IP11,IP300

PROCESS CTRL SS
CVPCAC0

AH-A961C-MC

FICHE 1 OF 1

MAY 1979
digital
MADE IN USA

```
                        .SBTTL  DOCUMENT
                        .REM .
```

IDENTIFICATION

PRODUCT CODE: AC-A959C-MC

PRODUCT NAME: CVPCACO PROCESS CTRL SS

DATE: 6-FEB-79

MAINTENANCE: P. MCKEE

AUTHOR: H. SZEJNWALD

```
                        1.0             ABSTRACT
                        ****************************************************

                                THIS PROGRAM IS A DIAGNOSTIC TOOL FOR TESTING THE
                                ENTIRE FAMILY OF PCS MODULES.  THE PROGRAM IS INTENDED TO BE
                                USE BY FIELD ENGINEERING AND MANUFACTURING .
                                BY USING THIS PROGRAM AN OPERATOR IS ABLE TO CHECK THE IOCM,
                                AND ANY DIGITAL OR ANALOG MODULE.  SINCE THE ASSUMPTION
                                IS MADE THAT ALL OUTPUT MODULES ARE CONNECTED TO THE
                                CUSTOMERS WIRING DURING THE TEST OF OUTPUT MODULES THE
                                DISABLE BIT IS SET.  THIS DIAGNOSTIC WILL NOT CHECK THE
                                PART OF HARDWARE THAT IS INTERFACING DIGITAL OR ANALOG
                                MODULES WITH CUSTOMERS WIRING (DRIVING TRANSISTOR, ISOLATING
                                TRANSFORMERS, PHOTO-COUPLERS ECT).  THIS DIAGNOSTIC DOES
                                NOT REQUIRE ANY EXTERNAL DEVICES OR SPECIAL CONNECTIONS.

                        2.0             HARDWARE REQUIREMENTS
                        ****************************************************


                                  1.  LSI 11 WITH 16K OF MEMORY OR ANY PDP11 CPU WITH
                                      UNIBUS BRIDGE INTERFACE.
                                  2.  CONSOLE TERMINAL
                                  3.  FLOPPY DISC OR SOME OTHER INPUT DEVICE
                                  4.  IOCM (M7958)

                        3.0             PROGRAM CONSIDERATION
                        ****************************************************

                        3.1             CPU COMPATIBILITY
                                THIS PROGRAM CAN BE USED BY THE LSI 11 OR BY THE PDP-11
                                /04,05,10,20,34,35,40,45,50 & 70 IF UN.BUS BRIDGE MODULE
                                IS USED.

                        3.2             XXDP

                                THIS PROGRAM CAN BE CHAINED BY XXDP & WILL NOT OVERLAY
                                THE LOADER

                                CHAIN MODE OPERATION

                                1.  THE INPUT DIALOGUE WITH AN OPERATOR IS BYPASSED
                                2.  THE SUBSYSTEM IS MAPPED IN MEMORY
                                3.  THE SYSTEM TEST IS AUTOMATICALLY EXECUTED

                                NORMAL OPERATION
                                1. START AT LOC 204
                                2. SELECT TEST
                                3. PROGRAM RUNS AND GOES BACK TO MONITOR

                        3.3     ACT/APT

                        ****************************************************
                                THIS PROGRAM IS ACT COMPATIBLE TO THE EXTENT THAT APT HOOKS
                                WILL BE IN THE PROGRAM & WILL WORK THRU THE "UPTON INTERFACE"
```

```
                        3.4     EXECUTION TIME

                        ********************************************************
                                MOST INDIVIDUAL TESTS TAKE LESS THEN 1 SEC.THE ONESHOT MODULE TEST
                                TAKES 15 SEC. THE SYSTEM TEST EXECUTION TIME DEPENDS
                                ON THE NUMBER  AND TYPE OF I/O MODULES.
                                THE TEST OF ANALOG MODULES TAKES ALSO UP TO 1 MIN PER MODULE.

                        3.5     DEFAULT ADDRESSES & VECTORS

                        ********************************************************
                                THE FOLLOWING IS A LIST OF THE DEFAULT ADDRESSES & VECTORS
                                OF ALL HARDWARE TO BE USED AND THEIR CORRESPONDING MEMORY LOCATIONS.
                                IN THE CASE THAT THE IOCM IS SET TO AN ADDRESS OTHER THAN
                                171000 THESE LOCATIONS MUST BE CHANGED.
                                        BASE:   .WORD   171000  ;FIRST ADDRESS OF I/O ADDRESS BLOCK
                                        CSR:    .WORD   171377  ;ADDRESS OF IOCM CSR REGISTER
                                        IAR:    .WORD   171376  ;       "               IAR REGISTER
                                        VECTO:  .WORD   234     ;INTERRUPT VECTOR OF IOCM
                                        VECTOA: .WORD   236     ;INTERRUPT VECTOR+2 OF IOCM

                        ********************************************************
                        4.0     OPERATING PROCEDURE
                        ********************************************************
                        ********************************************************

                        4.1     PROGRAM LOADING

                                ****************
                                THE PROGRAM CAN BE LOADED FROM PAPER TAPE OR FLOPPY
                                DISK USING STANDARD PROCEDURES.

                        4.2     PROGRAM STARTING
                                ********************

                                LOCATION        200 - STARTING ADDRESS TO RUN THE DIAGNOSTICS MONITOR.
                                                      THE PROGRAM WILL PRINT OPTIONS AVAILABLE TO
                                                      THE OPERATOR.

                                LOCATION        204 - STARTING ADDRESS OF MONITOR WITHOUT PRINTING
                                                      OPERATOR'S OPTIONS.

                        4.3     INPUT DIALOGUE
                                ******************

                                IF AN OPERATOR STARTS THE PROGRAM AT LOCATION 204  HE CAN
                                SELECT ONE OF FOLLOWING OPTIONS:


                        4.3.1   S-SYSTEM TEST

                                ********************
                                THIS OPTION OF THE PROGRAM WILL MAP ALL THE ANALOG
                                AND DIGITAL MODULES CONNECTED TO THE IOCM AND BUILD
                                A TABLE OF THEM IN MEMORY.  THEN IT WILL PICK UP
                                EACH ONE AND RUN THE TESTS THAT DO NOT REQUIRE OPERATOR INTERVENTION.  IF SWITCH 14
```

```
                        THE SOFTWARE SWITCH REGISTER IS SET ,IT WILL LOOP ON
                        CURRENT TEST UNTIL OPERATOR TYPES CONTROL C. AFTER THE  SELECTED # OF
                        PASSES, THE PROGRAM WILL PRINT PASS COUNT UNLESS SWITCH 13 IS SET.

                4.3.2   M - MAP OF DBUS
                        ********************

                        THIS OPTION ALLOWS AN OPERATOR TO CHECK WHICH I/O MODULE
                        ARE CONNECTED TO THE IOCM.  IT CHECKS ALL ADDRESSES BETWEEN
                        171000 AND 171375.  FOR ADDRESSES THAT ANSWER , IT CHECKS THE
                        GENERIC CODE AND TYPES THE ADDRESS AND INTERFACE TYPE.

                4.3.3   D - TEST DIGITAL MODULE
                        ******************************

                        THIS TEST WILL EXERCISE THE DIGITAL MODULE AT THE ADDRESS
                        SPECIFIED BY THE OPERATOR.  FOLLOWING IS A DESCRIPTION OF
                        THE DIGITAL MODULE TESTS FOR EACH MODULE TYPE..
                        EACH TEST IS RUN AS MANY  TIMES AS SELECTED BY THE L OPTION
                        AND THE PASS COUNT IS PRINTED.
                        OPERATOR MUST TYPE THE ADDRESS OF MUT. FIRST.

                G670:

                        1.  CLEAR MODULE WITH CBIT
                        2.  CHECK THAT COUNTERS ARE CLEAR
                        3.  CHECK READ/WRITE WITH PATTERNS (125,252,377,0)
                        4.  CHECK THAT TBIT INITIALIZES COUNTERS
                        5.  TEST THAT COUNTER A INTERRUPTS
                        6.  CHECK THAT INTERRUPT CLEARS
                        7.  CHECK THAT COUNTER HALTS AFTER INTERRUPT
                        8.  DO STEPS 4 THRU 7 FOR COUNTER B
                        9.  CHECK TRANSITION POINTS ON COUNTERS (A&B) SEPARATELY
                        10. CHECK TRANSITION POINTS OF COUNTERS TOGETHER
                        11. CHECK THAT COUNTERS DO NOT COUNT ON LOADING THE MSB ONLY
                        12. TEST THAT C BIT CLEARS MODULE

                M5010:

                        1.  SET D BIT
                        2.  CHECK THAT ALL BITS ARE ZERO
                        3.  SET T BIT
                        4.  CHECK THAT ALL BITS ARE ONES
                        5.  SET C BIT


                M5011:
                        *** NOTE *** IF THE INTERRUPT SWITCH IS
                        ********** OFF AN ERROR WILL BE RETURNED

                        1.  SET DBIT AND TBIT
                        2.  CHECK IF INPUTS ARE ALL ONES
                        3.  CLEAR TBIT
                        4.  CHECK IF INPUTS ARE ALL ZEROS
                        5.  CLEAR ALL INTERRUPTS
                        6.  SET TBIT
                        7.  ENABLE INTERRUPT
```

```
                                8.  CHECK IF INTERRUPT OCCURRED
                                    /AN ERROR WILL OCCUR HERE IF INTERRUPTS
                                    /ARE DISABLED BY A SWITCH SETTING ON THE MODULE
                                9.  CHECK IF INPUTS ARE ALL ONES
                                10. CHECK IF COS REGISTERS ARE ALL ONES
                                11. SET RIF BIT
                                12. CLEAR ALL COS REGISTERS
                                13. CHECK IF THEY ARE CLEAR
                                14. CLEAR T BIT
                                15. CHECK IF INTERRUPT OCCURED
                                16. CHECK IF ALL INPUTS ARE ZERO
                                17. RUN STEP 10 TO 13
                                18. CLEAR ALL INTERRUPTS


                        M5012-M5012YA:
                                *** NOTE *** IF THE INTERRUPT SWITCH IS
                                ************ OFF AN ERROR WILL BE RETURNED

                                1.  SET DBIT
                                2.  CHECK IF ALL BITS ARE ZERO
                                3.  SET TBIT
                                4.  CHECK IF ALL BITS ARE ONE
                                5.  CLEAR T BIT
                                6.  CLEAR ALL INTERRUPTS
                                7.  SET T BIT
                                8.  ENABLE INTERRUPT
                                9.  CHECK IF INTERRUPT OCCURRED
                                    /AN ERROR WILL OCCUR HERE IF INTERRUPTS
                                    /ARE DISABLED BY A SWITCH SETTING ON THE MODULE
                                10. SET  RIF BIT
                                11. CLEAR INTERRUPT
                                12. CHECK IF INTERRUPTS ARE CLEAR
                                13. CLEAR T BIT
                                14. RUN STEP 9 TO 12
                                15. CLEAR ALL INTERRUPTS


                        M5013:
                                *** NOTE *** IF THE INTERRUPT SWITCH IS
                                ************ OFF AN ERROR WILL BE RETURNED

                                1.  SET DBIT
                                2.  CHECK IF ALL BITS ARE ZERO
                                3.  SET TBIT
                                4.  TEST IF ALL BITS ARE ONE
                                5.  CLEAR T BIT
                                6.  CLEAR ALL INTERRUPTS
                                7.  SET T BIT
                                8.  ENABLE INTERRUPT
                                9.  CHECK IF INTERRUPT OCCURED
                                    /AN ERROR WILL OCCUR HERE IF INTERRUPTS
                                    /ARE DISABLED BY A SWITCH SETTING ON THE MODULE
                                10. SET RIF BIT
                                11. CLEAR INTERRUPT
```

```
                                12. CHECK IF INTERRUPT IS CLEAR
                                13. CLEAR T BIT
                                14. RUN STEP 9 TO 12
                                15. CLEAR ALL INTERRUPTS

                        M5014:
                                *** NOTE *** THE LINE CLOCK MUST BE ENABLED

                                1.  CLEAR MODULE
                                2.  CHECK THAT COUNTERS ARE CLEAR
                                3.  CHECK READ/WRITE OF COUNTERS
                                4.  CHECK THAT T BIT INITIALIZES MODULE
                                5.  CHECK INTERRUPTS ON COUNTER A
                                6.  CHECK THAT INTERRUPT CLEARS
                                7.  CHECK THAT COUNTER HALTS AFTER INTERRUPT
                                8.  DO 5 THRU 7 FOR COUNTER B
                                9.  CHECK TRANSITION POINTS
                                10. CHECK THAT COUNTERS COUNT AT 2KHZ
                                11. CHECK THAT A 10 SECOND TIME BASE IS
                                    CHOSEN FOR INTERNAL TEST
                                12. CHECK THAT C BIT CLEARS MODULE

                        M5016:

                                *** NOTE *** IF ALL INTERRUPT SWITCHES ARE OFF
                                ********** AN ERROR WILL BE RETURNED

                                1.  CLEAR MODULE WITH CBIT
                                2.  CHECK THAT STATUS REGISTOR IS CLEAR
                                3.  CHECK THAT ALL COUNTERS ARE CLEAR
                                4.  TEST STATUS REGISTER LOAD AND READ
                                5.  TEST THAT CLEAR ENABLE BIT SELF CLEARS
                                6.  CHECK THAT STATUS REGISTER CLEAR BIT
                                    CLEARS EACH COUNTER INDIVIDUALLY

                                STORE DIP SWITCH SETTINGS FOR THE FOLLOWING
                                AND PRINT THE SETTINGS IF NOT IN APT MODE

                                7.  CHECK THAT COUNTERS COUNT BINARILY UPWARD
                                8.  CHECK OVERFLOW
                                9.  DETERMINE IF LEGAL RADIX
                                10. CHECK OVERFLOW CLEARS
                                11. CHECK INDIVIDUAL COUNTER INTERRUPTS
                                12. CHECK THAT CBIT CLEARS MODULE

                M6010-M6010YA:

                                1.  SET D BIT
                                2.  CLEAR ALL 4 BYTES OF I/O REGISTER
                                3.  SET DATA PATTERN 125 IN FIRST BYTE
                                4.  CHECK IF IT IS SET
                                5.  CHECK IF OTHER BYTES ARE ZERO
                                6.  DO THE SAME WITH DATA 252 , 377 , 000
                                7.  DO THE SAME WITH OTHER BYTES

                M6011:
                                THE LINE CLOCK MUST BE ENABLED
```

```
                                        1.      SET D BIT
                                        2.      SET DATA PATTERN 252 AT MUT
                                        3.      CHECK IF IT IS SET
                                        4.      WAIT UP TO 10 SEC
                                        5.      DATA SHOULD BE CLEAR
                                        6.      REPEAT STEP 2 TO 5 WITH DATA EQUAL TO 125

                                M6012:
                                M6013:

                                        1.  SET D BIT
                                        2.  SET OUTPUT REGISTER TO FOLLOWING DATA PATTEPN
                                            0,377,252,125
                                        3.  CHECK IF IT IS SET

                                M6014:
                                        *** NOTE *** THE LINE CLOCK MUST BE ENABLED

                                        1.  CLEAR MODULE WITH CBIT
                                        2.  CHECK THAT COUNTERS ARE CLEAR
                                        3.  CHECK READ/WRITE WITH PATTERNS
                                        4.  CHECK THAT T BIT INITIALIZES MODULE
                                        5.  CHECK THAT COUNTERS INTERRUPTS
                                        6.  CHECK THAT INTERRUPT CLEARS
                                        7.  CHECK TRANSITION POINTS
                                        8.  CHECK THAT COUNTERS COUNT A 2KHZ
                                        9.  CHECK THAT C BIT CLEARS MODULE

                                4.3.4   I - TEST IOCM
                                        ********************

                                THIS TEST WILL EXERCISE ALL THE FEATURES OF THE IOCM. THE  FOLLOWING
                                IS A DESCRIPTION OF ALL THE TESTS.
                                THIS TEST IS RUN AS MANY TIMES AS SELECTED AND THE PASS COUNT IS TYPED

                                      TST1:  CHECKS IF EACH BIT OF THE IOCM IS CLEARED BY THE CBIT

                                      TST2:  CHECKS IF THE INTERRUPT ENABLE BIT (EBIT) CAN BE SET AND CLEARED

                                      TST3:  CHECKS IF THE MAINTENANCE BIT (MBIT) CAN BE SET AND CLEARED

                                      TST4:  CHECKS IF THE DISABLE BIT (DBIT) CAN BE SET AND CLEARED
                                               CHECKS IF DBIT GENERATES CLEAR

                                      TST5:  CHECKS IF THE TEST BIT (TBIT) CAN BE SET AND CLEARED

                                      TST6:  CHECKS IF THE GENERIC CODE BIT (GBIT) CAN BE SET AND CLEARED

                                      TST7:  CHECKS IF THE RIF BIT (RBIT) CAN BE SET AND CLEARED

                                      TST10:  CHECKS DBUS DATA PATHS IN A MAINTENANCE
                                              MODE.  IF THE MBIT IS SET AND THE CPU ADDRESSES ANY LOCATION
                                              BETWEEN 171000 AND 171375 IT SHOULD READ BACK THE  LOWER BYTE
                                              OF THE MODULE ADDRESS.

                                      TST11:  CHECKS MAINTENANCE INTERRUPT.  IF THE MBIT & EBIT
```

```
                                        ARE SET, THE IOCM WILL GENERATE AN INTERRUPT WITH VECTOR 234.
                                        THE IAR (171376) HAS THE LOWER  BYTE OF CSR ADDRESS (377).

                                4.3.5  W -WRAP AROUND TEST
                                       **********************

                                       THIS OPTION ALLOWS THE FIELD ENGINEER TO CONNECT MODULE M6010 TO
                                       EITHER M5010 OR M5011.THEN IT WRAPS AROUND A SET OF DATA PATTERNS( 125,252,377,0 ).
                                       IF SWITCH 14 IN THE SOFTWARE SWITCH REGISTER IS SET ,IT WILL
                                       LOOP ON THIS TEST UNTIL THE OPERATOR TYPES CONTROL C. EVERY SELECTED # OF
                                       PASSES IT WILL PRINT PASS COUNT UNLESS SWITCH 13 IS SET.
                                       ERRORS WILL BE PRINTED INDICATING GOOD DATA,BAD DATA AND THE PASS # AT WHICH
                                       IT OCCURRED.


                                4.3.6  F -FIELD TEST
                                       ************

                                       THE PURPOSE OF THIS TEST IS:
                                       TO OUTPUT ANY SELECTED DATA PATTERN TO AN OUTPUT
                                          MODULE SPECIFIED BY OPERATOR.
                                       B. TO MONITOR DATA FROM AN INPUT MODULE SPECIFIED BY OPERATOR.

                                ********************************************************************
                                WARNING:    THE FIELD ENGINEER MUST REALIZE THAT THIS IS THE ONLY TEST
                                            THAT PERMITS HIM TO OUTPUT DATA TO THE CUSTOMER'S WIRING.
                                            THERE WILL BE VOLTAGE APPLIED TO CUSTOMER EQUIPMENT CONNECTED
                                            TO THE OUTPUT MODULE UNDER TEST.
                                            THEREFORE PRECAUTIONS MUST BE TAKEN  THAT AN ERRONEOUS OUTPUT WILL
                                            NOT CAUSE DAMAGE TO THE FIELD EQUIPMENT AND THAT ALL TESTING IS
                                            CONDUCTED WITH THE CUSTOMERS KNOWLEDGE

                                ********************************************************************8
                                PROCEDURE:
                                          USER SELECTS THE ADDRESS OF THE MODULE UNDER TEST.
                                          IF IT IS AN OUTPUT MODULE THEN:

                                          USER SELECTS AND OUTPUTS A DATA PATTERN,ONE BYTE AT A TIME,
                                          TO THE MODULE.TYPE CONTROL/C TO ABORT TEST.
                                          AFTER ALL BYTES OF MODULE HAVE BEEN SELECTED, USER MUST TYPE CONTROL/C
                                          TO GO BACK TO MONITOR.
```

```
                                    EXAMPLE:

                                    TYPE ADDRESS OF MUT   171XXX
                                    TYPE ONE BYTE DATA PATTERN   YYY
                                    (WHERE YYY IS A 8 BIT DATA PATTERN TO OUTPUT.
                                     ALSO, THE REQUEST FOR DATA FOR OUTPUT WILL BE REPEATED UP TO
                                    4 TIMES DEPENDENT UPON MODULE TYPE)

                                    IF IT IS AN INPUT MODULE THEN:

                                    THE TEST MONITORS AND PRINTS DATA FROM MODULE INPUTS.
                                    IT PRINTS EVERY BYTE ADDRESS OF THE MODULE WITH ITS
                                    CONTENT DURING THE FIRST PASS AND IT CONTINUES TO
                                    MONITOR THE DATA WITHOUT PRINTING IT UNLESS A CHANGE
                                    IN THE DATA OCCURED. TYPE CONTROL/C TO RETURN TO MONITOR.

                                    EXAMPLE:

                                    TYPE ADDRESS OF MUT   171XXX
                                    171XXX :  YYY
                                    (WHERE YYY IS THE 8 BITS OF DATA READ FROM THE MODULE.
                                     THE SECOND LINE OF THE MESSAGE WILL BE RPEATED
                                     UP TO 4 TIMES, WITH INCREMENTING ADDRESS,
                                     DEPENDING UPON MODULE TYPE.

                          4.3.7     L - SET PASS COUNT
                                    ******************************

                                    TYPE THE OCTAL NUMBER OF $PASSATIONS FOR ALL TESTS.
                                    DEFAULT NUMBER IS 1.
                                    MAXIMUM NUMBER IS 177777


                          4.3.8     A630  DIGITAL TO ANALOG CONVERTER (DAC)
                                    ****************************************


                          4.3.8.1   AFTER TYPING A630 AN OPERATOR ENTERS THE MONITOR FOR
                                    TESTING A630.HE HAS AN OPTION OF SELECTING ON OF THE
                                    FOLLOWING TESTS:

                                    A - CALIBRATION TEST
                                    I - INTERNAL COMPARATOR TEST
                                    D - D BIT TEST (LOGIC TEST)


                          4.3.8.2         D-BIT TEST

                                         THE D-BIT TEST CHECKS THE GENERIC CODE AND LOGIC PORTIONS OF  THE
                                    MODULE  ONLY.   IT  DOES  NOT  EXERCISE ANY PORTION OF THE ANALOG
                                    CIRCUITRY.  THEREFORE, THE I-BIT TEST MUST BE RUN IN  CONJUNCTION
                                    WITH THE D-BIT TEST TO INSURE PROPER MODULE OPERATION.


                          4.3.8.3   I - INTERNAL COMPARATOR TEST
```

```
499                 **********************************************************************
500                 *           CAUTION:  THIS TEST CAUSES VOLTAGES  TO  APPEAR  ON  THE
501                 *           MODULE'S  OUTPUTS  WHICH  MAY  BE  DANGEROUS  TO  THE  CUSTOMER'S
502                 *           PROCESS.  ALSO, OVERLOADS OR SHORT CIRCUITS IN THE  FIELD  WIRING
503                 *           WOULD  CAUSE  A  FAILURE  OF  THIS TEST.  THEREFORE, THE I/O
504                 *           CABLE  MUST  BE  REMOVED  FROM  THE  MODULE  UNDER  TEST   BEFORE
505                 *           PROCEEDING.
506                 **********************************************************************
507
508
509                       THE INTERNAL STATUS BIT TEST ACTIVATES THE ANALOG PORTION OF  THE
510                       MODULE  AND UTILIZES ANALOG COMPARATORS TO MONITOR AND REPORT THE
511                       RELATIVE LEVELS OF THE  VOLTAGE  OUTPUTS.   THIS  TEST  DOES  NOT
512                       MONITOR  THE  CURRENT OUTPUTS AND PROVIDES NO INDICATION OF THEIR
513                       FUNCTIONALITY  OR  CALIBRATION.   FAILURE  TO  PASS   THIS   TEST
514                       INDICATES  THAT THE MODULE IS EITHER DEFECTIVE OR SEVERELY OUT OF
515                       CALIBRATION.  PASSAGE OF THIS TEST DOES  NOT  INDICATE  THAT  THE
516                       MODULE'S VOLTAGE OUTPUTS ARE NECESSARILY WITHIN CALIBRATION.
517
518
519                       WHEN AN ERROR IS DETECTED, THE EXPECTED AND  ACTUAL  STATUS  BITS
520                       ARE  PRINTED  BY  THE  DIAGNOSTIC.   THEY  ARE PRESENTED IN RIGHT
521                       JUSTIFIED FORM I.E.  0017 INDICATES THAT ALL  BITS  ARE  SET.   A
522                       FAILURE  IN ANY ONE CHANNEL WILL USUALLY CAUSE TWO STATUS BITS TO
523                       BE IN ERROR AT DIFFERENT TIMES IN THE DIAGNOSTIC ACCORDING TO THE
524                       TABLE BELOW.
525
526
527                                 FAULTED                 STATUS BITS AFFECTED
528                                 CHANNEL
529
530                                   CH 0                      S1 AND/OR S3
531                                   CH 1                      S1 AND/OR S2
532                                   CH 2                      S2 AND/OR S4
533                                   CH 3                      S3 AND/OR S4
534
535
536               4.3.8.4   A -  TEST OR CALIBRATION PROCEDURE
537
538                       NORMALLY THE DAC MODULE WILL BE CALIBRATED AND SEALED AT THE TIME
539                        OF  MANUFACTURE.   FIELD  RECALIBRATION  SHOULD ONLY BE ATTEMPTED
540                        WHEN THE CUSTOMER WISHES TO CHANGE THE CURRENT OUTPUT OPTION FROM
541                        THE  4  TO  20MA RANGE TO THE 0 TO 20MA RANGE OR A MALFUNCTION IS
542                        SUSPECTED.  (THE MODULE IS SHIPPED  WITH  THE  4  TO  20MA  RANGE
543                        SELECTED).   BEFORE  ATTEMPTING RECALIBRATION, IT IS IMPORTANT TO
544                        BECOME FAMILIAR WITH THE LOCATION OF THE VARIOUS ADJUSTMENTS  AND
545                        POINTS  WHERE  THE  TEST  EQUIPMENT  WILL  BE ATTACHED.  FOR THIS
546                        INFORMATION REFER TO THE TABLE BELOW AND DRAWING D-UA-A630-0-0 OR
547                        FIGURES  6  THROUGH  10  AND  TABLES 4 AND 5 OF THE A630 HARDWARE
548                        MANUAL.
549
550
551
552                                                  CHANNEL
553
554                                                0       1       2       3
555                                               ---     ---     ---     ---
```

VOLTAGE OFFSET ADJUST	R35	R57	R79	R101
CURRENT GAIN ADJUST	R36	R58	R80	R102
CURRENT OFFSET ADJUST	R44	R66	R88	R110
CURRENT RANGE SWITCHES*	E78-3,4	E78-1,2	E80-3,4	E80-1,2
BERG PINS: VOLTAGE OUT	5	13	37	47
GND	6	14	38	48
CURRENT OUT	7	15	39	49
GND	8	16	40	50
SCREW TERMINALS: VOLTAGE OUT	5	13	21	31
GND	6	14	22	32
CURRENT OUT	7	15	23	33
GND	8	16	24	34

* WHEN CALIBRATING THE 4-20MA RANGE, THESE SWITCHES MUST BE 'OFF' FOR THE PARTICULAR CHANNEL BEING CALIBRATED. FOR THE 0-20MA RANGE, THESE SWITCHES MUST BE 'ON' FOR THE PARTICULAR CHANNEL BEING CALIBRATED. SWITCH E79-5 SHOULD BE 'ON' AT ALL TIMES.

BEFORE BEGINNING CALIBRATION OF THE DAC MODULE, THE FOLLOWING EQUIPMENT IS NEEDED:

DVM WESTON SHLUMBERGER MODEL 443 OR EQUIVALENT

EXTENDER MODULE W904B

RESISTOR 500OHM, 0.01%, 0.3 WATT
REQUIRED FOR CURRENT CALIBRATION
DEC PART NO. 13-09985-00

ACCESS TO THE ADJUSTMENTS AND SWITCHES IS PERMITTED BY PUTTING THE DAC ON AN EXTENDER MODULE. CAUTION: THE SYSTEM MUST BE POWERED DOWN BEFORE INSERTING OR REMOVING ANY MODULE. ALSO, THE CUSTOMER'S FIELD WIRING MUST BE REMOVED FROM THE ASSOCIATED SCREW TERMINAL ASSEMBLY.

