC+2 ;;LEVEL 7
                                        MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
                                        MOV     #340,@#TRAPVEC+2;LEVEL 7
                                        MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
                                        MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                        MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
                                        CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
                                        CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
                                        MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
                                        MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
                                        MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
                                        MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
                                        MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
                                        MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
                                        MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
                                        CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
                                        BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT  -1
                                        BR      65$             ;;BRANCH IF NO TIMEOUT
                                64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
                                        RTI
                                65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
                                        MOV     #DISPREG,DISPLAY
                                66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

                                        CLR     $PASS           ;;CLEAR PASS COUNT
                                        BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
                                        BEQ     67$             ;;YES,USE NON-APT SWITCH
                                        MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
                                67$:
```

```
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
002312  005227  177777                    INC     #-1             ;;FIRST TIME?
002316  001044                            BNE     68$             ;;BRANCH IF NO
002320  022737  030164  000042            CMP     #$ENDAD,@#42    ;;ACT-11?
002326  001440                            BEQ     68$             ;;BRANCH IF YES
002330  104401  002376                    TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
002334  005737  000042                    TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
002340  001012                            BNE     70$             ;;BRANCH IF YES
002342  123727  001214  000001            CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
002350  001406                            BEQ     70$             ;;BRANCH IF YES
002352  023727  001140  000176            CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
002360  001005                            BNE     71$             ;;BRANCH IF NO
002362  104406                            GTSWR                   ;;GET SOFT-SWR SETTINGS
002364  000403                            BR      71$
002366  112737  000001  001134  70$:      MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
002374                        71$:
002374  000415                            BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:    .ASCIZ  <CRLF>#CZNCCA   NCV11 DIAGNOSTIC#<CRLF>
002430                          68$:
                                        .SBTTL  PRIME THE NCV11 ADDRESSES FROM THE DEFAULT VALUES
002430  112737  000103  052474            MOVB    #'C,RUNIT       ;LOAD ''CONTROL'' SECTION BUS TRAP FLAG
002436  012737  003210  000004            MOV     #BUSTRP,@#ERRVEC        ;LOAD BUS TRAP RETURN ADDRESS
002444  012700  001746                    MOV     #CSR,R0         ;GET ADDRESS POINTER
002450  013701  001250                    MOV     $BASE,R1        ;GET DEFAULT ADDRESS
002454  010120                  1$:       MOV     R1,(R0)+        ;LOAD AN VALUE
002456  062701  000002                    ADD     #2,R1           ;BUMP THE VALUE
002462  022700  001766                    CMP     #CSRHB,R0       ;TEST IF DONE
002466  001372                            BNE     1$              ;BR IF NOT
002470  013710  001250                    MOV     $BASE,(R0)      ;LOAD ODD BYTE
002474  005220                            INC     (R0)+           ;ADDRESSES
002476  013710  001250                    MOV     $BASE,(R0)
002502  062720  000011                    ADD     #11,(R0)+       ;
002506  013701  001244                    MOV     $VECT1,R1       ;GET DEFAULT VECTOR
002512  010102                            MOV     R1,R2           ;COPY R1
002514  000302                            SWAB    R2              ;EXCHANGE BYTES
002516  042702  177437                    BIC     #177437,R2      ;STRIP OFF ALL BUT BR LEVEL BITS
002522  010237  002002                    MOV     R2,BRLEV        ;SAVE FOR USE LATER
002526  042701  160000                    BIC     #160000,R1      ;CLEAR OFF BR LEVEL
002532  010120                  2$:       MOV     R1,(R0)+        ;LOAD VECTOR
002534  005721                            TST     (R1)+           ;BUMP THE VALUE
002536  022700  002002                    CMP     #VECTB1+2,R0    ;TEST IF DONE
002542  001373                            BNE     2$
002544  013737  001254  001740            MOV     $CDW1,KWCSR     ;GET CLOCK ADDRESS
002552  013737  001740  001742            MOV     KWCSR,KWCSR1    ;SET CLOCK HIGH BYTE ADDR.
002560  013737  001740  001744            MOV     KWCSR,KWPSR     ;SET CLOCK PRESET ADDRESS
002566  005237  001742                    INC     KWCSR1
002572  062737  000002  001744            ADD     #2,KWPSR
```

```
                                                .SBTTL  DIAGNOSTIC IDENTIFICATION TYPEOUT
002600  005737  050116                        TST     AGAN            ;TEST IF RESTART
002604  001017                                BNE     RBEGO           ;BR IF YES
002606  105737  001134                        TSTB    $AUTOB          ;TEST IF ACT/APT
002612  001405                                BEQ     3$              ;BR IF NOT
002614  012737  177777  050106                MOV     #-1,RUNDIF      ;INDICATE TO RUN DIFLIN IN CHAIN MODE
002622  000137  003356                        JMP     LOGIC           ;YES; RUN LOGIC AND DIFLIN TESTS
002626  005737  050110          3$:           TST     WFMODE          ;TEST IF OPTION CHECKOUT MODE
002632  001402                                BEQ     4$              ;BR IF YES
002634  104401                                TYPE
002636  046256                                LISTO                   ;TELL OPERATOR ABOUT OPTION AREA TESTS
002640  104401                          4$:   TYPE
002642  046466                                LIST                    ;TELL OPERATOR ABOUT THE TEST
                                                .SBTTL  KEYBOARD COMMAND DECODER
002644  104401  047101                  RBEGO:  TYPE,   LIST1           ;PRINT 'DOT'
002650  104411                          1$:     RDLIN                   ;READ THE RESPONSE
002652  013637  052474                        MOV     @(SP)+,RUNIT    ;GET THE CHARACTER
002656  142737  000040  052474                BICB    #40,RUNIT       ;ENSURE UPPER CASE
002664  112737  000042  052475                MOVB    #'",RUNIT+1     ;FIX ASCII MESSAGE
002672  122737  000101  052474                CMPB    #'A,RUNIT       ;DETERMINE IF AN A
002700  001002                                BNE     2$              ;BR IF NOT
002702  000137  003276                        JMP     TFSITE          ;RUN SITE ADJUSTMENT LOOP
002706  005737  050110          2$:           TST     WFMODE          ;TEST IF ON THE TESTER
002712  001406                                BEQ     3$              ;BR IF NOT
002714  122737  000102  052474                CMPB    #'B,RUNIT       ;DETERMINE IF AN B
002722  001002                                BNE     3$              ;BR IF NOT
002724  000137  042322                        JMP     BLAST           ;RUN THE BLASTER SECTION
002730  122737  000104  052474  3$:           CMPB    #'D,RUNIT       ;DETERMINE IF AN D
002736  001004                                BNE     4$              ;BR IF NOT
002740  005037  050106                        CLR     RUNDIF          ;INDICATE DIFLIN IS NOT TO BE RUN
002744  000137  043712                        JMP     DIFLIN          ;AND RUN IT
002750  122737  000106  052474  4$:           CMPB    #'F,RUNIT       ;DETERMINE IF AN F
002756  001005                                BNE     5$              ;BR IF NOT
002760  012737  177777  050106                MOV     #-1,RUNDIF      ;INDICATE RUN DIFLIN AFTER THE LOGIC TEST
002766  000137  003356                        JMP     LOGIC           ;AND RUN LOGIC TEST
002772  005737  050110          5$:           TST     WFMODE          ;TEST IF OPTION CHECKOUT MODE
002776  001406                                BEQ     6$              ;BR IF NOT
003000  122737  000111  052474                CMPB    #'I,RUNIT       ;DETERMIN IF AN I
003006  001002                                BNE     6$              ;BR IF NOT
003010  000137  047106                        JMP     POTIME          ;RUN IN-HOUSE ADJUSTMENT LOOP
003014  122737  000114  052474  6$:           CMPB    #'L,RUNIT       ;DETERMINE IF AN L
003022  001004                                BNE     7$              ;BR IF NOT
003024  005037  050106                        CLR     RUNDIF          ;INDICATE NO DIFLIN
003030  000137  003356                        JMP     LOGIC
003034  122737  000110  052474  7$:           CMPB    #'H,RUNIT       ;DETERMINE IF AN H
003042  001002                                BNE     10$             ;BR IF NOT
003044  000137  002032                        JMP     BEGIN           ;START OVER
003050  122737  000037  052474  10$:          CMPB    #37,RUNIT       ;DETERMINE IF AN "?"
003056  001002                                BNE     11$             ;BR IF NOT
003060  000137  002032                        JMP     BEGIN           ;START OVER
003064  005737  050110          11$:          TST     WFMODE          ;TEST IF OPTION CHECKOUT MODE
003070  001406                                BEQ     12$             ;BR IF NOT
003072  122737  000124  052474                CMPB    #'T,RUNIT       ;DETERMINE IF AN T
0031C0  001002                                BNE     12$             ;BR IF NOT
```

```
   736  003102  000137  042050                 JMP     BTALK           ;KEYBOARD LOOP WITH BLASTER
   737  003106  005737  050110          12$:    TST     WFMODE          ;TEST IF OPTION CHECKOUT MODE
   738  003112  001406                         BEQ     13$             ;BR IF NOT
   739  003114  122737  000103  052474         CMPB    #'C,RUNIT       ;DETERMINE IF AN C
   740  003122  001002                         BNE     13$             ;BR IF NOT
   741  003124  000137  043004                 JMP     PBLAST          ;RUN M8036 PROM BLASTING CODE
   742  003130  122737  000117  052474  13$:    CMPB    #'O,RUNIT       ;DETERMINE IF AN O
   743  003136  001002                         BNE     14$             ;BR IF NOT
   744  003140  000137  003240                 JMP     XTTY            ;CHANGE THE TTY ADDRESS
   745  003144  122737  000123  052474  14$:    CMPB    #'S,RUNIT       ;DETERMINE IF AN S
   746  003152  001002                         BNE     15$             ;BR IF NOT
   747  003154  104406                         GTSWR                   ;GET SWITCH REGISTER VALUE
   748  003156  000632                         BR      RBEGO           ;RESTART
   749  003160  122737  000115  052474  15$:    CMPB    #'M,RUNIT       ;DETERMINE IF AN M
   750  003166  001005                         BNE     FATFNG          ;BR IF NOT
   751  003170  012737  000377  050106         MOV     #377,RUNDIF     ;INDICATE DIFLIN WITH EXPAND REPORT
   752  003176  000137  043712                 JMP     DIFLIN          ;RUN DIF LIN
```

```
                                        ;THE OPERATOR SELECTED THE WRONG THING - THEY SOMETIME HAVE ''FAT FINGERS''
003202  104401                  FATFNG: TYPE
003204  001170                          $QUES                   ;TYPE QUESTION MARK
003206  000616                          BR      RBEGO
                                        ;RETURN TO HERE UPON UNEXPECTED BUS TRAP
003210  104401  052424          BUSTRP: TYPE,   FATAL0          ;REPORT FATAL TRAP TO THE OPERATOR
003214  011646                          MOV     (SP),-(SP)      ;GET TRAP ADDRESS
003216  104402                          TYPOC
003220  005777  175714          1$:     TST     @SWR            ;TEST IF HALT ON ERROR
003224  100001                          BPL     2$              ;BR IF SET
003226  000000                          HALT                    ;FATAL BUS TRAP DETECTED
003230  104401  052514          2$:     TYPE,   RUNITA
003234  000137  002012                  JMP     RTRT            ;RESTART
                                        ;ROUTINE TO INPUT NEW TTY ADDRESS FROM THE OPERATOR
003240  104401  052604          XTTY:   TYPE,   WARN0           ;TELL THE OPERATOR THE RULES
003244  104412                          RDOCT                   ;LISTEN TO THE OPERATOR
003246  012600                          MOV     (SP)+,R0        ;READ THEIR INPUT
003250  001754                          BEQ     FATFNG          ;BR IF JUST ''CR''
003252  005710                          TST     (R0)            ;ADDRESS THE DEVICE AND CHECK BUS TRAP
                                        ;IF NO BUS TRAP - THEY MUST KNOW WHAT THEY ARE DOING !
003254  012701  001144                  MOV     #$TKS,R1        ;GET POINTER
003260  010021                  1$:     MOV     R0,(R1)+        ;LOAD THE VALUE
003262  005720                          TST     (R0)+           ;BUMP VALUE
003264  020127  001154                  CMP     R1,#$TKS+10     ;TEST IF DONE ALL ADDRESSES
003270  001373                          BNE     1$              ;BR IF NOT
003272  000137  002042                  JMP     RTRTA           ;RETYPE THE HEADER AGAIN ON THE NEW TERMINAL

                                        ;OPERATOR CHOSE THE FIELD ADJUSTMENT LOOP
003276  104401  051250          TFSITE: TYPE,   PRIM6           ;TELL OPER. ABOUT A017
003302  104401  041374                  TYPE,   FIELDI          ;ASK THE OPERATOR FOR INPUT Z MODE
003306  005037  040410                  CLR     INOUTZ          ;DEFAULT TO USER SUPPLIED Z PULSES
003312  104411                          RDLIN                   ;READ OPER. INPUT
003314  013637  003350                  MOV     @(SP)+,10$      ;GET CHAR
003320  142737  000040  003350          BICB    #40,10$         ;ENSURE UPPER CASE
003326  122737  000131  003350          CMPB    #'Y,10$         ;TEST FOR 'Y' INPUT
003334  001003                          BNE     1$              ;BR IF NOT 'Y'
003336  012737  000002  040410          MOV     #2,INOUTZ       ;SET MAINT Z CONSTANT
003344  000137  037354          1$:     JMP     FSITE           ;NOW DO THE LOOP
003350  000000                  10$:    0

003352  000000                  ADNOKT: 0                       ;LAST ADDRESS WITHOUT M.M. OR XXDP
003354  000000                  NLSI11: 0                       ;NON-ZERO INDICATES LSI-11 PROCESSOR
```

```
 798
 799  003356  000005                    LOGIC:  RESET
 800  003360  005737  001202                    TST     $PASS           ;TEST IF FIRST PASS
 801  003364  001124                            BNE     TST1            ;;BR IF NOT
 802  003366  005737  050110                    TST     WFMODE          ;CHECK IF ON TESTER
 803  003372  001402                            BEQ     5$              ;BR IF NOT
 804  003374  104401  052332                    TYPE,   PRIM4           ;TELL OPERATOR CABLE MUST BE CONNECTED
 805  003400  005037  036120            5$:     CLR     $KT11           ;INDICATE NO KT11
 806  003404  004737  036062                    JSR     PC,$SIZE        ;SIZE BASIC MEMORY SIZE
 807  003410  162737  001000  036400            SUB     #1000,$LSTAD    ;PROTECT ABSLDR
 808  003416  005737  000042                    TST     @#42            ;TEST IF XXDP CHAIN MODE ?
 809  003422  001403                            BEQ     1$              ;BR IF NOT CHAIN MODE
 810  003424  162737  006200  036400            SUB     #3200.,$LSTAD   ;LEAVE ROOM FOR XXDP MONITOR
 811  003432  013737  036400  003352  1$:      MOV     $LSTAD,ADNOKT   ;SAVE BASIC MEMORY STACE
 812  003440  012737  000200  036120            MOV     #BIT7,$KT11     ;WITH M.M.
 813  003446  004737  036062                    JSR     PC,$SIZE        ;DETERMINE THE AMOUNT OF MEMORY
 814  003452  005737  036120                    TST     $KT11           ;TEST IF KT11 IS HERE
 815  003456  100025                            BPL     2$              ;BR IF NOT
 816  003460  012737  000200  172342            MOV     #200,@#KIPAR1   ;
 817  003466  012737  077406  172302            MOV     #77406,@#KIPDR1
 818  003474  012737  000400  172344            MOV     #400,@#KIPAR2
 819  003502  012737  077406  172304            MOV     #77406,@#KIPDR2
 820  003510  012737  077406  172306            MOV     #77406,@#KIPDR3 ;4K, UP R/W FOR 60000
 821  003516  012737  007600  172356            MOV     #7600,@#KIPAR7  ;I/O PAGE MAP
 822  003524  012737  077406  172316            MOV     #77406,@#KIPDR7 ;4K, UP, R/W FOR I/O PAGE ACCESS
 823  003532  105737  001134            2$:     TSTB    $AUTOB          ;TEST IF AUTO MODE
 824  003536  001016                            BNE     4$              ;;BR IF AUTO MODE
 825  003540  104401  052545                    TYPE,   MSGMEM          ;TELL OPERATOR THE AMOUNT OF MEMORY
 826  003544  013700  036402                    MOV     $LSTBK,R0       ;GET LAST GOOD BANK
 827  003550  006200                            ASR     R0
 828  003552  006200                            ASR     R0
 829  003554  006200                            ASR     R0
 830  003556  006200                            ASR     R0
 831  003560  006200                            ASR     R0
 832  003562  005200                            INC     R0              ;CONVERT
 833  003564  010046                            MOV     R0,-(SP)        ;TYPE OUT DEC. VALUE
 834  003566  104405                            TYPDS
 835  003570  104401  052565                    TYPE,   MSGK            ;AND THEN TYPE OUT THE REMAINDER OF MEMORY MESSAGE
 836  003574  013737  000004  002004  4$:      MOV     @#ERRVEC,$TEMP  ;SAVE LOC 4 VALUE
 837  003602  012737  003630  000004            MOV     #3$,@#ERRVEC    ;LOAD LOC 4
 838  003610  005037  003354                    CLR     NLSI11          ;INDICATE NOT A LSI-11 CPU
 839  003614  005037  177776                    CLR     PSW             ;SHOULD FAIL IF LSI-11
 840  003620  013737  002004  000004            MOV     $TEMP,@#ERRVEC  ;RESTORE LOC 4 VALUE
 841  003626  000403                            BR      TST1            ;;BR IF TEST
 842  003630  005237  003354            3$:     INC     NLSI11                  ;SET AN LSI-11 FLAG
 843  003634  000002                            RTI                             ;RETURN
```

```
                                        ;;*****************************************************************
                                        ;*TEST 1         VERIFY THE 8 NCV11 AND 2 NCV11 CLOCK BUS ADDRESSES RESPOND
                                        ;;*****************************************************************
003636  000004                  TST1:   SCOPE
003640  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003646  013737  000004  004014          MOV     @#ERRVEC,10$            ;SAVE BUS TRAP VECTOR
003654  012737  003752  000004          MOV     #1$,@#ERRVEC            ;LOAD BUS TRAP VECTOR
003662  005777  176060                  TST     @CSR                    ;ADDRESS
003666  005777  176056                  TST     @OFF                    ;
003672  005777  176054                  TST     @WCR                    ;THE
003676  005777  176052                  TST     @BAR                    ;
003702  005777  176050                  TST     @SFR                    ;NCV11
003706  005777  176046                  TST     @ADM                    ;
003712  005777  176046                  TST     @BAR1                   ;ADDRESSES
003716  012737  003760  000004          MOV     #2$,@#ERRVEC            ;LOAD NEW RETURN
003724  005777  176010                  TST     @KWCSR                  ;TEST CLOCK
003730  005777  176010                  TST     @KWPSR                  ;ADDRESSES
                                        ;TEMP FIX THE DIAG TO NOT USE CLOCK JUMPERS
003734  012737  000001  004016          MOV     #1,DEADKW       ;INDICATE NCV11 CLOCK IS NOT THERE;TEMP. FIX UNTIL ECO M
003742  013737  004014  000004          MOV     10$,@#ERRVEC            ;RESTORE THE BUS TRAP VECTOR
003750  000423                          BR      TST2            ;;BR AND TEST THE NCV11
003752  022626                  1$:     CMP     (SP)+,(SP)+             ;CLEAN THE STACK
003754  104001                          ERROR   1                       ;BUS TRAP WHEN REFERENCING THE NCV11
003756  000411                          BR      3$
003760  022626                  2$:     CMP     (SP)+,(SP)+             ;CLEAN THE STACK
003762  104041                          ERROR   41                      ;BUS TRAP WHEN REFERENCING THE NCV11 CLOCK
003764  012737  000001  004016          MOV     #1,DEADKW               ;INDICATE NCV11 CLOCK IS NOT THERE
003772  013737  004014  000004          MOV     10$,@#ERRVEC            ;RESTORE LOC 4
004000  000407                          BR      TST2            ;;
004002  013737  004014  000004  3$:     MOV     10$,@#ERRVEC            ;RESTORE BUS TRAP VECTOR
004010  000137  030076                  JMP     $EOP                    ;EXIT
004014  000000                  10$:    0
004016  000000                  DEADKW: 0                               ;NON-ZERO SAYS NO NCV11 CLOCK
```

```
                                        ;;*****************************************************************
                                        ;*TEST 2         FLOAT A 1 ACROSS 10 BITS OF THE COMMAND/STATUS REG.
                                        ;;*****************************************************************
004020  000004                  TST2:   SCOPE
004022  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
004030  012737  004044  001110          MOV     #1$,$LPERR      ;LOAD LOOP ADDRESS
004036  012737  004000  002004          MOV     #BIT11,$TEMP            ;LOAD INITIAL REG. VALUE

004044  013777  002004  175674  1$:     MOV     $TEMP,@CSR              ;LOAD CSR REG.
004052  017737  175670  001126          MOV     @CSR,$BDDAT             ;READ CSR
004060  013737  002004  001124          MOV     $TEMP,$GDDAT            ;LOAD EXPECTED
004066  052737  000200  001124          BIS     #BIT7,$GDDAT            ;FUDGE THE ''READY'' BIT
004074  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE VALUES
004102  001401                          BEQ     2$              ;;BR IF SAME
004104  104002                          ERROR   2                       ;UNEXPECTED VALUE IN THE CSR REGISTER

004106  006237  002004          2$:     ASR     $TEMP                   ;TRY THE NEXT DATA BIT
004112  022737  000001  002004          CMP     #1,$TEMP                ;TEST IF NOW BIT 0
004120  001351                          BNE     1$                      ;BR IF NOT

                                        ;;*****************************************************************
                                        ;*TEST 3         VERIFY THAT ''INIT'' CLEARS THE CSR REGISTER
                                        ;;*****************************************************************
004122  000004                  TST3:   SCOPE
004124  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004132  012777  007776  175606          MOV     #7776,@CSR              ;LOAD CSR REG.
004140  000005                          RESET                           ;INITILIZE THE REGISTER
004142  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED VALUE
004150  017737  175572  001126          MOV     @CSR,$BDDAT             ;READ CSR REG.
004156  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE VALUES
004164  001401                          BEQ     TST4            ;;BR IF EQUAL
004166  104002                          ERROR   2                       ;''BUS INIT'' FAILED TO CLEAR CSR REG.

                                        ;;*****************************************************************
                                        ;*TEST 4         VERIFY THAT ''CLEAR ALL'' CLEARS THE CSR REGISTER
                                        ;;*****************************************************************
004170  000004                  TST4:   SCOPE
004172  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004200  012777  007776  175540          MOV     #7776,@CSR              ;LOAD CSR REG.
004206  012777  004000  175542          MOV     #CLRALL,@SFR            ;GENERATE ''CLR ALL L''
004214  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED VALUE
004222  017737  175520  001126          MOV     @CSR,$BDDAT             ;READ THE CSR REG.
004230  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
004236  001401                          BEQ     TST5            ;;BR IF SAME
004240  104002                          ERROR   2                       ;''CLR ALL L'' FAILED TO CLEAR CSR REG.
```

```
                                ;;*****************************************************************
                                ;*TEST 5         VERIFY LOW BYTE OPERATION OF THE ''CSR'' REGISTER
                                ;;*****************************************************************
004242  000004          TST5:   SCOPE
004244  012737  000100  001160        MOV     #100,$TIMES     ;;DO 100 ITERATIONS
004252  012777  003636  175466        MOV     #3636,@CSR              ;LOAD CSR REGISTER
004260  012737  003600  001124        MOV     #3600,$GDDAT            ;LOAD EXPECTED VALUE
004266  105077  175454                CLRB    @CSR                    ;CLEAR LOW BYTE
004272  017737  175450  001126        MOV     @CSR,$BDDAT             ;READ STATUS REG.
004300  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
004306  001401                        BEQ     1$              ;;BR IF SAME
004310  104002                        ERROR   2                       ;CLEARING LOW BYTE OF THE CSR CHANGED THE HIGH B
004312  012777  003636  175426  1$:   MOV     #3636,@CSR              ;LOAD CSR REGISTER
004320  012737  000236  001124        MOV     #236,$GDDAT             ;LOAD EXPECTED VALUE
004326  105077  175434                CLRB    @CSRHB                  ;CLEAR HIGH BYTE OF THE CSR
004332  017737  175410  001126        MOV     @CSR,$BDDAT             ;READ STATUS
004340  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
004346  001401                        BEQ     TST6            ;;BR IS SAME
004350  104003                        ERROR   3                       ;CLEARING HIGH BYTE OF CSR CHANGED THE LOW BYTE
                                ;;*****************************************************************
                                ;*TEST 6         FLOAT A 1 ACROSS 4 BITS OF THE SPECIAL FUNCTION REGISTER
                                ;;*****************************************************************
004352  000004          TST6:   SCOPE
004354  012737  004370  001110        MOV     #1$,$LPERR              ;LOAD LOOP ADDRESS
004362  012737  000020  002004        MOV     #BIT4,$TEMP             ;LOAD INITIAL REG. VALUE

004370  013777  002004  175360  1$:   MOV     $TEMP,@SFR              ;LOAD SFR REG.
004376  017737  175354  001126        MOV     @SFR,$BDDAT             ;READ SFR
004404  013737  002004  001124        MOV     $TEMP,$GDDAT            ;LOAD EXPECTED
004412  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE THE VALUES
004420  001401                        BEQ     2$              ;;BR IF SAME
004422  104003                        ERROR   3                       ;UNEXPECTED VALUE IN THE SFR REGISTER
004424  006237  002004          2$:   ASR     $TEMP                   ;TRY THE NEXT DATA BIT
004430  022737  000001  002004        CMP     #1,$TEMP                ;TEST IF NOW BIT 0
004436  001354                        BNE     1$                      ;BR IF NOT

                                ;;*****************************************************************
                                ;*TEST 7         VERIFY THAT CLEARING HIGH BYTE OF SFR DOES NOT CLEAR LOW BYTE
                                ;;*****************************************************************
004440  000004          TST7:   SCOPE
004442  012737  000010  001160        MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004450  012777  000014  175300        MOV     #14,@SFR                ;LOAD THE S.F. REGISTER
004456  012737  000014  001124        MOV     #14,$GDDAT              ;LOAD THE EXPECTED VALUE
004464  105077  175300                CLRB    @SFRHB                  ;CLEAR HIGH BYTE OF S.F. REG.
004470  017737  175262  001126        MOV     @SFR,$BDDAT             ;READ THE REGISTER
004476  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE THE VALUES
004504  001401                        BEQ     TST10           ;;BR IF SAME
004506  104003                        ERROR   3                       ;CLEARING HIGH BYTE OF CSR REG. CHANGED THE LOW
```

```
 972                                    ;;*********************************************************************
 973                                    ;*TEST 10       VERIFY THAT "INIT" CLEARS THE SFR REGISTER
 974                                    ;;*********************************************************************
 975  004510  000004                   TST10:  SCOPE
 976  004512  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 977  004520  012777  000016  175230           MOV     #16,@SFR                ;LOAD SFR REG.
 978  004526  000005                           RESET                   ;INITILIZE THE REGISTER
 979  004530  005037  001124                   CLR     $GDDAT                  ;LOAD EXPECTED
 980  004534  017737  175216  001126           MOV     @SFR,$BDDAT             ;READ SFR REG.
 981  004542  001401                           BEQ     TST11           ;;BR IF EQUAL
 982  004544  104003                           ERROR   3                       ;"BUS INIT" FAILED TO CLEAR SFR REG.
 983
 984                                    ;;*********************************************************************
 985                                    ;*TEST 11       VERIFY THAT "CLEAR ALL" CLEARS THE SFR REGISTER
 986                                    ;;*********************************************************************
 987  004546  000004                   TST11:  SCOPE
 988  004550  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 989  004556  012777  000016  175172           MOV     #16,@SFR                ;LOAD SFR REG.
 990  004564  052777  004000  175164           BIS     #CLRALL,@SFR            ;GENERATE "CLR ALL L"
 991  004572  005037  001124                   CLR     $GDDAT                  ;LOAD EXPECTED VALUE
 992  004576  017737  175154  001126           MOV     @SFR,$BDDAT             ;READ THE SFR REG.
 993  004604  001401                           BEQ     TST12           ;;BR IF SAME
 994  004606  104003                           ERROR   3                       ;"CLR ALL L" FAILED TO CLEAR SFR REG.
 995
```

```
                                        ;;*******************************************************************
                                        ;*TEST 12       FLOAT A 1 ACROSS THE WORD COUNT REGISTER
                                        ;;*******************************************************************
004610  000004                  TST12:  SCOPE
004612  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
004620  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE
004626  012737  004634  001110          MOV     #1$,$LPERR              ;LOAD LOOP ADDRESS ON ERROR
                                
004634  012777  004000  175114  1$:     MOV     #CLRALL,@SFR            ;RESET THE DEVICE
004642  013777  001124  175102          MOV     $GDDAT,@WCR             ;LOAD WORD COUNT "A"
004650  017737  175076  001126          MOV     @WCR,$BDDAT             ;READ WORD COUNT
004656  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
004664  001401                        BEQ     2$              ;;BR IF SAME
004666  104004                        ERROR   4                       ;;WORD COUNT REG. IN ERROR
                                
004670  006337  001124          2$:     ASL     $GDDAT                  ;CHANGE THE DATA
004674  001357                        BNE     1$                      ;BR IF MORE DATA TO LOAD
                                        ;;*******************************************************************
                                        ;*TEST 13       FLOAT A 1 ACROSS THE BUS ADDRESS REGISTER
                                        ;;*******************************************************************
004676  000004                  TST13:  SCOPE
004700  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
004706  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE
004714  012737  004722  001110          MOV     #1$,$LPERR              ;LOAD LOOP ADDRESS ON ERROR
                                
004722  012777  004000  175026  1$:     MOV     #CLRALL,@SFR            ;RESET THE DEVICE
004730  013777  001124  175016          MOV     $GDDAT,@BAR             ;LOAD BUS ADDRESS "A"
004736  017737  175012  001126          MOV     @BAR,$BDDAT             ;READ BUS ADDRESS
004744  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
004752  001401                        BEQ     2$              ;;BR IF SAME
004754  104005                        ERROR   5                       ;BUS ADDRESS REG. IN ERROR
                                
004756  006337  001124          2$:     ASL     $GDDAT                  ;CHANGE THE DATA
004762  001357                        BNE     1$                      ;BR IF MORE DATA TO LOAD
                                        ;;*******************************************************************
                                        ;*TEST 14       FLOAT A 1 ACROSS THE OFFSET REGISTER
                                        ;;*******************************************************************
004764  000004                  TST14:  SCOPE
004766  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
004774  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE
005002  012737  005010  001110          MOV     #1$,$LPERR              ;LOAD LOOP ADDRESS ON ERROR
                                
005010  012777  004000  174740  1$:     MOV     #CLRALL,@SFR            ;RESET THE DEVICE
005016  013777  001124  174724          MOV     $GDDAT,@OFF             ;LOAD OFFSET "A"
005024  017737  174720  001126          MOV     @OFF,$BDDAT             ;READ OFFSET
005032  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
005040  001401                        BEQ     2$              ;;BR IF SAME
005042  104006                        ERROR   6                       ;OFFSET REG. IN ERROR
                                
005044  006337  001124          2$:     ASL     $GDDAT                  ;CHANGE THE DATA
005050  001357                        BNE     1$                      ;BR IF MORE DATA TO LOAD
```

```
1047                                    ;;*****************************************************************
1048                                    ;*TEST 15       VERIFY NO DUAL REGISTER SELECTION
1049                                    ;;*****************************************************************
1050  005052  000004                    TST15:  SCOPE
1051  005054  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1052                                    ;LOAD DIFFERENT NUMBERS INTO THE  WCR, BAR AND OFF REGISTERS
1053  005062  012777  004000  174666            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1054  005070  012777  011111  174652            MOV     #11111,@OFF             ;LOAD OFFSET REGISTER
1055  005076  012777  022222  174646            MOV     #22222,@WCR             ;LOAD W.C. REGISTER
1056  005104  012777  033333  174642            MOV     #33333,@BAR             ;LOAD B.A. REGISTER
1057  005112  012777  000036  174636            MOV     #36,@SFR                ;LOAD SPECIAL FUNCTION REGISTER
1058  005120  012777  007636  174620            MOV     #7636,@CSR              ;LOAD COMMAND/STATUS REGISTER
1059                                    ;NOW READ EACH REGISTER AND CHECK THE VALUE
1060  005126  012737  011111  001124            MOV     #11111,$GDDAT           ;LOAD EXPECTED VALUE
1061  005134  017737  174610  001126            MOV     @OFF,$BDDAT             ;READ A OFFSET REG.
1062  005142  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1063  005150  001401                            BEQ     1$              ;;BR IF SAME
1064  005152  104007                            ERROR   7                       ;DUAL ADDRESS ERROR
1065  005154  012737  022222  001124  1$:       MOV     #22222,$GDDAT           ;LOAD EXPECTED VALUE
1066  005162  017737  174564  001126            MOV     @WCR,$BDDAT             ;READ W.C. REG.
1067  005170  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1068  005176  001401                            BEQ     2$              ;;BR IF SAME
1069  005200  104007                            ERROR   7                       ;DUAL ADDRESS ERROR
1070  005202  012737  033333  001124  2$:       MOV     #33333,$GDDAT           ;LOAD EXPECTED VLAUE
1071  005210  017737  174540  001126            MOV     @BAR,$BDDAT             ;READ B.A. REG.
1072  005216  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1073  005224  001401                            BEQ     3$              ;;BR IF SAME
1074  005226  104007                            ERROR   7                       ;DUAL ADDRESS ERROR
1075  005230  012737  000036  001124  3$:       MOV     #36,$GDDAT              ;LOAD EXPECTED VALUE
1076  005236  017737  174514  001126            MOV     @SFR,$BDDAT             ;READ SPECIAL FUNCTION REG
1077  005244  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1078  005252  001401                            BEQ     4$              ;;BR IF SAME
1079  005254  104007                            ERROR   7                       ;DUAL ADDRESS ERROR
1080  005256  012737  007636  001124  4$:       MOV     #7636,$GDDAT            ;LOAD EXPECTED VALUE
1081  005264  017737  174456  001126            MOV     @CSR,$BDDAT             ;READ COMMAND/STATUS REGISTER
1082  005272  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1083  005300  001401                            BEQ     5$              ;;BR IF SAME
1084  005302  104007                            ERROR   7                       ;DUAL ADDRESS ERROR
1085  005304  012777  004000  174444  5$:       MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
```

```
                                        ;;*********************************************************************
                                        ;*TEST 16       VERIFY "CLR ALL" CLEARS THE EXTENDED OFFSET BITS
                                        ;;*********************************************************************
005312 000004                           TST16:  SCOPE
005314 012737 000010 001160                     MOV     #10,$TIMES      ;;DO 10 ITERATIONS
005322 005037 001124                            CLR     $GDDAT                  ;CLEAR EXPECTED VALUE
005326 012777 004000 174422                     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
005334 012777 000003 174406                     MOV     #3,@OFF                 ;LOAD EXTENDED OFFSET REGISTER
005342 012777 004000 174406                     MOV     #CLRALL,@SFR            ;CLEAR THE EXTENDED OFFSET REG.'S
005350 017737 174374 001126                     MOV     @OFF,$BDDAT             ;READ EXTENDED OFFSET REG.
005356 001401                                   BEQ     TST17           ;;BR IF CLEARED
005360 104006                                   ERROR   6                       ;CLEAR ALL FAILED TO CLEAR EXTENDED OFFSET REGIS
                                        ;;*********************************************************************
                                        ;*TEST 17       TEST THE "ACTIVE" FLOP CAN SET AND CLEAR
                                        ;;*********************************************************************
005362 000004                           TST17:  SCOPE
005364 012777 004000 174364                     MOV     #CLRALL,@SFR                    ;CLEAR THE DEVICE
005372 052777 000010 174356                     BIS     #TSTDMA,@SFR    ;SET THE "TEST DMA" TO PREVNT DATA TRANSFERS
005400 000240                                   NOP
005402 000240                                   NOP
005404 000240                                   NOP
005406 052777 000022 174332                     BIS     #BIT4!BIT1,@CSR         ;SET "MATRIX MODE" AND "BYTE" MODE
                                        ;NOW SET THE "ACTIVE" FLOP AND VERIFY "INTERFACE IDLE" GOES LOW
005414 012737 000022 001124                     MOV     #BIT4!BIT1,$GDDAT       ;LOAD EXPECTED VALUE
005422 052777 000001 174316                     BIS     #BIT0,@CSR              ;SET "ACTIVE" TO A 1
005430 017737 174312 001126                     MOV     @CSR,$BDDAT             ;READ STATUS
005436 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
005444 001404                                   BEQ     1$              ;;BR IF EXPECTED
005446 052777 000400 174302                     BIS     #ENDDMA,@SFR            ;STOP DMA IF POSSIBLE
005454 104001                                   ERROR   1                       ;"ACTIVE" FLOP FAILED TO SET
                                        ;POKE THE "END DMA" SIGNAL AND VERIFY "ACTIVE" CLEARS
005456 052777 000400 174272             1$:     BIS     #ENDDMA,@SFR            ;SEND "END DMA" SIGNAL
005464 012737 000222 001124                     MOV     #BIT7!BIT4!BIT1,$GDDAT  ;SET "INTERFACE IDLE" INTO EXPECTED
005472 017737 174250 001126                     MOV     @CSR,$BDDAT             ;READ STATUS
005500 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE
005506 001401                                   BEQ     2$              ;;BR IF SAME
005510 104001                                   ERROR   1                       ;"END DMA" AGAIN FAILED TO CLEAR "ACTIVE" FLOP
                                        ;POKE "ENABLE DMA" AND THEN ISSUE "CLR ALL" SIGNAL TO
                                        ;       ENSURE THE "ACTIVE"  FLOP CLEARS
005512 012737 000200 001124             2$:     MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
005520 052777 000001 174220                     BIS     #BIT0,@CSR              ;POKE "ENABLE DMA"
005526 052777 004000 174222                     BIS     #CLRALL,@SFR            ;GENERATE "CLR ALL"
005534 017737 174206 001126                     MOV     @CSR,$BDDAT             ;READ STATUS
005542 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE
005550 001401                                   BEQ     TST20           ;;BR IF SAME
005552 104001                                   ERROR   1                       ;"CLR ALL" FAILED TO CLEAR "ACTIVE" FLOPS
```

```
1133                                    ;;*********************************************************************
1134                                    ;*TEST 20        VERIFY Z INPUTS CAUSE THE LOW 8 BITS OF 32 BIT COUNTER TO CHANGEIN MATRI
1135                                    ;;*********************************************************************
1136  005554  000004                   TST20:  SCOPE
1137  005556  012737  000100  001160           MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1138  005564  012777  004000  174164           MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1139  005572  012777  000010  174156           MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
1140  005600  012777  000000  174144           MOV     #0,@WCR                 ;LOAD W.C. REG
1141  005606  012777  000000  174140           MOV     #0,@BAR                 ;LOAD B.A.  REG
1142  005614  012777  000022  174124           MOV     #BIT4!BIT1,@CSR         ;ENTER MATRIX MODE
1143  005622  052777  000001  174116           BIS     #BIT0,@CSR              ;GO "ACTIVE"
1144  005630  052777  000002  174120           BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
1145  005636  042777  000002  174112           BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
1146  005644  012737  000001  001124           MOV     #1,$GDDAT               ;LOAD EXPECTED
1147  005652  017737  174076  001126           MOV     @BAR,$BDDAT             ;READ LOW 16 BITS
1148  005660  001002                           BNE     1$              ;;BR IF NON-ZERO
1149  005662  104010                           ERROR   10                      ;Z INPUT FAILED TO CAUSE THE LOW BYTE OF 32 BIT
1150  005664  000420                           BR      TST21           ;;
1151  005666  017737  174060  001126  1$:      MOV     @WCR,$BDDAT             ;READ HIGH 16 BITS
1152  005674  001401                           BEQ     2$              ;;BR IF CLEARED
1153  005676  104011                           ERROR   11                      ;HIGH 16 BIS CHANGED IN ERROR
1154  005700  012737  000020  001124  2$:      MOV     #BIT4,$GDDAT            ;LOAD EXPECTED
1155  005706  017737  174034  001126           MOV     @CSR,$BDDAT             ;READ STATUS
1156  005714  032737  040000  001126           BIT     #BIT14,$BDDAT           ;TEST IF "CELL OVERFLOW"
1157  005722  001401                           BEQ     TST21           ;;BR IF CORRECT
1158  005724  104012                           ERROR   12                      ;"CELL OVERFLOW" FLOP SET IN ERROR
1159                                    ;;*********************************************************************
1160                                    ;*TEST 21        VERIFY Z INPUTS CAUSE THE LOW 16 BITS OF 32 BIT COUNTER TO CHANGE
1161                                    ;;*********************************************************************
1162  005726  000004                   TST21:  SCOPE
1163  005730  012737  000100  001160           MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1164  005736  012777  004000  174012           MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1165  005744  012777  000010  174004           MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
1166  005752  012777  000000  173772           MOV     #0,@WCR                 ;LOAD W.C. REG
1167  005760  012777  000376  173766           MOV     #376,@BAR               ;LOAD B.A.  REG
1168  005766  012777  000022  173752           MOV     #BIT4!BIT1,@CSR         ;ENTER MATRIX MODE
1169  005774  052777  000001  173744           BIS     #BIT0,@CSR              ;GO "ACTIVE"
1170  006002  052777  000002  173746           BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
1171  006010  042777  000002  173740           BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
1172  006016  012737  000400  001124           MOV     #BIT8,$GDDAT            ;LOAD EXPECTED
1173  006024  017737  173724  001126           MOV     @BAR,$BDDAT             ;READ LOW 16 BITS
1174  006032  105737  001127                   TSTB    $BDDAT+1                ;TEST HIGH BYTE
1175  006036  001002                           BNE     1$                      ;BR IF NON-ZERO
1176  006040  104010                           ERROR   10                      ;Z INPUT FAILED TO CAUSE THE HIGH BYTE OF THE LO
1177  006042  000423                           BR      TST22           ;;
1178  006044  012737  000000  001124  1$:      MOV     #0,$GDDAT               ;LOAD EXPECTED
1179  006052  017737  173674  001126           MOV     @WCR,$BDDAT             ;READ HIGH 16 BITS
1180  006060  001401                           BEQ     2$              ;;BR IF CLEARED
1181  006062  104011                           ERROR   11                      ;HIGH 16 BITS CHANGED IN ERROR
1182  006064  012737  000020  001124  2$:      MOV     #BIT4,$GDDAT            ;LOAD EXPECTED
1183  006072  017737  173650  001126           MOV     @CSR,$BDDAT             ;READ STATUS
1184  006100  032737  040000  001126           BIT     #BIT14,$BDDAT           ;TEST IF "CELL OVERFLOW"
1185  006106  001401                           BEQ     TST22           ;;BR IF CORRECT
1186  006110  104012                           ERROR   12                      ;"CELL OVERFLOW" FLOP SET IN ERROR
```

```
1187                                    ;;*****************************************************************
1188                                    ;*TEST 22       VERIFY Z INPUTS CAUSE THE LOW 24 BITS OF 32 BIT COUNTER TO CHANGE
1189                                    ;;*****************************************************************
1190  006112  000004                    TST22:  SCOPE
1191  006114  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1192  006122  012777  004000  173626            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1193  006130  012777  000010  173620            MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
1194  006136  012777  000000  173606            MOV     #0,@WCR                 ;LOAD W.C. REG
1195  006144  012777  177776  173602            MOV     #-2,@BAR                ;LOAD B.A.  REG
1196  006152  012777  000022  173566            MOV     #BIT4!BIT1,@CSR         ;ENTER MATRIX MODE
1197  006160  052777  000001  173560            BIS     #BIT0,@CSR              ;GO ''ACTIVE''
1198  006166  052777  000002  173562            BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
1199  006174  042777  000002  173554            BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
1200  006202  012737  000001  001124            MOV     #1,$GDDAT               ;LOAD EXPECTED
1201  006210  017737  173536  001126  1$:       MOV     @WCR,$BDDAT             ;READ HIGH 16 BITS
1202  006216  001002                            BNE     2$                      ;BR IF SET
1203  006220  104011                            ERROR   11                      ;LOW BYTE OF THE HIGH 16 BITS FAILED TO CHANGE
1204  006222  000413                            BR      TST23           ;;
1205  006224  012737  000020  001124  2$:       MOV     #BIT4,$GDDAT            ;LOAD EXPECTED
1206  006232  017737  173510  001126            MOV     @CSR,$BDDAT             ;READ STATUS
1207  006240  032737  040000  001126            BIT     #BIT14,$BDDAT           ;TEST IF ''CELL OVERFLOW''
1208  006246  001401                            BEQ     TST23           ;;BR IF CORRECT
1209  006250  104012                            ERROR   12                      ;''CELL OVERFLOW'' FLOP SET IN ERROR
1210                                    ;;*****************************************************************
1211                                    ;*TEST 23       VERIFY Z INPUTS CAUSE THE HIGH 8 BITS OF THE 32 BIT COUNTER TO CHANGE
1212                                    ;;*****************************************************************
1213  006252  000004                    TST23:  SCOPE
1214  006254  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1215  006262  012777  004000  173466            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1216  006270  012777  000010  173460            MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
1217  006276  012777  000377  173446            MOV     #377,@WCR               ;LOAD W.C. REG
1218  006304  012777  177776  173442            MOV     #-2,@BAR                ;LOAD B.A.  REG
1219  006312  012777  000022  173426            MOV     #BIT4!BIT1,@CSR         ;ENTER MATRIX MODE
1220  006320  052777  000001  173420            BIS     #BIT0,@CSR              ;GO ''ACTIVE''
1221  006326  052777  000002  173422            BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
1222  006334  042777  000002  173414            BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
1223  006342  012737  000400  001124            MOV     #BIT8,$GDDAT            ;LOAD EXPECTED
1224  006350  017737  173376  001126  1$:       MOV     @WCR,$BDDAT             ;READ HIGH 16 BITS
1225  006356  105737  001127                    TSTB    $BDDAT+1                ;TEST HIGH BYTE
1226  006362  001001                            BNE     TST24           ;;BR IF SET
1227  006364  104012                            ERROR   12              ;HIGH 8 BITS OF THE 32 BIT COUNTER FAILED TO CHANGE
```

```
                                        ;;*****************************************************************
                                        ;*TEST 24       VERIFY Z INPUTS DO NOT CAUSE THE LOW 8 BITS OF 32 BIT COUNTER TO CHANGE
                                        ;;*****************************************************************
006366 000004                   TST24:  SCOPE
006370 012737 000100 001160             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
006376 012777 004000 173352             MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
006404 012777 000000 173340             MOV     #0,@WCR                 ;LOAD W.C. REG
006412 012777 000000 173334             MOV     #0,@BAR                 ;LOAD B.A.  REG
006420 005077 173322                    CLR     @CSR                    ;ENSURE LIST MODE
006424 052777 000002 173324             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z' PULSES
006432 042777 000002 173316             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z' PULSES
006440 012737 000000 001124             MOV     #0,$GDDAT               ;LOAD EXPECTED
006446 017737 173302 001126             MOV     @BAR,$BDDAT             ;READ LOW 16 BITS
006454 001401                           BEQ     TST25           ;;BR IF ZERO
006456 104010                           ERROR   10              ;Z INPUT CAUSE THE 32 BIT COUNTER TO CHANGE IN LIST MODE

                                        ;;*****************************************************************
                                        ;*TEST 25       TEST THAT 'WCA OVFL' SETS Z/WC OVERFLOW FLOP AND ''CLR ALL'' CLEARS IT
                                        ;;*****************************************************************
006460 000004                   TST25:  SCOPE
006462 012737 000010 001160             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
006470 012777 004000 173260             MOV     #CLRALL,@SFR            ;CLEAR DEVICE
006476 012777 000010 173252             MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
006504 012777 177777 173240             MOV     #-1,@WCR                ;LOAD Z COUNTER
006512 012777 177776 173234             MOV     #-2,@BAR                ;LOAD Z COUNTER
006520 012777 000022 173220             MOV     #BIT4!BIT1,@CSR         ;ENTER 'MATRIX' MODE
006526 052777 000001 173212             BIS     #BIT0,@CSR              ;ENABLE NCV11
                                ;NOW ENABLE 'TEST Z' INPUTS
006534 052777 000002 173214             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z' PULSES
006542 042777 000002 173206             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z' PULSES
006550 013700 002006                    MOV     CPUDL0,R0               ;LOAD GROSS TIMER
006554 005001                           CLR     R1
006556 012737 040020 001124             MOV     #BIT14!BIT4,$GDDAT      ;LOAD EXPECTED STATUS

006564 032777 040000 173154     1$:     BIT     #BIT14,@CSR             ;TEST FOR Z/WC OVERFLOW
006572 001011                           BNE     2$                      ;BR IF SET
006574 005301                           DEC     R1                      ;DELAY
006576 001372                           BNE     1$
006600 005300                           DEC     R0                      ;DELAY
006602 001370                           BNE     1$
006604 012777 000400 173144             MOV     #ENDDMA,@SFR            ;STOP DMA TRANSFERS
006612 104012                           ERROR   12              ;AFTER A GROSS TIME, THE Z/WC OVERFLOW FLOP FAILED TO SE
006614 000 16                           BR      TST26           ;;

                                ;NOW GENERATE ''CLR ALL'' TO CLEAR Z/WC OVERFLOW BIT
006616 052777 004000 173132     2$:     BIS     #CLRALL,@SFR            ;CLEAR Z/WC FLOP
006624 012737 000200 001124             MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
006632 017737 173110 001126             MOV     @CSR,$BDDAT             ;READ STATUS
006640 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE
006646 001401                           BEQ     TST26           ;;BR IF SAME
006650 104012                           ERROR   12                      ;''CLR ALL'' FAILED TO CLEAR ''Z/WC'' FLOP
```

```
1279                                    ;;*********************************************************************
1280                                    ;*TEST 26       TEST THAT 'WCA OVFL' SETS Z/WC OVERFLOW FLOP AND ''CLR WC OVFL'' CLEARS IT
1281                                    ;;*********************************************************************
1282  006652  000004                    TST26:  SCOPE
1283  006654  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1284  006662  012777  004000  173066            MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1285  006670  012777  000010  173060            MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
1286  006676  012777  177777  173046            MOV     #-1,@WCR                ;LOAD Z COUNTER
1287  006704  012777  177776  173042            MOV     #-2,@BAR                ;LOAD Z COUNTER
1288  006712  012777  000022  173026            MOV     #BIT4!BIT1,@CSR         ;ENABLE 'MATRIX'' MODE
1289  006720  052777  000001  173020            BIS     #BIT0,@CSR              ;SET NCV11 ACTIVE
1290                                    ;NOW ENABLE 'TEST Z'' INPUTS
1291  006726  052777  000002  173022            BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1292  006734  042777  000002  173014            BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1293
1294  006742  013700  002006                    MOV     CPUDL0,R0               ;LOAD GROSS TIMER
1295  006746  005001                            CLR     R1
1296  006750  012737  040020  001124            MOV     #BIT14!BIT4,$GDDAT      ;LOAD EXPECTED STATUS
1297
1298  006756  032777  040000  172762  1$:       BIT     #BIT14,@CSR             ;TEST FOR Z/WC OVERFLOW
1299  006764  001014                            BNE     2$                      ;BR IF SET
1300  006766  005301                            DEC     R1                      ;DELAY
1301  006770  001372                            BNE     1$
1302  006772  005300                            DEC     R0                      ;DELAY
1303  006774  001370                            BNE     1$
1304  006776  012777  000400  172752            MOV     #ENDDMA,@SFR            ;STOP DAM TRANSFERS
1305  007004  042777  000002  172744            BIC     #TESTZ,@SFR             ;STOP ''Z'' INPUTS
1306  007012  104012                            ERROR   12      ;AFTER A GROSS TIME, THE Z/WC OVERFLOW FLOP FAILED TO SET
1307  007014  000416                            BR      TST27           ;;
1308
1309                                    ;NOW GENERATE ''CLR WC OVFL'' TO CLEAR Z/WC OVERFLOW BIT
1310  007016  052777  040000  172732  2$:       BIS     #CLRWCO,@SFR            ;CLEAR Z/WC FLOP
1311  007024  012737  000022  001124            MOV     #BIT4!BIT1,$GDDAT       ;LOAD EXPECTED
1312  007032  017737  172710  001126            MOV     @CSR,$BDDAT             ;READ STATUS
1313  007040  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1314  007046  001401                            BEQ     TST27           ;;BR IF SAME
1315  007050  104012                            ERROR   12                      ;''CLR WC OVFL'' FAILED TO CLEAR ''Z/WC'' FLOP
```

```
1316                                    ;;*****************************************************************
1317                                    ;*TEST 27       TEST THAT 'WCA OVFL'' GENERATES AN INTERRUPT
1318                                    ;;*****************************************************************
1319  007052  000004                    TST27:  SCOPE
1320  007054  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1321  007062  012777  004000  172666            MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1322  007070  012777  000010  172660            MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
1323  007076  012777  177777  172646            MOV     #-1,@WCR                ;LOAD Z COUNTER
1324  007104  012777  177776  172642            MOV     #-2,@BAR                ;LOAD Z COUNTER
1325  007112  012777  000022  172626            MOV     #BIT4!BIT1,@CSR         ;SET MATRIX MODE
1326  007120  052777  000001  172620            BIS     #BIT0,@CSR              ;SET NCV11 ACTIVE
1327                                    ;NOW ENABLE 'TEST Z'' INPUTS
1328  007126  052777  000002  172622            BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1329  007134  042777  000002  172614            BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1330  007142  013700  002006                    MOV     CPUDL0,R0               ;LOAD GROSS TIMER
1331  007146  005001                            CLR     R1
1332  007150  012737  040020  001124            MOV     #BIT14.BIT4,$GDDAT      ;LOAD EXPECTED STATUS
1333
1334  007156  032777  040000  172562  1$:       BIT     #BIT14,@CSR             ;TEST FOR Z/WC OVERFLOW
1335  007164  001014                            BNE     2$                      ;BR IF SET
1336  007166  005301                            DEC     R1                      ;DELAY
1337  007170  001372                            BNE     1$
1338  007172  005300                            DEC     R0                      ;DELAY
1339  007174  001370                            BNE     1$
1340  007176  012777  000400  172552            MOV     #ENDDMA,@SFR            ;STOP DMA TRANSFERS
1341  007204  042777  000002  172544            BIC     #TESTZ,@SFR             ;STOP 'Z' INPUTS
1342  007212  104012                            ERROR   12      ;AFTER A GROSS TIME, THE Z/WC OVERFLOW FLOP FAILED TO SET
1343  007214  000430                            BR      5$              ;;BR TO CLEAN UP
1344
1345                                    ;NOW ENABLE THE WC/Z OVERFLOW INTERRUPT BIT AND WAIT FOR AN INTERRUPT
1346  007216  012746  000000            2$:     MOV     #0,-(SP)
1347  007222  012746  007230                    MOV     #3$,-(SP)               ;LSI-11 HACK
1348  007226  000002                            RTI
1349  007230  012777  007274  172534  3$:       MOV     #4$,@VECTA0             ;LOAD INTR. VECTOR
1350  007236  052777  000100  172502            BIS     #BIT6,@CSR              ;ENABLE INTERRUPT
1351  007244  000240                            NOP
1352  007246  000240                            NOP
1353  007250  000240                            NOP
1354  007252  000240                            NOP
1355  007254  017737  172466  001126            MOV     @CSR,$BDDAT             ;READ STATUS
1356  007262  012777  004000  172466            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1357  007270  104013                            ERROR   13                      ;WC/Z OVERFLOW FAILED TO GENERATE INTERRUPT
1358  007272  000401                            BR      5$              ;;BR TO CLEAN UP
1359
1360  007274  022626                    4$:     CMP     (SP)+,(SP)+
1361  007276  005077  172444            5$:     CLR     @CSR
1362  007302  012777  004000  172446            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1363  007310  013777  001774  172454            MOV     VECTA1,@VECTA0
1364  007316  005077  172452                    CLR     @VECTA1
1365
```

```
                                ;;*****************************************************************
                                ;*TEST 30        VERIFY 'WCA OVFL' CLEARS 'ACTIVE'
                                ;;*****************************************************************
007322 000004                   TST30:  SCOPE
007324 012737 000010 001160             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
007332 012777 004000 172416             MOV     #CLRALL,@SFR            ;CLEAR DEVICE
007340 012777 000014 172410             MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA
007346 012777 177777 172376             MOV     #-1,@WCR                ;LOAD W.C. REG.
007354 012777 177776 172372             MOV     #-2,@BAR                ;LOAD BAR REG.
007362 005077 172362                    CLR     @OFF                    ;CLEAR OFFSET REG.
007366 012777 000022 172352             MOV     #BIT4!BIT1,@CSR         ;ENABLE MATRIX MODE
007374 052777 000001 172344             BIS     #BIT0,@CSR              ;SET 'ACTIVE'
007402 052777 000002 172346             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z' PULSES
007410 042777 000002 172340             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z' PULSES
007416 017737 172324 001126             MOV     @CSR,$BDDAT             ;READ STATUS
007424 012737 040022 001124             MOV     #BIT14!BIT4!BIT1,$GDDAT ;LOAD EXPECTED
007432 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE
007440 001402                           BEQ     1$              ;;BR IF SAME
007442 104012                           ERROR   12                      ;'WCA OVFL' FAILED TO CLEAR 'ACTIVE'
007444 000407                           BR      TST31           ;;
                                