THE 'A' TEST OR CALIBRATION ROUTINE OF THE DIAGNOSTIC PROGRAM WILL SYSTEMATICALLY AND INTERACTIVELY LEAD THE OPERATOR THROUGH THE REQUIRED STEPS. THE SPECIFIC COURSE OF ACTION AT EACH STEP IS DETAILED BELOW. FOR EACH ADJUSTMENT IT SHOULD FIRST BE DETERMINED IF THAT PARTICULAR ADJUSTMENT NEEDS TO BE MADE BEFORE BREAKING THE SEAL ON THE ADJUSTMENT. THE A TEST AND CHANNEL # SHOULD NOW BE SELECTED. COMPLETE CALIBRATION OF THE SELECTED

```
                    CHANNEL WILL BE COMPLETED BEFORE CONSIDERING ANOTHER CHANNEL.


               VOLTAGE REFERENCE ADJUST

                    THIS ADJUSTMENT IS COMMON TO ALL FOUR CHANNELS.  IF THE REFERENCE
                    HAS BEEN SET AND IS IN SPECIFICATION TYPE 'N' TO
                    BYPASS THIS SECTION;  OTHERWISE TYPE 'Y' TO PROCEED.


                    STEP 1:    CONNECT THE - LEAD OF THE VOLTMETER TO EITHER PIN 6 OF
                    THE  BERG  CONNECTOR  OR PIN 6 OF THE SCREW TERMINAL ASSEMBLY AND
                    THE + LEAD TO E62, PIN 14.  THE READING SHOULD BE +10.240V +-2MV
                    IF THE READING IS WITHIN SPECIFICATION TYPE CARRIAGE RETURN (CR),
                    OTHERWISE PROCEED AS FOLLOWS:

                              TRY TO BRING THE REFERENCE  WITHIN  SPECIFICATION  BY
                    ADJUSTING  ONLY R135 WITHOUT ALTERING SWITCHES E78 1-4.  NORMALLY
                    ADJUSTING R135 SHOULD BE SUFFICIENT.  IF  THE  REFERENCE  CAN  BE
                    ADJUSTED TYPE CARRIAGE RETURN (CR) OTHERWISE PROCEED AS FOLLOWS.

                              SET R135 FULLY CLOCKWISE.  USE SWITCHES E79-1 THRU  4
                    IN  ANY COMBINATION TO OBTAIN A READING AS CLOSE TO BUT LESS THAN
                    +10.240 V AS POSSIBLE.  THE EFFECTS OF THE SWITCHES ARE  BINARILY
                    WEIGHTED  WITH  E79-4  BEING  THE  LEAST SIGNIFICANT.  FINISH THE
                    ADJUSTMENT BY ADJUSTING R135 TO ACHIEVE A READING OF  +10.240  V.
                    WHEN DONE TYPE CARRIAGE RETURN (CR).

                    STEP 2:    REMOVE THE + LEAD OF THE VOLTMETER AND CONNECT TO  E62
                    PIN 13.  THE READING SHOULD BE +5.120 V +- 2MV.  IF THE READING IS
                    IN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE ADJUST R134
                    FOR THE PROPER READING.  WHEN DONE TYPE CARRIAGE RETURN (CR).


                       CHANNEL VOLTAGE OUTPUT ADJUST

                    REMOVE THE + LEAD OF THE VOLTMETER AND CONNECT TO  THE  SPECIFIED
                    VOLTAGE OUTPUT TERMINAL ON THE SCREW TERMINAL ASSEMBLY (SEE TABLE
                    ABOVE).  REMOVE THE - LEAD OF THE VOLTMETER AND CONNECT IT TO THE
                    SCREW  TERMINAL  JUST  BELOW  THE  +  LEAD.  BEFORE MAKING ANY
                    ADJUSTMENTS LOCATE  THE  VARIOUS  ADJUSTMENTS  RELATED  TO  THE
                    SELECTED CHANNEL.  FOR THIS INFORMATION REFER TO THE TABLE ABOVE
                    AND DRAWING UA-A630-0 OR FIGURE 7 OF THE A630 HARDWARE MANUAL.

                    STEP 1:    THE VOLTMETER READING SHOULD BE 0.000V +-5MV.  IF  THE
                    READING  IS IN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE
                    ADJUST THE PROPER VOLTAGE OFFSET POTENTIONETER.  WHEN  DONE  TYPE
                    CARRIAGE RETURN (CR).

                    STEP 2:    THE READING SHOULD CHANGE  TO  +10.230V +-  40MV.   IF
                    READING  IS OUT OF SPECIFICATION REPLACE THE MODULE.  THERE IS NO
                    ADJUSTMENT FOR THIS STEP.  TYPE CARRIAGE RETURN (CR).


               CHANNEL CURRENT OUTPUT ADJUST

                    THE  DIAGNOSTIC  WILL  ASK  IF  THE  CURRENT  OUTPUT  IS  TO  BE
```

CALIBRATED. IF NOT, TYPE 'N' TO BYPASS THE REST OF THE PROCEDURE. IF IT IS TO BE CALIBRATED, TYPE 'Y' TO PROCEED. THE DIAGNOSTIC WILL THEN ASK IF THE 4-20MA RANGE IS TO BE CALIBRATED. IF IT IS, TYPE 'Y', CHECK TO SEE THAT SWITCH E79-5 IS ON AND BOTH OF THE SPECIFIED CURRENT RANGE SWITCHES FOR THAT CHANNEL ARE OFF. (SEE TABLE ABOVE) A 'N' ANSWER WILL SELECT THE 0-20MA RANGE. IN THIS CASE, TURN ON BOTH OF THE SPECIFIED CURRENT RANGE SWITCHES FOR THAT CHANNEL AND SWITCH E79-5.

NOTE: THE FOLLOWING PROCEDURE IS USED FOR BOTH CURRENT RANGES.

THE ADJUSTMENT OF THE CURRENT OUTPUT REQUIRES THE USE OF EITHER A 500OHM, 0.01% PRECISION RESISTOR (DEC PART NO. 13-09985-00) OR ACCURATE CURRENT RANGES ON THE VOLTMETER. IF THE RESISTOR IS AVAILABLE, CONNECT IT BETWEEN THE SPECIFIED CURRENT OUTPUT TERMINAL AND THE GROUND TERMINAL JUST BELOW ON THE SCREW TERMINAL ASSEMBLY. (SEE TABLE ABOVE) CONNECT THE VOLTMETER LEADS DIRECTLY ACROSS THE RESISTOR LEADS WITH THE + LEAD OF THE VOLTMETER CONNECTED TO THE SPECIFIED CURRENT OUTPUT TERMINAL. IF CURRENT RANGES ON THE VOLTMETER ARE TO BE USED IN LIEU OF THE RESISTOR, CONNECT THE METER DIRECTLY TO THE TWO TERMINALS SPECIFIED.

STEP 1: THE VOLTMETER READING SHOULD BE +10.00V +- 5MV OR 20.000MA +- 10MA. IF THE READING IS IN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE ADJUST THE PROPER CURRENT GAIN POTENTIOMETER. WHEN DONE TYPE CARRIAGE RETURN (CR).

STEP 2: THE VOLTMETER READING SHOULD CHANGE TO +2.000V +- 5MV/4.000MA +-10UA IF IN THE 4-20MA RANGE OR 9.8MV +-5MV/19.5UA + 10UA IF IN THE 0-20 MA RANGE. IF THE READING IS IN SPECIFICATION TYPE CARRIAGE RETURN (CR), OTHERWISE ADJUST THE PROPER CURRENT OFFSET POTENTIOMETER. WHEN DONE TYPE CARRIAGE RETURN (CR).

THE DIAGNOSTIC WILL ASK IF THE CALIBRATION HAS BEEN REVERIFIED AND THE ADJUSTMENT OF EITHER POTENTIOMETER WILL AFFECT THE OTHER. THIS IS NECESSARY BECAUSE STEPS 1 AND 2 ABOVE ARE INTEREACTIVE TYPE 'Y' ONLY IF STEPS 1 AND 2 CAN BE COMPLETED SEQUENTIALLY AND IN THAT ORDER WITHOUT ADJUSTING EITHER POTENTIOMETER. IF EITHER ADJUSTMENT WAS MADE ON THIS PASS, TYPE 'N'. THE DIAGNOSTIC WILL GO BACK TO STEP 1 OF THE CURRENT OUTPUT ADJUSTMENT. WHEN A 'Y' IS TYPED, CALIBRATION OF THAT CHANNEL IS COMPLETE AND THE DIAGNOSTIC RETURN TO ITS STARTING POINT.

4.3.9 A014 -ANALOG TO DIGITAL CONVERTER

****

4.3.9.1 THE PURPOSE OF THIS PROGRAM IS TO PERFORM THE TESTS OF THE A014 WITH OPERATOR INTERVENTION.

THE OPERATOR CAN CHOOSE ONE OF THE FOLLOWING TESTS:
D - LOGIC TEST

```
                        C - CALIBRATE AND VERIFY OFFSET AND GAIN OF A014,A156 OR A157
                        L - TEST THE LINEARITY OF THE CONVERTER
                        M - CHECK IF CONVERTER SKIPS OR MISSES ANY OUTPUT CODE "MONOTONICITY TEST"
                        X - TEST IF A/D WORKS WITH A156 OR A157 MUX

              4.3.9.2 OPERATING PROCEDURES:
                        WHEN ANALOG MODULE TEST IS SELECTED IT PRINTS:
                        WHICH TEST (D,C,L,M,X OR H FOR HELP)?
                        OPERATOR MUST RUN THE FOLLOWING TESTS IN THIS ORDER AND SHOULD TYPE:
              4.3.9.3  D - A/D LOGIC TEST

                        THE LOGIC TEST VERIFIES THE FUNCTIONALITY OF THE A014 A/D
                        CONVERTER CONSISTING OF A MOTHER AND DAUGHTER BOARD AS FOLLOWS:
                        1. VERIFY ADDRESS RESPONSE
                        2. VERIFY BIT FUNCTIONALITY (ERROR BIT,DONE BIT, GO BIT)
                        3. VERIFY SINGLE ENDED MODE FUNCTIONALITY
                              CHANNEL SELECTION
                              GAIN SELECTION
                        4. VERIFY DIFFERENTIAL MODE FUNCTIONALITY
                        5. CHECK IF CONVERSION GETS DONE
                        6. VERIFY FUNCTIONS USING T-BIT
                              SET T-BIT AT IOCM
                              SELECT CHANNEL FOR +,-,OR 0 REFERENCE MAINTENANCE VOLTAGE
                              VERIFY INDIVIDUAL OUTPUTS
                              SET MAINTENANCE RAMP VOLTAGE
                              VERIFY THAT RAMP GOES FROM +10.240V TO -10.240V
                        5. SET D-BIT AT IOCM
                        6. VERIFY ZERO (4000) OUTPUT
                        8. CLEAR T-BIT AND D-BIT
                        9. RETURN TO MONITOR



              4.3.9.4  C- CALIBRATE AND VERIFY A/D CONVERTER

                        TO ENSURE COMPLIANCE WITH SPECIFICATIONS, THE VOLTAGE SOURCE USED IN
                        CALIBRATING THE A014 MUST HAVE BEEN ACCURATELY CALIBRATED WITHIN
                        THE PREVIOUS SIX MONTHS. THE USE OF AN ACCURATE CUSTOMER PROVIDED
                        VOLTAGE REFERENCE IS RECOMMENDED WHEN AVAILABLE, AS DIFFERENCES BETWEEN
                        THE CUSTOMER'S REFERENCE AND THAT USED FOR CALIBRATION COULD INDUCE
                        AN ERROR DURING CUSTOMER USE.
                        IN ADDITION THE PERSONNEL DOING THE CALIBRATION SHOULD BE FAMILIAR
                        WITH THE PROCEDURES FOR ALIGNMENT OF PRECISION ANALOG EQUIPMENT.
                        IF YOU ARE NOT SURE OF THE FOREGOING, DO NOT ATTEMPT TO CALIBRATE
                        THE A014; ITS FACTORY CALIBRATION IS PROBABLY BETTER THEN YOU
                        WILL BE ABLE TO ACCOMPLISH.

                        IN THIS TEST OPERATOR SHOULD CONNECT HIS VOLTAGE SOURCE
                        TO THE INPUT OF THE A014,A156 OR A157 MUX ON THE SELECTED CHANNEL.
                        THE PROGRAM WILL MAKE 8 CONVERSIONS ,CALCULATE THE AVERAGE
                        VALUE OF THIS CONVERSIONS AND PRINT THE RESULT IN OCTAL
                        FORMAT AND IN MILLIVOLTS. THE RESULTS WILL BE PRINTED EVERY
                        2 - 4 SEC.TO EXIT THIS TEST TYPE CONTROL A OR CONTROL C.
                        IN THE CASE OF THE A157 OPERATOR MUST ALSO SELECT THE GAIN.


                        R41 ADJUSTS THE ZERO OF THE A014. THIS IS BEST DONE AT A [illegible]DE
```

```
                        TRANSITION POINT, SO THAT MORE RESOLUTION THEN THE CONVERTER
                        QUANTIZATION LEVEL CAN BE OBTAINED.

                        R31 IS THE GAIN ADJUSTMENT POT; IT ADJUSTS BOTH PLUS AND MINUS
                        FULL SCALE SYMMETRICALLY ABOUT ZERO. AGAIN, BEST ADJUSTMENT RESOLUTION
                        CAN BE OBTAINED AT A CODE TRANSITION POINT.

                        R41 SHOULD ALWAYS BE ADJUSTED FIRST, USING AN INPUT VERY CLOSE TO ZERO.
                        R31 CAN BE ADJUSTED WITH AN INPUT CLOSE TO EITHER FULL SCALE OR TO
                        OPTIMIZE ACCURACY FOR A PARTICULAR APPLICATION, AT ANY
                        POINT IN THE INPUT RANGE SUFFICIENTLY FAR FROM ZERO.
                        FOR GREATEST AVERAGE ACCURACY OF AN EXPANDED SUBSYSTEM,
                        THE A014 SHOULD BE CALIBRATED WITH AN INPUT APPLIED THROUGH
                        THE ON-BOARD MULTIPLEXOR; IF IT IS DESIRED TO OPTIMIZE ACCURACY
                        THROUGH AN EXPANDER MULTIPLEXOR AT A SPECIFIC GAIN, THE
                        CALIBRATION INPUTS SHOULD BE APPLIED THROUGH THAT MULTIPLEXOR.
                        THE USER SHOULD RECOGNIZE THAT THIS MAY PLACE THE A014 ALONE OUT OF SPEC.

                        ZERO ADJUSTMENT -
                          1.    APPLY -2.5 MV TO THE SCREW TERMINALS OF THE SELECTED
                                CHANNEL AND START THE CALIBRATION TEST.
                          2.    VERIFY THAT THE OCTAL PRINTOUT VARIES BETWEEN 003777 AND 004000.
                                A 50 - 50 DISTRIBUTION WOULD BE IDEAL, BUT THIS IS NOT
                                USUALLY OBTAINABLE. IF NECESSARY, ADJUST R41(LOWER POT)
                                UNTIL THE PRINTOUT VARIES PER THE ABOVE. THIS ADJUSTS
                                THE TRANSITION POINT FROM 0.0 MV TO -5.0 MV.(ONE LSB.)
                           3.   APPLY +2.5 MV TO THE SAME CHANNEL AND VERIFY THAT THE PRINTOUT
                                IS 004000 OR 004001. VARIANCE IS NOT REQUIRED, BUT
                                IS ACCEPTABLE BETWEEN THESE TWO VALUES.
                        GAIN ADJUSTMENT -
                          1.    APPLY A VOLTAGE OF +10.2325V (FULL SCALE MINUS HALF LSB)
                                TO THE SELECTED CHANNEL AND START THE CALIBRATION TEST.
                          2.    VERIFY THAT THE PRINTOUT VARIES BETWEEN 007776 AND 007777.
                                IF NECESSARY ADJUST THE GAIN POT R31 UNTIL THE PROPER
                                PRINTOUT IS OBTAINED.
                          3.    APPLY -10.2375V TO THE SELECTED CHANNEL(FULL SCALE MINUS HALF LSB)
                                AND RESTART THE CALIBRATION TEST IF NECESSARY.
                          4.    VERIFY THAT THE OCTAL PRINTOUT IS EITHER 000001 OR 000000
                                OR VARYING BETWEEN THE TWO VALUES.
4.3.9.5   L- LINEARITY TEST

                        THIS IS A LINEARITY AND NOISE MEASUREMENT TEST USING
                        10V TEST RAMP.BEFORE RUNNING THIS TEST LOGIC TEST MUST BE RUN.

                        A. PROGRAM CALCULATES THE AVERAGE OF CONVERSIONS PER S ATE
                           BY DIVIDING THE TOTAL # OF CONVERSIONS THRU THE RAMP BY
                           4096 POINTS.
                        B. COMPARES # OF CONVERSIONS FOR EACH STATE WITH LOW LIMIT
                           AND HIGH LIMIT AVERAGES

                                LOW LIMIT AVERAGE = AVERAGE/2-1
                                HIGH LIMIT AVERAGE = AVERAGE +AVERAGE/2+1
                        C. STORES UP TO 20 ERRORS IN MEMORY, PRINTS ERROR MESSAGE IF
                           # OF CONVERSIONS FOR ANY STATE IS BELOW THE LOW LIMIT OR ABOVE
                           THE HIGH LIMIT AND RETURN TO MONITOR

                        FOR EXAMPLE:
```

```
                        LINEARITY TEST ERROR
                                TSTPNT  MIN     MAX     WAS(# OF CONVERSIONS)
                                7770    4       12      3
                                4000    4       12      13

                        IN THE EXEMPLE THE STATE 7770 WAS TOO NARROW AND THE
                        STATE AT 4000 (0V) WAS TOO WIDE.
                        D. PRINTS A MESSAGE "TOO MANY ERRORS" FOR 20 ERRORS OR MORE
                           AND RETURN TO MONITOR
                        E. PRINTS A MESSAGE "END OF RAMP" IF NO ERROR FOUND AND
                           RETURN TO MONITOR
                        F. OPERATOR TYPES CNTRC TO TERMINATE ERRORS PRINTOUT AND
                           RETURN TO MONITOR
                        G. AN EXCESSIVE # OF ERRORS INDICATES THAT THE CONVERTER
                           IS NOT RESPONDING IN A LINEAR MANNER.

                4.3.9.6  M -  MONOTONICITY TEST

                        THE OBJECTIVE OF THIS TEST IS TO READ EVERY POSSIBLE STATE  THRU
                        THE RAMP AT LEAST ONCE STARTING FROM THE BOTTOM (-10.240),GOING
                        UP THE RAMP (+10.240) AND THEN DOWN TO (-10.240).
                        BEFORE RUNNING THIS TEST LOGIC TEST MUST BE RUN.

                        1. COMPARES LAST CONVERSION WITH THE PREVIOUS ONE AT ALL POINTS.
                        2. CONTINUE CONVERSION WITH NEXT POINT IN RAMP IF ANY ERROR IS FOUND.
                        3. PRINT ERRORS AT THE END OF THE RAMP,IF LESS THAN 20 ERRORS FOUND.
                         THE ERRORS OCCUR IF THE PROGRAM DETECTS ANY STATE THAT IS MISSING
                         OR ANY NOISE THAT IS MORE THEN 2 BITS.

                        EXAMPLE: OUT OF RANGE BY TWO BITS
                                 GDDAT  BDDAT   RAMP(UP=0,DOWN=1)
                                 0      2       0               GOING UP THE RAMP
                                 7776   7774    1               GOING DOWN THE RAMP

                        4. PRINTS A MESSAGE "TOO MANY ERRORS" FOR 20 ERRORS OR MORE AND
                        RETURN TO MONITOR.
                        5. PRINTS A MESSAGE "END OF RAMP" FOR NO ERRORS FOUND
                        6. OPERATOR TYPES CNTR A TO TERMINATE ERRORS PRINTOUT AND RETURN TO MONITOR.


                4.3.9.6  X - MULTIPLEXER TEST (A156, A157)

                        THE OPERATOR IS ASKED FIRST WHICH MUX HE WANTS TO TEST.
                        THEN THE PROGRAM DETERMINES FROM THE GENERIC CODE IF IT IS
                        A156 SINGLE ENDED,A156 DIFF MODE OR A157 AND IT CHECKS THE
                        LOGIC OF THE SELECTED MUX.
                        THE TEST INCLUDES:
                        1. CHANNEL SELECT REGISTER
                        2. GAIN SELECT REGISTER
                        3. ERROR BIT FUNCTIONALITY
                        4. RESULTS OF THE CONVERSION WITH T BIT SET (4000)
```

```
                                4.3.10  T - SET SOFTWARE SWITCH REGISTER
                                        TYPE THE OCTAL NUMBER TO BE LOADED TO SWREG

                                        100000 - HALT ON ERROR
                                        40000  - LOOP ON TEST
                                        20000  - INHIBIT ERROR AND END OF PASS PRINT


                                4.3.11  CONTROL C

                                        TYPING CONTROL C CAUSES THE PRESENT TEST TO BE ABORTED
                                        AND PROGRAM RETURNS TO THE MONITOR.
                                        IN SOME CASES IT MAY TAKE UP TO 20 SECONDS FOR THIS TO
                                        HAPPEN.

                                4.3.12  CONTROL A

                                        IN CASE THAT THE MODULE HAS AN INDIVIDUAL MONITOR (A014,A630)
                                        BY TYPING CONTROL A DURING THE TEST EXECUTION THE PROGRAM RETURNS
                                        TO THE BEGINING OF THIS MONITOR.



                                .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 004364              JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
                    000100              .=100
     000100         040554     NOCLK
                    000102              .=102
     000102         000340     340
                    000042              .=42
     000042         000000              HALT
                    000046              .=46
     000046         006660              LOGIC
                    000204              .=204
     000204  000137 004676              JMP     SETCLK

                                .SBTTL  BASIC DEFINITIONS
                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
                    001100      STACK=  1100
                    000004      SCOPE   =       IOT
                                ;*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
                    177776      PSW     =       PS
                    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
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  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
001000                  SW9     =       SW09
000400                  SW8     =       SW08
000200                  SW7     =       SW07
000100                  SW6     =       SW06
000040                  SW5     =       SW05
000020                  SW4     =       SW04
000010                  SW3     =       SW03
000004                  SW2     =       SW02
000002                  SW1     =       SW01
000001                  SW0     =       SW00
                        ;*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
                001000          BIT9    =       BIT09
                000400          BIT8    =       BIT08
                000200          BIT7    =       BIT07
                000100          BIT6    =       BIT06
                000040          BIT5            BIT05
                000020          BIT4    =       BIT04
                000010          BIT3    =       BIT03
                000004          BIT2    =       BIT02
                000002          BIT1    =       BIT01
                000001          BIT0    =       BIT00
                                ;*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
  929                           .SBTTL  BIT DEFINITIONS
  930           000200          FBIT=BIT7                       ;FLAG BIT
  931           000100          EBIT=BIT6                       ;INTERRUPT ENABLE
  932           000040          MBIT=BIT5                       ;MAINTENANCE INTERRUPT
  933           000020          DBIT=BIT4                       ;I/O DISABLE BIT
  934           000010          TBIT=BIT3                       ;TEST BIT, INVERTS I/O BITS
  935           000004          GBIT=BIT2                       ;GENERIC CODE ENEBLE
  936           000002          CBIT=BIT1                       ;CLEAR I/O
  937           000001          RBIT=BIT0                       ;RIF BIT, CLEARS I/O INTERRUPT
 1000
 1032
 1039           001100          .=1100
 1040                           .SBTTL  APT PARAMETER BLOCK
                                ;;*****************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;*****************************************************************
                001100                  .$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    001100                  $APTHDR ;;POINT TO APT HEADER BLOCK
                001100                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.
```

APT PARAMETER BLOCK SEQ 0020

```
001100                          $APIMD:
001100  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001102  001202                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001104  000120                  $TSTM:  .WORD   120     ;;RUN TIM OF LONGEST TEST
001106  000600                  $PASTM: .WORD   600     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001110  000600                  $UNITM: .WORD   600     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001112  000052                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
  1241                                  .SBTTL  COMMON TAGS
                                        ;;*****************************************************************
                                        ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                        ;*USED IN THE PROGRAM.
                 001114                   .=1114
 001114                                 $CMTAG:                         ;;START OF COMMON TAGS
 001114  000000                             .WORD   0
 001116     000                         $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
 001117     000                         $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
 001120  000000                         $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
 001122  000000                         $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
 001124  000000                         $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
 001126  000000                         $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
 001130     000                         $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
 001131     001                         $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
 001132  000000                         $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
 001134  000000                         $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
 001136  000000                         $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
 001140  000000                         $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
 001142  000000                         $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
 001144  000000                             .WORD   0                   ;;RESERVED--NOT TO BE USED
 001146  000000                             .WORD   0
 001150     000                         $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
 001151     000                         $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
 001152  000000                             .WORD   0
 001154  177570                         SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
 001156  177570                         DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
 001160  177560                         $TKS:   177560                  ;;TTY KBD STATUS
 001162  177562                         $TKB:   177562                  ;;TTY KBD BUFFER
 001164  177564                         $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
 001166  177566                         $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
 001170     000                         $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
 001171     002                         $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
 001172     012                         $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
 001173     000                         $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
 001174  000000                         $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
 001176     077                         $QUES:  .ASCII  /?/             .;QUESTION MARK
 001177     015                         $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
 001200     012      000                $LF:    .ASCIZ  <12>            ;;LINE FEED
                                        ;;*****************************************************************
                                        .SBTTL  APT MAILBOX-ETABLE
                                        ;;*****************************************************************
                                        .EVEN
 001202                                 $MAIL:                          ;;APT MAILBOX
 001202  000000                         $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
 001204  000000                         $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
 001206  000000                         $TESTN: .WORD   ATESTN          ;;TEST NUMBER
 001210  000000                         $PASS:  .WORD   APASS           ;;PASS COUNT
 001212  000000                         $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
 001214  000000                         $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
 001216  000000                         $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
 001220  000000                         $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
 001222                                 $ETABLE:                        ;;APT ENVIRONMENT TABLE
 001222     000                         $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
 001223     000                         $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
 001224  000000                         $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
 001226  000000                         $USWR:  .WORD   ALSWR           ;;USER SWITCHES
```

```
001230  000000                  $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
001232     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001233     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001234  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001236     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001237     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001240  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001242     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001243     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001244  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001246     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001247     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001250  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001252  000000                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001254  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001256  000000                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001260  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001262  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001264  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001266  000000                  $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
001270  000000                  $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
001272  000000                  $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
001274  000000                  $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
001276  000000                  $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
001300  000000                  $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
001302  000000                  $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
001304  000000                  $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
001306  000000                  $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
001310  000000                  $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
001312  000000                  $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
001314  000000                  $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
001316  000000                  $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
001320  000000                  $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
001322  000000                  $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
001324  000000                  $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
001326                          $ETEND:
001326                          ATABL:  .BLKW   80.             ;TABLE A=ADDRESSES OF I/O MODULES
001566                          BTABL:  .BLKW   80.             ;TABLE B=GENERIC CODES OF THIS MODULES
002026  000141                  GENER:  .WORD   141             ;GENERIC CODE FOR NONISOLATED DC 32 BITS IN
002030  000121                          .WORD   121             ;GENERIC CODE FOR NONISOLATED DC 16 BITS IN
002032  000122                          .WORD   122             ;GENERIC CODE FOR ISOLATED DC 16 BITS IN
002034  000101                          .WORD   101             ;GENERIC CODE FOR AC 8 BITS IN
002036  000041                          .WORD   41              ;GENERIC CODE FOR NONISOLATED DC 32 BITS OUT
002040  000021                          .WORD   21              ;GENERIC CODE FOR ONESHOT 16 BITS OUT
002042  000001                          .WORD   1               ;GENERIC CODE FOR ISOLATED DC 8 BITS OUT
002044  000002                          .WORD   2               ;GENERIC CODE FOR AC 8 BITS OUT
```

```
002046  000123                          .WORD   123             ;16 BIT TTL COMPATABLE ISOLATED INPUT
002050  000043                          .WORD   43              ;32 BIT TTL COMPATABLE NONISOLATED OUTPUT
002052  000144                          .WORD   144             ;M5014 INPUT COUNTER MODE 1
002054  000145                          .WORD   145             ;M5014 INPUT COUNTER MODE 2
002056  000146                          .WORD   146             ;M5014 INPUT COUNTER MODE 3
002060  000147                          .WORD   147             ;M5014 INPUT COUNTER MODE 4
002062  000142                          .WORD   142             ;M5016 QUAD 8 BIT COUNTER/PRESCALAR
002064  000044                          .WORD   44              ;M6014 DUAL 16 BIT OUTPUT COUNTER/PRESCALAR DAUGHTER
002066  000044                          .WORD   44              ;G670 MOTHERBOARD
002070  000261                          .WORD   261             ;GENERIC CODE FOR FOUR CHANEL DAC
002072  000321                          .WORD   321             ;A/D SINGLE ENDED
002074  000301                          .WORD   301             ;A/D DIFFERENTIAL MODE
002076  000000                          .WORD   0
002100  005010                  MUT:    .WORD   5010
002102  005011                          .WORD   5011
002104  005012                          .WORD   5012
002106  005013                          .WORD   5013
002110  006010                          .WORD   6010
002112  006011                          .WORD   6011
002114  006012                          .WORD   6012
002116  006013                          .WORD   6013
002120  005012                          .WORD   5012            ;M5012-YA
002122  006010                          .WORD   6010            ;M6010-YA
002124  005014                          .WORD   5014            ;M5014
002126  005014                          .WORD   5014            ;M5014
002130  005014                          .WORD   5014            ;M5014
002132  005014                          .WORD   5014            ;M5014
002134  005016                          .WORD   5016            ;M5016
002136  006014                          .WORD   6014            ;M6014
002140  000670                          .WORD   670             ;G670
002142  000630                          .WORD   630
002144  000014                          .WORD   14
002146  000014                          .WORD   14
002150  011242                  MODUL:  .WORD   M5010           ;THIS TABLE HAS ADDRESSES OF I/O SUBRUTINE
002152  012670                          .WORD   M5011
002154  013520                          .WORD   M5012
002156  011344                          .WORD   M5013
002160  012006                          .WORD   M6010
002162  012372                          .WORD   M6011
002164  012200                          .WORD   M6012
002166  012200                          .WORD   M6013
002170  013520                          .WORD   M5012           ;M5012-YA SAME A M5012
002172  012006                          .WORD   M6010           ;M6010-YA SAME AS M6010
002174  016402                          .WORD   M5014           ;INPUT COUNTER
002176  016402                          .WORD   M5014           ;INPUT COUNTER
002200  016402                          .WORD   M5014           ;INPUT COUNTER
002202  016402                          .WORD   M5014           ;INPUT COUNTER
002204  024144                          .WORD   M5016           ;QUAD 8 BIT COUNTER/PRESCALAR
002206  021546                          .WORD   M6014           ;DUAL 16 BIT COUNTER DAUGHTER
002210  014176                          .WORD   G670            ;MOTHERBOARD
002212  026676                          .WORD   DCAMON
002214  032236                          .WORD   ADTST
002216  032236                          .WORD   ADTST
002220  000000                          .WORD   0
002222  000000                  YLOOP:  .WORD   0
002224  000000                  MXNUM:  .WORD   0
002226  000000                  DBUFF:  .WORD   0
```

```
002230  000000                  .WORD   0
002232     125          PATT:   .BYTE   125
002233     252                  .BYTE   252
002234     377                  .BYTE   377
002235     000                  .BYTE   0
002236  000042          XXDP:   .WORD   42
002240  000000          $MUT:   .WORD   0
002242  000000          BYTNUM: .WORD   0               ;NUMBER OF BITES OF I/O
002244  000000          TADDR:  .WORD   0               ;ADDRESS OF MUT
002246  000000          TADDR1: .WORD   0
002250  000000          TBADDR: .WORD   0
002252  000000          COSADR: .WORD   0               ;SECOND ADDRESS OF COS MODULE
002254  000000          DMUT:   .WORD   0               ;DIGITAL MODULE UNDER TEST
002256  000000          LONGIN: .WORD   0
002260  177546          CLKADR: .WORD   177546
002262  171377          CSR:    .WORD   171377
002264  171376          IAR:    .WORD   171376
002266  000234          VECTO:  .WORD   234
002270  000236          VECTOA: .WORD   236
002272  000000          VECT1:  .WORD   0
002274  000000          VECT1A: .WORD   0
002276  171000          BASE:   .WORD   171000          ;FIRST I/O
002300  000000          CLK:    .WORD   0
002302  000000          NORAMP: .WORD   0
002304  000000          TEMP:   .WORD   0
002306  000000          RERROR: .WORD   0
002310  000000          XXX:    .WORD   0
002312  000000          AFLAG:  .WORD   0
002314  000000          TEMP1:  .WORD   0
002316  000000          INTDAT: .WORD   0
002320  000001          PASCNT: .WORD   1
002322  000100          CLKVC:  .WORD   100
002324  000102          CLKVCA: .WORD   102
002326  032130          MOD:    .WORD   F5010
002330  032130                  .WORD   F5011
002332  032142                  .WORD   F5012
002334  032154                  .WORD   F5013
002336  032166                  .WORD   F6010
002340  032222                  .WORD   F6011
002342  032206                  .WORD   F6012
002344  032206                  .WORD   F6013
002346  032142                  .WORD   F5012           ;M5012-YA
002350  032166                  .WORD   F6010           ;M6010-YA
002352  000000                  .WORD   0
002354  000060          KBVEC:  .WORD   60

002356  000000          CHNUM:  .WORD   0
002360  000000          CH0:    .WORD   0
002362  000000          CH1:    .WORD   0
002364  000000          CH2:    .WORD   0
002366  000000          CH3:    .WORD   0
002370  000000          CH4:    .WORD   0
002372  000000          DCHAN:  .WORD   0
002374  000000          XCHAN:  .WORD   0
002376  000000          VCHAN:  .WORD   0
002400  000000          ICHAN:  .WORD   0
002402  171000          LDATA0: .WORD   171000
```

```
002404  171001                  HDATA0: .WORD   171001
002406  171002                  LDATA1: .WORD   171002
002410  171003                  HDATA1: .WORD   171003
002412  171004                  LDATA2: .WORD   171004
002414  171005                  HDATA2: .WORD   171005
002416  171006                  LDATA3: .WORD   171006
002420  171007                  HDATA3: .WORD   171007
002422  000000                  KOUNT:  .WORD   0
002424  000000                  ACOUNT: .WORD   0
002426  000000                  BCOUNT: .WORD   0
002430  000000                  CCOUNT: .WORD   0
002432  000000                  ANSW:   .WORD   0
002434  000000                  DCOUNT: .WORD   0
002436  000000                  CONV:   .WORD   0
002440  000000                  DATALO: .WORD   0
002442  000000                  CONVCT: .WORD   0
002444  000000                  ECOUNT: .WORD   0
002446                          RAMP1:  .BLKW   40.
002566                          RAMP2:  .BLKW   40.
002706                          RAMP3:  .BLKW   40.
003026  000000                  BLAST:  .WORD   0
003030  000000                  LAST:   .WORD   0
003032  000000                  LASTCN: .WORD   0
003034  000000                  ROTPAT: .WORD   0
003036  177777                  ROTFLG: .WORD   -1
003040  000000                  TEMP2:  0
003042  000000                  TEMP3:  0
003044  000000                  TEMP4:  0
003046  000000                  TEMP5:  0
003050  000000                  GBITE:  0
003052  000000                  STAT1:  0
003054  000000                  STAT2:  0
003056  000000                  LBYTE:  0
003060  000000                  HBYTE:  0
003062  000001                  GAINTB: .WORD   1       ;TABLE OF A157 GAINS
003064  000002                          .WORD   2
003066  000010                          .WORD   10
003070  000020                          .WORD   20
003072  000050                          .WORD   50
003074  000100                          .WORD   100
003076  000200                          .WORD   200
003100  001000                          .WORD   1000
003102  000000                          .WORD   0
003104  000000                  GAIN:   .WORD   0               ;CORRESPONDING OCTAL VALUES TO BE
003106  000020                          .WORD   20              ;LOADED INTO THE GAIN REGISTER OF A157
003110  000100                          .WORD   100
003112  000120                          .WORD   120
003114  000200                          .WORD   200
003116  000220                          .WORD   220
003120  000300                          .WORD   300
003122  000320                          .WORD   320
003124  000000                          .WORD   0
003126  000000                  ST1SAV: .WORD   0               ;STAT1 SAVE
003130  000000                  ST2SAV: .WORD   0       ;STAT2 SAVE
003132  000000                  ACNTH:  0               ;HIGH BYTE ADDRESS OF COUNTER A
003134  000000                  ACNTL:  0               ;LOW BYTE ADDRESS OF COUNNTER A
003136  000000                  BCNTH:  0               ;HIGH BYTE ADDRESS OF COUNTER B
```

```
.MAIN.  MACRO M1110  20-FEB-79 11:22  PAGE 8-5                                  SEQ 0026
APT MAILBOX-ETABLE

003140  000000                    BCNTL:  0               ;LOW BYTE ADDRESS OF COUNTER B
003142  000000                    CTRSH:  0               ;COUNTER UNDER TEST HIGH BYTE ADDRESS
003144  000000                    CTRSL:  0               ;COUNTER UNDER TEST LOW BYTE ADDRESS
003146  000000                    CNUM:   0               ;COUNTER NUMBER
003150  000000                    NUM:    0               ;
003152  000000                    VARI:   0               ;
003154  000000                    XLOOP:  0               ;
003156  000000                    TEMPB:  0               ;
003160  000000                    RETURN: 0
003162  000000                    AVRSAV: 0
003164     000      001     001   AVRTBL: .BYTE   0,1,1,2,2,3,3,4
003167     002      002     003
003172     003      004

                                 .EVEN
```

```
                                .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.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        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
     003174                     $ERRTB:
1242 003174  045250                     EM1
1243 003176  045461                     DH1
1244 003200  061450                     DT1
1245 003202  000000                     0
1246 003204  045277                     EM2
1247 003206  045461                     DH1
1248 003210  061450                     DT1
1249 003212  000000                     0
1250 003214  045512                     EM3
1251 003216  000000                     0
1252 003220  000000                     0
1253 003222  000000                     0
1254 003224  045543                     EM4
1255 003226  045461                     DH1
1256 003230  061450                     DT1
1257 003232  000000                     0
1258 003234  045571                     EM5
1259 003236  045611                     DH5
1260 003240  061464                     DT5
1261 003242  000000                     0
1262 003244  045644                     EM6
1263 003246  045461                     DH1
1264 003250  061450                     DT1
1265 003252  000000                     C
1266 003254  045713                     EM7
1267 003256  045730                     DH7
1268 003260  061430                     DT7
1269 003262  000000                     0
1270 003264  046000                     EM10
1271 003266  000000                     0
1272 003270  000000                     0
1273 003272  000000                     0
1274 003274  046033                     EM11
1275 003276  045730                     DH7
1276 003300  061430                     DT7
1277 003302  000000                     0
1278 003304  046071                     EM12
1279 003306  045611                     DH5
1280 003310  061464                     DT5
1281 003312  000000                     0
1282 003314  046121                     EM13
1283 003316  000000                     0
1284 003320  000000                     0
1285 003322  000000                     0
1286 003324  046167                     EM14
1287 003326  000000                     0
```

ERROR POINTER TABLE

```
1288 003330  000000                    0
1289 003332  000000                    0
1290 003334  046232                    EM15
1291 003336  000000                    0
1292 003340  000000                    0
1293 003342  000000                    0
1294 003344  046275                    EM16
1295 003346  045461                    DH1
1296 003350  061450                    DT1
1297 003352  000000                    0
1298 003354  045713                    EM17
1299 003356  045730                    DH7
1300 003360  061360                    DT17
1301 003362  000000                    0
1302 003364  046340                    EM20
1303 003366  045730                    DH7
1304 003370  061400                    DT20
1305 003372  000000                    0
1306 003374  046372                    EM21
1307 003376  046434                    DH21
1308 003400  061500                    DT21
1309 003402  000000                    0
1310 003404  046460                    EM22
1311 003406  000000                    0
1312 003410  000000                    0
1313 003412  000000                    0
1314 003414  046534                    EM23
1315 003416  046571                    DH23
1316 003420  061420                    DT23
1317 003422  000000                    0
1318 003424  046664                    EM24
1319 003426  000000                    0
1320 003430  000000                    0
1321 003432  000000                    0
1322 003434  047550                    EM25
1323 003436  000000                    0
1324 003440  000000                    0
1325 003442  000000                    0
1326 003444  055312                    EM26
1327 003446  045611                    DH5
1328 003450  061464                    DT5
1329 003452  000000                    0
1330
1331 003454  055246                    EM27
1332 003456  045611                    DH5
1333 003460  061464                    DT5
1334 003462  000000                    0
1335
1336 003464  056050                    EM30
1337 003466  056112                    DH30
1338 003470  061512                    DT30
1339 003472  000000                    0
1340
1341 003474  061105                    EM31
1342 003476  061235                    DH31
1343 003500  061670                    DT31
1344 003502  000000                    0
```

```
   1345
   1346 003504  061160                    EM32
   1347 003506  061235                    DH31
   1348 003510  061670                    DT31
   1349 003512  000000                    0
   1350
   1351 003514  055366                    EM33
   1352 003516  055610                    DH33
   1353 003520  061526                    DT33
   1354 003522  000000                    0
   1355
   1356 003524  055657                    EM34
   1357 003526  055744                    DH34
   1358 003530  061550                    DT34
   1359 003532  000000                    0
   1360
   1361 003534  056157                    EM35
   1362 003536  045730                    DH7
   1363 003540  061430                    DT7
   1364 003542  000000                    0
   1365
   1366 003544  056230                    EM36
   1367 003546  045611                    DH5
   1368 003550  061464                    DT5
   1369 003552  000000                    0
   1370
   1371 003554  056276                    EM37
   1372 003556  045611                    DH5
   1373 003560  061464                    DT5
   1374 003562  000000                    0
   1375
   1376 003564  056337                    EM40
   1377 003566  045611                    DH5
   1378 003570  061464                    DT5
   1379 003572  000000                    0
   1380
   1381 003574  056402                    EM41
   1382 003576  045611                    DH5
   1383 003600  061464                    DT5
   1384 003602  000000                    0
   1385
   1386 003604  056445                    EM42
   1387 003606  045730                    DH7
   1388 003610  061430                    DT7
   1389 003612  000000                    0
   1390
   1391 003614  056506                    EM43
   1392 003616  045611                    DH5
   1393 003620  061464                    DT5
   1394 003622  000000                    0
   1395
   1396 003624  056563                    EM44
   1397 003626  045611                    DH5
   1398 003630  061464                    DT5
   1399 003632  000000                    0
   1400
   1401 003634  056622                    EM45
```

ERROR POINTER TABLE SEQ 0030

```
1402 003636  045611                          DH5
1403 003640  061464                          DT5
1404 003642  000000                          0
1405
1406 003644  056667                          EM46
1407 003646  045730                          DH7
1408 003650  061430                          DT7
1409 003652  000000                          0
1410
1411 003654  056741                          EM47
1412 003656  045611                          DH5
1413 003660  061464                          DT5
1414 003662  000000                          0
1415
1416 003664  056767                          EM50
1417 003666  045730                          DH7
1418 003670  061430                          DT7
1419 003672  000000                          0
1420
1421 003674  000000                          0
1422 003676  000000                          0
1423 003700  061572                          DT51
1424 003702  061724                          DF51
1425
1426 003704  000000                          0
1427 003706  000000                          0
1428 003710  061606                          DT52
1429 003712  061730                          DF52
1430
1431 003714  050042                          EM53
1432 003716  000000                          0
1433 003720  000000                          0
1434 003722  000000                          0
1435
1436 003724  050000                          EM54
1437 003726  000000                          0
1438 003730  000000                          0
1439 003732  000000                          0
1440
1441 003734  050305                          EM55
1442 003736  045611                          DH5
1443 003740  061464                          DT5
1444 003742  000000                          0
1445
1446 003744  050360                          EM56
1447 003746  045611                          DH5
1448 003750  061464                          DT5
1449 003752  000000                          0
1450
1451
1452 003754  050463                          EM57
1453 003756  050512                          DH57
1454 003760  061624                          DT57
1455 003762  000000                          0
1456
1457 003764  050536                          EM60
1458 003766  050676                          DH61
```

```
1459 003770  061634                    DT61
1460 003772  000000                    0
1461
1462 003774  050572                    EM61
1463 003776  056022                    DH40
1464 004000  061420                    DT23
1465 004002  000000                    0
1466
1467 004004  056506                    EM43
1468 004006  050751                    DH62
1469 004010  061654                    DT62
1470 004012  000000                    0
1471
1472 004014  056445                    EM42
1473 004016  050676                    DH61
1474 004020  061634                    DT61
1475 004022  000000                    0
1476
1477 004024  056050                    EM30
1478 004026  050676                    DH61
1479 004030  061634                    DT61
1480 004032  000000                    0
1481
1482 004034  050641                    EM63
1483 004036  050751                    DH62
1484 004040  061654                    DT62
1485 004042  000000                    0
1486
1487 004044  056622                    EM45
1488 004046  050751                    DH62
1489 004050  061654                    DT62
1490 004052  000000                    0
1491
1492 004054  051010                    EM62
1493 004056  050751                    DH62
1494 004060  061654                    DT62
1495 004062  000000                    0
1496
1497 004064  056337                    EM40
1498 004066  050751                    DH62
1499 004070  061654                    DT62
1500 004072  000000                    0
1501
1502 004074  056276                    EM37
1503 004076  050751                    DH62
1504 004100  061654                    DT62
1505 004102  000000                    0
1506
1507 004104  056767                    EM50
1508 004106  050676                    DH61
1509 004110  061634                    DT61
1510 004112  000000                    0
1511
1512 004114  051051                    EM73
1513 004116  050751                    DH62
1514 004120  061654                    DT62
1515 004122  000000                    0
```

```
1516
1517 004124  051123                EM74    ;.ASCIZ /INTERRUP* WON'T CLEAR/ <12><15>
1518 004126  000000                0
1519 004130  000000                0
1520 004132  000000                0
1521
1522 004134  051153                EM75    ;.ASCIZ /CBIT FAILS TO CLEAR MODULE/<12><15>
1523 004136  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1524 004140  061340                DT100   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1525 004142  061711                DF100   ;.BYTE  0,0,0,0,0,1
1526
1527 004144  051210                EM76    ;.ASCIZ %COUNTER R/W FAILURE%<12><15>
1528 004146  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1529 004150  061340                DT100   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1530 004152  061711                DF100   ;.BYTE  0,0,0,0,0,1
1531
1532 004154  051236                EM77    ;.ASCIZ /IBIT FAILS TO INITIALIZE MODULE/<12><15>
1533 004156  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1534 004160  061340                DT100   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1535 004162  061711                DF100   ;.BYTE  0,0,0,0,0,1
1536
1537 004164  051300                EM100   ;.ASCIZ /COUNTER FAILS TO HALT AFTER INTERRUPT/<12><15>
1538 004166  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1539 004170  061340                DT100   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1540 004172  061711                DF100   ;.BYTE  0,0,0,0,0,1
1541
1542 004174  051350                EM101   ;.ASCIZ /COUNTER FAILS TO COUNT/<12><15>
1543 004176  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1544 004200  061340                DT100   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1545 004202  061711                DF100   ;.BYTE  0,0,0,0,0,1
1546
1547 004204  051401                EM102   ;.ASCIZ /WRONG COUNT IN COUNTER AFTER TRANSITION/<12><15>
1548 004206  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1549 004210  061340                DT100   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1550 004212  061711                DF100   ;.BYTE  0,0,0,0,0,1
1551
1552 004214  051453                EM103   ;.ASCIZ /NO INTERRUPT/<12><15>
1553 004216  045425                DH101   ;.ASCIZ /TST#   PC      MODULE  ADDRESS $PASS/
1554 004220  061324                DT101   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $PASS,
1555 004222  061717                DF101   ;.BYTE  0,0,0,0,1
1556
1557 004224  051472                EM104   ;.ASCIZ /RIF BIT NOT CLEARING INTERRUPT/<12><15>
1558 004226  000000                0
1559 004230  000000                0
1560 004232  000000                0
1561
1562 004234  051533                EM105   ;.ASCIZ /NO CLOCK/<15><12>
1563 004236  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1564 004240  061340                DT100   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1565 004242  061711                DF100   ;.BYTE  0,0,0,0,0,1
1566
1567 004244  051560                EM106   ;.ASCIZ /COUNTER NOT AT INITIALIZED FREQ/<12><15>
1568 004246  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1569 004250  061340                DT100   ;.WORD  $TESTN, $ERRPC  $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1570 004252  061711                DF100   ;.BYTE  0,0,0,0,0,1
1571
1572 004254  051622                EM107   ;.ASCIZ /TIME BASE ERROR/<15><12>
```

```
1573 004256  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1574 004260  061340                DT100   ;.WORD  $TESTN, $ERRPC, $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1575 004262  061711                DF100   ;.BYTE  0,0,0,0,0,1
1576
1577 004264  051644                EM110   ;.ASCIZ /UNEXPECTED COUNTER INTERRUPT/<15><12>
1578 004266  055422                DH110   ;.ASCIZ /TST#   PC      ADDRESS COUNTER MODULE  ITER/
1579 004270  061266                DT110   ;.WORD  $TESTN, $ERRPC, TADDR,  CNUM,   $MUT,   ITER,
1580 004272  061711                DF100   ;.BYTE  0,0,0,0,0,1
1581
1582 004274  051703                EM111   ;.ASCIZ /MODULE READ-WRITE FAILURE/<12><15>
1583 004276  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1584 004300  061340                DT100   ;.WORD  $TESTN, $ERRPC, $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1585 004302  061711                DF100   ;.BYTE  0,0,0,0,0,1
1586
1587 004304  051737                EM112   ;.ASCIZ /MODULE SELF-CLEAR ERROR/<12><15>
1588 004306  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1589 004310  061340                DT100   ;.WORD  $TESTN, $ERRPC, $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1590 004312  061711                DF100   ;.BYTE  0,0,0,0,0,1
1591
1592 004314  051771                EM113   ;.ASCIZ /INCORRECT COUNT IN COUNTER/<12><15>
1593 004316  055466                DH102   ;.ASCIZ /TST#   PC      ADDRESS COUNTER GDDAT   BDDAT   ITER/
1594 004320  061304                DT102   ;.WORD  TNUMB,  $ERRPC, TADDR,  CNUM,   $GDDAT, $BDDAT, ITER,
1595 004322  061702                DF102   ;.BYTE  0,0,0,0,0,0,1
1596
1597 004324  052026                EM114   ;.ASCIZ /M5016 OVERFLOW FAILURE/<12><15>
1598 004326  055537                DH114   ;.ASCIZ /TST#   PC      ADDRESS COUNTER GDOVF   BDOVF   ITER/
1599 004330  061304                DT102   ;.WORD  TNUMB,  $ERRPC, TADDR,  CNUM,   $GDDAT, $BDDAT, ITER,
1600 004332  061702                DF102   ;.BYTE  0,0,0,0,0,0,1
1601
1602 004334  052057                EM115   ;.ASCIZ /OVERFLOW FAILS TO CLEAR/<12><15>
1603 004336  055466                DH102   ;.ASCIZ /TST#   PC      ADDRESS COUNTER GDDAT   BDDAT   ITER/
1604 004340  061304                DT102   ;.WORD  TNUMB,  $ERRPC, TADDR,  CNUM,   $GDDAT, $BDDAT, ITER,
1605 004342  061702                DF102   ;.BYTE  0,0,0,0,0,0,1
1606
1607 004344  052111                EM116   ;.ASCIZ /CSR CBIT FAILS TO CLEAR COUNTER/<12><15>
1608 004346  045360                DH100   ;.ASCIZ /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   $PASS/
1609 004350  061340                DT100   ;.WORD  $TESTN, $ERRPC, $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
1610 004352  061711                DF100   ;.BYTE  0,0,0,0,0,1
1611
1612 004354  052153                EM117   ;.ASCIZ <12><15>/ILLEGAL RADIX/<12><15>
1613 004356  055422                DH110   ;.ASCIZ /TST#   PC      ADDRESS COUNTER MODULE  ITER/
1614 004360  061266                DT110   ;.WORD  $TESTN, $ERRPC, TADDR,  CNUM,   $MUT,   ITER,
1615 004362  061711                DF100   ;.BYTE  0,0,0,0,0,1
```

ERROR POINTER TABLE

1617

```
 1620 004364                                    START:
                                                .SBTTL  INITIALIZE THE COMMON TAGS
                                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
      004364  012706  001114                            MOV     #$CMTAG,R0      ;;FIRST LOCATION TO BE CLEARED
      004370  005026                            10$:    CLR     (R6)+           ;;CLEAR MEMORY LOCATION
      004372  022706  001154                            CMP     #SWR,R6 ;;DONE?
      004376  001374                                    BNE     .-6             ;;LOOP BACK IF NO
      004400  012706  001100                            MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                                ;;INITIALIZE A FEW VECTORS
      004404  012737  042322  000020                    MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
      004412  012737  000340  000022                    MOV     #340,@#IOTVEC+2 ;;LEVEL 7
      004420  012737  043172  000030                    MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
      004426  012737  000340  000032                    MOV     #340,@#EMTVEC+2 ;;LEVEL 7
      004434  012737  045010  000034                    MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
      004442  012737  000340  000036                    MOV     #340,@#TRAPVEC+2;LEVEL 7
      004450  013737  006710  006702                    MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
      004456  005037  001174                            CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
      004462  112737  000001  001131                    MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
      004470  012737  004470  001124                    MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
      004476  013746  000004                            MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
      004502  012737  004536  000004                    MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
      004510  012737  177570  001154                    MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
      004516  012737  177570  001156                    MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
      004524  022777  177777  174422                    CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
      004532  001012                                    BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                                ;;AND  THE HARDWARE SWR IS NOT -1
      004534  000403                                    BR      65$             ;;BRANCH IF NO TIMEOUT
      004536  012716  004544                64$:        MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
      004542  000002                                    RTI
      004544  012737  000176  001154        65$:        MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
      004552  012737  000174  001156                    MOV     #DISPREG,DISPLAY
      004560  012637  000004                66$:        MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
      004564  005037  001210                            CLR     $PASS           ;;CLEAR PASS COUNT
      004570  132737  000200  001223                    BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
      004576  001403                                    BEQ     67$             ;;YES,USE NON-APT SWITCH
      004600  012737  001224  001154                    MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
      004606                                67$:
```

```
1622                                        .SBTTL  DIAGNOSTICS MONITOR
1623                                        .SBTTL  TYPE PROGRAM NAME
                                            ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
     004606  005227  177777                       INC     #-1             ;;FIRST TIME?
     004612  001031                               BNE     68$             ;;BRANCH IF NO
     004614  022737  006724  000042               CMP     #$ENDAD,@#42    ;;ACT-11?
     004622  001425                               BEQ     68$             ;;BRANCH IF YES
     004624  104401  004632                       TYPE    ,69$            ;;TYPE ASCIZ STRING
     004630  000422                               BR      68$             ;;GET OVER THE ASCIZ
                                            ;;69$:  .ASCIZ  <CRLF>/DIGITAL IO SYSTEM TEST VERSION 1/<CRLF>
     004676                                 68$:
1624 004676  012737  004714  000004       SETCLK: MOV     #1$,ERRVEC
1625 004704  052777  000100  175346               BIS     #BIT6,@CLKADR   ;ENABLE LINE CLOCK INTERRUPT FOR UNIBUS COMPUTERS
1626 004712  000401                               BR      2$
1627 004714  022626                         1$:   CMP     (SP)+,(SP)+     ;TRAP IF LSI11
1628 004716  012737  005404  000004       2$:   MOV     #TMOVEC,ERRVEC          ;RESTORE ERROR VECTOR
1629 004724  012706  001100                       MOV     #1100,R6
1630 004730  005777  175302                       TST     @XXDP           ;ARE WE IN XXDP CHAIN MODE?
1631 004734  001404                               BEQ     6$              ;NO
1632 004736  005037  001210                       CLR     $PASS
1633 004742  000137  006002                       JMP     AUTO
1634
1635 004746  132737  000001  001222       6$:   BITB    #BIT0,$ENV      ;ARE WE UNDER APT
1636 004754  001413                               BEQ     4$              ;NO
1637 004756  132737  000040  001223               BITB    #BIT5,$ENV+1    ;INHIBIT PRINT?
1638 004764  001403                               BEQ     3$
1639 004766  052737  020000  000176               BIS     #BIT13,SWREG
1640 004774  005037  001210                 3$:   CLR     $PASS
1641 005000  000137  006002                       JMP     AUTO
1642 005004                                 4$:
     005004  012777  040722  175342               MOV     #KBINT,@KBVEC
     005012  152777  000100  174140               BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
1643 005020  012746  000000                       MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     005024  012746  005032                       MOV     #64$,-(SP)
     005030  000002                               RTI
     005032  000240                         64$:  NOP
1644 005034  000424                               BR      MONIT
1645 005036  104401  046732               TYPOPT: TYPE    ,M13            ;TEST OPTIONS
1646 005042  104401  046754                       TYPE    ,M14            ;S - SYSTEM TEST
1647 005046  104401  046775                       TYPE    ,M15            ;D - DIGITAL MODULE TEST
1648 005052  104401  047044                       TYPE    ,M17            ;M - MAP OF DBUS DEVICES
1649 005056  104401  047117                       TYPE    ,M19            ;I - IOCM TEST
1650 005062  104401  047145                       TYPE    ,M21            ;T - SET SWREG
1651 005066  104401  047353                       TYPE    ,M24            ;F - FIELD TEST
1652 005072  104401  047373                       TYPE    ,M25            ;W - WRAP AROUND TEST
1653 005076  104401  047412                       TYPE    ,M26            ;L - LOAD LOOP COUNT
1654 005102  104401  047501                       TYPE    ,M28            ;A - ANALOG MODULE TEST
1655
```

```
1657 005106  012706  001100          MONIT:  MOV     #1100,R6        ;SET STACK
1658 005112  005037  002312                  CLR     AFLAG
1659 005116  142777  000100  174034          BICB    #BIT6,@$TKS     ;DISABLE KB INTERRUPT
1660 005124  104401  052672                  TYPE    ,MASS12         ;TYPE TEST OPTION TO RUN
1661 005130  104406                          RDCHR
1662 005132  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1663 005134  010037  002432                  MOV     R0,ANSW
1664 005140  104401  002432                  TYPE    ,ANSW           ;ECHO CHAR
1665 005144  022700  000101                  CMP     #101,R0         ;A ?
1666 005150  001015                          BNE     2$
1667 005152  104410                          RDOCT
1668 005154  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
1669 005156  122701  000014                  CMPB    #14,R1          ;IS IT A014 - A/D
1670 005162  001002                          BNE     11$
1671 005164  000137  032324                  JMP     ATOD            ;JUMP TO A/D MONITOR
1672 005170  122701  000630          11$:    CMPB    #630,R1         ;IS IT A630 - DAC
1673 005174  001077                          BNE     10$             ;UNKNOW MODULE TYPE
1674 005176  004737  026732                  JSR     PC,DACSTR
1675 005202  000741                          BR      MONIT
1676 005204  104401  052175          2$:     TYPE    ,MASS0          ;CR,LF
1677 005210  022700  000110                  CMP     #'H,R0          ;A HELP COMMAND ?
1678 005214  001710                          BEQ     TYPOPT
1679 005216  022700  000123                  CMP     #123,R0         ;IS IT S?
1680 005222  001007                          BNE     1$              ;STAR AUTO TEST
1681 005224  012737  000001  002312          MOV     #1,AFLAG        ;SET MONITOR LOC FOR AUTO RUN
1682 005232  005037  001210                  CLR     $PASS
1683 005236  000137  006002                  JMP     AUTO
1684 005242  022700  000104          1$:     CMP     #104,R0
1685 005246  001002                          BNE     3$
1686 005250  000137  005416                  JMP     DIGIT           ;TEST DIGITAL MODULE
1687 005254  022700  000115          3$:     CMP     #115,R0         ;M ?
1688 005260  001002                          BNE     4$
1689 005262  000137  007370                  JMP     MAPE            ;MAP THE SYSTEM
1690 005266  022700  000130          4$:     CMP     #130,R0         ;X ?
1691 005272  001002                          BNE     5$
1692 005274  000137  032124                  JMP     EXERC           ;START EXERCISER
1693 005300  022700  000111          5$:     CMP     #111,R0         ;I ?
1694 005304  001002                          BNE     6$
1695 005306  000137  010142                  JMP     IOCM            ;TEST IOCM
1696 005312  022700  000124          6$:     CMP     #124,R0         ;T ?
1697 005316  001002                          BNE     7$
1698 005320  000137  005754                  JMP     SWREGS          ;SET SOFTWARE SWITCH REG.
1699 005324  022700  000106          7$:     CMP     #106,R0         ;F ?
1700 005330  001002                          BNE     8$
1701 005332  000137  007170                  JMP     FIELD           ;START FIELD TEST
1702 005336  022700  000127          8$:     CMP     #127,R0         ;W ?
1703 005342  001002                          BNE     9$
1704 005344  000137  006740                  JMP     LOOP            ;WRAP AROUND  TEST
1705 005350  022700  000114          9$:     CMP     #114,R0         ;LOAD LOOP COUNT
1706 005354  001007                          BNE     10$
1707 005356  104401  047444                  TYPE    ,M27            ;NUMBER OF $PASSATIONS =
1708 005362  104410                          RDOCT
1709 005364  012637  002320                  MOV     (SP)+,PASCNT    ;;POP STACK INTO PASCNT
1710 005370  000137  005106                  JMP     MONIT
1711 005374  104401  047136          10$:    TYPE    ,M20            ;?
1712 005400  000137  005106                  JMP     MONIT           ;OPERATOR TYPED WRONG CHARACTER
1713
```

```
 1714 005404  104401  054620          TMCVEC: TYPE    ,MASS45         ;UNEXPECTED TIMEOUT ERROR
 1715 005410  011646                         MOV     (R6),-(SP)      ;;SAVE (R6) FOR TYPEOUT
      005412  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1716 005414  000000                         HALT
```

```
1719                                            .SBTTL  DIGITAL MODULE MONITOR
1720 005416                                     DIGIT:
     005416  012777  040722  174730                    MOV     #KBINT,@KBVEC
     005424  152777  000100  173526                    BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
1721 005432  004737  040646                            JSR     PC,CNTRC
1722 005436  104401  052744                            TYPE    ,MASS13         ;TYPE ADDRESS OF MUT
1723 005442  104410                                    RDOCT
1724 005444  012637  002244                            MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
1725 005450  023737  002244  002276                    CMP     TADDR,BASE
1726 005456  103757                                    BLO     DIGIT
1727 005460  013746  000004                            MOV     ERRVEC,-(SP)    ;SAVE ERROR VECTOR
1728 005464  012737  005740  000004                    MOV     #5$,ERRVEC      ;SET LOC 4
1729 005472  004737  032024                            JSR     PC,KLEER        ;CLEAR THE IOCM
1730 005476  105777  174560                            TSTB    @CSR            ;MAKE SURE IOCM IS OK
1731 005502  001404                                    BEQ     17$
1732 005504  104010                                    EMT     10
1733 005506  012637  000004                            MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1734 005512  000207                                    RTS     PC
1735 005514  112777  000004  174540  17$:               MOVB    #GBIT,@CSR      ;SET GENERIC BIT
1736 005522  117700  174516                            MOVB    @TADDR,R0       ;R0=GENERIC CODE OF MUT
1737 005526  042700  177400                            BIC     #177400,R0
1738 005532  012701  002150                            MOV     #MODUL,R1       ;ADDRESS OF TEST
1739 005536  012703  002100                            MOV     #MUT,R3
1740 005542  012702  002026                            MOV     #GENER,R2       ;TABLE OF GENERIC CODES
1741 005546  020022                             3$:     CMP     R0,(R2)+
1742 005550  001050                                    BNE     1$
1743 005552  011337  002240                            MOV     (R3),$MUT
1744 005556  004737  032024                            JSR     PC,KLEER
1745 005562  011137  002254                            MOV     (R1),DMUT
1746 005566  005037  001210                     9$:    CLR     $PASS
1747 005572  004777  174456                     4$:    JSR     PC,@DMUT        ;TEST MODULE
1748 005576  004737  040646                            JSR     PC,CNTRC        ;CONTROL C?
1749 005602  005737  002320                            TST     PASCNT          ;IF ZERO LOOP ON TEST
1750 005606  001771                                    BEQ     4$
1751 005610  005237  001210                            INC     $PASS
1752 005614  023737  002320  001210                    CMP     PASCNT,$PASS
1753 005622  001363                                    BNE     4$
1754 005624  032737  020000  000176                    BIT     #BIT13,SWREG    ;INHIBIT PRINT ?
1755 005632  001005                                    BNE     7$
1756 005634  013746  002320                            MOV     PASCNT,-(SP)    ;;SAVE PASCNT FOR TYPEOUT
     005640  104402                                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1757 005642  104401  046640                            TYPE    ,M11            ;# OF  PASSES
1758 005646  032737  040000  000176  7$:               BIT     #BIT14,SWREG    ;LOOP ?
1759 005654  001344                                    BNE     9$
1760 005656  104401  053023                            TYPE    ,MASS15         ;END OF MODULE TEST
1761 005662  004737  032024                            JSR     PC,KLEER
1762 005666  000137  005106                            JMP     MONIT
1763 005672  062701  000002                     1$:    ADD     #2,R1
1764 005676  062703  000002                            ADD     #2,R3
1765 005702  021227  000261                            CMP     (R2),#261       ;END OF TABLE FOR 'DIGITAL' TESTS ?
1766 005706  001317                                    BNE     3$
1767 005710  104401  052636                            TYPE    ,MASS11         ;UNKNOWN GENERIC CODE
1768 005714  010046                                    MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
     005716  104402                                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1769 005720  104401  052175                            TYPE    ,MASS0          ;CR,LF
1770 005724  012637  000004                            MOV     (SP)+,ERRVEC    ;RESTORE ERRVEC
1771 005730  004737  032024                            JSR     PC,KLEER
```