007446 005037 001124            1$:     CLR     $GDDAT                  ;CLEAR THE EXPECTED
007452 017737 172272 001126             MOV     @OFF,$BDDAT             ;READ THE ACTUAL
007460 001401                           BEQ     TST31           ;;BR IF CLEARED
007462 104012                           ERROR   12                      ;'WCA INC X OFF' IN MATRIX MODE CHANGED
                                                                ;THE OFFSET REGISTER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 31       VERIFY JOYSTICK DONE FLOP SETS
                                        ;;*****************************************************************
007464  000004                  TST31:  SCOPE
007466  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
007474  012777  004000  172254          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
007502  052777  000001  172246          BIS     #REDJOY,@SFR            ;REQUEST JOYSTICK DATA
007510  013700  002010                  MOV     CPUDL1,R0               ;LOAD DELAY COUNTER
007514  105777  172236          1$:     TSTB    @SFR                    ;WAIT FOR JOYSTICK READY
007520  100411                          BMI     2$                      ;BR IF SET
007522  005300                          DEC     R0                      ;DELAY
007524  001373                          BNE     1$                      ;BR IF NOT EXHAUSTED
007526  017737  172224  001126          MOV     @SFR,$BDDAT             ;READ STATUS
007534  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
007542  104014                          ERROR   14                      ;''JOYSTICK READY'' FAILED TO SET
                                        ;NOW PERFORM A 'TST'' INSTRUCTION TO THE JOYSTICK REGISTER
                                        ;       THE ACCESS OF THIS BUS ADDRESS SHOULD CLEAR JOY READY FLOP''
007544  005777  172212          2$:     TST     @JOY                    ;ADDRESS THE REGISTER
007550  017737  172202  001126          MOV     @SFR,$BDDAT             ;READ STATUS
007556  105737  001126                  TSTB    $BDDAT                  ;TEST IF THE BIT CLEARED
007562  100003                          BPL     TST32           ;;BR IF YES
007564  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
007570  104014                          ERROR   14                      ;ADDRESSING THE JOYSTICK ADDRESS FAILED TO CLEAR
                                        ;;*****************************************************************
                                        ;*TEST 32       VERIFY THAT 'RESET'' INSTRUCTION CLEARS THE JOY READY FLOP
                                        ;;*****************************************************************
007572  000004                  TST32:  SCOPE
007574  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
007602  012777  004000  172146          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
007610  052777  000001  172140          BIS     #REDJOY,@SFR            ;REQUEST JOYSTICK DATA
007616  013700  002010                  MOV     CPUDL1,R0               ;LOAD DELAY
007622  105777  172130          1$:     TSTB    @SFR                    ;WAIT FOR JOYSTICK READY
007626  100411                          BMI     2$                      ;BR IF SET
007630  005300                          DEC     R0                      ;DELAY
007632  001373                          BNE     1$                      ;BR IF NOT EXHAUSTED
007634  017737  172116  001126          MOV     @SFR,$BDDAT             ;READ STATUS
007642  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED VALUE
007650  104014                          ERROR   14                      ;JOY READY FAILED TO SET
                                        ;NOW ISSUE A ''CLR ALL'' AND VERIFY THE ''JOY READY'' DOES NOT CLEAR
007652  052777  004000  172076  2$:     BIS     #CLRALL,@SFR            ;GENERATE ''CLR ALL''
007660  017737  172072  001126          MOV     @SFR,$BDDAT             ;READ STATUS
007666  012737  000200  001124          MOV     #BIT7,$GDDAT    ;LOAD EXPECTED VALUE
007674  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
007702  001401                          BEQ     3$              ;;BR IF SAME
007704  104014                          ERROR   14                      ;''CLR ALL'' CLEARED THE ''JOY READY'' FLOP IN ERROR
                                        ;NOW ISSUE A BUS ''RESET'' AND VERIFY THE JOY READY'' CLEARS
007706  000005                  3$:     RESET                           ;BUS ''INIT''
007710  017737  172042  001126          MOV     @SFR,$BDDAT             ;READ STATUS
007716  012737  000000  001124          MOV     #0,$GDDAT               ;LOAD EXPECTED
007724  017737  172026  001126          MOV     @SFR,$BDDAT             ;READ STATUS
007732  001401                          BEQ     TST33           ;;BR IF CLEARED
007734  104014                          ERROR   14                      ;BUS INIT FAILED TO CLEAR JOY READY
```

```
1444
1445                                    ;;********************* ***************************************************
1446                                    ;*TEST 33        JOYSTICK DATA PATH  =  GAIN =0  ZB ENABLE -0  RES. = 000
1447                                    ;;*****************************************************************************
1448  007736  000004                    TST33:  SCOPE
1449  007740  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1450  007746  012777  004000  172002            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1451  007754  012777  000014  171774            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1452  007762  012777  000000  171756            MOV     #0,@CSR         ;LOAD CSR REGISTER
1453  007770  052777  000001  171760            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1454  007776  105777  171754          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1455  010002  100375                            BPL     1$
1456  010004  017737  171752  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1457  010012  012737  000000  001124            MOV     #0,$GDDAT               ;LOAD EXPECTED
1458  010020  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1459  010026  001401                            BEQ     TST34           ;;BR IF SAME
1460  010030  104015                            ERROR   15                      ;JOYSTICK DATA PATH BIT SET
1461
1462                                    ;;*****************************************************************************
1463                                    ;*TEST 34        JOYSTICK DATA PATH  =  GAIN =1
1464                                    ;;*****************************************************************************
1465  010032  000004                    TST34:  SCOPE
1466  010034  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1467  010042  012777  004000  171706            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1468  010050  012777  000014  171700            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1469  010056  012777  002000  171662            MOV     #2000,@CSR              ;LOAD CSR REGISTER
1470  010064  052777  000001  171664            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1471  010072  105777  171660          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1472  010076  100375                            BPL     1$
1473  010100  017737  171656  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1474  010106  012737  124250  001124            MOV     #124250,$GDDAT          ;LOAD EXPECTED
1475  010114  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1476  010122  001401                            BEQ     TST35           ;;BR IF SAME
1477  010124  104015                            ERROR   15                      ;JOYSTICK DATA PATH ERROR
1478
1479                                    ;;*****************************************************************************
1480                                    ;*TEST 35        JOYSTICK DATA PATH  =  ZB ENABLF =1
1481                                    ;;*****************************************************************************
1482  010126  000004                    TST35:  SCOPE
1483  010130  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1484  010136  012777  004000  171612            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1485  010144  012777  000014  171604            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1486  010152  012777  004000  171566            MOV     #4000,@CSR              ;LOAD CSR REGISTER
1487  010160  052777  000001  171570            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1488  010166  105777  171564          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1489  010172  100375                            BPL     1$
1490  010174  017737  171562  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1491  010202  012737  050120  001124            MOV     #050120,$GDDAT          ;LOAD EXPECTED
1492  010210  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1493  010216  001401                            BEQ     TST36           ;;BR IF SAME
1494  010220  104015                            ERROR   15                      ;JOYSTICK DATA PATH ERROR
```

```
1495                                    ;;*****************************************************************
1496                                    ;*TEST 36       JOYSTICK DATA PATH  RES. = 001
1497                                    ;;*****************************************************************
1498  010222  000004                    TST36:  SCOPE
1499  010224  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1500  010232  012777  004000  171516            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1501  010240  012777  000014  171510            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1502  010246  012777  000023  171472            MOV     #23,@CSR                ;LOAD CSR REGISTER
1503  010254  052777  000001  171474            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1504  010262  105777  171470          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1505  010266  100375                            BPL     1$
1506  010270  017737  171466  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1507  010276  012737  000401  001124            MOV     #401,$GDDAT             ;LOAD EXPECTED
1508  010304  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1509  010312  001401                            BEQ     TST37           ;;BR IF SAME
1510  010314  104015                            ERROR   15                      ;JOYSTICK DATA PATH ERROR  RES. = 001
1511
1512                                    ;;*****************************************************************
1513                                    ;*TEST 37       JOYSTICK DATA PATH  RES. = 010
1514                                    ;;*****************************************************************
1515  010316  000004                    TST37:  SCOPE
1516  010320  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1517  010326  012777  004000  171422            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1518  010334  012777  000014  171414            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1519  010342  012777  000025  171376            MOV     #25,@CSR                ;LOAD CSR REGISTER
1520  010350  052777  000001  171400            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1521  010356  105777  171374          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1522  010362  100375                            BPL     1$
1523  010364  017737  171372  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1524  010372  012737  001002  001124            MOV     #1002,$GDDAT            ;LOAD EXPECTED
1525  010400  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1526  010406  001401                            BEQ     TST40           ;;BR IF SAME
1527  010410  104015                            ERROR   15                      ;JOYSTICK DATA PATH ERROR  RES. = 010
1528
1529                                    ;;*****************************************************************
1530                                    ;*TEST 40       JOYSTICK DATA PATH  RES. = 100
1531                                    ;;*****************************************************************
1532  010412  000004                    TST40:  SCOPE
1533  010414  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1534  010422  012777  004000  171326            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1535  010430  012777  000014  171320            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1536  010436  012777  000031  171302            MOV     #31,@CSR                ;LOAD CSR REGISTER
1537  010444  052777  000001  171304            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1538  010452  105777  171300          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1539  010456  100375                            BPL     1$
1540  010460  017737  171276  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1541  010466  012737  002004  001124            MOV     #2004,$GDDAT            ;LOAD EXPECTED
1542  010474  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1543  010502  001401                            BEQ     TST41           ;;BR IF SAME
1544  010504  104015                            ERROR   15                      ;JOY STICK DATA PATH ERROR  RES. = 100
```

```
1545                                    ;;*****************************************************************
1546                                    ;*TEST 41       VERIFY THE DATA INCREMENT FUNCTION
1547                                    ;;*****************************************************************
1548  010506  000004                    TST41:  SCOPE
1549  010510  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1550  010516  012777  004000  171232            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1551  010524  012777  000014  171224            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1552  010532  012777  000421  171206            MOV     #421,@CSR               ;LOAD CSR REGISTER
1553  010540  052777  000001  171210            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1554  010546  105777  171204          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1555  010552  100375                            BPL     1$
1556  010554  017737  171202  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1557  010562  012737  000401  001124            MOV     #401,$GDDAT             ;LOAD EXPECTED
1558  010570  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1559  010576  001401                            BEQ     TST42           ;;BR IF SAME
1560  010600  104016                            ERROR   16                      ;MAINT. CAM01 FAILED TO INCREMENT DATA REGISTER
1561
1562                                    ;;*****************************************************************
1563                                    ;*TEST 42       VERIFY THE DATA INCREMENT CARRY BIT
1564                                    ;;*****************************************************************
1565  010602  000004                    TST42:  SCOPE
1566  010604  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1567  010612  012777  004000  171136            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1568  010620  012777  000014  171130            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1569  010626  012777  002437  171112            MOV     #2437,@CSR              ;LOAD CSR REGISTER
1570  010634  052777  000001  171114            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1571  010642  105777  171110          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1572  010646  100375                            BPL     1$
1573  010650  017737  171106  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1574  010656  012737  130260  001124            MOV     #130260,$GDDAT          ;LOAD EXPECTED
1575  010664  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1576  010672  001401                            BEQ     TST43           ;;BR IF SAME
1577  010674  104016                            ERROR   16                      ;MAINT. CAM01 WITH RES.=7, G-1, ZB =0
1578                                                                    ;FAILED TO CAUSE DATA INCREMENT CARRY PROPERLY
1579
1580                                    ;;*****************************************************************
1581                                    ;*TEST 43       VERIFY THE DATA INCREMENT FUNCTION IS INHIBITED
1582                                    ;;*****************************************************************
1583  010676  000004                    TST43:  SCOPE
1584  010700  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1585  010706  012777  004000  171042            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1586  010714  012777  000014  171034            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1587  010722  012777  006437  171016            MOV     #6437,@CSR              ;LOAD CSR REGISTER
1588  010730  052777  000001  171020            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1589  010736  105777  171014          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1590  010742  100375                            BPL     1$
1591  010744  017737  171012  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1592  010752  012737  177777  001124            MOV     #177777,$GDDAT          ;LOAD EXPECTED
1593  010760  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1594  010766  001401                            BEQ     TST44           ;;BR IS SAME
1595  010770  104016                            ERROR   16                      ;FAILED TO INHIBIT DATA INCREMENT FUNCTION
1596                                                                    ;IF THE BAD DATA WAS 0
```

```
1597                                    ;;*****************************************************************
1598                                    ;*TEST 44       VERIFY THE DATA DECREMENT FUNCTION
1599                                    ;;*****************************************************************
1600  010772  000004                    TST44:  SCOPE
1601  010774  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1602  011002  012777  004000  170746            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1603  011010  012777  000014  170740            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1604  011016  012777  001023  170722            MOV     #1023,@CSR              ;LOAD CSR REGISTER
1605  011024  052777  000001  170724            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1606  011032  105777  170720          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1607  011036  100375                            BPL     1$
1608  011040  017737  170716  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1609  011046  012737  000000  001124            MOV     #0,$GDDAT               ;LOAD EXPECTED
1610  011054  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1611  011062  001401                            BEQ     TST45           ;;BR IF SAME
1612  011064  104017                            ERROR   17              ;IF DATA WAS 401, THE DATA DECREMENT FUNCTION FAILED
1613                                                                    ;OTHERWISE DECREMENTED DATA ERROR
1614
1615                                    ;;*****************************************************************
1616                                    ;*TEST 45       VERIFY THE DATA DECREMENT BORROW
1617                                    ;;*****************************************************************
1618  011066  000004                    TST45:  SCOPE
1619  011070  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1620  011076  012777  004000  170652            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1621  011104  012777  000014  170644            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1622  011112  012777  007037  170626            MOV     #7037,@CSR              ;LOAD CSR REGISTER
1623  011120  052777  000001  170630            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1624  011126  105777  170624          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1625  011132  100375                            BPL     1$
1626  011134  017737  170622  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1627  011142  012737  177376  001124            MOV     #177376,$GDDAT          ;LOAD EXPECTED
1628  011150  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1629  011156  001401                            BEQ     TST46           ;;BR IF SAME
1630  011160  104017                            ERROR   17                      ;DECREMENTED DATA ERROR
1631
1632                                    ;;*****************************************************************
1633                                    ;*TEST 46       VERIFY THE DATA DECREMENT FUNCTION IS INHIBITED
1634                                    ;;*****************************************************************
1635  011162  000004                    TST46:  SCOPE
1636  011164  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1637  011172  012777  004000  170556            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
1638  011200  012777  000014  170550            MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROLLER FLOP
1639  011206  012777  001021  170532            MOV     #1021,@CSR              ;LOAD CSR REGISTER
1640  011214  052777  000001  170534            BIS     #REDJOY,@SFR            ;REQUEST JOYSTICE DATA
1641  011222  105777  170530          1$:       TSTB    @SFR                    ;WAIT FOR JOYSTICE READY
1642  011226  100375                            BPL     1$
1643  011230  017737  170526  001126            MOV     @JOY,$BDDAT             ;READ THE REGISTER
1644  011236  012737  000000  001124            MOV     #0,$GDDAT               ;LOAD EXPECTED
1645  011244  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1646  011252  001401                            BEQ     TST47           ;;BR IF SAME
1647  011254  104017                            ERROR   17                      ;INHIBIT DECREMENT FUNCTION ERROR
1648                                                                    ;IF DATA WAS 177777
```

```
1649                                    ;;*****************************************************************
1650                                    ;*TEST 47        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 7 GAIN - 0 ZB ENABLE = C
1651                                    ;;*****************************************************************
1652  011256  000004                    TST47:  SCOPE
1653  011260  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1654  011266  012777  004000  170462            MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1655  011274  012777  000014  170454            MOV     #TSTDMA!TSTCON,@SFR     ;SET ''TEST DMA AND CONTROL''
1656  011302  012777  000036  170436            MOV     #36,@CSR                ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1657  011310  052777  000002  170440            BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
1658  011316  052777  000001  170422            BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1659  011324  042777  000002  170424            BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
1660  011332  017737  170422  001126            MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1661  011340  012737  003407  001124            MOV     #3407,$GDDAT            ;LOAD EXPECTED VALUE
1662  011346  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1663  011354  001401                            BEQ     TST50           ;;BR IF SAME
1664  011356  104020                            ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1665                                    ;                       RESOLUTION = 7  GAIN = 0  ZB ENABLE = 0
1666                                    ;;*****************************************************************
1667                                    ;*TEST 50        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 7 GAIN = 1 ZB ENABLE = 0
1668                                    ;;*****************************************************************
1669  011360  000004                    TST50:  SCOPE
1670  011362  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1671  011370  012777  004000  170360            MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1672  011376  012777  000014  170352            MOV     #TSTDMA!TSTCON,@SFR     ;SET ''TEST DMA AND CONTROL''
1673  011404  012777  002036  170334            MOV     #2036,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1674  011412  052777  000002  170336            BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
1675  011420  052777  000001  170320            BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1676  011426  042777  000002  170322            BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
1677  011434  017737  170320  001126            MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1678  011442  012737  127657  001124            MOV     #127657,$GDDAT          ;LOAD EXPECTED VALUE
1679  011450  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1680  011456  001401                            BEQ     TST51           ;;BR IF SAME
1681  011460  104020                            ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1682                                    ;                       RESOLUTION = 7  GAIN = 1  ZB ENABLE = 0
1683                                    ;;*****************************************************************
1684                                    ;*TEST 51        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 7 GAIN = 0 ZB ENABLE = 1
1685                                    ;;*****************************************************************
1686  011462  000004                    TST51:  SCOPE
1687  011464  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1688  011472  012777  004000  170256            MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1689  011500  012777  000014  170250            MOV     #TSTDMA!TSTCON,@SFR     ;SET ''TEST DMA AND CONTROL''
1690  011506  012777  004036  170232            MOV     #4036,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1691  011514  052777  000002  170234            BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
1692  011522  052777  000001  170216            BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1693  011530  042777  000002  170220            BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
1694  011536  017737  170216  001126            MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1695  011544  012737  053527  001124            MOV     #053527,$GDDAT          ;LOAD EXPECTED VALUE
1696  011552  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1697  011560  001401                            BEQ     TST52           ;;BR IF SAME
1698  011562  104020                            ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1699                                    ;                       RESOLUTION = 7  GAIN = 0  ZB ENABLE = 1
```

```
                                        ;;*****************************************************************
                                        ;*TEST 52        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 6 GAIN = 0 ZB ENABLE = 0
                                        ;;*****************************************************************
011564 000004                   TST52:  SCOPE
011566 012737 000010 001160             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
011574 012777 004000 170154             MOV     #CLRALL,@SFR            ;CLEAR DEVICE
011602 012777 000014 170146             MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
011610 012777 000034 170130             MOV     #34,@CSR                ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
011616 052777 000002 170132             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
011624 052777 000001 170114             BIS     #BIT0,@CSR              ;SET ENABLE NCV11
011632 042777 000002 170116             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
011640 017737 170114 001126             MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
011646 012737 001406 001124             MOV     #1406,$GDDAT            ;LOAD EXPECTED VALUE
011654 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
011662 001401                           BEQ     TST53           ;;BR IF SAME
011664 104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                        ;                       RESOLUTION = 6  GAIN = 0  ZB ENABLE = 0
                                        ;;*****************************************************************
                                        ;*TEST 53        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 6 GAIN = 1 ZB ENABLE = 0
                                        ;;*****************************************************************
011666 000004                   TST53:  SCOPE
011670 012737 000010 001160             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
011676 012777 004000 170052             MOV     #CLRALL,@SFR            ;CLEAR DEVICE
011704 012777 000014 170044             MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
011712 012777 002034 170026             MOV     #2034,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
011720 052777 000002 170030             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
011726 052777 000001 170012             BIS     #BIT0,@CSR              ;SET ENABLE NCV11
011734 042777 000002 170014             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
011742 017737 170012 001126             MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
011750 012737 053656 001124             MOV     #53656,$GDDAT           ;LOAD EXPECTED VALUE
011756 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
011764 001401                           BEQ     TST54           ;;BR IF SAME
011766 104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                        ;                       RESOLUTION = 6  GAIN = 1  ZB ENABLE = 0
                                        ;;*****************************************************************
                                        ;*TEST 54        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 6 GAIN = 0 ZB ENABLE = 1
                                        ;;*****************************************************************
011770 000004                   TST54:  SCOPE
011772 012737 000010 001160             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
012000 012777 004000 167750             MOV     #CLRALL,@SFR            ;CLEAR DEVICE
012006 012777 000014 167742             MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
012014 012777 004034 167724             MOV     #4034,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
012022 052777 000002 167726             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
012030 052777 000001 167710             BIS     #BIT0,@CSR              ;SET ENABLE NCV11
012036 042777 000002 167712             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
012044 017737 167710 001126             MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
012052 012737 125526 001124             MOV     #125526,$GDDAT          ;LOAD EXPECTED VALUE
012060 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
012066 001401                           BEQ     TST55           ;;BR IF SAME
012070 104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                        ;                       RESOLUTION = 6  GAIN = 0  ZB ENABLE = 1
```

```
                                        ;;*****************************************************************
                                        ;*TEST 55        TEST ADDRESS MAKER  -  MATRIX MODE - RES   5 GAIN - 0 ZB ENABLE = 0
                                        ;;*****************************************************************
012072 000004                           TST55:  SCOPE
012074 012737 000010 001160                     MOV     #10,$TIMES      ;;DO 10 ITERATIONS
012102 012777 004000 167646                     MOV     #CLRALL,@SFR            ;CLEAR DEVICE
012110 012777 000014 167640                     MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
012116 012777 000032 167622                     MOV     #32,@CSR                ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
012124 052777 000002 167624                     BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
012132 052777 000001 167606                     BIS     #BIT0,@CSR              ;SET ENABLE NCV11
012140 042777 000002 167610                     BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
012146 017737 167606 001126                     MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
012154 012737 000402 001124                     MOV     #402,$GDDAT             ;LOAD EXPECTED VALUE
012162 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
012170 001401                                   BEQ     TST56           ;;BR IF SAME
012172 104020                                   ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                        ;                       RESOLUTION = 5  GAIN = 0  ZB ENABLE = 0
                                        ;;*****************************************************************
                                        ;*TEST 56        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 5 GAIN = 1 ZB ENABLE - 0
                                        ;;*****************************************************************
012174 000004                           TST56:  SCOPE
012176 012737 000010 001160                     MOV     #10,$TIMES      ;;DO 10 ITERATIONS
012204 012777 004000 167544                     MOV     #CLRALL,@SFR            ;CLEAR DEVICE
012212 012777 000014 167536                     MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
012220 012777 002032 167520                     MOV     #2032,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
012226 052777 000002 167522                     BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
012234 052777 000001 167504                     BIS     #BIT0,@CSR              ;SET ENABLE NCV11
012242 042777 000002 167506                     BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
012250 017737 167504 001126                     MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
012256 012737 025526 001124                     MOV     #25526,$GDDAT           ;LOAD EXPECTED VALUE
012264 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
012272 001401                                   BEQ     TST57           ;;BR IF SAME
012274 104020                                   ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                        ;                       RESOLUTION = 5  GAIN = 1  ZB ENABLE = 0
                                        ;;*****************************************************************
                                        ;*TEST 57        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 5 GAIN - 0 ZB ENABLE   1
                                        ;;*****************************************************************
012276 000004                           TST57:  SCOPE
012300 012737 000010 001160                     MOV     #10,$TIMES      ;;DO 10 ITERATIONS
012306 012777 004000 167442                     MOV     #CLRALL,@SFR            ;CLEAR DEVICE
012314 012777 000014 167434                     MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
012322 012777 004032 167416                     MOV     #4032,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
012330 052777 000002 167420                     BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
012336 052777 000001 167402                     BIS     #BIT0,@CSR              ;SET ENABLE NCV11
012344 042777 000002 167404                     BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
012352 017737 167402 001126                     MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
012360 012737 052452 001124                     MOV     #52452,$GDDAT           ;LOAD EXPECTED VALUE
012366 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
012374 001401                                   BEQ     TST60           ;;BR IF SAME
012376 104020                                   ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                        ;                       RESOLUTION = 5  GAIN = 0  ZB ENABLE = 1
```

```
1802                                    ;;*****************************************************************
1803                                    ;*TEST 60       TEST ADDRESS MAKER  -  MATRIX MODE - RES   4 GAIN - 0 ZB ENABLE - 0
1804                                    ;;*****************************************************************
1805  012400  000004                   TST60:  SCOPE
1806  012402  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1807  012410  012777  004000  167340           MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1808  012416  012777  000014  167332           MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1809  012424  012777  000030  167314           MOV     #30,@CSR                ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1810  012432  052777  000002  167316           BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1811  012440  052777  000001  167300           BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1812  012446  042777  000002  167302           BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1813  012454  017737  167300  001126           MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1814  012462  012737  000202  001124           MOV     #202,$GDDAT             ;LOAD EXPECTED VALUE
1815  012470  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1816  012476  001401                           BEQ     TST61           ;;BR IF SAME
1817  012500  104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1818                                  ;                       RESOLUTION = 4  GAIN = 0  ZB ENABLE = 0
1819                                  ;;*****************************************************************
1820                                  ;*TEST 61       TEST ADDRESS MAKER  -  MATRIX MODE - RES = 4 GAIN = 1 ZB ENABLE   0
1821                                  ;;*****************************************************************
1822  012502  000004                   TST61:  SCOPE
1823  012504  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1824  012512  012777  004000  167236           MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1825  012520  012777  000014  167230           MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1826  012526  012777  002030  167212           MOV     #2030,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1827  012534  052777  000002  167214           BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1828  012542  052777  000001  167176           BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1829  012550  042777  000002  167200           BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1830  012556  017737  167176  001126           MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1831  012564  012737  012726  001124           MOV     #012726,$GDDAT          ;LOAD EXPECTED VALUE
1832  012572  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1833  012600  001401                           BEQ     TST62           ;;BR IF SAME
1834  012602  104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1835                                  ;                       RESOLUTION = 4  GAIN = 1  ZB ENABLE = 0
1836                                  ;;*****************************************************************
1837                                  ;*TEST 62       TEST ADDRESS MAKER  -  MATRIX MODE - RES = 4 GAIN = 0 ZB ENABLE = 1
1838                                  ;;*****************************************************************
1839  012604  000004                   TST62:  SCOPE
1840  012606  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1841  012614  012777  004000  167134           MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1842  012622  012777  000014  167126           MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1843  012630  012777  004030  167110           MOV     #4030,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1844  012636  052777  000002  167112           BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1845  012644  052777  000001  167074           BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1846  012652  042777  000002  167076           BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1847  012660  017737  167074  001126           MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1848  012666  012737  025252  001124           MOV     #025252,$GDDAT          ;LOAD EXPECTED VALUE
1849  012674  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1850  012702  001401                           BEQ     TST63           ;;BR IF SAME
1851  012704  104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1852                                  ;                       RESOLUTION - 4  GAIN = 0  ZB ENABLE = 1
```

```
1853                                    ;;*****************************************************************
1854                                    ;*TEST 63       TEST ADDRESS MAKER  -  MATRIX MODE - RES   3 GAIN   0 ZB ENABLE - 0
1855                                    ;;*****************************************************************
1856  012706  000004                   TST63:  SCOPE
1857  012710  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1858  012716  012777  004000  167032           MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1859  012724  012777  000014  167024           MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1860  012732  012777  000026  167006           MOV     #26,@CSR                ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1861  012740  052777  000002  167010           BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1862  012746  052777  000001  166772           BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1863  012754  042777  000002  166774           BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1864  012762  017737  166772  001126           MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1865  012770  012737  000000  001124           MOV     #0,$GDDAT               ;LOAD EXPECTED VALUE
1866  012776  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1867  013004  001401                           BEQ     TST64           ;;BR IF SAME
1868  013006  104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1869                                   ;                       RESOLUTION = 3  GAIN = 0  ZB ENABLE = 0
1870                                    ;;*****************************************************************
1871                                    ;*TEST 64       TEST ADDRESS MAKER  -  MATRIX MODE - RES = 3 GAIN = 1 ZB ENABLE - 0
1872                                    ;;*****************************************************************
1873  013010  000004                   TST64:  SCOPE
1874  013012  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1875  013020  012777  004000  166730           MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1876  013026  012777  000014  166722           MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1877  013034  012777  002026  166704           MOV     #2026,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1878  013042  052777  000002  166706           BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1879  013050  052777  000001  166670           BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1880  013056  042777  000002  166672           BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1881  013064  017737  166670  001126           MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1882  013072  012737  005252  001124           MOV     #5252,$GDDAT            ;LOAD EXPECTED VALUE
1883  013100  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1884  013106  001401                           BEQ     TST65           ;;BR IF SAME
1885  013110  104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1886                                   ;                       RESOLUTION = 3  GAIN = 1  ZB ENABLE = 0
1887                                    ;;*****************************************************************
1888                                    ;*TEST 65       TEST ADDRESS MAKER  -  MATRIX MODE - RES = 3 GAIN = 0 ZB ENABLE = 1
1889                                    ;;*****************************************************************
1890  013112  000004                   TST65:  SCOPE
1891  013114  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1892  013122  012777  004000  166626           MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1893  013130  012777  000014  166620           MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1894  013136  012777  004026  166602           MOV     #4026,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1895  013144  052777  000002  166604           BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1896  013152  052777  000001  166566           BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1897  013160  042777  000002  166570           BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1898  013166  017737  166566  001126           MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1899  013174  012737  012424  001124           MOV     #12424,$GDDAT           ;LOAD EXPECTED VALUE
1900  013202  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1901  013210  001401                           BEQ     TST66           ;;BR IF SAME
1902  013212  104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1903                                   ;                       RESOLUTION - 3  GAIN = 0  ZB ENABLE   1
```

```
1904                                    ;;*****************************************************************
1905                                    ;*TEST 66       TEST ADDRESS MAKER  -  MATRIX MODE - RES - 2 GAIN   0 ZB ENABLE = C
1906                                    ;;*****************************************************************
1907  013214  000004                    TST66:  SCOPE
1908  013216  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1909  013224  012777  004000  166524            MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1910  013232  012777  000014  166516            MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1911  013240  012777  000024  166500            MOV     #24,@CSR                ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1912  013246  052777  000002  166502            BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1913  013254  052777  000001  166464            BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1914  013262  042777  000002  166466            BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1915  013270  017737  166464  001126            MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1916  013276  012737  000000  001124            MOV     #0,$GDDAT               ;LOAD EXPECTED VALUE
1917  013304  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1918  013312  001401                            BEQ     TST67           ;;BR IF SAME
1919  013314  104020                            ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1920                                    ;                       RESOLUTION = 2  GAIN = 0  ZB ENABLE - 0
1921                                    ;;*****************************************************************
1922                                    ;*TEST 67       TEST ADDRESS MAKER  -  MATRIX MODE - RES = 2 GAIN - 1 ZB ENABLE - 0
1923                                    ;;*****************************************************************
1924  013316  000004                    TST67:  SCOPE
1925  013320  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1926  013326  012777  004000  166422            MOV     #CLRALL,@SFR            ;CLEAR DFVICE
1927  013334  012777  000014  166414            MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1928  013342  012777  002024  166376            MOV     #2024,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1929  013350  052777  000002  166400            BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1930  013356  052777  000001  166362            BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1931  013364  042777  000002  166364            BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1932  013372  017737  166362  001126            MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1933  013400  012737  002552  001124            MOV     #2552,$GDDAT            ;LOAD EXPECTED VALUE
1934  013406  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1935  013414  001401                            BEQ     TST70           ;;BR IF SAME
1936  013416  104020                            ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1937                                    ;                       RESOLUTION = 2  GAIN = 1  ZB ENABLE = 0
1938                                    ;;*****************************************************************
1939                                    ;*TEST 70       TEST ADDRESS MAKER  -  MATRIX MODE - RES = 2 GAIN = 0 ZB ENABLE - 1
1940                                    ;;*****************************************************************
1941  013420  000004                    TST70:  SCOPE
1942  013422  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1943  013430  012777  004000  166320            MOV     #CLRALL,@SFR            ;CLEAR DEVICE
1944  013436  012777  000014  166312            MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
1945  013444  012777  004024  166274            MOV     #4024,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
1946  013452  052777  000002  166276            BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
1947  013460  052777  000001  166260            BIS     #BIT0,@CSR              ;SET ENABLE NCV11
1948  013466  042777  000002  166262            BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
1949  013474  017737  166260  001126            MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
1950  013502  012737  005224  001124            MOV     #5224,$GDDAT            ;LOAD EXPECTED VALUE
1951  013510  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
1952  013516  001401                            BEQ     TST71           ;;BR IF SAME
1953  013520  104020                            ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
1954                                    ;                       RESOLUTION   2  GAIN = 0  ZB ENABLE - 1
```

```
                                ;;*****************************************************************
                                ;*TEST 71        TEST ADDRESS MAKER  -  MATRIX MODE - RES - 1 GAIN   0 ZB ENABLE - 0
                                ;;*****************************************************************
013522 000004                   TST71:  SCOPE
013524 012737 000010 001160             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
013532 012777 004000 166216             MOV     #CLRALL,@SFR            ;CLEAR DEVICE
013540 012777 000014 166210             MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
013546 012777 000022 166172             MOV     #22,@CSR                ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
013554 052777 000002 166174             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
013562 052777 000001 166156             BIS     #BIT0,@CSR              ;SET ENABLE NCV11
013570 042777 000002 166160             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
013576 017737 166156 001126             MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
013604 012737 000000 001124             MOV     #0,$GDDAT               ;LOAD EXPECTED VALUE
013612 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
013620 001401                           BEQ     TST72           ;;BR IF SAME
013622 104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                ;                       RESOLUTION = 1  GAIN = 0  ZB ENABLE - 0
                                ;;*****************************************************************
                                ;*TEST 72        TEST ADDRESS MAKER  -  MATRIX MODE - RES - 1 GAIN = 1 ZB ENABLE - 0
                                ;;*****************************************************************
013624 000004                   TST72:  SCOPE
013626 012737 000010 001160             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
013634 012777 004000 166114             MOV     #CLRALL,@SFR            ;CLEAR DEVICE
013642 012777 000014 166106             MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
013650 012777 002022 166070             MOV     #2022,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
013656 052777 000002 166072             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
013664 052777 000001 166054             BIS     #BIT0,@CSR              ;SET ENABLE NCV11
013672 042777 000002 166056             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
013700 017737 166054 001126             MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
013706 012737 001265 001124             MOV     #1265,$GDDAT            ;LOAD EXPECTED VALUE
013714 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
013722 001401                           BEQ     TST73           ;;BR IF SAME
013724 104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                ;                       RESOLUTION = 1  GAIN = 1  ZB ENABLE = 0
                                ;;*****************************************************************
                                ;*TEST 73        TEST ADDRESS MAKER  -  MATRIX MODE - RES = 1 GAIN = 0 ZB ENABLE - 1
                                ;;*****************************************************************
013726 000004                   TST73:  SCOPE
013730 012737 000010 001160             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
013736 012777 004000 166012             MOV     #CLRALL,@SFR            ;CLEAR DEVICE
013744 012777 000014 166004             MOV     #TSTDMA!TSTCON,@SFR     ;SET 'TEST DMA AND CONTROL''
013752 012777 004022 165766             MOV     #4022,@CSR              ;LOAD RESOLUTION, GAIN, ZB ENABLE VALUE
013760 052777 000002 165770             BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
013766 052777 000001 165752             BIS     #BIT0,@CSR              ;SET ENABLE NCV11
013774 042777 000002 165754             BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
014002 017737 165752 001126             MOV     @ADM,$BDDAT             ;READ THE ADDRESS DATA MAKER
014010 012737 002512 001124             MOV     #2512,$GDDAT            ;LOAD EXPECTED VALUE
014016 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
014024 001401                           BEQ     TST74           ;;BR IF SAME
014026 104020                           ERROR   20                      ;INCORRECT ADDRESS MAKER DATA
                                ;                       RESOLUTION - 1  GAIN = 0  ZB ENABLE   1
```

```
                                        ;;*****************************************************************
                                        ;*TEST 74        TEST ADDRESS MAKER  -  LIST MODE - ZB ENABLE = 0  GAIN - 0
                                        ;;*****************************************************************
014030  000004                  TST74:  SCOPE
014032  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
014040  012777  004000  165710          MOV     #CLRALL,@SFR            ;CLEAR DEVICE
014046  012777  000014  165702          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST 'DMA AND CONTROL''
014054  012777  000000  165664          MOV     #0,@CSR                 ;ENSURE LIST MODE
014062  052777  000002  165666          BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
014070  052777  000001  165650          BIS     #BIT0,@CSR              ;SET ENABLE NCV11
014076  042777  000002  165652          BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
014104  017737  165650  001126          MOV     @ADM,$BDDAT             ;READ THE ADDRESS MAKER VALUE
014112  012737  003407  001124          MOV     #3407,$GDDAT            ;LOAD EXPECTED VALUE
014120  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
014126  001401                        BEQ     TST75           ;;BR IF SAME
014130  104021                        ERROR   21                      ;INCORRECT ADDRESS MAKER DATA
                                                                ;RESOLUTION 7 <DEFAULT WHEN ZB IS NOT ENABLED>

                                        ;;*****************************************************************
                                        ;*TEST 75        TEST ADDRESS MAKER  -  LIST MODE - ZB ENABLE = 1  GAIN - 0
                                        ;;*****************************************************************
014132  000004                  TST75:  SCOPE
014134  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
014142  012777  004000  165606          MOV     #CLRALL,@SFR            ;CLEAR DEVICE
014150  012777  000014  165600          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST 'DMA AND CONTROL''
014156  012777  004000  165562          MOV     #4000,@CSR              ;ENSURE LIST MODE AND ZB ENABLED
014164  052777  000002  165564          BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
014172  052777  000001  165546          BIS     #BIT0,@CSR              ;SET ENABLE NCV11
014200  042777  000002  165550          BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
014206  017737  165546  001126          MOV     @ADM,$BDDAT             ;READ THE ADDRESS MAKER VALUE
014214  012737  052452  001124          MOV     #52452,$GDDAT           ;LOAD EXPECTED VALUE
014222  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPECTED TO READ
014230  001401                        BEQ     TST76           ;;BR IF SAME
014232  104021                        ERROR   21                      ;INCORRECT ADDRESS MAKER DATA
                                                                ;RESOLUTION 5 <DEFAULT WHEN ZB IS ENABLED>

                                        ;;*****************************************************************
                                        ;*TEST 76        TEST ADDRESS MAKER  -  LIST MODE - ZB ENABLE - 0  GAIN = 1
                                        ;;*****************************************************************
014234  000004                  TST76:  SCOPE
014236  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
014244  012777  004000  165504          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
014252  012777  000014  165476          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST 'DMA AND CONTROL''
014260  012777  002000  165460          MOV     #2000,@CSR              ;SET GAIN FLOP
014266  052777  000002  165462          BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
014274  052777  000001  165444          BIS     #BIT0,@CSR              ;ENABLE THE NCV11
014302  042777  000002  165446          BIC     #TESTZ,@SFR             ;DISABLE THE 'TEST Z'' PULSES
014310  017737  165444  001126          MOV     @ADM,$BDDAT             ;READ THE ADDRESS MAKER VALUE
014316  012737  127657  001124          MOV     #127657,$GDDAT          ;LOAD THE EXPECTED
014324  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
014332  001401                        BEQ     TST77           ;;BR IF SAME
014334  104021                        ERROR   21                      ;INCORRECT ADDRESS MAKER DATA
                                                                ;RESOLUTION 7 -  GAIN FLOP SET
```

```
                                        ;;*********************************************************************
                                        ;*TEST 77       ENABLE A ONE WORD TRANSFER      SECTION LIST MODE
                                        ;;*********************************************************************
                                        ;
014336 000004                   TST77:  SCOPE
014340 012737 000040 001160             MOV     #40,$TIMES      ;;DO 40 ITERATIONS
014346 012777 004000 165402             MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
014354 012737 125252 060000             MOV     #125252,BUF0            ;PRIME TARGET BUFFER
014362 012777 011110 165360             MOV     #11110,@OFF             ;LOAD THE OFFSET VALUE WITH A NUMBER
014370 012777 177777 165354             MOV     #-1,@WCR                ;SET UP 1 WORD TRANSFER
014376 012777 060000 165350             MOV     #BUF0,@BAR              ;LOAD BUS ADDRESS FOR RESULT
014404 012777 000014 165344             MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE TEST CONTROL AND DMA FLOPS
014412 052777 000001 165326             BIS     #BIT0,@CSR              ;ENABLE DEVICE
014420 052777 000002 165330             BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
014426 042777 000002 165322             BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
014434 000240                           NOP
014436 000240                           NOP
014440 000240                           NOP
014442 052777 010000 165306             BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
014450 000240                           NOP
014452 000240                           NOP
014454 000240                           NOP
014456 017737 165264 001126             MOV     @CSR,$BDDAT             ;READ STATUS
014464 012777 004000 165264             MOV     #CLRALL,@SFR            ;RESET THE DEVICE
014472 012737 040200 001124             MOV     #40200,$GDDAT           ;LOAD EXPECTED
014500 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;TEST STATUS
014506 001402                           BEQ     4$              ;;BR IF EXPECTED
014510 104036                           ERROR   36                      ;UNEXPECTED STATUS AFTER A 1 WORD TRANSFER
014512 000453                           BR      TST100          ;;
014514 017737 165234 001126     4$:     MOV     @BAR,$BDDAT             ;READ BUS ADDRESS
014522 012737 060002 001124             MOV     #BUF0+2,$GDDAT          ;LOAD EXPECTED
014530 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
014536 001401                           BEQ     1$              ;;BR IF SAME
014540 104022                           ERROR   22                      ;INCORRECT BUS ADDRESS VALUE
                                        ;AFTER A 1 WORD TRANSFER
                                        ;
014542 017737 165204 001126     1$:     MOV     @WCR,$BDDAT             ;READ W.C. REGISTER
014550 012737 000000 001124             MOV     #0,$GDDAT               ;LOAD EXPECTED VALUE
014556 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
014564 001401                           BEQ     2$              ;;BR IF SAME
014566 104023                           ERROR   23                      ;INCORRECT WORD COUNT REGISTER VALUE
                                        ;AFTER A 1 WORD TRANSFER
014570 005037 001124            2$:     CLR     $GDDAT                  ;CLEAR THE EXPECTED VALUE
014574 017737 165150 001126             MOV     @OFF,$BDDAT             ;READ THE OFFSET REGISTER
014602 001401                           BEQ     3$              ;;BR IF CLEARED
014604 104024                           ERROR   24                      ;OFFSET REG. FAILED TO CLEAR AFTER
                                        ;1 LIST MODE XFR.
014606 013737 060000 001126     3$:     MOV     BUF0,$BDDAT             ;GET BUFFER DATA
014614 012737 003407 001124             MOV     #3407,$GDDAT            ;LOAD EXPECTED
014622 012737 060000 001122             MOV     #BUF0,$BDADR            ;LOAD BAD ADDRESS
014630 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE DATA
014636 001401                           BEQ     TST100          ;;BR IF SAME
014640 104037                           ERROR   37                      ;STATUS WAS OK BUT DATA WAS INCORRECT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 100      ENABLE A 512 WORD TRANSFER      SECTION LIST MODE
                                        ;;*****************************************************************
014642 000004                           TST100: SCOPE
014644 012737 000040 001160                     MOV     #40,$TIMES      ;;DO 40 ITERATIONS
014652 012777 004000 165076                     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
014660 012700 060000                            MOV     #BUF0,R0                ;LOAD BUFFER POINTER
014664 012720 125252                    1$:     MOV     #125252,(R0)+           ;PRESET THE BUFFER WITH DATA
014670 020027 062000                            CMP     R0,#BUF1                ;TEST IF DONE
014674 001373                                   BNE     1$                      ;BR IF NOT
014676 012777 177000 165046                     MOV     #-512.,@WCR             ;SET UP 512. WORD TRANSFER
014704 012777 060000 165042                     MOV     #BUF0,@BAR              ;LOAD BUS ADDRESS FOR RESULT
014712 012777 000014 165036                     MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE TEST CONTROL AND DMA FLOPS
014720 052777 000001 165020                     BIS     #BIT0,@CSR              ;ENABLE DEVICE
014726 012737 001000 002004                     MOV     #512.,$TEMP             ;LOAD THE COUNTER
014734                                  2$:
014734 052777 000002 165014                     BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
014742 042777 000002 165006                     BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
014750 052777 010000 165000                     BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
014756 005337 002004                            DEC     $TEMP                   ;FINISHED ALL WORDS?
014762 001364                                   BNE     2$                      ;BR UNTILL DONE
                                        ;THE TRANSFER IS NOW COMPLETE
014764 017737 164756 001126                     MOV     @CSR,$BDDAT             ;READ STATUS
014772 012737 040200 001124                     MOV     #40200,$GDDAT           ;LOAD EXPECTED STATUS
015000 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE DATA
015006 001402                                   BEQ     3$              ;;BR IF EXPECTED STATUS
015010 104036                                   ERROR   36                      ;UNEXPECTED STATUS AFTER 512 WORD TRANSFER
015012 000465                                   BR      TST101          ;;
015014 005037 001124                    3$:     CLR     $GDDAT                  ;CLEAR EXPECTED
015020 017737 164724 001126                     MOV     @OFF,$BDDAT             ;READ OFFSET REG.
015026 001401                                   BEQ     4$              ;;BR IF CLEARED
015030 104006                                   ERROR   6               ;UNEXPECTED OFFSET REGISTER BIT SET
015032 012777 004000 164716             4$:     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
015040 017737 164710 001126                     MOV     @BAR,$BDDAT             ;READ BUS ADDRESS
015046 012737 062000 001124                     MOV     #BUF1,$GDDAT            ;LOAD EXPECTED BAR VALUE
015054 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
015062 001401                                   BEQ     5$              ;;BR IF SAME
015064 104022                                   ERROR   22      ;INCORRECT BUS ADDRESS VALUE AFTER A 1 WORD TRANSFER
015066 017737 164660 001126             5$:     MOV     @WCR,$BDDAT             ;READ W.C. REGISTER
015074 012737 000000 001124                     MOV     #0,$GDDAT               ;LOAD EXPECTED W.C. VALUE
015102 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
015110 001401                                   BEQ     6$              ;;BR IF SAME
015112 104023                                   ERROR   23      ;INCORRECT WORD COUNT REGISTER VALUE AFTER A 1 WORD TRANSFER
015114 012737 003407 001124             6$:     MOV     #3407,$GDDAT            ;LOAD EXPECTED DATA
015122 012737 060000 001122                     MOV     #BUF0,$BDADR            ;LOAD STARTING ADDRESS
015130 017737 163766 001126             7$:     MOV     @$BDADR,$BDDAT          ;READ DATA WORD
015136 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE DATA
015144 001401                                   BEQ     10$             ;;BR IF EXPECTED
015146 104037                                   ERROR   37                      ;INCORRECT DATA IN LIST MODE XFER.
015150 062737 000002 001122             10$:    ADD     #2,$BDADR               ;UPDATE POINTER
015156 022737 062000 001122                     CMP     #BUF0+1024.,$BDADR      ;TEST IF END OF BUFFER
015164 001361                                   BNE     7$              ;;BR IF NOT DONE
```

```
                                ;;*****************************************************************
                                ;*TEST 101       VERIFY "TIMEOUT" FLOP SETS AND "CLR ALL" CLEARS IT
                                ;;*****************************************************************
015166  000004                  TST101: SCOPE
015170  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
015176  012777  004000  164552          MOV     #CLRALL,@SFR            ;CLEAR DEVICE
015204  012777  177777  164540          MOV     #-1,@WCR                ;LOAD W.C. REGISTER
015212  012777  000003  164530          MOV     #3,@OFF                 ;LOAD EXTENDED ADDRESS BITS
015220  012777  160000  164526          MOV     #160000,@BAR            ;LOAD BUS ADDRESS REGISTER TO A NON-EXIENT ADDRE
015226  012777  000014  164522          MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROL AND DMA
015234  012777  000016  164504          MOV     #16,@CSR                ;LOAD RESOLUTION TO VERIFY "SFR INIT" CLEARS
015242  052777  000001  164476          BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
015250  052777  000002  164500          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
015256  042777  000002  164472          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
015264  000240                        NOP
015266  000240                        NOP
015270  000240                        NOP
015272  052777  010000  164456          BIS     #BIT12,@SFR             ;ALLOW  1 DMA TRANSFER
015300  000240                        NOP
015302  000240                        NOP
015304  000240                        NOP
015306  012737  140200  001124          MOV     #BIT15!BIT14.BIT7,$GDDAT        ;LOAD EXPECTED
015314  017737  164426  001126          MOV     @CSR,$BDDAT             ;READ STATUS REG.
015322  100402                        BMI     1$                      ;BR IF "TIMEOUT" FLOP IS SET
015324  104025                        ERROR   25                      ;"TIMEOUT" FLOP FAILED TO SET
015326  000423                        BR      TST102          ;;
015330  023737  001124  001126  1$:   CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
015336  001401                        BEQ     2$              ;;BR IF SAME
015340  104025                        ERROR   25                      ;"TIMEOUT" FLOP DID SET BUT FAILED TO GENERATE "
015342  052777  004000  164406  2$:   BIS     #CLRALL,@SFR            ;GENERATE AN "CLR ALL" TO CLEAR TIMEOUT FLOP
015350  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
015356  017737  164364  001126          MOV     @CSR,$BDDAT             ;READ STATUS
015364  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUE
015372  001401                        BEQ     TST102          ;;BR IF SAME
015374  104025                        ERROR   25                      ;"CLR ALL" FAILED TO CLEAR TIMEOUT FLOP
                                ;;*****************************************************************
                                ;*TEST 102       VERIFY "TIMEOUT" FLOP SETS AND "CLR TIMEOUT" CLEARS IT
                                ;;*****************************************************************
015376  000004                  TST102: SCOPE
015400  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
015406  012777  004000  164342          MOV     #CLRALL,@SFR            ;CLEAR DEVICE
015414  012777  177777  164330          MOV     #-1,@WCR                ;LOAD W.C. REGISTER
015422  012777  000003  164320          MOV     #3,@OFF                 ;LOAD EXTENDED ADDRESS BITS
015430  012777  160000  164316          MOV     #160000,@BAR            ;LOAD BUS ADDRESS REGISTER TO A NON-EXIENT ADDRE
015436  012777  000014  164312          MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROL AND DMA
015444  052777  000001  164274          BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
015452  052777  000002  164276          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
015460  042777  000002  164270          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
015466  000240                        NOP
015470  000240                        NOP
015472  000240                        NOP
015474  052777  010000  164254          BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
015502  000240                        NOP
015504  000240                        NOP
```

```
2218  015506  000240                          NOP
2219  015510  012737  140200  001124          MOV     #BIT15.BIT14!BIT7,$GDDAT        ;LOAD EXPECTED
2220  015516  017737  164224  001126          MOV     @CSR,$BDDAT             ;READ STATUS REG.
2221  015524  100402                          BMI     1$                      ;BR IF "TIMEOUT" FLOP IS SET
2222  015526  104025                          ERROR   25                      ;"TIMEOUT" FLOP FAILED TO SET
2223  015530  000423                          BR      TST103          ;;
2224  015532  023737  001124  001126  1$:     CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
2225  015540  001401                          BEQ     2$              ;;BR IF SAME
2226  015542  104025                          ERROR   25                      ;"TIMEOUT" FLOP DID SET BUT FAILED TO GENERATE "
2227  015544  052777  100000  164204  2$:     BIS     #BIT15,@SFR             ;GENERATE AN "CLR TIMEOUT" TO CLEAR TIMEOUT FLOP
2228  015552  012737  040200  001124          MOV     #BIT14!BIT7,$GDDAT      ;LOAD EXPECTED
2229  015560  017737  164162  001126          MOV     @CSR,$BDDAT             ;READ STATUS
2230  015566  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUE
2231  015574  001401                          BEQ     TST103          ;;BR IF SAME
2232  015576  104025                          ERROR   25                      ;"CLR TIMEOUT" FAILED TO CLEAR TIMEOUT FLOP
2233                                ;;*****************************************************************
2234                                ;*TEST 103       VERIFY "TIMEOUT" INTERRUPT
2235                                ;;*****************************************************************
2236  015600  000004                TST103: SCOPE
2237  015602  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
2238  015610  012777  004000  164140          MOV     #CLRALL,@SFR            ;CLEAR DEVICE
2239  015616  012777  177777  164126          MOV     #-1,@WCR                ;LOAD W.C. REGISTER
2240  015624  012777  000003  164116          MOV     #3,@OFF                 ;LOAD FXTENDED ADDRESS BITS
2241  015632  012777  160000  164114          MOV     #160000,@BAR            ;LOAD BUS ADDRESS REGISTER TO A NON-EXIENT ADDRE
2242  015640  012777  000014  164110          MOV     #TSTCON!TSTDMA,@SFR     ;SET TEST CONTROL AND DMA
2243  015646  012746  000000                  MOV     #0,-(SP)
2244  015652  012746  015660                  MOV     #1$,-(SP)
2245  015656  000002                          RTI
2246  015660  012777  015752  164104  1$:     MOV     #2$,@VECTA0
2247  015666  000240                          NOP
2248  015670  000240                          NOP
2249  015672  000240                          NOP
2250  015674  000240                          NOP
2251  015676  052777  000101  164042          BIS     #BIT6!BIT0,@CSR         ;ENABLE THE DEVICE
2252  015704  052777  000002  164044          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
2253  015712  042777  000002  164036          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
2254  015720  000240                          NOP
2255  015722  000240                          NOP
2256  015724  000240                          NOP
2257  015726  052777  010000  164022          BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSGFER
2258  015734  000240                          NOP
2259  015736  000240                          NOP
2260  015740  000240                          NOP
2261  015742  005077  164000                  CLR     @CSR                    ;CLEAR ENABLE
2262  015746  104026                          ERROR   26                      ;"TIMEOUT" FAILED TO INTERRUPT
2263  015750  000401                          BR      3$              ;;BR TO CLEAN UP
2264
2265  015752  022626                  2$:     CMP     (SP)+,(SP)+             ;CLEAN THE STACK
2266  015754  005077  163766          3$:     CLR     @CSR
2267  015760  012777  004000  163770          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
2268  015766  013777  001774  163776          MOV     VECTA1,@VECTA0          ;RESET VECTOR
2269  015774  005077  163774                  CLR     @VECTA1
```

```
                                        ;;*****************************************************************
                                        ;*TEST 104      VERIFY INCREMENTING INTO EXTENDED ADDRESS BIT 16
                                        ;;*****************************************************************
016000  000004                          TST104: SCOPE
016002  012737  000040  001160                  MOV     #40,$TIMES      ;;DO 40 ITERATIONS
016010  012777  004000  163740                  MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
016016  023727  036402  002140                  CMP     $LSTBK,#2140            ;TEST IF ENOUGH MEMORY
016024  103450                                  BLO     TST105          ;;BR IF NOT ENOUGH MEMORY
016026  012777  177777  163716                  MOV     #-1,@WCR                ;LOAD 1 WORD XFR
016034  012777  177776  163712                  MOV     #177776,@BAR            ;LOAD LAST ADDRESS
016042  005077  163702                          CLR     @OFF                    ;ENSURE CLEARED EXTENDED ADDRESS BITS
016046  012777  000014  163702                  MOV     #TSTCON!TSTDMA,@SFR     ;ENABLE CONTROLER
016054  012777  000016  163664                  MOV     #16,@CSR                ;ENABLE THE MODE
016062  052777  000001  163656                  BIS     #BIT0,@CSR              ;ENABLE THE NCV11
016070  052777  000002  163660                  BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
016076  042777  000002  163652                  BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
016104  052777  010000  163644                  BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
016112  000240                                  NOP
016114  000240                                  NOP
016116  000240                                  NOP
016120  012737  000001  001124                  MOV     #1,$GDDAT               ;LOAD EXPECTED VALUE
016126  017737  163616  001126                  MOV     @OFF,$BDDAT             ;READ ACUTAL VALUE
016134  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;COMPARE
016142  001401                                  BEQ     TST105          ;;BR IF SAME
016144  104006                                  ERROR   6               ;EXTENDED ADDRESS BIT 16 FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 105      VERIFY INCREMENTING INTO EXTENDED ADDRESS BIT 17
                                        ;;*****************************************************************
016146  000004                          TST105: SCOPE
016150  012737  000040  001160                  MOV     #40,$TIMES      ;;DO 40 ITERATIONS
016156  012777  004000  163572                  MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
016164  023727  036402  006140                  CMP     $LSTBK,#6140            ;TEST MEMORY SPACE >100K
016172  103454                                  BLO     TST106          ;;BR IF NOT ENOUGH MEMORY
016174  012777  177777  163550                  MOV     #-1,@WCR                ;LOAD 1 WORD XFR
016202  012777  177776  163544                  MOV     #177776,@BAR            ;LOAD LAST ADDRESS
016210  012777  000001  163532                  MOV     #1,@OFF                 ;LOAD EXTENDED ADDRESS BIT
016216  012777  000014  163532                  MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE TEST CONTROL
016224  012777  000016  163514                  MOV     #16,@CSR                ;ENABLE THE MODE
016232  052777  000001  163506                  BIS     #BIT0,@CSR              ;ENAVLE THE DEVICE
016240  052777  000002  163510                  BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
016246  042777  000002  163502                  BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
016254  000240                                  NOP
016256  000240                                  NOP
016260  000240                                  NOP
016262  052777  010000  163466                  BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
016270  000240                                  NOP
016272  000240                                  NOP
016274  000240                                  NOP
016276  012737  000002  001124                  MOV     #2,$GDDAT               ;LOAD EXPECTED
016304  017737  163440  001126                  MOV     @OFF,$BDDAT             ;READ ACTUAL
016312  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;COMPARE
016320  001401                                  BEQ     TST106          ;;BR IF SAME
016322  104006                                  ERROR   6               ;EXTENDED ADDRESS BIT 17 FAILED TO SET
```