SEQ 0040

```
 1772 005734  000137  005106                    JMP     MONIT
 1773 005740  022626                    5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
 1774 005742  104024                            EMT     24
 1775 005744  012637  000004                    MOV     (SP)+,ERRVEC
 1776 005750  000137  005106                    JMP     MONIT
```

```
                                        .SBTTL  SET SOFTWARE SWITCH REGISTER
                                
005754  104401  047164      SWREGS: TYPE    ,M22            ;SWICHES OPTIONS
005760  104401  047341              TYPE    ,M23            ;SW REG =
005764  104410                      RDOCT
005766  012637  000176              MOV     (SP)+,SWREG     ;;POP STACK INTO SWREG
005772  104401  052175              TYPE    ,MASSO          ;CR,LF
005776  000137  005106              JMP     MONIT
```

```
                                        .SBTTL  AUTO TEST MONITOR

                                                ;THIS PART OF A PROGRAM IS A MONITOR FOR AUTOMATIC TEST.
                                                ;FIRST IT TEST IOCM. THEN IT GENERATES A TABLE AND B TABLE.
                                                ;THEN IT TESTS EACH INDIVIDUAL I/O MODULE CONNECTED TO DBUS.
                                                ;REGISTERS R0 & R1 SERVE AS THE POINTERS FOR MUT

006002                                  AUTO:
006002  012777  040722  174344                  MOV     #KBINT,@KBVEC
006010  152777  000100  173142                  BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
006016  004737  010210                          JSR     PC,TIOCM
006022  004737  032024                          JSR     PC,KLEER        ;CLEAR THE IOCM
006026  105777  174230                          TSTB    @CSR            ;MAKE SURE IOCM IS OK
006032  001402                                  BEQ     7$
006034  104010                                  EMT     10
006036  000207                                  RTS     PC
006040  013746  000004                  7$:     MOV     @#ERRVEC,-(SP)  ;SAVE ERROR VECTOR
006044  012737  006240  000004                  MOV     #1$,ERRVEC      ;SET ERROR VECTOR
006052  013700  002276                          MOV     BASE,R0         ;R0-171000
006056  112777  000004  174176                  MOVB    #GBIT,@CSR      ;SET GENERIC CODE ENABLE
006064  012702  001326                          MOV     #ATABL,R2
006070  012705  001566                          MOV     #BTABL,R5
006074  111001                          5$:     MOVB    (R0),R1         ;R1 GENERIC CODE,READ GENERIC CODE
006076  042701  177400                          BIC     #177400,R1
006102  004737  040420                          JSR     PC,TABLE        ;FIND WHAT I/O IT IS
006106  000164  006112                          JMP     3$(R4)
006112  000426                          3$:     BR      10$
006114  000425                                  BR      10$
006116  000426                                  BR      12$
006120  000426                                  BR      13$
006122  000422                                  BR      10$
006124  000423                                  BR      12$
006126  000423                                  BR      13$
006130  000422                                  BR      13$
006132  000420                                  BR      12$             ;M5012-YA
006134  000415                                  BR      10$             ;M6010-YA
006136  000414                                  BR      10$             ;M5014 INPUT COUNTER
006140  000413                                  BR      10$             ;M5014 INPUT COUNTER
006142  000412                                  BR      10$             ;M5014 INPUT COUNTER
006144  000411                                  BR      10$             ;M5014 INPUT COUNTER
006146  000412                                  BR      12$             ;M5016 QUAD 8 BIT COUNTER
006150  000407                                  BR      10$             ;M6014 DUAL 16 BIT COUNTER
006152  000406                                  BR      10$             ;G670 MOTHERBOARD
006154  000403                                  BR      9$
006156  000404                                  BR      10$
006160  000403                                  BR      10$
006162  000405                                  BR      13$
006164  062700  000004                  9$:     ADD     #4,R0
006170  005200                          10$:    INC     R0
006172  005200                          11$:    INC     R0
006174  005200                          12$:    INC     R0
006176  005200                          13$:    INC     R0
006200  020037  002264                          CMP     R0,IAR
006204  103733                                  BLO     5$
006206  005012                                  CLR     (R2)
006210  022702  001326                          CMP     #ATABL,R2
```

```
1849 006214  001017                          BNE     6$
1850 006216  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1851 006222  142777  000004  174032          BICB    #GBIT,@CSR
1852 006230  104401  055041                  TYPE    ,MASS50         ;NO I/O
1853 006234  000137  006524                  JMP     26$
1854 006240  022626                  1$:     CMP     (SP)+,(SP)+
1855 006242  005200                          INC     R0
1856 006244  020037  002264                  CMP     R0,IAR
1857 006250  103711                          BLO     5$
1858 006252  005015                          CLR     (R5)
1859 006254  012637  000004          6$:     MOV     (SP)+,@#ERRVEC  ;START TESTING INDIVIDUAL I/O
1860 006260  142777  000004  173774          BICB    #GBIT,@CSR      ;CLEAR G BIT
1861 006266  004737  010210          25$:    JSR     PC,TIOCM
1862 006272  005000                          CLR     R0
1863 006274  005001                  22$:    CLR     R1
1864 006276  026061  001566  002026  23$:    CMP     BTABL(R0),GENER(R1)
1865 006304  001416                          BEQ     21$
1866 006306  062701  000002                  ADD     #2,R1
1867 006312  005761  002026                  TST     GENER(R1)
1868 006316  001367                          BNE     23$
1869 006320  016037  001566  003040          MOV     BTABL(R0),TEMP2
1870 006326  016037  001326  003042          MOV     ATABL(R0),TEMP3
1871 006334  104057                          EMT     57
1872 006336  000137  005106                  JMP     MONIT
1873 006342  016137  002150  002254  21$:    MOV     MODUL(R1),DMUT
1874 006350  016037  001326  002244          MOV     ATABL(R0),TADDR
1875 006356  016137  002100  002240          MOV     MUT(R1),$MUT
1876 006364  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1877 006366  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1878 006370  004777  173660                  JSR     PC,@DMUT        ;TEST THIS MODULE
1879 006374  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1880 006376  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
1881 006400  022761  000014  002100          CMP     #14,MUT(R1)             ;WAS IT A014
1882 006406  001037                          BNE     30$
1883 006410  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1884 006412  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1885 006414  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
1886 006420  012737  006642  000004          MOV     #31$,ERRVEC
1887 006426  012737  000040  002224          MOV     #40,MXNUM               ;SET FIRST MUX NUMBER
1888 006434  152777  000001  173620  32$:    BISB    #RBIT,@CSR
1889 006442  113777  002224  174402          MOVB    MXNUM,@STAT1
1890 006450  105777  174376                  TSTB    @STAT1                  ;CHECK IF RESPONDS
1891 006454  004737  037060                  JSR     PC,MUX1                 ;TEST MUX
1892 006460  062737  000040  002224  33$:    ADD     #40,MXNUM               ;DO NEXT MUX
1893 006466  022737  000400  002224          CMP     #400,MXNUM              ;LAST ONE?
1894 006474  001357                          BNE     32$
1895 006476  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1896 006502  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
1897 006504  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1898 006506  062700  000002          30$:    ADD     #2,R0
1899 006512  005760  001566                  TST     BTABL(R0)
1900 006516  001266                          BNE     22$
1901 006520  004737  040646                  JSR     PC,CNTRC        ;CONTROL C ?
1902 006524  004737  032024          26$:    JSR     PC,KLEER
1903 006530  005777  173502                  TST     @XXDP           ;RUNNING UNDER XXDP CHAIN MODE?
1904 006534  001051                          BNE     LOGIC
1905 006536  132737  000001  001222          BITB    #BIT0,$ENV      ;RUNNINNG UNDER APT?
```

```
 1906 006544  001045                          BNE     LOGIC
 1907 006546  005737  002320                  TST     PASCNT
 1908 006552  001645                          BEQ     25$
 1909 006554  005237  001210                  INC     $PASS
 1910 006560  023737  002320  001210          CMP     PASCNT,$PASS    ;IS IT LAST $PASS
 1911 006566  001237                          BNE     25$
 1912 006570  005037  001210                  CLR     $PASS
 1913 006574  032737  020000  000176          BIT     #BIT13,SWREG    ;INHIBIT PRINT
 1914 006602  001005                          BNE     20$
 1915 006604  013746  002320                  MOV     PASCNT,-(SP)    ;;SAVE PASCNT FOR TYPLOUT
      006610  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1916 006612  104401  046640                  TYPE    ,M11            ;# OF  PASSES COMPLETED
 1917 006616  032737  040000  000176  20$:   BIT     #BIT14,SWREG    ;DO WE LOOP ON TESTS
 1918 006624  001220                          BNE     25$
 1919 006626  104401  053075          24$:    TYPE    ,MASS17         ;END OF SYSTEM TEST
 1920 006632  005037  002312                  CLR     AFLAG
 1921 006636  000137  005106                  JMP     MONIT
 1922
 1923
 1924 006642  022626                  31$:    CMP     (SP)+,(SP)+
 1925 006644  152777  000001  173410          BISB    #RBIT,@CSR
 1926 006652  105777  174174                  TSTB    @STAT1
 1927 006656  000700                          BR      33$
 1928
 1929
 1930 006660                          LOGIC:
                                      .SBTTL  END OF PASS ROUTINE
                                      ;;**************************************************************
                                      ;*INCREMENT THE PASS NUMBER ($PASS)
                                      ;*IF THERES A MONITOR GO TO IT
                                      ;*IF THERE ISN'T JUMP TO AUTO
006660                                $EOP:
006660  000004                                SCOPE
006662  005037  001116                        CLR     $TSTNM          ;;ZERO THE TEST NUMBER
006666  005237  001210                        INC     $PASS           ;;INCREMENT THE PASS NUMBER
006672  042737  100000  001210                BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
006700  005327                              DEC     (PC)+           ;;LOOP?
006702  000001                        $EOPCT: .WORD   1
006704  003013                                BGT     $DOAGN          ;;YES
006706  012737                                MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
006710  000001                        $ENDCT: .WORD   1
006712  006702                                $EOPCT
006714  013700  000042                $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
006720  001405                                BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
006722  000005                                RESET                   ;;CLEAR THE WORLD
006724  004710                        $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
006726  000240                                NOP                     ;;SAVE ROOM
006730  000240                                NOP                     ;;FOR
006732  000240                                NOP                     ;;ACT11
006734                                $DOAGN:
006734  000137                                JMP     @(PC)+          ;;RETURN
006736  006002                        $RTNAD: .WORD   AUTO
```

```
                                        .SBTTL  LOOP TEST
                                        ;THIS TEST ENABLE FIELD ENGINEER TO CONNECT OUTPUT MODULE TO INPUT MODULE
                                        ;OUTPUT TEST PATTERNS AND READ THEM BACK
                                        ;
                                        ;
006740  104401  053321          LOOP:   TYPE    ,MASS23         ;CONNECT OUTPUT TO INPUT MODULE
006744  012777  040722  173402          MOV     #KBINT,@KBVEC
006752  152777  000100  172200          BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
006760  104401  053234                  TYPE    ,MASS21         ;TYPE ADDRESS OF OUTPUT MUT
006764  104410                          RDOCT
006766  012637  002244                  MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
006772  104401  053267                  TYPE    ,MASS22         ;TYPE ADDRESS OF INPUT MUT
006776  104410                          RDOCT
007000  012637  002314                  MOV     (SP)+,TEMP1     ;;POP STACK INTO TEMP1
007004  112777  000004  173250          MOVB    #GBIT,@CSR      ;SET GENERIC BIT
007012  117700  173226                  MOVB    @TADDR,R0       ;GENERIC CODE OF OUTPUT MUT
007016  020027  000041                  CMP     R0,#41          ;IS IT M6010?
007022  001051                          BNE     1$              ;NOT M6010, ERROR
007024  117701  173264                  MOVB    @TEMP1,R1       ;GENERIC CODE OF INPUT MUT
007030  020127  000141                  CMP     R1,#141         ;IS IT M5010?
007034  001403                          BEQ     2$
007036  020127  000121                  CMP     R1,#121         ;IS IT M5011?
007042  001044                          BNE     6$
007044  005037  00121^          2$:     CLR     $PASS
007050  004737  041240          3$:     JSR     PC,LOPTST       ;TEST SUBROUTINE
007054  004737  040564                  JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
007060  004737  040646                  JSR     PC,CNTRC        ;CONTROL-C?
007064  005237  001210                  INC     $PASS
007070  023737  002320  001210          CMP     PASCNT,$PASS    ;
007076  001364                          BNE     3$
007100  032737  020000  000176          BIT     #BIT13,SWREG    ;INHIBIT PRINTOUT
007106  001005                          BNE     5$
007110  013746  002320                  MOV     PASCNT,-(SP)    ;;SAVE PASCNT FOR TYPEOUT
007114  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007116  104401  046640                  TYPE    ,M11            ;PASSES COMPLETED
007122  032737  040000  000176  5$:     BIT     #BIT14,SWREG    ;LOOP
007130  001345                          BNE     2$
007132  104401  053362                  TYPE    ,MASS25         ;END OF LOOP TEST
007136  004737  032024                  JSR     PC,KLEER
007142  000137  005106                  JMP     MONIT
007146  104401  053405          1$:     TYPE    ,MASS26         ;WRONG MODULE -OUTPUT MUST BE M6010
007152  000402                          BR      4$
007154  104401  053454          6$:     TYPE    ,MASS28         ;WRONG MODULE- INPUT MUST BE EITHER M5010 OR M5011
007160  004737  032024          4$:     JSR     PC,KLEER
007164  000137  005106                  JMP     MONIT
```

```
                                        .SBTTL  MODULE TEST
                                                
                                                ;THIS TEST ENABLE FIELD ENGINEER TO SELECT AND OUTPUT ANY DATA
                                                ;PATTERN TO OUTPUT MODULES,MONITOR AND PRINT DATA FROM INPUT MODULES
                                                ;
                                                ;
007170  104401  052744          FIELD:  TYPE    ,MASS13         ;TYPE ADDRESS OF MUT
007174  012777  040722  173152          MOV     #KBINT,@KBVEC
007202  152777  000100  171750          BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
007210  104410                          RDOCT
007212  012637  002244                  MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
007216  013746  000004                  MOV     ERRVEC, -(SP)   ;SAVE ERROR VECTOR
007222  012737  007354  000004          MOV     #5$,ERRVEC      ;SET LOC 4
007230  112777  000004  173024          MOVB    #GBIT,@CSR      ;SET GENERIC BIT
007236  117700  173002                  MOVB    @TADDR,R0       ;GENERIC CODE OF MUT
007242  042700  177400                  BIC     #177400,R0
007246  012701  002326                  MOV     #MOD,R1         ;ADDRESS OF TEST
007252  012702  002026                  MOV     #GENER,R2       ;TABLE OF GENERIC CODES
007256  020022                  3$:     CMP     R0,(R2)+
007260  001013                          BNE     2$
007262  004737  032024                  JSR     PC,KLEER
007266  011137  002254                  MOV     (R1),DMUT
007272  004777  172756                  JSR     PC,@DMUT        ;TEST MODULE
007276  104401  053166                  TYPE    ,MASS20         ;TYPE CONTROL-C TO RETURN TO MONITOR
007302  004737  040646          1$:     JSR     PC,CNTRC        ;CONTROL-C ?
007306  000775                          BR      1$
007310  062701  000002          2$:     ADD     #2,R1
007314  062703  000002                  ADD     #2,R3
007320  005711                          TST     (R1)
007322  001355                          BNE     3$
007324  104401  052636                  TYPE    ,MASS11         ;UNKNOWN GENERIC CODE
007330  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
007332  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007334  104401  052175                  TYPE    ,MASS0          ;CR,LF
007340  012637  000004                  MOV     (SP)+,ERRVEC    ;RESTORE ERRVEC
007344  004737  032024                  JSR     PC,KLEER
007350  000137  005106                  JMP     MONIT
007354  022626                  5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
007356  104024                          EMT     24
007360  012637  000004                  MOV     (SP)+,ERRVEC
007364  000137  005106                  JMP     MONIT
```

```
                                        .SBTTL  MAP OF DBUS


                                        ;THIS TEST WILL LIST ALL I/O INTERFACES CONNECTED TO IOCM
                                        ;IT WILL ALSO LIST QBUS ADDRESSES


007370  013746  000004        MAPE:   MOV     @#ERRVEC,-(SP)  ;SAVE ERROR VECTOR
007374  012737  010062  000004          MOV     #1$,ERRVEC      ;SET ERROR VECTOR
007402  012777  040722  172744          MOV     #KBINT,@KBVEC
007410  152777  000100  171542          BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
007416  013700  002276                  MOV     BASE,R0         ;R0=171000
007422  004737  032024                  JSR     PC,KLEER        ;CLEAR EVERYTHING FOR SURE
007426  004737  040564                  JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
007432  112777  000004  172622          MOVB    #GBIT,@CSR      ;SET GENERIC CODE ENABLE
007440  111001                  5$:     MOVB    (R0),R1         ;R1=GENERIC CODE,READ GENERIC CODE
007442  042701  177400                  BIC     #177400,R1
007446  004737  040374                  JSR     PC,GENCOD       ;FIND WHAT I/O IT IS
007452  104401  052200                  TYPE    ,MASS2          ;ADDRESS
007456  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
007460  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007462  000164  007466                  JMP     3$(R4)
007466  104401  052212          3$:     TYPE    ,MASS3          ;M5010
007472  000562                          BR      10$
007474  104401  052261                  TYPE    ,MASS4          ;M5011
007500  000557                          BR      10$
007502  104401  052330                  TYPE    ,MASS5          ;M5012
007506  000556                          BR      12$
007510  104401  052374                  TYPE    ,MASS6          ;M5013
007514  000554                          BR      13$
007516  104401  052426                  TYPE    ,MASS7          ;M6010
007522  000546                          BR      10$
007524  104401  052476                  TYPE    ,MASS8          ;M6011
007530  000545                          BR      12$
007532  104401  052537                  TYPE    ,MASS9          ;M6012
007536  000543                          BR      13$
007540  104401  052603                  TYPE    ,MASS10         ;M6013
007544  000540                          BR      13$
007546  104401  054742                  TYPE    ,MASS48         ;M5012-YA
007552  000534                          BR      12$
007554  104401  055001                  TYPE    ,MASS49         ;M6010-YA
007560  000527                          BR      10$
007562  104401  055071                  TYPE    ,MASS56         ;M5014
007566  000524                          BR      10$
007570  104401  055071                  TYPE    ,MASS56         ;M5014
007574  000521                          BR      10$
007576  104401  055071                  TYPE    ,MASS56         ;M5014
007602  000516                          BR      10$
007604  104401  055071                  TYPE    ,MASS56         ;M5014
007610  000513                          BR      10$
007612  104401  055122                  TYPE    ,MASS52         ;M5016
007616  000512                          BR      12$
007620  104401  055160                  TYPE    ,MASS53         ;M6014
```

```
2072 007624  000505                          BR      10$
2073 007626  104401  055217                  TYPE    ,MASS54         ;G670
2074 007632  000502                          BR      10$
2075 007634  104401  053537                  TYPE    ,MASS29         ;A630
2076 007640  000475                          BR      9$
2077 007642  104401  053602                  TYPE    ,MASS30         ;A014 SE
2078 007646  000412                          BR      30$
2079 007650  104401  053650                  TYPE    ,MASS31         ;A014 DM
2080 007654  000407                          BR      30$
2081 007656  104401  052636                  TYPE    ,MASS11         ;UNKNOWN GENERIC CODE
2082 007662  010146                          MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
     007664  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2083 007666  104401  052175                  TYPE    ,MASS0          ;CR,LF
2084 007672  000465                          BR      13$
2085 007674  012737  010120  000004  30$:    MOV     #2$,ERRVEC      ;SET NEW TIME OUT VECTOR
2086 007702  005004                          CLR     R4              ;SET MUX # POINTER
2087 007704  005003                          CLR     R3
2088 007706  062700  000002                  ADD     #2,R0           ;SET A/D TO SECOND ADDRESS
2089 007712  062703  000040          25$:    ADD     #40,R3          ;INC MUX #
2090 007716  005204                          INC     R4
2091 007720  032704  000010                  BIT     #BIT3,R4        ;LAST MUX?
2092 007724  001037                          BNE     24$
2093 007726  110310                          MOVB    R3,(R0)         ;SET MUX #
2094 007730  111001                          MOVB    (R0),R1
2095 007732  042701  177400                  BIC     #177400,R1      ;GET GENERIC CODE
2096 007736  104401  053724                  TYPE    ,MASS32         ;MUX #
2097 007742  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
     007744  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2098 007746  022701  000342                  CMP     #342,R1
2099 007752  001003                          BNE     21$
2100 007754  104401  053733                  TYPE    ,MASS33         ;A156 - SINGLE ENDED
2101 007760  000754                          BR      25$
2102
2103 007762  022701  000322          21$:    CMP     #322,R1
2104 007766  001003                          BNE     22$
2105 007770  104401  053763                  TYPE    ,MASS34         ;A156 - DIFFERENTIAL
2106 007774  000746                          BR      25$
2107
2108 007776  022701  000323          22$:    CMP     #323,R1
2109 010002  001003                          BNE     23$
2110 010004  104401  054020                  TYPE    ,MASS35         ;A157
2111 010010  000740                          BR      25$
2112
2113 010012                          23$:
     010012  010146                          MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
     010014  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2114 010016  104401  052636                  TYPE    ,MASS11         ;UNKNOWN GENERIC CODE
2115 010022  000733                          BR      25$
2116
2117 010024  012737  010062  000004  24$:    MOV     #1$,ERRVEC      ;RESTORE TIME-OUT VECTOR
2118 010032  000404                          BR      12$
2119
2120 010034  062700  000004          9$:     ADD     #4,R0
2121 010040  005200                  10$:    INC     R0              ;INC ADDRESS BY APPROPRIATE NUMBER OF BYTES
2122 010042  005200                  11$:    INC     R0
2123 010044  005200                  12$:    INC     R0
2124 010046  005200                  13$:    INC     R0
```

```
2125 010050  020037  002264                CMP     R0,IAR          ;LAST ADDRESS ?
2126 010054  103013                        BHIS    6$              ;YES, BRANCH
2127 010056  000137  007440                JMP     5$
2128 010062  022626                1$:     CMP     (SP)+,(SP)+     ;HERE IF ADDRESS IS NOT RESPONDING
2129 010064  004737  040646                JSR     PC,CNTRC
2130 010070  005200                        INC     R0              ;INC ADDRESS
2131 010072  020037  002264                CMP     R0,IAR          ;LAST ONE ?
2132 010076  001402                        BEQ     6$              ;
2133 010100  000137  007440                JMP     5$
2134 010104  012637  000004        6$:     MOV     (SP)+,@#ERRVEC  ;RESTORE ERROR VECTOR
2135 010110  004737  032024                JSR     PC,KLEER
2136 010114  000137  005106                JMP     MONIT
2137
2138
2139 010120  022626                2$:     CMP     (SP)+,(SP)+             ;HERE IF NO MUXES
2140 010122  152777  000001  172132        BISB    #RBIT,@CSR              ;CLEAR HANG A/D
2141 010130  004737  040646                JSR     PC,CNTRC
2142 010134  105710                        TSTB    (R0)
2143 010136  000137  007712                JMP     25$
```

```
                                .SBTTL  IOCM TEST


                                        ;THIS PART OF DIAGNOSTIC TEST THE IOCM
                                        ;IT HAS 9 TESTS. IT CHECKS IF ALL THE BITS OF THE IOCM
                                        ;CAN BE SET AND CLEAR , CHECKS MAINTENANCE
                                        ;INTERRUPT AND CHECKS ALL ADDRESSES IN MAINTENANCE MODE
```

```
2153 010142                                IOCM:
     010142  013746  000004                     MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
2154 010146  012777  040722  172200           MOV     #KBINT,@KBVEC
     010154  152777  000100  170776           BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
2155 010162  004737  042224                   JSR     PC,SWLOOP
2156 010166  010210                           TIOCM                   ;DO IOCM TEST
2157 010170  104401  053052                   TYPE    ,MASS16         ;IOCM TEST PASSED
2158 010174  012637  000004                   MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
2159 010200  004737  032024                   JSR     PC,KLEER
2160 010204  000137  005106                   JMP     MONIT
2161
2162                                          ;THIS TEST CHECKS IF EACH BIT OF CSR
2163                                          ;IS CLEAR BY CBIT
2164 010210                                TIOCM:
                                           ;;*****************************************************************
     010210  000004                        TST1:   SCOPE
2165 010212  112737  000001  001206             MOVB    #1,$TESTN
2166 010220  012737  010270  000004             MOV     #10$,ERRVEC
2167 010226  112777  000074  172026             MOVB    #74,@CSR        ;SET ALL BITS
2168 010234  004737  032024                   JSR     PC,KLEER                ;SET CBIT
2169 010240  005037  001142                   CLR     $BDDAT
2170 010244  005037  001140                   CLR     $GDDAT
2171 010250  117737  172006  001142           MOVB    @CSR,$BDDAT     ;STORE CONTENTS OF CSR IN BDDAT
2172 010256  105737  001142                   TSTB    $BDDAT          ;IS CSR CLEAR
2173 010262  001405                           BEQ     2$              ;YES
2174 010264  104001                           EMT     1
2175 010266  000207                           RTS     PC              ;FATAL ERROR RETURN TO MONITOR
2176 010270                                10$:
     010270  104003                           EMT     3
2177 010272  022626                           CMP     (SP)+,(SP)+     ;RESTORE STACK
2178 010274  000207                           RTS     PC
2179 010276                                2$:
```

```
                                                ;THIS TEST CHECKS E BIT
                                        ;;******************************************************************
                                TST2:   SCOPE
 010276  000004
2183 010300  112737  000002  001206             MOVB    #2,$TESTN
2184 010306  012737  000100  001140             MOV     #EBIT,$GDDAT    ;SET TESTED BIT
2185 010314  113737  046633  045346             MOVB    M7,EM2X         ;SET EBIT IN ERROR MESSAGE
2186 010322  004737  040450                     JSR     PC,BITSET
2187 010326  104002                             EMT     2
2188 010330  005037  001140                     CLR     $GDDAT
2189 010334  004737  040450                     JSR     PC,BITSET
2190 010340  104002                             EMT     2
2191
2192                                            ;THIS TEST CHECKS MBIT
2193                                    ;;******************************************************************
                                TST3:   SCOPE
 010342  000004
2194 010344  112737  000003  001206             MOVB    #3,$TESTN
2195 010352  012737  000240  001140             MOV     #MBIT!FBIT,$GDDAT;SET MAITENANCE BIT
2196 010360  113737  046630  045346             MOVB    M6,EM2X         ;SET ERROR MESSAGE
2197 010366  004737  040450                     JSR     PC,BITSET
2198 010372  104002                             EMT     2
2199 010374  005037  001140                     CLR     $GDDAT
2200 010400  004737  040450                     JSR     PC,BITSET
2201 010404  104002                             EMT     2
2202
2203
2204                                            ;THIS TEST CHECKS DBIT
2205                                    ;;******************************************************************