```
                                ;;*****************************************************************
                                ;*TEST 106       VERIFY ''SET EVENT'' DATA GENERATES A 177777 DATA WORD
                                ;;*****************************************************************
016324  000004                  TST106: SCOPE
016326  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
016334  012777  004000  163414          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
016342  012777  177777  163402          MOV     #-1,@WCR                ;SET UP 1 WORD TRANSFER
016350  012777  060000  163376          MOV     #BUF0,@BAR              ;LOAD BUS ADDRESS FOR RESULT
016356  012777  000014  163372          MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE TEST CONTROL AND DMA FLOPS
016364  012737  007070  060000          MOV     #7070,BUF0              ;PRESET DATA
016372  052777  000001  163346          BIS     #BIT0,@CSR              ;ENABLE DEVICE
016400  000240                          NOP
016402  000240                          NOP
016404  000240                          NOP
016406  052777  002000  163342          BIS     #BIT10,@SFR             ;SET ''EVENT'' FLOP
016414  052777  010000  163334          BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
016422  000240                          NOP
016424  000240                          NOP
016426  000240                          NOP
016430  013737  060000  001126          MOV     BUF0,$BDDAT             ;READ DATA
016436  012737  177777  001124          MOV     #-1,$GDDAT              ;LOAD EXPECTED
016444  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
016452  001401                          BEQ     TST107          ;;BR IF SAME
016454  104027                          ERROR   27                      ;INCORRECT DATA VALUE FOR ''EVENT'' MARK
                                                                ;AFTER A 1 WORD TRANSFER
                                ;;*****************************************************************
                                ;*TEST 107       VERIFY ''SET TIME'' DATA GENERATES A 000000 DATA WORD
                                ;;*****************************************************************
016456  000004                  TST107: SCOPE
016460  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
016466  012777  004000  163262          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
016474  012777  177777  163250          MOV     #-1,@WCR                ;SET UP 1 WORD TRANSFER
016502  012777  060000  163244          MOV     #BUF0,@BAR              ;LOAD BUS ADDRESS FOR RESULT
016510  012777  000014  163240          MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE TEST CONTROL AND DMA FLOPS
016516  012737  007070  060000          MOV     #7070,BUF0              ;PRESET THE DATA
016524  052777  000001  163214          BIS     #BIT0,@CSR              ;ENABLE DEVICE
016532  000240                          NOP
016534  000240                          NOP
016536  000240                          NOP
016540  052777  001000  163210          BIS     #BIT9,@SFR              ;SET ''TIME'' FLOP
016546  052777  010000  163202          BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
016554  000240                          NOP
016556  000240                          NOP
016560  000240                          NOP
016562  013737  060000  001126          MOV     BUF0,$BDDAT             ;READ DATA
016570  012737  000000  001124          MOV     #0,$GDDAT               ;LOAD EXPECTED
016576  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
016604  001401                          BEQ     TST110          ;;BR IF SAME
016606  104027                          ERROR   27                      ;INCORRECT DATA VALUE FOR ''TIME'' MARK
                                                                ;AFTER A 1 WORD TRANSFER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 110       VERIFY "CLOCK ST1" GENERATES A EVENT (177777) DATA WORD
                                        ;;*****************************************************************
016610 000004                           TST110: SCOPE
016612 012737 000040 001160                     MOV     #40,$TIMES      ;;DO 40 ITERATIONS
016620 012777 004000 163130                     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
016626 005737 004016                            TST     DEADKW                  ;TEST IF NCV11 CLOCK IS PRESENT
016632 001044                                   BNE     TST111          ;;BR IF NOT
016634 012777 177777 163110                     MOV     #-1,@WCR                ;SET UP 1 WORD TRANSFER
016642 012777 060000 163104                     MOV     #BUF0,@BAR              ;LOAD BUS ADDRESS FOR RESULT
016650 012777 000014 163100                     MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE TEST CONTROL AND DMA FLOPS
016656 012737 007070 060000                     MOV     #7070,BUF0              ;PRESET THE DATA
016664 052777 000001 163054                     BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
016672 000240                                   NOP
016674 000240                                   NOP
016676 052777 000400 163034                     BIS     #BIT8,@KWCSR            ;GENERATE CLOCK ST1 TO SET "EVENT" FLOP
016704 052777 010000 163044                     BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
016712 000240                                   NOP
016714 000240                                   NOP
016716 013737 060000 001126                     MOV     BUF0,$BDDAT             ;READ BUFFER DATA
016724 012737 177777 001124                     MOV     #-1,$GDDAT              ;LOAD EXPECTED DATA
016732 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE
016740 001401                                   BEQ     TST111          ;;BR IF EXPECTED
016742 104040                                   ERROR   40                      ;CLOCK ST1 FAILED TO GENERATE EVENT FLAG
                                                                                ;CHECK THE M8026 TO M7952 JUMPERS
                                        ;;*****************************************************************
                                        ;*TEST 111       VERIFY "CLOCK OVERFLOW" GENERATES A TIME (000000) DATA WORD
                                        ;;*****************************************************************
016744 000004                           TST111: SCOPE
016746 012737 000040 001160                     MOV     #40,$TIMES      ;;DO 40 ITERATIONS
016754 012777 004000 162774                     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
016762 005737 004016                            TST     DEADKW                  ;TEST IF NCV11 CLOCK IS PRESENT
016766 001052                                   BNE     TST112          ;;BR IF NOT
016770 012777 177777 162754                     MOV     #-1,@WCR                ;SET UP 1 WORD TRNSFER
016776 012777 060000 162750                     MOV     #BUF0,@BAR              ;LOAD BUS ADDRESS FOR RESULT
017004 012777 000014 162744                     MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE TEST CONTROL AND DMA FLOPS
017012 012737 007070 060000                     MOV     #7070,BUF0              ;PRESET THE DATA
017020 052777 000001 162720                     BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
017026 005077 162706                            CLR     @KWCSR                  ;CLEAR STATUS
017032 012777 177776 162704                     MOV     #-2,@KWPSR              ;LOAD COUNTER PRESET
017040 012777 000011 162672                     MOV     #11,@KWCSR              ;ENABLE 1 MHZ. RATE AND CLOCK GO
017046 105777 162666                    1$:     TSTB    @KWCSR                  ;WAIT FOR CLOCK
017052 100375                                   BPL     1$
017054 052777 010000 162674                     BIS     #BIT12,@SFR             ;ALLOW 1 DMA TRANSFER
017062 000240                                   NOP
017064 000240                                   NOP
017066 013737 060000 001126                     MOV     BUF0,$BDDAT             ;READ DATA
017074 012737 000000 001124                     MOV     #0,$GDDAT               ;LOAD EXPECTED
017102 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
017110 001401                                   BEQ     TST112          ;;BR IF SAME
017112 104040                                   ERROR   40                      ;CLOCK OVERFLOW FAILED TO GENERATE "TIME MARKS"
                                                                                ;CHECK THE M8026 TO M7952 JUMPERS
```

```
2426                                    ;;*****************************************************************
2427                                    ;*TEST 112      VERIFY "SET TIME" OVERRIDES "SET EVENT" DATA
2428                                    ;;*****************************************************************
2429  017114  000004                    TST112: SCOPE
2430  017116  012737  000040  001160            MOV     #40.,$TIMES       ;;DO 40 ITERATIONS
2431  017124  012777  004000  162624            MOV     #CLRALL,@SFR              ;CLEAR THE DEVICE
2432  017132  012777  177777  162612            MOV     #-1,@WCR                  ;SET UP 1 WORD TRANSFER
2433  017140  012777  060000  162606            MOV     #BUF0,@BAR                ;LOAD BUS ADDRESS FOR RESULT
2434  017146  012777  000014  162602            MOV     #TSTDMA!TSTCON,@SFR       ;ENABLE TEST CONTROL AND DMA FLOPS
2435  017154  012737  007070  060000            MOV     #7070,BUF0                ;PRESET THE DATA
2436  017162  052777  000001  162556            BIS     #BIT0,@CSR                ;ENABLE DEVICE
2437  017170  000240                            NOP
2438  017172  000240                            NOP
2439  017174  000240                            NOP
2440  017176  052777  003000  162552            BIS     #BIT10!BIT9,@SFR          ;SET "TIME AND EVENT" FLOPS
2441  017204  052777  010000  162544            BIS     #BIT12,@SFR               ;ALLOW 1 DMA TRANSFER
2442  017212  000240                            NOP
2443  017214  000240                            NOP
2444  017216  000240                            NOP
2445  017220  013737  060000  001126            MOV     BUF0,$BDDAT               ;READ DATA
2446  017226  012737  000000  001124            MOV     #0,$GDDAT                 ;LOAD EXPECTED
2447  017234  023737  001124  001126            CMP     $GDDAT,$BDDAT             ;COMPARE VALUES
2448  017242  001401                            BEQ     TST113            ;;BR IF SAME
2449  017244  104027                            ERROR   27                ;"TIME" MARK FAILED TO OVERIDE "EVENT" OR DATA M
2450                                    ;;*****************************************************************
2451                                    ;*TEST 113      DO A ONE WORD MATRIX MODE TRANSFER      -CHECK FOR INCREMENT FUNCTION
2452                                    ;;*****************************************************************
2453  017246  000004                    TST113: SCOPE
2454  017250  012737  000040  001160            MOV     #40.,$TIMES       ;;DO 40 ITERATIONS
2455  017256  012777  004000  162472            MOV     #CLRALL,@SFR              ;CLEAR THE DEVICE
2456  017264  005037  060000                    CLR     BUF0                      ;CLEAR INITIAL INREMENT LOCATION
2457  017270  012777  060000  162452            MOV     #BUF0,@OFF                ;LOAD INITIAL OFFSET REGISTER
2458  017276  012777  000022  162442            MOV     #BIT4!BIT1,@CSR           ;ENABLE MATRIX MODE
2459  017304  012777  000014  162444            MOV     #TSTDMA!TSTCON,@SFR       ;SET TEST DMA AND CONTROL
2460  017312  052777  000001  162426            BIS     #BIT0,@CSR                ;ENABLE THE DEVICE
2461  017320  052777  000002  162430            BIS     #TESTZ,@SFR               ;ENABLE "TEST Z" PULSES
2462  017326  042777  000002  162422            BIC     #TESTZ,@SFR               ;DISABLE "TEST Z" PULSES
2463  017334  000240                            NOP
2464  017336  000240                            NOP
2465  017340  000240                            NOP
2466  017342  052777  010000  162406            BIS     #BIT12,@SFR               ;ALLOW 1 TRANSFER
2467  017350  000240                            NOP
2468  017352  000240                            NOP
2469  017354  000240                            NOP
2470  017356  013737  060000  001126            MOV     BUF0,$BDDAT               ;READ THE BUF0 LOCATION
2471  017364  012737  000001  001124            MOV     #1,$GDDAT                 ;LOAD EXPECTED VALUE
2472  017372  023737  001124  001126            CMP     $GDDAT,$BDDAT             ;COMPARE VALUES
2473  017400  001401                            BEQ     TST114            ;;BR IF SAME
2474  017402  104030                            ERROR   30                ;INCORRECT DATA IN MATRIX MODE
2475                                                            ;IF DATA WAS 0, THE ADDRESS ACCESSED WAS PROBLY WRONG
2476                                                            ;IF DATA WAS NON ZERO, THE "INCR" REGISTER IS STUCK TO A 1
```

```
                                        ;;*********************************************************************
                                        ;*TEST 114      VERIFY EACH BIT OF THE INCREMENT REG. DATA PATH
                                        ;;*********************************************************************
017404  000004                          TST114: SCOPE
017406  012737  000040  001160                  MOV     #40,$TIMES      ;;DO 40 ITERATIONS
                                        ;CHECK FOR A ''CARRY'' INTO EACH BIT
                                        ;       IE:     000002-000001
                                        ;               000004-000003
                                        ;               000010-000007
                                        ;               000020-000017
                                        ;               000040-000037
                                        ;               000100-000077
                                        ;               000200-000177
                                        ;               ETC
017414  012737  017430  001110                  MOV     #1$,$LPERR              ;LOAD RETURN ADDRESS IF ERROR
017422  012737  000002  002004                  MOV     #2,$TEMP                ;LOAD INITIAL VALUE

017430  012777  004000  162320  1$:     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
017436  012777  060000  162304          MOV     #BUF0,@OFF              ;LOAD INITIAL OFFSET REGISTER
017444  012777  000024  162274          MOV     #BIT4!BIT2,@CSR         ;ENABLE WORD MATRIX MODE
017452  012777  000014  162276          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
017460  052777  000001  162260          BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
017466  013737  002004  060000          MOV     $TEMP,BUF0              ;LOAD PRESET VALUE
017474  005337  060000                  DEC     BUF0                    ;TO EXPECTED -1
017500  052777  000002  162250          BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
017506  042777  000002  162242          BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
017514  052777  010000  162234          BIS     #BIT12,@SFR             ;ALLOW 1 TRANSFER
017522  013737  002004  001124          MOV     $TEMP,$GDDAT            ;LOAD TYPEOUT EXPECTED
017530  013737  060000  001126          MOV     BUF0,$BDDAT             ;READ THE BUF0 LOCATION
017536  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
017544  001401                          BEQ     2$              ;;BR IF SAME
017546  104030                          ERROR   30              ;INCORRECT DATA IN THE INCREMENT REGISTER
017550  017737  162172  001126  2$:     MOV     @CSR,$BDDAT             ;READ STATUS
017556  032737  020000  001126          BIT     #BIT13,$BDDAT           ;TEST FOR UNEXPECTED CELL OVERFLOW
017564  001407                          BEQ     3$                      ;BR IF NOT
017566  012737  000024  001124          MOV     #BIT4!BIT2,$GDDAT       ;LOAD THE EXPECTED TYPEOUT
017574  104031                          ERROR   31              ;UNEXPECTED ''CELL OVERFLOW'' STATUS
017576  052777  020000  162152          BIS     #BIT13,@SFR             ;TRY TO CLEAR THE FLAG
017604  006337  002004          3$:     ASL     $TEMP                   ;CHANGE THE EXPCTED
017610  001307                          BNE     1$                      ;BR IF MORE DATA BITS
```

```
                                ;;*****************************************************************
                                ;*TEST 115      CHECK FOR LOW BYTE ''INC OVFL'' TO SET CELL OVERFLOW AND ''CLR CELL'' TO CLE
                                ;;*****************************************************************
017612 000004                   TST115: SCOPE
017614 012737 000040 001160             MOV     #40,$TIMES      ;;DO 40 ITERATIONS
017622 012777 004000 162126             MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
017630 005077 162116                    CLR     @WCR                    ;CLEAR WC
017634 005077 162114                    CLR     @BAR                    ;CLEAR BAR
017640 012777 060000 162102             MOV     #BUF0,@OFF              ;LOAD INITIAL OFFSET REGISTER
017646 012777 000022 162072             MOV     #BIT4!BIT1,@CSR         ;ENABLE BYTE MATRIX MODE
017654 012777 000014 162074             MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
017662 052777 000001 162056             BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
017670 052777 000002 162060             BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
017676 042777 000002 162052             BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
017704 000240                           NOP
017706 000240                           NOP
017710 000240                           NOP
017712 112737 000377 060000             MOVB    #377,BUF0               ;SET LOW BYTE OF BUF0 LOC. TO BYTE -1
017720 112737 000200 060001             MOVB    #200,BUF0+1             ;SET HIGH BYTE OF BUF0 TO KNOWN VALUE
017726 052777 010000 162022             BIS     #BIT12,@SFR             ;ALLOW 1 TRANSFER
017734 000240                           NOP
017736 000240                           NOP
017740 000240                           NOP
017742 017737 162000 001126             MOV     @CSR,$BDDAT             ;READ STATUS
017750 012737 020022 001124             MOV     #BIT13!BIT4!BIT1,$G.DAT ;LOAD EXPECTED STATUS
017756 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
017764 001401                           BEQ     1$              ;;BR IF SAME
017766 104031                           ERROR   31                      ;''CELL OVERFLOW'' FLOP FAILED TO SET
                                                                ;IN BYTE MODE FROM A LOW BYTE OVERFLOW


                                ;NOW GENERATE ''CLR CELL'' TO CLEAR CELL OVERFLOW FLOP
017770 052777 020000 161760     1$:     BIS     #BIT13,@SFR             ;GENERATE ''CLR CELL''
017776 017737 161744 001126             MOV     @CSR,$BDDAT             ;READ STATUS
020004 012737 000022 001124             MOV     #BIT4!BIT1,$GDDAT       ;LOAD EXPECTED
020012 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
020020 00' 01                           BEQ     2$              ;;BR IF SAME
020022 104031                           ERROR   31                      ;''CLR CELL'' FAILED TO CLEAR ''CELL OVFL'' FLOP


                                ;NOW VERIFY THE BYTE DATA CELL
020024 013737 060000 001126     2$:     MOV     BUF0,$BDDAT             ;READ DATA
020032 012737 100377 001124             MOV     #100377,$GDDAT          ;LOAD EXPECTED
020040 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE
020046 001401                           BEQ     TST116          ;;BR IF SAME
020050 104031                           ERROR   31                      ;OVERFLOW FROM INC. REG. FAILED TO INHIBIT
                                                        ;''INC CNT'' FROM CHANGING THE DATA OR
                                                        ;''CSR BYTE CELLS'' FAILED TO INHIBIT ''INC ENA HI''
```

```
                                        ;;*********************************************************************
                                        ;*TEST 116      CHECK FOR WORD "INC OVFL" TO SET CELL OVERFLOW AND "CLR ALL" TO CLEAR IT
                                        ;;*********************************************************************
020052  000004                  TST116: SCOPE
020054  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
020062  012777  004000  161666          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
020070  005077  161660                  CLR     @BAR
020074  005077  161652                  CLR     @WCR
020100  012777  060000  161642          MOV     #BUF0,@OFF              ;LOAD INITIAL OFFSET REGISTER
020106  012777  000024  161632          MOV     #BIT4!BIT2,@CSR         ;ENABLE MATRIX MODE
020114  012777  000014  161634          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
020122  052777  000001  161616          BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
020130  052777  000002  161620          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
020136  042777  000002  161612          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
020144  000240                          NOP
020146  000240                          NOP
020150  000240                          NOP
020152  012737  177777  060000          MOV     #-1,BUF0                ;SET BUF0 LOC. TO WORD -1
020160  052777  010000  161570          BIS     #BIT12,@SFR             ;ALLOW 1 TRANSFER
020166  000240                          NOP
020170  000240                          NOP
020172  000240                          NOP
020174  017737  161546  001126          MOV     @CSR,$BDDAT             ;READ STATUS
020202  012737  020024  001124          MOV     #BIT13!BIT4!BIT2,$GDDAT ;LOAD EXPECTED STATUS
020210  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
020216  001401                          BEQ     1$              ;;BR IF SAME
020220  104031                          ERROR   31                      ;"CELL OVERFLOW" FLOP FAILED TO SET
                                ;NOW GENERATE "CLR ALL" TO CLEAR CELL OVERFLOW FLOP
020222  052777  004000  161526  1$:     BIS     #CLRALL,@SFR            ;GENERATE "CLR ALL"
020230  017737  161512  001126          MOV     @CSR,$BDDAT             ;READ STATUS
020236  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
020244  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
020252  001401                          BEQ     2$              ;;BR IF SAME
020254  104031                          ERROR   31                      ;"CLR ALL" FAILED TO CLEAR "CELL OVFL" FLOP
                                ;NOW VERIFY THE WORD CELL DATA
020256  013737  060000  001126  2$:     MOV     BUF0,$BDDAT             ;READ DATA
020264  012737  177777  001124          MOV     #-1,$GDDAT              ;LOAD EXPECTED
020272  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
020300  001401                          BEQ     TST117          ;;BR IF SAME
020302  104031                          ERROR   31
```

```
                                        ;;*************************************************************************
                                        ;*TEST 117      CHECK FOR ''CELL OVERFLOW'' INTERRUPT
                                        ;;*************************************************************************
020304 000004                   TST117: SCOPE
020306 012737 000040 001160             MOV     #40,$TIMES      ;;DO 40 ITERATIONS
020314 012777 004000 161434             MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
020322 005077 161426                    CLR     @BAR
020326 005077 161420                    CLR     @WCR
020332 012777 060000 161410             MOV     #BUF0,@OFF              ;LOAD INITIAL OFFSET REGISTER
020340 012777 000064 161400             MOV     #BIT5!BIT4!BIT2,@CSR    ;ENABLE INTR. AND MATRIX MODE
020346 012777 000014 161402             MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
020354 052777 000001 161364             BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
020362 052777 000002 161366             BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
020370 042777 000002 161360             BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
020376 000240                           NOP
020400 000240                           NOP
020402 000240                           NOP
020404 012737 177777 060000             MOV     #-1,BUF0                ;SET BUF0 LOC. TO WORD -1
020412 012746 000000                    MOV     #0,-(SP)
020416 012746 020424                    MOV     #1$,-(SP)
020422 000002                           RTI
020424 012777 020452 161344     1$:     MOV     #2$,@VECTB0             ;LOAD RETURN VECTOR
020432 052777 010000 161316             BIS     #BIT12,@SFR             ;ALLOW 1 TRANSFER
020440 000240                           NOP
020442 000240                           NOP
020444 000240                           NOP
020446 104032                           ERROR   32                      ;''CELL OVERFLOW'' FAILED TO CAUES AN INTERRUPT
020450 000414                           BR      3$              ;;BR TO CLEAN UP
020452 022626                   2$:     CMP     (SP)+,(SP)+
020454 017737 161266 001126             MOV     @CSR,$BDDAT             ;READ STATUS
020462 012737 020264 001124             MOV     #BIT13!BIT7!BIT5!BIT4!BIT2,$GDDAT       ;LOAD EXPECTED
020470 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE VALUE
020476 001401                           BEQ     3$              ;;BR IF ''ACTIVE'' CLEARED
020500 104031                           ERROR   31                      ;''ACTIVE'' FAILED TO CLEAR
020502 012777 004000 161246     3$:     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
020510 013777 002000 161260             MOV     VECTB1,@VECTB0          ;RESET THE VECTORS
020516 005077 161256                    CLR     @VECTB1
```

```
                                        ;;*****************************************************************
                                        ;*TEST 120      VERIFY CORRECT BR LEVEL THE NCV-11
                                        ;;*****************************************************************
020522  000004                          TST120: SCOPE
020524  012737  000040  001160                  MOV     #40,$TIMES      ;;DO 40 ITERATIONS
020532  013746  002002                          MOV     BRLEV,-(SP)     ;PUSH THE EXPECTED BR LEVEL ON STACK
020536  012746  020544                          MOV     #1$,-(SP)       ;PUST THE RETURN ADDRESS ON STACK
020542  000002                                  RTI             ;FAKE AN INTERRUPT TO BE LSI-11 AND PDP-11 COMPATABLE
020544  012777  004000  161204  1$:             MOV     #CLRALL,@SFR    ;CLEAR THE DEVICE
020552  012777  020746  161216                  MOV     #10$,@VECTB0    ;LOAD UNEXPECTED INTERRUPT RETURN
020560  012777  000340  161212                  MOV     #340,@VECTB1
020566  005077  161162                          CLR     @BAR            ;INIT THE LOW Z COUNT
020572  005077  161154                          CLR     @WCR            ;INIT THE HIGH Z COUNT
020576  012777  060000  161144                  MOV     #BUF0,@OFF      ;PRIME THE OFFSET REGISTER
020604  012777  000064  161134                  MOV     #64,@CSR        ;SELECT INTR. AND MATRIX MODE
020612  012777  000014  161136                  MOV     #TSTDMA!TSTCON,@SFR     ;MAINT. MODE
020620  012737  177777  060000                  MOV     #-1,BUF0        ;PRIME THE TARGET LOCATION
020626  052777  000001  161112                  BIS     #BIT0,@CSR      ;ENABLE THE NCV-11
020634  052777  000002  161114                  BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
020642  042777  000002  161106                  BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
020650  000240                                  NOP
020652  000240                                  NOP
020654  000240                                  NOP
020656  052777  010000  161072                  BIS     #BIT12,@SFR     ;ENABLE 1 TRANSFER
020664  000240                                  NOP
020666  000240                                  NOP
020670  000240                                  NOP
020672  000240                                  NOP
020674  000240                                  NOP
020676  000240                                  NOP
020700  013700  002002                          MOV     BRLEV,R0        ;GET CURRENT BR LEVEL
020704  162700  000040                          SUB     #40,R0          ;AND MAKE IT 1 LEVEL LOWER
020710  005737  003354                          TST     NLSI11          ;TEST IF LSI-11 CPU
020714  001401                                  BEQ     2$              ;BR IF NOT
020716  005000                                  CLR     R0              ;LOAD BR LEVEL 0
020720  012777  020754  161050  2$:             MOV     #20$,@VECTB0    ;RELOAD TO EXPECTED VECTOR
020726  010046                                  MOV     R0,-(SP)        ;PUSH ADJUSTED BR LEVEL
020730  012746  020736                          MOV     #3$,-(SP)       ;PUSH RETURN ADDRESS
020734  000002                                  RTI                     ;LOWER CPU BR LEVEL
020736  000240                          3$:     NOP
020740  000240                                  NOP
020742  104042                                  ERROR   42      ;NCV11 FAILED TO INTERRUPT - INCORRECT NCV11 BR LEVEL?
020744  000404                                  BR      21$             ;;BR TO CLEAN-UP
                                        ;UNEXPECTED INTERRUPT WITH BR LEVEL INDICATED
020746  022626                          10$:    CMP     (SP)+,(SP)+     ;CLEAN THE STACK
020750  104042                                  ERROR   42              ;NCV11 INTERRUPTED ON AN INCORRECT BR LEVEL
020752  000401                                  BR      21$             ;;BR TO CLEANUP
                                        ;EXPECTED INTERRUPT DID OCCUR
020754  022626                          20$:    CMP     (SP)+,(SP)+     ;CLEAN THE STACK
020756  012777  004000  160772  21$:            MOV     #CLRALL,@SFR    ;CLEAN THE DEVICE
020764  013777  002000  161004                  MOV     VECTB1,@VECTB0  ;RESET THE VECTOR
020772  005077  161002                          CLR     @VECTB1
```

```
                                ;;*****************************************************************
                                ;*TEST 121       VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 13
                                ;;*****************************************************************
020776  000004                  TST121: SCOPE
021000  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
021006  012777  004000  160742          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
021014  012777  060000  160726          MOV     #60000,@OFF             ;LOAD THE OFFSET REG.
021022  012777  000014  160726          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
021030  012777  000024  160710          MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
021036  052777  000001  160702          BIS     #BIT0,@CSR              ;ENABLE NCV11
021044  052777  000002  160704          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
021052  042777  000002  160676          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
021060  005037  060000                  CLR     @#60000                 ;CLEAR THE TARGET LOCATION
021064  052777  010000  160664          BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
021072  012737  060000  001122          MOV     #60000,$BDADR           ;LOAD EXPECTED ADDRESS
021100  012737  000001  001124          MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
021106  017737  160010  001126          MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
021114  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
021122  001401                        BEQ     TST122          ;;BR IF SAME
021124  104033                        ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                ;THE WRONG ADDRESS - EXPECTED ADR. WAS 60000
                                ;;*****************************************************************
                                ;*TEST 122       VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 12
                                ;;*****************************************************************
021126  000004                  TST122: SCOPE
021130  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
021136  012777  004000  160612          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
021144  012777  070000  160576          MOV     #70000,@OFF             ;LOAD THE OFFSET REG.
021152  012777  000014  160576          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
021160  012777  000024  160560          MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
021166  052777  000001  160552          BIS     #BIT0,@CSR              ;ENABLE NCV11
021174  052777  000002  160554          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
021202  042777  000002  160546          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
021210  005037  070000                  CLR     @#70000                 ;CLEAR THE TARGET LOCATION
021214  052777  010000  160534          BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
021222  012737  070000  001122          MOV     #70000,$BDADR           ;LOAD EXPECTED ADDRESS
021230  012737  000001  001124          MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
021236  017737  157660  001126          MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
021244  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
021252  001401                        BEQ     TST123          ;;BR IF SAME
021254  104033                        ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                ;THE WRONG ADDRESS - EXPECTED ADR. WAS 70000
```

```
                                        ;;*****************************************************************
                                        ;*TEST 123      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 11
                                        ;;*****************************************************************
021256  000004                          TST123: SCOPE
021260  012737  000100  001160                  MOV     #100,$TIMES     ;;DO 100 ITERATIONS
021266  012777  004000  160462                  MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
021274  012777  064000  160446                  MOV     #64000,@OFF             ;LOAD THE OFFSET REG.
021302  012777  000014  160446                  MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
021310  012777  000024  160430                  MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
021316  052777  000001  160422                  BIS     #BIT0,@CSR              ;ENABLE NCV11
021324  052777  000002  160424                  BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
021332  042777  000002  160416                  BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
021340  005037  064000                          CLR     @#64000                 ;CLEAR THE TARGET LOCATION
021344  052777  010000  160404                  BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
021352  012737  064000  001122                  MOV     #64000,$BDADR           ;LOAD EXPECTED ADDRESS
021360  012737  000001  001124                  MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
021366  017737  157530  001126                  MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
021374  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;COMPARE
021402  001401                                  BEQ     TST124          ;;BR IF SAME
021404  104033                                  ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                        ;THE WRONG ADDRESS - EXPECTED ADR. WAS 64000
                                        ;;*****************************************************************
                                        ;*TEST 124      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 10
                                        ;;*****************************************************************
021406  000004                          TST124: SCOPE
021410  012737  000100  001160                  MOV     #100,$TIMES     ;;DO 100 ITERATIONS
021416  012777  004000  160332                  MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
021424  012777  062000  160316                  MOV     #62000,@OFF             ;LOAD THE OFFSET REG.
021432  012777  000014  160316                  MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
021440  012777  000024  160300                  MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
021446  052777  000001  160272                  BIS     #BIT0,@CSR              ;ENABLE NCV11
021454  052777  000002  160274                  BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
021462  042777  000002  160266                  BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
021470  005037  062000                          CLR     @#62000                 ;CLEAR THE TARGET LOCATION
021474  052777  010000  160254                  BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
021502  012737  062000  001122                  MOV     #62000,$BDADR           ;LOAD EXPECTED ADDRESS
021510  012737  000001  001124                  MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
021516  017737  157400  001126                  MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
021524  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;COMPARE
021532  001401                                  BEQ     TST125          ;;BR IF SAME
021534  104033                                  ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                        ;THE WRONG ADDRESS - EXPECTED ADR. WAS 62000
```

```
                                        ;;*****************************************************************
                                        ;*TEST 125      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 09
                                        ;;*****************************************************************
021536  000004                          TST125: SCOPE
021540  012737  000100  001160                MOV     #100,$TIMES     ;;DO 100 ITERATIONS
021546  012777  004000  160202                MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
021554  012777  061000  160166                MOV     #61000,@OFF             ;LOAD THE OFFSET REG.
021562  012777  000014  160166                MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
021570  012777  000024  160150                MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
021576  052777  000001  160142                BIS     #BIT0,@CSR              ;ENABLE NCV11
021604  052777  000002  160144                BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
021612  042777  000002  160136                BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
021620  005037  061000                       CLR     @#61000                 ;CLEAR THE TARGET LOCATION
021624  052777  010000  160124                BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
021632  012737  061000  001122                MOV     #61000,$BDADR           ;LOAD EXPECTED ADDRESS
021640  012737  000001  001124                MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
021646  017737  157250  001126                MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
021654  023737  001124  001126                CMP     $GDDAT,$BDDAT           ;COMPARE
021662  001401                               BEQ     TST126          ;;BR IF SAME
021664  104033                               ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                        ;THE WRONG ADDRESS - EXPECTED ADR. WAS 61000
                                        ;;*****************************************************************
                                        ;*TEST 126      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 08
                                        ;;*****************************************************************
021666  000004                          TST126: SCOPE
021670  012737  000100  001160                MOV     #100,$TIMES     ;;DO 100 ITERATIONS
021676  012777  004000  160052                MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
021704  012777  060400  160036                MOV     #60400,@OFF             ;LOAD THE OFFSET REG.
021712  012777  000014  160036                MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
021720  012777  000024  160020                MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
021726  052777  000001  160012                BIS     #BIT0,@CSR              ;ENABLE NCV11
021734  052777  000002  160014                BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
021742  042777  000002  160006                BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
021750  005037  060400                       CLR     @#60400                 ;CLEAR THE TARGET LOCATION
021754  052777  010000  157774                BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
021762  012737  060400  001122                MOV     #60400,$BDADR           ;LOAD EXPECTED ADDRESS
021770  012737  000001  001124                MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
021776  017737  157120  001126                MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
022004  023737  001124  001126                CMP     $GDDAT,$BDDAT           ;COMPARE
022012  001401                               BEQ     TST127          ;;BR IF SAME
022014  104033                               ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                        ;THE WRONG ADDRESS - EXPECTED ADR. WAS 60400
```

```
2821                                    ;;*****************************************************************
2822                                    ;*TEST 127      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 07
2823                                    ;;*****************************************************************
2824  022016  000004                   TST127: SCOPE
2825  022020  012737  000100  001160           MOV     #100,$TIMES     ;;DO 100 ITERATIONS
2826  022026  012777  004000  157722           MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
2827  022034  012777  060200  157706           MOV     #60200,@OFF             ;LOAD THE OFFSET REG.
2828  022042  012777  000014  157706           MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
2829  022050  012777  000024  157670           MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
2830  022056  052777  000001  157662           BIS     #BIT0,@CSR              ;ENABLE NCV11
2831  022064  052777  000002  157664           BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
2832  022072  042777  000002  157656           BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
2833  022100  005037  060200                   CLR     @#60200                 ;CLEAR THE TARGET LOCATION
2834  022104  052777  010000  157644           BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
2835  022112  012737  060200  001122           MOV     #60200,$BDADR           ;LOAD EXPECTED ADDRESS
2836  022120  012737  000001  001124           MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
2837  022126  017737  156770  001126           MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
2838  022134  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE
2839  022142  001401                           BEQ     TST130          ;;BR IF SAME
2840  022144  104033                           ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
2841                                                                   ;THE WRONG ADDRESS - EXPECTED ADR. WAS 60200
2842                                    ;;*****************************************************************
2843                                    ;*TEST 130      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 06
2844                                    ;;*****************************************************************
2845  022146  000004                   TST130: SCOPE
2846  022150  012737  000100  001160           MOV     #100,$TIMES     ;;DO 100 ITERATIONS
2847  022156  012777  004000  157572           MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
2848  022164  012777  060100  157556           MOV     #60100,@OFF             ;LOAD THE OFFSET REG.
2849  022172  012777  000014  157556           MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
2850  022200  012777  000024  157540           MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
2851  022206  052777  000001  157532           BIS     #BIT0,@CSR              ;ENABLE NCV11
2852  022214  052777  000002  157534           BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
2853  022222  042777  000002  157526           BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
2854  022230  005037  060100                   CLR     @#60100                 ;CLEAR THE TARGET LOCATION
2855  022234  052777  010000  157514           BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
2856  022242  012737  060100  001122           MOV     #60100,$BDADR           ;LOAD EXPECTED ADDRESS
2857  022250  012737  000001  001124           MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
2858  022256  017737  156640  001126           MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
2859  022264  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE
2860  022272  001401                           BEQ     TST131          ;;BR IF SAME
2861  022274  104033                           ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
2862                                                                   ;THE WRONG ADDRESS - EXPECTED ADR. WAS 60100
```

```
                                        ;;*******************************************************************
                                        ;*TEST 131      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 05
                                        ;;*******************************************************************
022276  000004                          TST131: SCOPE
022300  012737  000100  001160                  MOV     #100,$TIMES     ;;DO 100 ITERATIONS
022306  012777  004000  157442                  MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
022314  012777  060040  157426                  MOV     #60040,@OFF             ;LOAD THE OFFSET REG.
022322  012777  000014  157426                  MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
022330  012777  000024  157410                  MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
022336  052777  000001  157402                  BIS     #BIT0,@CSR              ;ENABLE NCV11
022344  052777  000002  157404                  BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
022352  042777  000002  157376                  BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
022360  005037  060040                        CLR     @#60040                 ;CLEAR THE TARGET LOCATION
022364  052777  010000  157364                  BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
022372  012737  060040  001122                  MOV     #60040,$BDADR           ;LOAD EXPECTED ADDRESS
022400  012737  000001  001124                  MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
022406  017737  156510  001126                  MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
022414  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;COMPARE
022422  001401                                  BEQ     TST132          ;;BR IF SAME
022424  104033                                  ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                        ;THE WRONG ADDRESS - EXPECTED ADR. WAS 60040
                                        ;;*******************************************************************
                                        ;*TEST 132      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 04
                                        ;;*******************************************************************
022426  000004                          TST132: SCOPE
022430  012737  000100  001160                  MOV     #100,$TIMES     ;;DO 100 ITERATIONS
022436  012777  004000  157312                  MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
022444  012777  060020  157276                  MOV     #60020,@OFF             ;LOAD THE OFFSET REG.
022452  012777  000014  157276                  MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
022460  012777  000024  157260                  MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
022466  052777  000001  157252                  BIS     #BIT0,@CSR              ;ENABLE NCV11
022474  052777  000002  157254                  BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
022502  042777  000002  157246                  BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
022510  005037  060020                        CLR     @#60020                 ;CLEAR THE TARGET LOCATION
022514  052777  010000  157234                  BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
022522  012737  060020  001122                  MOV     #60020,$BDADR           ;LOAD EXPECTED ADDRESS
022530  012737  000001  001124                  MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
022536  017737  156360  001126                  MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
022544  023737  001124  001126                  CMP     $GDDAT,$BDDAT           ;COMPARE
022552  001401                                  BEQ     TST133          ;;BR IF SAME
022554  104033                                  ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                        ;THE WRONG ADDRESS - EXPECTED ADR. WAS 60020
```

```
                                        ;;*********************************************************************
                                        ;*TEST 133      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 03
                                        ;;*********************************************************************
022556  000004                        TST133: SCOPE
022560  012737  000100  001160                MOV     #100,$TIMES     ;;DO 100 ITERATIONS
022566  012777  004000  157162                MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
022574  012777  060010  157146                MOV     #60010,@OFF             ;LOAD THE OFFSET REG.
022602  012777  000014  157146                MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
022610  012777  000024  157130                MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
022616  052777  000001  157122                BIS     #BIT0,@CSR              ;ENABLE NCV11
022624  052777  000002  157124                BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
022632  042777  000002  157116                BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
022640  005037  060010                      CLR     @#60010                 ;CLEAR THE TARGET LOCATION
022644  052777  010000  157104                BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
022652  012737  060010  001122                MOV     #60010,$BDADR           ;LOAD EXPECTED ADDRESS
022660  012737  000001  001124                MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
022666  017737  156230  001126                MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
022674  023737  001124  001126                CMP     $GDDAT,$BDDAT           ;COMPARE
022702  001401                              BEQ     TST134          ;;BR IF SAME
022704  104033                              ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                    ;THE WRONG ADDRESS - EXPECTED ADR. WAS 60010
                                        ;;*********************************************************************
                                        ;*TEST 134      VERIFY A 1 WORD MATRIX MODE XFR USING OFFSET  - 02
                                        ;;*********************************************************************
022706  000004                        TST134: SCOPE
022710  012737  000100  001160                MOV     #100,$TIMES     ;;DO 100 ITERATIONS
022716  012777  004000  157032                MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
022724  012777  040004  157016                MOV     #40004,@OFF             ;LOAD THE OFFSET REG.
022732  012777  000014  157016                MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
022740  012777  000024  157000                MOV     #BIT4!BIT2,@CSR         ;SELECT MATRIX WORD MODE
022746  052777  000001  156772                BIS     #BIT0,@CSR              ;ENABLE NCV11
022754  052777  000002  156774                BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
022762  042777  000002  156766                BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
022770  005037  040004                      CLR     @#40004                 ;CLEAR THE TARGET LOCATION
022774  052777  010000  156754                BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
023002  012737  040004  001122                MOV     #40004,$BDADR           ;LOAD EXPECTED ADDRESS
023010  012737  000001  001124                MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
023016  017737  156100  001126                MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
023024  023737  001124  001126                CMP     $GDDAT,$BDDAT           ;COMPARE
023032  001401                              BEQ     TST135          ;;BR IF SAME
023034  104033                              ERROR   33                      ;OFFSET INPUT TO MATRIX MUX. SELECTED
                                                                    ;THE WRONG ADDRESS - EXPECTED ADR. WAS 40004
```

```
                                ;;*****************************************************************
                                ;*TEST 135      VERIFY THE ADM INPUT TO THE MATRIX MUX. USING GAIN ENABLE
                                ;;*****************************************************************
023036  000004                  TST135: SCOPE
023040  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
                                ;WITH GAIN SET IN TEST CONTROL MODE, THE SELF TEST DATA SHOULD BE BYTE 250
                                ;WHEN PROCESSED THRU THE ADDRESS MAKER LOGIC THE NEW VALUE IS '2552'
                                ;VERIFY THAT THE MATRIX MUX CAN ADD 62550 AND 2552 CORRECTLY
023046  012777  004000  156702          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
023054  012777  062550  156666          MOV     #BUF0+2550,@OFF         ;LOAD THE OFFSET REG.
023062  012777  000014  156666          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
023070  012777  002024  156650          MOV     #2024,@CSR              ;ENABLE GAIN, WORD MATRIX MODE
023076  052777  000001  156642          BIS     #BIT0,@CSR              ;ENABLE THE NCV11
023104  052777  000002  156644          BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
023112  042777  000002  156636          BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
023120  005037  065322                  CLR     @#BUF0+5322             ;CLEAR THE TARGET ADDRESS
023124  005037  062722                  CLR     @#BUF0+2722             ;CLEAR THE TARGET ADDRESS IF ''TESTX'' FAILS
023130  052777  010000  156620          BIS     #BIT12,@SFR             ;ALLOW 1 TRANSFER
023136  000240                          NOP
023140  000240                          NOP
023142  000240                          NOP
023144  012737  065322  001122          MOV     #BUF0+5322,$BDADR       ;LOAD THE EXPECTED ADDRESS
023152  012737  000001  001124          MOV     #1,$GDDAT               ;LOAD THE EXPECTED DATA
023160  017737  155736  001126          MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
023166  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
023174  001413                          BEQ     TST136          ;;BR IF SAME
023176  005737  062722                  TST     @#BUF0+2722             ;TEST OTHER ADDRESS
023202  001002                          BNE     1$                      ;BR IF NON-ZERO
023204  104033                          ERROR   33                      ;MATRIX MODE ADDER ERROR
                                ;USING GAIN AND WORD MATRIX MODE, THE ADDRESSES SELECTED WAS INCORRECT
023206  000406                  BR      TST136          ;;
023210  013737  062722  001126  1$:     MOV     @#BUF0+2722,$BDDAT      ;LOAD INCORRECT DATA
023216  005037  001124                  CLR     $GDDAT                  ;CLEAR THE EXPECTED DATA
023222  104033                          ERROR   33                      ;MATRIX MODE ADDER ERROR DUE TO
                                                                ;''TESTX'' LOGIC SET IN ERROR
```

```
                                        ;;*****************************************************************
                                        ;*TEST 136      VERIFY THE ADM INPUT TO THE MATRIX MUX. ADDER  USING ZB ENABLE
                                        ;;*****************************************************************
023224  000004                  TST136: SCOPE
023226  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
                                    ;WITH ZB ENABLE SET IN TEST CONTROL MODE, THE SELF TEST DATA SHOULD BE BYTE 224
                                    ;WHEN PROCESSED THRU THE ADDRESS MAKER LOGIC THE NEW VALUE IS 12450
                                    ;VERIFY THAT THE MATRIX MUX CAN ADD CORRECTLY
023234  012777  004000  156514          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
023242  012777  065224  156500          MOV     #BUF0+5224,@OFF         ;LOAD THE OFFSET REGISTER
023250  012777  000014  156500          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
023256  012777  004024  156462          MOV     #4024,@CSR              ;ENABLE ZB AND WORD MATRIX MODE
023264  052777  000002  156464          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
023272  052777  000001  156446          BIS     #BIT0,@CSR              ;ENABLE NCV11
023300  042777  000001  156450          BIC     #BIT0,@SFR              ;DISABLE "TEST Z" PULSES
023306  005037  072450                  CLR     @#BUF0+12450            ;CLEAR THE TARGET LOCATION
023312  052777  010000  156436          BIS     #BIT12,@SFR             ;ALLOW 1 TRANSFER
023320  012737  072450  001122          MOV     #BUF0+12450,$BDADR      ;LOAD THE EXPECTED ADDRESS
023326  012737  000001  001124          MOV     #1,$GDDAT               ;LOAD THE EXPECTED DATA
023334  017737  155562  001126          MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
023342  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
023350  001401                        BEQ     TST137          ;;BR IF SAME
023352  104033                        ERROR   33              ;ADM INPUT TO THE MATRIX MUX SELECTED
                                                                ;WRONG ADDRESS  -  EXPECTED ADDRESS WAS BUF0 + 2450
                                        ;;*****************************************************************
                                        ;*TEST 137      VERIFY LOW BYTE OPERATION OF THE "TESTX" FLOP
                                        ;;*****************************************************************
023354  000004                  TST137: SCOPE
023356  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
023364  012777  004000  156364          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
023372  012777  060000  156350          MOV     #BUF0,@OFF              ;LOAD OFFSET REG.
023400  012777  000014  156350          MOV     #TSTDMA!TSTCON,@SFR     ;LOAD TEST DMA AND CONTROL
023406  012777  002024  156332          MOV     #2024,@CSR              ;SET "GAIN" AND WORD MATRIX MODE
023414  052777  000001  156324          BIS     #BIT0,@CSR              ;SET NCV11 ACTIVE
023422  052777  000002  156326          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
023430  042777  000002  156320          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
023436  052777  000020  156312          BIS     #BIT4,@SFR              ;SET "TESTX" FLOP
023444  005037  062552                  CLR     @#BUF0+2552             ;CLEAR THE TEST FAILED LOCATION
023450  005037  060152                  CLR     @#BUF0+0152             ;CLEAR THE TESTX WORKED LOCATION
023454  052777  010000  156274          BIS     #BIT12,@SFR             ;ALLOW 1 TRANSFER
023462  000240                        NOP
023464  000240                        NOP
023466  000240                        NOP
023470  012737  000001  001124          MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
023476  013737  060152  001126          MOV     @#BUF0+0152,$BDDAT      ;READ THE ACTUAL DATA
023504  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE DATA
023512  001413                        BEQ     TST140          ;;BR IF SAME
023514  005737  062552                  TST     @#BUF0+2552             ;TEST TESTX FAILED LOCATION
023520  001002                        BNE     1$              ;;BR IF YES
023522  104034                        ERROR   34              ;TESTX FAILED TO INHIBIT BITS 8-15 OF THE "SUM ADDER"
023524  000406                        BR      TST140          ;;
023526  013737  062552  001126  1$:     MOV     @#BUF0+2552,$BDDAT      ;GET ACTUAL DATA
023534  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED
023540  104034                        ERROR   34                      ;TESTX FAILED TO INHIBIT BITS 8-15 OF THE "SUM A
```

```
3036                                    ;;*****************************************************************
3037                                    ;*TEST 140      VERIFY BIT 12 ADDER INPUT TO MATRIX MODE MUX
3038                                    ;;*****************************************************************
3039  023542  000004                    TST140: SCOPE
3040  023544  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
3041  023552  012777  004000  156176            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
3042  023560  012777  050000  156162            MOV     #50000,@OFF             ;LOAD THE OFFSET REG.
3043  023566  012777  000014  156162            MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTRO_
3044  023574  012777  002030  156144            MOV     #2030,@CSR              ;SELECT GAIN MATRIX BYTE MODE
3045  023602  052777  000001  156136            BIS     #BIT0,@CSR              ;ENABLE NCV11
3046  023610  052777  000002  156140            BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
3047  023616  042777  000002  156132            BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
3048  023624  005037  062726                    CLR     @#BUF0+2726             ;CLEAR THE TARGET LOCATION
3049  023630  052777  010000  156120            BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
3050  023636  012737  062726  001122            MOV     #BUF0+2726,$BDADR       ;LOAD EXPECTED ADDRESS
3051  023644  012737  000001  001124            MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
3052  023652  017737  155244  001126            MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
3053  023660  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
3054  023666  001401                            BEQ     TST141          ;;BR IF SAME
3055  023670  104033                            ERROR   33                      ;MATRIX ADDER FAILED TO SELECT CORRECT ADDRESS
3056                                                                    ;THE WRONG ADDRESS - EXPECTED ADR. WAS BUF0+2726
3057                                    ;;*****************************************************************
3058                                    ;*TEST 141      VERIFY BIT 13 ADDER INPUT TO MATRIX MODE MUX
3059                                    ;;*****************************************************************
3060  023672  000004                    TST141: SCOPE
3061  023674  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
3062  023702  012777  004000  156046            MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
3063  023710  012777  037774  156032            MOV     #37774,@OFF             ;LOAD THE OFFSET REG.
3064  023716  012777  000014  156032            MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
3065  023724  012777  002032  156014            MOV     #2032,@CSR              ;SELECT GAIN MATRIX BYTE MODE
3066  023732  052777  000001  156006            BIS     #BIT0,@CSR              ;ENABLE NCV11
3067  023740  052777  000002  156010            BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
3068  023746  042777  000002  156002            BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
3069  023754  005037  065522                    CLR     @#BUF0+5522             ;CLEAR THE TARGET LOCATION
3070  023760  052777  010000  155770            BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
3071  023766  012737  065522  001122            MOV     #BUF0+5522,$BDADR       ;LOAD EXPECTED ADDRESS
3072  023774  012737  000001  001124            MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
3073  024002  017737  155114  001126            MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
3074  024010  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
3075  024016  001401                            BEQ     TST142          ;;BR IF SAME
3076  024020  104033                            ERROR   33                      ;MATRIX ADDER FAILED TO SELECT CORRECT ADDRESS
3077                                                                    ;THE WRONG ADDRESS - EXPECTED ADR. WAS BUF0+5522
```

```
                                        ;;*********************************************************************
                                        ;*TEST 142       VERIFY BIT 14 ADDER INPUT TO MATRIX MODE MUX
                                        ;;*********************************************************************
                                        TST142: SCOPE
                                                MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                        ;OFFSET =       50000
                                        ;ADM OUTPUT     =       53656
                                        ;TARGET         =       123656
                                                MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
                                                CMP     $LSTBK,#1340            ;TEST IF ENOUGH MEMORY >20K
                                                BLO     TST143          ;;BR IF NO ROOM
                                                MOV     #50000,@OFF             ;LOAD THE OFFSET REG.
                                                MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
                                                MOV     #2034,@CSR              ;SELECT GAIN MATRIX BYTE MODE
                                                BIS     #BIT0,@CSR              ;ENABLE NCV11
                                                BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z' PULSES
                                                BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z' PULSES
                                                CLR     @#BUF0+43656            ;CLEAR THE TARGET LOCATION
                                                BIS     #BIT12,@SFR             ;ALLOW 1 WORD TRANSFER
                                                MOV     #BUF0+43656,$BDADR      ;LOAD EXPECTED ADDRESS
                                                MOV     #1,$GDDAT               ;LOAD EXPECTED DATA
                                                MOV     @$BDADR,$BDDAT          ;READ THE ACTUAL DATA
                                                CMP     $GDDAT,$BDDAT           ;COMPARE
                                                BEQ     TST143          ;;BR IF SAME
                                                ERROR   33                      ;MATRIX ADDER FAILED TO SELECT CORRECT ADDRESS
                                                                        ;THE WRONG ADDRESS - EXPECTED ADR. WAS BUF0+43656
```

```
                                        ;;*****************************************************************
                                        ;*TEST 143      CHECK FOR HIGH BYTE "INC OVFL" TO SET CELL OVERFLOW
                                        ;;*****************************************************************
024162  000004                  TST143: SCOPE
024164  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
024172  012777  004000  155556          MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
024200  005077  155546                  CLR     @WCR                    ;CLEAR W.C.
024204  005077  155544                  CLR     @BAR                    ;CLEAR B.A.
024210  012777  060000  155532          MOV     #BUF0,@OFF              ;LOAD INITIAL OFFSET REGISTER
024216  012777  002022  155522          MOV     #BIT10!BIT4!BIT1,@CSR   ;ENABLE MATRIX MODE BYTE AND SET GAIN
024224  012777  000014  155524          MOV     #TSTDMA!TSTCON,@SFR     ;SET TEST DMA AND CONTROL
024232  052777  000001  155506          BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
024240  052777  000002  155510          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
024246  042777  000002  155502          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
024254  000240                          NOP
024256  000240                          NOP
024260  000240                          NOP
024262  112737  000100  061264          MOVB    #100,BUF0+1264          ;LOAD LOW BYTE TO A KNOWN VALUE
024270  112737  000377  061265          MOVB    #377,BUF0+1265          ;SET HIGH BYTE TO 377
024276  052777  010000  155452          BIS     #BIT12,@SFR             ;ALLOW 1 TRANSFER
024304  000240                          NOP
024306  000240                          NOP
024310  000240                          NOP
024312  012737  022022  001124          MOV     #BIT13!BIT10!BIT4!BIT1,$GDDAT   ;LOAD EXPECTED STATUS
024320  017737  155422  001126          MOV     @CSR,$BDDAT             ;READ THE ACTUAL STATUS
024326  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
024334  001401                          BEQ     1$              ;;BR IF SAME
024336  104030                          ERROR   30              ;CELL OVERFLOW FAILED TO SET
                                                                ;WHEN INCREMENTED THE HIGH BYTE
024340  013737  061264  001126  1$:     MOV     BUF0+1264,$BDDAT        ;READ BUF0 LOCATION
024346  012737  177500  001124          MOV     #177500,$GDDAT          ;LOAD EXPECTED VALUE
024354  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE VALUES
024362  001401                          BEQ     TST144          ;;BR IF SAME
024364  104030                          ERROR   30              ;TARGET LOC. DATA WAS INCORRECT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 144      VERIFY EACH BIT OF THE LOWER 16 BIT Z COUNTER   (TESTER ONLY)
                                        ;;*****************************************************************
024366 000004                           TST144: SCOPE
024370 012737 000400 001160                     MOV     #400,$TIMES     ;;DO 400 ITERATIONS
024376 005737 050110                            TST     WFMODE                  ;TEST IF TESTER MODE
024402 001454                                   BEQ     TST145          ;;BR IF NOT
024404 012737 000001 001124                     MOV     #1,$GDDAT               ;LOAD EXPECTED VALUE
024412 012737 024420 001110                     MOV     #1$,$LPERR              ;LOAD LOOP ADDRESS
                                
024420 012777 004000 155330 1$:         MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
024426 012777 000010 155322                     MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
024434 012777 000000 155310                     MOV     #0,@WCR                 ;LOAD UPPER 16 BITS
024442 013777 001124 155304                     MOV     $GDDAT,@BAR             ;
024450 005377 155300                            DEC     @BAR                    ;LOAD INITIAL COUNTER VALUE
024454 012777 000022 155264                     MOV     #BIT4!BIT1,@CSR         ;ENTER MATRIX MODE
024462 052777 000001 155256                     BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
024470 052777 000001 155230                     BIS     #BIT0,@DACSR            ;GENERATE "Z" PULSE
024476 105777 155224                    2$:     TSTB    @DACSR                  ;WAIT FOR Z PULSE COMPLETION
024502 100375                                   BPL     2$
024504 017737 155244 001126                     MOV     @BAR,$BDDAT             ;READ EXPECTED REGISTER VALUE
024512 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE
024520 001402                                   BEQ     3$              ;;BR IF SAME
024522 104010                                   FRROR   10              ;LOWER 16 BITS OF THE Z COUNTER IN ERROR
024524 000403                                   BR      TST145          ;;
024526 006337 001124                    3$:     ASL     $GDDAT                  ;CHANGE THE DATA BIT
024532 001332                                   BNE     1$                      ;BR AND TRY NEXT BIT
                                        ;;*****************************************************************
                                        ;*TEST 145      VERIFY EACH BIT OF THE HIGHER 16 BIT Z COUNTER  (TESTER ONLY)
                                        ;;*****************************************************************
024534 000004                           TST145: SCOPE
024536 012737 000400 001160                     MOV     #400,$TIMES     ;;DO 400 ITERATIONS
024544 005737 050110                            TST     WFMODE                  ;TEST IF TESTER MODE
024550 001454                                   BEQ     TST146          ;;BR IF NOT
024552 012737 000001 001124                     MOV     #1,$GDDAT               ;LOAD EXPECTED VALUE
024560 012737 024566 001110                     MOV     #1$,$LPERR              ;LOAD LOOP ADDRESS
                                
024566 012777 004000 155162 1$:         MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
024574 012777 000010 155154                     MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
024602 013777 001124 155142                     MOV     $GDDAT,@WCR             ;
024610 005377 155136                            DEC     @WCR                    ;LOAD INITIAL COUNTER VALUE
024614 012777 177777 155132                     MOV     #-1,@BAR                ;LOAD LOWER 16 BITS
024622 012777 000022 155116                     MOV     #BIT4!BIT1,@CSR         ;ENTER MATRIX MODE
024630 052777 000001 155110                     BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
024636 052777 000001 155062                     BIS     #BIT0,@DACSR            ;GENERATE "Z" PULSE
024644 105777 155056                    2$:     TSTB    @DACSR                  ;WAIT FOR Z PULSE COMPLETION
024650 100375                                   BPL     2$
024652 017737 155074 001126                     MOV     @WCR,$BDDAT             ;READ EXPECTED REGISTER VALUE
024660 023737 001124 001126                     CMP     $GDDAT,$BDDAT           ;COMPARE
024666 001402                                   BEQ     3$              ;;BR IF SAME
024670 104011                                   ERROR   11              ;HIGHER 16 BITS OF THE Z COUNTER IN ERROR
024672 000403                                   BR      TST146          ;;
024674 006337 001124                    3$:     ASL     $GDDAT                  ;CHANGE THE DATA BIT
024700 001332                                   BNE     1$                      ;BR AND TRY NEXT BIT
```

```
                                ;;*****************************************************************
                                ;*TEST 146       VERIFY THAT CAMERA 01 CHANNEL IS OPERATIONAL    (TESTER ONLY)
                                ;;*****************************************************************
024702 000004                   TST146: SCOPE
024704 012737 000400 001160             MOV     #400,$TIMES     ;;DO 400 ITERATIONS
024712 005737 050110                    TST     WFMODE                  ;TEST IF TESTER MODE
024716 001443                           BEQ     TST147          ;;BR IF NOT
024720 012737 000001 001124             MOV     #1,$GDDAT               ;LOAD EXPECTED VALUE

024726 012777 004000 155022     1$:     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
024734 012777 000010 155014             MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
024742 012777 000000 155002             MOV     #0,@WCR                 ;LOAD UPPER 16 BITS
024750 012777 000000 154776             MOV     #0,@BAR                 ;LOAD LOWER 16 BITS
024756 012777 000422 154762             MOV     #BIT8!BIT4!BIT1,@CSR    ;ENTER MATRIX MODE ON CAMERA 01
024764 052777 000001 154754             BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
024772 052777 000001 154726             BIS     #BIT0,@DACSR            ;GENERATE "Z" PULSE
025000 105777 154722            2$:     TSTB    @DACSR                  ;WAIT FOR Z PULSE COMPLETION
025004 100375                           BPL     2$
025006 017737 154742 001126             MOV     @BAR,$BDDAT             ;READ EXPECTED REGISTER VALUE
025014 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE
025022 001401                           BEQ     TST147          ;;BR IF SAME
025024 104010                           ERROR   10                      ;LOWER 16 BITS OF THE Z COUNTER
                                                                ;IN ERROR WHEN USING CAMERA 01

                                ;;*****************************************************************
                                ;*TEST 147       VERIFY THAT CAMERA 10 CHANNEL IS OPERATIONAL    (TESTER ONLY)
                                ;;*****************************************************************
025026 000004                   TST147: SCOPE
025030 012737 000100 001160             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
025036 005737 050110                    TST     WFMODE                  ;TEST IF TESTER MODE
025042 001443                           BEQ     TST150          ;;BR IF NOT
025044 012737 000001 001124             MOV     #1,$GDDAT               ;LOAD EXPECTED VALUE

025052 012777 004000 154676     1$:     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
025060 012777 000010 154670             MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
025066 012777 000000 154656             MOV     #0,@WCR                 ;LOAD UPPER 16 BITS
025074 012777 000000 154652             MOV     #0,@BAR                 ;LOAD LOWER 16 BITS
025102 012777 001022 154636             MOV     #BIT9!BIT4!BIT1,@CSR    ;ENTER MATRIX MODE
025110 052777 000001 154630             BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
025116 052777 000001 154602             BIS     #BIT0,@DACSR            ;GENERATE "Z" PULSE
025124 105777 154576            2$:     TSTB    @DACSR                  ;WAIT FOR Z PULSE COMPLETION
025130 100375                           BPL     2$
025132 017737 154616 001126             MOV     @BAR,$BDDAT             ;READ EXPECTED REGISTER VALUE
025140 023737 001124 001126             CMP     $GDDAT,$BDDAT           ;COMPARE
025146 001401                           BEQ     TST150          ;;BR IF SAME
025150 104010                           ERROR   10                      ;LOWER 16 BITS OF THE Z COUNTER
                                                                ;IN ERROR WHEN USING CAMERA 10
```

```
3239                                    ;;*****************************************************************
3240                                    ;*TEST 150      VERIFY THAT CAMERA 11 CHANNEL IS OPERATIONAL    (TESTER ONLY)
3241                                    ;;*****************************************************************
3242  025152  000004                    TST150: SCOPE
3243  025154  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
3244  025162  005737  050110                    TST     WFMODE                  ;TEST IF TESTER MODE
3245  025166  001443                    BEQ     TST151          ;;BR IF NOT
3246  025170  012737  000001  001124            MOV     #1,$GDDAT               ;LOAD EXPECTED VALUE
3247
3248  025176  012777  004000  154552  1$:       MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
3249  025204  012777  000010  154544            MOV     #TSTDMA,@SFR            ;SET TEST DMA FLOP
3250  025212  012777  000000  154532            MOV     #0,@WCR                 ;LOAD UPPER 16 BITS
3251  025220  012777  000000  154526            MOV     #0,@BAR                 ;LOAD LOWER 16 BITS
3252  025226  012777  001422  154512            MOV     #BIT9!BIT8!BIT4!BIT1,@CSR       ;ENTER MATRIX MODE
3253  025234  052777  000001  154504            BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
3254  025242  052777  000001  154456            BIS     #BIT0,@DACSR            ;GENERATE "Z" PULSE
3255  025250  105777  154452          2$:       TSTB    @DACSR                  ;WAIT FOR Z PULSE COMPLETION
3256  025254  100375                    BPL     2$
3257  025256  017737  154472  001126            MOV     @BAR,$BDDAT             ;READ EXPECTED REGISTER VALUE
3258  025264  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
3259  025272  001401                    BEQ     TST151          ;;BR IF SAME
3260  025274  104010                    ERROR   10                      ;LOWER 16 BITS OF THE Z COUNTER
3261                                                                    ;IN ERROR WHEN USING CAMERA 11
```