                                TST4:   SCOPE
 010406  000004
2206 010410  112737  000004  001206             MOVB    #4,$TESTN
2207 010416  012737  000020  001140             MOV     #DBIT,$GDDAT
2208 010424  113737  046625  045346             MOVB    M5,EM2X         ;FIX ERROR MESSAGE
2209 010432  004737  040450                     JSR     PC,BITSET       ;SET D BIT
2210 010436  104002                             EMT     2
2211 010440  005037  001140                     CLR     $GDDAT
2212 010444  004737  040450                     JSR     PC,BITSET
2213 010450  104002                             EMT     2
2214 010452  005037  001140                     CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
2215 010456  112777  000074  171576             MOVB    #74,@CSR        ;SET FEW BITS AT CSR
2216 010464  142777  000020  171570             BICB    #DBIT,@CSR      ;CLEAR DBIT
2217 010472  005037  002222                     CLR     YLOOP           ;WAIT
 010476  005237  002222             64$:       INC     YLOOP
 010502  023727  002222  000007                 CMP     YLOOP,#7
 010510  001372                                 BNE     64$
2218 010512  117737  171544  001142             MOVB    @CSR,$BDDAT
2219 010520  132737  000177  001142             BITB    #177,$BDDAT
2220 010526  001401                             BEQ     1$              ;CHECK IF DBIT CLEARING CLEARS CSR
2221 010530  104016                             EMT     16
2222 010532  004737  032024             1$:     JSR     PC,KLEER
```

```
                                        ;THIS TEST CHECKS IBIT
                                ;;*****************************************************************
                                TST5:   SCOPE
                                        MOVB    #5,$TESTN
                                        MOV     #IBIT,$GDDAT
                                        MOVB    M4,EM2X         ;SET ERROR MESSAGE
                                        JSR     PC,BITSET       ;SET AND CHECK I BIT
                                        EMT     2
                                        CLR     $GDDAT
                                        JSR     PC,BITSET       ;CLEAR I BIT
                                        EMT     2
                                        JSR     PC,CLRINT       ;CLEAR ALL INTERRUPT CAUSED BY I BIT

                                        ;THIS TEST CHECKS GBIT
                                ;;*****************************************************************
                                TST6:   SCOPE
                                        MOVB    #6,$TESTN
                                        MOV     #GBIT,$GDDAT
                                        MOVB    M3,EM2X         ;FIX ERROR MESSAGE
                                        JSR     PC,BITSET       ;SET AND CHECK G BIT
                                        EMT     2
                                        CLR     $GDDAT
                                        JSR     PC,BITSET       ;CLEAR G BIT
                                        EMT     2


                                        ;THIS TEST CHECKS RBIT
                                ;;*****************************************************************
                                TST7:   SCOPE
                                        MOVB    #7,$TESTN
                                        MOV     #RBIT,$GDDAT
                                        MOVB    M2,EM2X         ;SET ERROR MESSAGE
                                        JSR     PC,BITSET       ;SET BIT
                                        EMT     2
                                        CLR     $GDDAT
                                        JSR     PC,BITSET       ;CLEAR BIT
                                        EMT     2
```

```
2260
2261
2262                                    ;THIS TEST CHECKS ALL BITS OF DBUS IN A MAINTENANCE MODE.
2263                                    ;IF MBIT IS SET AND CPU ADDRESSES ANY LOCATION BETWEEN
2264                                    ;171000 AND 171375 IT SHOULD READ BACK A LOWER
2265                                    ;BYTE OF AN ADDRESS.
2266
2267
2268                                ;;**********************************************************
     010716  000004                 TST10:  SCOPE
2269 010720  112737  000010  001206         MOVB    #10,$TESTN
2270 010726  112777  000040  171326         MOVB    #MBIT,@CSR      ;SET MAINTENANCE MODE
2271 010734  013700  002276                 MOV     BASE,R0
2272 010740  005001                         CLR     R1
2273 010742  005037  001140                 CLR     $GDDAT
2274 010746  005037  001142                 CLR     $BDDAT
2275 010752  111001                 1$:     MOVB    (R0),R1         ;READ FIRST ADDRESS
2276 010754  042701  177400                 BIC     #177400,R1
2277 010760  120001                         CMPB    R0,R1           ;CHECK IF DBUS=ADDRESS
2278 010762  001405                         BEQ     2$
2279 010764  110037  001140                 MOVB    R0,$GDDAT
2280 010770  110137  001142                 MOVB    R1,$BDDAT
2281 010774  104004                         EMT     4
2282 010776  005200                 2$:     INC     R0
2283 011000  122700  000376                 CMPB    #376,R0         ;IS IT THE LAST ADDRESS
2284 011004  001362                         BNE     1$
2285 011006  004737  032024                 JSR     PC,KLEER                ;CLEAR IOCM
```

```
                                        ;THIS TEST WILL CHECK MAINTENANCE INTERRUPT
                                        ;IF MBIT & EBIT ARE SET IOCM WILL GENERATE AN INTERRUPT
                                        ;AT LOCATION 234 AND IAR WILL HAVE LOWER BYTE
                                        ;OF CSR ADDRESS


                                ;;*****************************************************************
011012  000004                  TST11:  SCOPE
011014  112737  000011  001206          MOVB    #11,$TESTN
011022  012777  011116  171236          MOV     #5$,@VECTO      ;SET VECTOR ADDRESS
011030  012777  000340  171232          MOV     #PR7,@VECTOA    ;SET VECTOR+2 ADDRESS
011036  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
011042  012737  011234  000004          MOV     #8$,ERRVEC
011050  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
011054  012746  011062                  MOV     #64$,-(SP)
011060  000002                          RTI
011062  000240                  64$:    NOP
011064  112777  000140  171170  1$:     MOVB    #EBIT!MBIT,@CSR;START INTERRUPT
011072  005037  002222                  CLR     YLOOP           ;WAIT.
011076  005237  002222          65$:    INC     YLOOP
011102  023727  002222  177777          CMP     YLOOP,#-1
011110  001372                          BNE     65$
011112  104014                          EMT     14
011114  000433                          BR      6$
011116  022626                  5$:     CMP     (SP)+,(SP)+     ;INTERRUPT WORKED
011120  117737  171140  002304          MOVB    @IAR,TEMP       ;READ ADDRESS OF INTERRUPTING MODULE
011126  122737  000377  002304          CMPB    #377,TEMP       ;IAR SHOULD HAVE ALL ONES
011134  001411                          BEQ     7$
011136  012737  000377  001140          MOV     #377,$GDDAT
011144  005037  001142                  CLR     $BDDAT          ;ASSURE HIGH BYTE IS CLEAR
011150  113737  002304  001142          MOVB    TEMP,$BDDAT     ;PREPARE FOR ERROR TYPEOUT
011156  104006                          EMT     6
011160  152777  000001  171074  7$:     BISB    #RBIT,@CSR
011166  105777  171070                  TSTB    @CSR            ;CLEAR INTERRUPT
011172  132777  000200  171062          BITB    #EBIT,@CSR      ;CHECK IF INTERRUPT CLEAR
011200  001401                          BEQ     6$
011202  104015                          EMT     15
011204  004737  032024          6$:     JSR     PC,KLEER                ;CLEAR CSR
011210  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
011214  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
011220  012746  011226                  MOV     #66$,-(SP)
011224  000002                          RTI
011226  000240                  66$:    NOP
011230  000004                          SCOPE
011232  000207                          RTS     PC

011234  022626                  8$:     CMP     (SP)+,(SP)+
011236  104022                          EMT     22
011240  000761                          BR      6$
```

```
2330
2331                                    .SBTTL  DIGITAL MODUL TEST ,M5010
2332
2333                                            ;THIS TEST WILL CHECK M5010 MODULE
2334                                            ;32 BIT NONISOLATED DC SENSE
2335
2336
2337
2338 011242                             M5010:
2339                                    ;;*****************************************************************
     011242  000004                     TST12:  SCOPE
2340 011244  112737  000012  001206             MOVB    #12,$TESTN
2341 011252  152777  000020  171002             BISB    #DBIT,@CSR      ;SET DISABLE BIT
2342 011260  112737  000000  001140             MOVB    #0,$GDDAT       ;SET WHAT DATA SHOULD BE
     011266  012737  000004  002242             MOV     #4,BYTNUM       ;SET NUMBER OF BYTES
     011274  004737  040166                     JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
2343
2344 011300  152777  000010  170754             BISB    #IBIT,@CSR
2345 011306  112737  000377  001140             MOVB    #377,$GDDAT     ;SET WHAT DATA SHOULD BE
     011314  012737  000004  002242             MOV     #4,BYTNUM       ;SET NUMBER OF BYTES
     011322  004737  040166                     JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
2346 011326  142777  000010  170726             BICB    #IBIT,@CSR
2347 011334  004737  040564                     JSR     PC,CLRINT
2348 011340  000004                             SCOPE
2349 011342  000207                             RTS     PC
```

```
                                .SBTTL  DIGITAL MODULE TEST ,M5013


                                        ;THIS TEST CHECKS M5013 MODULE
                                        ;8 BIT AC SENSE


011344                          M5013:
                                ;;*****************************************************************
011344  000004                  TST13:  SCOPE
011346  112737  000013  001206          MOVB    #13,$TESTN
011354  152777  000020  170700          BISB    #DBIT,@CSR      ;SET DISABLE BIT
011362  112737  000000  001140          MOVB    #0,$GDDAT       ;SET WHAT DATA SHOULD BE
011370  012737  000001  002242          MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
011376  004737  040166                  JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT

011402  152777  000010  170652          BISB    #TBIT,@CSR      ;SET COMPLIM BIT
011410  004737  040564                  JSR     PC,CLRINT
011414  112737  000377  001140          MOVB    #377,$GDDAT     ;SET WHAT DATA SHOULD BE
011422  012737  000001  002242          MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
011430  004737  040166                  JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
011434  142777  000010  170620          BICB    #TBIT,@CSR
011442  004737  040564                  JSR     PC,CLRINT

011446  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
011452  012746  011460                  MOV     #64$,-(SP)
011456  000002                          RTI
011460  000240                  64$:    NOP
011462  012777  011566  170576          MOV     #5$,@VECTO      ;SET INTERRUPTVECTOR ADDRESS
011470  012777  000340  170572          MOV     #PR7,@VECTOA
011476  152777  000010  170556          BISB    #TBIT,@CSR      ;START INTERRUPT
011504  152777  000100  170550          BISB    #EBIT,@CSR      ;ENABLE INTERRUPT
011512  005037  002222                  CLR     YLOOP           ;WAIT
011516  005237  002222          65$:    INC     YLOOP
011522  023727  002222  177777          CMP     YLOOP,#-1
011530  001372                          BNE     65$
011532  104005                          EMT     5
                                    ;; /AN ERROR WILL OCCUR HERE IF INTERRUPTS
                                    ;; /ARE DISABLED BY A SWITCH SETTING ON THE MODULE
011534  142777  000100  170520  7$:     BICB    #EBIT,@CSR      ;CLEAR CSR
011542  142777  000010  170512          BICB    #TBIT,@CSR
011550  004737  040564                  JSR     PC,CLRINT       ;CLEAR ALL INTERRUPT
011554  012737  005404  000004          MOV     #TMOVEC,EPRVEC
011562  000004                          SCOPE
011564  000207                          RTS     PC              ;RETURN TO MONITOR

011566  022626                  5$:     CMP     (SP)+,(SP)+     ;RESET STACK
011570  005037  002314                  CLR     TEMP1
011574  113737  002277  002305          MOVB    BASE+1,TEMP+1   ;INITIALIZE TEMP
011602  012737  011776  000004          MOV     #4$,ERRVEC
011610  117737  170450  002304  6$:     MOVB    @IAR,TEMP       ;CHECK WHICH I/O INTERRUPTED
011616  152777  000001  170436          BISB    #RBIT,@CSR      ;HERE IF INTERRUPT
011624  123737  002304  002244          CMPB    TEMP,TADDR      ;IS IT MUT
011632  001411                          BEQ     2$
```

```
2397 011634  105777  170444                    TSTB    @IEMP
2398 011640  005237  002314                    INC     TEMP1           ;LOOP NO MORE THEN 400 TIMES
2399 011644  022737  000400  002314            CMP     #400,TEMP1
2400 011652  001356                            BNE     6$
2401 011654  104005                            EMT     5
2402 011656  105777  170362          2$:       TSTB    @IADDR          ;CLEAR INTERRUPT
2403 011662  132777  000200  170372            BITB    #IBIT,@CSR      ;CHECK IF INTERRUPT CLEAR
2404 011670  001413                            BEQ     8$
2405 011672  117737  170366  002304            MOVB    @IAR,TEMP
2406 011700  023737  002304  002244            CMP     TEMP,IADDR
2407 011706  001002                            BNE     9$
2408 011710  104015                            EMT     15
2409 011712  000710                            BR      7$
2410 011714  004737  040564          9$:       JSR     PC,CLRINT
2411 011720  132777  000010  170334  8$:       BITB    #TBIT,@CSR
2412 011726  001702                            BEQ     7$
2413 011730  012746  000000                    MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     011734  012746  011742                    MOV     #66$,-(SP)
     011740  000002                            RTI
     011742  000240          66$:      NOP
2414 011744  142777  000010  170310            BICB    #TBIT,@CSR      ;CHECK IF CLEARING TBIT CAUSE INTERRUPT
2415 011752  005037  002222                    CLR     YLOOP           ;WAIT
     011756  005237  002222          67$:      INC     YLOOP
     011762  023727  002222  177777            CMP     YLOOP,#-1
     011770  001372                            BNE     67$
2416 011772  104005                            EMT     5
2417 011774  000657                            BR      7$
2418 011776  022626                  4$:       CMP     (SP)+,(SP)+
2419 012000  104005                            EMT     5
2420 012002  000137  011656                    JMP     2$
2421
```

```
                                .SBTTL  DIGITAL MODULE TEST ,M6010,M6010YA


                                        ;THIS TEST CHECKS M6010 MODULE
                                        ;32 BIT NONISOLATED DC OUT


                                M6010:
                                ;;*************************************************************
012006  000004                  TST14:  SCOPE
012010  112737  000014  001206          MOVB    #14,$TESTN
012016  152777  000020  170236          BISB    #DBIT,@CSR      ;SET D BIT
012024  013700  002244                  MOV     TADDR,R0
012030  105020                          CLRB    (R0)+           ;CLEAR ALL FOUR BYTES OF I/O REGISTER
012032  105020                          CLRB    (R0)+
012034  105020                          CLRB    (R0)+
012036  105020                          CLRB    (R0)+
012040  005000                          CLR     R0
012042  005001                  4$:     CLR     R1
012044  005037  002226                  CLR     DBUFF           ;CLEAR IMIGE OF I/O REGISRERS
012050  005037  002230                  CLR     DBUFF+2
012054  116160  002232  002226  3$:     MOVB    PATT(R1),DBUFF(R0);SET DATA PATTERN IN IMIGE
012062  013737  002244  002304          MOV     TADDR,TEMP
012070  060037  002304                  ADD     R0,TEMP
012074  116177  002232  170202          MOVB    PATT(R1),@TEMP  ;SET DATA IN I/O REG
012102  005002                          CLR     R2
012104  013737  002244  002314  2$:     MOV     TADDR,TEMP1
012112  060237  002314                  ADD     R2,TEMP1
012116  117737  170172  001142          MOVB    @TEMP1,$BDDAT   ;READ I/O REGISTFR
012124  123762  001142  002226          CMPB    $BDDAT,DBUFF(R2);IS DATA OK?
012132  001404                          BEQ     1$
012134  116237  002226  001140          MOVB    DBUFF(R2),$GDDAT
012142  104017                          EMT     17
012144  005202                  1$:     INC     R2
012146  022702  000004                  CMP     #4,R2
012152  001354                          BNE     2$              ;TEST IF OTHER BYTES ARE OK
012154  005201                          INC     R1
012156  022701  000004                  CMP     #4,R1           ;LAST PATTERN?
012162  001334                          BNE     3$              ;TEST NEXT DATA PATTERN
012164  005200                          INC     R0
012166  022700  000004                  CMP     #4,R0           ;LAST BYTE?
012172  001323                          BNE     4$              ;NO,TEST NEXT BYTE
012174  000004                          SCOPE
012176  000207                          RTS     PC
```

```
                                .SBTTL  DIGITAL MODULE TEST , M6012,M6013


                                        ;THIS TEST CHECKS MODULES M6012 AND M6013
                                        ;8 BIT AC AND DC OUT


012200                          M6012:
012200                          M6013:
                                ;;*****************************************************************
012200  000004                  TST15:  SCOPE
012202  112737  000015  001206          MOVB    #15,$TESTN
012210  152777  000020  170044          BISB    #DBIT,@CSR
012216  012737  000125  001140          MOV     #125,$GDDAT     ;SET DATA PATTERN
012224  012737  000001  002242          MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
012232  004737  040332                  JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
012236  012737  000001  002242          MOV     #1,BYTNUM
012244  004737  040166                  JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY

012250  012737  000252  001140          MOV     #252,$GDDAT     ;SET DATA PATTERN
012256  012737  000001  002242          MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
012264  004737  040332                  JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
012270  012737  000001  002242          MOV     #1,BYTNUM
012276  004737  040166                  JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY

012302  012737  000377  001140          MOV     #377,$GDDAT     ;SET DATA PATTERN
012310  012737  000001  002242          MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
012316  004737  040332                  JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
012322  012737  000001  002242          MOV     #1,BYTNUM
012330  004737  040166                  JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY

012334  012737  000000  001140          MOV     #0,$GDDAT       ;SET DATA PATTERN
012342  012737  000001  002242          MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
012350  004737  040332                  JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
012354  012737  000001  002242          MOV     #1,BYTNUM
012362  004737  040166                  JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY
012366  000004                  SCOPE
012370  000207                  RTS     PC
```

```
                                        .SBTTL  DIGITAL MODULE TEST ,M6011

                                                ;THIS TEST CHECKS MODULE M6011
                                                ;ONE SHOT DC OUT


012372                                  M6011:
                                        ;;*****************************************************************
012372  000004                          TST16:  SCOPE
012374  112737  000016  001206                MOVB    #16,$TESTN
012402  013737  002244  002246                MOV     TADDR,TADDR1
012410  005237  002246                        INC     TADDR1
012414  152777  000020  167640                BISB    #DBIT,@CSR
012422  005037  002300                        CLR     CLK             ;CLEAR CLOCK COUNTER
012426  012777  040556  167666                MOV     #COUNT,@CLKVC   ;SET LOC 100- CLOCK VECTOR
012434  012777  000340  167662                MOV     #PR7,@CLKVCA    ;SET LOC 102
012442  012746  000000                        MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
012446  012746  012454                        MOV     #64$,-(SP)
012452  000002                              RTI
012454  000240                          64$:  NOP
012456  012737  000252  001140                MOV     #252,$GDDAT     ;SET DATA PATTERN
012464  012737  000002  002242                MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
012472  004737  040332                        JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
012476  012737  000002  002242                MOV     #2,BYTNUM
012504  004737  040166                        JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY
012510  005001                                CLR     R1
012512  005201                          11$:  INC     R1              ;CHECK IF LINE CLOCK IS INTERRUPTING
012514  001376                                BNE     11$
012516  005737  002300                        TST     CLK
012522  001010                                BNE     1$
012524  104025                                EMT     25
012526  000453                                BR      M6011B
012530  105777  167510                  2$:   TSTB    @TADDR
012534  001003                                BNE     1$
012536  105777  167504                        TSTB    @TADDR1
012542  001410                                BEQ     M6011A
012544  022737  001132  002300          1$:   CMP     #602.,CLK               ;WAIT 10SEC
012552  001366                                BNE     2$
012554  005037  001140                        CLR     $GDDAT
012560  004737  040250                        JSR     PC,TSTONE       ;PATTERN SHOULD BE CLEAR
012564                                  M6011A:
012564  005037  002300                        CLR     CLK             ;CLEAR CLOCK COUNTER
012570  012737  000125  001140                MOV     #125,$GDDAT     ;SET DATA PATTERN
012576  012737  000002  002242                MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
012604  004737  040332                        JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
012610  012737  000002  002242                MOV     #2,BYTNUM
012616  004737  040166                        JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY
012622  105777  167416                  2$:   TSTB    @TADDR
012626  001003                                BNE     1$
012630  105777  167412                        TSTB    @TADDR1
012634  001410                                BEQ     M6011B
012636  022737  001132  002300          1$:   CMP     #602.,CLK       ;WAIT 10 SEC
012644  001366                                BNE     2$
012646  005037  001140                        CLR     $GDDAT
012652  004737  040250                        JSR     PC,TSTONE       ;PATTERN SHOULD BE CLEAR
012656  000004                          M6011B: SCOPE
012660  012777  040554  167434                MOV     #NOCLK,@CLKVC   ;RESTORE CLOCK VECTOR
```

```
   2515 012666  000207                          RTS     PC
```

```
                                        .SBTTL  DIGITAL MODULE TEST ,M5011


                                                ;THIS TEST CHECKS M5011 MODULE
                                                ;16 BIT CHANGE OF STATE SENSE



                                        M5011:
                                        ;;**************************************************************
012670  000004                          TST17:  SCOPE
2526 012672  112737  000017  001206            MOVB    #17,$TESTN
2527 012700  013737  002244  002252            MOV     TADDR,COSADR
2528 012706  062737  000002  002252            ADD     #2,COSADR
2529 012714  152777  000030  167340            BISB    #DBIT!TBIT,@CSR;SET DISABLE BIT AND TBIT
2530 012722  112737  000377  001140            MOVB    #377,$GDDAT     ;SET WHAT DATA SHOULD BE
     012730  012737  000002  002242            MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
     012736  004737  040166                    JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
2531
2532 012742  142777  000010  167312            BICB    #TBIT,@CSR      ;CLEAR COMPLIM BIT
2533 012750  112737  000000  001140            MOVB    #0,$GDDAT       ;SET WHAT DATA SHOULD BE
     012756  012737  000002  002242            MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
     012764  004737  040166                    JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
2534 012770  004737  040564                    JSR     PC,CLRINT
2535 012774  012746  000000                    MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     013000  012746  013006                    MOV     #64$,-(SP)
     013004  000002                            RTI
     013006  000240                  64$:      NOP
2536 013010  012777  013114  167250            MOV     #1$,@VECTO      ;SET INTERRUPT VECTOR
2537 013016  012777  000340  167244            MOV     #PR7,@VECTOA
2538 013024  152777  000010  167230            BISB    #TBIT,@CSR      ;START INTERRUPT
2539 013032  152777  000100  167222            BISB    #EBIT,@CSR      ;ENABLE INTERRUPT
2540 013040  005037  002222                    CLR     YLOOP           ;WAIT
     013044  005237  002222            65$:    INC     YLOOP
     013050  023727  002222  177777            CMP     YLOOP,#-1
     013056  001372                            BNE     65$
2541 013060  104005                            EMT     5
2542                                        ;; /AN ERROR WILL OCCUR HERE IF INTERRUPTS
2543                                        ;; /ARE DISABLED BY A SWITCH SETTING ON THE MODULE
2544 013062  142777  000100  167172  2$:     BICB    #EBIT,@CSR      ;CLEAR CSR
2545 013070  142777  000010  167164          BICB    #TBIT,@CSR
2546 013076  012737  005404  000004          MOV     #TMOVEC,ERRVEC
2547 013104  004737  040564                  JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
2548 013110  000004                          SCOPE
2549 013112  000207                          RTS     PC              ;RETURN TO MONITOR
2550
2551 013114  022626                  1$:     CMP     (SP)+,(SP)+     ;ADJUST STOCK POINTER
2552 013116  005037  002314                  CLR     TEMP1
2553 013122  113737  002277  002305          MOVB    BASE+1,TEMP+1   ;INITIALIZE TEMP
2554 013130  012737  013510  000004          MOV     #10$,ERRVEC
2555 013136  117737  167122  002304  7$:     MOVB    @IAR,TEMP       ;FIND WHICH I/O INTERRUPTED
2556 013144  123737  002304  002244          CMPB    TEMP,TADDR      ;IS IT MUT
2557 013152  001416                          BEQ     8$              ;YES
2558 013154  152777  000001  167100          BISB    #RBIT,@CSR      ;CLEAR INTERRUPT
2559 013162  105777  167116                  TSTB    @TEMP
2560 013166  005237  002314                  INC     TEMP1
2561 013172  022737  000400  002314          CMP     #400,TEMP1
```

```
 2562 013200  001356                          BNE     7$
 2563 013202  104005                          EMT     5
 2564 013204  000137  013062                  JMP     2$
 2565 013210  142777  000100  167044  8$:     BICB    #EBIT,@CSR
 2566 013216  112737  000377  001140          MOVB    #377,$GDDAT
 2567 013224  117737  167022  001142          MOVB    @COSADR,$BDDAT  ;TEST COS REGISTER
 2568 013232  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;IS IT ALL ONES
 2569 013240  001401                          BEQ     3$
 2570 013242  104020                          EMT     20
 2571 013244  152777  000001  167010  3$:     BISB    #RBIT,@CSR
 2572 013252  105777  166774                  TSTB    @COSADR         ;CLEAR INTERRUPT
 2573 013256  005037  001140                  CLR     $GDDAT
 2574 013262  117737  166764  001142          MOVB    @COSADR,$BDDAT
 2575 013270  001401                          BEQ     4$
 2576 013272  104013                          EMT     13
 2577 013274  005237  002252          4$:     INC     COSADR
 2578 013300  005237  002244                  INC     TADDR           ;TEST NEXT BYTE
 2579 013304  117737  166754  002304          MOVB    @IAR,TEMP
 2580 013312  123737  002304  002244          CMPB    TEMP,TADDR      ;CHECK IF IT INTERRUPTED
 2581 013320  001403                          BEQ     11$
 2582 013322  104005                          EMT     5
 2583 013324  000137  013062                  JMP     2$
 2584 013330  112737  000377  001140  11$:    MOVB    #377,$GDDAT
 2585 013336  117737  166710  001142          MOVB    @COSADR,$BDDAT  ;CHECK IF COS REGISTER IS ALL ONES
 2586 013344  123737  001140  001142          CMPB    $GDDAT,$BDDAT
 2587 013352  001401                          BEQ     5$
 2588 013354  104020                          EMT     20
 2589 013356  152777  000001  166676  5$:     BISB    #RBIT,@CSR
 2590 013364  105777  166662                  TSTB    @COSADR         ;CLEAR INTERRUPT
 2591 013370  005037  001140                  CLR     $GDDAT
 2592 013374  117737  166652  001142          MOVB    @COSADR,$BDDAT
 2593 013402  001401                          BEQ     6$
 2594 013404  104013                          EMT     13
 2595 013406  005337  002252          6$:     DEC     COSADR
 2596 013412  005337  002244                  DEC     TADDR           ;RESET ADDRESS
 2597 013416  004737  040564                  JSR     PC,CLRINT       ;CLEAR REMAINING INTERRUPTS
 2598 013422  132777  000010  166632          BITB    #TBIT,@CSR      ;CHECK IF CLEARING T BIT CAUSE INTERRUPT
 2599 013430  001614                          BEQ     2$
 2600 013432  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
      013436  012746  013444                  MOV     #66$,-(SP)
      013442  000002                          RTI
      013444  000240          66$:    NOP