```
                                        ;;*****************************************************************
                                        ;*TEST 151      DYNAMIC MATRIX MODE ADDRESS
                                        ;;*****************************************************************
025276  000004                          TST151: SCOPE
025300  012737  000002  001160                  MOV     #2,$TIMES       ;;DO 2 ITERATIONS
                                        ;CELL INCREMENT OF A FLOATING LOCATION (60000,70000,64000,62000,61000)
                                        ;FILL THE MEMORY BUFFER WITH A KNOWN PATTERN (125252)
                                        ;COLLECT DATA OF 0 AND INCREMENT A TARGET LOCATION
                                        ;VERIFY THAT NO OTHER ADDRESS IS CHANGED
                                        ;AFTER VERIFYING THE WHOLE BUFFER - UPDATE THE OFFSET REGISTER
                                        ;       AND INCREMENT ANOTHER LOCATION
025306  012737  060000  050114          ;       MOV     #BUF0,CURENT    ;PRIME THE CURRENT TARGET LOC.
025314  012737  020000  025662          ;       MOV     #BIT13,100$     ;LOAD FORCE BIT
025322  012737  025330  001110          ;       MOV     #2$,$LPERR      ;LOAD RETURN ADDRESS
                                        ;PRIME THE BUFFER WITH A 125252 PATTERN
025330  012700  060000                  2$:     MOV     #BUF0,R0        ;LOAD POINTER TO BUFFER
025334  012701  125252                          MOV     #125252,R1      ;LOAD VALUE
025340  013702  103352                          MOV     ADNOKT,R2       ;LOAD BUFFER END ADDRESS
025344  010120                          3$:     MOV     R1,(R0)+        ;LOAD BUFFER WITH DATA
025346  020002                          ;       CMP     R0,R2           ;TEST FOR END
025350  001375                          ;       BNE     3$
025352  012777  177770  022534          ;       MOV     #-10,@CURENT    ;PRIME THE TARGET LOCATION

025360  012777  004000  154370          ;       MOV     #CLRALL,@SFR    ;INIT THE DEVICE
025366  013777  050114  154354          ;       MOV     CURENT,@OFF     ;LOAD OFFSET TO TARGET
025374  012777  177777  154350          ;       MOV     #-1,@WCR        ;PRIME THE HIGH 16 BIT Z COUNTER
025402  005077  154346                  ;       CLR     @BAR            ;CLEAR LOW 16 BIT COUNTER
025406  012777  000004  154342          ;       MOV     #TSTCON,@SFR    ;ENABLE TEST CONNECTOR
025414  012777  000024  154324          ;       MOV     #24,@CSR        ;LOAD MATRIX WORD MODE
025422  052777  000002  154326          ;       BIS     #TESTZ,@SFR     ;SET TEST Z FLOP
025430  052777  000001  154310          ;       BIS     #BIT0,@CSR      ;ENABLE THE NCV11

025436  012700  000004                  ;       MOV     #4,R0           ;PRIME THE DELAY
025442  005001                          ;       CLR     R1
025444  032777  060000  154274  4$:     ;       BIT     #BIT14!BIT1_,@CSR       ;TEST FOR CELL OR Z OVERFLOW
025452  001014                          ;       BNE     5$              ;;BR IF EITHER IS SET
025454  005301                          ;       DEC     R1              ;DELAY
025456  001372                          ;       BNE     4$
025460  005300                          ;       DEC     R0              ;DELAY
025462  001370                          ;       BNE     4$
025464  017737  154256  001126          ;       MOV     @CSR,$BDDAT     ;READ BAD STATUS
025472  012737  060224  001124          ;       MOV     #BIT14!BIT13+224,$GDDAT ;LOAD EXPECTED STATUS
025500  104002                          ;       ERROR   2               ;DYNAMIC MATRIX MODE STATUS ERROR
025502  000470                          ;       BR      TST152          ;;
025504  012700  060000          5$:     MOV     #BUF0,R0        ;GET BUFFER POINER
025510  023700  050114          6$:     CMP     CURENT,R0       ;TEST IF TARGET ADDRESS
025514  001415                          BEQ     7$              ;;BR IF YES
025516  010037  001122                  MOV     R0,$BDADR       ;GET BAD ADDRESS FOR TYPE-OUT
025522  011037  001126                  MOV     (R0),$BDDAT     ;GET BAD DATA
025526  012737  125252  001124          MOV     #125252,$GDDAT  ;LOAD EXPECTED DATA
025534  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE
025542  001417                          BEQ     10$             ;;BR IF SAME
025544  104033                          ERROR   33              ;CHANGED AN INCORRECT TARGET LOCATION
025546  000446                          BR      TST152          ;;
```

```
3316  025550  013737  050114  001122  7$:     MOV     CURENT,$BDADR   ;LOAD EXPECTED ADDRESS
3317  025556  017737  022332  001126          MOV     @CUREN~,$BDDAT  ;LOAD ACTUAL DATA
3318  025564  012737  003407  001124          MOV     #3407,$GDDAT    ;LOAD EXPECTED DATA
3319  025572  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE
3320  025600  001400                          BEQ     10$             ;;BR IF SAME
3321
3322  025602  005720                  10$:    TST     (R0)+           ;BUMP THE POINTER INTO THE BUFFER
3323  025604  023700  003352                  CMP     ADNOKT,R0       ;TEST IF FINISHED THE BUFFER
3324  025610  001337                          BNE     6$              ;BR AND RETEST THE REST OF THE BUFFER
3325
3326  025612  005737  001202                  TST     $PASS           ;TEST IF FIRST PASS
3327  025616  001422                          BEQ     TST152          ;;BR IF YES
3328  025620  032777  004000  153312          BIT     #SW11,@SWR      ;TEST INHIBIT INTER.
3329  025626  001016                          BNE     TST152          ;;BR IF SET
3330  025630  006237  025662                  ASR     100$            ;CHANGE THE FORCED ADDR. BIT
3331  025634  022737  000002  025662          CMP     #2,100$         ;TEST IF FINISHED
3332  025642  001410                          BEQ     TST152          ;;BR IF FINISHED
3333  025644  012737  060000  050114          MOV     #BUF0,CURENT    ;MAKE UP NEW ADDRESS
3334  025652  053737  025662  050114          BIS     100$,CURENT     ;
3335  025660  000623                          BR      2$
3336  025662  020000                  100$:   BIT13
3337                                          ;;*****************************************************************
3338                                          ;*TEST 152       DYNAMIC LIST MODE ADDRESS
3339                                          ;;*****************************************************************
3340  025664  000004                  TST152: SCOPE
3341  025666  012737  000002  001160          MOV     #2,$TIMES       ;;DO 2 ITERATIONS
3342                                          ;FILL THE MEMORY BUFFER WITH A KNOWN PATTERN (125252)
3343                                          ;DO A 1 WORD TRANSFER TO A TAGET LOCATION
3344                                          ;VERIFY THAT NO OTHER LOCATION IN THE BUFFER IS CHANGED
3345                                          ;AFTER VERIFYING BUMP THE BUS ADDRESS AND DO SUB-TEST AGAIN
3346  025674  012737  060000  050114          MOV     #BUF0,CURENT    ;PRIME THE CURRENT TARGET LOC.
3347  025702  012737  020000  026242          MOV     #BIT13,100$     ;LOAD FORCE BIT
3348  025710  012737  025716  001110          MOV     #2$,$LPERR      ;LOAD RETURN ADDRESS
3349                                          ;PRIME THE BUFFER WITH A 125252 PATTERN
3350  025716  012700  060000          2$:     MOV     #BUF0,R0        ;LOAD POINTER TO BUFFER
3351  025722  012701  125252                  MOV     #125252,R1      ;LOAD VALUE
3352  025726  013702  003352                  MOV     ADNOKT,R2       ;LOAD BUFFER END ADDRESS
3353  025732  010120                  3$:     MOV     R1,(R0)+        ;LOAD BUFFER WITH DATA
3354  025734  020002                          CMP     R0,R2           ;TEST FOR END
3355  025736  001375                          BNE     3$
3356
3357  025740  012777  004000  154010          MOV     #CLRALL,@SFR    ;INIT THE DEVICE
3358  025746  012777  177777  153776          MOV     #-1,@WCR        ;PRIME THE WORD COUNT
3359  025754  013777  050114  153772          MOV     CURENT,@BAR     ;LOAD TARGET LOCATION POINTER
3360  025762  012777  000004  153766          MOV     #TSTCON,@SFR    ;ENABLE MAINT. MODE
3361  025770  052777  000001  153750          BIS     #BIT0,@CSR      ;ENABLE THE NCV11
3362  025776  052777  000002  153752          BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
3363  026004  042777  000002  153744          BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
3364
3365  026012  012700  000004                  MOV     #4,R0           ;PRIME THE DELAY
3366  026016  005001                          CLR     R1
3367  026020  032777  060000  153720  4$:     BIT     #BIT14!BIT13,@CSR       ;TEST FOR CELL OR Z OVERFLOW
3368  026026  001014                          BNE     5$              ;;BR IF EITHER IS SET
3369  026030  005301                          DEC     R1              ;DELAY
```

```
3370  026032  001372                        BNE     4$
3371  026034  005300                        DEC     R0              ;DELAY
3372  026036  001370                        BNE     4$
3373  026040  017737  153702  001126        MOV     @CSR,$BDDAT     ;READ BAD STATUS
3374  026046  012737  060224  001124        MOV     #BIT14.BIT13+224,$GDDAT ;LOAD EXPECTED STATUS
3375  026054  104036                        ERROR   36              ;DYNAMIC LIST MODE STATUS ERROR
3376  026056  000472                        BR      TST153          ;;
3377  026060  012700  060000          5$:   MOV     #BUF0,R0        ;GET BUFFER POINER
3378  026064  023700  050114          6$:   CMP     CURENT,R0       ;TEST IF TARGET ADDRESS
3379  026070  001415                        BEQ     7$              ;;BR IF YES
3380  026072  010037  001122                MOV     R0,$BDADR       ;GET BAD ADDRESS FOR TYPE-OUT
3381  026076  011037  001126                MOV     (R0),$BDDAT     ;GET BAD DATA
3382  026102  012737  125252  001124        MOV     #125252,$GDDAT  ;LOAD EXPECTED DATA
3383  026110  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;COMPARE
3384  026116  001421                        BEQ     10$             ;;BR IF SAME
3385  026120  104021                        ERROR   21              ;CHANGED AN INCORRECT TARGET LOCATION
3386  026122  000450                        BR      TST153          ;;
3387  026124  013737  050114  001122  7$:   MOV     CURENT,$BDADR   ;LOAD EXPECTED ADDRESS
3388  026132  017737  021756  001126        MOV     @CURENT,$BDDAT  ;GET ACTUAL DATA
3389  026140  012737  003407  001124        MOV     #3407,$GDDAT    ;LOAD EXPECTED DATA
3390  026146  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;COMPARE
3391  026154  001402                        BEQ     10$             ;;BR IF SAME
3392  026156  104037                        ERROR   37              ;LIST MODE DATA ERROR
3393  026160  000431                        BR      TST153          ;;
3394  026162  005720                10$:    TST     (R0)+           ;BUMP THE POINTER INTO THE BUFFER
3395  026164  023700  003352                CMP     ADNOKT,R0       ;TEST IF FINISHED THE BUFFER
3396  026170  001335                        BNE     6$              ;BR AND RETEST THE REST OF THE BUFFER
3397
3398  026172  005737  001202                TST     $PASS           ;TEST IF FIRST PASS
3399  026176  001422                        BEQ     TST153          ;;BR IF FIRST PASS
3400  026200  032777  004000  152732        BIT     #SW11,@SWR      ;TEST INHIBIT INTER.
3401  026206  001016                        BNE     TST153          ;;BR IF SET
3402  026210  006237  026242                ASR     100$            ;CHANGE THE FORCE ADDRESS
3403  026214  022737  000002  026242        CMP     #2,100$         ;TEST IF END
3404  026222  001410                        BEQ     TST153          ;;BR IF FINISHED
3405  026224  012737  060000  050114        MOV     #BUF0,CURENT    ;MAKE NEW ADDRESS
3406  026232  053737  026242  050114        BIS     100$,CURENT     ;
3407  026240  000626                        BR      2$
3408  026242  020000                100$:   BIT13
```

```
                                        ;;*********************************************************************
                                        ;*TEST 153      DYNAMIC LIST MODE TRANSFER - MAXIMUM BUFFER LENGTH IN LOWER 28K
                                        ;;*********************************************************************
026244  000004                          TST153: SCOPE
026246  012737  000040  001160                  MOV     #40,$TIMES      ;;DO 40 ITERATIONS
026254  013700  003352                          MOV     ADNOKT,R0       ;GET LAST ADDRESS
026260  162700  060000                          SUB     #BUF0,R0        ;DETERMINE THE WORD COUNT VALUE
026264  005400                                  NEG     R0
026266  006200                                  ASR     R0
                                        ;PRIME THE BUFFER WITH A 125252 PATTERN
026270  012703  060000                  2$:     MOV     #BUF0,R3        ;LOAD POINTER TO BUFFER
026274  012701  125252                          MOV     #125252,R1      ;LOAD VALUE
026300  013702  003352                          MOV     ADNOKT,R2       ;LOAD BUFFER END ADDRESS
026304  010123                          3$:     MOV     R1,(R3)+        ;LOAD BUFFER WITH DATA
026306  020302                                  CMP     R3,R2           ;TEST FOR END
026310  001375                                  BNE     3$
                                        ;NOW COLLET THE LIST MODE DATA
026312  012777  004000  153436                  MOV     #CLRALL,@SFR    ;INIT THE DEVICE
026320  010077  153426                          MOV     R0,@WCR         ;PRIME THE WORD COUNT
026324  012777  060000  153422                  MOV     #BUF0,@BAR      ;LOAD TARGET LOCATION POINTER
026332  012777  000004  153416                  MOV     #TSTCON,@SFR    ;ENABLE MAINT. MODE
026340  052777  000001  153400                  BIS     #BIT0,@CSR      ;ENABLE THE NCV11
026346  052777  000002  153402                  BIS     #TESTZ,@SFR     ;ENABLE MAINT. Z
026354  012700  000004                          MOV     #4,R0           ;PRIME THE DELAY
026360  032777  060000  153360  4$:     BIT     #BIT14.BIT13,@CSR       ;TEST FOR CELL OR Z OVERFLOW
026366  001014                                  BNE     5$              ;;BR IF EITHER IS SET
026370  005301                                  DEC     R1              ;DELAY
026372  001372                                  BNE     4$
026374  005300                                  DEC     R0              ;DELAY
026376  001370                                  BNE     4$
026400  017737  153342  001126                  MOV     @CSR,$BDDAT     ;READ BAD STATUS
026406  012737  060200  001124                  MOV     #BIT14!BIT13+200,$GDDAT ;LOAD EXPECTED STATUS
026414  104036                                  ERROR   36      ;DYNAMIC LIST MODE STATUS ERROR - NO WORD COUNT OVERFLOW
026416  000423                                  BR      TST154          ;;
026420  012700  060000          5$:     MOV     #BUF0,R0        ;GET BUFFER POINER
026424  010037  001122          6$:     MOV     R0,$BDADR       ;GET BAD ADDRESS FOR TYPE-OUT
026430  011037  001126                  MOV     (R0),$BDDAT     ;GET BAD DATA
026434  012737  003407  001124          MOV     #3407,$GDDAT    ;LOAD EXPECTED DATA
026442  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE
026450  001402                          BEQ     10$             ;;BR IF SAME
026452  104037                          ERROR   37              ;LIST MODE DATA ERROR
026454  000404                          BR      TST154          ;;
026456  005720                  10$:    TST     (R0)+           ;BUMP THE POINTER INTO THE BUFFER
026460  023700  003352                  CMP     ADNOKT,R0       ;TEST IF FINISHED THE BUFFER
026464  001357                          BNE     6$              ;BR AND RETEST THE REST OF THE BUFFER
```

```
                                ;;*****************************************************************
                                ;*TEST 154       ONE MATRIX DATA TRANSFER TO EACH 4K EXTENDED MEMORY
                                ;;*****************************************************************
026466  000004                  TST154: SCOPE
026470  012737  000002  001160          MOV     #2,$TIMES       ;;DO 2 ITERATIONS
026476  005737  036120                  TST     $KT11           ;TEST IF KT-11 INSTALLED
026502  100127                          BPL     TST155          ;;BR IF NONE
                                ;DO A BYTE MATRIX MODE TRANSFER TO A 1 BYTE LOCATION IN EACH 4K EXTENDED
                                ;       MEMORY BANK
026504  012737  000600  041772          MOV     #600,OUT        ;LOAD INITIAL BANK VALUE
026512  013737  041772  172346  1$:     MOV     OUT,@#KIPAR3    ;LOAD KT-PAR REG #3 <BUFFER IS AT LOC 60000>
026520  012700  060000                  MOV     #BUF0,R0        ;LOAD BUFFER POINTER
026524  012710  125252                  MOV     #125252,(R0)    ;PRIME BAD TARGET LOC. IF EXT ADD FAILS
026530  052737  001400  177572          BIS     #1400,@#SR0     ;ENABLE MAINT. MODE KT-11
026536  012710  000370                  MOV     #370,(R0)       ;LOAD TARGET LOCATION DATA VALUE
026542  042737  001400  177572          BIC     #1400,@#SR0     ;DISABLE MAINT. MODE KT-11
026550  013701  041772                  MOV     OUT,R1          ;GET BANK VALUE
026554  005002                          CLR     R2              ;CLEAR EXT. ADD. TEMPORARY
026556  006301                          ASL     R1              ;MOVE LEFT
026560  006301                          ASL     R1              ;MOVE LEFT
026562  006301                          ASL     R1              ;MOVE LEFT
026564  006301                          ASL     R1              ;MOVE LEFT
026566  006301                          ASL     R1
026570  006102                          ROL     R2              ;SAVE EA BITS
026572  006301                          ASL     R1
026574  006102                          ROL     R2              ;SAVE EA BITS
026576  010237  041756                  MOV     R2,NARROW       ;SAVE EA BITS FOR TYPEOUT
026602  010137  001122                  MOV     R1,$BDADR       ;SAVE ADDRESS BITS FOR TYPEOUT
026606  060201                          ADD     R2,R1           ;MAKE COMPLETE ADDRESS

                                ;NOW GET READY TO DO THE TRANSFER
026610  012777  004000  153140          MOV     #CLRALL,@SFR    ;INIT THE NCV11
026616  010177  153126                  MOV     R1,@OFF         ;LOAD COMBINED BUFFER ADDRESS
026622  012777  000022  153116          MOV     #BIT4!BIT1,@CSR ;ENABLE THE NCV11
026630  012777  000014  153120          MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE MAINT NCV11 MODE
026636  052777  000001  153102          BIS     #BIT0,@CSR      ;ENABLE THE NCV11
026644  052777  000002  153104          BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z' PULSES
026652  042777  000002  153076          BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z' PULSES
026660  012737  000371  001124          MOV     #371,$GDDAT     ;LOAD EXPECTED DATA
026666  052777  010000  153062          BIS     #BIT12,@SFR     ;ENABLE 1 BYTE TRANSFER
026674  052737  000001  177572          BIS     #BIT0,@#SR0     ;ENABLE KT-11
026702  011037  001126                  MOV     (R0),$BDDAT     ;GET ACTUAL DATA
026706  042737  000001  177572          BIC     #BIT0,@#SR0     ;DISABLE KT-11
026714  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE DATA
026722  001401                          BEQ     2$              ;;BR IF SAME
026724  104035                          ERROR   35              ;DATA TRANSFER ERROR TO EXTENDED MEMORY
026726  005737  001202          2$:     TST     $PASS           ;TEST PASS COUNTER
026732  001413                          BEQ     TST155          ;;BR IF FIRST PASS
026734  032777  004000  152176          BIT     #SW11,@SWR      ;TEST IF INHIBIT INTER.
026742  001007                          BNE     TST155          ;;BR IF SET
026744  062737  000200  041772          ADD     #200,OUT        ;UPDATE BANK VALUE
026752  023737  036402  041772          CMP     $LSTBK,OUT      ;TEST IF DONE
026760  101254                          BHI     1$              ;BR IF NOT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 155      ONE LIST DATA TRANSFER TO EACH 4K EXTENDED MEMORY
                                        ;;*****************************************************************
026762  000004                          TST155: SCOPE
026764  012737  000002  001160                  MOV     #2,$TIMES       ;;DO 2 ITERATIONS
026772  005737  036120                          TST     $KT11           ;TEST IF KT-11 INSTALLED
026776  100130                          ;       BPL     TST156          ;;BR IF NONE
                                        ;DO A 1 WORD MODE TRANSFER TO A 1 WORD BUFFER IN EACH 4K EXTENDED MEMORY BANK
027000  012737  000600  041772                  MOV     #600,OUT        ;LOAD INITIAL BANK VALUE <60000>
027006  013737  041772  172346  1$:             MOV     OUT,@#KIPAR3    ;LOAD KT-PAR REG #3 <BUFFER IS AT LOC 60000>
027014  012700  060000                          MOV     #BUF0,R0        ;LOAD BUFFER POINTER
027020  012710  052525                          MOV     #52525,(R0)     ;LOAD BAD TARGET LOC. IF EXT ADD FAILS
027024  052737  001400  177572                  BIS     #1400,@#SR0     ;ENABLE MAINT. MODE KT-11
027032  012710  125252                          MOV     #125252,(R0)    ;PRESET THE TARGET LOCATION
027036  042737  001400  177572                  BIC     #1400,@#SR0     ;DISABLE MAINT. MODE KT-11
027044  013701  041772                          MOV     OUT,R1          ;GET BANK VALUE
027050  005002                                  CLR     R2              ;CLEAR EXT. ADD TEMPORARY
027052  006301                                  ASL     R1
027054  006301                                  ASL     R1
027056  006301                                  ASL     R1
027060  006301                                  ASL     R1
027062  006301                                  ASL     R1
027064  006102                                  ROL     R2              ;SAVE EXT. ADDRESS BITS
027066  006301                                  ASL     R1
027070  006102                                  ROL     R2              ;SAVE EXT. ADDRESS BITS
                                        ;NOW GET READY TO DO THE TRANSFER
027072  012777  004000  152656                  MOV     #CLRALL,@SFR    ;INIT THE NCV11
027100  010277  152644                          MOV     R2,@OFF         ;LOAD THE EXTENDED ADDRESS BITS
027104  012777  177777  152640                  MOV     #-1,@WCR        ;LOAD WORD COUNT
027112  010177  152636                          MOV     R1,@BAR         ;LOAD BUS ADDRESS
027116  012777  000014  152632                  MOV     #TS[illegible]MA!TSTCON,@SFR    ;ENABLE MAINT NCV11 MODE
027124  052777  000001  152614                  BIS     #BI[illegible],@CSR     ;ENABLE THE NCV11
027132  052777  000002  152616                  BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
027140  042777  000002  152610                  BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
027146  010137  001122                          MOV     R1,$BDADR       ;SAVE TARGET ADDRESS
027152  010237  041756                          MOV     R2,NARROW       ;SAVE EA BITS FOR ERROR TYPEOUT
027156  052777  010000  152572                  BIS     #BIT12,@SFR     ;ENABLE 1 BYTE TRANSFER
027164  012737  003407  001124                  MOV     #3407,$GDDAT    ;LOAD EXPECTED DATA
027172  052737  000001  177572                  BIS     #BIT0,@#SR0     ;ENABLE KT-11
027200  011037  001126                          MOV     (R0),$BDDAT     ;GET ACTUAL DATA
027204  042737  000001  177572                  BIC     #BIT0,@#SR0     ;DISABLE KT-11
027212  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;COMPARE DATA
027220  001401                                  BEQ     2$              ;;BR IF SAME
027222  104035                                  ERROR   35              ;LIST MODE DATA TRANSFER ERROR TO EXTENDED MEMORY
027224  005737  001202          2$:             TST     $PASS           ;TEST PASS COUNTER
027230  001413                                  BEQ     TST156          ;;BR IF FIRST PASS
027232  032777  004000  151700                  BIT     #SW11,@SWR      ;TEST INHIBIT INTER.
027240  001007                                  BNE     TST156          ;;BR IF SET
027242  062737  000200  041772                  ADD     #200,OUT        ;UPDATE BANK VALUE
027250  023737  036402  041772                  CMP     $LSTBK,OUT      ;TEST IF DONE
027256  101253                                  BHI     1$              ;BR IF NOT
```

```
                        ;;*****************************************************************
                        ;*TEST 156      VERIFY BIT 15 MATRIX ADDER INPUT
                        ;;*****************************************************************
027260  000004          TST156: SCOPE
027262  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                        ;ADM OUTPUT     =       127657
                        ;OFFSET         =       140000
                        ;TARGET         =       267657
027270  023727  036402  003140          CMP     $LSTBK,#3140    ;TEST IF >52K IS AVAILABLE
027276  103476                          BLO     TST157          ;;BR IF NO MORE MEMORY
027300  012737  002676  172346          MOV     #2676,@#KIPAR3  ;LOAD KT-PAR REG #3 <BUFFER IS AT LOC 60000>
027306  012700  060056                  MOV     #BUF0+56,R0     ;LOAD BUFFER POINTER
027312  012710  125252                  MOV     #125252,(R0)    ;LOAD BAD TARGET LOC. IF EXT ADD FAILS
027316  052737  001400  177572          BIS     #1400,@#SR0     ;ENABLE MAINT. MODE KT-11
027324  005010                          CLR     (R0)            ;LOAD TARGET LOCATION DATA VALUE
027326  042737  001400  177572          BIC     #1400,@#SR0     ;DISABLE MAINT. MODE KT-11

                        ;NOW GET READY TO DO THE TRANSFER
027334  012777  004000  152414          MOV     #CLRALL,@SFR    ;INIT THE NCV11
027342  012777  140000  152400          MOV     #140000,@OFF    ;LOAD COMBINED BUFFER ADDRESS
027350  012777  002036  152370          MOV     #2036,@CSR      ;ENABLE THE NCV11
027356  012777  000014  152372          MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE MAINT NCV11 MODE
027364  052777  000001  152354          BIS     #BIT0,@CSR      ;ENABLE THE NCV11
027372  052777  000002  152356          BIS     #TESTZ,@SFR             ;ENABLE ''TEST Z'' PULSES
027400  042777  000002  152350          BIC     #TESTZ,@SFR             ;DISABLE ''TEST Z'' PULSES
027406  000240                          NOP
027410  000240                          NOP
027412  052777  010000  152336          BIS     #BIT12,@SFR     ;ENABLE 1 BYTE TRANSFER
027420  000240                          NOP
027422  000240                          NOP
027424  000240                          NOP
027426  012737  000001  041756          MOV     #1,NARROW       ;SAVE EA BITS FOR ERROR TYPEOUT
027434  012737  000400  001124          MOV     #400,$GDDAT     ;LOAD EXPECTED DATA
027442  052737  000001  177572          BIS     #BIT0,@#SR0     ;ENABLE KT-11
027450  011037  001126                  MOV     (R0),$BDDAT     ;SAVE ACTUAL DATA
027454  042737  000001  177572          BIC     #BIT0,@#SR0     ;DISABLE KT-11
027462  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE DATA
027470  001401                          BEQ     TST157          ;;BR IF SAME
027472  104020                          ERROR   20              ;BIT 15 INPUT TO MATRIX MODE ADDER FAILED
```

```
3598                                    ;;*****************************************************************
3599                                    ;*TEST 157      VERIFY HIGH BYTE OPERATION OF THE 'TEST X'
3600                                    ;;*****************************************************************
3601  027474  000004                    TST157: SCOPE
3602  027476  012737  000040  001160            MOV     #40,$TIMES      ;;DO 40 ITERATIONS
3603                                    ;TEST X         =       1
3604                                    ;OFFSET         =100000
3605                                    ;ADM OUTPUT     =127657
3606                                    ;TARGET         =100257
3607  027504  012777  004000  152244            MOV     #CLRALL,@SFR    ;CLEAR THE DEVICE
3608  027512  023727  036402  001140            CMP     $LSTBK,#1140    ;TEST IF LEAST 20K
3609  027520  103445                            BLO     TST160          ;;BR IF NOT
3610  027522  012777  100000  152220            MOV     #100000,@OFF    ;SET BIT 15 OF OFFSET
3611  027530  012777  000034  152220            MOV     #TSTDMA!TSTCON!BIT4,@SFR        ;SET TEST DMA, CONTROL AND TESTX
3612  027536  012777  002036  152202            MOV     #2036,@CSR      ;ENABLE THE NCV11
3613  027544  052777  000001  152174            BIS     #BIT0,@CSR      ;ENABLE THE NCV11
3614  027552  052777  000002  152176            BIS     #TESTZ,@SFR             ;ENABLE 'TEST Z'' PULSES
3615  027560  042777  000002  152170            BIC     #TESTZ,@SFR             ;DISABLE 'TEST Z'' PULSES
3616  027566  005037  100256                    CLR     @#BUF0+20256    ;CLEAR THE TARGET LOC.
3617  027572  052777  010000  152156            BIS     #BIT12,@SFR     ;ALLOW 1 DMA TRANSFER
3618  027600  012737  100256  001122            MOV     #BUF0+20256,$BDADR      ;LOAD EXPECTED ADDRESSES VALUE
3619  027606  012737  000400  001124            MOV     #400,$GDDAT     ;LOAD EXPECTED VALUE READ
3620  027614  017737  151302  001126            MOV     @$BDADR,$BDDAT  ;READ ACTUAL DATA
3621  027622  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;COMPARE
3622  027630  001401                            BEQ     TST160          ;;BR IF SAME
3623  027632  104034                            ERROR   34              ;MATRIX ADDER INPUT OF ADM15 AND TESTX L
3624                                                            ;FAILED TO BE INHIBITED
```

```
                                        ;;*****************************************************************
                                        ;*TEST 160      VERIFY BIT 16 INPUT TO THE MATRIX MODE ADDER
                                        ;;*****************************************************************
027634  000004                  TST160: SCOPE
027636  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                            ;ADM OUTPUT     =       253527
                            ;OFFSET         =       210000
                            ;TARGET         =       463527
027644  023727  036402  005140          CMP     $LSTBK,#5140    ;TEST IF ENOUGH MEMORY >84K
027652  103500                          BLO     TST161          ;;BR IF NOT
027654  012737  004635  172346          MOV     #4635,@#KIPAR3  ;LOAD KT-PAR REG #3 <BUFFER IS AT LOC 60000>
027662  012700  060026                  MOV     #BUF0+26,R0     ;LOAD BUFFER POINTER
027666  012710  125252                  MOV     #125252,(R0)    ;LOAD BAD TARGET LOC.
027672  052737  001400  177572          BIS     #1400,@#SR0     ;ENABLE MAINT. MODE KT-11
027700  005010                          CLR     (R0)            ;PRESET THE TARGET LOCATION
027702  042737  001400  177572          BIC     #1400,@#SR0     ;DISABLE MAINT. MODE KT-11

                                ;NOW GET READY TO DO THE TRANSFER
027710  012777  004000  152040          MOV     #CLRALL,@SFR    ;INIT THE NCV11
027716  012777  010001  152024          MOV     #10001,@OFF     ;LOAD THE EXTENDED ADDRESS BITS
027724  012777  000014  152024          MOV     #TSTDMA!TSTCON,@SFR     ;ENABLE MAINT NCV11 MODE
027732  012777  004036  152006          MOV     #4036,@CSR      ;LOAD MATRIX MODE AND ZB ENABLE
027740  052777  000001  152000          BIS     #BIT0,@CSR      ;ENABLE THE NCV11
027746  052777  000002  152002          BIS     #TESTZ,@SFR             ;ENABLE "TEST Z" PULSES
027754  042777  000002  151774          BIC     #TESTZ,@SFR             ;DISABLE "TEST Z" PULSES
027762  000240                          NOP
027764  000240                          NOP
027766  052777  010000  151762          BIS     #BIT12,@SFR     ;ENABLE 1 BYTE TRANSFER
027774  000240                          NOP
027776  000240                          NOP
030000  000240                          NOP
030002  012737  000002  041756          MOV     #2,NARROW       ;SAVE EA BITS FOR ERROR TYPEOUT
030010  012737  000400  001124          MOV     #400,$GDDAT     ;LOAD EXPECTED DATA
030016  052737  000001  177572          BIS     #BIT0,@#SR0     ;ENABLE KT-11
030024  111037  001126                  MOVB    (R0),$BDDAT     ;SAVE ACTUAL DATA
030030  042737  000001  177572          BIC     #BIT0,@#SR0     ;DISABLE KT-11
030036  105037  001126                  CLRB    $BDDAT          ;MASK OFF LOW BYTE
030042  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE DATA
030050  001401                          BEQ     TST161          ;;BR IF SAME
030052  104020                          ERROR   20              ;BIT 16 INPUT TO MATRIX MODE ADDER FAILED
```

```
                                        ;;*********************************************************************
                                        ;*TEST 161      DETERMINE IF DIFLIN IS TO BE RUN (F)
                                        ;;*********************************************************************
030054  000004                  TST161: SCOPE
030056  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
030064  005737  050106                  TST     RUNDIF          ;TEST IF DIFLIN IS TO BE RUN
030070  001402                          BEQ     $EOP            ;;BR IF NOT TO BE RUN
030072  000137  043712                  JMP     DIFLIN          ;JUMP AND RUN DIFLIN
                                        .SBTTL  END OF PASS ROUTINE

                                        ;;*********************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*TYPE 'END PASS #XXXXX'' (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO LOGIC

030076                          $EOP:
030076  000004                          SCOPE
030100  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
030104  005037  001160                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
030110  005237  001202                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
030114  042737  100000  001202          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
030122  005327                          DEC     (PC)+           ;;LOOP?
030124  000001                  $EOPCT: .WORD   1
030126  003022                          BGT     $DOAGN          ;;YES
030130  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
030132  000001                  $ENDCT: .WORD   1
030134  030124                          $EOPCT
030136  104401  030203                  TYPE    ,$ENDMG         ;;TYPE 'END PASS #''
030142  013746  001202                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
030146  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
030150  104401  030200                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
030154  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
030160  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
030162  000005                          RESET                   ;;CLEAR THE WORLD
030164  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
030166  000240                          NOP                     ;;SAVE ROOM
030170  000240                          NOP                     ;;FOR
030172  000240                          NOP                     ;;ACT11
030174                          $DOAGN:
030174  000137                          JMP     @(PC)+          ;;RETURN
030176  003356                  $RTNAD: .WORD   LOGIC
030200     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
030203     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
030210  050040  051501  020123
030216  000043
```

```
                                                .SBTTL  ERROR ASCII MESSAGES

030220  034115  031060  004466  EM1:    .ASCIZ  /M8026  NCV11 BUS ADDRESS TIMEOUT/
030226  041516  030526  020061
030234  052502  020123  042101
030242  051104  051505  020123
030250  044524  042515  052517
030256  000124
030260  034115  031060  004466  EM2:    .ASCIZ  /M8026  COMMAND-STATUS REGISTER ERROR/
030266  047503  046515  047101
030274  026504  052123  052101
030302  051525  051040  043505
030310  051511  042524  020122
030316  051105  047522  000122
030324  034115  031060  004466  EM3:    .ASCIZ  /M8026  SPECIAL FUNCTION REGISTER ERROR/
030332  050123  041505  040511
030340  020114  052506  041516
030346  044524  047117  051040
030354  043505  051511  042524
030362  020122  051105  047522
030370  000122
030372  034115  031060  004466  EM4:    .ASCIZ  /M8026  WORD COUNT REGISTER ERROR/
030400  047527  042122  041440
030406  052517  052116  051040
030414  043505  051511  042524
030422  020122  051105  047522
030430  000122
030432  034115  031060  004466  EM5:    .ASCIZ  /M8026  BUS ADDRESS REGISTER ERROR/
030440  052502  020123  042101
030446  051104  051505  020123
030454  042522  044507  052123
030462  051105  042440  051122
030470  051117     000
030473     115  030070  033062  EM6:    .ASCIZ  /M8026  OFFSET REGISTER ERROR/
030500  047411  043106  042523
030506  020124  042522  044507
030514  052123  051105  042440
030522  051122  051117     000
030527     115  030070  033062  EM7:    .ASCIZ  /M8026  DUAL REGISTER SELECTION ERROR/
030534  042011  040525  020114
030542  042522  044507  052123
030550  051105  051440  046105
030556  041505  044524  047117
030564  042440  051122  051117
030572     000
030573     115  030070  033062  EM10:   .ASCIZ  /M8026-M8036    LOW 16 BIT Z COUNT ERROR/
030600  046455  030070  033063
030606  046011  053517  030440
030614  020066  044502  020124
030622  020132  047503  047125
030630  020124  051105  047522
030636  000122
030640  034115  031060  026466  EM11:   .ASCIZ  /M8026-M8036    HIGH 16 BIT Z COUNT ERROR/
030646  034115  031460  004466
```

```
 3766  030654  044510  044107  030440
 3767  030662  020066  044502  020124
 3768  030670  020132  047503  047125
 3769  030676  020124  051105  047522
 3770  030704  000122
 3771  030706  034115  031060  004466  EM12:   .ASCIZ  /M8026  Z COUNT STATUS ERROR/
 3772  030714  020132  047503  047125
 3773  030722  020124  052123  052101
 3774  030730  051525  042440  051122
 3775  030736  051117     000
 3776  030741     115  030070  033062  EM13:   .ASCIZ  /M8026  Z COUNT INTERRUPT ERROR/
 3777  030746  055011  041440  052517
 3778  030754  052116  044440  052116
 3779  030762  051105  052522  052120
 3780  030770  042440  051122  051117
 3781  030776     000
 3782  030777     115  030070  033063  EM14:   .ASCIZ  /M8036  JOYSTICK STATUS ERROR/
 3783  031004  045011  054517  052123
 3784  031012  041511  020113  052123
 3785  031020  052101  051525  042440
 3786  031026  051122  051117     000
 3787  031033     115  030070  033063  EM15:   .ASCIZ  /M8036  JOYSTICK DATA ERROR/
 3788  031040  045011  054517  052123
 3789  031046  041511  020113  040504
 3790  031054  040524  042440  051122
 3791  031062  051117     000
 3792  031065     115  030070  033063  EM16:   .ASCIZ  /M8036  DATA INCREMENT ERROR/
 3793  031072  042011  052101  020101
 3794  031100  047111  051103  046505
 3795  031106  047105  020124  051105
 3796  031114  047522  000122
 3797  031120  034115  031460  004466  EM17:   .ASCIZ  /M8036  DATA DECREMENT ERROR/
 3798  031126  040504  040524  042040
 3799  031134  041505  042522  042515
 3800  031142  052116  042440  051122
 3801  031150  051117     000
 3802  031153     115  030070  033062  EM20:   .ASCIZ  /M8026-M8036    MATRIX MODE ADDRESS MAKER DATA ERROR/
 3803  031160  046455  030070  033063
 3804  031166  046411  052101  044522
 3805  031174  020130  047515  042504
 3806  031202  040440  042104  042522
 3807  031210  051523  046440  045501
 3808  031216  051105  042040  052101
 3809  031224  020101  051105  047522
 3810  031232  000122
 3811  031234  034115  031060  004466  EM21:   .ASCIZ  /M8026  LIST MODE ADDRESS MAKER DATA ERROR/
 3812  031242  044514  052123  046440
 3813  031250  042117  020105  042101
 3814  031256  051104  051505  020123
 3815  031264  040515  042513  020122
 3816  031272  040504  040524  042440
 3817  031300  051122  051117     000
 3818  031305     115  030070  033062  EM22:   .ASCIZ  /M8026  LIST MODE TRANSFER BUS ADDRESS ERROR/
 3819  031312  046011  051511  020124
```

```
 3820  031320  047515  042504  052040
 3821  031326  040522  051516  042506
 3822  031334  020122  052502  020123
 3823  031342  042101  051104  051505
 3824  031350  020123  051105  047522
 3825  031356  000122
 3826  031360  034115  031060  004466  EM23:   .ASCIZ  /M8026  LIST MODE TRANSFER WORD COUNT ERROR/
 3827  031366  044514  052123  046440
 3828  031374  042117  020105  051124
 3829  031402  047101  043123  051105
 3830  031410  053440  051117  020104
 3831  031416  047503  047125  020124
 3832  031424  051105  047522  000122
 3833  031432  034115  031060  004466  EM24:   .ASCIZ  /M8026  LIST MODE TRANSFER OFFSET ERROR/
 3834  031440  044514  052123  046440
 3835  031446  042117  020105  051124
 3836  031454  047101  043123  051105
 3837  031462  047440  043106  042523
 3838  031470  020124  051105  047522
 3839  031476  000122
 3840  031500  034115  031060  004466  EM25:   .ASCIZ  /M8026  TIMEOUT STATUS ERROR/
 3841  031506  044524  042515  052517
 3842  031514  020124  052123  052101
 3843  031522  051525  042440  051122
 3844  031530  051117     000
 3845  031533     115  030070  033062  EM26:   .ASCIZ  /M8026  TIMEOUT INTERRUPT ERROR/
 3846  031540  052011  046511  047505
 3847  031546  052125  044440  052116
 3848  031554  051105  052522  052120
 3849  031562  042440  051122  051117
 3850  031570     000
 3851  031571     115  030070  033062  EM27:   .ASCIZ  /M8026  SET 'EVENT' OR 'TIME' DATA ERROR/
 3852  031576  051411  052105  021040
 3853  031604  053105  047105  021124
 3854  031612  047440  020122  052042
 3855  031620  046511  021105  042040
 3856  031626  052101  020101  051105
 3857  031634  047522  000122
 3858  031640  034115  031060  004466  EM30:   .ASCIZ  /M8026  CELL INCREMENT DATA ERROR/
 3859  031646  042503  046114  044440
 3860  031654  041516  042522  042515
 3861  031662  052116  042040  052101
 3862  031670  020101  051105  047522
 3863  031676  000122
 3864  031700  034115  031060  004466  EM31:   .ASCIZ  /M8026  CELL OVERFLOW STATUS ERROR/
 3865  031706  042503  046114  047440
 3866  031714  042526  043122  047514
 3867  031722  020127  052123  052101
 3868  031730  051525  042440  051122
 3869  031736  051117     000
 3870  031741     115  030070  033062  EM32:   .ASCIZ  /M8026  CELL OVERFLOW INTERRUPT ERROR/
 3871  031746  041411  046105  020114
 3872  031754  053117  051105  046106
 3873  031762  053517  044440  052116
```

```
3874  031770  051105  052522  052120
3875  031776  042440  051122  051117
3876  032004     000
3877  032005     115  030070  033062  EM33:   .ASCIZ  /M8026  MATRIX MODE ADDRESS MUX ERROR/
3878  032012  046411  052101  044522
3879  032020  020130  047515  042504
3880  032026  040440  042104  042522
3881  032034  051523  046440  054125
3882  032042  042440  051122  051117
3883  032050     000
3884  032051     115  030070  033062  EM34:   .ASCIZ  /M8026  ''TESTX'' FUNCTION ERROR/
3885  032056  021011  042524  052123
3886  032064  021130  043040  047125
3887  032072  052103  047511  020116
3888  032100  051105  047522  000122
3889  032106  034115  031060  004466  EM35:   .ASCIZ  /M8026  DATA ERROR WHEN TRANSFERING TO EXTENDED MEMORY/
3890  032114  040504  040524  042440
3891  032122  051122  051117  053440
3892  032130  042510  020116  051124
3893  032136  047101  043123  051105
3894  032144  047111  020107  047524
3895  032152  042440  052130  047105
3896  032160  042504  020104  042515
3897  032166  047515  054522     000
3898  032173     115  030070  033062  EM36:   .ASCIZ  /M8026  LIST MODE TRANSFER STATUS ERROR/
3899  032200  046011  051511  020124
3900  032206  047515  042504  052040
3901  032214  040522  051516  042506
3902  032222  020122  052123  052101
3903  032230  051525  042440  051122
3904  032236  051117     000
3905  032241     115  030070  033062  EM37:   .ASCIZ  /M8026  LIST MODE TRANSFER DATA ERROR/
3906  032246  046011  051511  020124
3907  032254  047515  042504  052040
3908  032262  040522  051516  042506
3909  032270  020122  040504  040524
3910  032276  042440  051122  051117
3911  032304     000
3912  032305     112  046525  042520  EM40:   .ASCIZ  /JUMPER-M8026-M7952     'EVENT'' OR ''TIME'' MARK ERROR/
3913  032312  026522  034115  031060
3914  032320  026466  033515  032471
3915  032326  004462  042442  042526
3916  032334  052116  020042  051117
3917  032342  021040  044524  042515
3918  032350  020042  040515  045522
3919  032356  042440  051122  051117
3920  032364     000
3921  032365     115  034467  031065  EM41:   .ASCIZ  /M7952  CLOCK BUS ADDRESS TIMEOUT/
3922  032372  041411  047514  045503
3923  032400  041040  051525  040440
3924  032406  042104  042522  051523
3925  032414  052040  046511  047505
3926  032422  052125     000
3927  032425     115  031070  033461  EM42:   .ASCIZ  /M8217  INCORRECT INTERRUPT LEVEL/
```

```
 3928  032432  044411  041516  051117
 3929  032440  042522  052103  047440
 3930  032446  052116  051105  052040
 3931  032454  052120  046040  052040
 3932  032462  046105     000
 3933
 3934  032465     105  051122  041520  DH1:    .ASCIZ  /ERRPC  ADDR/
 3935  032472  040411  042104  000122
 3936  032500  051105  050122  004503  DH2:    .ASCIZ  /ERRPC  ADDR    GOOD    BAD/
 3937  032506  042101  051104  043411
 3938  032514  047517  004504  040502
 3939  032522  000104
 3940  032524  051105  050122  004503  DH3:    .ASCIZ  /ERRPC  ADDR    GOOD    BAD     BADADR/
 3941  032532  042101  051104  043411
 3942  032540  047517  004504  040502
 3943  032546  004504  040502  040504
 3944  032554  051104     000
 3945  032557     105  051122  041520  DH4:    .ASCIZ  /ERRPC  ADDR    GOOD    BAD     EA ADR  BADADR/
 3946  032564  040411  042104  004522
 3947  032572  047507  042117  041011
 3948  032600  042101  042411  020101
 3949  032606  042101  004522  040502
 3950  032614  040504  051104     000
 3951  032621     015     012     007  OUTRNG: .BYTE   15,12,7
 3952  032624  047516  021040  021132          .ASCII  /NO "Z" PULSES OR /
 3953  032632  050040  046125  042523
 3954  032640  020123  051117     040
 3955  032645     111  050116  052125          .ASCII  /INPUT VOLTAGE OUT OF RANGE ON CHANNEL #/
 3956  032652  053040  046117  040524
 3957  032660  042507  047440  052125
 3958  032666  047440  020106  040522
 3959  032674  043516  020105  047117
 3960  032702  041440  040510  047116
 3961  032710  046105  021440
 3962  032714     060     015     012  OUTCHN: .BYTE   60,15,12,7,0
 3963  032717     007     000
 3964          032722                          .EVEN
 3965  032722  001116  001250  000000  DT1:    .WORD   $ERRPC,$BASE,0
 3966  032730  001116  001250  001124  DT2:    .WORD   $ERRPC,$BASE,$GDDAT,$BDDAT,0
 3967  032736  001126  000000
 3968  032742  001116  001250  001124  DT3:    .WORD   $ERRPC,$BASE,$GDDAT,$BDDAT,$BDADR,0
 3969  032750  001126  001122  000000
 3970  032756  001116  001250  001124  DT4:    .WORD   $ERRPC,$BASE,$GDDAT,$BDDAT,NARROW,$BDADR,0
 3971  032764  001126  041756  001122
 3972  032772  000000
 3973  032774  001116  001254  000000  DT5:    .WORD   $ERRPC,$CDW1,0
 3974  033002  000000                  DF0:    .WORD   0
```

```
                                        .SBTTL  TTY INPUT ROUTINE
                                        ;;*****************************************************************
                                        .ENABL  LSB
                                        ;;*****************************************************************
                                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                        ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                        ;*WHEN OPERATING IN TTY FLAG MODE.
033004  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
033012  001074                        BNE     15$             ;;BRANCH IF NO
033014  105777  146124                TSTB    @$TKS           ;;CHAR THERE?
033020  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
033022  117746  146120                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
033026  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
033032  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
033036  001062                        BNE     15$             ;;NO, RETURN TO USER
033040  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
033046  001456                        BEQ     15$             ;;BRANCH IF YES
                                
033050  104401  033531                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
033054  104401  033536        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
033060  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
033064  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
033066  104401  033547                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
033072  005046                19$:    CLR     -(SP)           ;;CLEAR COUNTER
033074  005046                        CLR     -(SP)           ;;THE NEW SWR
033076  105777  146042        7$:     TSTB    @$TKS           ;;CHAR THERE?
033102  100375                        BPL     7$              ;;IF NOT TRY AGAIN
                                
033104  117746  146036                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
033110  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
                                
                                
                                
033114  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
033120  001005                        BNE     10$             ;;BRANCH IF NOT
033122  104401  033524                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
033126  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
033132  000757                        BR      19$             ;;LET'S TRY IT AGAIN
                                
                                
033134  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
033140  001022                        BNE     16$             ;;BRANCH IF NO
033142  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
033146  001403                        BEQ     11$             ;;BRANCH IF YES
033150  016677  000002  145762        MOV     2(SP),@SWR      ;;SAVE NEW SWR
033156  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
033162  104401  001171        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
033166  123727  001135  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
033174  001003                        BNE     15$             ;;BRANCH IF NOT
033176  012777  000100  145740        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
033204  000002                15$:    RTI                     ;;RETURN
```

```
 4029  033206  004737  034604          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
 4030  033212  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
 4031  033216  002420                          BLT     18$             ;;BRANCH IF YES
 4032  033220  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
 4033  033224  003015                          BGT     18$             ;;BRANCH IF YES
 4034  033226  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 4035  033232  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
 4036  033236  001403                          BEQ     17$             ;;BRANCH IF YES
 4037  033240  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
 4038  033242  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
 4039  033244  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
 4040  033246  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 4041  033252  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 4042  033256  000707                          BR      7$              ;;GET THE NEXT ONE
 4043  033260  104401  001170          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 4044  033264  000720                          BR      20$             ;;SIMULATE CONTROL-U
 4045                                  .DSABL  LSB
 4046
 4047
 4048                                  ;;**************************************************************************
 4049                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 4050                                  ;*CALL:
 4051                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
 4052                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 4053                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
 4054                                  ;
 4055
 4056  033266  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
 4057  033270  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
 4058  033276  105777  145642          1$:     TSTB    @$TKS           ;;WAIT FOR
 4059  033302  100375                          BPL     1$              ;;A CHARACTER
 4060  033304  117766  145636  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
 4061  033312  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 4062  033320  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 4063  033326  001013                          BNE     3$              ;;BRANCH IF NO
 4064  033330  105777  145610          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
 4065  033334  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
 4066  033336  117746  145604                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
 4067  033342  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4068  033346  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 4069  033352  001366                          BNE     2$              ;;IF NOT DISCARD IT
 4070  033354  000750                          BR      1$              ;;YES, RESUME
 4071  033356  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
 4072  033364  002407                          BLT     4$              ;;BRANCH IF YES
 4073  033366  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 4074  033374  003003                          BGT     4$              ;;BRANCH IF YES
 4075  033376  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 4076  033404  000002                  4$:     RTI                     ;;GO BACK TO USER
 4077                                  ;;**************************************************************************
 4078                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 4079                                  ;*CALL:
 4080                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 4081                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 4082                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
```

```
 4083
 4084  033406  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 4085  033410  012703  033514            1$:   MOV     #$TTYIN,R3      ;;GET ADDRESS
 4086  033414  022703  033524            2$:   CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
 4087  033420  101405                          BLOS    4$              ;;BR IF YES
 4088  033422  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 4089  033424  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
 4090  033426  122713  000177           10$:   CMPB    #177,(R3)       ;;IS IT A RUBOUT
 4091  033432  001003                          BNE     3$              ;;SKIP IF NOT
 4092  033434  104401  001170            4$:   TYPE    ,$QUES          ;;TYPE A '?'
 4093  033440  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
 4094  033442  111337  033512            3$:   MOVB    (R3),9$         ;;ECHO THE CHARACTER
 4095  033446  104401  033512                  TYPE    ,9$
 4096  033452  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 4097  033456  001356                          BNE     2$              ;;LOOP IF NOT RETURN
 4098  033460  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 4099  033464  104401  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
 4100  033470  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 4101  033472  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 4102  033474  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 4103  033502  012766  033514  000004          MOV     #$TTYIN,4(SP)
 4104  033510  000002                          RTI                     ;;RETURN
 4105  033512     000                    9$:   .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
 4106  033513     000                          .BYTE   0               ;;TERMINATOR
 4107  033514  000010                 $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
 4108  033524  052536  005015     000 $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
 4109  033531     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
 4110  033536  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
 4111  033544  020075     000
 4112  033547     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
 4113  033554  036440  000040
```

```
                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

033560  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
033562  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
033570  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
033572  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
033574  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
033576  104411                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
033600  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
033602  005001                          CLR     R1              ;;CLEAR DATA WORD
033604  005002                          CLR     R2
033606  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
033610  001412                          BEQ     3$              ;;IF ZERO GET OUT
033612  006301                          ASL     R1              ;;*2
033614  006102                          ROL     R2
033616  006301                          ASL     R1              ;;*4
033620  006102                          ROL     R2
033622  006301                          ASL     R1              ;;*8
033624  006102                          ROL     R2
033626  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
033632  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
033634  000764                          BR      2$              ;;LOOP
033636  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
033640  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
033644  010237  033660                  MOV     R2,$HIOCT
033650  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
033652  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
033654  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
033656  000002                          RTI                     ;;RETURN
033660  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
```

```
                                        .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                        ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                        ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                        ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                        ;*REPLACED WITH SPACES.
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                        ;*      TYPDS                   ;;GO TO THE ROUTINE

033662                                  $TYPDS:
033662  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
033664  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
033666  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
033670  010346                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
033672  010546                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
033674  012746  020200                        MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
033700  016605  000020                        MOV     20(SP),R5       ;;GET THE INPUT NUMBER
033704  100004                               BPL     1$              ;;BR IF INPUT IS POS.
033706  005405                               NEG     R5              ;;MAKE THE BINARY NUMBER POS.
033710  112766  000055  000001               MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
033716  005000                        1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
033720  012703  034076                       MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
033724  112723  000040                       MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
033730  005002                        2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
033732  016001  034066                       MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
033736  160105                        3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
033740  002402                               BLT     4$              ;;BR IF DONE
033742  005202                               INC     R2              ;;INCREASE THE BCD DIGIT BY 1
033744  000774                               BR      3$
033746  060105                        4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
033750  005702                               TST     R2              ;;CHECK IF BCD DIGIT=0
033752  001002                               BNE     5$              ;;FALL THROUGH IF 0
033754  105716                               TSTB    (SP)            ;;STILL DOING LEADING 0'S?
033756  100407                               BMI     7$              ;;BR IF YES
033760  106316                        5$:    ASLB    (SP)            ;;MSD?
033762  103003                               BCC     6$              ;;BR IF NO
033764  116663  000001  177777               MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
033772  052702  000060                 6$:   BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
033776  052702  000040                 7$:   BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
034002  110223                               MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
034004  005720                               TST     (R0)+           ;;JUST INCREMENTING
034006  020027  000010                       CMP     R0,#10          ;;CHECK THE TABLE INDEX
034012  002746                               BLT     2$              ;;GO DO THE NEXT DIGIT
034014  003002                               BGT     8$              ;;GO TO EXIT
034016  010502                               MOV     R5,R2           ;;GET THE LSD
034020  000764                               BR      6$              ;;GO CHANGE TO ASCII
034022  105726                        8$:    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
034024  100003                               BPL     9$              ;;BR IF NO
034026  116663  177777  177776               MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
034034  105013                        9$:    CLRB    (R3)            ;;SET THE TERMINATOR
034036  012605                               MOV     (SP)+,R5        ;;POP STACK INTO R5
```

```
4206  034040  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
4207  034042  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
4208  034044  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4209  034046  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4210  034050  104401  034076                 TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
4211  034054  016666  000002  000004         MOV     2(SP),4(SP)     ;;ADJUST THE STACK
4212  034062  012616                          MOV     (SP)+,(SP)
4213  034064  000002                          RTI                     ;;RETURN TO USER
4214  034066  023420                $DTBL:    10000.
4215  034070  001750                          1000.
4216  034072  000144                          100.
4217  034074  000012                          10.
4218  034076  000004                $DBLK:    .BLKW   4
4219                                .SBTTL  SCOPE HANDLER ROUTINE
4220
4221                                ;;**********************************************************************
4222                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
4223                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
4224                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
4225                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4226                                ;*SW14=1         LOOP ON TEST
4227                                ;*SW11=1         INHIBIT ITERATIONS
4228                                ;*SW09=1         LOOP ON ERROR
4229                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
4230                                ;*CALL
4231                                ;*      SCOPE           ;;SCOPE-IOT
4232
4233  034106                        $SCOPE:
4234  034106  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
4235  034110  004737  046042                  JSR     PC,CTRLCG               ;TEST FOR CTRL C OR G
4236  034114  032777  040000  145016  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
4237  034122  001114                          BNE     $OVER           ;;YES IF SW14=1
4238                                ;#####START OF CODE FOR THE XOR TESTER#####
4239  034124  000416                $XTSTR: BR      6$              ;;IF RUNNING ON THE ''XOR'' TESTER CHANGE
4240                                                                ;;THIS INSTRUCTION TO A ''NOP'' (NOP=240)
4241  034126  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
4242  034132  012737  034152  000004         MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
4243  034140  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
4244  034144  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
4245  034150  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
4246  034152  022626                5$:       CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
4247  034154  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
4248  034160  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
4249  034162                        6$:·#####END OF CODE FOR THE XOR TESTER#####
4250  034162  032777  000400  144750          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
4251  034170  001404                          BEQ     2$              ;;BR IF NO
4252  034172  127737  144742  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
4253  034200  001465                          BEQ     $OVER           ;;BR IF YES
4254  034202  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
4255  034206  001421                          BEQ     3$              ;;BR IF NO
4256  034210  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
4257  034216  101015                          BHI     3$              ;;BR IF NO
4258  034220  032777  001000  144712          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
4259  034226  001404                          BEQ     4$              ;;BR IF NO
```

```
4260  034230  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
4261  034236  000446                          BR      $OVER
4262  034240  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
4263  034244  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
4264  034250  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
4265  034252  032777  004000  144660  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
4266  034260  001011                          BNE     1$              ;;BR IF YES
4267  034262  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
4268  034266  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
4269  034270  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
4270  034274  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
4271  034302  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
4272  034304  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
4273  034312  013737  034370  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
4274  034320  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
4275  034324  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
4276  034332  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
4277  034336  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
4278  034342  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
4279  034346  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4280  034354  013777  001102  144560  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
4281  034362  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
4282  034366  000002                          RTI                     ;;FIXES PS
4283  034370  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
4284                                  .SBTTL  TYPE ROUTINE
4285
4286                                  ;;**************************************************************
4287                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
4288                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
4289                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
4290                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
4291                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
4292                                  ;*
4293                                  ;*CALL:
4294                                  ;*1) USING A TRAP INSTRUCTION
4295                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
4296                                  ;*OR
4297                                  ;*      TYPE
4298                                  ;*      MESADR
4299                                  ;*
4300
4301  034372  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
4302  034376  100002                          BPL     1$              ;;BR IF YES
4303  034400  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
4304  034402  000430                          BR      3$              ;;LEAVE
4305  034404  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
4306  034406  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
4307  034412  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
4308  034420  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
4309  034422  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
4310  034430  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
4311  034432  010037  034442                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
4312  034436  004737  035444                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
4313  034442  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
```