 2601 013446  142777  000010  166606          BICB    #TBIT,@CSR      ;CLEAR INPUTS
 2602 013454  152777  000100  166600          BISB    #EBIT,@CSR
 2603 013462  005037  002222                  CLR     YLOOP           ;WAIT
      013466  005237  002222          67$:    INC     YLOOP
      013472  023727  002222  177777          CMP     YLOOP,#-1
      013500  001372                          BNE     67$
 2604 013502  104005                          EMT     5
 2605 013504  000137  013062                  JMP     2$
 2606
 2607 013510  022626          10$:    CMP     (SP)+,(SP)+     ;RESET STACK POINTER
 2608 013512  104005                          EMT     5
 2609 013514  000137  013062                  JMP     2$
```

```
                                        .SBTTL  DIGITAL MODULE TEST M5012,M5012YA


                                                ;THIS TEST CHECKS M5012 MODULE
                                                ;ISOLATED 16 BIT DC INPUT


013520                                  M5012:
                                        ;;*****************************************************************
013520  000004                          TST20:  SCOPE
013522  012737  000020  001206                MOV     #20,$TESTN
013530  152777  000020  166524                BISB    #DBIT,@CSR
013536  112737  000000  001140                MOVB    #0,$GDDAT       ;SET WHAT DATA SHOULD BE
013544  012737  000002  002242                MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
013552  004737  040166                        JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT

013556  152777  000010  166476                BISB    #TBIT,@CSR
013564  112737  000377  001140                MOVB    #377,$GDDAT     ;SET WHAT DATA SHOULD BE
013572  012737  000002  002242                MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
013600  004737  040166                        JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
013604  142777  000010  166450                BICB    #TBIT,@CSR
013612  004737  040564                        JSR     PC,CLRINT
013616  012746  000000                        MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
013622  012746  013630                        MOV     #64$,-(SP)
013626  000002                                RTI
013630  000240                          64$:    NOP
013632  012777  013736  166426                MOV     #6$,@VECTO      ;SET INTERRUPT VECTOR
013640  012777  000340  166422                MOV     #PR7,@VECTOA
013646  152777  000100  166406                BISB    #EBIT,@CSR      ;ENABLE INTERRUPT
013654  152777  000010  166400                BISB    #TBIT,@CSR      ;START INTERRUPT
013662  005037  002222                        CLR     YLOOP           ;WAIT
013666  005237  002222                65$:    INC     YLOOP
013672  023727  002222  177777                CMP     YLOOP,#-1
013700  001372                                BNE     65$
013702  104005                                EMT     5
                                                ;; /AN ERROR WILL OCCUR HERE IF INTERRUPTS
                                                ;; /ARE DISABLED BY A SWITCH SETTING ON THE MODULE
013704  142777  000100  166350  7$:     BICB    #EBIT,@CSR      ;CLEAR CSR
013712  142777  000010  166342                BICB    #TBIT,@CSR
013720  012737  005404  000004                MOV     #TMOVEC,ERRVEC
013726  004737  040564                        JSR     PC,CLRINT       ;CLEAR ALL INTERRUPT
013732  000004                                SCOPE
013734  000207                                RTS     PC              ;RETURN TO MONITOR

013736  022626                          6$:     CMP     (SP)+,(SP)+     ;RESET STACK POINTER
013740  005037  002314                        CLR     TEMP1
013744  113737  002277  002305                MOVB    BASE+1,TEMP+1   ;INITIALIZE TEMP
013752  012737  014172  000004                MOV     #4$,ERRVEC
013760  117737  166300  002304  3$:     MOVB    @IAR,TEMP       ;FIND WHICH I/O INTERRUPTED
013766  152777  000001  166266                BISB    #RBIT,@CSR
013774  123737  002304  002244                CMPB    TEMP,TADDR      ;IS IT MUT
014002  001412                                BEQ     2$              ;YES
014004  105777  166274                        TSTB    @TEMP           ;CLEAR INTERRUPT
014010  005237  002314                10$:    INC     TEMP1
014014  022737  000400  002314                CMP     #400,TEMP1
014022  001356                                BNE     3$
014024  104005                                EMT     5
```

```
                                                BR      7$
                                2$:             TSTB    @TADDR          ;CLEAR INTERRUPT
                                                INC     TADDR
                                                BISB    #RBIT,@CSR
                                                TSTB    @TADDR          ;CLEAR INTERRUPT IN NEXT BYTE
                                                DEC     TADDR
                                                BITB    #IBIT,@CSR      ;CHECK IF INTERRUPT CLEAR
                                                BEQ     8$
                                                MOVB    @IAR,TEMP
                                                CMPB    TEMP,TADDR
                                                BNE     9$
                                                EMT     15
                                                BR      7$
                                9$:             JSR     PC,CLRINT       ;CLEAR ALL REMINDING INTERRUPTS
                                8$:             BITB    #TBIT,@CSR
                                                BEQ     7$
                                                MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
                                                MOV     #66$,-(SP)
                                                RTI
                                66$:            NOP
                                                BICB    #TBIT,@CSR      ;CHECK IF CLEARING T BIT CAUSE INTERRUPT
                                                CLR     YLOOP           ;WAIT
                                67$:            INC     YLOOP
                                                CMP     YLOOP,#-1
                                                BNE     67$
                                                EMT     5
                                                BR      7$

                                4$:             CMP     (SP)+,(SP)+     ;RESET STACK
                                                BR      10$
```

```
                                        .SBTTL  TESTING DIGITAL MODULE G670
                                                ;MOTHERBOARD FOR M5014 AND M6014
2686 014176                             G670:
                                        ;;*****************************************************************
     014176  000004                     1ST21:  SCOPE
2687 014200  012737  000021  001206             MOV     #21,$TESTN      ;LOAD THE TEST NUMBER
2688 014206  012746  000340                     MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
     014212  012746  014220                     MOV     #64$,-(SP)
     014216  000002                             RTI
     014220  000240                     64$:    NOP
2689                                            ;ADDRESS INITIALIZATION
2690 014222  013737  002244  003134             MOV     TADDR,ACNTL     ;INITIALIZE COUNTER A LOW-BYTE
2691 014230  013737  002244  003132             MOV     TADDR,ACNTH     ;INITIALIZE COUNTER A HIGH-BYTE
2692 014236  062737  000001  003132             ADD     #1,ACNTH        ;MAKE COUNTER A ADDRESS
2693 014244  013737  002244  003140             MOV     TADDR,BCNTL     ;INITIALIZE COUNTER B LOW-BYTE
2694 014252  062737  000002  003140             ADD     #2,BCNTL        ;MAKE COUNTER B ADDRESS
2695 014260  013737  002244  003136             MOV     TADDR,BCNTH     ;INITIALIZE COUNTER B HIGH-BYTE
2696 014266  062737  000003  003136             ADD     #3,BCNTH        ;MAKE COUNTER B ADDRESS
2697
2698                                            ;START WITH A KNOWN CONDITION -- CLEAR THE MODULE
2699
2700 014274  004737  032024                     JSR     PC,KLEER        ;CLEAR MODULE.
2701 014300  112777  000020  165754             MOVB    #DBIT,@CSR      ;SHUT OFF THE OUTSIDE WORLD
2702
2703                                            ;STATUS  OF  G670  AT  THIS  TIME  SHOULD  BE  --
2704
2705                                            ;       COUNTER A = 0           COUNTER B - 0
2706
2707                                            ;CHECK THAT COUNTERS ARE CLEAR
2708
2709 014306  005037  001140                     CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
2710 014312  013737  003134  003144             MOV     ACNTL,CTRSL     ;TEST COUNTER A
2711 014320  013737  003132  003142             MOV     ACNTH,CTRSH
2712
2713 014326  013737  003144  002244  1$:       MOV     CTRSL,TADDR     ;LOAD TEST ADDRESS
2714 014334  117737  166602  001143             MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE OF COUNTER.
2715 014342  117737  166576  001142             MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF COUNTER.
2716 014350  023737  001140  001142             CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
2717 014356  001401                             BEQ     2$              ;-THEN SKIP ERROR
2718
2719 014360  104075                             EMT     75
2720
2721 014362  023737  003140  002244  2$:       CMP     BCNTL,TADDR     ;IF DONE CHECKING CLEAR
2722 014370  001407                             BEQ     TSTPAT          ;-THEN DO PATTERN TEST
2723 014372  013737  003140  003144             MOV     BCNTL,CTRSL     ;-ELSE TEST COUNTER B
2724 014400  013737  003136  003142             MOV     BCNTH,CTRSH
2725 014406  000747                             BR      1$
2726
```

```
                                                ;TEST COUNTERS READ/WRITE CAPABILITY WITH PATTERNS
                                                ;125,252,377,AND 0 IN THAT ORDER.

014410  005000                          TSTPAT: CLR     R0              ;INITIALIZE PATTERN POINTER.

014412  013737  003134  002244          1$:     MOV     ACNTL,TADDR     ;TEST COUNTER A
014420  013737  003134  003144                  MOV     ACNTL,CTRSL
014426  013737  003132  003142                  MOV     ACNTH,CTRSH
014434  116037  002232  001140                  MOVB    PATT(R0),$GDDAT ;GET EXPECTED PATTERN
014442  116037  002232  001141                  MOVB    PATT(R0),$GDDAT+1

014450  113777  001141  166464  2$:     MOVB    $GDDAT+1,@CTRSH ;WRITE INTO HIGH BYTE OF COUNTER
014456  113777  001140  166460                  MOVB    $GDDAT,@CTRSL   ;WRITE PATTERN INTO LOW BYTE
014464  117737  166452  001143                  MOVB    @CTRSH,$BDDAT+1 ;READ COUNTERS HIGH BYTE
014472  117737  166446  001142                  MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF COUNTER
014500  023737  001140  001142                  CMP     $GDDAT,$BDDAT   ;IF PATTERN IS AS EXPECTED
014506  001401                                  BEQ     3$              ;-THEN SKIP ERROR

014510  104076                                  EMT     76

014512  023737  003140  002244  3$:     CMP     BCNTL,TADDR     ;IF COUNTER B JUST TESTED
014520  001414                                  BEQ     4$              ;-THEN DO NEXT PATTERN

014522  005137  001140                          COM     $GDDAT          ;-ELSE PRODUCE EXPECTED PATTERN
014526  013737  003140  002244                  MOV     BCNTL,TADDR     ; TEST COUNTER B
014534  013737  003140  003144                  MOV     BCNTL,CTRSL
014542  013737  003136  003142                  MOV     BCNTH,CTRSH
014550  000737                                  BR      2$

014552  005200                          4$:     INC     R0              ;MOVE PATTERN POINTER
014554  122760  000123  002232                  CMPB    #123,PATT(R0)   ;IF NOT AT END OF PATTERN TABLE
014562  001313                                  BNE     1$              ;-THEN DO NEW PATTERN
```

```
                                                ;CHECK THAT TBIT INITIALIZES COUNTERS

014564  005037  001140          TINIZ:  CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
014570  013737  003134  002244          MOV     ACNTL,TADDR     ;INITIALIZE ADDRESS TO TEST
014576  013737  003134  003144          MOV     ACNTL,CTRSL
014604  013737  003132  003142          MOV     ACNTH,CTRSH

014612  112777  177777  166322  1$:     MOVB    #-1,@CTRSH      ;WRITE INTO HIGH BYTE OF COUNTER
014620  112777  177777  166316          MOVB    #-1,@CTRSL      ;WRITE PATTERN INTO LOW BYTE
014626  152777  000010  165426          BISB    #TBIT,@CSR      ;INITIALIZE COUNTER
014634  142777  000010  165420          BICB    #TBIT,@CSR      ;DON'T LEAVE BITS HANGING AROUND
014642  117737  166274  001143          MOVB    @CTRSH,$BDDAT+1 ;READ COUNTERS HIGH BYTE
014650  117737  166270  001142          MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF COUNTER
014656  023737  001140  001 42          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
014664  001401                          BEQ     2$              ;-THEN SKIP ERROR

014666  104077                          EMT     77

014670  023737  002244  003140  2$:     CMP     TADDR,BCNTL     ;IF DONE TESTING TBIT INITIALIZE
014676  001412                          BEQ     INTHLT          ;-THEN TEST INTERRUPTS
014700  013737  003140  002244          MOV     BCNTL,TADDR     ;-ELSE TEST COUNTER B
014706  013737  003140  003144          MOV     BCNTL,CTRSL
014714  013737  003136  003142          MOV     BCNTH,CTRSH
014722  000733                          BR      1$

014724  152777  000010  165330  INTHLT: BISB    #TBIT,@CSR      ;START INTERNAL FREQ SOURCE
014732  005037  002222                  CLR     YLOOP           ;WAIT
014736  005237  002222          64$:    INC     YLOOP
014742  023727  002222  177777          CMP     YLOOP,#-1
014750  001372                          BNE     64$
014752  004737  040564                  JSR     PC,CLRINT       ;CLEAR ANY INTERRUPTS CAUSED BY TBIT.

014756  013737  003134  002244  1$:     MOV     ACNTL,TADDR     ;INITIALIZE FIRST COUNTER TO TEST
014764  013737  003134  003144          MOV     ACNTL,CTRSL
014772  013737  003132  003142          MOV     ACNTH,CTRSH
015000  012777  015124  165260          MOV     #2$,@VECTO      ;INITIALIZE INTERRUPT VECTOR
015006  012777  000340  165254          MOV     #PR7,@VECTOA    ;INITIALIZE INTERRUPT PRIORITY
015014  012737  000002  003150          MOV     #2,NUM          ;INITIALIZE A COUNT
```

```
                                                ;ALLOW AN INTERRUPT AND WAIT
                                        8$:
015022  012746  000000                          MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
015026  012746  015034                          MOV     #65$,-(SP)
015032  000002                                  RTI
015034  000240                        65$:      NOP
015036  152777  000100  165216                  BISB    #EBIT,@CSR      ;ALLOW THE INTERRUPT
015044  113777  003151  166070                  MOVB    NUM+1,@CTRSH    ;CLEAR HIGH BYTE BUFFER
015052  113777  003150  166064                  MOVB    NUM,@CTRSL      ;LOAD COUNTER
015060  005037  002222                          CLR     YLOOP           ;WAIT
015064  005237  002222                66$:      INC     YLOOP
015070  023727  002222  177777                  CMP     YLOOP,#-1
015076  001372                                  BNE     66$

015100  104103                                  EMT     103

015102  012746  000340                          MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
015106  012746  015114                          MOV     #67$,-(SP)
015112  000002                                  RTI
015114  000240                        67$:      NOP
015116  000523                                  BR      7$              ;SKIP INTERRUPT ROUTINE
```

```
                                        ;TIMEOUT HANDLER FOR INTERRUPT CHECK

015120  022626              33$:      CMP     (SP)+,(SP)+     ;RESET STACK POINTER
015122  000452                        BR      38$             ;CONTINUE INTERRUPT CHECKS

                                    ;ON INTERRUPT DO 2$

015124  022626              2$:       CMP     (SP)+,(SP)+     ;RESET STACK POINTER
015126  013746  000004              MOV     ERRVEC,-(SP)    ;SAVE TIMEOUT VECTOR
015132  013746  000006              MOV     ERRVEC+2,-(SP)  ;SAVE TIMEOUT PRIORITY
015136  013737  015120  000004      MOV     33$,ERRVEC      ;ON TIME OUT GO TO 33$
015144  012737  000340  000006      MOV     #PR7,ERRVEC+2   ;ASSURE PSW REMAINS PRIORITY 7
015152  005037  002314              CLR     TEMP1           ;PREPARE TO CONSTRUCT INTERRUPT ADDRESS
015156  013737  002276  002304      MOV     BASE,TEMP       ;INITIALIZE TEMP
015164  152777  000001  165070  3$: BISB    #RBIT,@CSR      ;PREPARE TO CLEAR INTERRUPT

                                        ;CHECK IF COUNTER INTERRUPTS

015172  117737  165066  002304      MOVB    @IAR,TEMP       ;GET LOW BYTE OF INTERRUPT ADDRESS

015200  023737  003134  002244      CMP     ACNTL,TADDR     ;IF COUNTER A IS BEING TESTED
015206  001406                      BEQ     35$             ;-THEN SEE IF COUNTER B INTERRUPTS

015210  023737  003134  002304      CMP     ACNTL,TEMP      ;-ELSE IF A DIDN'T INTERRUPT
015216  001006                      BNE     36$             ;--THEN SKIP ERROR

015220              34$:
015220  104110                      EMT     110
015222  000410                      BR      37$             ;---CLEAR UNWANTED INTERRUPT

015224  023737  003140  002304  35$:  CMP     BCNTL,TEMP      ;IF COUNTER B INTERRUPTED
015232  001772                      BEQ     34$             ;-THEN IT WAS UNEXPECTED

015234  023737  002304  002244  36$:  CMP     TEMP,TADDR      ;IF COUNTER INTERRUPTED
015242  001411                      BEQ     4$              ;-THEN CLEAN UP.

                                    ;CLEAR UNWANTED INTERRUPT

015244  105777  165034          37$:  TSTB    @TEMP           ;CLEAR UNWANTED INTERRUPT
015250  005237  002314          38$:  INC     TEMP1           ;INCREMENT INTERRUPT+TIMEOUT COUNT
015254  022737  000400  002314      CMP     #400,TEMP1      ;IF NOT EXCESSIVE COUNT NUMBER
015262  001340                      BNE     3$              ;-THEN CHECK AGAIN

015264  104103                      EMT     103

015266  012637  000004          4$:   MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
015272  012637  000006                MOV     (SP)+,ERRVEC+2  ;RESTORE TIMEOUT PRIORITY
015276  004737  040564                JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
015302  132777  000200  164752        BITB    #FBIT,@CSR      ;IF INTERRUPT CLEARS
015310  001401                      BEQ     5$              ;-THEN SKIP ERROR.

015312  104074                      EMT     74

015314  005037  001140          5$:   CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
015320  005037  002222                CLR     YLOOP           ;WAIT
015324  005237  002222          68$:  INC     YLOOP
015330  023727  002222  177777        CMP     YLOOP,#-1
```

```
        015336  001372                          BNE     68$
   2865 015340  117737  165576  001143          MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE
   2866 015346  117737  165572  001142          MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE BUFFER
   2867 015354  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF COUNTER HALTED AT 0
   2868 015362  001401                          BEQ     7$              ;-THEN SKIP ERROR
   2869
   2870 015364  104100                          EMT     100
   2871
   2872                                 ;-END 2$
   2873
   2874 015366  023737  003140  002244  7$:     CMP     BCNTL,TADDR     ;IF FINISHED CHECKING INTERRUPTS
   2875 015374  001413                          BEQ     TRANPT          ;-THEN TEST TRANSITION POINTS
   2876 015376  013737  003140  002244          MOV     BCNTL,TADDR     ;-ELSE CHECK COUNTER B
   2877 015404  013737  003140  003144          MOV     BCNTL,CTRSL
   2878 015412  013737  003136  003142          MOV     BCNTH,CTRSH
   2879 015420  000137  015022                  JMP     8$
```

```
2881 015424  112777  000030  164630  TRANPT: MOVB    #DBIT!TBIT,@CSR ;CLEAR EXTRANEOUS BITS AT CSR.
2882 015432  013737  003134  002244          MOV     ACNTL,TADDR     ;INITIALIZE COUNTER TO TEST
2883 015440  013737  003134  003144          MOV     ACNTL,CTRSL
2884 015446  013737  003132  003142          MOV     ACNTH,CTRSH
2885
2886 015454  012737    0000  003150  1$:     MOV     #BIT15,NUM      ;INITIALIZE TRANSITION NUMBER
2887
2888 015462  000257                  5$:     CCC                     ;CLEAR ALL CONDITION CODES
2889 015464  006037  003150                  ROR     NUM             ;DETERMINE NEXT TRANSITION VALUE
2890 015470  103016                          BCC     2$              ;IF NOT FINISHED THIS COUNTER DO 2$.
2891
2892 015472  023737  002244  003140  6$:     CMP     TADDR,BCNTL     ;IF COUNTER B WAS CHECKED
2893 015500  001454                          BEQ     7$              ;-THEN CHECK BOTH TOGETHER
2894 015502  013737  003140  002244          MOV     BCNTL,TADDR     ;-ELSE PREPARE TO TEST COUNTER B
2895 015510  013737  003140  003144          MOV     BCNTL,CTRSL
2896 015516  013737  003136  003142          MOV     BCNTH,CTRSH
2897 015524  000753                          BR      1$              ;
2898
2899 015526  005037  003154          2$:     CLR     XLOOP           ;INITIALIZE LOOP COUNT
2900 015532  013737  003150  001140          MOV     NUM,$GDDAT      ;COMPUTE EXPECTED DATA
2901 015540  005337  001140                  DEC     $GDDAT
2902
2903 015544  113777  003151  165370          MOVB    NUM+1,@CTRSH    ;LOAD  HIGH BYTE OF COUNTER
2904 015552  113777  003150  165364          MOVB    NUM,@CTRSL      ;LOAD COUNTER
2905
2906 015560  117737  165356  001143  3$:     MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE
2907 015566  117737  165352  001142          MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE
2908 015574  023737  003150  001142          CMP     NUM,$BDDAT      ;IF COUNT CHANGED
2909 015602  001005                          BNE     4$              ;-THEN CHECK DATA
2910 015604  005337  003154                  DEC     XLOOP           ;-ELSE IF DELAY NOT UP
2911 015610  001363                          BNE     3$              ;--THEN LOOK FOR A COUNT
2912
2913 015612  104101                          EMT     101
2914
2915 015614  000722                          BR      5$
2916
2917 015616  023737  001140  001142  4$:     CMP     $GDDAT,$BDDAT   ;IF COUNT WAS GOOD
2918 015624  001716                          BEQ     5$              ;-THEN SKIP ERROR
2919
2920 015626  104102                          EMT     102
2921
2922 015630  000714                          BR      5$              ;-THEN DO ANOTHER TRANSITION CHECK
2923
2924
```

```
                                                ;CHECK BOTH COUNTERS TOGETHER
                                                
015632  012737  100000  003150  7$:     MOV     #BIT15,NUM      ;INITIALI/      NSITION NUMBER
                                                
015640  000257                  12$:     CCC                     ;CLEAR ALL CONDITION CODES
015642  006037  003150                   ROR     NUM             ;DETERMINE NEXT TRANSITION
015646  103520                           BCS     16$             ;CHECK MSB WHEN DONE
                                                
015650  013737  003134  002244  8$:      MOV     ACNTL,TADDR     ;INITIALIZE TEST ADDRESS
015656  005037  003154                   CLR     XLOOP           ;INITIALIZE LOOP COUNT
015662  013737  003150  001140           MOV     NUM,$GDDAT      ;COMPUTE EXPECTED DATA
015670  005337  001140                   DEC     $GDDAT
                                                
015674  113777  003151  165230           MOVB    NUM+1,@ACNTH    ;LOAD HIGH BYTE BUFFER
015702  113777  003150  165224           MOVB    NUM,@ACNTL      ;LOAD COUNTER A
015710  113777  003150  165222           MOVB    NUM,@BCNTL      ;LOAD COUNTER B
                                                
015716  117737  165210  001143  9$:      MOVB    @ACNTH,$BDDAT+1 ;READ COUNTER A HIGH BYTE
015724  117737  165204  001142           MOVB    @ACNTL,$BDDAT   ;READ COUNTER A LOW BYTE
015732  023737  003150  001142           CMP     NUM,$BDDAT      ;IF COUNT HAS CHANGED
015740  001005                           BNE     10$             ;-THEN READ COUNTER B
015742  005337  003154                   DEC     XLOOP           ;-ELSE IF DELAY NOT UP
015746  001363                           BNE     9$              ;--THEN LOOK FOR A COUNT
                                                
015750  104101                           EMT     101
                                                
015752  000732                           BR      12$             ;DO NEXT TRANSITION POINT
                                                
015754  117737  165156  002305  10$:     MOVB    @BCNTH,TEMP+1   ;READ COUNTER B HIGH BYTE
015762  117737  165152  002304           MOVB    @BCNTL,TEMP     ;READ COUNTER B LOW BYTE
015770  023737  003150  002304           CMP     NUM,TEMP        ;IF COUNT IN B HAS CHANGED
015776  001023                           BNE     11$             ;-THEN CHECK IF COUNT IN A IS GOOD
                                                
016000  005337  003154                   DEC     XLOOP           ;-ELSE DECREMENT LOOP COUNT
016004  001363                           BNE     10$             ;--IF DELAY NOT UP CHECK COUNTER B
                                                
016006  013737  001142  002314           MOV     $BDDAT,TEMP1    ;---ELSE SAVE COUNT FROM COUNTER A
016014  013737  002304  001142           MOV     TEMP,$BDDAT     ;----PREPARE TO TYPE COUNTER B ERROR
016022  013737  003140  002244           MOV     BCNTL,TADDR
                                                
016030  104101                           EMT     101
                                                
016032  013737  002314  001142           MOV     TEMP1,$BDDAT    ;----RESTORE COUNTER A READING
016040  013737  003134  002244           MOV     ACNTL,TADDR     ;----RESTORE TEST ADDRESS COUNTER A
                                                
016046  023737  001140  001142  11$:     CMP     $GDDAT,$BDDAT   ;IF COUNTER A DATA GOOD
016054  001401                           BEQ     21$             ;-THEN SKIP ERROR
                                                
016056  104102                           EMT     102
                                                
016060  013737  002304  001142  21$:     MOV     TEMP,$BDDAT     ;PREPARE TO CHECK COUNTER B
016066  013737  003140  002244           MOV     BCNTL,TADDR
016074  023737  001140  001142           CMP     $GDDAT,$BDDAT   ;IF COUNT IN B IS GOOD
016102  001401                           BEQ     22$             ;-THEN SKIP ERROR
                                                
016104  104102                           EMT     102
                                                
```

```
 2983 016106  000654                  22$:    BR      12$             ;DO NEXT TRANSITION POINT
 2984
 2985
 2986
 2987                                         ;CHECK THAT COUNTERS DO NOT COUNT ON LOADING MSB ONLY
 2988
 2989 016110  012737  100000  001140  16$:    MOV     #BIT15,$GDDAT   ;DETERMINE EXPECTED DATA
 2990
 2991 016116  113777  001141  165006          MOVB    $GDDAT+1,@ACNTH ;LOAD HIGH BYTE BUFFER
 2992 016124  113777  001140  165002          MOVB    $GDDAT,@ACNTL   ;LOAD COUNTER A
 2993 016132  113777  001140  165000          MOVB    $GDDAT,@BCNTL   ;LOAD COUNTER B
 2994
 2995 016140  005037  002222                  CLR     YLOOP           ;WAIT
      016144  005237  002222          64$:    INC     YLOOP
      016150  023727  002222  000040          CMP     YLOOP,#40
      016156  001372                          BNE     64$
 2996
 2997 016160  117737  164746  001143          MOVB    @ACNTH,$BDDAT+1 ;READ COUNTER A HIGH BYTE
 2998 016166  117737  164742  001142          MOVB    @ACNTL,$BDDAT   ;READ COUNTER A LOW BYTE
 2999
 3000 016174  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF COUNTER A DATA GOOD
 3001 016202  001401                          BEQ     17$             ;-THEN SKIP ERROR
 3002
 3003 016204  104102                          EMT     102
 3004
 3005 016206  117737  164724  001143  17$:    MOVB    @BCNTH,$BDDAT+1 ;READ COUNTER B HIGH BYTE
 3006 016214  117737  164720  001142          MOVB    @BCNTL,$BDDAT   ;READ COUNTER B LOW BYTE
 3007
 3008 016222  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF COUNTER B DATA GOOD
 3009 016230  001401                          BEQ     13$             ;-THEN SKIP ERROR
 3010
 3011 016232  104102                          EMT     102
 3012
```

```
3014                                                    ;CHECK THAT CBIT INITIALIZES MODULE
3015
3016 016234  005037  001140          13$:    CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
3017 016240  112777  177777  164664          MOVB    #-1,@ACNTH      ;LOAD HIGH BYTE BUFFER
3018 016246  112777  000001  164660          MOVB    #1,@ACNTL       ;LOAD COUNTER A
3019 016254  112777  000001  164656          MOVB    #1,@BCNTL       ;LOAD COUNTER B
3020 016262  004737  032024                  JSR     PC,KLEER        ;CLEAR ALL MODULES
3021 016266  152777  000020  163766          BISB    #DBIT,@CSR      ;SHUT OFF WORLD
3022 016274  117737  164632  001143          MOVB    @ACNTH,$BDDAT+1 ;READ COUNTER A HIGH BYTE
3023 016302  117737  164626  001142          MOVB    @ACNTL,$BDDAT   ;READ LOW BYTE OF COUNTER A
3024 016310  117737  164622  002305          MOVB    @BCNTH,TEMP+1   ;READ COUNTER B HIGH BYTE
3025 016316  117737  164616  002304          MOVB    @BCNTL,TEMP     ;READ COUNTER B LOW BYTE
3026 016324  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF COUNTER A DATA IS GOOD
3027 016332  001404                          BEQ     14$             ;-THEN CHECK COUNTER B
3028 016334  013737  003134  002244          MOV     ACNTL,TADDR     ;PREPARE FOR ERROR
3029
3030 016342  104074                          EMT     74
3031
3032 016344  013737  002304  001142  14$:    MOV     TEMP,$BDDAT
3033 016352  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF COUNTER B DATA GOOD
3034 016360  001404                          BEQ     15$             ;-THEN SKIP ERROR
3035 016362  013737  003140  002244          MOV     BCNTL,TADDR     ;-ELSE PREPARE FOR ERROR
3036
3037 016370  104074                          EMT     74
3038
3039 016372  013737  003134  002244  15$:    MOV     ACNTL,TADDR     ;RESTORE TEST ADDRESS FOR LOOPING
3040 016400  000207                          RTS     PC
```

```
                                        .SBTTL  TESTING DIGITAL MODULE M5014
                                                ;M5014 DUAL 16-BIT INPUT COUNTER

                                        M5014:
                                        ;;*****************************************************************
        016402  000004                  TST22:  SCOPE
3046    016404  012737  000022  001206          MOV     #22,$TESTN      ;LOAD THE TEST NUMBER
3047
3048    016412  104411                          SAVREG
3049    016414  012746  000340                  MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
        016420  012746  016426                  MOV     #64$,-(SP)
        016424  000002                          RTI
        016426  000240                  64$:    NOP
3050                                            ;ADDRESS INITIALIZATION
3051    016430  013737  002244  003134          MOV     TADDR,ACNTL     ;INITIALIZE COUNTER A LOW-BYTE
3052    016436  013737  002244  003132          MOV     TADDR,ACNTH     ;INITIALIZE COUNTER A HIGH-BYTE
3053    016444  062737  000001  003132          ADD     #1,ACNTH        ;MAKE COUNTER A ADDRESS
3054    016452  013737  002244  003140          MOV     TADDR,BCNTL     ;INITIALIZE COUNTER B LOW-BYTE
3055    016460  062737  000002  003140          ADD     #2,BCNTL        ;MAKE COUNTER B ADDRESS
3056    016466  013737  002244  003136          MOV     TADDR,BCNTH     ;INITIALIZE COUNTER B HIGH-BYTE
3057    016474  062737  000003  003136          ADD     #3,BCNTH        ;MAKE COUNTER B ADDRESS
3058
3059                                            ;START WITH A KNOWN CONDITION -- CLEAR THE MODULE
3060
3061    016502  004737  032024                  JSR     PC,KLEER        ;CLEAR MODULE.
3062    016506  152777  000020  163546          BISB    #DBIT,@CSR      ;SHUT OFF WORLD
3063
3064                                            ;STATUS  OF  M5014  AT  THIS  TIME  SHOULD  BE  --
3065
3066                                            ;       COUNTER A = 0           COUNTER B = 0
3067
3068                                            ;CHECK THAT COUNTERS ARE CLEAR
3069
3070    016514  005037  001140                  CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
3071    016520  013737  003134  002244          MOV     ACNTL,TADDR     ;TEST COUNTER A
3072    016526  013737  003134  003144          MOV     ACNTL,CTRSL
3073    016534  013737  003132  003142          MOV     ACNTH,CTRSH
3074
3075    016542  117737  164374  001143  1$:     MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE OF COUNTER.
3076    016550  117737  164370  001142          MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF COUNTER.
3077    016556  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
3078    016564  001401                          BEQ     2$              ;-THEN SKIP ERROR
3079
3080    016566  104075                          EMT     75
3081
3082    016570  023737  003140  002244  2$:     CMP     BCNTL,TADDR     ;IF DONE CHECKING CLEAR
3083    016576  001412                          BEQ     TSTPT2          ;-THEN DO PATTERN TEST
3084    016600  013737  003140  002244          MOV     BCNTL,TADDR     ;-ELSE TEST COUNTER B
3085    0166    013737  003140  003144          MOV     BCNTL,CTRSL
3086    0166 .  013737  003136  003142          MOV     BCNTH,CTRSH
3087    016622  000747                          BR      1$
```

```
                                                ;CHECK THE READ/WRITE CAPABILITIES OF THE COUNTERS

016624  005000                        TSTPT2: CLR     R0              ;INITIALIZE PATTERN POINTER.
016626  142777  000010  163426                BICB    #TBIT,@CSR      ;ASSURE THAT TBIT IS OFF

016634  013737  003134  002244        1$:     MOV     ACNTL,TADDR     ;TEST COUNTER A
016642  013737  003134  003144                MOV     ACNTL,CTRSL
016650  013737  003132  003142                MOV     ACNTH,CTRSH
016656  116037  002232  001140                MOVB    PATT(R0),$GDDAT ;GET EXPECTED PATTERN
016664  116037  002232  001141                MOVB    PATT(R0),$GDDAT+1

016672  152777  000010  163362        2$:     BISB    #TBIT,@CSR      ;INITIALIZE BOARD
016700  113777  001141  164234                MOVB    $GDDAT+1,@CTRSH ;WRITE INTO HIGH BYTE OF COUNTER
016706  113777  001140  164230                MOVB    $GDDAT,@CTRSL   ;WRITE PATTERN INTO LOW BYTE
016714  117737  164222  001143                MOVB    @CTRSH,$BDDAT+1 ;READ COUNTERS HIGH BYTE
016722  117737  164216  001142                MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF COUNTER
016730  023737  001140  001142                CMP     $GDDAT,$BDDAT   ;IF PATTERN IS AS EXPECTED
016736  001401                                BEQ     3$              ;-THEN SKIP ERROR

016740  104076                                EMT     76

016742  142777  000010  163312        3$:     BICB    #TBIT,@CSR      ;STOP COUNTING
016750  023737  003140  002244                CMP     BCNTL,TADDR     ;IF COUNTER B JUST TESTED
016756  001414                                BEQ     4$              ;-THEN DO NEXT PATTERN

016760  005137  001140                        COM     $GDDAT          ;-ELSE PRODUCE EXPECTED PATTERN
016764  013737  003140  002244                MOV     BCNTL,TADDR     ; TEST COUNTER B
016772  013737  003140  003144                MOV     BCNTL,CTRSL
017000  013737  003136  003142                MOV     BCNTH,CTRSH
017006  000731                                BR      2$

017010  005200                        4$:     INC     R0              ;MOVE PATTERN POINTER
017012  122760  000123  002232                CMPB    #123,PATT(R0)   ;IF NOT AT END OF PATTERN TABLE
017020  001305                                BNE     1$              ;-THEN DO NEW PATTERN
```

```
3124                                            ;CHECK THAT TBIT INITIALIZES MODULE
3125
3126 017022  005037  001140          TINTMD: CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
3127 017026  112777  177777  164076          MOVB    #-1,@ACNTH      ;LOAD THE HIGH BYTE BUFFER.
3128 017034  112777  177777  164072          MOVB    #-1,@ACNTL      ;LOAD COUNTER A WITH ALL 1'S.
3129 017042  112777  177777  164070          MOVB    #-1,@BCNTL      ;LOAD COUNTER B WITH ALL 1'S.
3130 017050  112777  000020  163204          MOVB    #DBIT,@CSR      ;ASSURE CSR IS CLEAR.
3131
3132 017056  142777  000010  163176          BICB    #TBIT,@CSR      ;INITIALIZE BOARD.
3133 017064  152777  000010  163170          BISB    #TBIT,@CSR
3134 017072  013737  003134  002244          MOV     ACNTL,TADDR     ;INITIALIZE FIRST TEST ADDRESS
3135 017100  013737  003134  003144          MOV     ACNTL,CTRSL
3136 017106  013737  003132  003142          MOV     ACNTH,CTRSH
3137
3138 017114  117737  164022  001143  1$:     MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE OF TEST COUNTER
3139 017122  117737  164016  001142          MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF TEST COUNTER
3140 017130  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
3141 017136  001401                          BEQ     2$              ;-THEN SKIP ERROR
3142
3143 017140  104077                          EMT     77
3144
3145 017142  023737  003140  002244  2$:     CMP     BCNTL,TADDR     ;IF DONE CHECKING INIT
3146 017150  001412                          BEQ     INTHL2          ;-THEN CHECK INERRUPTS
3147 017152  013737  003140  002244          MOV     BCNTL,TADDR     ;-ELSE TEST COUNTER B.
3148 017160  013737  003140  003144          MOV     BCNTL,CTRSL
3149 017166  013737  003136  003142          MOV     BCNTH,CTRSH
3150 017174  000747                          BR      1$
3151
3152 017176  112777  000020  163056  INTHL2: MOVB    #DBIT,@CSR      ;START INTERNAL FREQ SOURCE
3153 017204  152777  000010  163050          BISB    #TBIT,@CSR
3154 017212  005037  002222                  CLR     YLOOP           ;WAIT
     017216  005237  002222          64$:    INC     YLOOP
     017222  023727  002222  000077          CMP     YLOOP,#77
     017230  001372                          BNE     64$
3155 017232  004737  040564                  JSR     PC,CLRINT       ;CLEAR ANY INTERRUPTS CAUSED BY TBIT.
3156
3157 017236  013737  003134  002244  1$:     MOV     ACNTL,TADDR     ;INITIALIZE FIRST COUNTER TO TEST
3158 017244  013737  003134  003144          MOV     ACNTL,CTRSL
3159 017252  013737  003132  003142          MOV     ACNTH,CTRSH
3160 017260  012777  017412  163000          MOV     #2$,@VECTO      ;INITIALIZE INTERRUPT VECTOR
3161 017266  012777  000340  162774          MOV     #PR7,@VECTOA    ;INITIALIZE INTERRUPT PRIORITY
3162 017274  012737  177776  003150          MOV     #-2,NUM         ;INITIALIZE A COUNT
3163
```

```
3165                                            ;ALLOW AN INTERRUPT AND WAIT
3166
3167 017302                                8$:
     017302  012746  000000                      MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     017306  012746  017314                      MOV     #65$,-(SP)
     017312  000002                              RTI
     017314  000240                     65$:     NOP
3168 017316  152777  000100  162736              BISB    #EBIT,@CSR      ;ALLOW THE INTERRUPT
3169 017324  113777  003151  163610              MOVB    NUM+1,@CTRSH    ;CLEAR HIGH BYTE BUFFER
3170 017332  113777  003150  163604              MOVB    NUM,@CTRSL      ;LOAD COUNTER
3171 017340  005037  002222                      CLR     YLOOP           ;WAIT
     017344  005237  002222                66$:  INC     YLOOP
     017350  023727  002222  177777              CMP     YLOOP,#-1
     017356  001372                              BNE     66$
3172
3173 017360  142777  000100  162674              BICB    #EBIT,@CSR      ;SHUT OFF INTERRUPTS
3174 017366  012746  000340                      MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
     017372  012746  017400                      MOV     #67$,-(SP)
     017376  000002                              RTI
     017400  000240                     67$:     NOP
3175
3176 017402  104103                              EMT     103
3177
3178 017404  000524                              BR      7$              ;SKIP INTERRUPT ROUTINE
```

```
                                        ;TIMEOUT HANDLER FOR INTERRUPT CHECK

017406  022626                  33$:   CMP     (SP)+,(SP)+     ;RESET STACK POINTER
017410  000452                         BR      38$             ;CONTINUE INTERRUPT CHECKS

                                        ;ON INTERRUPT DO 2$

017412  022626                  2$:    CMP     (SP)+,(SP)+     ;RESET STACK POINTER
017414  013746  000004                 MOV     ERRVEC,-(SP)    ;SAVE TIMEOUT VECTOR
017420  013746  000006                 MOV     ERRVEC+2,-(SP)  ;SAVE TIMEOUT PRIORITY
017424  013737  017406  000004         MOV     33$,ERRVEC      ;ON TIME OUT GO TO 33$
017432  012737  000340  000006         MOV     #PR7,ERRVEC+2   ;ASSURE PSW REMAINS PRIORITY 7
017440  005037  002314                 CLR     TEMP1           ;PREPARE TO CONSTRUCT INTERRUPT ADDRESS
017444  013737  002276  002304         MOV     BASE,TEMP       ;INITIALIZE TEMP
017452  152777  000001  162602  3$:    BISB    #RBIT,@CSR      ;PREPARE TO CLEAR INTERRUPT

                                        ;CHECK IF COUNTER INTERRUPTS

017460  117737  162600  002304         MOVB    @IAR,TEMP       ;GET LOW BYTE OF INTERRUPT ADDRESS

017466  023737  003134  002244         CMP     ACNTL,TADDR     ;IF COUNTER A IS BEING TESTED
017474  001406                         BEQ     35$             ;-THEN SEE IF COUNTER B INTERRUPTS

017476  023737  003134  002304         CMP     ACNTL,TEMP      ;-ELSE IF A DIDN'T INTERRUPT
017504  001006                         BNE     36$             ;--THEN SKIP ERROR

017506                          34$:
017506  104110                         EMT     110
017510  000410                         BR      37$             ;---CLEAR UNWANTED INTERRUPT

017512  023737  003140  002304  35$:   CMP     BCNTL,TEMP      ;IF COUNTER B INTERRUPTED
017520  001772                         BEQ     34$             ;-THEN IT WAS UNEXPECTED

017522  023737  002304  002244  36$:   CMP     TEMP,TADDR      ;IF COUNTER INTERRUPTED
017530  001411                         BEQ     4$              ;-THEN CLEAN UP.

                                        ;CLEAR UNWANTED INTERRUPT

017532  105777  162546          37$:   TSTB    @TEMP           ;CLEAR UNWANTED INTERRUPT
017536  005237  002314          38$:   INC     TEMP1           ;INCREMENT INTERRUPT+TIMEOUT COUNT
017542  022737  000400  002314         CMP     #400,TEMP1      ;IF NOT EXCESSIVE COUNT NUMBER
017550  001340                         BNE     3$              ;-THEN CHECK AGAIN

017552  104103                         EMT     103

017554  012637  000004          4$:    MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
017560  012637  000006                 MOV     (SP)+,ERRVEC+2  ;RESTORE TIMEOUT PRIORITY
017564  004737  040564                 JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
017570  132777  000200  162464         BITB    #FBIT,@CSR      ;IF INTERRUPT CLEARS
017576  001401                         BEQ     5$              ;THEN SKIP ERROR.

017600  104074                         EMT     74

017602  012737  177777  001140  5$:    MOV     #-1,$GDDAT      ;INITIALIZE EXPECTED DATA
017610  005037  002222                 CLR     YLOOP           ;WAIT
017614  005237  002222          68$:   INC     YLOOP
017620  023727  002222  000500         CMP     YLOOP,#500
```

```
        017626  001372                          BNE     68$
   3234 017630  117737  163306  001143          MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE
   3235 017636  117737  163302  001142          MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE BUFFER
   3236 017644  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF COUNTER HALTED AT -1
   3237 017652  001401                          BEQ     7$              ;-THEN SKIP ERROR
   3238
   3239 017654  104100                          EMT     100
   3240
   3241                                 ;-END 2$
   3242
   3243 017656  023737  003140  002244  7$:     CMP     BCNTL,TADDR     ;IF FINISHED CHECKING INTERRUPTS
   3244 017664  001413                          BEQ     TRANP2          ;-THEN TEST TRANSITION POINTS
   3245 017666  013737  003140  002244          MOV     BCNTL,TADDR     ;-ELSE CHECK COUNTER B
   3246 017674  013737  003140  003144          MOV     BCNTL,CTRSL
   3247 017702  013737  003136  003142          MOV     BCNTH,CTRSH
   3248 017710  000137  017302                  JMP     8$
```

```
                                        ;TEST THE TRANSITION POINTS
                                        ;
017714  112777  000020  162340  TRANP2: MOVB    #DBIT,@CSR      ;CLEAR EXTRANEOUS BITS AT CSR.
017722  152777  000010  162332          BISB    #TBIT,@CSR
017730  013737  003134  002244          MOV     ACNTL,TADDR     ;INITIALIZE COUNTER TEST ADDRESS
017736  013701  003132                  MOV     ACNTH,R1        ;INITIALIZE HIGH BYTE ADDRESS
017742  013700  003134                  MOV     ACNTL,R0        ;INITIALIZE LOW BYTE ADDRESS
                                        ;
017746  012737  100000  003150  1$:     MOV     #BIT15,NUM      ;INITIALIZE TRANSITION NUMBER
                                        ;
017754  000257                  5$:     CCC                     ;CLEAR ALL CONDITION CODES
017756  006037  003150                  ROR     NUM             ;DETERMINE NEXT TRANSITION VALUE
017762  103014                          BCC     2$              ;IF NOT FINISHED THIS COUNTER DO 2$.
                                        ;
017764  023737  002244  003140  6$:     CMP     TADDR,BCNTL     ;IF COUNTER B WAS CHECKED
017772  001447                          BEQ     STSTI2          ;-THEN CHECK FREQUENCY
017774  013737  003140  002244          MOV     BCNTL,TADDR     ;-ELSE PREPARE TO TEST COUNTER B
020002  013701  003136                  MOV     BCNTH,R1        ;--LOAD HIGH BYTE ADDRESS
020006  013700  003140                  MOV     BCNTL,R0        ;--LOAD LOW BYTE ADDRESS
020012  000755                          BR      1$              ;
                                        ;
020014  005005                  2$:     CLR     R5              ;INITIALIZE LOOP COUNT
020016  013737  003150  001140          MOV     NUM,$GDDAT      ;COMPUTE EXPECTED DATA
020024  013737  003150  002304          MOV     NUM,TEMP        ;COMPUTE DATA TO LOAD
020032  005337  002304                  DEC     TEMP
                                        ;
020036  113711  002305                  MOVB    TEMP+1,(R1)     ;LOAD  HIGH BYTE OF COUNTER
020042  113710  002304                  MOVB    TEMP,(R0)       ;LOAD COUNTER
                                        ;
020046  111137  001143          3$:     MOVB    (R1),$BDDAT+1   ;READ HIGH BYTE
020052  111037  001142                  MOVB    (R0),$BDDAT     ;READ LOW BYTE
020056  023737  002304  001142          CMP     TEMP,$BDDAT     ;IF COUNT CHANGED
020064  001004                          BNE     4$              ;-THEN CHECK DATA
020066  005305                          DEC     R5              ;-ELSE IF DELAY NOT UP
020070  001366                          BNE     3$              ;--THEN LOOK FOR A COUNT
                                        ;
020072  104101                          EMT     101
                                        ;
020074  000727                          BR      5$
                                        ;
020076  023737  001140  001142  4$:     CMP     $GDDAT,$BDDAT   ;IF COUNT WAS GOOD
020104  001723                          BEQ     5$              ;-THEN SKIP ERROR
                                        ;
020106  104102                          EMT     102
                                        ;
020110  000721                          BR      5$              ;-THEN DO ANOTHER TRANSITION CHECK
```

```
                                                ;TEST THAT COUNTERS COUNT TO WITHIN 20% OF 5KHZ

020112  004737  032624          STSTI2: JSR     PC,KLEER        ;CLEAR EVERYTHING
020116  112777  000020  162136          MOVB    #DBIT,@CSR      ;SHUT OFF WORLD
020124  152777  000010  162130          BISB    #IBIT,@CSR      ;ENABLE 5KHZ CLOCKING
020132  005037  002222                  CLR     YLOOP           ;WAIT
020136  005237  002222          64$:    INC     YLOOP
020142  023727  002222  000077          CMP     YLOOP,#77
020150  001372                          BNE     64$
020152  004737  040564                  JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
020156  012777  040556  162136          MOV     #COUNT,@CLKVC   ;INITIALIZE CLOCK VECTOR
020164  012777  000340  162132          MOV     #PR7,@CLKVCA    ;SET UP PRIORITY FOR CLK INT
020172  012777  020600  162066          MOV     #4$,@VECTO      ;INITIALIZE INTERRUPT VECTOR
020200  012777  000340  162062          MOV     #PR7,@VECTOA    ;SET UP PRIORITY FOR INT

020206  012737  001750  003152          MOV     #1000.,VARI     ;INITIALIZE ALLOWED ERROR
020214  012737  166170  001140          MOV     #-5000.,$GDDAT  ;INITIALIZE EXPECTED DATA
020222  163737  003152  001140          SUB     VARI,$GDDAT     ;SUBTRACT FUDGE FACTOR
020230  005037  003150                  CLR     NUM             ;INITIALIZE PROGRAM POSITION POINTER

                                                ;ALLOW THE INTERRUPTS

020234  152777  000100  162020          BISB    #EBIT,@CSR      ;ENABLE INTERRUPT BIT AT CSR
020242  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
020246  012746  020254                  MOV     #65$,-(SP)
020252  000002                          RTI
020254  000240          65$:    NOP

020256  113777  001141  162646          MOVB    $GDDAT+1,@ACNTH ;LOAD HIGH BYTE BUFFER
020264  113777  001140  162642          MOVB    $GDDAT,@ACNTL   ;LOAD COUNTER A
020272  113777  001140  162640          MOVB    $GDDAT,@BCNTL   ;LOAD COUNTER B

                                                ;TEST THAT THE CLOCK IS ON

020300  005037  002300                  CLR     CLK             ;INITIALIZE CLOCK
020304  005037  002222                  CLR     YLOOP           ;INITIALIZE LOOP COUNT
020310  005737  002300          11$:    TST     CLK             ;IF CLOCK IS COUNTING
020314  001005                          BNE     15$             ;-THEN CHECK THAT COUNTERS COUNT
020316  005237  002222                  INC     YLOOP           ;-ELSE IF CLOCK MAY COUNT
020322  001372                          BNE     11$             ;--THEN CHECK FOR COUNT

020324  104025                          EMT     25
020326  000511                          BR      14$             ;--SKIP CLOCK DEPENDENT TESTS

020330  012737  000001  003150  15$:    MOV     #1,NUM          ;SET A CLOCK FLAG

                                                ;WAIT UP TO ONE SECOND FOR A COUNT

020336  105777  162570          12$:    TSTB    @ACNTH          ;LOAD LOW BYTE BUFFER
020342  127737  162566  001140          CMPB    @ACNTL,$GDDAT   ;IF COUNTER A IS COUNTING
020350  001015                          BNE     13$             ;-THEN CHECK FREQUENCY
020352  105777  162560                  TSTB    @BCNTH          ;-ELSE CHECK COUNTER B
020356  127737  162556  001140          CMPB    @BCNTL,$GDDAT   ;--IF COUNTER B IS COUNTING
020364  001007                          BNE     13$             ;---THEN CHECK FREQUENCY
020366  023727  002300  000074          CMP     CLK,#60.        ;---ELSE IF 1 SECOND NOT UP
020374  002760                          BLT     12$             ;----THEN WAIT FOR COUNT
```

```
3348                                            ;WAIT ONE FULL SECOND
3349 020376  012737  000002  003150            MOV     #2,NUM          ;INDICATE PROGRAM MADE IT HERE
3350
3351 020404  005037  002300          13$:      CLR     CLK             ;INITIALIZE THE CLOCK
3352
3353 020410  023727  002300  000074  1$:       CMP     CLK,#60.        ;IF 1 SECOND NOT UP
3354 020416  002774                            BLT     1$              ;-THEN WAIT
3355
3356 020420  142777  000110  161634            BICB    #EBIT.IBIT,@CSR ;DON'T LEAVE THESE BITS HANGING
3357 020426  012746  000340                    MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
     020432  012746  020440                    MOV     #66$,-(SP)
     020436  000002                            RTI
     020440  000240                  66$:      NOP
3358
3359 020442  117737  162464  001143            MOVB    @ACNTH,$BDDAT+1 ;GET CONTENTS OF COUNTER A
3360 020450  117737  162460  001142            MOVB    @ACNTL,$BDDAT
3361 020456  117737  162454  003157            MOVB    @BCNTH,TEMPB+1  ;GET CONTENTS OF COUNTER B
3362 020464  117737  162450  003156            MOVB    @BCNTL,TEMPB
3363
3364 020472  013737  003152  001140            MOV     VARI,$GDDAT     ;GET EXPECTED COUNT
3365 020500  006337  003152                    ASL     VARI            ;DOUBLE THE VARIANCE
3366 020504  005437  003152                    NEG     VARI            ;GET TWO'S COMPLIMENT
3367
3368 020510  023737  003152  001142            CMP     VARI,$BDDAT     ;IF COUNT IS WITHIN 20% OF 5KHZ
3369 020516  100404                            BMI     2$              ;-THEN SKIP ERROR
3370
3371 020520  013737  003134  002244            MOV     ACNTL,TADDR     ;-ELSE PREPARE FOR ERROR
3372 020526  104106                            EMT     106
3373
3374 020530  023737  003152  003156  2$:       CMP     VARI,TEMPB      ;IF COUNT IS WITHIN 20% OF 5KHZ
3375 020536  100404                            BMI     3$              ;-THEN SKIP ERROR
3376
3377 020540  013737  003140  002244            MOV     BCNTL,TADDR     ;-ELSE PREPARE FOR ERROR
3378 020546  104106                            EMT     106
3379
3380 020550  000504                  3$:       BR      TSECTB          ;CHECK TIME BASE
3381
3382 020552  142777  000110  161502  14$:      BICB    #EBIT!TBIT,@CSR ;DON'T LEAVE BITS HANGING
3383 020560  012746  000340                    MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
     020564  012746  020572                    MOV     #67$,-(SP)
     020570  000002                            RTI
     020572  000240                  67$:      NOP
3384 020574  000137  021370                    JMP     CBITCM          ;SKIP CLOCK DEPENDENT TESTS
```

```
3386                                                    ;ON INTERRUPT DO 4$
3387
3388 020600  022626                          4$:       CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
3389 020602  013737  002300  002314                     MOV     CLK,TEMP1       ;GET TIME AT INTERRUPT
3390 020610  013737  002276  002304                     MOV     BASE,TEMP       ;INITIALIZE TEMP
3391 020616  117737  161442  002304                     MOVB    @IAR,TEMP       ;GET INTERRUPTING ADDRESS
3392
3393 020624  023737  003134  002304                     CMP     ACNTL,TEMP      ;IF NOT COUNTER A INTERRUPTING
3394 020632  001005                                     BNE     5$              ;-THEN SKIP ERROR
3395
3396 020634  013737  003134  002244                     MOV     ACNTL,TADDR     ;-ELSE PREPARE FOR ERROR TYPEOUT
3397 020642  104106                                     EMT     106
3398 020644  000437                                     BR      7$              ;CLEAN UP
3399
3400 020646  023737  003140  002304  5$:       CMP     BCNTL,TEMP      ;IF NOT COUNTER B INTERRUPTING
3401 020654  001005                                     BNE     6$              ;-THEN SKIP ERROR
3402
3403 020656  013737  003140  002244                     MOV     BCNTL,TADDR     ;-ELSE PREPARE FOR ERROR TYPEOUT
3404 020664  104106                                     EMT     106
3405
3406 020666  000426                                     BR      7$              ;CLEAN UP
3407
3408 020670  152777  000001  161364  6$:       BISB    #RBIT,@CSR      ;CLEAR UNWANTED INTERRUPT
3409 020676  105777  161402                           TSTB    @TEMP
3410 020702  012746  000000                           MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     020706  012746  020714                           MOV     #68$,-(SP)
     020712  000002                                   RTI
     020714  000240                          68$:     NOP
3411 020716  005737  003150                           TST     NUM             ;IF NOT AT 11$ MARK
3412 020722  001002                                   BNE     16$             ;-THEN GO TO CORRECT MARK
3413 020724  000137  020310                           JMP     11$             ;-ELSE CONTINUE AT 11$ MARK
3414 020730  022737  000001  003150  16$:       CMP     #1,NJM          ;IF AT THE  1$ MARK
3415 020736  001224                                   BNE     1$              ;-THEN GO TO IT
3416 020740  000137  020336                           JMP     12$             ;-ELSE GO TO THE 12$ MARK
3417
3418 020744  004737  040564                  7$:      JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
3419 020750  132777  000200  161304                   BITB    #FBIT,@CSR      ;IF INTERRUPT CLEARS
3420 020756  001401                                   BEQ     1SECTB          ;-THEN SKIP ERROR
3421 020760  104104                                   EMT     104
```

```
                                                ;CHECK THAT A 10 SECOND TIME BASE IS CHOSEN
                                                
020762                                  TSECTB:
020762  012746  000340                          MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
020766  012746  020774                          MOV     #64$,-(SP)
020772  000002                                  RTI
020774  000240                          64$:    NOP
020776  012737  141520  001140                  MOV     #50000.,$GDDAT  ;INITIALIZE EXPECTED DATA
021004  142777  000110  161250                  BICB    #TBIT!EBIT,@CSR ;ASSURE TBIT AND EBIT ARE CLEAR
021012  012777  021312  161246                  MOV     #4$,@VECTO      ;INITIALIZE INTERRUPT VECTOR
021020  012777  000340  161242                  MOV     #PR7,@VECTOA    ;INITIALIZE INTERRUPT PSW STATUS
021026  152777  000010  161226                  BISB    #TBIT,@CSR      ;INITIALIZE MODULE
021034  004737  040564                          JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
021040  105077  162066                          CLRB    @ACNTH          ;CLEAR HIGH BYTE BUFFER
021044  105077  162064                          CLRB    @ACNTL          ;CLEAR COUNTER A
021050  105077  162064                          CLRB    @BCNTL          ;CLEAR COUNTER B
021054  005037  002300                          CLR     CLK             ;INITIALIZE CLOCK COUNT
                                                
021060  152777  000100  161174                  BISB    #EBIT,@CSR      ;ALLOW INTERRUPTS
021066  012746  000000                          MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
021072  012746  021100                          MOV     #65$,-(SP)
021076  000002                                  RTI
021100  000240                          65$:    NOP
021102  005037  002222                          CLR     YLOOP           ;WAIT
021106  005237  002222                  6$:     INC     YLOOP
021112  023727  002222  177777                  CMP     YLOOP,#-1
021120  001372                                  BNE     66$
021122  005737  002300                          TST     CLK             ;IF CLOCK IS COUNTING
021126  001014                                  BNE     1$              ;-THEN WAIT 11 SECS
021130  112777  000020  161124                  MOVB    #DBIT,@CSR      ;-ELSE CLEAN UP THE CSR
021136  012746  000340                          MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
021142  012746  021150                          MOV     #67$,-(SP)
021146  000002                                  RTI
021150  000240                          67$:    NOP
                                                
021152  104025                                  EMT     25
021154  000137  021370                          JMP     CBITCM          ;  AND CHECK THAT CBIT CLEARS MUT
                                                
021160  023727  002300  001332          1$:     CMP     CLK,#730.       ;IF A LITTLE MORE THAN 12 SECONDS NOT UP
021166  002774                                  BLT     1$              ;-THEN WAIT
                                                
021170  012746  000340                          MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
021174  012746  021202                          MOV     #68$,-(SP)
021200  000002                                  RTI
021202  000240                          68$:    NOP
021204  012777  000020  161050                  MOV     #DBIT,@CSP      ;CLEAR EXTRANEOUS BITS
021212  104103                                  EMT     103
                                                
021214  013737  003132  003142          5$:     MOV     ACNTH,CTRSH     ;INITIALIZE FIRST TEST ADDRESS
021222  013737  003134  003144                  MOV     ACNTL,CTRSL
                                                
021230  013737  003144  002244          2$:     MOV     CTRSL,TADDR     ;LOAD TEST ADDRESS