```
4314  034444  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
4315  034452  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
4316  034454  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
4317  034456  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
4318  034460  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
4319  034462  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
4320  034464  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
4321  034470  000002                          RTI                     ;;RETURN
4322  034472  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
4323  034476  001430                          BEQ     8$
4324  034500  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
4325  034504  001006                          BNE     5$
4326  034506  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
4327  034510  104401                          TYPE                    ;;TYPE A CR AND LF
4328  034512  001171                          $CRLF
4329  034514  105037  034650                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
4330  034520  000755                          BR      2$              ;;GET NEXT CHARACTER
4331  034522  004737  034604          5$.     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
4332  034526  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
4333  034532  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
4334  034534  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
4335                                                                  ;;AND THE NULL CHAR.
4336  034540  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
4337  034544  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
4338  034546  004737  034604                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
4339  034552  105337  034650                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
4340  034556  000770                          BR      7$              ;;LOOP
4341
4342                                  ;HORIZONTAL TAB PROCESSOR
4343
4344  034560  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
4345  034564  004737  034604          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
4346  034570  132737  000007  034650          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
4347  034576  001372                          BNE     9$              ;;TAB STOP
4348  034600  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
4349  034602  000724                          BR      2$              ;;GET NEXT CHARACTER
4350  034604  105777  144340          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
4351  034610  100375                          BPL     $TYPEC
4352  034612  116677  000002  144332          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
4353  034620  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
4354  034626  001003                          BNE     1$              ;;BRANCH IF NO
4355  034630  105037  034650                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
4356  034634  000406                          BR      $TYPEX          ;;EXIT
4357  034636  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
4358  034644  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
4359  034646  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
4360  034650  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
4361  034652  000207          $TYPEX: RTS     PC
4362
```

```
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M               ;;M=1 OR 0
                                        ;*                                      ;;1=TYPE LEADING ZEROS
                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT

034654  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
034660  116637  000001  035077          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
034666  112637  035101                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
034672  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
034676  000406                          BR      $TYPON
034700  112737  000001  035077  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
034706  112737  000006  035101          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
034714  112737  000005  035076  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
034722  010346                          MOV     R3,-(SP)        ;;SAVE R3
034724  010446                          MOV     R4,-(SP)        ;;SAVE R4
034726  010546                          MOV     R5,-(SP)        ;;SAVE R5
034730  113704  035101                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
034734  005404                          NEG     R4
034736  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
034742  110437  035100                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
034746  113704  035077                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
034752  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
034756  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
034760  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
034762  000404                          BR      3$              ;;GO DO MSB
034764  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
034766  006105                          ROL     R5
034770  006105                          ROL     R5
034772  010503                          MOV     R5,R3
034774  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
034776  105337  035100                  DECB    $OMODE          ;;TYPE THIS DIGIT?
035002  100016                          BPL     7$              ;;BR IF NO
035004  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
035010  001002                          BNE     4$              ;;TEST FOR 0
```

```
4417  035012  005704                          TST     R4              ;;SUPPRESS THIS 0?
4418  035014  001403                          BEQ     5$              ;;BR IF YES
4419  035016  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
4420  035020  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
4421  035024  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
4422  035030  110337  035074                  MOVB    R3,8$           ;;SAVE FOR TYPING
4423  035034  104401  035074                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
4424  035040  105337  035076          7$:     DECB    $OCNT           ;;COUNT BY 1
4425  035044  003347                          BGT     2$              ;;BR IF MORE TO DO
4426  035046  002402                          BLT     6$              ;;BR IF DONE
4427  035050  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
4428  035052  000744                          BR      2$              ;;GO DO THE LAST DIGIT
4429  035054  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
4430  035056  012604                          MOV     (SP)+,R4        ;;RESTORE R4
4431  035060  012603                          MOV     (SP)+,R3        ;;RESTORE R3
4432  035062  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
4433  035070  012616                          MOV     (SP)+,(SP)
4434  035072  000002                          RTI                     ;;RETURN
4435  035074     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
4436  035075     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
4437  035076     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
4438  035077     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
4439  035100  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
4440
4441                                  .SBTTL  ERROR HANDLER ROUTINE
4442
4443                                  ;;**************************************************************************
4444                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
4445                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
4446                                  ;*AND GO TO $ERRTYP ON ERROR
4447                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4448                                  ;*SW15-1        HALT ON ERROR
4449                                  ;*SW13-1        INHIBIT ERROR TYPEOUTS
4450                                  ;*SW10=1        BELL ON ERROR
4451                                  ;*SW09=1        LOOP ON ERROR
4452                                  ;*CALL
4453                                  ;*      ERROR   N       ;;ERROR-EMT AND N-ERROR ITEM NUMBER
4454
4455  035102                          $ERROR:
4456  035102  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
4457  035104  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
4458  035110  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
4459  035112  013777  001102  144022          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
4460  035120  032777  002000  144012          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
4461  035126  001402                          BEQ     1$              ;;NO - SKIP
4462  035130  104401  001164                  TYPE    ,$BELL          ;;RING BELL
4463  035134  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
4464  035140  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
4465  035144  162737  000002  001116          SUB     #2,$ERRPC
4466  035152  117737  143740  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
4467  035160  032777  020000  143752          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
4468  035166  001004                          BNE     20$             ;;SKIP TYPEOUTS
4469  035170  004737  035302                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
4470  035174  104401  001171                  TYPE    ,$CRLF
```

```
4471  035200                                  20$:
4472  035200  122737  000001  001214                CMPB    #APTEN.,$ENV    ;;RUNNING IN APT MODE
4473  035206  001007                              BNE     2$              ;;NO,SKIP APT ERROR REPORT
4474  035210  113737  001114  035222              MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
4475  035216  004737  035454                      JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
4476  035222     000                  21$:        .BYTE   0
4477  035223     000                              .BYTE   0
4478  035224  000777                  22$:        BR      22$             ;;APT ERROR LOOP
4479  035226  005777  143706          2$:         TST     @SWR            ;;HALT ON ERROR
4480  035232  100002                              BPL     3$              ;;SKIP IF CONTINUE
4481  035234  000000                              HALT                    ;;HALT ON ERROR!
4482  035236  104407                              CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
4483  035240  032777  001000  143672  3$:         BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
4484  035246  001402                              BEQ     4$              ;;BR IF NO
4485  035250  013716  001110                      MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
4486  035254  005737  001162          4$:         TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
4487  035260  001402                              BEQ     5$              ;;BR IF NONE
4488  035262  013716  001162                      MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
4489  035266                          5$:
4490  035266  022737  030164  000042              CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
4491  035274  001001                              BNE     6$              ;;BRANCH IF NO
4492  035276  000000                              HALT                    ;;YES
4493  035300                          6$:
4494  035300  000002                              RTI                     ;;RETURN
4495
4496                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
4497
4498                                  ;;**************************************************************
4499                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
4500                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
4501                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
4502
4503  035302                          $ERRTYP:
4504  035302  104401  001171                      TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
4505  035306  010046                              MOV     R0,-(SP)        ;;SAVE R0
4506  035310  005000                              CLR     R0              ;;PICKUP THE ITEM INDEX
4507  035312  153700  001114                      BISB    @#$ITEMB,R0
4508  035316  001004                              BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
4509                                                                      ;;TYPE THE PC OF THE ERROR
4510  035320  013746  001116                      MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
4511                                                                      ;;ERROR ADDRESS
4512  035324  104402                              TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4513  035326  000426                              BR      6$              ;;GET OUT
4514  035330  005300                  1$:         DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
4515  035332  006300                              ASL     R0              ;;      WORK FOR THE ERROR TABLE
4516  035334  006300                              ASL     R0
4517  035336  006300                              ASL     R0
4518  035340  062700  001256                      ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
4519  035344  012037  035354                      MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
4520  035350  001404                              BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
4521  035352  104401                              TYPE                    ;;TYPE THE "ERROR MESSAGE"
4522  035354  000000                  2$:         .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
4523  035356  104401  001171                      TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
4524  035362  012037  035372          3$:         MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
```

```
 4525  035366  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
 4526  035370  104401                          TYPE                    ;;TYPE THE 'DATA HEADER'
 4527  035372  000000                  4$:     .WORD   0               ;;'DATA HEADER' POINTER GOES HERE
 4528  035374  104401  001171                  TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
 4529  035400  011000                  5$:     MOV     (R0),R0         ;;PICKUP 'DATA TABLE' POINTER
 4530  035402  001004                          BNE     7$              ;;GO TYPE THE DATA
 4531  035404  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
 4532  035406  104401  001171                  TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
 4533  035412  000207                          RTS     PC              ;;RETURN
 4534  035414                          7$:
 4535  035414  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
 4536  035416  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4537  035420  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
 4538  035422  001770                          BEQ     6$              ;;BR IF NO
 4539  035424  104401  035432                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
 4540  035430  000771                          BR      7$              ;;LOOP
 4541  035432  020040      000         8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
 4542          035436                          .EVEN
```

```
                                        .SBTTL  APT COMMUNICATIONS ROUTINE

                                        ;;**************************************************************************
035436  112737  000001  035702  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
035444  112737  000001  035700  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
035452  000403                  $ATY2:  BR      $ATYC
035454  112737  000001  035702  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
035462                          $ATYC:
035462  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
035464  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
035466  105737  035700                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
035472  001450                          BEQ     5$              ;;IF NOT:  BR
035474  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
035502  001031                          BNE     3$              ;;IF NOT:  BR
035504  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
035512  001425                          BEQ     3$              ;;IF NOT:  BR
035514  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
035520  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
035526  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
035532  001375                          BNE     1$              ;;IF NOT:  WAIT
035534  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
035540  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
035542  001376                          BNE     2$
035544  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
035550  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
035552  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
035556  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
035564  000413                          BR      5$
035566  017637  000004  035612  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
035574  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
035602  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
035606  004737  034372                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
035612  000000                  4$:     .WORD   0
035614                          5$:
035614  105737  035702          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
035620  001416                          BEQ     12$             ;;IF NOT:  BR
035622  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
035626  001413                          BEQ     12$             ;;IF NOT:  BR
035630  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
035634  001375                          BNE     11$             ;;IF NOT:  WAIT
035636  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
035644  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
035652  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
035656  105037  035702          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
035662  105037  035701                  CLRB    $LFLG           ;;CLEAR LOG FLAG
035666  105037  035700                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
035672  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
035674  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
035676  000207                          RTS     PC              ;;RETURN
035700     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
035701     000          $LFLG:  .BYTE   0               ;;LOG FLAG
035702     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
035704                          .EVEN
000200                  APTSIZE=200
```

```
                        000001                  APTENV=001
                        000100                  APTSPOOL=100
                        000040                  APTCSUP=040
                                                .SBTTL  POWER DOWN AND UP ROUTINES

                                                ;;*****************************************************************
                                                ;POWER DOWN ROUTINE
035704  012737  036044  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
035712  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
035720  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
035722  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
035724  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
035726  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
035730  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
035732  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
035734  017746  143200                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
035740  010637  036050                MOV     SP,$SAVR6       ;;SAVE SP
035744  012737  035756  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
035752  000000                        HALT
035754  000776                        BR      .-2             ;;HANG UP

                                                ;;*****************************************************************
                                                ;POWER UP ROUTINE
035756  012737  036044  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
035764  013706  036050                MOV     $SAVR6,SP       ;;GET SP
035770  005037  036050                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
035774  005237  036050        1$:     INC     $SAVR6          ;;WAIT FOR THE INC
036000  001375                        BNE     1$              ;;OF  WORD
036002  012677  143132                MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
036006  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
036010  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
036012  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
036014  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
036016  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
036020  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
036022  012737  035704  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
036030  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
036036  104401                        TYPE                    ;REPORT THE POWER FAILURE
036040  036052                $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
036042  000002                        RTI
036044  000000                $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
036046  000776                        BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
036050  000000                $SAVR6: 0                       ;;PUT THE SP HERE
036052  005015  047520  042527  $POWER: .ASCIZ  <15><12>'POWER'
036060  000122
                                        .EVEN
```

```
                                        .SBTTL  ROUTINE TO SIZE MEMORY

                                        ;;**********************************************************************
                                        ;*CALL:
                                        ;*      JSR     PC,$SIZE
                                        ;*      RETURN
                                        ;*$LSTAD WILL CONTAIN:
                                        ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
                                        ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
                                        ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
                                        ;*$KT11 IS THE MEMORY MANAGEMENT KEY
                                        ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
                                        ;*      MUST BE SETUP BEFORE THE CALL
                                        ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
                                        ;*      DETERMINED BY ROUTINE

036062  010046                  $SIZE: MOV     R0,-(SP)         ;;SAVE R0 ON THE STACK
036064  010146                         MOV     R1,-(SP)         ;;SAVE R1 ON THE STACK
036066  010246                         MOV     R2,-(SP)         ;;SAVE R2 ON THE STACK
036070  010346                         MOV     R3,-(SP)         ;;SAVE R3 ON THE STACK
036072  013746  000004                 MOV     @#ERRVEC,-(SP)   ;;SAVE PRESENT ERROR VECTOR PS & PC
036076  013746  000006                 MOV     @#ERRVEC+2,-(SP)
036102  010600                         MOV     SP,R0            ;;SAVE THE STACK POINTER
                                ;;SET THE ERRVEC PS TO THE PRESENT PS
036104  104400                         TRAP                     ;;PUSH OLD PSW AND PC ON STACK
036106  012637  000006                 MOV     (SP)+,@#ERRVEC+2         ;;SAVE THE PSW IN @#ERRVEC+2
036112  012701  003776                 MOV     #3776,R1         ;;SETUP ADDRESS
036116  105727                         TSTB    (PC)+            ;;USE MEMORY MANAGEMENT?
036120  000200                  $KT11: .WORD   200              ;;SET TO USE MEMORY MANAGEMENT
036122  100070                         BPL     $CORE            ;;BR IF NO
036124  012737  036276  000004         MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
036132  005737  177572                 TST     @#SR0            ;;KT11 ARE YOU THERE?
036136  052737  100000  036120         BIS     #100000,$KT11    ;;YES--SET KT11 KEY
036144  012737  036404  000250         MOV     #KTERR,@#MMVEC   ;;SET IN CASE OF ERROR
036152  012737  000340  000252         MOV     #340,@#MMVEC+2
036160  005046                         CLR     -(SP)            ;;INITIALIZE FOR 'PAR' LOADING
036162  012702  172340                 MOV     #KIPAR0,R2       ;;ADDRESS OF FIRST 'PAR'
036166  012703  000010                 MOV     #^D8,R3          ;;LOAD EIGHT 'PAR.'S' AND EIGHT 'PDR.'S'
036172  012762  077406  177740  1$:    MOV     #77406,-40(R2)   ;;PDR = 4K, UP, READ/WRITE
036200  011622                         MOV     (SP),(R2)+       ;;LOAD 'PAR'
036202  062716  000200                 ADD     #200,(SP)        ;;UPDATE FOR NEXT 'PAR'
036206  077307                         SOB     R3,1$            ;;LOOP UNTIL ALL EIGHT ARE LOADED
036210  012742  177600                 MOV     #177600,-(R2)    ;;SETUP KIPAR7 FOR I/O
036214  005042                         CLR     -(R2)            ;;SETUP KIPAR6 FOR TESTING
036216  012737  036234  000004         MOV     #2$,@#ERRVEC     ;;CATCH TIMEOUT IF NO SR3
036224  012737  000020  172516         MOV     #20,@#SR3        ;;ENABLE 22 BIT MODE
036232  000401                         BR      3$               ;;THIS PDP-11 HAS A SR3 REGISTER
036234  022626                  2$:    CMP     (SP)+,(SP)+      ;;CLEAN OFF THE STACK--NO SR3
036236  005237  177572          3$:    INC     @#SR0            ;;TURN ON MEMORY MANAGEMENT
036242  012737  036266  000004         MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
036250  005737  143776          4$:    TST     @#143776         ;;TRAP ON NON-EX-MEM
036254  062712  000040                 ADD     #40,(R2)         ;;MAKE A 1K STEP
036260  023712  172356                 CMP     @#KIPAR7,(R2)    ;;LAST ONE?
036264  101371                         BHI     4$               ;;NO--TRY IT
```

```
 4697  036266  011202                 $KTOUT: MOV     (R2),R2          ;;GET LAST BANK+1
 4698  036270  005037  177572                 CLR     @#SR0            ;;TURN OFF MEMORY MANAGEMENT
 4699  036274  000421                         BR      $SIZEX
 4700  036276  042737  100000  036120 $KTNEX: BIC     #100000,$KT11    ;;KT11 NON-EXISTENT
 4701  036304  012737  036334  000004 $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
 4702  036312  005002                         CLR     R2               ;;SET UP BANK
 4703  036314  062701  004000         1$:     ADD     #4000,R1         ;;INCREMENT BY 1K
 4704  036320  062702  000040                 ADD     #40,R2           ;;1K STEP
 4705  036324  005711                         TST     (R1)             ;;TRAP ON TIME OUT
 4706  036326  022701  177776                 CMP     #177776,R1       ;;LAST ONE
 4707  036332  001370                         BNE     1$               ;;NO--TRY AGAIN
 4708  036334  162701  004000 $CROUT: SUB     #4000,R1
 4709  036340  162702  000040 $SIZEX: SUB     #40,R2           ;;DROP BACK
 4710  036344  010006                         MOV     R0,SP            ;;RESTORE THE STACK
 4711  036346  012637  000006                 MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
 4712  036352  012637  000004                 MOV     (SP)+,@#ERRVEC
 4713  036356  010137  036400                 MOV     R1,$LSTAD        ;;LAST ADDRESS
 4714  036362  010237  036402                 MOV     R2,$LSTBK        ;;LAST BANK
 4715  036366  012603                         MOV     (SP)+,R3         ;;RESTORE R3
 4716  036370  012602                         MOV     (SP)+,R2         ;;RESTORE R2
 4717  036372  012601                         MOV     (SP)+,R1         ;;RESTORE R1
 4718  036374  012600                         MOV     (SP)+,R0         ;;RESTORE R0
 4719  036376  000207                         RTS     PC
 4720  036400  000000         $LSTAD: .WORD   0                ;;CONTAINS THE LAST ADDRESS
 4721  036402  000000         $LSTBK: .WORD   0                ;;CONTAINS THE LAST BANK
 4722  036404  000000         KTERR:  HALT                     ;KT11 FAILURE
 4723  036406  000776                 BR      KTERR
```

```
                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

                                ;;******************************************************************
                                ;*SAVE R0-R5
                                ;*CALL:
                                ;*      SAVREG
                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                ;*
                                ;*TOP---(+16)
                                ;* +2---(+18)
                                ;* +4---R5
                                ;* +6---R4
                                ;* +8---R3
                                ;*+10---R2
                                ;*+12---R1
                                ;*+14---R0

036410                          $SAVREG:
036410  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
036412  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
036414  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
036416  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
036420  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
036422  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
036424  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
036430  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
036434  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
036440  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
036444  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
036446                          $RESREG:
036446  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
036452  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
036456  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
036462  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
036466  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
036470  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
036472  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
036474  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
036476  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
036500  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
036502  000002                          RTI
```

```
                                        .SBTTL  INTEGER MULTIPLY ROUTINE

                                        ;;*****************************************************************
                                        ;*CALL
                                        ;*      MOV     MULTIPLER,-(SP)
                                        ;*      MOV     MULTIPLICAND,-(SP)
                                        ;*      JSR     PC,@#$MULT
                                        ;*      RETURN  ;;PRODUCT IS ON THE STACK
                                        ;*
                                        ;*      STACK   PRODUCT
                                        ;*      -----   -------
                                        ;*      TOP     LSB'S
                                        ;*      +2      MSB'S

036504                                  $MULT:
036504  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
036506  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
036510  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
036512  005046                                 CLR     -(SP)           ;;CLEAR THE SIGN KEY
036514  016601  000012                         MOV     12(SP),R1       ;;GET THE MULTIPLICAND
036520  100002                                 BPL     1$              ;;BR IF PLUS
036522  005216                                 INC     (SP)            ;;SET THE SIGN KEY
036524  005401                                 NEG     R1              ;;MAKE THE MULTIPLICAND POSTIVE
036526  016602  000014            1$:          MOV     14(SP),R2       ;;GET THE MULTIPLIER
036532  100002                                 BPL     2$              ;;BR IF PLUS
036534  005316                                 DEC     (SP)            ;;UPDATE THE SIGN KEY
036536  005402                                 NEG     R2              ;;MAKE THE MULTIPLIER POSTIVE
036540  012746  000021            2$:          MOV     #17.,-(SP)      ;;SET THE LOOP COUNT
036544  005000                                 CLR     R0              ;;SETUP FOR THE MULTIPLY LOOP
036546  103001                    3$:          BCC     4$              ;;DON'T ADD IF MULTIPLICAND = 0
036550  060200                                 ADD     R2,R0
036552  006000                    4$:          ROR     R0              ;;POSITION THE PARITIAL PRODUCT AND
036554  006001                                 ROR     R1              ;;THE MULTIPLICAND
036556  005316                                 DEC     (SP)            ;;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
036560  001372                                 BNE     3$              ;;BR IF NO
036562  022616                                 CMP     (SP)+,(SP)      ;;SHOULD PRODUCT BE NEGATIVE?
036564  001403                                 BEQ     5$              ;;GO TO EXIT IF NO
036566  005400                                 NEG     R0              ;;YES--SO MAKE IT SO
036570  005401                                 NEG     R1
036572  005600                                 SBC     R0
036574  005726                    5$:          TST     (SP)+           ;;CLEAR SIGN INFO. OFF OF STACK
036576  010066  000012                         MOV     R0,12(SP)       ;;PUT THE PRODUCT ON THE STACK (MSB'S)
036602  010166  000010                         MOV     R1,10(SP)       ;;LSB'S
036606  012602                                 MOV     (SP)+,R2        ;;POP STACK INTO R2
036610  012601                                 MOV     (SP)+,R1        ;;POP STACK INTO R1
036612  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
036614  000207                                 RTS     PC
```

```
                                .SBTTL  INTEGER DIVIDE ROUTINE

                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
                                ;*DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
                                ;*A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER.
                                ;*DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
                                ;*SAVE SIGN AS THE DIVIDEND.
                                ;*CALL:
                                ;*      MOV     LOW DIVIDEND,-(SP) ;;THE HIGH DIVIDEND MUST BE < 1/2
                                ;*      MOV     HIGH DIVIDEND,-(SP);    AS LARGE AS THE DIVISOR
                                ;*      MOV     DIVISOR,-(SP)
                                ;*      JSR     PC,$DIV
                                ;*      RETURN                  ;;QUOTIENT & REMAINDER ARE ON THE STACK
                                ;*      'V' 0   IMPLIES NO ERROR
                                ;*      'V' 1   IMPLIES ERROR OCCURRED
                                ;*              'C'=0   DIVIDE OVERFLOW OCCURRED
                                ;*              'C'=1   ATTEMPTED TO DIVIDE BY ZERO
                                ;*
                                ;*
                                ;*      STACK   NO ERROR        OVERFLOW        DIVIDE BY ZERO
                                ;*      -----   --------        --------        --------------
                                ;*      TOP     REMAINDER       ALL ZEROS       ALL ONES
                                ;*      +2      QUOTIENT        ALL ZEROS       ALL ONES

036616                          $DIV:
036616  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
036620  042716  000017                  BIC     #17,(SP)        ;;STRIP AWAY CONDITION CODES
036624  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
036626  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
036630  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
036632  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
036634  005046                          CLR     -(SP)           ;;SAVE A PLACE FOR SIGNS
036636  012746  000021                  MOV     #17.,-(SP)      ;;SETUP THE ITERATION COUNTER
036642  016601  000024                  MOV     24(SP),R1       ;;PICKUP THE DIVIDEND
036646  016600  000022                  MOV     22(SP),R0
036652  100005                          BPL     1$              ;;CHECK THE SIGN
036654  105366  000003                  DECB    3(SP)           ;;KEEP TRACK OF THE SIGN
036660  005400                          NEG     R0              ;;AND NEGATE THE ORIGINAL
036662  005401                          NEG     R1              ;;NUMBER
036664  005600                          SBC     R0
036666  016602  000020          1$:     MOV     20(SP),R2       ;;PICKUP THE DIVISOR
036672  002407                          BLT     2$              ;;CHECK THE SIGN
036674  003011                          BGT     3$              ;;DIVISOR OF 0 IS A NO-NO
036676  052766  000003  000014          BIS     #3,14(SP)       ;;SET 'V' & 'C'
036704  012700  177777                  MOV     #-1,R0          ;;SET REMAINDER TO ALL ONES
036710  000424                          BR      7$              ;;EXIT
036712  005266  000002          2$:     INC     2(SP)           ;;KEEP TRACK OF DIVISORS SIGN
036716  000401                          BR      4$
036720  005402                  3$:     NEG     R2              ;;NEGATE THE ORIGINAL NUMBER
036722  000241                  4$:     CLC                     ;;CLEAR 'C'
036724  000405                          BR      6$              ;;START FORMING QUOTIENT
036726  006100                  5$:     ROL     R0              ;;POSITION MSB'S
036730  010003                          MOV     R0,R3           ;;COPY
```

```
 4870  036732  060203                        ADD     R2,R3           ;;COMPARE DIVIDEND & DIVISOR
 4871  036734  103001                        BCC     6$              ;;BR IF DIVIDEND > DIVISOR
 4872  036736  010300                        MOV     R3,R0           ;;REMAINDER AFTER THIS LOOP
 4873  036740  006101                6$:     ROL     R1              ;;QUOTIENT BIT ENTERS HERE
 4874  036742  005316                        DEC     (SP)            ;;DONE?
 4875  036744  001370                        BNE     5$              ;;BR IF NO
 4876  036746  005701                        TST     R1              ;;OVERFLOW?
 4877  036750  100005                        BPL     8$              ;;BR IF NO
 4878  036752  052766  000002  000014        BIS     #2,14(SP)       ;;SET 'V' IN RETURN STATUS WORD
 4879  036760  005000                        CLR     R0              ;;SET REMAINDER TO ALL ZEROS
 4880  036762  010001                7$:     MOV     R0,R1           ;;COPY REMAINDER INTO QUOTIENT
 4881  036764  005726                8$:     TST     (SP)+           ;;CLEAR COUNTER FROM STACK
 4882  036766  005716                        TST     (SP)            ;;REMAINDER SIGN CORRECTION NEEDED?
 4883  036770  002004                        BGE     9$              ;;BR IF NO
 4884  036772  005400                        NEG     R0              ;;NEGATE REMAINDER
 4885  036774  105066  000001              CLRB    1(SP)           ;;CLEAR SIGN
 4886  037000  005316                        DEC     (SP)            ;;BUT DON'T FORGET QUOTIENT
 4887  037002  005726                9$:     TST     (SP)+           ;;QUOTIENT SIGN CORRECTION NEEDED?
 4888  037004  001401                        BEQ     10$             ;;BR IF NO
 4889  037006  005401                        NEG     R1              ;;NEGATE QUOTIENT
 4890  037010  010166  000020        10$:    MOV     R1,20(SP)       ;;RETURN QUOTIENT AND
 4891  037014  010066  000016                MOV     R0,16(SP)       ;;REMAINDER TO USER
 4892  037020  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 4893  037022  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 4894  037024  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 4895  037026  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 4896  037030  012666  000002                MOV     (SP)+,2(SP)     ;;SETUP TO RETURN CONDITION CODES
 4897  037034  000002                        RTI                     ;;RETURN
```

```
                                .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                                ;;*********************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                ;*POSITIVE.
                                ;*CALL
                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                ;*      JSR     PC,@#$DB2D
                                ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                ;;IS ON THE STACK

037036  104413          $DB2D:  SAVREG                          ;;SAVE REGISTERS
037040  016602  000002          MOV     2(SP),R2                ;;PICKUP THE DATA POINTER
037044  012700  037216          MOV     #$DECVL,R0              ;;GET ADDRESS OF '$DECVL' STRING
037050  010066  000002          MOV     R0,2(SP)                ;;PUT ADDRESS OF ASCIZ STRING ON STACK
037054  012201                  MOV     (R2)+,R1                ;;PICKUP THE BINARY NUMBER
037056  012202                  MOV     (R2)+,R2
037060  012737  000012  037134  MOV     #10.,4$                 ;;SET UP TO DO 10 CONVERSIONS
037066  012704  037146          MOV     #$TNPWR,R4              ;;ADDRESS OF TEN POWER
037072  012705  037150          MOV     #$TNPWR+2,R5
037076  005003          1$:     CLR     R3                      ;;CLEAR PARTIAL
037100  161401          2$:     SUB     (R4),R1                 ;;SUBTRACT TEN POWER
037102  005602                  SBC     R2
037104  161502                  SUB     (R5),R2
037106  002402                  BLT     3$                      ;;BR IF TEN POWER TO LARGE
037110  005203                  INC     R3                      ;;ADD 1 TO PARTIAL
037112  000772                  BR      2$                      ;;LOOP
037114  062401          3$:     ADD     (R4)+,R1                ;;RESTORE SUBTRACTED VALUE
037116  005502                  ADC     R2
037120  062402                  ADD     (R4)+,R2
037122  022525                  CMP     (R5)+,(R5)+             ;;MOVE TO NEXT TEN POWER
037124  052703  000060          BIS     #'0,R3                  ;;CHANGE PARTIAL TO ASCII
037130  110320                  MOVB    R3,(R0)+                ;;SAVE IT
037132  005327                  DEC     (PC)+                   ;;DONE?
037134  000000          4$:     .WORD   0
037136  001357                  BNE     1$                      ;;BR IF NO
037140  105020                  CLRB    (R0)+                   ;;TERMINATOR
037142  104414                  RESREG                          ;;RESTORE REGISTERS
037144  000207                  RTS     PC                      ;;RETURN
037146  145000          $TNPWR: 145000                          ;;1.0E09
037150  035632                  35632
037152  160400                  160400                          ;;1.0E08
037154  002765                  2765
037156  113200                  113200                          ;;1.0E07
037160  000230                  230
037162  041100                  041100                          ;;1.0E06
037164  000017                  17
037166  103240                  103240                          ;;1.0E05
037170  000001                  1
037172  023420                  23420                           ;;1.0E04
037174  000000                  0
037176  001750                  1750                            ;;1.0E03
```

```
 4952  037200  000000                          0
 4953  037202  000144                          144                     ;;1.0E02
 4954  037204  000000                          0
 4955  037206  000012                          12                      ;;1.0E01
 4956  037210  000000                          0
 4957  037212  000001                          1                       ;;1.0E00
 4958  037214  000000                          0
 4959  037216  000014              $DECVL: .BLKB   12.                 ;;RESERVE STORAGE FOR ASCIZ STRING
 4960                              .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
 4961
 4962                              ;;**********************************************************************
 4963                              ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
 4964                              ;*UNSIGNED DECIMAL ASCIZ NUMBER.
 4965                              ;*CALL
 4966                              ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
 4967                              ;*      JSR     PC,@#$SB2D      ;;CALL
 4968                              ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK
 4969
 4970
 4971  037232  016637  000002  037262  $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
 4972  037240  012746  037262                  MOV     #1$,-(SP)       ;;SET POINTER
 4973  037244  004737  037036                  JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
 4974  037250  062716  000005                  ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
 4975  037254  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
 4976  037260  000207                          RTS     PC              ;;RETURN
 4977  037262  000000  000000          1$:     .WORD   0,0
```

```
                                .SBTTL  TRAP DECODER

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

037266  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
037270  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
037274  005740                  TST     -(R0)           ;;BACKUP BY 2
037276  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
037300  006300                  ASL     R0              ;;POSITION FOR INDEXING
037302  016000  037322          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
037306  000200                  RTS     R0              ;;GO TO ROUTINE


                                ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO

037310  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
037312  016666  000004  000002  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
037320  000002                  RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE ''TRAP'' INSTRUCTION.

                                ;       ROUTINE
                                ;       -------
037322  037310          $TRPAD: .WORD   $TRAP2
037324  034372                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
037326  034700                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
037330  034654                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
037332  034714                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
037334  033662                  $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

037336  033054                  $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

037340  033004                  $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
037342  033266                  $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
037344  033406                  $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
037346  033560                  $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
037350  036410                  $SAVREG ;;CALL=SAVREG   TRAP+13(104413) SAVE R0-R5 ROUTINE
037352  036446                  $RESREG ;;CALL=RESREG   TRAP+14(104414) RESTORE R0-R5 ROUTINE
```

```
                                                        .SBTTL  A TO D FIELD SITE AND ADJUSTMENT LOOP
037354  004737  046042          FSITE:  JSR     PC,CTRLCG       ;TEST FOR CTRL C OR G
                                ;CAMERA #0
037360  032777  000100  141552          BIT     #SW06,@SWR      ;TEST IF CAM 0 G=1
037366  001004                          BNE     1$
037370  004537  040102                  JSR     R5,CONVRT       ;CONVERT
037374  000000                          0                       ;CAMERA 0
037376  040474                          CAM0G1                  ;STORE RESULTS
037400  032777  000001  141532  1$:     BIT     #SW00,@SWR      ;TEST IF CAM 0 G=2
037406  001004                          BNE     2$
037410  004537  040102                  JSR     R5,CONVRT       ;CONVERT
037414  002000                          BIT10                   ;GAIN = 2
037416  040476                          CAM0G2
                                ;CAMERA #1
037420  032777  000200  141512  2$:     BIT     #SW7,@SWR       ;TEST IF CAM 1 G=1
037426  001004                          BNE     3$
037430  004537  040102                  JSR     R5,CONVRT       ;CONVERT
037434  000400                          BIT8                    ;CAMERA 1
037436  040500                          CAM1G1                  ;STORE RESULTS
037440  032777  000002  141472  3$:     BIT     #SW01,@SWR      ;TEST IF CAM 1 G=2
037446  001004                          BNE     4$
037450  004537  040102                  JSR     R5,CONVRT       ;CONVERT
037454  002400                          BIT10!BIT8              ;CAMERA 1 GAIN - 2
037456  040502                          CAM1G2                  ;RESULTS
                                ;CAMERA #2
037460  032777  000400  141452  4$:     BIT     #SW8,@SWR       ;TEST IF CAM 2 G=1
037466  001004                          BNE     5$
037470  004537  040102                  JSR     R5,CONVRT       ;CONVERT
037474  001000                          BIT9                    ;CAMERA #2
037476  040504                          CAM2G1                  ;RESULTS
037500  032777  000004  141432  5$:     BIT     #SW2,@SWR       ;TEST IF CAM 2 G-2
037506  001004                          BNE     6$
037510  004537  040102                  JSR     R5,CONVRT       ;CONVERT
037514  003000                          BIT10.BIT9              ;GAIN - 2
037516  040506                          CAM2G2                  ;RESULTS
                                ;CAMERA #3
037520  032777  001000  141412  6$:     BIT     #SW9,@SWR       ;TEST IF CAM 3 G=1
037526  001004                          BNE     7$
037530  004537  040102                  JSR     R5,CONVRT       ;CONVERT
037534  001400                          BIT9!BIT8               ;CAMERA #3
037536  040510                          CAM3G1                  ;RESULTS
037540  032777  000010  141372  7$.     BIT     #SW3,@SWR       ;TEST IF CAM 3 G=2
037546  001004                          BNE     10$
037550  004537  040102                  JSR     R5,CONVRT       ;CONVERT
037554  003400                          BIT10.BIT9!BIT8         ;GAIN = 2
037556  040512                          CAM3G2                  ;RESULTS
                                ;JOYSTICK
037560  032777  000020  141352  10$:    BIT     #SW4,@SWR       ;TEST IF JOYSTICK
037566  001011                          BNE     CALRPT
037570  052777  000001  142160          BIS     #BIT0,@SFR      ;ASK FOR JOYSTICK
037576  105777  142154          11$:    TSTB    @SFR            ;WAIT FOR JOY DONE
037602  100375                          BPL     11$
037604  017737  142152  040514          MOV     @JOY,JOYG1      ;SAVE THE RESULT
```

```
                                        ;NOW TEST IS TYPEOUT IS ENABLED
                                        
037612  004537  040060          CALRPT: JSR     R5,CKTSWR       ;TEST BIT 6
037616  000100                          BIT6
037620  000405                          BR      1$              ;BR IF SET
037622  104401  040516                  TYPE,   CAM0TX          ;REPORT CAMERA #0
037626  004537  040412                  JSR     R5,CAMUNP       ;REPORT VALUES
037632  040474                          CAM0G1
037634  004537  040060          1$:     JSR     R5,CKTSWR       ;TEST BIT 0
037640  000001                          BIT0
037642  000405                          BR      2$
037644  104401  040535                  TYPE,   CAM0TW          ;REPORT G-2
037650  004537  040412                  JSR     R5,CAMUNP
037654  040476                          CAM0G2
                                        
037656  004537  040060          2$:     JSR     R5,CKTSWR       ;TEST BIT 7
037662  000200                          BIT7
037664  000405                          BR      3$              ;BR IF SET
037666  104401  040554                  TYPE,   CAM1TX          ;REPORT CAMERA #1
037672  004537  040412                  JSR     R5,CAMUNP       ;REPORT VALUES
037676  040500                          CAM1G1
037700  004537  040060          3$:     JSR     R5,CKTSWR       ;TEST BIT 1
037704  000002                          BIT1
037706  000405                          BR      4$
037710  104401  040573                  TYPE,   CAM1TW          ;REPORT CAM 1 G=2
037714  004537  040412                  JSR     R5,CAMUNP
037720  040502                          CAM1G2
                                        
037722  004537  040060          4$:     JSR     R5,CKTSWR       ;TEST BIT 8
037726  000400                          BIT8
037730  000405                          BR      5$              ;BR IF SET
037732  104401  040612                  TYPE,   CAM2TX          ;REPORT CAMERA #2
037736  004537  040412                  JSR     R5,CAMUNP       ;REPORT VALUES
037742  040504                          CAM2G1
037744  004537  040060          5$:     JSR     R5,CKTSWR       ;TEST BIT 2
037750  000004                          BIT2
037752  000405                          BR      6$
037754  104401  040631                  TYPE,   CAM2TW
037760  004537  040412                  JSR     R5,CAMUNP       ;REPORT CAMERA #2 G=2
037764  040506                          CAM2G2
                                        
037766  004537  040060          6$:     JSR     R5,CKTSWR       ;TEST BIT 9
037772  001000                          BIT9
037774  000405                          BR      7$              ;BR IF SET
037776  104401  040650                  TYPE,   CAM3TX          ;REPORT CAMERA #3
040002  004537  040412                  JSR     R5,CAMUNP       ;REPORT VLAUES
040006  040510                          CAM3G1
040010  004537  040060          7$:     JSR     R5,CKTSWR       ;TEST BIT 3
040014  000010                          BIT3
040016  000405                          BR      10$
040020  104401  040667                  TYPE,   CAM3TW          ;REPORT CAMERA #3 G-2
040024  004537  040412                  JSR     R5,CAMUNP
040030  040512                          CAM3G2
```

```
 5131  040032  004537  040060          10$:    JSR     R5,CKTSWR       ;TEST BIT 4
 5132  040036  000020                          BIT4
 5133  040040  000405                          BR      1'$
 5134  040042  104401  040706                  TYPE,   JOYTW
 5135  040046  004537  040412                  JSR     R5,CAMUNP       ;REPORT JOYSTICK
 5136  040052  040514                          JOYG1
 5137  040054  000137  037354          11$:    JMP     FSITE
 5138
 5139  040060  032577  141054          CKTSWR: BIT     (R5)+,@SWR              ;TEST IF INHIBIT THIS CAMERA
 5140  040064  001005                          BNE     1$                      ;BR IF YES
 5141  040066  032777  020000  141044          BIT     #SW13,@SWR              ;TEST INHIBIT TYPE-OUT
 5142  040074  001001                          BNE     1$                      ;BR IF YES
 5143  040076  005725                          TST     (R5)+                   ;BUMP EXIT POINTER
 5144  040100  000205                  1$:     RTS     R5                      ;EXIT
```

```
                                        ;SUBROUTINE TO TAKE 8 CONVERSIONS AND AVERAGE FOUR OF THEM
040102 012537 040406                    CONVRT: MOV     (R5)+,11$       ;SAVE CAMERA CHANNEL
040106 012777 004000 141642             4$:     MOV     #CLRALL,@SFR            ;CLEAR THE DEVICE
040114 013777 040406 141624                     MOV     11$,@CSR                ;SELECT CAMERA AND GAIN
040122 012777 060000 141624                     MOV     #BUF0,@BAR              ;LOAD BUS ADDRESS
040130 012777 177770 141614                     MOV     #-10,@WCR               ;LOAD WORD COUNT
040136 000240                                   NOP
040140 000240                                   NOP
040142 000240                                   NOP
040144 000240                                   NOP
040146 013777 040410 141602                     MOV     INOUTZ,@SFR             ;SET Z INPUTS IF ENABLED
040154 000240                                   NOP
040156 000240                                   NOP
040160 012700 000004                            MOV     #4.,R0                  ;LOAD COUNTER
040164 005001                                   CLR     R1                      ;FOR OUT OF RANGE VOLTAGE
040166 000240                                   NOP
040170 052777 000001 141550                     BIS     #BIT0,@CSR              ;ENABLE THE DEVICE
040176 105777 141544                    1$:     TSTB    @CSR                    ;WAIT FOR IDLE
040202 100430                                   BMI     3$
040204 005301                                   DEC     R1                      ;DELAY
040206 001373                                   BNE     1$
040210 004737 046042                            JSR     PC,CTRLCG               ;TEST FOR CTRL G OR C
040214 005300                                   DEC     R0                      ;FINISHED OUT OF RANGE TIMER
040216 001367                                   BNE     1$
040220 032777 020000 140712                     BIT     #SW13,@SWR              ;TEST IF INHIBIT TYPEOUT
040226 001010                                   BNE     5$                      ;BR IF INHIBIT
040230 113700 040407                            MOVB    11$+1,R0                ;GET CAMERA #
040234 062700 000060                            ADD     #60,R0                  ;MAKE INTO ASCII
040240 110037 032714                            MOVB    R0,OUTCHN               ;INSERT INTO ASCII ERROR MESSAGE
040244 104401 032621                            TYPE,   OUTRNG                  ;TELL OPERATOR
040250 032777 100000 140662             5$:     BIT     #SW15,@SWR              ;TEST IF HALT ON NO CONVERSION ERROR
040256 001401                                   BEQ     6$                      ;BR IF CLEARED
040260 000000                                   HALT    ;NO CONVERT FLAG - INPUT VOLTAGE WAS OUT OF RANGE OR NO ''Z'' PULSES
040262 000711                           6$:     BR      4$
040264 012700 060000                    3$:     MOV     #BUF0,R0                ;LOAD POINTER
040270 005003                                   CLR     R3                      ;CLEAR RESULT
040272 005004                                   CLR     R4
040274 012737 000004 040404                     MOV     #4,10$                  ;LOAD COUNTER
040302 112002                           2$:     MOVB    (R0)+,R2                ;GET LOW BYTE
040304 112001                                   MOVB    (R0)+,R1                ;GET HIGH BYTE
040306 042702 177400                            BIC     #177400,R2
040312 042701 177400                            BIC     #177400,R1
040316 060104                                   ADD     R1,R4                   ;UPDATE RESULT
040320 060203                                   ADD     R2,R3                   ;UPDATE RESULT
040322 005337 040404                            DEC     10$                     ;FINISHED ?
040326 001365                                   BNE     2$
040330 006203                                   ASR     R3
040332 006203                                   ASR     R3
040334 000240                                   NOP
040336 000240                                   NOP
040340 006204                                   ASR     R4
040342 006204                                   ASR     R4
040344 000240                                   NOP
040346 000240                                   NOP
```

```
5199  040350  000240                          NOP
5200  040352  010437  040404                  MOV     R4,10$                  ;SAVE IT
5201  040356  105037  040405                  CLRB    10$+1                   ;CLEAR HIGH BYTE
5202  040362  110337  040405                  MOVB    R3,10$+1                ;LOAD HIGH BYTE
5203  040366  013735  040404                  MOV     10$,@(R5)+              ;SAVE THE RESULT
5204  040372  000240                          NOP
5205  040374  000240                          NOP
5206  040376  000240                          NOP
5207  040400  000240                          NOP
5208  040402  000205                          RTS     R5
5209  040404  000000                  10$:    0
5210  040406  000000                  11$:    0
5211  040410  000002                  INOUTZ: TESTZ           ;0= USE CUSTOMER Z PULSES, 1= MAINT Z PULSES
5212
5213                                  ;SUBROUTINE TO TYPE THE CAMERA MESSAGE
5214  040412  013537  040470          CAMUNP: MOV     @(R5)+,10$              ;GET  RESULT
5215  040416  113737  040471  040472          MOVB    10$+1,11$
5216  040424  105037  040471                  CLRB    10$+1
5217  040430  105037  040473                  CLRB    11$+1
5218  040434  013746  040470                  MOV     10$,-(SP)
5219  040440  104403                          TYPOS
5220  040442     003     001                  .BYTE   3,1                     ;TYPE 3 DIGIT NUMBER
5221  040444  104401  040724                  TYPE,   G1B                     ;AND TRALING ASCII
5222  040450  013746  040472                  MOV     11$,-(SP)
5223  040454  104403                          TYPOS
5224  040456     003     001                  .BYTE   3,1                     ;TYPE 3 DIGIT NUMBER
5225  040460  000240                          NOP
5226  040462  000240                          NOP
5227  040464  000240                          NOP
5228  040466  000205                          RTS     R5                      ;EXIT
5229  040470  000000                  10$:    0
5230  040472  000000                  11$:    0
5231  040474  000000                  CAM0G1: 0
5232  040476  000000                  CAM0G2: 0
5233  040500  000000                  CAM1G1: 0
5234  040502  000000                  CAM1G2: 0
5235  040504  000000                  CAM2G1: 0
5236  040506  000000                  CAM2G2: 0
5237  040510  000000                  CAM3G1: 0
5238  040512  000000                  CAM3G2: 0
5239  040514  000000                  JOYG1:  0
5240
5241  040516     015     012          CAM0TX: .BYTE   15,12
5242  040520  040503  030115  020060          .ASCIZ  /CAM00 G=1 X=/
5243  040526  036507  020061  036530
5244  040534     000
5245  040535     015     012          CAM0TW: .BYTE   15,12
5246  040537     103  046501  030060          .ASCIZ  /CAM00 G=2 X=/
5247  040544  043440  031075  054040
5248  040552  000075
5249  040554     015     012          CAM1TX: .BYTE   15,12
5250  040556  040503  030115  020061          .ASCIZ  /CAM01 G=1 X=/
5251  040564  036507  020061  036530
5252  040572     000
```

```
 5253  040573     015     012          CAM1TW: .BYTE   15,12
 5254  040575     103  046501  030460          .ASCIZ  /CAM01 G=2 X-/
 5255  040602  043440  031075  054040
 5256  040610  000075
 5257  040612     015     012          CAM2TX: .BYTE   15,12
 5258  040614  040503  030115  020062          .ASCIZ  /CAM02 G=1 X=/
 5259  040622  036507  020061  036530
 5260  040630     000
 5261  040631     015     012          CAM2TW: .BYTE   15,12
 5262  040633     103  046501  031060          .ASCIZ  /CAM02 G=2 X=/
 5263  040640  043440  031075  054040
 5264  040646  000075
 5265  040650     015     012          CAM3TX: .BYTE   15,12
 5266  040652  040503  030115  020063          .ASCIZ  /CAM03 G=1 X=/
 5267  040660  036507  020061  036530
 5268  040666     000
 5269  040667     015     012          CAM3TW: .BYTE   15,12
 5270  040671     103  046501  031460          .ASCIZ  /CAM03 G=2 X=/
 5271  040676  043440  031075  054040
 5272  040704  000075
 5273  040706                       JOYTX:
 5274  040706     015     012          JOYTW:  .BYTE   15,12
 5275  040710  047512  051531  044524          .ASCIZ  /JOYSTICK X=/
 5276  040716  045503  054040  000075
 5277  040724  054440  000075       G1B:    .ASCIZ  / Y=/
 5278  040730  005015  044504  043106  GAIN:   .ASCII  <15><12>/DIFFERENTIAL LINEARITY:/
 5279  040736  051105  047105  044524
 5280  040744  046101  046040  047111
 5281  040752  040505  044522  054524
 5282  040760     072
 5283  040761     040  020040  040507  GAINX:  .ASCII  /   GAIN = /
 5284  040766  047111  036440     040
 5285  040773     061     040     040  GAIN1:  .BYTE   61,40,40,0
 5286  040776     000
 5287  040777     040  026455  000040  DASH:   .ASCIZ  / -- /
 5288  041004  046040  041123  005015  LSBMSG: .ASCIZ  / LSB/<15><12>
 5289  041012     000
 5290  041013     040  045523  050111  SKPMSG: .ASCIZ  / SKIPPED STATE(S)/<15><12>
 5291  041020  042520  020104  052123
 5292  041026  052101  024105  024523
 5293  041034  005015     000
 5294  041037     040  025052  051105  ERMSG:  .ASCIZ  / **ERROR**/<15><12>
 5295  041044  047522  025122  006452
 5296  041052  000012
 5297  041054  020040  020040  047440  OKMSG:  .ASCIZ  /     OK/<15><12>
 5298  041062  006513  000012
 5299  041066  047040  051101  047522  NARMSG: .ASCIZ  # NARROW STATE(S)#<15><12>
 5300  041074  020127  052123  052101
 5301  041102  024105  024523  005015
 5302  041110     000
 5303  041111     040  044527  042504  WIDMSG: .ASCIZ  # WIDE STATE(S)#<15><12>
 5304  041116  051440  040524  042524
 5305  041124  051450  006451  000012
 5306  041132  052123  052101  026505  STATE:  .ASCIZ  /STATE-- WIDTH/<15><12>
```

```
 5307  041140  020055  044527  052104
 5308  041146  006510  000012
 5309  041152  046040  041123  046440  LINEA:  .ASCIZ  / LSB MAXIMUS AT /
 5310  041160  054101  046511  051525
 5311  041166  040440  020124     000
 5312  041173     057     000          SLASH:  .ASCIZ  #/#
 5313  041175     122  046105  052101  MSG21:  .ASCIZ  /RELATIVE ACCURACY:/<15><12>
 5314  041202  053111  020105  041501
 5315  041210  052503  040522  054503
 5316  041216  006472  000012
 5317  041222  005015  047522  020115  COMP:   .ASCIZ  <15><12>/ROM BLASTING COMPLETED/
 5318  041230  046102  051501  044524
 5319  041236  043516  041440  046517
 5320  041244  046120  052105  042105
 5321  041252     000
 5322  041253     015  052012  046511  TIMEO:  .ASCII  <15><12>/TIMEOUT FROM BLASTER   CHECK SWITCHES ON BLASTER/
 5323  041260  047505  052125  043040
 5324  041266  047522  020115  046102
 5325  041274  051501  042524  020122
 5326  041302  020040  044103  041505
 5327  041310  020113  053523  052111
 5328  041316  044103  051505  047440
 5329  041324  020116  046102  051501
 5330  041332  042524     122
 5331  041335     015  020012  051117          .ASCIZ  <15><12>/ OR REPLACE ROM IN BLASTER/<15><12>
 5332  041342  051040  050105  040514
 5333  041350  042503  051040  046517
 5334  041356  044440  020116  046102
 5335  041364  051501  042524  006522
 5336  041372  000012
 5337  041374     015     012          FIELDI: .BYTE   15,12
 5338  041376  047111  042524  047122          .ASCIZ  /INTERNAL MAINT. Z PULSES <Y FOR YES> ? /
 5339  041404  046101  046440  044501
 5340  041412  052116  020056  020132
 5341  041420  052520  051514  051505
 5342  041426  036040  020131  047506
 5343  041434  020122  042531  037123
 5344  041442  037440  000040
 5345  041446  005015  053101  051105  AVRGO:  .ASCIZ  <15><12>/AVERAGE OF 128 STATES = /
 5346  041454  043501  020105  043117
 5347  041462  030440  034062  051440
 5348  041470  040524  042524  020123
 5349  041476  020075     000
 5350  041501     015     012     007  UNFIX:  .BYTE   15,12,7
 5351  041504  054105  042503  042105          .ASCIZ  /EXCEEDED CORRECTION COUNTER FOR /
 5352  041512  042105  041440  051117
 5353  041520  042522  052103  047511
 5354  041526  020116  047503  047125
 5355  041534  042524  020122  047506
 5356  041542  020122     000
 5357  041545     015     012     007  ABORTB: .BYTE   15,12,7
 5358  041550  047522  020115  046102          .ASCII  /ROM BLASTING ABORTED/
 5359  041556  051501  044524  043516
 5360  041564  040440  047502  052122
```

```
 5361  041572  042105
 5362  041574     015      012      007            .BYTE   15,12,7,0
 5363  041577     000
 5364  041600  020040  026455  020040  SGNVAL:  .ASCII  /  --  /
 5365  041606     053      040          SGNVL1:  .BYTE   53,40
 5366  041610     060      060      052  LSBTXT:  .BYTE   60,60,52,60,15,12,0
 5367  041613     060      015      012
 5368  041616     000
 5369  041617     015      012      000  RTN:     .BYTE   15,12,0,0
 5370  041622     000
 5371  041623     040  042502  047514  BELMSG:  .ASCIZ  / BELOW LIMIT/<15><12>
 5372  041630  020127  044514  044515
 5373  041636  006524  000012
 5374  041642  047040  051117  040515  NORMSG:  .ASCIZ  / NORMAL STATES/<15><12>
 5375  041650  020114  052123  052101
 5376  041656  051505  005015     000
 5377  041663     040  041101  053117  ABOMSG:  .ASCIZ  / ABOVE LIMIT/<15><12>
 5378  041670  020105  044514  044515
 5379  041676  006524  000012
 5380  041702  050040  051501  042523  PASMSG:  .ASCIZ  / PASSED/<15><12>
 5381  041710  006504  000012
 5382  041714  043040  044501  042514  FAIMSG:  .ASCIZ  / FAILED/<15><12>
 5383  041722  006504  000012
 5384                                            .EVEN
 5385  041726  000000                  LSBAVG:  0
 5386  041730  000000                  LSBSVQ:  0
 5387  041732  000000                  LSBSVR:  0
 5388  041734  000000                  LSBSVW:  0
 5389  041736  000000                  DIFEX1:  0
 5390  041740  042162                  ROMPNT:  ROMVAL
 5391  041742  027340                  NOMIAL:  12000.          ;AVERAGE COUNT
 5392
 5393                                    ;THESE LOC.'S CLEARED BY DIFLIN
 5394  041744  000000                  AVGVAL:  0
 5395  041746  000000                  DIFERR:  0
 5396  041750  000000                  BELOW:   0
 5397  041752  000000                  ABOVE:   0
 5398  041754  000000                  NORMAL:  0
 5399  041756  000000                  NARROW:  0
 5400  041760  000000                  NARA:    0
 5401  041762  000000                  NARB:    0
 5402  041764  000000                  WIDA:    0
 5403  041766  000000                  WIDB:    0
 5404  041770  000000                  DIF:     0
 5405  041772  000000                  OUT:     0
 5406  041774  000000                  FIRST:   0
```

```
                                        ;KEYBOARD CONVERSATION TO THE BLASTER   SELECT TEST "T"
                                        ;APPLY POWER TO BLASTER AND DEPRESS I/O AND EXECUTE BUTTONS
                                        ;ALL INDICATORS SHOULD LIGHT    THE BLASTER SHOULD SEND A ">" CHARACTER
                                        ;TYPE THE COMMANDS FOLLOWED BY A "CR", SOME WILL ECHO THE CR AND OTHERS
                                        ;WILL NOT       TYPE "ESC"/"ALT" KEYS TO ABORT CURRNET COMMAND
                                        ;TYPE   KF2/2           ;THIS SETS OCTAL COMMAND MODE
                                        ;TYPE   FM30            ;THIS SET OCTAL INPUT/OUTPUT
                                        ;TYPE   SP[0037/8]      ;THIS SIZES THE PROM -- 0037/8 IS REPORTED BY THE BLASTER
                                        ;TYPE   LD              ;THIS READS THE PROM INTO THE RAM
                                        ;TYPE   D00/37          ;THIS OUTPUTS THE RAM CONTENTS  TO THE TTY
                                        ;TYPE   ZP              ;THIS EXITS COMPUTER CONTROL TO THE BUTTON MODE

041776  105777  137674         LOOPC: TSTB    @DLICSR          ;BLASTER INPUT ?
042002  100007                        BPL     LOOPD            ;NO
042004  000240                        NOP
042006  000240                        NOP
042010  000240                        NOP
042012  017720  137662                MOV     @DLIBD,(R0)+     ;SAVE THE CHARACTER
042016  005237  042160                INC     BLICNT           ;UPDATE COUNTER

042022  105777  137122         LOOPD: TSTB    @$TPS            ;PRINTER READY ?
042026  100363                        BPL     LOOPC            ;BR IF NOT
042030  005737  042160                TST     BLICNT           ;ANY CHARACTERS
042034  001413                        BEQ     LOOPE            ;BR IF NOT
042036  012177  137110                MOV     (R1)+,@$TPB      ;PRINT THE CHAR.
042042  005337  042160                DEC     BLICNT
042046  001353                        BNE     LOOPC            ;BR IF MORE DATA
042050  012700  060000         BTALK: MOV     #BUF0,R0
042054  010001                        MOV     R0,R1
042056  005037  042160                CLR     BLICNT
042062  000745                        BR      LOOPC
                                ;COME HERE IS NO BLASTER DATA TO BE TYPED
042064  105777  137054         LOOPE: TSTB    @$TKS            ;KEYBOARD INPUT
042070  100342                        BPL     LOOPC            ;BR IF NOT
042072  017702  137050                MOV     @$TKB,R2                 ;READ CHAR
042076  042702  177600                BIC     #177600,R2       ;MASK THE DATA
042102  022702  000003                CMP     #3,R2            ;TEST FOR CTRL C
042106  001413                        BEQ     2$               ;BR IF CTRL C
042110  022702  000177                CMP     #177,R2          ;TEST FOR RUBOUT
042114  001412                        BEQ     LOOPF            ;BR IF ESC
042116  020227  000140                CMP     R2,#140          ;TEST IF LOWER CASE
042122  103402                        BLO     1$               ;BR IF NOT
042124  042702  000040                BIC     #40,R2           ;MAKE UPPER CASE
042130  010277  137550         1$:    MOV     R2,@DLODB        ;LOAD BLASTER DATA OUT
042134  000720                        BR      LOOPC
042136  000137  002644         2$:    JMP     RBEG0            ;JUMP TO RESTART
042142  012777  000033  137534 LOOPF: MOV     #33,@DLODB       ;LOAD ESC <CANCLE>
042150  012777  000134  136774        MOV     #'\,@$TPB
042156  000707                        BR      LOOPC
042160  000000         BLICNT: 0
```

```
5458
5459                                    ;A017 CORRECTION PROM VALUES    <MEMORY RAM>
5460
5461  042162    200     200     200   ROMVAL: .BYTE   200,200,200,200,200,200,200,200 ;GAIN - 1 VALUE
5462  042165    200     200     200
5463  042170    200     200
5464  042172    200     200     200           .BYTE   200,200,200,200,200,200,200,200
5465  042175    200     200     200
5466  042200    200     200
5467  042202    200     200     200           .BYTE   200,200,200,200,200,200,200,200 ;GAIN = 2 VALUE
5468  042205    200     200     200
5469  042210    200     200
5470  042212    200     200     200           .BYTE   200,200,200,200,200,200,200,200
5471  042215    200     200     200
5472  042220    200     200
5473
5474                                    ;A017 CORRECTION PRAM VALUES    <RAM MEMORY>
5475                                    ;ROMVAL MUST BE IN COMPLEMENTED AND REVERSED ORDER <LSB=MSB>
5476
5477  042222    376     376     376   RAMVAL: .BYTE   376,376,376,376,376,376,376,376 ;GAIN = 1 VALUE
5478  042225    376     376     376
5479  042230    376     376
5480  042232    376     376     376           .BYTE   376,376,376,376,376,376,376,376
5481  042235    376     376     376
5482  042240    376     376
5483  042242    376     376     376           .BYTE   376,376,376,376,376,376,376,376 ;GAIN = 2 VALUE
5484  042245    376     376     376
5485  042250    376     376
5486  042252    376     376     376           .BYTE   376,376,376,376,376,376,376,376
5487  042255    376     376     376
5488  042260    376     376
5489
5490                                    ;M8036 PROGRAM PROM VALUES
5491
5492
5493  042262    215     244     116   PROROM: .BYTE   215,244,116,144,215,244,116,144 ;M8036 PROGRAM PROM
5494  042265    144     215     244
5495  042270    116     144
5496  042272    215     244     116           .BYTE   215,244,116,144,215,244,116,144
5497  042275    144     215     244
5498  042300    116     144
5499  042302    116     144     216           .BYTE   116,144,216,244,377,377,377,377
5500  042305    244     377     377
5501  042310    377     377
5502  042312    377     377     377           .BYTE   377,377,377,377,377,377,377,377
5503  042315    377     377     377
5504  042320    377     377
```

```
5505                                        ;BLASTER SECTION
5506  042322  104401  051046          BLAST:  TYPE,   PRIM0           ;TELL OPERATOR ABOUT THE CABLE
5507  042326  104401  051460                  TYPE,   PRIM5           ;AND SWITCHES
5508  042332  042777  000001  137336        BIC     #BIT0,@DLICSR   ;ENSURE INIT. ROM CONTROL LINE
5509  042340  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER TO INITILIZE
5510  042344  052644                        ESCP                    ;REALLY WE ARE WAITING FOR THE OPERATOR
5511  042346  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER THE KEYBOARD INPUT MODE
5512  042352  052646                        KF0
5513  042354  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER THE DATA FORMAT <ASCII-OCTAL>
5514  042360  052655                        FM0
5515                                      ;PRIME THE MEMORY RAM BUFFER
5516  042362  012700  042162                MOV     #ROMVAL,R0      ;GET POINTER
5517  042366  112720  000200          1$:   MOVB    #200,(R0)+      ;LOAD THE BUFFER
5518  042372  022700  042222                CMP     #ROMVAL+40,R0   ;TEST IF DONE
5519  042376  001373                        BNE     1$              ;BR IF BUFFER NOT PRIMED
5520  042400  012737  000036  046146        MOV     #36,PRIME1      ;LOAD GAIN AND RESOL. VALUES
5521  042406  004537  046152                JSR     R5,LIMITS       ;LOAD DIFLIN ERROR LIMIT VALUES
5522  042412  046216                        G1LIM0                  ;G1 LIMIT - USERS
5523  042414  046226                        G1LIM1                  ;G1 LIMIT - OPTION CHECKOUT
5524  042416  112737  000061  040773        MOVB    #'1,GAIN1       ;LOAD GAIN TYPEOUT VALUE
5525  042424  012737  042163  041740        MOV     #ROMVAL+1,ROMPNT        ;LOAD RAM POINTER
5526  042432  004737  042476                JSR     PC,4$           ;TEST THAT GAIN
5527  042436  012737  002036  046146        MOV     #2036,PRIME1    ;LOAD GAIN AND RESOL. VALUES
5528  042444  004537  046152                JSR     R5,LIMITS       ;LOAD DIFLIN ERROR LIMIT VALUES
5529  042450  046236                        G2LIM0                  ;G2 LIMIT - USERS
5530  042452  046246                        G2LIM1                  ;G2 LIMIT - OPTION CHECKOUT
5531  042454  112737  000062  040773        MOVB    #'2,GAIN1       ;LOAD GAIN TYPEOUT VALUE
5532  042462  012737  042203  041740        MOV     #ROMVAL+21,ROMPNT       ;LOAD RAM POINTER
5533  042470  004737  042476                JSR     PC,4$           ;TEST GAIN OF 2
5534  042474  000436                        BR      10$
5535  042476  012737  000003  042646  4$:   MOV     #3,100$         ;LOAD LOOP COUNTER
5536  042504  012737  000001  041736  5$:   MOV     #1,DIFEX1       ;LOAD DIFLIN EXIT FLAG
5537  042512  012737  042530  046150        MOV     #6$,DIFEX0      ;LOAD DIFLIN EXIT RETURN
5538  042520  004737  042650                JSR     PC,ZAPRAM       ;ENSURE BLASTER RAM HAS BEEN LOADED WITH
5539                                                                ;THE "RAMVAL" VALUE
5540  042524  000137  044064                JMP     DIFLN0          ;RUN "DIFLIN"
5541  042530  004737  043422          6$:   JSR     PC,ADJFIX       ;ADJUST AND FIX THE CORRECTION WEIGHTS
5542  042534  005337  042646                DEC     100$            ;HAS THIS SECTION BEEN EXECITED N TIMES
5543  042540  001361                        BNE     5$              ;BR IF NOT
5544  042542  005737  041746                TST     DIFERR          ;TEST IF ERROR OCCURRED?
5545  042546  001410                        BEQ     8$              ;BR IF NOT
5546  042550  104401  041501          7$:   TYPE,   UNFIX           ;TELL THE OPERATOR
5547  042554  104401  040761                TYPE,   GAINX           ;ABOUT IT
5548  042560  104401  041545                TYPE,   ABORTB          ;TELL OPER. THE BLASTING IS ABORTED
5549  042564  000137  002644                JMP     RBEG0           ;RESTART
5550  042570  000207                  8$:   RTS     PC              ;EXIT
```

```
5551                                    ;THE RAM DATA IS NOW CORRECT - BLAST THE BLANK ROM
5552  042572  004537  043112  10$:      JSR     R5,RDBLST       ;TELL THE BLASTER TO COMP. THE DATA
5553  042576  053075                    CMO
5554  042600  004537  043112            JSR     R5,RDBLST       ;TELL THE BLASTER TO CHECK-SUM DATA
5555  042604  053105                    CCO
5556  042606  004537  043112            JSR     R5,RDBLST       ;TELL THE BLASTER TO BLAST THE ROM
5557  042612  053110                    PGO
5558  042614  005237  043274            INC     NOEXIT          ;SET FLAG TO EXIT WITHOUT RECPT OF '>' CHAR FROM BLASTER
5559  042620  004537  043112            JSR     R5,RDBLST       ;TELL THE BLASTER TO RETURN TO KEYPAD MODE
5560  042624  053120                    ZPO
5561                                    ;INFORM THE OPERATOR ABOUT THE BLASTER ROM
5562  042626  104401  052051            TYPE,   PRIM1   ;TELL OPERATOR TO REMOVE RAM CABLE AND INSTERT ROM INTO A017
5563  042632  104411                    RDLIN
5564  042634  012600                    MOV     (SP)+,R0        ;CLEAN STACK
5565  042636  104401  041222            TYPE,   COMP            ;COMPLETED
5566  042642  000137  002644            JMP     RBEGO
5567  042646  000000            100$:   0
5568
5569                                    ;SUBROUTINE TO CONVERT THE MEMORY RAM VALUE INTO RAM MEMORY VALUE
5570
5571  042650  012703  042222  ZAPRAM:   MOV     #RAMVAL,R3      ;LOAD OUTPUT POINTER
5572  042654  012700  042162            MOV     #ROMVAL,R0      ;LOAD INPUT POINTER
5573  042660  112001            1$:     MOVB    (R0)+,R1        ;GET A BYTE
5574  042662  105101                    COMB    R1              ;INVERT VALUE
5575  042664  005002                    CLR     R2              ;CLEAR RESULT
5576  042666  006001                    ROR     R1              ;MOVE INTO CARRY BIT
5577  042670  006102                    ROL     R2              ;MOVE CARRY INTO BIT 0
5578  042672  006001                    ROR     R1              ;MOVE INTO CARRY BIT
5579  042674  006102                    ROL     R2              ;MOVE CARRY INTO BIT 0
5580  042676  006001                    ROR     R1              ;MOVE INTO CARRY BIT
5581  042700  006102                    ROL     R2              ;MOVE CARRY INTO BIT 0
5582  042702  006001                    ROR     R1              ;MOVE INTO CARRY BIT
5583  042704  006102                    ROL     R2              ;MOVE CARRY INTO BIT 0
5584  042706  006001                    ROR     R1              ;MOVE INTO CARRY BIT
5585  042710  006102                    ROL     R2              ;MOVE CARRY INTO BIT 0
5586  042712  006001                    ROR     R1              ;MOVE INTO CARRY BIT
5587  042714  006102                    ROL     R2              ;MOVE CARRY INTO BIT 0
5588  042716  006001                    ROR     R1              ;MOVE INTO CARRY BIT
5589  042720  006102                    ROL     R2              ;MOVE CARRY INTO BIT 0
5590  042722  006001                    ROR     R1              ;MOVE INTO CARRY BIT
5591  042724  006102                    ROL     R2              ;MOVE CARRY INTO BIT 0
5592  042726  110223                    MOVB    R2,(R3)+        ;SAVE OUTPUT BYTE
5593  042730  020027  042222            CMP     R0,#ROMVAL+40   ;TEST IF END
5594  042734  001351                    BNE     1$              ;BR IF NOT
5595                                ;SUBROUTINE TO LOAD MEMORY RAM INTO RAM MEMORY
5596  042736  004537  043276            JSR     R5,CONROM       ;CONVERT 'ROMVAL' TABLE INTO BLASTER FORMAT
5597  042742  042222                    RAMVAL
5598  042744  052777  000001  136724    BIS     #BIT0,@DLICSR   ;ENABLE RAM MEMORY TO BE LOADED
5599  042752  004537  043112            JSR     R5,RDBLST       ;TELL THE BLASTER THE NEW RAM DATA
5600  042756  052663                    DIO
5601  042760  004537  043112            JSR     R5,RDBLST       ;TELL THE BLASTER TO CHECKSUM THE DATA
5602  042764  053105                    CCO
5603  042766  004537  043112            JSR     R5,RDBLST       ;TELL THE BLASTER TO PROGRAM THE RAM-PAK
5604  042772  053110                    PGO
```

```
 5605  042774  042777  000001  136674          BIC     #BIT0,@DLICSR   ;RE-ENTER EMMULATE MODE
 5606  043002  000207                          RTS     PC              ;EXIT
```

```
                                        ;BLASTING THE PROGRAM PROM (M8036)
                                        
043004  104401  051622        PBLAST: TYPE,   PRIM3           ;INFORM THE OPERATOR
043010  042777  000001  136660        BIC     #BIT0,@DLICSR   ;ENSURE ROM MODE
043016  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER TO INITILIZE
043022  052644                        ESCP
043024  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER THE KEYBOARD INPUT MODE
043030  052646                        KF0
043032  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER THE DATA FORMAT <ASCII-OCTAL>
043036  052655                        FM0
043040  004537  043276                JSR     R5,CONROM       ;CONVERT THE 'PROROM'' TABLE INTO BLASTER FORMAT
043044  042262                        PROROM
043046  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER THE PROM DATA
043052  052663                        DI0
043054  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER TO CHECKSUM THE DATA
043060  053105                        CC0
043062  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER TO BLAST THE PROM
043066  053110                        PG0
043070  005237  043274                INC     NOEXIT          ;SET FLAG TO EXIT WITHOUT RECPT OF '>'' FROM BLASTER
043074  004537  043112                JSR     R5,RDBLST       ;TELL THE BLASTER TO RETURN TO KEYPAD MODE
043100  053120                        ZP0
043102  104401  041222                TYPE,   COMP            ;INFORM THE OPERATOR IT'S DONE
043106  000137  002644                JMP     RBEG0
```

```
5630
5631                                    ;COME THERE TO OUTPUT A MESSAGE TO THE BLASTER
5632                                    ;EXIT ONLY WHEN THE BLASTER HAS ECHOED AN '>'
5633                                    ;REPORT AN ERROR UPON TIMEOUT UNLESS 'NOEXIT' IS NON-ZERO
5634  043112  010046                RDBLST: MOV     R0,-(SP)        ;SAVE R0
5635  043114  012500                        MOV     (R5)+,R0        ;GET ARG.
5636  043116  012737  000012  043272        MOV     #10.,13$        ;LOAD DELAY FOR ERROR COUNTER
5637  043124  111037  043264                MOVB    (R0),10$        ;LOAD A CHAR
5638  043130  017737  136544  043266        MOV     @DLIBD,11$      ;FALSE READ TO CLEAR READY
5639  043136  105777  136540          1$:   TSTB    @DLOCSR         ;WAIT FOR OUTPUT READY
5640  043142  100375                        BPL     1$
5641  043144  105737  043264                TSTB    10$             ;TEST IF FINISHED ?
5642  043150  001406                        BEQ     2$
5643  043152  112037  043264                MOVB    (R0)+,10$       ;GET A CHAR.
5644  043156  001403                        BEQ     2$              ;BR IF TERM
5645  043160  113777  043264  136516        MOVB    10$,@DLODB      ;OUTPUT THE CHARACTER
5646  043166  105777  136504          2$:   TSTB    @DLICSR         ;WAIT FOR INPUT
5647  043172  100416                        BMI     4$              ;BR IF INPUT
5648  043174  005337  043270                DEC     12$             ;DELAY
5649  043200  001372                        BNE     2$
5650  043202  005337  043272                DEC     13$             ;DELAY
5651  043206  001367                        BNE     2$
5652  043210  005737  043274                TST     NOEXIT          ;TEST IF EXIT IS ALLOWED WITHOUT '>'
5653  043214  001017                        BNE     5$              ;BR IF YES
5654  043216  104401  041253                TYPE,   TIMEO
5655  043222  000137  002644                JMP     RBEG0
5656  043226  000240                        NOP
5657  043230  017737  136444  043266  4$:   MOV     @DLIBD,11$      ;READ CHAR
5658  043236  042737  177600  043266        BIC     #177600,11$     ;MASK OFF BITS
5659  043244  022737  000076  043266        CMP     #'>,11$         ;TEST FOR >
5660  043252  001331                        BNE     1$              ;TRY NEXT CHAR.
5661  043254  012600                  5$:   MOV     (SP)+,R0        ;RESTORE R0
5662  043256  005037  043274                CLR     NOEXIT
5663  043262  000205                        RTS     R5              ;EXIT
5664
5665  043264  000000                10$:    0
5666  043266  000000                11$:    0
5667  043270  000000                12$:    0
5668  043272  000000                13$:    0
5669  043274  000000                NOEXIT: 0
```

```
5670
5671                                    ;CONVERT ROM DATA INTO PROM BLASTER FORMAT
5672  043276  012700  052676            CONROM: MOV     #DIDATA+3,R0    ;LOAD RESULT POINTER
5673  043302  012501                            MOV     (R5)+,R1        ;LOAD INITIAL VALUE POINTER
5674  043304  010137  043360                    MOV     R1,10$          ;COPY THE START
5675  043310  062737  000040  043360            ADD     #40,10$         ;MAKE THE END ADDRESS
5676  043316  111102                    1$:     MOVB    (R1),R2         ;GET A BYTE OF DATA
5677  043320  004737  043402                    JSR     PC,SHUF0        ;CONVERT IT TO ASCII
5678  043324  111102                            MOVB    (R1),R2         ;GET SAME BYTE AGAIN
5679  043326  004737  043374                    JSR     PC,SHUF3        ;CONVERT 2ND DIGIT
5680  043332  112102                            MOVB    (R1)+,R2        ;GET MSD
5681  043334  004737  043362                    JSR     PC,SHUF6        ;CONVERT IT
5682  043340  062700  000007                    ADD     #7,R0           ;UPDATE RESULT POINTER
5683  043344  000240                            NOP
5684  043346  000240                            NOP
5685  043350  020137  043360                    CMP     R1,10$          ;TEST IF DONE
5686  043354  001360                            BNE     1$
5687  043356  000205                            RTS     R5              ;EXIT
5688  043360  000000                    10$:    0
5689
5690                                    ;ROTATE THE DATA RIGHT AND MAKE ASCII CHARACTER
5691
5692  043362  042702  177477            SHUF6:  BIC     #177477,R2      ;MASK OFF BITS
5693  043366  006202                            ASR     R2
5694  043370  006202                            ASR     R2
5695  043372  006202                            ASR     R2
5696  043374  006202                    SHUF3:  ASR     R2
5697  043376  006202                            ASR     R2
5698  043400  006202                            ASR     R2
5699  043402  042702  177770            SHUF0:  BIC     #177770,R2      ;MASK OFF BITS
5700  043406  062702  000060                    ADD     #60,R2          ;MAKE ASCII
5701  043412  110240                            MOVB    R2,-(R0)        ;LOAD RESULT BYTE
5702  043414  000240                            NOP
5703  043416  000240                            NOP
5704  043420  000207                            RTS     PC              ;EXIT
5705
```

```
5706                                    ;SUBROUTINE TO ADJUST THE LSB VALUES
5707  043422  012700  062036    ADJFIX: MOV     #BUF1+36,R0     ;GET BUFFER POINTER
5708  043426  013702  041740            MOV     ROMPNT,R2       ;GET ROM POINTER
5709  043432  010204                    MOV     R2,R4
5710  043434  062704  000017            ADD     #17,R4          ;MAKE END VALUE
5711  043440  012001            10$:    MOV     (R0)+,R1        ;GET LSB (17) M-1
5712  043442  061001                    ADD     (R0),R1         ;ADD LSB (00) M
5713  043444  163701  041744            SUB     AVGVAL,R1       ;SUB 2 LSB AVG.
5714  043450  163701  041744            SUB     AVGVAL,R1
5715  043454  010137  041734            MOV     R1,LSBSVW       ;SAVE FOR LATER USAGE
5716  043460  000240                    NOP
5717  043462  000240                    NOP
5718  043464  000240                    NOP
5719  043466  000240                    NOP
5720  043470  000240                    NOP
5721  043472  000240                    NOP
5722  043474  000240                    NOP
5723  043476  000240                    NOP
5724  043500  000240                    NOP
5725  043502  000240                    NOP
5726  043504  000240                    NOP
5727  043506  000240                    NOP
5728  043510  000240                    NOP
5729  043512  000240                    NOP
5730  043514  000240                    NOP
5731  043516  000240                    NOP
5732
5733  043520  010146                    MOV     R1,-(SP)        ;SAVE ON STACK
5734  043522  012746  000316            MOV     #316,-(SP)      ;MULTI BY 316 (8)
5735  043526  004737  036504            JSR     PC,$MULT
5736  043532  012637  043706            MOV     (SP)+,100$      ;SAVE LSW
5737  043536  012637  043710            MOV     (SP)+,101$      ;SAVE MSW
5738
5739  043542  013746  043706            MOV     100$,-(SP)      ;
5740  043546  013746  043710            MOV     101$,-(SP)
5741  043552  013746  041744            MOV     AVGVAL,-(SP)
5742  043556  004737  036616            JSR     PC,$DIV
5743  043562  102004                    BVC     1$              ;BR IF NO ERROR
5744  043564  000000                    HALT
5745  043566  000000                    HALT
5746  043570  000240                    NOP
5747  043572  000240                    NOP
5748  043574  012637  041732    1$:     MOV     (SP)+,LSBSVR    ;SAVE INTGR. REMAINDER
5749  043600  012637  041730            MOV     (SP)+,LSBSVQ    ;SAVE QUOTIENT
5750
5751                                    ;NOW ADD THE QUOTIENT TO THE MEMORY RAM VALUE
5752
5753  043604  111203                    MOVB    (R2),R3         ;GET CURRENT MEMORY RAM VALUE
5754  043606  000240                    NOP
5755  043610  042703  177400            BIC     #177400,R3      ;CLEAR OFF UPPER BITS
5756  043614  000240                    NOP
5757  043616  000240                    NOP
5758  043620  000240                    NOP
5759  043622  000240                    NOP
```

```
 5760  043624  163703  041730                  SUB     LSBSVQ,R3       ;ADD THE QUOTIENT
 5761  043630  000240                          NOP
 5762  043632  000240                          NOP
 5763  043634  000240                          NOP
 5764  043636  000240                          NOP
 5765  043640  000240                          NOP
 5766  043642  110322                          MOVB    R3,(R2)+        ;RELOAD MEMORY RAM VALUE
 5767
 5768                          ;NOW UPDATE 'M' VALUE POINTER
 5769
 5770  043644  062700  000036                  ADD     #36,R0          ;UPDATE POINTER
 5771  043650  011001                          MOV     (R0),R1         ;GET VALUE
 5772  043652  000240                          NOP
 5773  043654  000240                          NOP
 5774  043656  000240                          NOP
 5775  043660  000240                          NOP
 5776  043662  063701  041734                  ADD     LSBSVW,R1       ;CORRECT THE VALUE (#10)
 5777  043666  000240                          NOP
 5778  043670  000240                          NOP
 5779  043672  000240                          NOP
 5780  043674  000240                          NOP
 5781  043676  010110                          MOV     R1,(R0)         ;RESTORE FOR NEXT USAGE
 5782  043700  020204                          CMP     R2,R4           ;TEST WHEN FINISHED
 5783  043702  001256                          BNE     10$             ;BR UNTIL DONE
 5784
 5785  043704  000207                          RTS     PC              ;EXIT
 5786
 5787  043706  000000              100$:       0
 5788  043710  000000              101$:       0
```

```
5789
5790                                    ;;DIFFERENTIAL LINEARITY
5791  043712  005037  041736            DIFLIN: CLR     DIFEX1          ;CLEAR EXIT FLAG
5792  043716  112737  000061  040773            MOVB    #'1,GAIN1       ;LOAD GAIN MESSAGE VALUE
5793  043724  105737  001134                    TSTB    $AUTOB          ;TEST IF UNDER MONITOR
5794  043730  001007                            BNE     10$             ;BR IF  YES
5795  043732  104401  040730                    TYPE,   GAIN
5796  043736  005737  043712                    TST     DIFLIN          ;TEST BIT 15
5797  043742  100402                            BMI     10$             ;DONT TELL OPER. ABOUT A017
5798  043744  104401  051046                    TYPE,   PRIM0           ;TELL OPERATOR ABOUT A017
5799  043750  012737  000036  046146  10$:     MOV     #36,PRIME1      ;LOAD RESOLUTION AND GAIN
5800  043756  004537  046152                    JSR     R5,LIMITS       ;LOAD DIFLIN TOLERANCE
5801  043762  046216                            G1LIM0                  ;G1 USER LIMIT
5802  043764  046226                            G1LIM1                  ;G1 OPTION AREA
5803  043766  012737  043776  046150            MOV     #1$,DIFEX0      ;LOAD RETURN ADDRESS
5804  043774  000433                            BR      DIFLN0
5805  043776  112737  000062  040773  1$:      MOVB    #'2,GAIN1       ;LOAD GAIN MESSAGE VALUE
5806  044004  012737  002036  046146            MOV     #2036,PRIME1    ;LOAD RESOLUTION AND GAIN
5807  044012  004537  046152                    JSR     R5,LIMITS       ;LOAD DIFLIN TOLER^NCE
5808  044016  046236                            G2LIM0                  ;G2 USER LIMIT
5809  044020  046246                            G2LIM1                  ;G2 OPTION AREA
5810  044022  012737  044044  046150            MOV     #2$,DIFEX0      ;LOAD RETURN ADDRESS
5811  044030  105737  001134                    TSTB    $AUTOB          ;RUNNING UNDER MONITOR
5812  044034  001002                            BNE     11$             ;BR IF YES
5813  044036  104401  040730                    TYPE,   GAIN
5814  044042  000410                    11$:    BR      DIFLN0
5815  044044  022737  177777  050106  2$:      CMP     #-1,RUNDIF      ;ENTER VIA 'F'' SELECTION ?
5816  044052  001002                            BNE     3$              ;BR IF NOT
5817  044054  000137  030076                    JMP     $EOP            ;YES REPORT END OF PASS
5818  044060  000137  002644            3$:     JMP     RBEG0
5819
5820
5821                                    ;DIFLIN ROUTINE
5822  044064  012700  041744            DIFLN0: MOV     #AVGVAL,R0      ;LOAD CLEARING POINTER
5823  044070  005020                    DIFLNX: CLR     (R0)+
5824  044072  022700  041776                    CMP     #FIRST+2,R0     ;FINISHED ?
5825  044076  001374                            BNE     DIFLNX          ;BR IF NOT DONE
5826  044100  012700  060000                    MOV     #BUF0,R0        ;LOAD BUFFER POINTER
5827  044104  005020                    1$:     CLR     (R0)+           ;CLEAR THE DATA BUFFER
5828  044106  022700  064000                    CMP     #BUF2,R0        ;TEST IF FINISHED
5829  044112  001374                            BNE     1$              ;BR IF NOT
5830                                    ;LOOK FOR THE DATA REGION FOR TO
5831  044114  004737  045774                    JSR     PC,STDAT0               ;FIND THE NON-ZERO TO ZERO TRANS. DATA
5832                                    ;DATA HAS NOW GONE INTO THE GOOD 0 DATA REGION
5833  044120  012777  000000  135616            MOV     #0,@KWPSR               ;INIT THE CLOCK PRESET
5834  044126  012777  000025  135604            MOV     #25,@KWCSR              ;ENABLE THE CLOCK
5835                                    ;NOW TIME ''T1'' LENGTH
5836  044134  004737  045640            2$:     JSR     PC,LISTDT               ;GET CURRENT DATA VALUE
5837  044140  022702  000000                    CMP     #0,R2                   ;TEST FOR EXIT OF 0 DATA REGION
5838  044144  001773                            BEQ     2$                      ;BR UNTILL DONE
5839  044146  105077  135570                    CLRB    @KWCSR1                 ;
5840  044152  152777  000002  135562            BISB    #BIT1,@KWCSR1           ;FIRE ST2
5841  044160  017737  135560  046032            MOV     @KWPSR,DIFT1            ;SAVE COUNTER VALUE
5842                                    ;NOW TIME ''T2'' LENGTH
```

```
5843  044166  004737  045640          3$:     JSR     PC,LISTDT               ;GET CURRENT DATA VALUE
5844  044172  022702  000377                  CMP     #377,R2                 ;TEST FOR ENTRY INTO MAX DATA REGION
5845  044176  001373                          BNE     3$                      ;BR IF NOT
5846  044200  105077  135536                  CLRB    @KWCSR1
5847  044204  152777  000002  135530          BISB    #BIT1,@KWCSR1           ;FIRE ST2
5848  044212  017737  135526  046034          MOV     @KWPSR,DIFT2            ;SAVE COUNTER VALUE
5849                                  ;NOW TIME "T3" LENGTH
5850  044220  004737  045640          4$:     JSR     PC,LISTDT               ;GET CURRENT DATA VALUE
5851  044224  022702  000377                  CMP     #377,R2                 ;TEST FOR EXIT OF MAX DATA REGION
5852  044230  001773                          BEQ     4$                      ;BR IF NOT
5853  044232  105077  135504                  CLRB    @KWCSR1
5854  044236  152777  000002  135476          BISB    #BIT1,@KWCSR1           ;FIRE ST2
5855  044244  017737  135474  046036          MOV     @KWPSR,DIFT3            ;SAVE COUNTER VALUE
5856  044252  005077  135462                  CLR     @KWCSR                  ;STOP CLOCK
5857                                  ;NOW DETERMINE THE AVG TIME (T3+T2)/2 AND SAVE IN "HT2T3"
5858  044256  013700  046034                  MOV     DIFT2,R0                ;GET T2 VALUE
5859  044262  063700  046036                  ADD     DIFT3,R0                ;ADD T3 VALUE
5860  044266  006000                          ROR     R0                      ;/2
5861  044270  010037  046040                  MOV     R0,HT2T3                ;SAVE FOR LATER USE
5862                                  ;NOW RETURN TO LIST MODE AND LOOK FOR THE NON-ZERO TO ZERO EDGE AGAIN
5863  044274  004737  045774          DIFLN1: JSR     PC,STDATO                       ;FIND THE EDGE
5864                                  ;NOW START THE CLOCK USING THE AVG TIME
5865                                  ;AND DO AN MATRIX MODE COLLECTION
5866  044300  013700  046040                  MOV     HT2T3,R0                ;GET AVG CLOCK
5867  044304  005400                          NEG     R0                      ;FIX FOR CLOCK PRESET REG
5868  044306  010077  135432                  MOV     R0,@KWPSR               ;LOAD CLOCK PRESET
5869  044312  012777  000020  135420          MOV     #20,@KWCSR              ;PRIME CLOCK
5870  044320  012777  004000  135430          MOV     #CLRALL,@SFR            ;INIT THE NCV11
5871  044326  012777  060000  135414          MOV     #BUF0,@OFF              ;LOAD OFFSET
5872  044334  005077  135412                  CLR     @WCR                    ;CLEAR Z COUNTER
5873  044340  005077  135410                  CLR     @BAR                    ;
5874  044344  013777  046146  135374          MOV     PRIME1,@CSR             ;SET MODE AND RES.
5875  044352  052777  000022  135376          BIS     #TESTZ!BIT4,@SFR        ;SET TESTZ AND TESTX
5876  044360  052777  000001  135360          BIS     #BIT0,@CSR              ;ENABLE NCV11
5877  044366  052777  000001  135344          BIS     #BIT0,@KWCSR            ;ENABLE CLOCK
5878  044374  105777  135340          1$:     TSTB    @KWCSR                  ;WAIT FOR CLOCK FLAG
5879  044400  100375                          BPL     1$
5880  044402  052777  000400  135346          BIS     #ENDDMA,@SFR            ;STOP DMA XFR
```

```
                                        ;THE MATRIX MODE XFR IF NOW COMPLETED
                                        ;NOW ADD THE CURRENT B TE DATA COLLECTION INTO A TOTAL INCREMENT BUFFER
044410  005037  050100                          CLR     $TEMP1
044414  012700  060000                          MOV     #BUF0,R0
044420  012701  062000                          MOV     #BUF1,R1

044424  111037  050100          2$:     MOVB    (R0),$TEMP1             ;GET THE WORD
044430  105020                          CLRB    (R0)+                   ;CLEAR THE BYTE TABLE ENTRY
044432  063721  050100                  ADD     $TEMP1,(R1)+            ;UPDATE THE LIST
044436  103003                          BCC     3$                      ;BR IF NO OVERFLOW
044440  005741                          TST     -(R1)                   ;POSITON POINTER
044442  012721  177777                  MOV     #-1,(R1)+               ;SET TO 177777
044446  022701  064000          3$:     CMP     #BUF2,R1                ;FINISHED?
044452  001364                          BNE     2$                      ;BR IF NOT
                                        ;GET THE AVERAGE OF THE 128 CENTER STATES
044454  012700  062100                  MOV     #BUF1+64.,R0    ;GET INITIAL POINTER
044460  005001                          CLR     R1
044462  005002                          CLR     R2
044464  062001                  4$:     ADD     (R0)+,R1        ;GET A VALUE
044466  103001                          BCC     5$              ;BR IF NO CARRY OVERFLOW
044470  005202                          INC     R2              ;UPDATE MSW
044472  022700  062300          5$:     CMP     #BUF1+192.,R0   ;FINISHED THE BUFFER
044476  001372                          BNE     4$              ;BR IF NOT DONE AVERAGE
044500  000241                          CLC
044502  006002                          ROR     R2              ;ROTATE MSW
044504  006001                          ROR     R1              ;INTO LSW
044506  000241                          CLC
044510  006002                          ROR     R2              ;ROTATE MSW
044512  006001                          ROR     R1              ;INTO LSW
044514  000241                          CLC
044516  006002                          ROR     R2              ;ROTATE MSW
044520  006001                          ROR     R1              ;INTO LSW
044522  000241                          CLC
044524  006002                          ROR     R2              ;ROTATE MSW
044526  006001                          ROR     R1              ;INTO LSW
044530  000241                          CLC
044532  006002                          ROR     R2              ;ROTATE MSW
044534  006001                          ROR     R1              ;INTO LSW
044536  000241                          CLC
044540  006002                          ROR     R2              ;ROTATE MSW
044542  006001                          ROR     R1              ;INTO LSW
044544  010137  041744                  MOV     R1,AVGVAL       ;SAVE THE AVERAGE
044550  004737  046042                  JSR     PC,CTRLCG       ;TEST FOR ^C OR ^G
044554  023737  041744  041742          CMP     AVGVAL,NOMIAL   ;TEST IF AVG. IF >
044562  103644                          BLO     DIFLN1          ;BR IF NOT
```

```
5927                                    ;READ THE TOTAL INCREMENT BUFFER AND DETERMINE IF ANY VALUES OUT OF RANGE
5928
5929  044564  013737  041744  045260  READ1: MOV     AVGVAL,101$     ;GET AVERAGE
5930  044572  012700  000001          MOV     #1,R0
5931  044576  012701  062002          MOV     #BUF1+2,R1
5932  044602  012137  045256          1$:     MOV     (R1)+,100$      ;GET A WORD
5933  044606  163737  045260  045256          SUB     101$,100$       ;REMOVE THE AVERAGE
5934  044614  100006                  BPL     2$
5935  044616  005137  045256          COM     100$
5936  044622  112737  000055  041606          MOVB    #'-,SGNVL1      ;INSERT ''-'' SIGN
5937  044630  000403                  BR      3$
5938  044632  112737  000053  041606  2$:     MOVB    #'+,SGNVL1      ;INSERT ''+'' SIGN
5939  044640  013746  045256          3$:     MOV     100$,-(SP)
5940  044644  012746  001750          MOV     #1000.,-(SP)
5941  044650  004737  036504          JSR     PC,$MULT        ;MULTIPLY NUM-AVG BY 1000.
5942  044654  012637  045274          MOV     (SP)+,107$      ;GET RESULT
5943  044660  012637  045276          MOV     (SP)+,110$
5944  044664  013746  045274          MOV     107$,-(SP)      ;DIVIDE RESULT
5945  044670  013746  045276          MOV     110$,-(SP)
5946  044674  013746  045260          MOV     101$,-(SP)      ;DIVIDE AGAIN
5947  044700  004737  036616          JSR     PC,$DIV
5948  044704  102003                  BVC     4$
5949  044706  000000                  HALT    ;FATAL ERROR DURING CAL. OF ERROR TOLERANCES
5950  044710  000240                  NOP
5951  044712  000240                  NOP
5952  044714  012637  045266          4$:     MOV     (SP)+,104$      ;GET REMAINDER
5953  044720  012637  045270          MOV     (SP)+,105$      ;GET QOUT
5954  044724  013702  045260          MOV     101$,R2         ;ROUND UP IF NEEDED
5955  044730  006202                  ASR     R2
5956  044732  023702  045266          CMP     104$,R2         ;COMPARE RESULT
5957  044736  002402                  BLT     5$
5958  044740  005237  045270          INC     105$            ;ROUND UP
5959  044744  000240                  5$:     NOP
5960                                  ;DIFLIN ERROR CHECK
5961
5962  044746  123737  045270  046207  6$:     CMPB    105$,NORTOL     ;TEST AGAINST NORMAL
5963  044754  103446                  BLO     50$             ;BR IF OK
5964  044756  123737  045270  046211          CMPB    105$,NARTOL     ;TEST AGAINST WIDE/NARROW TOLERANCE
5965  044764  103416                  BLO     20$             ;BR IF OK
5966  044766  123737  045270  046213          CMPB    105$,NURTOL     ;TEST AGAINST LARGE/SMALL TOLERANCE
5967  044774  103424                  BLO     21$             ;BR IF OK
5968  044776  122737  000053  041606          CMPB    #'+,SGNVL1      ;TEST IF +
5969  045004  001403                  BEQ     7$
5970  045006  005237  041750          INC     BELOW           ;COUNT THE LOWER VALUE
5971  045012  000435                  BR      51$
5972  045014  005237  041752          7$:     INC     ABOVE           ;COUNT THE HIGHER VALUE
5973  045020  000432                  BR      51$
5974  045022  122737  000053  041606  20$:    CMPB    #'+,SGNVL1      ;TEST IF +
5975  045030  001403                  BEQ     22$
5976  045032  005237  041762          INC     NARB            ;COUNT THE LOWER
5977  045036  000423                  BR      51$
5978  045040  005237  041766          22$:    INC     WIDB            ;COUNT THE HIGHER
5979  045044  000420                  BR      51$
5980  045046  122737  000053  041606  21$:    CMPB    #'+,SGNVL1      ;TEST IF +
```

```
 5981  045054  001403                          BEQ     23$
 5982  045056  005237  041760                  INC     NARA            ;COUNT THE LOWER
 5983  045062  000411                          BR      51$
 5984  045064  005237  041764          23$:    INC     WIDA            ;COUNT THE HIGHER
 5985  045070  000406                          BR      51$
 5986  045072  005237  041754          50$:    INC     NORMAL          ;COUNT THE NORMAL
 5987  045076  122737  000115  052474          CMPB    #'M,RUNIT       ;TEST IF FORCE REPORT
 5988  045104  001054                          BNE     77$
 5989  045106  105737  001134          51$:    TSTB    $AUTOB          ;TEST IF MONITOR ?
 5990  045112  001051                          BNE     77$             ;BR IF YES
 5991                                  ;REPORT THE STATE INFORMATION
 5992  045114  005737  041774                  TST     FIRST           ;FIRST TYPEOUT ?
 5993  045120  001013                          BNE     10$             ;BR IF YES
 5994  045122  005237  041774                  INC     FIRST           ;SET FLAG
 5995  045126  104401  041446                  TYPE    ,AVRGO          ;TELL THE OPERATOR THE AVERG.
 5996  045132  013746  041744                  MOV     AVGVAL,-(SP)
 5997  045136  104405                          TYPDS
 5998  045140  104401  041617                  TYPE,   RTN
 5999  045144  104401  041132                  TYPE,   STATE           ;ADD HEADER INFO
 6000  045150  010046                  10$:    MOV     R0,-(SP)        ;LOAD STATE #
 6001  045152  104403                          TYPOS
 6002  045154     003     001                  .BYTE   3,1
 6003  045156  122737  000115  052474          CMPB    #'M,RUNIT       ;TEST IF EXPAND OUTPUT SELECTED
 6004  045164  001005                          BNE     11$
 6005  045166  104401  040777                  TYPE,   DASH            ;INSERT SPACING
 6006  045172  013746  045256                  MOV     100$,-(SP)      ;REPORT DIFFERENCE
 6007  045176  104405                          TYPDS
 6008  045200  013746  045270          11$:    MOV     105$,-(SP)
 6009  045204  004737  037232                  JSR     PC,$SB2D        ;CONVERT TO ASCII
 6010  045210  012602                          MOV     (SP)+,R2        ;GET RESULT POINTER TO MESSAGE
 6011  045212  062702  000002                  ADD     #2,R2           ;ADJUST POINTER
 6012  045216  112237  041610                  MOVB    (R2)+,LSBTXT
 6013  045222  112237  041611                  MOVB    (R2)+,LSBTXT+1
 6014  045226  112237  041613                  MOVB    (R2)+,LSBTXT+3
 6015  045232  104401  041600                  TYPE,   SGNVAL
 6016
 6017  045236  004737  046042          77$:    JSR     PC,CTRLCG       ;TEST IF CTRL C/G
 6018  045242  005200                          INC     R0
 6019  045244  022700  000377                  CMP     #255.,R0        ;TEST IF DONE
 6020  045250  001414                          BEQ     READ            ;BR IF DONE
 6021  045252  000137  044602                  JMP     1$              ;TRY NEXT VALUE
 6022
 6023  045256  000000                  100$:   0
 6024  045260  000000                  101$:   0
 6025  045262  000000                  102$:   0
 6026  045264  000000                  103$:   0
 6027  045266  000000                  104$:   0
 6028  045270  000000                  105$:   0
 6029  045272  000000                  106$:   0
 6030  045274  000000                  107$:   0
 6031  045276  000000                  110$:   0
 6032  045300  000000                  111$:   0
```

```
6033
6034                                        ;NOW ACCOUNT FOR ALL THE ERRORS
6035  045302  005037  041746          READ:  CLR     DIFERR          ;CLEAR FLAG
6036  045306  013702  041752                 MOV     ABOVE,R2        ;GET HIGH VALUE
6037  045312  063702  041750                 ADD     BELOW,R2        ;ADD LOW
6038  045316  120237  046214                 CMPB    R2,OBSCNT       ;TEST IF EXCEEDS LIMIT <OBEASE ABOVE/BELOW>
6039  045322  101026                         BHI     1$              ;BR IF ERRORS
6040  045324  013702  041760                 MOV     NARA,R2         ;GET LOW VALUE
6041  045330  063702  041764                 ADD     WIDA,R2         ;ADD HIGH
6042  045334  120237  046212                 CMPB    R2,NURCNT       ;TEST IF EXCEEDS LIMIT <LARGE ABOVE/BELOW>
6043  045340  101017                         BHI     1$
6044  045342  013702  041762                 MOV     NARB,R2         ;GET LOW VALUE
6045  045346  063702  041766                 ADD     WIDB,R2         ;ADD HIGH
6046  045352  120237  046210                 CMPB    R2,NARCNT       ;TEST IF EXCEEDS LIMIT <SMALL ABOVE/BELOW>
6047  045356  101010                         BHI     1$
6048  045360  123737  046206  041754        CMPB    NORCNT,NORMAL   ;TEST IF MIN. NORMAL COUNT HAS BEEN HIT
6049  045366  101004                         BHI     1$
6050  045370  012737  041702  045516        MOV     #PASMSG,READ7   ;LOAD MESSAGE POINTER
6051  045376  000405                         BR      2$
6052  045400  012737  041714  045516  1$:    MOV     #FAIMSG,READ7   ;LOAD MESSAGE POINTER
6053  045406  005237  041746                 INC     DIFERR          ;SET ERROR FLAG
6054  045412  013746  041750          2$:    MOV     BELOW,-(SP)     ;GET NO. OF STATES BELOW LIMITS
6055  045416  104405                         TYPDS                   ;REPORT VALUE
6056  045420  104401  041623                 TYPE    ,BELMSG         ;TYPE MESSAGE
6057  045424  013702  041760                 MOV     NARA,R2         ;GET NARROW LOW
6058  045430  063702  041762                 ADD     NARB,R2         ;ADD NARROW HIGH
6059  045434  010246                         MOV     R2,-(SP)        ;GET NO. OF NARROW STATES
6060  045436  104405                         TYPDS
6061  045440  104401  041066                 TYPE    ,NARMSG         ;TYPE MESSAGE
6062  045444  013746  041754                 MOV     NORMAL,-(SP)    ;GET NORMAL COUNT
6063  045450  104405                         TYPDS
6064  045452  104401  041642                 TYPE,   NORMSG          ;REPORT NORMAL TEXT
6065  045456  013702  041764                 MOV     WIDA,R2         ;GET WIDE LOW
6066  045462  063702  041766                 ADD     WIDB,R2         ;ADD WIDE HIGH
6067  045466  010246                         MOV     R2,-(SP)
6068  045470  104405                         TYPDS
6069  045472  104401  041111                 TYPE    ,WIDMSG         ;TYPE MESSAGE
6070  045476  013746  041752                 MOV     ABOVE,-(SP)
6071  045502  104405                         TYPDS                   ;TYPE NO. OF STATES ABOVE LIMIT
6072  045504  104401  041663                 TYPE    ,ABOMSG         ;TYPE MESSAGE
6073
6074                                        ;REPORT PASS/FAIL MESSAGE
6075
6076  045510  104401  040730                 TYPE,   GAIN            ;RE-TYPE THE GAIN MESSAGE
6077  045514  104401                         TYPE
6078  045516  041702                 READ7:  PASMSG                  ;REPORT PASS/FAIL
6079
```

```
6080
6081                                        ;TYPE OUT MEMORY RAM AND RAM MEMORY VALUES IF SELECTED
6082  045520  032777  004000  133412  READXX: BIT     #SW11,@SWR      ;TEST IF THIS  IS WANTED
6083  045526  001440                          BEQ     4$              ;BR IF NOT
6084  045530  005737  041736                  TST     DIFEX1          ;TEST IF BLASTING MODE
6085  045534  001435                          BEQ     4$              ;BR IF NOT
6086
6087  045536  012700  042162                  MOV     #ROMVAL,R0      ;LOAD INITIAL POINTER
6088  045542  004737  045572          1$:     JSR     PC,2$           ;TYPE OUT _INE
6089  045546  022700  042262                  CMP     #ROMVAL+100,R0  ;TEST IF ALL DONE
6090  045552  001373                          BNE     1$              ;BR IF NOT
6091  045554  104401  041617                  TYPE,   RTN
6092  045560  104401  041617                  TYPE,   RTN
6093  045564  104401  041617                  TYPE,   RTN
6094  045570  000417                          BR      4$
6095
6096  045572  012701  000020          2$:     MOV     #16.,R1         ;LOAD WIDE COUNTER
6097  045576  104401  041617                  TYPE,   RTN
6098  045602  112002                  3$:     MOVB    (R0)+,R2        ;GET A BYTE
6099  045604  042702  177400                  BIC     #177400,R2      ;MASK OFF HIGHER BITS
6100  045610  010246                          MOV     R2,-(SP)        ;LOAD VALUE
6101  045612  104403                          TYPOS
6102  045614     003     001                  .BYTE   3,1
6103  045616  104401  041002                  TYPE,   DASH+3
6104  045622  005301                          DEC     R1              ;FINISHED ?
6105  045624  001366                          BNE     3$
6106  045626  000207                          RTS     PC              ;EXIT
6107
6108
6109  045630  005037  041736          4$:     CLR     DIFEX1
6110  045634  000177  000310                  JMP     @DIFEX0         ;EXIT THIS CRAZYNESS
6111
```

```
                                        ;DIF LIN SUBROUTINE LIST MODE DATA COLLECTOR
                                        
045640  012777  004000  134110  LISTDT: MOV     #CLRALL,@SFR    ;CLEAR THE DEVICE
045646  005077  134076                  CLR     @OFF            ;PRIME OFFSET REG.
045652  012777  177770  134072          MOV     #-8.,@WCR       ;LOAD WC.
045660  012777  060000  134066          MOV     #BUF0,@BAR      ;LOAD BUS ADDR.
045666  012777  000000  134052          MOV     #0,@CSR         ;SET RES. AND MODE
045674  032737  002000  046146          BIT     #2000,PRIME1    ;TEST GAIN BIT
045702  001403                          BEQ     3$              ;BR IF CLEARED
045704  052777  002000  134034          BIS     #2000,@CSR      ;SELECT GAIN = 2
045712  052777  000002  134036  3$:     BIS     #TESTZ,@SFR     ;SET Z PULSES
045720  052777  000001  134020          BIS     #BIT0,@CSR      ;ENABLE THE DEVICE
                                        
045726  105777  134014          1$:     TSTB    @CSR            ;WAIT FOR IDLE
045732  100375                          BPL     1$
045734  012700  060000                  MOV     #BUF0,R0        ;LOAD POINTER TO NEW DATA
045740  005002                          CLR     R2
045742  012703  000010                  MOV     #8.,R3          ;LOAD COUNTER
045746  011001                  2$:     MOV     (R0),R1         ;GET DATA WORD
045750  005020                          CLR     (R0)+           ;CLR BUFFER WORD
045752  042701  177400                  BIC     #177400,R1      ;MAKE OFF BITS
045756  060102                          ADD     R1,R2           ;UPDATE TOTAL
045760  005303                          DEC     R3              ;FINISHED ?
045762  001371                          BNE     2$              ;BR IF NOT
045764  006202                          ASR     R2
045766  006202                          ASR     R2
045770  006202                          ASR     R2
045772  000207                          RTS     PC              ;EXIT
                                        ;SUBROUTINE TO FIND THE TRUE EDGE OF ZERO DATA
045774  012737  177777  046030  STDAT0: MOV     #-1,10$         ;SET ENTRY FLAG
046002  004737  045640          1$:     JSR     PC,LISTDT       ;GET DATA
046006  005702                          TST     R2              ;CHECK DATA
046010  001403                          BEQ     2$              ;BR IF ZERO
046012  005037  046030                  CLR     10$             ;CLEAR ALREADY NON-ZERO DATA FLAG
046016  000771                          BR      1$
                                        ;DATA WAS ZERO - CHECK IF WE STARTED IN 0 DATA REGION
046020  005737  046030          2$:     TST     10$             ;TEST FLAG
046024  001366                          BNE     1$              ;BR IF NOT A GOOD TIME TO EXIT
                                        ;DATA HAS NOW GONE TO A GOOD ZERO DATA REGION
                                        ;NOW EXIT
046026  000207                          RTS     PC              ;BYEBYE
046030  000000                  10$:    0
046032  000000                  DIFT1:  0
046034  000000                  DIFT2:  0
046036  000000                  DIFT3:  0
046040  000000                  HT2T3:  0
```

```
                                        ;SUBROUTINE TO TEST IF OPERATOR TYPED CTRL C OR G
046042  105777  133076          CTRLCG: TSTB    @$TKS           ;INPUT FLAG
046046  100033                  BPL     2$              ;BR IF NON
046050  017737  133072  046140  MOV     @$TKB,CTRCHA    ;GET CHARACTER
046056  042737  177600  046140  BIC     #177600,CTRCHA  ;MASK OFF BITS
046064  022737  000003  046140  CMP     #3,CTRCHA       ;TEST IF CTRL C
046072  001014                  BNE     1$              ;BR IF NOT ^C
046074  052777  000400  133654  BIS     #ENDDMA,@SFR    ;STOP NPR'S
046102  000240                  NOP
046104  000240                  NOP
046106  000240                  NOP
046110  052777  004000  133640  BIS     #CLRALL,@SFR    ;CLEAR THE DEVICE
046116  005726                  TST     (SP)+           ;CLEAN STACK
046120  000137  002012          JMP     RTRT            ;RE-START
046124  022737  000007  046140  1$:     CMP     #7,CTRCHA       ;TEST IF CTRL G
046132  001001                  BNE     2$              ;BR IF NOT
046134  104406                  GTSWR                   ;GET SWR VALUE
046136  000207          2$:     RTS     PC              ;EXIT
046140  000000          CTRCHA: 0
046142  177777          PRIME:  -1.
046144  177400          PRIME0: 177400
046146  000036          PRIME1: 36
046150  000000          DIFEX0: 0

                                        ;SUBROUTINE TO DETERMINE ERROR LIMITS FOR DIFLIN
046152  012500          LIMITS: MOV     (R5)+,R0        ;GET 1ST ARG.
046154  005737  050110          TST     WFMODE          ;TEST IF ON TESTER
046160  001402                  BEQ     1$              ;BR IF NOT
046162  012500                  MOV     (R5)+,R0        ;GET TESTER LIMITS
046164  000401                  BR      2$
046166  005725          1$:     TST     (R5)+           ;BUMP ADDRESS
046170  012701  046206  2$:     MOV     #NORCNT,R1      ;GET POINTER
046174  012021                  MOV     (R0)+,(R1)+     ;GET VALUE
046176  012021                  MOV     (R0)+,(R1)+
046200  012021                  MOV     (R0)+,(R1)+
046202  012021                  MOV     (R0)+ (R1)+
046204  000205                  RTS     R5              ;EXIT
```

```
                                        ;ACTUAL VALUES FOR DIF LIN ERROR TOLERANCE

046206  322             NORCNT: .BYTE   210.            ;'NORMAL'' MIN COUNT VALUE
046207  013             NORTOL: .BYTE   11.             ;'NORMAL'' TOLERANCE VALUE
046210  036             NARCNT: .BYTE   30.             ;'NARROW/WIDE'' MAX. COUNT VALUE
046211  032             NARTOL: .BYTE   26.             ;'NARROW/WIDE'' TOLERANCE VALUE
046212  016             NURCNT: .BYTE   14.             ;'LARGE/SMALL'' MAX. COUNT VALUE
046213  145             NURTOL: .BYTE   101.            ;'LARGE/SMALL'' TOLERANCE VALUE
046214  000             OBSCNT: .BYTE   0.              ;'OBESITY'' MAX. COUNT VALUE
046215  377             OBSTOL: .BYTE   377             ;'OBESITY'' TOLERANCE VALUE

                        ;DIFLIN ERROR TOLERANCE         GAIN = 1        USER

046216  342     025     G1LIM0: .BYTE   226.,21.        ;226 MIN. COUNT, -<2.0%
046220  024     063             .BYTE   20.,51.         ;20 MAX. COUNT, =<5.0%
046222  010     145             .BYTE   8.,101.         ;8 MAX. COUNT, =<10.%
046224  000     377             .BYTE   0.,377          ;0 MAX COUNT, >10%

                        ;DIFLIN ERROR TOLERANCE         GAIN = 1        OPTION AREA

046226  342     024     G1LIM1: .BYTE   226.,20.        ;226 MIN COUNT, =<1.9%
046230  024     061             .BYTE   20.,49.         ;20 MAX. COUNT,  <4.8%
046232  010     140             .BYTE   8.,96.          ;8 MAX COUNT, =<9.5%
046234  000     377             .BYTE   0,377           ;0 MAX COUNT, >9.5%

                        ;DIFLIN ERROR TOLERANCE         GAIN = 2        USER

046236  330     031     G2LIM0: .BYTE   216.,25.        ;216. MIN COUNT, =<2.4%
046240  036     073             .BYTE   30.,59.         ;30. MAX COUNT, =<6.0%
046242  010     171             .BYTE   8.,121.         ;14. MAX COUNT, =<12.0%
046244  000     377             .BYTE   0.,377          ;0 MAX COUNT, >12.0%

                        ;DIFLIN ERROR TOLERANCE         GAIN = 2        OPTION AREA

046246  330     025     G2LIM1: .BYTE   216.,21.        ;216. MIN COUNT, =<2.0%
046250  036     063             .BYTE   30.,51.         ;30. MAX COUNT, =<5.0%
046252  010     145             .BYTE   8.,101.         ;8. MAX COUNT, =<10.0%
046254  000     377             .BYTE   0.,377          ;0 MAX COUNT, >10.0%
```

```
 6235  046256     015     012          LISTO:  .BYTE   15,12
 6236  046260  020111  020075  047111          .ASCII  /I - INITIAL A017 ADJUSTMENT USING THE ANALOG TESTER/
 6237  046266  052111  040511  020114
 6238  046274  030101  033461  040440
 6239  046302  045104  051525  046524
 6240  046310  047105  020124  051525
 6241  046316  047111  020107  044124
 6242  046324  020105  047101  046101
 6243  046332  043517  052040  051505
 6244  046340  042524     122
 6245  046343     015     012                  .BYTE   15,12
 6246  046345     102  036440  041040          .ASCII  /B   BLASTING OF THE LINEARITY CORRECTION PROM/
 6247  046352  040514  052123  047111
 6248  046360  020107  043117  052040
 6249  046366  042510  046040  047111
 6250  046374  040505  044522  054524
 6251  046402  041440  051117  042522
 6252  046410  052103  047511  020116
 6253  046416  051120  046517
 6254  046422     015     012                  .BYTE   15,12
 6255  046424  020103  020075  047503          .ASCIZ  /C - CONTROL PROGRAM PROM BLASTING/
 6256  046432  052116  047522  020114
 6257  046440  051120  043517  040522
 6258  046446  020115  051120  046517
 6259  046454  041040  040514  052123
 6260  046462  047111  000107
 6261  046466     015     012          LIST:   .BYTE   15,12
 6262  046470  020114  020075  047514          .ASCII  /L - LOGIC TEST OF M8026 AND M8036/
 6263  046476  044507  020103  042524
 6264  046504  052123  047440  020106
 6265  046512  034115  031060  020066
 6266  046520  047101  020104  034115
 6267  046526  031460     066
 6268  046531     015     012                  .BYTE   15,12
 6269  046533     104  036440  042040          .ASCII  /D = DIFFERENTIAL LINEARITY OF THE A017/
 6270  046540  043111  042506  042522
 6271  046546  052116  040511  020114
 6272  046554  044514  042516  051101
 6273  046562  052111  020131  043117
 6274  046570  052040  042510  040440
 6275  046576  030460     067
 6276  046601     015     012                  .BYTE   15,12
 6277  046603     106  036440  043040          .ASCII  /F = FINAL ACCEPTANCE (L AND D)/
 6278  046610  047111  046101  040440
 6279  046616  041503  050105  040524
 6280  046624  041516  020105  046050
 6281  046632  040440  042116  042040
 6282  046640     051
 6283  046641     015     012                  .BYTE   15,12
 6284  046643     101  036440  040440          .ASCII  /A - ADJUSTMENT OF A017 AT FIELD SITE/
 6285  046650  045104  051525  046524
 6286  046656  047105  020124  043117
 6287  046664  040440  030460  020067
 6288  046672  052101  043040  042511
```

```
 6289  046700  042114  051440  052111
 6290  046706     105
 6291  046707     015     012                     .BYTE   15,12
 6292  046711     117  036440  047440             .ASCII  /O = OPERATE WITH TTY AT ADDRESS NNNNNN/
 6293  046716  042520  040522  042524
 6294  046724  053440  052111  020110
 6295  046732  052124  020131  052101
 6296  046740  040440  042104  042522
 6297  046746  051523  047040  047116
 6298  046754  047116     116
 6299  046757     015     012                     .BYTE   15,12
 6300  046761     123  036440  051440             .ASCII  /S = SOFTWARE SWITCH REGISTER CHANGE/
 6301  046766  043117  053524  051101
 6302  046774  020105  053523  052111
 6303  047002  044103  051040  043505
 6304  047010  051511  042524  020122
 6305  047016  044103  047101  042507
 6306  047024     015     012                     .BYTE   15,12
 6307  047026  020110  020075  042510             .ASCIZ  /H = HELP THE OPERATOR AND RETYPE THIS LIST/
 6308  047034  050114  052040  042510
 6309  047042  047440  042520  040522
 6310  047050  047524  020122  047101
 6311  047056  020104  042522  054524
 6312  047064  042520  052040  044510
 6313  047072  020123  044514  052123
 6314  047100     000
 6315  047101     015     012          LIST1:  .BYTE   15,12
 6316  047103     056     000          DOT:    .ASCIZ  /./
 6317
 6318          047106                          .EVEN
 6319
```

```
6320
6321  047106  000005                  POTIME: RESET                   ;CLEAR THE WORLD
6322  047110  104401  052242                  TYPE,   PRIM2           ;TELL OPERATOR ABOUT CABLES
6323  047114  012777  006035  132564          MOV     #6035,@VTCHAR   ;HOME AND ERASE SCREEN
6324  047122  005077  132562                  CLR     @VTYPOS         ;CLEAR THE MAP P.C.
6325  047126  005037  046140                  CLR     CTRCHA          ;CLEAR TTY CHARACTER
6326
6327                                  ;ERASE THE SCREEN MAP
6328  047132  105777  132560          1$:     TSTB    @VTCSR          ;WAIT FOR READY
6329  047136  100375                          BPL     1$              ;
6330  047140  052777  001000  132550          BIS     #BIT9,@VTCSR    ;CLEAR THE MAP
6331  047146  105777  132544          2$:     TSTB    @VTCSR          ;WAIT FOR READY AGAIN
6332  047152  100375                          BPL     2$
6333  047154  042777  001000  132534          BIC     #BIT9,@VTCSR    ;CLEAR THE ERASE BIT
6334                                  ;NOW LOAD THE REF. PATTERN INTO THE MAP
6335  047162  012701  010421                  MOV     #10421,R1       ;LOAD THE PIXEL VALUE
6336  047166  012700  003740                  MOV     #3740,R0        ;LOAD THE PIXEL ADDRESS
6337  047172  004737  050044          3$:     JSR     PC,DISPLY       ;LOAD THE DATA
6338  047176  005200                          INC     R0              ;UPDATE THE ADDRESS
6339  047200  022700  004000                  CMP     #4000,R0        ;FINISHED ?
6340  047204  001372                          BNE     3$              ;BR UNTIL DONE
6341
6342  047206  012700  000000                  MOV     #0,R0           ;LOAD PIXEL ADDRESS
6343  047212  004737  050044          4$:     JSR     PC,DISPLY       ;LOAD THE DATA
6344  047216  005200                          INC     R0              ;UPDATE THE PIXEL ADDRESS
6345  047220  022700  000040                  CMP     #40,R0          ;TEST IF DONE
6346  047224  001372                          BNE     4$              ;BR IF DONE THE VERTICAL
6347
6348  047226  012700  007740                  MOV     #7740,R0        ;LOAD PIXEL ADDRESS
6349  047232  004737  050044          5$:     JSR     PC,DISPLY       ;LOAD THE DATA
6350  047236  005200                          INC     R0              ;UPDATE THE PIXEL ADDRESS
6351  047240  022700  010000                  CMP     #10000,R0       ;TEST IF DONE
6352  047244  001372                          BNE     5$              ;BR IF DONE
6353
6354                                  ;POSITION THE MAP AND DISPLAY
6355
6356  047246  012777  000005  132442          MOV     #5,@VTCSR       ;LOAD ORGIN VALUE
6357  047254  012777  000060  132442          MOV     #60,@VTINT      ;LOAD LUT 0
6358  047262  012777  000463  132434          MOV     #.+63,@VTINT    ;LOAD LUT 1
6359  047270  052777  000400  132420          BIS     #BIT8,@VTCSR    ;ENABLE THE MAP
```

```
6360                                    ;NOW SAMPLE THE CAMERA CHANNELS
6361  047276                            SAMCAM:
6362                                    ;CAMERA 0, X INPUT
6363  047276  004537  047422                  JSR     R5,ADJCAM       ;
6364  047302  050126                          ADMSG0                  ;CAM 0 MESSAGE
6365  047304  001730                          DAC0                    ;DAC ADDRESS
6366  047306     000     000                  .BYTE   0,0             ;CAMERA 0, X AXIS
6367                                    ;DO CAMERA 1, X INPUT
6368  047310  004537  047422                  JSR     R5,ADJCAM
6369  047314  050312                          ADMSG2                  ;CAM 1, X MESSAGE
6370  047316  001732                          DAC1                    ;DAC ADDRESS
6371  047320     001     000                  .BYTE   1,0             ;CAMERA 1, X AXIS
6372                                    ;DO CAMERA 2, X INPUT
6373  047322  004537  047422                  JSR     R5,ADJCAM
6374  047326  050476                          ADMSG4                  ;CAM 2, X MESSAGE
6375  047330  001734                          DAC2                    ;DAC ADDRESS
6376  047332     002     000                  .BYTE   2,0             ;CAMERA 2, X AXIS
6377                                    ;DO CAMERA 3, X INPUT
6378  047334  004537  047422                  JSR     R5,ADJCAM
6379  047340  050662                          ADMSG6                  ;CAM 3, X MESSAGE
6380  047342  001736                          DAC3                    ;DAC ADDRESS
6381  047344     003     000                  .BYTE   3,0             ;CAMERA 3, X AXIS
6382                                    ;NOW SAMPLE THE Y CAMERA CHANNELS
6383                                    ;DO CAMERA 0 Y INPUT
6384  047346  004537  047422                  JSR     R5,ADJCAM
6385  047352  050220                          ADMSG1                  ;CAM 0 Y MESSAGE
6386  047354  001732                          DAC1                    ;DAC ADDRESS
6387  047356     000     001                  .BYTE   0,1             ;CAMERA 0, Y AXIS
6388                                    ;DO CAMERA 1, Y INPUT
6389  047360  004537  047422                  JSR     R5,ADJCAM
6390  047364  050404                          ADMSG3                  ;CAM 1, Y MESSAGE
6391  047366  001730                          DAC0                    ;DAC ADDRESS
6392  047370     001     001                  .BYTE   1,1             ;CAMERA 1, Y AXIS
6393                                    ;DO CAMERA 2, Y INPUT
6394  047372  004537  047422                  JSR     R5,ADJCAM
6395  047376  050570                          ADMSG5                  ;CAM 2, Y MESSAGE
6396  047400  001736                          DAC3                    ;DAC ADDRESS
6397  047402     002     001                  .BYTE   2,1             ;CAMERA 2, Y AXIS
6398                                    ;DO CAMERA 3, Y INPUT
6399  047404  004537  047422                  JSR     R5,ADJCAM
6400  047410  050754                          ADMSG7                  ;CAM 3, Y MESSAGE
6401  047412  001734                          DAC2                    ;DAC ADDRESS
6402  047414     003     001                  .BYTE   3,1             ;CAMERA 3 Y AXIS
6403
6404  047416  000137  002644                  JMP     RBEG0
```

```
                                        ;REPORT THE MESSAGE FIRST

047422 012500                   ADJCAM: MOV     (R5)+,R0        ;GET ASCII POINTER
047424 005777 132256            3$:     TST     @VTCHAR         ;WAIT FOR READY
047430 100375                           BPL     3$
047432 012777 012000 132254             MOV     #12000,@VTCXY   ;POSITION THE CAHRACTERS
047440 005777 132242            1$:     TST     @VTCHAR         ;WAIT FOR CHAR. READY
047444 100375                           BPL     1$
047446 112077 132234                    MOVB    (R0)+,@VTCHAR   ;LOAD THE CHAR.
047452 105710                           TSTB    (R0)            ;TEST IF TERM.
047454 001371                           BNE     1$              ;BR IF NOT

047456 013537 050104                    MOV     @(R5)+,DACADR   ;SAVE ADDRESS
047462 012537 050112                    MOV     (R5)+,CAMVAL    ;SAVE CAMERA DATA

047466 012777 002315 000410     2$:     MOV     #2315,@DACADR   ;PRESET DAC TO - GAIN INPUT
047474 004737 047576                    JSR     PC,SAMDAT
047500 010137 050120                    MOV     R1,VAL0         ;SAVE RESULTS
047504 004737 050022                    JSR     PC,FIXLIN       ;POSITION THE CROSS HAIR LINE
047510 012777 004000 000366             MOV     #4000,@DACADR   ;PRESET DAC TO OFFSET
047516 004737 047576                    JSR     PC,SAMDAT       ;SAMPLE THE DATA
047522 010137 050122                    MOV     R1,VAL1         ;SAVE AVERAGE RESULTS
047526 004737 050022                    JSR     PC,FIXLIN       ;POSITION THE CROSS HAIR LINE
047532 012777 005463 000344             MOV     #5463,@DACADR   ;PRESET DAC TO + GAIN INPUT
047540 004737 047576                    JSR     PC,SAMDAT       ;SAMPLE THE DATA
047544 010137 050124                    MOV     R1,VAL2         ;SAVE AVERAGE RESULTS
047550 004737 050022                    JSR     PC,FIXLIN       ;POSITION THE CROSS HAIR LINE
047554 004737 046042                    JSR     PC,CTRLCG       ;TEST FOR CTRL C/G
047560 122737 000103 046140             CMPB    #'C,CTRCHA      ;CHECK IF CHARACTER WAS A \C\
047566 001337                           BNE     2$              ;BR IF NOT
047570 005037 046140                    CLR     CTRCHA          ;CLEAR CHAR
047574 000205                           RTS     R5              ;EXIT

                                ;COLLECT 64 SAMPLES FROM THE SELECTED CAMERA
                                ;R1 CONTAINS THE AVERAGE

047576 012700 060000            SAMDAT: MOV     #BUF0,R0
047602 012701 062000                    MOV     #BUF1,R1
047606 005020                   1$:     CLR     (R0)+           ;CLEAR THE BUFFER
047610 020001                           CMP     R0,R1           ;FINISHED ?
047612 001375                           BNE     1$              ;BR IF NOT

047614 012777 177700 132130             MOV     #-100,@WCR      ;LOAD WORD COUNT
047622 012777 060000 132124             MOV     #BUF0,@BAR      ;LOAD ADDRESS
047630 113777 050112 132130             MOVB    CAMVAL,@CSRHB   ;SELECT THE CAMERA
047636 052777 000002 132112             BIS     #TESTZ,@SFR     ;SET TEST Z ENABLE
047644 052777 000001 132074             BIS     #BIT0,@CSR      ;ENABLE NCV11

047652 105777 132070            2$:     TSTB    @CSR            ;WAIT UNTIL DONE
047656 100375                           BPL     2$

047660 012777 004000 132070             MOV     #CLRALL,@SFR    ;CLEAR THE DEVICE
047666 005037 050100                    CLR     $TEMP1
047672 005037 050102                    CLR     $TEMP2
```

```
6459  047676  005001                          CLR     R1
6460  047700  012700  060000                  MOV     #BUF0,R0        ;LOAD BUFFER POINTER
6461  047704  105737  050113                  TSTB    CAMVAL+1        ;TEST IF X OR Y DATA
6462  047710  001413                          BEQ     5$              ;BR IF X
6463  047712  012037  050100          3$:     MOV     (R0)+,$TEMP1    ;GET THE DATA
6464  047716  113737  050101  050102          MOVB    $TEMP1+1,$TEMP2 ;GET HIGH BYTE DATA
6465  047724  063701  050102                  ADD     $TEMP2,R1       ;UPDATE AVERAGE
6466  047730  022700  060200                  CMP     #BUF0+200,R0    ;FINISHED DATA ?
6467  047734  001366                          BNE     3$              ;BR IF NOT
6468  047736  000412                          BR      10$
6469  047740  012037  050100          5$:     MOV     (R0)+,$TEMP1    ;GET DATA
6470  047744  113737  050100  050102          MOVB    $TEMP1,$TEMP2
6471  047752  063701  050102                  ADD     $TEMP2,R1       ;UPDATE THE AVERAGE
6472  047756  022700  060200                  CMP     #BUF0+200,R0    ;FINISHED DATA ?
6473  047762  001366                          BNE     5$              ;BR IF NOT
6474  047764  000240                  10$:    NOP
6475  047766  000240                          NOP
6476  047770  006201                          ASR     R1              ;AVERAGE THE DATA
6477  047772  000240                          NOP
6478  047774  006201                          ASR     R1              ;AVERAGE THE DATA
6479  047776  000240                          NOP
6480  050000  006201                          ASR     R1              ;AVERAGE THE DATA
6481  050002  000240                          NOP
6482  050004  006201                          ASR     R1              ;AVERAGE THE DATA
6483  050006  000240                          NOP
6484  050010  006201                          ASR     R1              ;AVERAGE THE DATA
6485  050012  000240                          NOP
6486  050014  006201                          ASR     R1              ;AVERAGE THE DATA
6487  050016  000240                          NOP
6488  050020  000207                          RTS     PC              ;EXIT
6489
6490                                  ;SUBROUTINE TO LOAD X OR Y CROSSHAIRS
6491  050022  006201                  FIXLIN: ASR     R1
6492  050024  000240                          NOP
6493  050026  000240                          NOP
6494  050030  000240                          NOP
6495  050032  062701  000100                  ADD     #100,R1
6496  050036  110177  131650                  MOVB    R1,@VTXPOS
6497  050042  000207                          RTS     PC
6498
6499                                  ;SUBROUTINE TO LOAD PIXEL DATA
6500  050044  042777  000400  131644  DISPLY: BIC     #400,@VTCSR     ;DISABLE MAP
6501  050052  010077  131642                  MOV     R0,@VTMAP       ;LOAD MAP PC
6502  050056  105777  131634          1$:     TSTB    @VTCSR          ;WAIT FOR MAP READY
6503  050062  100375                          BPL     1$
6504  050064  010177  131632                  MOV     R1,@VTPX        ;LOAD PIXEL DATA
6505  050070  052777  000400  131620          BIS     #400,@VTCSR     ;ENABLE MAP
6506  050076  000207                          RTS     PC
6507
6508  050100  000000                  $TEMP1: 0
6509  050102  000000                  $TEMP2: 0
6510  050104  000000                  DACADR: 0
6511  050106  000000                  RUNDIF: 0
6512  050110  000000                  WFMODE: 0               ;WF OPTION AREA FLAG
```

```
 6513  050112  000000                    CAMVAL: 0
 6514  050114  000000                    CURENT: 0
 6515  050116  000000                    AGAN:   0
 6516  050120  000000                    VAL0:   0
 6517  050122  000000                    VAL1:   0
 6518  050124  000000                    VAL2:   0
 6519  050126  040503  042515  040522  ADMSG0: .ASCII  /CAMERA 00/
 6520  050134  030040    060
 6521  050137    015     012                  .BYTE   15,12
 6522  050141    101  045104  051525          .ASCII  /ADJUST R09 FOR X OFFSET/
 6523  050146  020124  030122  020071
 6524  050154  047506  020122  020130
 6525  050162  043117  051506  052105
 6526  050170    015     012                  .BYTE   15,12
 6527  050172  042101  052512  052123          .ASCIZ  /ADJUST R24 FOR X GAIN/
 6528  050200  051040  032062  043040
 6529  050206  051117  054040  043440
 6530  050214  044501  000116
 6531  050220  040503  042515  040522  ADMSG1: .ASCII  /CAMERA 00/
 6532  050226  030040    060
 6533  050231    015     012                  .BYTE   15,12
 6534  050233    101  045104  051525          .ASCII  /ADJUST R13 FOR Y OFFSET/
 6535  050240  020124  030522  020063
 6536  050246  047506  020122  020131
 6537  050254  043117  051506  052105
 6538  050262    015     012                  .BYTE   15,12
 6539  050264  042101  052512  052123          .ASCIZ  /ADJUST R20 FOR Y GAIN/
 6540  050272  051040  030062  043040
 6541  050300  051117  054440  043440
 6542  050306  044501  000116
 6543  050312  040503  042515  040522  ADMSG2: .ASCII  /CAMERA 01/
 6544  050320  030040    061
 6545  050323    015     012                  .BYTE   15,12
 6546  050325    101  045104  051525          .ASCII  /ADJUST R10 FOR X OFFSET/
 6547  050332  020124  030522  020060
 6548  050340  047506  020122  020130
 6549  050346  043117  051506  052105
 6550  050354    015     012                  .BYTE   15,12
 6551  050356  042101  052512  052123          .ASCIZ  /ADJUST R23 FOR X GAIN/
 6552  050364  051040  031462  043040
 6553  050372  051117  054040  043440
 6554  050400  044501  000116
 6555  050404  040503  042515  040522  ADMSG3: .ASCII  /CAMERA 01/
 6556  050412  030040    061
 6557  050415    015     012                  .BYTE   15,12
 6558  050417    101  045104  051525          .ASCII  /ADJUST R14 FOR Y OFFSET/
 6559  050424  020124  030522  020064
 6560  050432  047506  020122  020131
 6561  050440  043117  051506  052105
 6562  050446    015     012                  .BYTE   15,12
 6563  050450  042101  052512  052123          .ASCIZ  /ADJUST R19 FOR Y GAIN/
 6564  050456  051040  034461  043040
 6565  050464  051117  054440  043440
 6566  050472  044501  000116
```

```
 6567  050476  040503  042515  040522  ADMSG4: .ASCII  /CAMERA 02/
 6568  050504  030040     062
 6569  050507     015     012                  .BYTE   15,12
 6570  050511     101  045104  051525          .ASCII  /ADJUST R11 FOR X OFFSET/
 6571  050516  020124  030522  020061
 6572  050524  047506  020122  020130
 6573  050532  043117  051506  052105
 6574  050540     015     012                  .BYTE   15,12
 6575  050542  042101  052512  052123          .ASCIZ  /ADJUST R22 FOR X GAIN/
 6576  050550  051040  031062  043040
 6577  050556  051117  054040  043440
 6578  050564  044501  000116
 6579  050570  040503  042515  040522  ADMSG5: .ASCII  /CAMERA 02/
 6580  050576  030040     062
 6581  050601     015     012                  .BYTE   15,12
 6582  050603     101  045104  051525          .ASCII  /ADJUST R15 FOR Y OFFSET/
 6583  050610  020124  030522  020065
 6584  050616  047506  020122  020131
 6585  050624  043117  051506  052105
 6586  050632     015     012                  .BYTE   15,12
 6587  050634  042101  052512  052123          .ASCIZ  /ADJUST R18 FOR Y GAIN/
 6588  050642  051040  034061  043040
 6589  050650  051117  054440  043440
 6590  050656  044501  000116
 6591  050662  040503  042515  040522  ADMSG6: .ASCII  /CAMERA 03/
 6592  050670  030040     063
 6593  050673     015     012                  .BYTE   15,12
 6594  050675     101  045104  051525          .ASCII  /ADJUST R12 FOR X OFFSET/
 6595  050702  020124  030522  020062
 6596  050710  047506  020122  020130
 6597  050716  043117  051506  052105
 6598  050724     015     012                  .BYTE   15,12
 6599  050726  042101  052512  052123          .ASCIZ  /ADJUST R21 FOR X GAIN/
 6600  050734  051040  030462  043040
 6601  050742  051117  054040  043440
 6602  050750  044501  000116
 6603  050754  040503  042515  040522  ADMSG7: .ASCII  /CAMERA 03/
 6604  050762  030040     063
 6605  050765     015     012                  .BYTE   15,12
 6606  050767     101  045104  051525          .ASCII  /ADJUST R16 FOR Y OFFSET/
 6607  050774  020124  030522  020066
 6608  051002  047506  020122  020131
 6609  051010  043117  051506  052105
 6610  051016     015     012                  .BYTE   15,12
 6611  051020  042101  052512  052123          .ASCIZ  /ADJUST R17 FOR Y GAIN/
 6612  051026  051040  033461  043040
 6613  051034  051117  054440  043440
 6614  051042  044501  000116
 6615
 6616  051046     015     012          PRIMO:  .BYTE   15,12
 6617  051050  044124  020105  042523          .ASCII  /THE SELF-TEST CONNECTOR MUST BE INSTALLED ON A017/
 6618  051056  043114  052055  051505
 6619  051064  020124  047503  047116
 6620  051072  041505  047524  020122
```

```
 6621  051100  052515  052123  041040
 6622  051106  020105  047111  052123
 6623  051114  046101  042514  020104
 6624  051122  047117  040440  030460
 6625  051130     067
 6626  051131     015     012                    .BYTE   15,12
 6627  051133     124  042510  051440            .ASCII  \THE SWITCH ON A017 MUST BE IN 'MAINT.'' POSITION (TOWARD THE I/O CONNECT
 6628  051140  044527  041524  020110
 6629  051146  047117  040440  030460
 6630  051154  020067  052515  052123
 6631  051162  041040  020105  047111
 6632  051170  021040  040515  047111
 6633  051176  027124  020042  047520
 6634  051204  044523  044524  047117
 6635  051212  024040  047524  040527
 6636  051220  042122  052040  042510
 6637  051226  044440  047457  041440
 6638  051234  047117  042516  052103
 6639  051242  051117     051
 6640  051245     015     012     000            .BYTE   15,12,0
 6641  051250     015     012            PRIM6:  .BYTE   15,12
 6642  051252  044124  020105  042523            .ASCII  /THE SELF-TEST CONNECTOR MUST BE REMOVED FROM THE A017/
 6643  051260  043114  052055  051505
 6644  051266  020124  047503  047116
 6645  051274  041505  047524  020122
 6646  051302  052515  052123  041040
 6647  051310  020105  042522  047515
 6648  051316  042526  020104  051106
 6649  051324  046517  052040  042510
 6650  051332  040440  030460     067
 6651  051337     015     012                    .BYTE   15,12
 6652  051341     124  042510  051440            .ASCII  \THE SWITCH ON A017 MUST BE IN 'OPER.'' POSITION (AWAY FROM THE I/O CONNE
 6653  051346  044527  041524  020110
 6654  051354  047117  040440  030460
 6655  051362  020067  052515  052123
 6656  051370  041040  020105  047111
 6657  051376  021040  050117  051105
 6658  051404  021056  050040  051517
 6659  051412  052111  047511  020116
 6660  051420  040450  040527  020131
 6661  051426  051106  046517  052040
 6662  051434  042510  044440  047457
 6663  051442  041440  047117  042516
 6664  051450  052103  051117     051
 6665  051455     015     012     000            .BYTE   15,12,0
 6666  051460  047111  042523  052122    PRIM5:  .ASCII  /INSERT RAM-PAK CABLE INTO THE SOCKET ON THE A017/
 6667  051466  051040  046501  050055
 6668  051474  045501  041440  041101
 6669  051502  042514  044440  052116
 6670  051510  020117  044124  020105
 6671  051516  047523  045503  052105
 6672  051524  047440  020116  044124
 6673  051532  020105  030101  033461
 6674  051540     015     012                    .BYTE   15,12
```

```
 6675  051542  042523  020124  040522          .ASCII  /SET RAM-PAK SWITCHES TO ''IN'', 'NORM'' AND 'OFF''/
 6676  051550  026515  040520  020113
 6677  051556  053523  052111  044103
 6678  051564  051505  052040  020117
 6679  051572  044442  021116  020054
 6680  051600  047042  051117  021115
 6681  051606  040440  042116  021040
 6682  051614  043117  021106
 6683  051620     015     012                  .BYTE   15,12
 6684  051622  047111  042523  052122  PRIM3:  .ASCII  /INSERT BLANK ROM INTO ROM BLASTER SOCKET/
 6685  051630  041040  040514  045516
 6686  051636  051040  046517  044440
 6687  051644  052116  020117  047522
 6688  051652  020115  046102  051501
 6689  051660  042524  020122  047523
 6690  051666  045503  052105
 6691  051672     015     012                  .BYTE   15,12
 6692  051674  046120  040505  042523          .ASCII  \PLEASE DEPRESS THE ''I/O'' AND 'EXECUTE'' BUTTONS TOGETHER\
 6693  051702  042040  050105  042522
 6694  051710  051523  052040  042510
 6695  051716  021040  027511  021117
 6696  051724  040440  042116  021040
 6697  051732  054105  041505  052125
 6698  051740  021105  041040  052125
 6699  051746  047524  051516  052040
 6700  051754  043517  052105  042510
 6701  051762     122
 6702  051763     015     012                  .BYTE   15,12
 6703  051765     124  042510  032040          .ASCII  \THE 4 PROCESS LEDS WILL LIGHT WHEN DONE CORRECTLY\
 6704  051772  050040  047522  042503
 6705  052000  051523  046040  042105
 6706  052006  020123  044527  046114
 6707  052014  046040  043511  052110
 6708  052022  053440  042510  020116
 6709  052030  047504  042516  041440
 6710  052036  051117  042522  052103
 6711  052044  054514
 6712  052046     015     012     000          .BYTE   15,12,0
 6713  052051     015     012             PRIM1:  .BYTE   15,12
 6714  052053     120  042514  051501          .ASCII  /PLEASE REMOVE RAM-PAK CABLE/
 6715  052060  020105  042522  047515
 6716  052066  042526  051040  046501
 6717  052074  050055  045501  041440
 6718  052102  041101  042514
 6719  052106     015     012                  .BYTE   15,12
 6720  052110  040440  042116  044440          .ASCII  / AND INSERT THE BLASTED ROM INTO SOCKET ON A017 MODULE/
 6721  052116  051516  051105  020124
 6722  052124  044124  020105  046102
 6723  052132  051501  042524  020104
 6724  052140  047522  020115  047111
 6725  052146  047524  051440  041517
 6726  052154  042513  020124  047117
 6727  052162  040440  030460  020067
 6728  052170  047515  052504  042514
```

```
6729  052176     015     012                      .BYTE   15,12
6730  052200  042504  051120  051505              .ASCII  /DEPRESS THE ''CR'' KEY WHEN READY/
6731  052206  020123  044124  020105
6732  052214  041442  021122  045440
6733  052222  054505  053440  042510
6734  052230  020116  042522  042101
6735  052236     131
6736  052237     015     012     000              .BYTE   15,12,0
6737  052242     015     012              PRIM2:  .BYTE   15,12
6738  052244  050042  030512  020042              .ASCII  /''PJ1'' CABLE MUST BE CONNECTED TO THE A017 CONNECTOR/
6739  052252  040503  046102  020105
6740  052260  052515  052123  041040
6741  052266  020105  047503  047116
6742  052274  041505  042524  020104
6743  052302  047524  052040  042510
6744  052310  040440  030460  020067
6745  052316  047503  047116  041505
6746  052324  047524     122
6747  052327     015     012     000              .BYTE   15,12,0
6748  052332     015     012              PRIM4:  .BYTE   15,12
6749  052334  050042  031512  020042              .ASCII  /''PJ3'' CABLE MUST BE CONNECTED TO THE M8036 CONNECTOR/
6750  052342  040503  046102  020105
6751  052350  052515  052123  041040
6752  052356  020105  047503  047116
6753  052364  041505  042524  020104
6754  052372  047524  052040  042510
6755  052400  046440  030070  033063
6756  052406  041440  047117  042516
6757  052414  052103  051117
6758  052420     015     012     000              .BYTE   15,12,0
6759          052424                              .EVEN
6760  052424     015     012     007      FATAL0: .BYTE   15,12,7 ;''RUNIT'' MUST BE ON A WORD BOUNDRY
6761  052427     106  052101  046101              .ASCII  /FATAL BUS TRAP WHEN PERFORMING TEST ''/
6762  052434  041040  051525  052040
6763  052442  040522  020120  044127
6764  052450  047105  050040  051105
6765  052456  047506  046522  047111
6766  052464  020107  042524  052123
6767  052472  021040
6768  052474  021103  040411  020124      RUNIT:  .ASCIZ  /C''     AT LOCATION /   ;''RUNIT'' MUST BE A WORD BOUNDRY
6769  052502  047514  040503  044524
6770  052510  047117  000040
6771  052514  005015  051120  043517      RUNITA: .ASCIZ  <15><12>/PROGRAM RESTARTING/    ;''RUNIT'' MUST BE A WORD BOUNDRY
6772  052522  040522  020115  042522
6773  052530  052123  051101  044524
6774  052536  043516     000
6775  052541     015     012     007              .BYTE   15,12,7,0
6776  052544     000
6777  052545     015     012              MSGMEM: .BYTE   15,12
6778  052547     122  047125  044516              .ASCIZ  /RUNNING WITH /
6779  052554  043516  053440  052111                       .
6780  052562  020110     000
6781  052565     040  020113  043117      MSGK:   .ASCIZ  / K OF MEMORY/<15><12>
6782  052572  046440  046505  051117
```

```
  6783  052600  006531  000012
  6784  052604     015     012     007  WARNO:  .BYTE   15,12,~
  6785  052607     117  052103  046101          .ASCIZ  /OCTAL ADDRESS OF TERMINAL = /
  6786  052614  040440  042104  042522
  6787  052622  051523  047440  020106
  6788  052630  042524  046522  047111
  6789  052636  046101  036440  000040
  6790  052644     033     000          ESCP:   .BYTE   33,0
  6791  052646  043113  027462     062  KF0:    .ASCII  \KF2/2\
  6792  052653     015     000                  .BYTE   15,0
  6793  052655     106  031515     060  FM0:    .ASCII  \FM30\
  6794  052661     015     000                  .BYTE   15,0
  6795  052663     104  030111  031457  DIO:    .ASCII  \DIO/37\
  6796  052670     067
  6797  052671     015     002                  .BYTE   15,2
  6798  052673                      DIDATA:
  6799  052673     060     060     060          .BYTE   60,60,60,40
  6800  052676     040
  6801  052677     060     060     060          .BYTE   60,60,60,40
  6802  052702     040
  6803  052703     060     060     060          .BYTE   60,60,60,40
  6804  052706     040
  6805  052707     060     060     060          .BYTE   60,60,60,40
  6806  052712     040
  6807  052713     060     060     060          .BYTE   60,60,60,40
  6808  052716     040
  6809  052717     060     060     060          .BYTE   60,60,60,40
  6810  052722     040
  6811  052723     060     060     060          .BYTE   60,60,60,40
  6812  052726     040
  6813  052727     060     060     060          .BYTE   60,60,60,40
  6814  052732     040
  6815  052733     060     060     060          .BYTE   60,60,60,40
  6816  052736     040
  6817  052737     060     060     060          .BYTE   60,60,60,40
  6818  052742     040
  6819  052743     060     060     060          .BYTE   60,60,60,40
  6820  052746     040
  6821  052747     060     060     060          .BYTE   60,60,60,40
  6822  052752     040
  6823  052753     060     060     060          .BYTE   60,60,60,40
  6824  052756     040
  6825  052757     060     060     060          .BYTE   60,60,60,40
  6826  052762     040
  6827  052763     060     060     060          .BYTE   60,60,60,40
  6828  052766     040
  6829  052767     060     060     060          .BYTE   60,60,60,40
  6830  052772     040
  6831  052773     060     060     060          .BYTE   60,60,60,40
  6832  052776     040
  6833  052777     060     060     060          .BYTE   60,60,60,40
  6834  053002     040
  6835  053003     060     060     060          .BYTE   60,60,60,40
  6836  053006     040
```

```
 6837  053007      060      060      060            .BYTE   60,60,60,40
 6838  053012      040
 6839  053013      060      060      060            .BYTE   60,60,60,40
 6840  053016      040
 6841  053017      060      060      060            .BYTE   60,60,60,40
 6842  053022      040
 6843  053023      060      060      060            .BYTE   60,60,60,40
 6844  053026      040
 6845  053027      060      060      060            .BYTE   60,60,60,40
 6846  053032      040
 6847  053033      060      060      060            .BYTE   60,60,60,40
 6848  053036      040
 6849  053037      060      060      060            .BYTE   60,60,60,40
 6850  053042      040
 6851  053043      060      060      060            .BYTE   60,60,60,40
 6852  053046      040
 6853  053047      060      060      060            .BYTE   60,60,60,40
 6854  053052      040
 6855  053053      060      060      060            .BYTE   60,60,60,40
 6856  053056      040
 6857  053057      060      060      060            .BYTE   60,60,60,40
 6858  053062      040
 6859  053063      060      060      060            .BYTE   60,60,60,40
 6860  053066      040
 6861  053067      060      060      060            .BYTE   60,60,60,40
 6862  053072      040
 6863  053073      003      000                     .BYTE   3,0
 6864  053075      103   030115   031457  CM0:      .ASCII  \CM0/37\
 6865  053102      067
 6866  053103      015      000                     .BYTE   15,0
 6867  053105      103   000103           CC0:      .ASCIZ  /CC/
 6868  053110   043520   027460   033463  PG0:      .ASCII  \PG0/37\
 6869  053116      015      000                     .BYTE   15,0
 6870  053120   050132           ZP0:      .ASCII  /ZP/
 6871  053122      015      000                     .BYTE   15,0
 6872                                               .EVEN
 6873
 6874                           ;BUFFER AREA
 6875
 6876           060000                              .=60000
 6877  060000   001000           BUF0:     .BLKW   512.
 6878  062000   001000           BUF1:     .BLKW   512.
 6879  064000   000240           BUF2:     NOP
 6880
 6881           000001                              .END
```

```
ABASE = 172760          277#    388     429     547     548     549     550     551     552     553     554     555     556
ABOMSG  041663         5377#   6072
ABORTB  041545         5357#   5548
ABOVE   041752         5397#   5972*   6036    6070
ACDW1 = 170420          279#    388     431     541     542     543
ACDW2 = 000000          388
ACPUOP= 000000          388     403
ADDW0 = 000000          388
ADDW1 = 000000          388
ADDW10= 000000          388
ADDW11= 000000          388
ADDW12= 000000          388
ADDW13= 000000          388
ADDW14= 000000          388
ADDW15= 000000          388
ADDW2 = 000000          388
ADDW3 = 000000          388
ADDW4 = 000000          388
ADDW5 = 000000          388
ADDW6 = 000000          388
ADDW7 = 000000          388
ADDW8 - 000000          388
ADDW9 = 000000          388
ADEVCT= 000000          388     394
ADEVM - 000000          388     430
ADJCAM  047422         6363    6368    6373    6378    6384    6389    6394    6399    6407#
ADJFIX  043422         5541    5707#
ADM     001760          552#    856     1660    1677    1694    1711    1728    1745    1762    1779    1796    1813    1830
                       1847    1864    1881    1898    1915    1932    1949    1966    1983    2000    2017    2035    2053
ADMSG0  050126         6364    6519#
ADMSG1  050220         6385    6531#
ADMSG2  050312         6369    6543#
ADMSG3  050404         6390    6555#
ADMSG4  050476         6374    6567#
ADMSG5  050570         6395    6579#
ADMSG6  050662         6379    6591#
ADMSG7  050754         6400    6603#
ADNOKT  003352          796#    811*    3279    3323    3352    3395    3414    3421    3452
AENV  - 000000          388     399
AENVM = 000000          388     400
AFATAL= 000000          388     391
AGAN    050116          582*    585*    587*    683     6515#
AMADR1= 000000          388     416
AMADR2= 000000          388     420
AMADR3= 000000          388     423
AMADR4= 000000          388     426
AMAMS1= 000000          388     410
AMAMS2= 000000          388     418
AMAMS3= 000000          388     421
AMAMS4= 000000          388     424
AMSGAD= 000000          388     396
AMSGLG= 000000          388     397
AMSGTY- 000000          388     390
AMTYP1  000000          388     411
```

```
EM25    031500          489    3840#
EM26    031533          491    3845#
EM27    031571          493    3851#
EM3     030324          453    3726#
EM30    031640          495    3858#
EM31    031700          497    3864#
EM32    031741          499    3870#
EM33    032005          501    3877#
EM34    032051          503    3884#
EM35    032106          505    3889#
EM36    032173          507    3898#
EM37    032241          509    3905#
EM4     030372          455    3733#
EM40    032305          511    3912#
EM41    032365          513    3921#
EM42    032425          515    3927#
EM5     030432          457    3739#
EM6     030473          459    3745#
EM7     030527          461    3750#
ENDDMA= 000400          575#   1115   1118   1268   1304   1340   5880   6165
ERMSG   041037         5294#
ERRVEC= 000004          119#    614    615*   626*   656*   836    837*   840*   849    850*   858*   863*   871*
                        873*   4241   4242*  4244*  4247*  4663   4664   4668*  4673*  4687*  4692*  4701*  4711*
                       4712*
ESCP    052644         5510   5612   6790#
FAIMSG  041714         5382#  6052
FATALO  052424          760   6760#
FATFNG  003202          750    754#   773
FIELDI  041374          785   5337#
FIRST   041774         5406#  5824   5992   5994*
FIXLIN  050022         6423   6427   6431   6491#
FMO     052655         5514   5616   6793#
FSITE   037354          793   5025#  5137
GAIN    040730         5278#  5795   5813   6076
GAINX   040761         5283#  5547
GAIN1   040773         5285#  5524*  5531*  5792*  5805*
GNS     = ****** U      298    652   5009   5010   5011   5012   5013   5015   5017   5018   5019   5020   5021
                       5022
GTSWR   = 104406        647    747   5015#  6174
G1B     040724         5221   5277#
G1LIM0  046216         5522   5801   6209#
G1LIM1  046226         5523   5802   6216#
G2LIM0  046236         5529   5808   6223#
G2LIM1  046246         5530   5809   6230#
HT      = 000011          29#  4322   4363
HT2T3   046040         5861*  5866   6157#
INOUTZ  040410          786*   792*  5155   5211#
IOTVEC= 000020          124#   598*   599*
JOY     001762          553#  1409   1456   1473   1490   1506   1523   1540   1556   1573   1591   1608   1626
                       1643   5076
JOYG1   040514         5076*  5136   5239#
JOYTW   040706         5134   5274#
JOYTX   040706         5273#
KDPAR0= 177360          267#
```

```
KDPAR1= 172362            268#
KDPAR2= 172364            269#
KDPAR3= 172366            270#
KDPAR4= 172370            271#
KDPAR5= 172372            272#
KDPAR6= 172374            273#
KDPAR7= 172376            274#
KDPDR0= 172320            245#
KDPDR1= 172322            246#
KDPDR2= 172324            247#
KDPDR3= 172326            248#
KDPDR4= 172330            249#
KDPDR5= 172332            250#
KDPDR6= 172334            251#
KDPDR7= 172336            252#
KF0     052646           5512    5614    6791#
KIPAR0= 172340            256#   4649    4679
KIPAR1= 172342            257#    816*
KIPAR2= 172344            258#    818*
KIPAR3= 172346            259#   3464*   3516*   3569*   3635*
KIPAR4= 172350            260#
KIPAR5= 172352            261#
KIPAR6= 172354            262#
KIPAR7= 172356            263#    821*   4695
KIPDR0= 172300            234#
KIPDR1= 172302            235#    817*
KIPDR2= 172304            236#    819*
KIPDR3= 172306            237#    820*
KIPDR4= 172310            238#
KIPDR5= 172312            239#
KIPDR6= 172314            240#
KIPDR7= 172316            241#    822*
KTERR   036404           4676    4722#   4723
KWCSR   001740            541#    677*    678     679     859    2389*   2412*   2414*   2415    5834*   5856*   5869*   5877*
                         5878
KWCSR1  001742            542#    678*    680*   5839*   5840*   5846*   5847*   5853*   5854*
KWPSR   001744            543#    679*    681*    860    2413*   5833*   5841    5848    5855    5868*
LF      = 000012           30#   4357    4363
LIMITS  046152           5521    5528    5800    5807    6183#
LINEA   041152           5309#
LIST    046466            694    6261#
LISTDT  045640           5836    5843    5850    6114#   6142
LISTO   046256            692    6235#
LIST1   047101            696    6315#
LOGIC   003356            688     716     725     799#   3707
LOOPC   041776           5420#   5429    5434    5438    5441    5452    5456
LOOPD   042022           5421    5428#
LOOPE   042064           5431    5440#
LOOPF   042142           5447    5454#
LSBAVG  041726           5385#
LSBMSG  041004           5288#
LSBSVQ  041730           5386#   5749*   5760
LSBSVR  041732           5387#   5748*
LSBSVW  041734           5388#   5715*   5776
```

```
LSBTXT  041610          5366#   6012*   6013*   6014*
MMVEC = 000250           135#   4676*   4677*
MORTST= 000000           580#    877    3975    5023
MCGK    052565           835    6781#
MSG%EM  052545           825    6777#
MSG21   041175          5313#
NARA    041760          5400#   5982*   6040    6057
NARB    041762          5401#   5976*   6044    6058
NARCNT  046210          6046    6200#
NARMSG  041066          5299#   6061
NARROW  041756          3480*   3543*   3590*   3656*   3970    5399#
NARTOL  046211          5964    6201#
NLSI11  003354           797#    838*    842*   2673
NOEXIT  043274          5558*   5625*   5652    5662*   5669#
NOMIAL  041742          5391#   5924
NORCNT  046206          6048    6189    6198#
NORMAL  041754          5398#   5986*   6048    6062
NORMSG  041642          5374#   6064
NORTOL  046207          5962    6199#
NURCNT  046212          6042    6202#
NURTOL  046213          5966    6203#
OBSCNT  046214          6038    6204#
OBSTOL  046215          6205#
OFF     001750           548#    852    1039*   1040    1054*   1061    1093*   1095    1375*   1388    2066*   2101    2140
                        2171*   2206*   2240*   2280*   2291    2306*   2320    2457*   2495*   2525*   2572*   2612*   2654*
                        2701*   2722*   2743*   2764*   2785*   2806*   2827*   2848*   2869*   2890*   2911*   2932*   2956*
                        2991*   3012*   3042*   3063*   3089*   3112*   3286*   3486*   3535*   3578*   3610*   3644*   5871*
                        6115*
OKMSG   041054          5297#
OUT     041772          3463*   3464    3470    3504*   3505    3515*   3516    3522    3556*   3557    5405#
OUTCHN  032714          3962#   5173*
OUTRNG  032621          3951#   5174
PASMSG  041702          5380#   6050    6078
PBLAST  043004           741    5609#
PC    -%000007            50#    806*    813*   3688*   3691*   3701*   3706    4029*   4235*   4312*   4571*   4338*   4345*
                        4359*   4361*   4469*   4475*   4533*   4574*   4591*   4670    4719*   4815*   4933*   4938*   4973*
                        4976*   5025*   5166*   5526*   5533*   5538*   5541*   5550*   5606*   5671*   5679*   5681*   5704*
                        5735*   5742*   5785*   5831*   5836*   5843*   5850*   5863*   5923*   5941*   5947*   6009*   6017*
                        6088*   6106*   6139*   6142*   6152*   6175*   6337*   6543*   6349*   6421*   6423*   6425*   6427*
                        6429*   6431*   6432*   6488*   6497*   6506*
PGO     053110          5557    5604    5624    6868#
PIRQ  - 177772            56#
PIRQVE- 000240           130#
POTIME  047106           721    6321#
PRIME   046142          6177#
PRIME0  046144          6178#
PRIME1  046146          5520*   5527*   5799*   5806*   5874    6119    6179#
PRIM0   051046          5506    5798    6616#
PRIM1   052051          5562    6713#
PRIM2   052242          6322    6737#
PRIM3   051622          5609    6684#
PRIM4   052332           804    6748#
PRIM5   051460          5507    6666#
PRIM6   051250           784    6641#
```

```
PROROM  042262           5493#   5618
PR0    = 000000            53#
PR1    = 000040            54#
PR2    = 000100            55#
PR3    = 000140            56#
PR4    = 000200            57#
PR5    = 000240            58#
PR6    = 000300            59#
PR7    = 000340            60#
PS     = 177776            33#     34
PSW    = 177776            34#    839*
PWRVEC= 000024            125#    604*    605*   4604*   4605*   4614*   4620*   4632*   4633*
RAMVAL  042222           5477#   5571    5597
RBEG0   002644            684     696#    748     756    5453    5549    5566    5629    5655    5818    6404
RDBLST  043112           5509    5511    5513    5552    5554    5556    5559    5599    5601    5603    5611    5613    5615
                         5619    5621    5623    5626    5634#
RDCHR  = 104410           4088   5018#
RDLIN  = 104411            697     787    4129   5019#   5563
RDOCT  = 104412            771   5020#
READ    045302           6020   6035#
RFADXX  045520           6082#
READ1   044564           5929#
READ7   045516           6050*   6052*   6078#
REDJOY= 000001            579#    1398    1421    1453    1470    1487    1503    1520    1537    1553    1570    1588    1605
                          1623    1640
RESREG= 104414           4937    5022#
RESVEC= 000010            120#
ROMPNT  041740           5390#   5525*   5532*   5708
ROMVAL  042162           5390    5461#   5516    5518    5525    5532    5572    5593    6087    6089
RTN     041617           5369#   5998    6091    6092    6093    6097
RTRT    002012            303     582#    767    6171
RTRTA   002042            583     586     589#    781
RUNDIF  050106            687*    711*    715*    724*    751*   3670    5815   6511#
RUNIT   052474            655*    698*    699*    700*    701     706     709     713     719     722     726     729     734
                          739     742     745     749    5987    6003   6768#
RUNITA  052514            766   6771#
R0      %000000            41#    657*    659*    661     663*    664*    665*    666*    673*    675     772*    774     777
                          778     826*    827*    828*    829*    830*    831*    832*    833    1258*   1266*   1294*   1302*
                         1330*   1338*   1399*   1402*   1422*   1425*   2117*   2118*   2119    2671*   2672*   2675*   2677
                         3277*   3280*   3281    3294*   3300*   3306*   3307    3309    3310    3322    3323    3350*   3353*
                         3354    3365*   3371*   3377*   3378    3380    3381    3394    3395    3414*   3415*   3416*   3417*
                         3427    3432*   3437*   3443*   3444    3445    3451    3452    3465*   3466*   3468*   3495    3517*
                         3518*   3520*   3547    3570*   3571*   3573*   3593    3636*   3637*   3639*   3659    3698*   3701
                         4126    4130*   4133    4149*   4165    4175*   4179    4195    4196    4209*   4305    4306*   4311
                         4316    4319*   4505    4506*   4507*   4514*   4515*   4516*   4517*   4518*   4519    4524    4529*
                         4531*   4535    4537    4551    4559*   4563    4564    4566*   4567*   4568    4590*   4606    4631*
                         4659    4665*   4710    4718*   4742    4767*   4784    4797*   4799*   4800*   4806*   4808*   4810
                         4814*   4844    4851*   4854*   4856*   4861*   4868*   4869    4872*   4879*   4880    4884*   4891
                         4895*   4913*   4914    4932*   4936*   4986    4987*   4988    4989*   4990*   4991*   4992*   5158*
                         5167*   5171*   5172*   5173    5179*   5183    5184    5425*   5435*   5436    5516*   5517*   5518
                         5564*   5572*   5573    5593    5634    5635*   5637    5643    5661*   5672*   5682*   5701*   5707*
                         5711    5712    5770*   5771    5781*   5822*   5823*   5824    5826*   5827*   5828    5858*   5859*
                         5860*   5861    5866*   5867*   5868    5884*   5887    5888*   5896*   5899    5902    5930*   6000
                         6018*   6019    6087*   6089    6098    6127*   6130    6131*   6183*   6186*   6190    6191    6192
```

```
                    6193    6336*   6338*   6339    6342*   6344*   6345    6348*   6350*   6351    6407*   6413    6414
                    6441*   6443*   6444    6460*   6463    6466    6469    6472    6501
R1      =%000001      42#    658*    659     660*    667*    668     672*    673     674     776*    777*    779    1259*
                    1264*   1295*   1300*   1331*   1336*   3278*   3280    3295*   3298*   3351*   3353    3366*   3369*
                    3420*   3422    3435*   3470*   3472*   3473*   3474*   3475*   3476*   3478*   3481    3482*   3486
                    3522*   3524*   3525*   3526*   3527*   3528*   3530*   3537    3542    4127    4131*   4135*   4137*
                    4139*   4142*   4145    4148*   4166    4179*   4180    4184    4208*   4552    4589*   4607    4630*
                    4660    4669*   4703*   4705    4706    4708*   4713    4717*   4743    4766*   4785    4788*   4791*
                    4801*   4807*   4811    4813*   4845    4850*   4855*   4873*   4876    4880*   4889*   4890    4894*
                    4915*   4921*   4927*   5159*   5164*   5184*   5186*   5187    5432    5436*   5573*   5574*   5576*
                    5578*   5580*   5582*   5584*   5586*   5588*   5590*   5673*   5674    5676    5678    5680    5685
                    5711*   5712*   5713*   5714*   5715    5733    5771*   5776*   5781    5885*   5889*   5891    5892*
                    5893    5897*   5899*   5906*   5909*   5912*   5915*   5918*   5921*   5922    5931*   5932    6096*
                    6104*   6130*   6132*   6133    6189*   6190*   6191*   6192*   6193*   6335*   6422    6426    6430
                    6442*   6444    6459*   6465*   6471*   6476*   6478*   6480*   6482*   6484*   6486*   6491*   6495*
                    6496    6504
R2      -%000002      43#    668*    669*    670*    671     3279*   3281    3352*   3354    3421*   3423    3471*   3477*
                    3479*   3480    3482    3523*   3529*   3531*   3535    3543    4128    4132*   4136*   4138*   4140*
                    4146    4147*   4167    4178*   4182*   4185    4192*   4193*   4194    4199*   4207*   4608    4629*
                    4661    4679*   4681*   4682*   4685*   4686*   4694*   4695    4697*   4702*   4704*   4709*   4714
                    4716*   4744    4765*   4786    4792*   4795*   4799    4812*   4846    4857*   4865*   4870    4893*
                    4912*   4915    4916*   4922*   4923*   4928*   4929*   5183*   5185*   5188    5442*   5443*   5444
                    5446    5448    5450*   5451    5575*   5577*   5579*   5581*   5583*   5585*   5587*   5589*   5591*
                    5592    5676*   5678*   5680*   5692*   5693*   5694*   5695*   5696*   5697*   5698*   5699*   5700*
                    5701    5708*   5709    5753    5766*   5782    5837    5844    5851    5898*   5901*   5905*   5908*
                    5911*   5914*   5917*   5920*   5954*   5955*   5956    6010*   6011*   6012    6013    6014    6036*
                    6037*   6038    6040*   6041*   6042    6044*   6045*   6046    6057*   6058*   6059    6065*   6066*
                    6067    6098*   6099*   6100    6128*   6133*   6136*   6137*   6138*   6143
R3      -%000003      44#    3419*   3422*   3423    4084    4085*   4086    4089*   4090    4094    4096    4098*   4100*
                    4168    4176*   4177*   4191*   4194*   4203*   4204*   4206*   4396    4405*   4411*   4412*   4415*
                    4420*   4421*   4422    4431*   4609    4628*   4662    4680*   4684*   4715*   4745    4764*   4847
                    4869*   4870*   4872    4892*   4920*   4925*   4931*   4932    5180*   5188*   5191*   5192*   5202
                    5571*   5592*   5753*   5755*   5760*   5766    6129*   6134*
R4      -%000004      45#    4397    4399*   4400*   4401*   4402    4403*   4417    4419*   4427*   4430*   4610    4627*
                    4746    4763*   4918*   4921    4927    4929    5181*   5187*   5195*   5196*   5200    5709*   5710*
                    5782
R5      %000005       46#    4169    4171*   4173*   4180*   4184*   4199    4205*   4398    4404*   4406*   4408*   4409*
                    4410*   4411    4429*   4611    4626*   4747    4762*   4919*   4923    4930    5029*   5034*   5040*
                    5045*   5051*   5056*   5062*   5067*   5079*   5083*   5085*   5089*   5092*   5096*   5098*   5102*
                    5105*   5109*   5111*   5115*   5118*   5122*   5124*   5128*   5131*   5135*   5139    5143    5144*
                    5146    5203*   5208*   5214    5228*   5509*   5511*   5513*   5521*   5528*   5552*   5554*   5556*
                    5559*   5596*   5599*   5601*   5603*   5611*   5613*   5615*   5617*   5619*   5621*   5623*   5626*
                    5635    5663*   5673    5687*   5800*   5807*   6183    6186    6188    6194*   6363*   6368*   6373*
                    6378*   6384*   6389*   6394*   6399*   6407    6417    6418    6436*
R6      %000006       47#     592*    593*    594
R7      -%000007      48#
SAMCAM  047276      6361#
SAMDAT  047576      6421    6425    6429    6441#
SAVREG= 104413      491'    5021#
SDPAR0- 172260       223#
SDPAR1= 172262       224#
SDPAR2= 172264       225#
SDPAR3= 172266       226#
SDPAR4= 172270       227#
```

SDPAR5= 172272 228#
SDPAR6= 172274 229#
SDPAR7= 172276 230#
SDPDR0= 172220 201#
SDPDR1= 172222 202#
SDPDR2= 172224 203#
SDPDR3= 172226 204#
SDPDR4= 172230 205#
SDPDR5= 172232 206#
SDPDR6= 172234 207#
SDPDR7= 172236 208#
SFR 001756 551# 855 916* 949* 950 964* 967 977* 980 989* 990* 992 1004*
1021* 1038* 1053* 1057* 1076 1085* 1092* 1094* 1103* 1104* 1115* 1118* 1128*
1138* 1139* 1144* 1145* 1164* 1165* 1170* 1171* 1192* 1193* 1198* 1199* 1215*
1216* 1221* 1222* 1233* 1237* 1238* 1249* 1250* 1256* 1257* 1268* 1273* 1284*
1285* 1291* 1292* 1304* 1305* 1310* 1321* 1322* 1328* 1329* 1340* 1341* 1356*
1362* 1371* 1372* 1378* 1379* 1397* 1398* 1400 1404 1410 1420* 1421* 1423
1427 1431* 1432 1439 1441 1450* 1451* 1453* 1454 1467* 1468* 1470* 1471
1484* 1485* 1487* 1488 1500* 1501* 1503* 1504 1517* 1518* 1520* 1521 1534*
1535* 1537* 1538 1550* 1551* 1553* 1554 1567* 1568* 1570* 1571 1585* 1586*
1588* 1589 1602* 1603* 1605* 160[illegible] 1620* 1621* 1623* 1624 1637* 1638* 1640*
1641 1654* 1655* 1657* 1659* 167[illegible]* 1672* 1674* 1676* 1688* 1689* 1691* 1693*
1705* 1706* 1708* 1710* 1722* 1723* 1725* 1727* 1739* 1740* 1742* 1744* 1756*
1757* 1759* 1761* 1773* 1774* 1776* 1778* 1790* 1791* 1793* 1795* 1807* 1808*
1810* 1812* 1824* 1825* 1827* 1829* 1841* 1842* 1844* 1846* 1858* 1859* 1861*
1863* 1875* 1876* 1878* 1880* 1892* 1893* 1895* 1897* 1909* 1910* 1912* 1914*
1926* 1927* 1929* 1931* 1943* 1944* 1946* 1948* 1960* 1961* 1963* 1965* 1977*
1978* 1980* 1982* 1994* 1995* 1997* 1999* 2011* 2012* 2014* 2016* 2029* 2030*
2032* 2034* 2047* 2048* 2050* 2052* 2064* 2069* 2071* 2072* 2076* 2081* 2116*
2123* 2127* 2128* 2129* 2143* 2169* 2173* 2176* 2177* 2181* 2193* 2204* 2208*
2210* 2211* 2215* 2227* 2238* 2242* 2252* 2253* 2257* 2267* 2275* 2281* 2284*
2285* 2286* 2301* 2307* 2310* 2311* 2315* 2329* 2332* 2338* 2339* 2354* 2357*
2363* 2364* 2379* 2384* 2390* 2404* 2409* 2417* 2431* 2434* 2440* 2441* 2455*
2459* 2461* 2462* 2466* 2494* 2497* 2501* 2502* 2503* 2514* 2522* 2527* 2529*
2530* 2536* 2549* 2569* 2574* 2576* 2577* 2582* 2592* 2609* 2614* 2616* 2617*
2626* 2638* 2649* 2656* 2659* 2660* 2664* 2692* 2700* 2702* 2705* 2706* 2708*
2721* 2723* 2726* 2727* 2729* 2742* 2744* 2747* 2748* 2750* 2763* 2765* 2768*
2769* 2771* 2784* 2786* 2789* 2790* 2792* 2805* 2807* 2810* 2811* 2813* 2826*
2828* 2831* 2832* 2834* 2847* 2849* 2852* 2853* 2855* 2868* 2870* 2873* 2874*
2876* 2889* 2891* 2894* 2895* 2897* 2910* 2912* 2915* 2916* 2918* 2931* 2933*
2936* 2937* 2939* 2955* 2957* 2960* 2961* 2964* 2990* 2992* 2994* 2996* 2998*
3011* 3013* 3016* 3017* 3018* 3021* 3041* 3043* 3046* 3047* 3049* 3062* 3064*
3067* 3068* 3070* 3086* 3090* 3093* 3094* 3096* 3109* 3114* 3116* 3117* 3123*
3148* 3149* 3175* 3176* 3201* 3202* 3225* 3226* 3248* 3249* 3285* 3289* 3291*
3357* 3360* 3362* 3363* 3426* 3429* 3431* 3485* 3488* 3490* 3491* 3493* 3534*
3538* 3540* 3541* 3544* 3577* 3580* 3582* 3583* 3586* 3607* 3611* 3614* 3615*
3617* 3643* 3645* 3648* 3649* 3652* 5073* 5074 5147* 5155* 5870* 5875* 5880*
6114* 6122* 6165* 6169* 6450* 6456*
SFRHB 001770 556# 966*
SGNVAL 041600 5364# 6015
SGNVL1 041606 5365# 5936* 5938* 5968 5974 5980
SHUF0 043402 5677 5699#
SHUF3 043374 5679 5696#
SHUF6 043362 5681 5692#

```
SIPAR0= 172240          212#
SIPAR1= 172242          213#
SIPAR2= 172244          214#
SIPAR3= 172246          215#
SIPAR4= 172250          216#
SIPAR5= 172252          217#
SIPAR6= 172254          218#
SIPAR7= 172256          219#
SIPDR0= 172200          190#
SIPDR1= 172202          191#
SIPDR2= 172204          192#
SIPDR3= 172206          193#
SIPDR4= 172210          194#
SIPDR5= 172212          195#
SIPDR6= 172214          196#
SIPDR7= 172216          197#
SKPMSG  041013         5290#
SLASH   041173         5312#
SP     -%000006          49#   596*  614*  622*  626   698   761*  772   788   833*  865   868   1346*
                       1347*  1360  2243*  2244*  2265  2622*  2623*  2632  2646*  2647*  2677*  2678*  2686
                       2691   3695*  3989*  3990*  3991  3998*  4001*  4002*  4006*  4007*  4011  4014*  4018
                       4020   4022  4023*  4030  4032  4034*  4035  4037*  4038*  4039*  4040*  4041*  4056*
                       4057*  4060*  4061*  4062  4066*  4067*  4068  4071  4073  4075*  4084*  4089  4100
                       4101*  4102*  4103*  4124*  4125*  4126*  4127*  4128*  4130  4133*  4141*  4142  4144
                       4145*  4147  4148  4149  4165*  4166*  4167*  4168*  4169*  4170*  4171  4174*  4187
                       4189*  4191  4201  4203  4205  4206  4207  4208  4209  4211*  4212*  4241*  4244
                       4246   4247  4276  4277  4281*  4305*  4306  4316*  4318  4319  4320*  4322  4324
                       4326   4332  4334*  4336*  4344*  4348  4352  4353  4357  4388*  4389  4390  4391*
                       4396*  4397*  4398*  4404  4429  4430  4431  4432*  4433*  4464  4485*  4488*  4505*
                       4510*  4531  4535*  4551*  4552*  4559  4560*  4571  4572*  4573*  4583  4584*  4589
                       4590   4606*  4607*  4608*  4609*  4610*  4611*  4612*  4613  4621*  4625  4626  4627
                       4628   4629  4630  4631  4659*  4660*  4661*  4662*  4663*  4664*  4665  4668  4678*
                       4682   4683*  4690  4710*  4711  4712  4715  4716  4717  4718  4742*  4743*  4744*
                       4745*  4746*  4747*  4748*  4749*  4750*  4751*  4758*  4759*  4760*  4761*  4762  4763
                       4764   4765  4766  4767  4784*  4785*  4786*  4787*  4788  4790*  4792  4794*  4796*
                       4802*  4804  4809  4810*  4811*  4812  4813  4814  4843*  4844*  4845*  4846*  4847*
                       4848*  4849*  4850  4851  4853*  4857  4860*  4863*  4874*  4878*  4881  4882  4885*
                       4886*  4887  4890*  4891*  4892  4893  4894  4895  4896*  4912  4914*  4971  4972*
                       4974*  4975*  4986*  4987  4997*  4998*  5218*  5222*  5564  5634*  5661  5733*  5734*
                       5736   5737  5739*  5740*  5741*  5748  5749  5939*  5940*  5942  5943  5944*  5945*
                       5946*  5952  5953  5996*  6000*  6006*  6008*  6010  6054*  6059*  6062*  6067*  6070*
                       6100*  6170
SR0     177572          139#  3467*  3469*  3494*  3496*  3519*  3521*  3546*  3548*  3572*  3574*  3592*  3594*
                       3638*  3640*  3658*  3660*  4674  4691*  4698*
SR1     = 177574        140#
SR2     = 177576        141#
SR3     = 172516        142#  4688*
STACK   = 001100         24#   596
STATE   041132         5306#  5999
STDAT0  045774         5831   5863  6141#
STKLMT= 177774           35#
SWR     001140          368#   594  616*  618  624*  631*  645  763  3328  3400  3502  3554  3985
                       4022*  4236  4250  4252  4258  4265  4460  4467  4479  4483  4612  4625*  5027
                       5032   5038  5043  5049  5054  5060  5065  5071  5139  5141  5169  5175  6082
```

```
SWREG   000176          300#    624     645     3985    3998
SW0     = 000001         88#
SW00    = 000001         78#     88     5032
SW01    = 000002         77#     87     5043
SW02    = 000004         76#     86
SW03    = 000010         75#     85
SW04    = 000020         74#     84
SW05    = 000040         73#     83
SW06    = 000100         72#     82     5027
SW07    = 000200         71#     81
SW08    = 000400         70#     80
SW09    = 001000         69#     79
SW1     = 000002         87#
SW10    = 002000         68#
SW11    = 004000         67#   3328     3400    3502    3554    6082
SW12    = 010000         66#
SW13    = 020000         65#   5141     5169
SW14    = 040000         64#
SW15    = 100000         63#   5175
SW2     = 000004         86#   5054
SW3     - 000010         85#   5065
SW4     = 000020         84#   5071
SW5     = 000040         83#
SW6     = 000100         82#
SW7     - 000200         81#   5038
SW8     = 000400         80#   5049
SW9     - 001000         79#   5060
TBITVE= 000014          121#
TESTER  002020          304     584#
TESTZ - 000002          578#   1144    1145    1170    1171    1198    1199    1221    1222    1237    1238    1256    1257
                       1291   1292    1305    1328    1329    1341    1378    1379    1657    1659    1674    1676    1691
                       1693   1708    1710    1725    1727    1742    1744    1759    1761    1776    1778    1793    1795
                       1810   1812    1827    1829    1844    1846    1861    1863    1878    1880    1895    1897    1912
                       1914   1929    1931    1946    1948    1963    1965    1980    1982    1997    1999    2014    2016
                       2032   2034    2050    2052    2071    2072    2127    2128    2176    2177    2210    2211    2252
                       2253   2284    2285    2310    2311    2461    2462    2501    2502    2529    2530    2576    2577
                       2616   2617    2659    2660    2705    2706    2726    2727    2747    2748    2768    2769    2789
                       2790   2810    2811    2831    2832    2852    2853    2873    2874    2894    2895    2915    2916
                       2936   2937    2960    2961    2994    3016    3017    3046    3047    3067    3068    3093    3094
                       3116   3117    3291    3362    3363    3431    3490    3491    3540    3541    3582    3583    3614
                       3615   3648    3649    5211    5875    6122    6450
TFSITE  003276          703     784#
TIMEO   041253         5322#   5654
TKVEC   = 000060         128#
TPVEC   = 000064         129#
TRAPVE= 000034           127#    602*    603*
TRTVEC= 000014           122#
TSTCON= 000004           577#   1372    1451    1468    1485    1501    1518    1535    1551    1568    1586    1603    1621
                       1638   1655    1672    1689    1706    1723    1740    1757    1774    1791    1808    1825    1842
                       1859   1876    1893    1910    1927    1944    1961    1978    1995    2012    2030    2048    2069
                       2123   2173    2208    2242    2281    2307    2332    2357    2384    2409    2434    2459    2497
                       2527   2574    2614    2656    2702    2723    2744    2765    2786    2807    2828    2849    2870
                       2891   2912    2933    2957    2992    3013    3043    3064    3090    3114    3289    3360    3429
                       3488   3538    3580    3611    3645
```

```
TSTDMA= 000010            576#  1104  1139  1165  1193  1216  1250  1285  1322  1372  1451  1468  1485
                         1501  1518  1535  1551  1568  1586  1603  1621  1638  1655  1672  1689  1706
                         1723  1740  1757  1774  1791  1808  1825  1842  1859  1876  1893  1910  1927
                         1944  1961  1978  1995  2012  2030  2048  2069  2123  2173  2208  2242  2281
                         2307  2332  2357  2384  2409  2434  2459  2497  2527  2574  2614  2656  2702
                         2723  2744  2765  2786  2807  2828  2849  2870  2891  2912  2933  2957  2992
                         3013  3043  3064  3090  3114  3149  3176  3202  3226  3249  3488  3538  3580
                         3611  3645
TST1      003636          801   841   847#
TST10     004510          969   975#
TST100    014642         2086  2109  2114#
TST101    015166         2138  2167#
TST102    015376         2189  2197  2202#
TST103    015600         2223  2231  2236#
TST104    016000         2273#
TST105    016146         2277  2293  2299#
TST106    016324         2303  2322  2327#
TST107    016456         2346  2352#
TST11     004546          981   987#
TST110    016610         2371  2377#
TST111    016744         2381  2396  2402#
TST112    017114         2406  2423  2429#
TST113    017246         2448  2453#
TST114    017404         2473  2480#
TST115    017612         2520#
TST116    020052         2560  2567#
TST117    020304         2602  2607#
TST12     004610          993   999#
TST120    020522         2644#
TST121    020776         2698#
TST122    021126         2713  2719#
TST123    021256         2734  2740#
TST124    021406         2755  2761#
TST125    021536         2776  2782#
TST126    021666         2797  2803#
TST127    022016         2818  2824#
TST13     004676         1016#
TST130    022146         2839  2845#
TST131    022276         2860  2866#
TST132    022426         2881  2887#
TST133    022556         2902  2908#
TST134    022706         2923  2929#
TST135    023036         2944  2950#
TST136    023224         2972  2977  2985#
TST137    023354         3003  3009#
TST14     004764         1033#
TST140    023542         3028  3032  3039#
TST141    023672         3054  3060#
TST142    024022         3075  3081#
TST143    024162         3088  3101  3107#
TST144    024366         3136  3141#
TST145    024534         3144  3162  3168#
TST146    024702         3171  3189  3195#
TST147    025026         3198  3212  3219#
```

```
TST64   013010          1867    1873#
TST65   013112          1884    1890#
TST66   013214          1901    1907#
TST67   013316          1918    1924#
TST7    004440           962#
TST70   013420          1935    1941#
TST71   013522          1952    1958#
TST72   013624          1969    1975#
TST73   013726          1986    1992#
TST74   014030          2003    2009#
TST75   014132          2020    2027#
TST76   014234          2038    2045#
TST77   014336          2056    2062#
TYPDS - 104405           834    3696    5013#   5997    6007    6055    6060    6063    6068    6071
TYPE  = 104401           639     691     693     696     754     760     766     770     784     785     804     825     835
                        3694    3697    3996    3997    4000    4013    4024    4043    4092    4095    4099    4210    4327
                        4423    4462    4470    4504    4521    4523    4526    4528    4532    4539    4634    5009#   5082
                        5088    5095    5101    5108    5114    5121    5127    5134    5174    5221    5506    5507    5546
                        5547    5548    5562    5565    5609    5628    5654    5795    5798    5813    5995    5998    5999
                        6005    6015    6056    6061    6064    6069    6072    6076    6077    6091    6092    6093    6097
                        6103    6322
TYPOC - 104402           762    3999    4512    4536    5010#
TYPON = 104404          5012#
TYPOS = 104403          5011#   5219    5223    6001    6101
UDPAR0= 177660           179#
UDPAR1= 177662           180#
UDPAR2= 177664           181#
UDPAR3= 177666           182#
UDPAR4= 177670           183#
UDPAR5= 177672           184#
UDPAR6= 177674           185#
UDPAR7= 177676           186#
UDPDR0= 177620           157#
UDPDR1= 177622           158#
UDPDR2= 177624           159#
UDPDR3= 177626           160#
UDPDR4= 177630           161#
UDPDR5= 177632           162#
UDPDR6= 177634           163#
UDPDR7= 177636           164#
UIPAR0= 177640           168#
UIPAR1= 177642           169#
UIPAR2= 177644           170#
UIPAR3= 177646           171#
UIPAR4= 177650           172#
UIPAR5= 177652           173#
UIPAR6= 177654           174#
UIPAR7= 177656           175#
UIPDR0= 177600           146#
UIPDR1= 177602           147#
UIPDR2= 177604           148#
UIPDR3= 177606           149#
UIPDR4= 177610           150#
UIPDR5= 177612           151#
```

```
UIPDR6= 177614          152#
UIPDR7= 177616          153#
UNFIX   041501         5350#  5546
VAL0    050120         6422*  6516#
VAL1    050122         6426*  6517#
VAL2    050124         6430*  6518#
VECTA0  001772          560#  1349*  1363*  2246*  2268*
VECTA1  001774          561#  1363   1364*  2268   2269*
VECTB0  001776          562#  2625*  2639*  2650*  2676*  2693*
VECTB1  002000          563#   675   2639   2640*  2651*  2693   2694*
VTCHAR  001706          524#  6323*  6408   6411   6413*
VTCSR   001716          528#  6328   6330*  6331   6333*  6356*  6359*  6500*  6502   6505*
VTCXY   001714          527#  6410*
VTINT   001724          531#  6357*  6358*
VTMAP   001720          529#  6501*
VTPX    001722          530#  6504*
VTXPOS  001712          526#  6496*
VTYPOS  001710          525#  6324*
WARN0   052604          770   6784#
WCR     001752          549#   853   1005*  1006   1055*  1066   1140*  1151   1166*  1179   1194*  1201   1217*
                       1224   1234*  1251*  1286*  1323*  1373*  2067*  2094   2121*  2150   2170*  2205*  2239*
                       2278*  2304*  2330*  2355*  2382*  2407*  2432*  2523*  2571*  2611*  2653*  3110*  3150*
                       3177*  3178*  3185   3203*  3227*  3250*  3287*  3358*  3427*  3536*  5150*  5872*  6116*
                       6447*
WFMODE  050110          584*   588*   689    704    717    732    737    802   3143   3170   3197   3221   3244
                       6184   6512#
WIDA    041764         5402#  5984*  6041   6065
WIDB    041766         5403#  5978*  6045   6066
WIDMSG  041111         5303#  6069
XTTY    003240          744    770#
ZAPRAM  042650         5538   5571#
ZP0     053120         5560   5627   6870#
$APTHD  001000          328    334#
$ASTAT= ****** U       4577   4592
$ATYC   035462         4548   4550#
$ATY1   035436         4546#
$ATY3   035444         4312   4547#
$ATY4   035454         4475   4549#
$AUTOB  001134          365#   649*   685    823   3993   4114   5793   5811   5989
$BASE   001250          429#   658    663    665   3965   3966   3968   3970
$BDADR  001122          360#  2107*  2156*  2157   2161*  2162   2709*  2711   2730*  2732   2751*  2753   2772*
                       2774   2793*  2795   2814*  2816   2835*  2837   2856*  2858   2877*  2879   2898*  2900
                       2919*  2921   2940*  2942   2968*  2970   2999*  3001   3050*  3052   3071*  3073   3097*
                       3099   3309*  3316*  3380*  3387*  3444*  3481*  3542*  3618*  3620   3968   3970
$BDDAT  001126          362#   886*   889    905*   906    918*   919    931*   932    938*   939    950*   952
                        967*   968    980*   992*  1006*  1007   1023*  1024   1040*  1041   1061*  1062   1066*
                       1067   1071*  1072   1076*  1077   1081*  1082   1095*  1112*  1113   1120*  1121   1129*
                       1130   1147*  1151*  1155*  1156   1173*  1174   1179*  1183*  1184   1201*  1206*  1207
                       1224*  1225   1240*  1275*  1276   1312*  1313   1355*  1380*  1382   1388*  1404*  1410*
                       1411   1427*  1432*  1434   1439*  1441*  1456*  1458   1473*  1475   1490*  1492   1506*
                       1508   1523*  1525   1540*  1542   1556*  1558   1573*  1575   1591*  1593   1608*  1610
                       1626*  1628   1643*  1645   1660*  1662   1677*  1679   1694*  1696   1711*  1713   1728*
                       1730   1745*  1747   1762*  1764   1779*  1781   1796*  1798   1813*  1815   1830*  1832
                       1847*  1849   1864*  1866   1881*  1883   1898*  1900   1915*  1917   1932*  1934   1949*
```

```
$MSGAD  001210          396#  4563*  4566
$MSGLG  001212          397#  4568*
$MSGTY  001174          390#  4561   4569*  4581   4585*
$MSWR   033536         3997   4110#
$MTYP1  001225          411#
$MTYP2  001231          419#
$MTYP3  001235          422#
$MTYP4  001241          425#
$MULT   036504         4783#  5735   5941
$MXCNT  034370         4273   4283#
$NULL   001154          374#  4334   4363
$NWTST= 000001          844#   877#   897#   910#   923#   942#   959#   972#   984#   996#  1013#  1030#  1047#
                       1086#  1099#  1133#  1159#  1187#  1210#  1228#  1244#  1279#  1316#  1366#  1392#  1415#
                       1445#  1462#  1479#  1495#  1512#  1529#  1545#  1562#  1580#  1597#  1615#  1632#  1649#
                       1666#  1683#  1700#  1717#  1734#  1751#  1768#  1785#  1802#  1819#  1836#  1853#  1870#
                       1887#  1904#  1921#  1938#  1955#  1972#  1989#  2006#  2024#  2042#  2059#  2111#  2164#
                       2199#  2233#  2270#  2296#  2324#  2349#  2374#  2399#  2426#  2450#  2477#  2517#  2564#
                       2604#  2641#  2695#  2716#  2737#  2758#  2779#  2800#  2821#  2842#  2863#  2884#  2905#
                       2926#  2947#  2982#  3006#  3036#  3057#  3078#  3104#  3138#  3165#  3192#  3216#  3239#
                       3262#  3337#  3409#  3454#  3507#  3559#  3598#  3625#  3665#
$OCNT   035076         4395*  4424*  4437#
$OMODE  035100         4390*  4394*  4399   4402*  4413*  4439#
$OVER   034354         4237   4253   4261   4271   4280#
$PASS   001202          393#   628*   800   3326   3398   3500   3552   3686*  3687*  3695   3708   4267   4284
$PASTM  001006          338#
$POWER  036052         4635   4640#
$PWRDN  035704          604   4604#  4632
$PWRMG  036040         4635#
$PWRUP  035756         4614   4620#
$QUES   001170          387#   755   4043   4092   4108   4363   4495
$RDCHR  033266         4056#  5018
$RDDEC= ****** U       5021
$RDLIN  033406         4084#  5019
$RDOCT  033560         4124#  5020
$RDSZ = 000010         4077#
$RESRE  036446         4757#  5022
$RTNAD  030176         3707#
$R2A  = ****** U       5023
$SAVRE  036410         4741#  5021
$SAVR6  036050         4613*  4621   4622*  4623*  4639#
$SB2D   037232         4971#  6009
$SCOPE  034106          598   4233#
$SETUP= 000137          582#   597    598    600    602    604    606    607    608    610    637    640   3684
                       3980   4114   4234   4456   4482   4490
$SIZE   036062          806    813   4659#
$SIZEX  036340         4699   4709#
$STUP = 177777          582#
$SVLAD  034320         4245   4274#
$SVPC = 000106          312#   317
$SWR  = 167400            4#    21    285    286    287    288    289    290    291    378    379    380    607
                        608    610    611    848    881    901    914    927    946    963    976    988   1000
                       1017   1034   1051   1090   1103   1137   1163   1191   1214   1232   1248   1283   1320
                       1370   1396   1419   1449   1466   1483   1499   1516   1533   1549   1566   1584   1601
                       1619   1636   1653   1670   1687   1704   1721   1738   1755   1772   1789   1806   1823
```

```
COMMEN   131#
ENDCOM   131#
ERROR     25#  866  869  891  908  921  934  941  954  970  982  994 1009 1026 1043
          1064 1069 1074 1079 1084 1097 1116 1123 1132 1149 1153 1158 1176 1181 1186
          1203 1209 1227 1242 1269 1278 1306 1315 1342 1357 1384 1390 1406 1414 1429
          1436 1443 1460 1477 1494 1510 1527 1544 1560 1577 1595 1612 1630 1647 1664
          1681 1698 1715 1732 1749 1766 1783 1800 1817 1834 1851 1868 1885 1902 1919
          1936 1953 1970 1987 2004 2021 2039 2057 2085 2091 2098 2103 2110 2137 2142
          2148 2154 2160 2188 2192 2198 2222 2226 2232 2262 2294 2323 2347 2372 2397
          2424 2449 2474 2508 2513 2544 2554 2561 2590 2597 2603 2630 2637 2682 2687
          2714 2735 2756 2777 2798 2819 2840 2861 2882 2903 2924 2945 2975 2980 3004
          3031 3035 3055 3076 3102 3131 3137 3161 3188 3213 3237 3260 3304 3314 3375
          3385 3392 3441 3449 3499 3551 3597 3623 3664
ESCAPE   131#
GETPRI   131#  4667 4841
GETSWR   131#  640#
JOYMAC  1444#  1450 1467 1484 1500 1517 1534 1550 1567 1585 1602 1620 1637
MULT     131#
NEWTST   131#  844  877  897  910  923  942  959  972  984  996 1013 1030 1047 1086
          1099 1133 1159 1187 1210 1228 1244 1279 1316 1366 1392 1415 1445 1462 1479
          1495 1512 1529 1545 1562 1580 1597 1615 1632 1649 1666 1683 1700 1717 1734
          1751 1768 1785 1802 1819 1836 1853 1870 1887 1904 1921 1938 1955 1972 1989
          2006 2024 2042 2059 2111 2164 2199 2233 2270 2296 2324 2349 2374 2399 2426
          2450 2477 2517 2564 2604 2641 2695 2716 2737 2758 2779 2800 2821 2842 2863
          2884 2905 2926 2947 2982 3006 3036 3057 3078 3104 3138 3165 3192 3216 3239
          3262 3337 3409 3454 3507 3559 3598 3625 3665
OFFBIT  2695#  2716 2737 2758 2779 2800 2821 2842 2863 2884 2905 2926
POP      131#  4147 4205 4589 4590 4625 4626 4762 4812 4892
PUSH     131#  4126 4164 4550 4552 4573 4606 4612 4742 4783 4844
REPORT   131#
RESLM   1649#  1666 1683 1700 1717 1734 1751 1768 1785 1802 1819 1836 1853 1870 1887
          1904 1921 1938 1955 1972 1989
SCOPE     26#  847  880  900  913  926  945  962  975  987  999 1016 1033 1050 1089
          1102 1136 1162 1190 1213 1231 1247 1282 1319 1369 1395 1418 1448 1465 1482
          1498 1515 1532 1548 1565 1583 1600 1618 1635 1652 1669 1686 1703 1720 1737
          1754 1771 1788 1805 1822 1839 1856 1873 1890 1907 1924 1941 1958 1975 1992
          2009 2027 2045 2062 2114 2167 2202 2236 2273 2299 2327 2352 2377 2402 2429
          2453 2480 2520 2567 2607 2644 2698 2719 2740 2761 2782 2803 2824 2845 2866
          2887 2908 2929 2950 2985 3009 3039 3060 3081 3107 3141 3168 3195 3219 3242
          3265 3340 3412 3457 3510 3562 3601 3628 3668 3683
SETPRI   131#
SETTRA  5001#  5010 5011 5012 5013 5015 5017 5018 5019 5020 5021 5022
SETUP    131#  589
SETZ     582#  1144 1170 1198 1221 1237 1256 1291 1328 1378 2071 2126 2176 2210 2252
          2284 2310 2461 2501 2529 2576 2616 2659 2705 2726 2747 2768 2789 2810 2831
          2852 2873 2894 2915 2936 2960 3016 3046 3067 3093 3116 3362 3490 3540 3582
          3614 3648
SKIP     131#  801  824  841  864  872  890  907  920  933  940  953  969  981  993
          1008 1025 1042 1063 1068 1073 1078 1083 1096 1114 1122 1131 1148 1150 1152
          1157 1177 1180 1185 1204 1208 1226 1241 1270 1277 1307 1314 1343 1358 1383
          1385 1389 1412 1435 1442 1459 1476 1493 1509 1526 1543 1559 1576 1594 1611
          1629 1646 1663 1680 1697 1714 1731 1748 1765 1782 1799 1816 1833 1850 1867
          1884 1901 1918 1935 1952 1969 1986 2003 2020 2038 2056 2084 2086 2090 2097
          2102 2109 2136 2138 2141 2147 2153 2159 2163 2189 2191 2197 2223 2225 2231
```

```
                2263    2277    2293    2303    2322    2346    2371    2381    2396    2406    2423    2448    2473    2507    2543
                2553    2560    2589    2596    2602    2631    2636    2683    2688    2713    2734    2755    2776    2797    2818
                2839    2860    2881    2902    2923    2944    2972    2977    3003    3028    3030    3032    3054    3075    3088
                3101    3130    3136    3144    3160    3162    3171    3187    3189    3198    3212    3222    3236    3245    3259
                3297    3305    3308    3313    3315    3320    3327    3329    3332    3368    3376    3379    3384    3386    3391
                3393    3399    3401    3404    3434    3442    3448    3450    3460    3498    3501    3503    3513    3550    3553
                3555    3568    3596    3609    3622    3634    3663    3671
SLASH            131#
SOFTSW             8#
SPACE            131#
STARS            131#     310     321     323     330     343     384     387     844     846     877     879     897     899     910
                 912      923     925     942     944     959     961     972     974     984     986     996     998    1013    1015
                1030    1032    1047    1049    1086    1088    1099    1101    1133    1135    1159    1161    1187    1189    1210
                1212    1228    1230    1244    1246    1279    1281    1316    1318    1366    1368    1392    1394    1415    1417
                1445    1447    1462    1464    1479    1481    1495    1497    1512    1514    1529    1531    1545    1547    1562
                1564    1580    1582    1597    1599    1615    1617    1632    1634    1649    1651    1666    1668    1683    1685
                1700    1702    1717    1719    1734    1736    1751    1753    1768    1770    1785    1787    1802    1804    1819
                1821    1836    1838    1853    1855    1870    1872    1887    1889    1904    1906    1921    1923    1938    1940
                1955    1957    1972    1974    1989    1991    2006    2008    2024    2026    2042    2044    2059    2061    2111
                2113    2164    2166    2199    2201    2233    2235    2270    2272    2296    2298    2324    2326    2349    2351
                2374    2376    2399    2401    2426    2428    2450    2452    2477    2479    2517    2519    2564    2566    2604
                2606    2641    2643    2695    2697    2716    2718    2737    2739    2758    2760    2779    2781    2800    2802
                2821    2823    2842    2844    2863    2865    2884    2886    2905    2907    2926    2928    2947    2949    2982
                2984    3006    3008    3036    3038    3057    3059    3078    3080    3104    3106    3138    3140    3165    3167
                3192    3194    3216    3218    3239    3241    3262    3264    3337    3339    3409    3411    3454    3456    3507
                3509    3559    3561    3598    3600    3625    3627    3665    3667    3675    3977    3980    4048    4077    4116
                4154    4221    4286    4365    4443    4498    4545    4602    4618    4645    4726    4771    4818    4900    4962
                4980
SWRSU            131#    612#
TRMTRP          5001#
TYPBIN           131#
TYPDEC           131#    3695
TYPNAM           131#     633
TYPNUM           131#
TYPOCS           131#
TYPOCT           131#    3998    4510    4534
TYPTXT           131#
$$CMRE           341#
$$CMTM           341#
$$ESCA           131#
$$NEWT           131#     844     877     897     910     923     942     959     972     984     996    1013    1030    1047    1086
                1099    1133    1159    1187    1210    1228    1244    1279    1316    1366    1392    1415    1445    1462    1479
                1495    1512    1529    1545    1562    1580    1597    1615    1632    1649    1666    1683    1700    1717    1734
                1751    1768    1785    1802    1819    1836    1853    1870    1887    1904    1921    1938    1955    1972    1989
                2006    2024    2042    2059    2111    2164    2199    2233    2270    2296    2324    2349    2374    2399    2426
                2450    2477    2517    2564    2604    2641    2695    2716    2737    2758    2779    2800    2821    2842    2863
                2884    2905    2926    2947    2982    3006    3036    3057    3078    3104    3138    3165    3192    3216    3239
                3262    3337    3409    3454    3507    3559    3598    3625    3665
$$SET           5001#    5010    5011    5012    5013    5015    5017    5018    5019    5020    5021    5022
$$SETM           628#
$$SKIP           131#     801     841     864     872     907     920     940     969     981     993    1096    1131    1150    1157
                1177    1185    1204    1208    1226    1241    1270    1277    1307    1314    1385    1389    1412    1442    1459
                1476    1493    1509    1526    1543    1559    1576    1594    1611    1629    1646    1663    1680    1697    1714
                1731    1748    1765    1782    1799    1816    1833    1850    1867    1884    1901    1918    1935    1952    1969
```

	2691	2712	2733	2754	2775	2796	2817	2838	2859	2880	2901	2922	2943	2971	3002
	3027	3053	3074	3087	3100	3129	3135	3159	3186	3211	3235	3258	3281	3307	3312
	3319	3323	3331	3354	3378	3383	3390	3395	3403	3423	3447	3452	3497	3505	3549
	3557	3567	3595	3608	3621	3633	3662	3985	3991	4011	4018	4030	4032	4062	4068
	4071	4073	4086	4196	4246	4270	4490	4690	4695	4706	4804	4930	5444	5446	5448
	5518	5593	5659	5685	5782	5815	5824	5828	5837	5844	5851	5893	5902	5924	5956
	6019	6089	6163	6172	6339	6345	6351	6444	6466	6472					
CMPB	643	701	706	709	713	719	722	726	729	734	739	742	745	749	790
	3993	4025	4090	4096	4252	4256	4307	4322	4324	4332	4353	4357	4472	4555	5962
	5964	5966	5968	5974	5980	5987	6003	6038	6042	6046	6048	6433			
COM	5935														
COMB	5574														
DEC	1264	1266	1300	1302	1336	1338	1402	1425	2130	2500	3152	3178	3298	3300	3369
	3371	3435	3437	3688	4514	4794	4802	4874	4886	4933	5164	5167	5189	5433	5542
	5648	5650	6104	6134											
DECB	4336	4339	4413	4424	4853										
EMT	25														
HALT	298	765	4303	4481	4492	4615	4637	4722	5177	5744	5745	5949			
INC	582	584	635	664	680	832	842	3686	4040	4182	4269	4419	4427	4463	4585
	4623	4691	4790	4863	4925	5426	5558	5625	5901	5958	5970	5972	5976	5978	5982
	5984	5986	5994	6018	6053	6338	6344	6350							
INCB	4274	4359	4457												
IOT	26														
JMP	302	303	304	688	703	708	712	716	721	725	728	731	736	741	744
	752	767	781	793	874	3672	3706	5137	5453	5540	5549	5566	5629	5655	5817
	5818	6021	6110	6171	6404										
JSR	806	813	3701	4029	4235	4312	4331	4338	4345	4469	4475	4574	4973	5025	5029
	5034	5040	5045	5051	5056	5062	5067	5079	5083	5085	5089	5092	5096	5098	5102
	5105	5109	5111	5115	5118	5122	5124	5128	5131	5135	5166	5509	5511	5513	5521
	5526	5528	5533	5538	5541	5552	5554	5556	5559	5596	5599	5601	5603	5611	5613
	5615	5617	5619	5621	5623	5626	5677	5679	5681	5735	5742	5800	5807	5831	5836
	5843	5850	5863	5923	5941	5947	6009	6017	6088	6142	6337	6343	6349	6363	6368
	6373	6378	6384	6389	6394	6399	6421	6423	6425	6427	6429	6431	6432		
MOV	592	596	598	599	600	601	602	603	604	605	606	610	611	614	615
	616	617	622	624	625	626	631	656	657	658	659	663	665	667	668
	671	673	677	678	679	687	698	715	751	761	772	776	777	788	792
	811	812	816	817	818	819	820	821	822	826	833	836	837	840	848
	849	850	858	862	863	870	871	873	881	882	883	885	886	887	901
	902	904	905	914	915	916	917	918	927	928	929	931	935	936	938
	946	947	949	950	951	963	964	965	967	976	977	980	988	989	992
	1000	1001	1002	1004	1005	1006	1017	1018	1019	1021	1022	1023	1034	1035	1036
	1038	1039	1040	1051	1053	1054	1055	1056	1057	1058	1060	1061	1065	1066	1070
	1071	1075	1076	1080	1081	1085	1090	1092	1093	1094	1095	1103	1110	1112	1119
	1120	1126	1129	1137	1138	1139	1140	1141	1142	1146	1147	1151	1154	1155	1163
	1164	1165	1166	1167	1168	1172	1173	1178	1179	1182	1183	1191	1192	1193	1194
	1195	1196	1200	1201	1205	1206	1214	1215	1216	1217	1218	1219	1223	1224	1232
	1233	1234	1235	1239	1240	1248	1249	1250	1251	1252	1253	1258	1260	1268	1274
	1275	1283	1284	1285	1286	1287	1288	1294	1296	1304	1311	1312	1320	1321	1322
	1323	1324	1325	1330	1332	1340	1346	1347	1349	1355	1356	1362	1363	1370	1371
	1372	1373	1374	1376	1380	1381	1388	1396	1397	1399	1404	1405	1410	1419	1420
	1422	1427	1428	1432	1433	1439	1440	1441	1449	1450	1451	1452	1456	1457	1466
	1467	1468	1469	1473	1474	1483	1484	1485	1486	1490	1491	1499	1500	1501	1502
	1506	1507	1516	1517	1518	1519	1523	1524	1533	1534	1535	1536	1540	1541	1549
	1550	1551	1552	1556	1557	1566	1567	1568	1569	1573	1574	1584	1585	1586	1587

1591	1592	1601	1602	1603	1604	1608	1609	1619	1620	1621	1622	1626	1627	1636	
1637	1638	1639	1643	1644	1653	1654	1655	1656	1660	1661	1670	1671	1672	1673	
1677	1678	1687	1688	1689	1690	1694	1695	1704	1705	1706	1707	1711	1712	1721	
1722	1723	1724	1728	1729	1738	1739	1740	1741	1745	1746	1755	1756	1757	1758	
1762	1763	1772	1773	1774	1775	1779	1780	1789	1790	1791	1792	1796	1797	1806	
1807	1808	1809	1813	1814	1823	1824	1825	1826	1830	1831	1840	1841	1842	1843	
1847	1848	1857	1858	1859	1860	1864	1865	1874	1875	1876	1877	1881	1882	1891	
1892	1893	1894	1898	1899	1908	1909	1910	1911	1915	1916	1925	1926	1927	1928	
1932	1933	1942	1943	1944	1945	1949	1950	1959	1960	1961	1962	1966	1967	1976	
1977	1978	1979	1983	1984	1993	1994	1995	1996	2000	2001	2010	2011	2012	2013	
2017	2018	2028	2029	2030	2031	2035	2036	2046	2047	2048	2049	2053	2054	2063	
2064	2065	2066	2067	2068	2069	2080	2081	2082	2087	2088	2094	2095	2101	2105	
2106	2107	2115	2116	2117	2118	2121	2122	2123	2125	2133	2134	2140	2143	2144	
2145	2150	2151	2155	2156	2157	2168	2169	2170	2171	2172	2173	2174	2185	2186	
2194	2195	2203	2204	2205	2206	2207	2208	2219	2220	2228	2229	2237	2238	2239	
2240	2241	2242	2243	2244	2246	2267	2268	2274	2275	2278	2279	2281	2282	2290	
2291	2300	2301	2304	2305	2306	2307	2308	2319	2320	2328	2329	2330	2331	2332	
2333	2343	2344	2353	2354	2355	2356	2357	2358	2368	2369	2378	2379	2382	2383	
2384	2385	2393	2394	2403	2404	2407	2408	2409	2410	2413	2414	2420	2421	2430	
2431	2432	2433	2434	2435	2445	2446	2454	2455	2457	2458	2459	2470	2471	2481	
2491	2492	2494	2495	2496	2497	2499	2504	2505	2509	2512	2521	2522	2525	2526	
2527	2540	2541	2550	2551	2557	2558	2568	2569	2572	2573	2574	2581	2586	2587	
2593	2594	2599	2600	2608	2609	2612	2613	2614	2621	2622	2623	2625	2633	2634	
2638	2639	2645	2646	2647	2649	2650	2651	2654	2655	2656	2657	2671	2676	2677	
2678	2692	2693	2699	2700	2701	2702	2703	2709	2710	2711	2720	2721	2722	2723	
2724	2730	2731	2732	2741	2742	2743	2744	2745	2751	2752	2753	2762	2763	2764	
2765	2766	2772	2773	2774	2783	2784	2785	2786	2787	2793	2794	2795	2804	2805	
2806	2807	2808	2814	2815	2816	2825	2826	2827	2828	2829	2835	2836	2837	2846	
2847	2848	2849	2850	2856	2857	2858	2867	2868	2869	2870	2871	2877	2878	2879	
2888	2889	2890	2891	2892	2898	2899	2900	2909	2910	2911	2912	2913	2919	2920	
2921	2930	2931	2932	2933	2934	2940	2941	2942	2951	2955	2956	2957	2958	2968	
2969	2970	2978	2986	2990	2991	2992	2993	2999	3000	3001	3010	3011	3012	3013	
3014	3025	3026	3033	3040	3041	3042	3043	3044	3050	3051	3052	3061	3062	3063	
3064	3065	3071	3072	3073	3082	3086	3089	3090	3091	3097	3098	3099	3108	3109	
3112	3113	3114	3127	3128	3133	3134	3142	3145	3146	3148	3149	3150	3151	3153	
3158	3169	3172	3173	3175	3176	3177	3179	3180	3185	3196	3199	3201	3202	3203	
3204	3205	3210	3220	3223	3225	3226	3227	3228	3229	3234	3243	3246	3248	3249	
3250	3251	3252	3257	3266	3273	3274	3275	3277	3278	3279	3280	3283	3285	3286	
3287	3289	3290	3294	3302	3303	3306	3309	3310	3311	3316	3317	3318	3333	3341	
3346	3347	3348	3350	3351	3352	3353	3357	3358	3359	3360	3365	3373	3374	3377	
3380	3381	3382	3387	3388	3389	3405	3413	3414	3419	3420	3421	3422	3426	3427	
3428	3429	3432	3439	3440	3443	3444	3445	3446	3458	3463	3464	3465	3466	3468	
3470	3480	3481	3485	3486	3487	3488	3492	3495	3511	3515	3516	3517	3518	3520	
3522	3534	3535	3536	3537	3538	3542	3543	3545	3547	3563	3569	3570	3571	3577	
3578	3579	3580	3590	3591	3593	3602	3607	3610	3611	3612	3618	3619	3620	3629	
3635	3636	3637	3643	3644	3645	3646	3656	3657	3669	3691	3695	3698	3998	4022	
4027	4056	4057	4084	4085	4100	4101	4102	4103	4124	4125	4126	4127	4128	4130	
4145	4146	4147	4148	4149	4165	4166	4167	4168	4169	4170	4171	4176	4179	4199	
4205	4206	4207	4208	4209	4211	4212	4241	4242	4244	4247	4260	4272	4273	4276	
4277	4280	4281	4305	4306	4311	4319	4334	4388	4396	4397	4398	4404	4411	4429	
4430	4431	4432	4433	4459	4464	4485	4488	4505	4510	4519	4524	4529	4531	4535	
4551	4552	4559	4563	4568	4569	4571	4573	4583	4589	4590	4604	4605	4606	4607	
4608	4609	4610	4611	4612	4613	4614	4620	4621	4625	4626	4627	4628	4629	4630	
4631	4632	4633	4659	4660	4661	4662	4663	4664	4665	4668	4669	4673	4676	4677	

```
        4679  4680  4681  4682  4685  4687  4688  4692  4697  4701  4710  4711  4712  4713  4714
        4715  4716  4717  4718  4742  4743  4744  4745  4746  4747  4748  4749  4750  4751  4758
        4759  4760  4761  4762  4763  4764  4765  4766  4767  4784  4785  4786  4788  4792  4796
        4810  4811  4812  4813  4814  4844  4845  4846  4847  4849  4850  4851  4857  4861  4869
        4872  4880  4890  4891  4892  4893  4894  4895  4896  4912  4913  4914  4915  4916  4917
        4918  4919  4971  4972  4975  4986  4987  4991  4997  4998  5076  5146  5147  5148  5149
        5150  5155  5158  5179  5182  5200  5203  5214  5218  5222  5425  5432  5435  5436  5442
        5451  5454  5455  5516  5520  5525  5527  5532  5535  5536  5537  5564  5571  5572  5634
        5635  5636  5638  5657  5661  5672  5673  5674  5707  5708  5709  5711  5715  5733  5734
        5736  5737  5739  5740  5741  5748  5749  5771  5781  5799  5803  5806  5810  5822  5826
        5833  5834  5841  5848  5855  5858  5861  5866  5868  5869  5870  5871  5874  5884  5885
        5892  5896  5922  5929  5930  5931  5932  5939  5940  5942  5943  5944  5945  5946  5952
        5953  5954  5996  6000  6006  6008  6010  6036  6040  6044  6050  6052  6054  6057  6059
        6062  6065  6067  6070  6087  6096  6100  6114  6116  6117  6118  6127  6129  6130  6141
        6161  6183  6186  6189  6190  6191  6192  6193  6323  6335  6336  6342  6348  6356  6357
        6358  6407  6410  6417  6418  6420  6422  6424  6426  6428  6430  6441  6442  6447  6448
        6456  6460  6463  6469  6501  6504
MOVB     609   649   655   700  2534  2535  3121  3122  3659  3989  4006  4060  4066  4089  4094
        4133  4174  4177  4191  4194  4203  4275  4279  4316  4344  4352  4389  4390  4393  4394
        4395  4399  4402  4403  4422  4466  4474  4546  4547  4549  4932  4989  5171  5173  5183
        5184  5202  5215  5517  5524  5531  5573  5592  5637  5643  5645  5676  5678  5680  5701
        5753  5766  5792  5805  5887  5936  5938  6012  6013  6014  6098  6413  6449  6464  6470
        6496
NEG     3416  4173  4400  4791  4795  4806  4807  4854  4855  4865  4884  4889  5867
NOP     1105  1106  1107  1351  1352  1353  1354  2073  2074  2075  2077  2078  2079  2178  2179
        2180  2182  2183  2184  2212  2213  2214  2216  2217  2218  2247  2248  2249  2250  2254
        2255  2256  2258  2259  2260  2287  2288  2289  2312  2313  2314  2316  2317  2318  2335
        2336  2337  2340  2341  2342  2360  2361  2362  2365  2366  2367  2387  2388  2391  2392
        2418  2419  2437  2438  2439  2442  2443  2444  2463  2464  2465  2467  2468  2469  2531
        2532  2533  2537  2538  2539  2578  2579  2580  2583  2584  2585  2618  2619  2620  2627
        2628  2629  2661  2662  2663  2665  2666  2667  2668  2669  2670  2680  2681  2965  2966
        2967  3022  3023  3024  3118  3119  3120  3124  3125  3126  3584  3585  3587  3588  3589
        3650  3651  3653  3654  3655  3702  3703  3704  5151  5152  5153  5154  5156  5157  5160
        5193  5194  5197  5198  5199  5204  5205  5206  5207  5225  5226  5227  5422  5423  5424
        5656  5683  5684  5702  5703  5716  5717  5718  5719  5720  5721  5722  5723  5724  5725
        5726  5727  5728  5729  5730  5731  5746  5747  5754  5756  5757  5758  5759  5761  5762
        5763  5764  5765  5772  5773  5774  5775  5777  5778  5779  5780  5950  5951  5959  6166
        6167  6168  6474  6475  6477  6479  6481  6483  6485  6487  6492  6493  6494  6879
RESET    799   903   978  1438  3700  6321
ROL     3477  3479  3529  3531  4136  4138  4140  4406  4408  4409  4410  4412  4868  4873  5577
        5579  5581  5583  5585  5587  5589  5591
ROR     4800  4801  5576  5578  5580  5582  5584  5586  5588  5590  5860  5905  5906  5908  5909
        5911  5912  5914  5915  5917  5918  5920  5921
RTI      623   843  1348  2245  2624  2648  2679  4028  4076  4104  4150  4213  4282  4321  4434
        4494  4636  4752  4768  4897  4999
RTS     4361  4533  4591  4719  4815  4938  4976  4992  5144  5208  5228  5550  5606  5663  5687
        5704  5785  6106  6139  6152  6175  6194  6436  6488  6497  6506
SBC     4808  4856  4922
SOB     4684
SUB      807   810  2672  3415  4180  4465  4566  4708  4709  4921  4923  5713  5714  5760  5933
SWAB     669
TRAP    4667  4842  5001  5010  5011  5012  5013  5015  5017  5018  5019  5020  5021  5022
TST      641   674   683   689   704   717   732   737   763   774   778   800   802   808   814
         851   852   853   854   855   856   857   859   860  1409  2380  2405  2673  2973  3029
```

```
          1499  1500  1510  1513  1514  1515  1516  1517  1527  1530  1531  1532  1533  1534  1544
          1546  1547  1548  1549  1550  1560  1563  1564  1565  1566  1567  1577  1581  1582  1583
          1584  1585  1595  1598  1599  1600  1601  1602  1612  1616  1617  1618  1619  1620  1630
          1633  1634  1635  1636  1637  1647  1650  1651  1652  1653  1654  1664  1667  1668  1669
          1670  1671  1681  1684  1685  1686  1687  1688  1698  1701  1702  1703  1704  1705  1715
          1718  1719  1720  1721  1722  1732  1735  1736  1737  1738  1739  1749  1752  1753  1754
          1755  1756  1766  1769  1770  1771  1772  1773  1783  1786  1787  1788  1789  1790  1800
          1803  1804  1805  1806  1807  1817  1820  1821  1822  1823  1824  1834  1837  1838  1839
          1840  1841  1851  1854  1855  1856  1857  1858  1868  1871  1872  1873  1874  1875  1885
          1888  1889  1890  1891  1892  1902  1905  1906  1907  1908  1909  1919  1922  1923  1924
          1925  1926  1936  1939  1940  1941  1942  1943  1953  1956  1957  1958  1959  1960  1970
          1973  1974  1975  1976  1977  1987  1990  1991  1992  1993  1994  2004  2007  2008  2009
          2010  2011  2021  2025  2026  2027  2028  2029  2039  2043  2044  2045  2046  2047  2057
          2060  2061  2062  2063  2064  2085  2087  2091  2098  2103  2110  2112  2113  2114  2115
          2116  2137  2139  2142  2148  2154  2160  2164  2165  2166  2167  2168  2169  2190  2192
          2198  2200  2201  2202  2203  2204  2224  2226  2232  2234  2235  2236  2237  2238  2264
          2271  2272  2273  2274  2275  2278  2294  2297  2298  2299  2300  2301  2304  2323  2325
          2326  2327  2328  2329  2347  2350  2351  2352  2353  2354  2372  2375  2376  2377  2378
          2379  2382  2397  2400  2401  2402  2403  2404  2407  2424  2427  2428  2429  2430  2431
          2449  2451  2452  2453  2454  2455  2474  2478  2479  2480  2481  2482  2508  2518  2519
          2520  2521  2522  2544  2554  2561  2565  2566  2567  2568  2569  2590  2597  2603  2605
          2606  2607  2608  2609  2632  2637  2642  2643  2644  2645  2646  2684  2689  2696  2697
          2698  2699  2700  2714  2717  2718  2719  2720  2721  2735  2738  2739  2740  2741  2742
          2756  2759  2760  2761  2762  2763  2777  2780  2781  2782  2783  2784  2798  2801  2802
          2803  2804  2805  2819  2822  2823  2824  2825  2826  2840  2843  2844  2845  2846  2847
          2861  2864  2865  2866  2867  2868  2882  2885  2886  2887  2888  2889  2903  2906  2907
          2908  2909  2910  2924  2927  2928  2929  2930  2931  2945  2948  2949  2950  2951  2952
          2973  2978  2983  2984  2985  2986  2987  3004  3007  3008  3009  3010  3011  3029  3031
          3033  3037  3038  3039  3040  3041  3055  3058  3059  3060  3061  3062  3076  3079  3080
          3081  3082  3083  3089  3102  3105  3106  3107  3108  3109  3131  3137  3139  3140  3141
          3142  3143  3145  3161  3163  3166  3167  3168  3169  3170  3172  3188  3190  3193  3194
          3195  3196  3197  3199  3213  3217  3218  3219  3220  3221  3223  3237  3240  3241  3242
          3243  3244  3246  3260  3263  3264  3265  3266  3267  3298  3306  3309  3314  3316  3321
          3328  3330  3333  3338  3339  3340  3341  3342  3369  3377  3380  3385  3387  3392  3394
          3400  3402  3405  3410  3411  3412  3413  3414  3435  3443  3449  3451  3455  3456  3457
          3458  3459  3461  3499  3502  3504  3508  3509  3510  3511  3512  3514  3551  3554  3556
          3560  3561  3562  3563  3564  3569  3597  3599  3600  3601  3602  3603  3610  3623  3626
          3627  3628  3629  3630  3635  3664  3665  3666  3667  3668  3669  3670  3672  3676  3678
          3679  3681  3684  3690  3693  3694  3698  3700  3706  3708  3709  3712  3975  3978  3979
          3981  4009  4045  4049  4077  4078  4085  4087  4090  4092  4108  4114  4117  4119  4152
          4155  4222  4225  4230  4236  4238  4249  4252  4253  4254  4256  4258  4265  4269  4274
          4276  4280  4283  4284  4287  4316  4366  4444  4447  4457  4464  4469  4470  4471  4479
          4490  4494  4495  4499  4514  4543  4546  4547  4550  4577  4592  4603  4612  4613  4619
          4625  4626  4636  4643  4646  4669  4678  4722  4727  4772  4819  4843  4901  4963  4981
          4987  4990  5009  5010  5011  5012  5013  5014  5015  5016  5017  5018  5019  5020  5021
          5022  5023

.EQUIV      25    26    34    79    80    81    82    83    84    85    86    87    88   107   108
           109   110   111   112   113   114   115   116

.EVEN       386   653  3964  4542  4595  4642  5384  6318  6759  6872

.IF          12    23    89   117   143   154   176   187   198   220   242   264   288   289   290
           291   292   301   310   313   315   321   323   3??   343   347   349   378   379   380
           384   385   387   410   418   421   424   427   428   429   430   431   432   434   582
           591   596   598   600   602   604   606   607   608   610   628   636   637   638   640
           643   652   801   824   841   844   846   848   849   864   872   877   879   881   882
```

	879	880	881	890	898	899	900	901	908	911	912	913	914	921	924
	925	926	927	933	941	943	944	945	946	953	960	961	962	963	970
	973	974	975	976	982	985	986	987	988	994	997	998	999	1000	1008
	1014	1015	1016	1017	1025	1031	1032	1033	1034	1042	1048	1049	1050	1051	1063
	1068	1073	1078	1083	1087	1088	1089	1090	1097	1100	1101	1102	1103	1114	1122
	1132	1134	1135	1136	1137	1148	1151	1152	1158	1160	1161	1162	1163	1178	1180
	1186	1188	1189	1190	1191	1205	1209	1211	1212	1213	1214	1227	1229	1230	1231
	1232	1242	1245	1246	1247	1248	1271	1278	1280	1281	1282	1283	1308	1315	1317
	1318	1319	1320	1343	1358	1367	1368	1369	1370	1383	1386	1390	1393	1394	1395
	1396	1413	1416	1417	1418	1419	1435	1443	1446	1447	1448	1449	1460	1463	1464
	1465	1466	1477	1480	1481	1482	1483	1494	1496	1497	1498	1499	1510	1513	1514
	1515	1516	1527	1530	1531	1532	1533	1544	1546	1547	1548	1549	1560	1563	1564
	1565	1566	1577	1581	1582	1583	1584	1595	1598	1599	1600	1601	1612	1616	1617
	1618	1619	1630	1633	1634	1635	1636	1647	1650	1651	1652	1653	1664	1667	1668
	1669	1670	1681	1684	1685	1686	1687	1698	1701	1702	1703	1704	1715	1718	1719
	1720	1721	1732	1735	1736	1737	1738	1749	1752	1753	1754	1755	1766	1769	1770
	1771	1772	1783	1786	1787	1788	1789	1800	1803	1804	1805	1806	1817	1820	1821
	1822	1823	1834	1837	1838	1839	1840	1851	1854	1855	1856	1857	1868	1871	1872
	1873	1874	1885	1888	1889	1890	1891	1902	1905	1906	1907	1908	1919	1922	1923
	1924	1925	1936	1939	1940	1941	1942	1953	1956	1957	1958	1959	1970	1973	1974
	1975	1976	1987	1990	1991	1992	1993	2004	2007	2008	2009	2010	2021	2025	2026
	2027	2028	2039	2043	2044	2045	2046	2057	2060	2061	2062	2063	2084	2087	2090
	2097	2102	2110	2112	2113	2114	2115	2136	2139	2141	2147	2153	2159	2163	2165
	2166	2167	2168	2190	2191	2198	2200	2201	2202	2203	2224	2225	2232	2234	2235
	2236	2237	2263	2271	2272	2273	2274	2278	2294	2297	2298	2299	2300	2304	2323
	2325	2326	2327	2328	2347	2350	2351	2352	2353	2372	2375	2376	2377	2378	2382
	2397	2400	2401	2402	2403	2407	2424	2427	2428	2429	2430	2449	2451	2452	2453
	2454	2474	2478	2479	2480	2481	2507	2518	2519	2520	2521	2543	2553	2561	2565
	2566	2567	2568	2589	2596	2603	2605	2606	2607	2608	2631	2636	2642	2643	2644
	2645	2683	2688	2696	2697	2698	2699	2714	2717	2718	2719	2720	2735	2738	2739
	2740	2741	2756	2759	2760	2761	2762	2777	2780	2781	2782	2783	2798	2801	2802
	2803	2804	2819	2822	2823	2824	2825	2840	2843	2844	2845	2846	2861	2864	2865
	2866	2867	2882	2885	2886	2887	2888	2903	2906	2907	2908	2909	2924	2927	2928
	2929	2930	2945	2948	2949	2950	2951	2973	2978	2983	2984	2985	2986	3004	3007
	3008	3009	3010	3029	3030	3033	3037	3038	3039	3040	3055	3058	3059	3060	3061
	3076	3079	3080	3081	3082	3089	3102	3105	3106	3107	3108	3130	3137	3139	3140
	3141	3142	3145	3160	3163	3166	3167	3168	3169	3172	3187	3190	3193	3194	3195
	3196	3199	3213	3217	3218	3219	3220	3223	3237	3240	3241	3242	3243	3246	3260
	3263	3264	3265	3266	3297	3306	3308	3313	3316	3320	3328	3330	3333	3338	3339
	3340	3341	3368	3377	3379	3384	3387	3391	3394	3400	3402	3405	3410	3411	3412
	3413	3434	3443	3448	3451	3455	3456	3457	3458	3461	3498	3502	3504	3508	3509
	3510	3511	3514	3550	3554	3556	3560	3561	3562	3563	3569	3597	3599	3600	3601
	3602	3610	3623	3626	3627	3628	3629	3635	3664	3666	3667	3668	3669	3670	3671
	3676	3680	3684	3689	3692	3708	3978	3981	4049	4051	4056	4077	4078	4087	4091
	4108	4117	4155	4222	4249	4252	4253	4256	4283	4284	4287	4366	4444	4446	4460
	4490	4495	4499	4514	4543	4546	4603	4619	4636	4646	4658	4710	4727	4772	4819
	4901	4963	4981	4987											
.IFNDF	3975	5023													
.IFT	653	4051	4056	4135	4151	4152	4264	4470	4661	4669	4714	4721			
.IFTF	653	3996	4049	4052	4131	4135	4151	4262	4469	4658	4663	4710	4717		
.IIF	11	16	21	285	286	287	288	291	292	298	384	388	597	600	606
	607	608	610	611	637	3678	3684	3685	3696	3708	3712	3978	3999	4100	4108
	4114	4225	4226	4227	4228	4229	4230	4234	4263	4264	4280	4283	4284	4363	4447
	4448	4449	4450	4451	4456	4482	4490	4495	4511	4536	5009	5010	5011	5012	5013

```
          5015    5017    5018    5019    5020    5021    5022
.IRP       582     844     877     897     910     923     942     959     972     984     996    1013    1030    1047    1086
          1099    1133    1159    1187    1210    1228    1244    1279    1316    1366    1392    1415    1445    1462    1479
          1495    1512    1529    1545    1562    1580    1597    1615    1632    1649    1666    1683    1700    1717    1734
          1751    1768    1785    1802    1819    1836    1853    1870    1887    1904    1921    1938    1955    1972    1989
          2006    2024    2042    2059    2111    2164    2199    2233    2270    2296    2324    2349    2374    2399    2426
          2450    2477    2517    2564    2604    2641    2695    2716    2737    2758    2779    2800    2821    2842    2863
          2884    2905    2926    2947    2982    3006    3036    3057    3078    3104    3138    3165    3192    3216    3239
          3262    3337    3409    3454    3507    3559    3598    3625    3665    4126    4147    4165    4205    4235    4551
          4552    4573    4589    4590    4606    4612    4625    4626    4742    4762    4784    4812    4844    4892
.LIST        1     131     291     298     378     385     388     582     612     637     640     653     844     848     877
           881     897     901     910     914     923     927     942     946     959     963     972     976     984     988
           996    1000    1013    1017    1030    1034    1047    1051    1086    1090    1099    1103    1133    1137    1159
          1163    1187    1191    1210    1214    1228    1232    1244    1248    1279    1283    1316    1320    1366    1370
          1392    1396    1415    1419    1445    1449    1462    1466    1479    1483    1495    1499    1512    1516    1529
          1533    1545    1549    1562    1566    1580    1584    1597    1601    1615    1619    1632    1636    1649    1653
          1666    1670    1683    1687    1700    1704    1717    1721    1734    1738    1751    1755    1768    1772    1785
          1789    1802    1806    1819    1823    1836    1840    1853    1857    1870    1874    1887    1891    1904    1908
          1921    1925    1938    1942    1955    1959    1972    1976    1989    1993    2006    2010    2024    2028    2042
          2046    2059    2063    2111    2115    2164    2168    2199    2203    2233    2237    2270    2274    2296    2300
          2324    2328    2349    2353    2374    2378    2399    2403    2426    2430    2450    2454    2477    2481    2517
          2521    2564    2568    2604    2608    2641    2645    2695    2699    2716    2720    2737    2741    2758    2762
          2779    2783    2800    2804    2821    2825    2842    2846    2863    2867    2884    2888    2905    2909    2926
          2930    2947    2951    2982    2986    3006    3010    3036    3040    3057    3061    3078    3082    3104    3108
          3138    3142    3165    3169    3192    3196    3216    3220    3239    3243    3262    3266    3337    3341    3409
          3413    3454    3458    3507    3511    3559    3563    3598    3602    3625    3629    3665    3669    3684    3700
          4077    4229    4490    5001    5009    5010    5011    5012    5013    5014    5015    5016    5017    5018    5019
          5020    5021    5022    5023
.MACRO     292     341     582     628    1444    1649    2695    5001
.MCALL       6       7       8       9      10     131     385     612     640
.MEXIT     433
.NLIST       2     131     291     298     378     385     388     582     612     637     640     653     844     848     877
           881     897     901     910     914     923     927     942     946     959     963     972     976     984     988
           996    1000    1013    1017    1030    1034    1047    1051    1086    1090    1099    1103    1133    1137    1159
          1163    1187    1191    1210    1214    1228    1232    1244    1248    1279    1283    1316    1320    1366    1370
          1392    1396    1415    1419    1445    1449    1462    1466    1479    1483    1495    1499    1512    1516    1529
          1533    1545    1549    1562    1566    1580    1584    1597    1601    1615    1619    1632    1636    1649    1653
          1666    1670    1683    1687    1700    1704    1717    1721    1734    1738    1751    1755    1768    1772    1785
          1789    1802    1806    1819    1823    1836    1840    1853    1857    1870    1874    1887    1891    1904    1908
          1921    1925    1938    1942    1955    1959    1972    1976    1989    1993    2006    2010    2024    2028    2042
          2046    2059    2063    2111    2115    2164    2168    2199    2203    2233    2237    2270    2274    2296    2300
          2324    2328    2349    2353    2374    2378    2399    2403    2426    2430    2450    2454    2477    2481    2517
          2521    2564    2568    2604    2608    2641    2645    2695    2699    2716    2720    2737    2741    2758    2762
          2779    2783    2800    2804    2821    2825    2842    2846    2863    2867    2884    2888    2905    2909    2926
          2930    2947    2951    2982    2986    3006    3010    3036    3040    3057    3061    3078    3082    3104    3108
          3138    3142    3165    3169    3192    3196    3216    3220    3239    3243    3262    3266    3337    3341    3409
          3413    3454    3458    3507    3511    3559    3563    3598    3602    3625    3629    3665    3669    3684    3700
          4077    4229    4490    5001    5009    5010    5011    5012    5013    5014    5015    5016    5017    5018    5019
          5020    5021    5022    5023
.PAGE      341     434    2270
.REPT      298    3472    3524    5576    5716    5904    6476    6799
.SBTTL      21     131     281     292     301     308     319     341     385     434     590     633     640     654     682
           695     844     877     897     910     923     942     959     972     984     996    1013    1030    1047    1086
          1099    1133    1159    1187    1210    1228    1244    1279    1316    1366    1392    1415    1445    1462    1479
```

	1495	1512	1529	1545	1562	1580	1597	1615	1632	1649	1666	1683	1700	1717	1734
	1751	1768	1785	1802	1819	1836	1853	1870	1887	1904	1921	1938	1955	1972	1989
	2006	2024	2042	2059	2111	2164	2199	2233	2270	2296	2324	2349	2374	2399	2426
	2450	2477	2517	2564	2604	2641	2695	2716	2737	2758	2779	2800	2821	2842	2863
	2884	2905	2926	2947	2982	3006	3036	3057	3078	3104	3138	3165	3192	3216	3239
	3262	3337	3409	3454	3507	3559	3598	3625	3665	3673	3712	3975	4114	4152	4219
	4284	4363	4441	4496	4543	4600	4643	4724	4769	4816	4898	4960	4978	5001	5024
.TITLE	11														
.WORD	298	299	300	316	335	336	337	338	339	340	349	352	353	354	355
	358	359	360	361	362	363	364	367	368	369	390	391	392	393	394
	395	396	397	401	402	403	416	420	423	426	427	428	429	430	431
	3689	3692	3707	3965	3966	3968	3970	3973	3974	4151	4313	4360	4439	4522	4527
	4575	4635	4671	4720	4721	4934	4977	5008							

ERRORS DETECTED: 0

```
 *CZNCCA,CZNCCA/CRF/SOL=CZNCCA
RUN-TIME: 43 31 6 SECONDS
CORE USED:  30K
```