021236  117737  161700  001143                  MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE OF TEST COUNTER
021244  117737  161674  001142                  MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF TEST COUNTER
021252  023737  001140  001142                  CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
021260  001401                                  BEQ     3$              ;-THEN SKIP ERROR
```

```
3464 021262  104107                          EMT     107
3465
3466 021264  023737  003140  002244  3$:     CMP     BCNTL,TADDR     ;IF DONE CHECKING TIME BASE
3467 021272  001436                          BEQ     CBITCM          ;-THEN CHECK CBIT INIT
3468 021274  013737  003140  003144          MOV     BCNTL,CTRSL     ;-ELSE TEST COUNTER B.
3469 021302  013737  003136  003142          MOV     BCNTH,CTRSH
3470 021310  000747                          BR      2$
3471
3472 021312  022626                  4$:     CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
3473 021314  013737  002276  002304          MOV     BASE,TEMP       ;INITIALIZE TEMP
3474 021322  117737  160736  002304          MOVB    @IAR,TEMP       ;GET LOW BYTE INTERRUPTING ADDRESS
3475
3476 021330  023737  003140  002304          CMP     BCNTL,TEMP      ;IF B INTERRUPTED
3477 021336  001726                          BEQ     5$              ;-THEN SEE IF TIME BASE IS CORRECT
3478
3479 021340  152777  000001  160714          BISB    #RBIT,@CSR      ;-ELSE CLEAR UNWANTED INTERRUPT
3480 021346  105777  160732                  TSTB    @TEMP
3481 021352  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     021356  012746  021364                  MOV     #69$,-(SP)
     021362  000002                          RTI
     021364  000240                  69$:    NOP
3482 021366  000674                          BR      1$              ;-CONTINUE 12 SECOND WAIT
```

```
                                                ;CHECK THAT CBIT INITIALIZES MODULE

021370  005037  001140          CBITCM: CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
021374  112777  177777  161530          MOVB    #-1,@ACNTH      ;LOAD THE HIGH BYTE BUFFER.
021402  112777  177777  161524          MOVB    #-1,@ACNTL      ;LOAD COUNTER A WITH ALL 1'S.
021410  112777  177777  161522          MOVB    #-1,@BCNTL      ;LOAD COUNTER B WITH ALL 1'S.
021416  112777  000020  160636          MOVB    #DBIT,@CSR      ;ASSURE CSR IS CLEAR.

021424  004737  032024                  JSR     PC,KLEER        ;INITIALIZE SYSTEM
021430  152777  000020  160624          BISB    #DBIT,@CSR      ;SHUT OFF WORLD

021436  013737  003134  003144          MOV     ACNTL,CTRSL     ;INITIALIZE FIRST TEST ADDRESS
021444  013737  003132  003142          MOV     ACNTH,CTRSH

021452  013737  003144  002244  1$:     MOV     CTRSL,TADDR     ;INITIALIZE ADDRESS UNDER TEST
021460  117737  161456  001143          MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE OF TEST COUNTER
021466  117737  161452  001142          MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF TEST COUNTER
021474  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
021502  001401                          BEQ     2$              ;-THEN SKIP ERROR

021504  104075                          EMT     75

021506  023737  003140  002244  2$:     CMP     BCNTL,TADDR     ;IF DONE CHECKING INIT
021514  001407                          BEQ     M5014R          ;-THEN RETURN FROM SUBROUTINE
021516  013737  003140  003144          MOV     BCNTL,CTRSL     ;-ELSE TEST COUNTER B.
021524  013737  003136  003142          MOV     BCNTH,CTRSH
021532  000747                          BR      1$
021534  104412                  M5014R: RESREG                  ;RESTORE REGISTERS
021536  013737  003134  002244          MOV     ACNTL,TADDR     ;RESTORE TEST ADDRESS FOR LOOPING
021544  000207                          RTS     PC              ;RETURN TO MAIN PROGRAM.
```

```
                                        .SBTTL  TESTING DIGITAL MODULE M6014
                                                ;M6014 DUAL 16-BIT OUTPUT COUNTER
                                        M6014:
                                        ;;*****************************************************************
021546  000004                          TST23:  SCOPE
3519 021550  012737  000023  001206            MOV     #23,$TESTN      ;LOAD THE TEST NUMBER
3520
3521 021556  012746  000340                    MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
     021562  012746  021570                    MOV     #64$,-(SP)
     021566  000002                            RTI
     021570  000240                    64$:    NOP
3522                                            ;ADDRESS INITIALIZATION
3523 021572  013737  002244  003134            MOV     TADDR,ACNTL     ;INITIALIZE COUNTER A LOW-BYTE
3524 021600  013737  002244  003132            MOV     TADDR,ACNTH     ;INITIALIZE COUNTER A HIGH-BYTE
3525 021606  062737  000001  003132            ADD     #1,ACNTH        ;MAKE COUNTER A ADDRESS
3526 021614  013737  002244  003140            MOV     TADDR,BCNTL     ;INITIALIZE COUNTER B LOW-BYTE
3527 021622  062737  000002  003140            ADD     #2,BCNTL        ;MAKE COUNTER B ADDRESS
3528 021630  013737  002244  003136            MOV     TADDR,BCNTH     ;INITIALIZE COUNTER B HIGH-BYTE
3529 021636  062737  000003  003136            ADD     #3,BCNTH        ;MAKE COUNTER B ADDRESS
3530
3531                                            ;START WITH A KNOWN CONDITION -- CLEAR THE MODULE
3532
3533 021644  004737  032024                    JSR     PC,KLEER        ;CLEAR MODULE.
3534 021650  152777  000020  160404            BISB    #DBIT,@CSR      ;SHUT OFF WORLD
3535
3536                                            ;STATUS  OF  M6014  AT  THIS  TIME  SHOULD  BE  --
3537
3538                                            ;       COUNTER A = 0           COUNTER B = 0
3539
3540                                            ;CHECK THAT COUNTERS ARE CLEAR
3541
3542 021656  005037  001140                    CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
3543 021662  013737  003134  003144            MOV     ACNTL,CTRSL     ;TEST COUNTER A
3544 021670  013737  003132  003142            MOV     ACNTH,CTRSH
3545
3546 021676  013737  003144  002244  1$:       MOV     CTRSL,TADDR     ;INITIALIZE TEST ADDRESS
3547 021704  117737  161232  001143            MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE OF COUNTER.
3548 021712  117737  161226  001142            MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF COUNTER.
3549 021720  023737  001140  001142            CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
3550 021726  001401                            BEQ     2$              ;-THEN SKIP ERROR
3551
3552 021730  104075                            EMT     75
3553
3554 021732  023737  003140  002244  2$:       CMP     BCNTL,TADDR     ;IF DONE CHECKING CLEAR
3555 021740  001407                            BEQ     TSTPT1          ;-THEN DO PATTERN TEST
3556 021742  013737  003140  003144            MOV     BCNTL,CTRSL     ;-ELSE TEST COUNTER B
3557 021750  013737  003136  003142            MOV     BCNTH,CTRSH
3558 021756  000747                            BR      1$
3559
```

```
                                                ;CHECK THE READ/WRITE CAPABILITIES OF THE COUNTERS

021760  005000                        TSTPT1: CLR     R0              ;INITIALIZE PATTERN POINTER.
021762  142777  000010  160272                BICB    #TBIT,@CSR      ;ASSURE THAT TBIT IS OFF

021770  013737  003134  003144        1$:     MOV     ACNTL,CTRSL     ;TEST COUNTER A
021776  013737  003132  003142                MOV     ACNTH,CTRSH
022004  116037  002232  001140                MOVB    PATT(R0),$GDDAT ;GET EXPECTED PATTERN
022012  116037  002232  001141                MOVB    PATT(R0),$GDDAT+1

022020  152777  000010  160234  2$:     BISB    #TBIT,@CSR      ;INITIALIZE BOARD
022026  013737  003144  002244                MOV     CTRSL,TADDR     ;INITIALIZE TEST ADDRESS
022034  113777  001141  161100                MOVB    $GDDAT+1,@CTRSH ;WRITE INTO HIGH BYTE OF COUNTER
022042  113777  001140  161074                MOVB    $GDDAT,@CTRSL   ;WRITE PATTERN INTO LOW BYTE
022050  117737  161066  001143                MOVB    @CTRSH,$BDDAT+1 ;READ COUNTERS HIGH BYTE
022056  117737  161062  001142                MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF COUNTER
022064  023737  001140  001142                CMP     $GDDAT,$BDDAT   ;IF PATTERN IS AS EXPECTED
022072  001401                                BEQ     3$              ;-THEN SKIP ERROR

022074  104076                                EMT     76

022076  142777  000010  160156  3$:     BICB    #TBIT,@CSR      ;STOP COUNTING
022104  023737  003140  002244                CMP     BCNTL,TADDR     ;IF COUNTER B JUST TESTED
022112  001411                                BEQ     4$              ;-THEN DO NEXT PATTERN

022114  005137  001140                        COM     $GDDAT          ;-ELSE PRODUCE EXPECTED PATTERN
022120  013737  003140  003144                MOV     BCNTL,CTRSL     ; TEST COUNTER B
022126  013737  003136  003142                MOV     BCNTH,CTRSH
022134  000731                                BR      2$

022136  005200                        4$:     INC     R0              ;MOVE PATTERN POINTER
022140  122760  000123  002232                CMPB    #123,PATT(R0)   ;IF NOT AT END OF PATTERN TABLE
022146  001310                                BNE     1$              ;-THEN DO NEW PATTERN
```

```
3595
3596                                            ;CHECK THAT TBIT INITIALIZES MODULE
3597
3598 022150  005037  001140          TINIMO: CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
3599 022154  112777  177777  160750          MOVB    #-1,@ACNTH      ;LOAD THE HIGH BYTE BUFFER.
3600 022162  112777  177777  160744          MOVB    #-1,@ACNTL      ;LOAD COUNTER A WITH ALL 1'S.
3601 022170  112777  177777  160742          MOVB    #-1,@BCNTL      ;LOAD COUNTER B WITH ALL 1'S.
3602 022176  112777  000020  160056          MOVB    #DBIT,@CSR      ;ASSURE CSR IS CLEAR.
3603
3604 022204  152777  000010  160050          BISB    #TBIT,@CSR      ;INITIALIZE BOARD.
3605 022212  013737  003134  003144          MOV     ACNTL,CTRSL     ;INITIALIZE FIRST TEST ADDRESS
3606 022220  013737  003132  003142          MOV     ACNTH,CTRSH
3607
3608 022226  013737  003144  002244  1$:     MOV     CTRSL,TADDR     ;INITIALIZE TEST ADDRESS
3609 022234  117737  160702  001143          MOVB    @CTRSH,$BDDAT+1 ;READ HIGH BYTE OF TEST COUNTER
3610 022242  117737  160676  001142          MOVB    @CTRSL,$BDDAT   ;READ LOW BYTE OF TEST COUNTER
3611 022250  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
3612 022256  001401                          BEQ     2$              ;-THEN SKIP ERROR
3613
3614 022260  104077                          EMT     77
3615
3616 022262  023737  003140  002244  2$:     CMP     BCNTL,TADDR     ;IF DONE CHECKING INIT
3617 022270  001407                          BEQ     INTHL1          ;-THEN CHECK MODULE INTERRUPTS
3618 022272  013737  003140  003144          MOV     BCNTL,CTRSL     ;-ELSE TEST COUNTER B.
3619 022300  013737  003136  003142          MOV     BCNTH,CTRSH
3620 022306  000747                          BR      1$
3621
3622                                            ;CHECK THAT MODULE INTERRUPTS
3623
3624 022310  112777  000030  157744  INTHL1: MOVB    #DBIT!TBIT,@CSR ;START INTERNAL FREQ SOURCE
3625 022316  005037  002222                  CLR     YLOOP           ;WAIT
     022322  005237  002222          64$:    INC     YLOOP
     022326  023727  002222  177777          CMP     YLOOP,#-1
     022334  001372                          BNE     64$
3626 022336  004737  040564                  JSR     PC,CLRINT       ;CLEAR ANY INTERRUPTS CAUSED BY TBIT.
3627
3628 022342  013737  003132  002244  1$:     MOV     ACNTH,TADDR     ;INITIALIZE FIRST COUNTER TO TEST
3629 022350  012777  022502  157710          MOV     #2$,@VECTO      ;INITIALIZE INTERRUPT VECTOR
3630 022356  012777  000340  157704          MOV     #PR7,@VECTOA    ;INITIALIZE INTERRUPT PRIORITY
3631 022364  012737  000002  003150          MOV     #2,NUM          ;INITIALIZE A COUNT
3632
```

```
                                                ;ALLOW AN INTERRUPT AND WAIT
                                        8$:
022372                                          MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
022372  012746  000000                          MOV     #65$,-(SP)
022376  012746  022404                          RTI
022402  000002                                  NOP
022404  000240                          65$:    BISB    #EBIT,@CSR      ;ALLOW THE INTERRUPT
022406  152777  000100  157646                  MOVB    NUM+1,@TADDR    ;CLEAR HIGH BYTE BUFFER
022414  113777  003151  157622                  BICB    #BIT0,TADDR      PREPARE TO LOAD COUNTER
022422  142737  000001  002244                  MOVB    NUM,@TADDR      ;LOAD COUNTER
022430  113777  003150  157606                  CLR     YLOOP           ;WAIT
022436  005037  002222                  66$:    INC     YLOOP
022442  005237  002222                          CMP     YLOOP,#-1
022446  023727  002222  177777                  BNE     66$
022454  001372
                                                EMT     103
022456  104103
                                                MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
022460  012746  000340                          MOV     #67$,-(SP)
022464  012746  022472                          RTI
022470  000002                          67$:    NOP
022472  000240                                  BR      7$              ;SKIP INTERRUPT ROUTINE
022474  000531
```

```
                                        ;TIMEOUT HANDLER FOR INTERRUPT CHECK

022476  022626                33$:     CMP     (SP)+,(SP)+     ;RESET STACK POINTER
022500  000452                         BR      38$             ;CONTINUE INTERRUPT CHECK.

                                        ;ON INTERRUPT DO 2$

022502  022626                2$:      CMP     (SP)+,(SP)+     ;RESET STACK POINTER
022504  013746  000004                MOV     ERRVEC,-(SP)    ;SAVE TIMEOUT VECTOR
022510  013746  000006                MOV     ERRVEC+2,-(SP)  ;SAVE TIMEOUT PRIORITY
022514  013737  022476  000004       MOV     33$,ERRVEC      ;ON TIME OUT GO TO 33$
022522  012737  000340  000006       MOV     #PR7,ERRVEC+2   ;ASSURE PSW REMAINS PRIORITY 7
022530  005037  002314               CLR     TEMP1           ;PREPARE TO CONSTRUCT INTERRUPT ADDRESS
022534  013737  002276  002304       MOV     BASE,TEMP       ;INITIALIZE TEMP
022542  152777  000001  157512  3$:  BISB    #RBIT,@CSR      ;PREPARE TO CLEAR INTERRUPT

                                        ;CHECK IF COUNTER INTERRUPTS

022550  117737  157510  002304       MOVB    @IAR,TEMP       ;GET LOW BYTE OF INTERRUPT ADDRESS

022556  023737  003134  002244       CMP     ACNTL,TADDR     ;IF COUNTER A IS BEING TESTED
022564  001406                       BEQ     35$             ;-THEN SEE IF COUNTER B INTERRUPTS

022566  023737  003134  002304       CMP     ACNTL,TEMP      ;-ELSE IF A DIDN'T INTERRUPT
022574  001006                       BNE     36$             ;--THEN SKIP ERROR

022576                          34$:
022576  104110                       EMT     110
022600  000410                       BR      37$             ;---CLEAR UNWANTED INTERRUPT

022602  023737  003140  002304  35$: CMP     BCNTL,TEMP      ;IF COUNTER B INTERRUPTED
022610  001772                       BEQ     34$             ;-THEN IT WAS UNEXPECTED

022612  023737  002304  002244  36$: CMP     TEMP,TADDR      ;IF COUNTER INTERRUPTED
022620  001411                       BEQ     4$              ;-THEN CLEAN UP.

                                        ;CLEAR UNWANTED INTERRUPT

022622  105777  157456          37$: TSTB    @TEMP           ;CLEAR UNWANTED INTERRUPT
022626  005237  002314          38$: INC     TEMP1           ;INCREMENT INTERRUPT+TIMEOUT COUNT
022632  022737  000400  002314       CMP     #400,TEMP1      ;IF NOT EXCESSIVE COUNT NUMBER
022640  001340                       BNE     3$              ;-THEN CHECK AGAIN

022642  104103                       EMT     103

022644  012637  000004          4$:  MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
022650  012637  000006               MOV     (SP)+,ERRVEC+2  ;RESTORE TIMEOUT PRIORITY
022654  004737  040564               JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
022660  132777  000200  157374       BITB    #FBIT,@CSR      ;IF INTERRUPT CLEARS
022666  001401                       BEQ     5$              ;THEN SKIP ERROR.

022670  104074                       EMT     74

022672  005037  001140          5$:  CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
022676  005037  002222               CLR     YLOOP           ;WAIT
022702  005237  002222          68$: INC     YLOOP
022706  023727  002222  177777       CMP     YLOOP,#-1
```

```
.MAIN.  MACRO M1110  20-FEB-79 11:22  PAGE 59-1                                  SEQ 0095
TESTING DIGITAL MODULE M6014

        022714  001372                          BNE     68$
  3702  022716  052737  000001  002244          BIS     #BIT0,TADDR     ;PREPARE TO READ HIGH BYTE
  3703  022724  117737  157314  001143          MOVB    @TADDR,$BDDAT+1 ;READ HIGH BYTE
  3704  022732  042737  000001  002244          BIC     #BIT0,TADDR     ;PREPARE TO READ LOW BYTE BUFFER
  3705  022740  117737  157300  001142          MOVB    @TADDR,$BDDAT   ;READ LOW BYTE BUFFER
  3706  022746  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF COUNTER HALTED AT 0
  3707  022754  001401                          BEQ     7$              ;-THEN SKIP ERROR
  3708
  3709  022756  104100                          EMT     100
  3710
  3711                                  ;-END 2$
  3712
  3713  022760  023737  003140  002244  7$:     CMP     BCNTL,TADDR     ;IF FINISHED CHECKING INTERRUPTS
  3714  022766  001405                          BEQ     TRANP1          ;-THEN TEST TRANSITION POINTS
  3715  022770  013737  003136  002244          MOV     BCNTH,TADDR     ;-ELSE CHECK COUNTER B
  3716  022776  000137  022372                  JMP     8$
```

```
3718 023002  112777  000030  157252  TRANP1: MOVB    #DBIT!TBIT,@CSR ;CLEAR EXTRANEOUS BITS AT CSR.
3719 023010  013737  003132  002244          MOV     ACNTH,TADDR     ;INITIALIZE COUNTER TO TEST
3720
3721 023016  012737  100000  003150  1$:     MOV     #BIT15,NUM      ;INITIALIZE TRANSITION NUMBER
3722
3723 023024  000257                  5$:     CCC                     ;CLEAR ALL CONDITION CODES
3724 023026  006037  003150                  ROR     NUM             ;DETERMINE NEXT TRANSITION VALUE
3725 023032  103010                          BCC     2$              ;IF NOT FINISHED THIS COUNTER DO 2$.
3726
3727 023034  023737  002244  003140  6$:     CMP     TADDR,BCNTL     ;IF COUNTER B WAS CHECKED
3728 023042  001462                          BEQ     SISTIH          ;-THEN CHECK BOTH TOGETHER
3729 023044  013737  003136  002244          MOV     BCNTH,TADDR     ;-ELSE PREPARE TO TEST COUNTER B
3730 023052  000761                          BR      1$              ;
3731
3732 023054  005037  003154          2$:     CLR     XLOOP           ;INITIALIZE LOOP COUNT
3733 023060  013737  003150  001140          MOV     NUM,$GDDAT      ;COMPUTE EXPECTED DATA
3734 023066  005337  001140                  DEC     $GDDAT
3735
3736 023072  152737  000001  002244          BISB    #BIT0,TADDR     ;PREPARE TO READ HIGH BYTE
3737 023100  113777  003151  157136          MOVB    NUM+1,@TADDR    ;LOAD  HIGH BYTE OF COUNTER
3738 023106  142737  000001  002244          BICB    #BIT0,TADDR     ;PREPARE TO LOAD COUNTER
3739 023114  113777  003150  157122          MOVB    NUM,@TADDR      ;LOAD COUNTER
3740
3741 023122  152737  000001  002244  3$:     BISB    #BIT0,TADDR     ;PREPARE TO READ HIGH BYTE
3742 023130  117737  157110  001143          MOVB    @TADDR,$BDDAT+1 ;READ HIGH BYTE
3743 023136  142737  000001  002244          BICB    #BIT0,TADDR     ;PREPARE TO READ LOW BYTE
3744 023144  117737  157074  001142          MOVB    @TADDR,$BDDAT   ;READ LOW BYTE
3745 023152  023737  003150  001142          CMP     NUM,$BDDAT      ;IF COUNT CHANGED
3746 023160  001005                          BNE     4$              ;-THEN CHECK DATA
3747 023162  005337  003154                  DEC     XLOOP           ;-ELSE IF DELAY NOT UP
3748 023166  001355                          BNE     3$              ;--THEN LOOK FOR A COUNT
3749
3750 023170  104101                          EMT     101
3751
3752 023172  000714                          BR      5$
3753
3754 023174  023737  001140  001142  4$:     CMP     $GDDAT,$BDDAT   ;IF COUNT WAS GOOD
3755 023202  001710                          BEQ     5$              ;-THEN SKIP ERROR
3756
3757 023204  104102                          EMT     102
3758
3759 023206  000706                          BR      5$              ;-THEN DO ANOTHER TRANSITION CHECK
3760
```

```
                                                ;TEST THAT COUNTERS COUNT TO WITHIN 20% OF 2KHZ

023210  105077  157716          STSTIM: CLRB    @ACNTH          ;LOAD THE HIGH BYTE BUFFER
023214  105077  157714                  CLRB    @ACNTL          ;CLEAR COUNTER A
023220  105077  157714                  CLRB    @BCNTL          ;CLEAR COUNTER B
023224  004737  040564                  JSR     PC,CLRINT       ;ASSURE PENDING INTERRUPTS ARE SERVICED

023230  142777  000010  157024          BICB    #TBIT,@CSR      ;PREPARE TO INITIALIZE BOARD
023236  112777  000030  157016          MOVB    #DBIT!TBIT,@CSR ;INITIALIZE BOARD
023244  004737  040564                  JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
023250  012777  040556  157044          MOV     #COUNT,@CLKVC   ;INITIALIZE CLOCK VECTOR
023256  012777  000340  157040          MOV     #PR7,@CLKVCA    ;SET UP PRIORITY FOR CLK INT
023264  012777  023534  156774          MOV     #4$,@VECTO      ;INITIALIZE INTERRUPT VECTOR
023272  012777  000340  156770          MOV     #PR7,@VECTOA    ;SET UP PRIORITY FOR INT

023300  012737  000620  003152          MOV     #400.,VARI      ;INITIALIZE ALLOWED ERROR
023306  012737  003720  001140          MOV     #2000.,$GDDAT   ;INITIALIZE EXPECTED DATA
023314  063737  003152  001140          ADD     VARI,$GDDAT     ;ADD FUDGE FACTOR

                                        ;ALLOW THE INTERRUPTS

023322  152777  000100  156732          BISB    #EBIT,@CSR      ;ENABLE INTERRUPT BIT AT CSR
023330  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
023334  012746  023342                  MOV     #64$,-(SP)
023340  000002                          RTI
023342  000240                  64$:    NOP

023344  113777  001141  157560          MOVB    $GDDAT+1,@ACNTH ;LOAD HIGH BYTE BUFFER
023352  005037  002300                  CLR     CLK
023356  005037  003154                  CLR     XLOOP           ;INITIALIZE CLOCK CHECK VARIANT
023362  113777  001140  157544          MOVB    $GDDAT,@ACNTL   ;LOAD COUNTER A
023370  113777  001140  157542          MOVB    $GDDAT,@BCNTL   ;LOAD COUNTER B

023376  005237  003154          8$:     INC     XLOOP           ;IF CLOCK SHOULD NOT HAVE INTERRUPTED
023402  001003                          BNE     9$              ;-THEN SEE IF CLOCK COUNTED

023404  104025                          EMT     25
023406  000137  023672                  JMP     CBITCL          ;--SKIP THIS TEST

023412  005737  002300          9$:     TST     CLK             ;IF CLOCK HASN'T COUNTED
023416  001767                          BEQ     8$              ;-THEN SEE IF IT SHOULD HAVE

023420  023727  002300  000074  1$:     CMP     CLK,#60.        ;IF 1 SECOND NOT UP
023426  001374                          BNE     1$              ;-THEN WAIT

023430  012746  000340                  MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
023434  012746  023442                  MOV     #65$,-(SP)
023440  000002                          RTI
023442  000240                  65$:    NOP
023444  142777  000110  156610          BICB    #EBIT!TBIT,@CSR ;DON'T LEAVE THESE BITS HANGING

023452  117737  157454  001143          MOVB    @ACNTH,$BDDAT+1 ;GET CONTENTS OF COUNTER A
023460  117737  157450  001142          MOVB    @ACNTL,$BDDAT
023466  117737  157444  003157          MOVB    @BCNTH,TEMPB+1  ;GET CONTENTS OF COUNTER B
023474  117737  157440  003156          MOVB    @BCNTL,TEMPB

023502  006337  003152                  ASL     VARI            ;DOUBLE THE VARIANCE
```

```
 3813
 3814 023506  023737  003152  001142          CMP     VARI,$PDDAT     ;IF COUNT IS WITHIN 20% OF 2KHZ
 3815 023514  100001                          BPL     2$              ;-THEN SKIP ERROR
 3816
 3817 023516  104106                          EMT     106
 3818
 3819 023520  023737  003152  003156  2$:     CMP     VARI,TEMPB      ;IF COUNT IS WITHIN 20% OF 2KHZ
 3820 023526  100001                          BPL     3$              ;-THEN SKIP ERROR
 3821
 3822 023530  104106                          EMT     106
 3823
 3824 023532  000457                  3$:     BR      CBITCL          ;CHECK THAT CBIT CLEARS MODULE.
```

```
                                                ;ON INTERRUPT DO 4$
                                
023534  022626                          4$:     CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
023536  013737  002300  002314                  MOV     CLK,TEMP1       ;GET TIME AT INTERRUPT
023544  013737  002276  002304                  MOV     BASE,TEMP       ;INITIALIZE TEMP
023552  117737  156506  002304                  MOVB    @IAR,TEMP       ;GET INTERRUPTING ADDRESS

023560  023737  003134  002304                  CMP     ACNTL,TEMP      ;IF NOT COUNTER A INTERRUPTING
023566  001005                                  BNE     5$              ;-THEN SKIP ERROR

023570  013737  003134  002244                  MOV     ACNTL,TADDR     ;-ELSE PREPARE FOR ERROR TYPEOUT
023576  104106                                  EMT     106
023600  000425                                  BR      7$              ;CLEAN UP

023602  023737  003140  002304          5$:     CMP     BCNTL,TEMP      ;IF NOT COUNTER B INTERRUPTING
023610  001005                                  BNE     6$              ;-THEN SKIP ERROR

023612  013737  003140  002244                  MOV     BCNTL,TADDR     ;-ELSE PREPARE FOR ERROR TYPEOUT
023620  104106                                  EMT     106

023622  000414                                  BR      7$              ;CLEAN UP

023624  152777  000001  156430          6$:     BISB    #RBIT,@CSR      ;PREPARE TO CLEAR UNEXPECTED INTERRUPT
023632  105777  156446                          TSTB    @TEMP           ;CLEAR THE INTERRUPT
023636  012746  000000                          MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
023642  012746  023650                          MOV     #66$,-(SP)
023646  000002                                  RTI
023650  000240                          66$:    NOP
023652  000662                                  BR      1$              ;SEE IF TIME IS UP

023654  004737  040564                  7$:     JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
023660  132777  000200  156374                  BITB    #FBIT,@CSR      ;IF INTERRUPT CLEARS
023666  00140'                                  BEQ     CBITCL          ;-THEN SKIP ERROR
023670  104104                                  EMT     104
```

```
3858                                            ;CHECK THAT CBIT INITIALIZES MODULE
3859
3860 023672  005037  001140          CBITCL: CLR     $GDDAT          ;INITIALIZE EXPECTED DATA.
3861 023676  112777  177777  157226          MOVB    #-1,@ACNTH      ;LOAD THE HIGH BYTE BUFFER.
3862 023704  112777  177777  157222          MOVB    #-1,@ACNTL      ;LOAD COUNTER A WITH ALL 1'S.
3863 023712  112777  177777  157220          MOVB    #-1,@BCNTL      ;LOAD COUNTER B WITH ALL 1'S.
3864 023720  112777  000020  156334          MOVB    #DBIT,@CSR      ;ASSURE CSR IS CLEAR.
3865
3866 023726  004737  032024                  JSR     PC,KLEER        ;INITIALIZE SYSTEM
3867 023732  152777  000020  156322          BISB    #DBIT,@CSR      ;SHUT OFF WORLD
3868
3869 023740  013737  003132  002244          MOV     ACNTH,TADDR     ;INITIALIZE FIRST TEST ADDRESS
3870
3871 023746  117737  156272  001143  1$:     MOVB    @TADDR,$BDDAT+1 ;READ HIGH BYTE OF TEST COUNTER
3872 023754  142737  000001  002244          BICB    #BIT0,TADDR     ;PREPARE TO READ LOW BYTE.
3873 023762  117737  156256  001142          MOVB    @TADDR,$BDDAT   ;READ LOW BYTE OF TEST COUNTER
3874 023770  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
3875 023776  001401                          BEQ     2$              ;-THEN SKIP ERROR
3876
3877 024000  104075                          EMT     75
3878
3879 024002  023737  003140  002244  2$:     CMP     BCNTL,TADDR     ;IF DONE CHECKING INIT
3880 024010  001404                          BEQ     3$              ;-THEN RETURN FROM SUBROUTINE
3881 024012  013737  003136  002244          MOV     BCNTH,TADDR     ;-ELSE TEST COUNTER B.
3882 024020  000752                          BR      1$
3883 024022  013737  003134  002244  3$:     MOV     ACNTL,TADDR     ;RESTORE ORIGINAL TEST ADDRESS
3884 024030  000207                          RTS     PC              ;RETURN TO MAIN PROGRAM.
```

```
                                .SBTTL  FIELD TESTING DIGITAL MODULE M5016
                                ;M5016 QUAD 8-BIT COUNTER/PRESCALER MODULE TEST
                                ;
                                ;               TEST FLOW
                                ;
                                ;       1.  CLEAR MODULE WITH CBIT
                                ;       2.  CHECK THAT STATUS REGISTER IS CLEAR
                                ;       3.  CHECK THAT ALL COUNTERS ARE CLEAR
                                ;       4.  TEST STATUS REGISTER LOAD AND READ
                                ;       5.  TEST THAT CLEAR ENABLE BIT SELF CLEARS
                                ;       6.  CHECK THAT STATUS REGISTER CLEAR BIT CLEARS
                                ;             EACH COUNTER INDIVIDUALLY
                                ;
                                ;      STORE DIP SWITCH SETTINGS FOR THE FOLLOWING
                                ;      AND PRINT THE SETTINGS IF NOT IN APT MODE
                                ;
                                ;       7.  CHECK THAT COUNTERS COUNT BINARILY UPWARD
                                ;       8.  CHECK THAT COUNTER OVERFLOWS
                                ;       9.  DETERMINE IF LEGAL RADIX
                                ;      10.  DETERMINE IF COUNTERS RESET ON OVERFLOW
                                ;      11.  CHECK THAT OVERFLOW CLEARS
                                ;      12.  CHECK THAT CBIT CLEARS MODULE
                                ;
                                ;LOCAL VARIABLES

024032  171000                  SR5016: .WORD   171000          ;ADDRESS OF STATUS REGISTER
024034  171001                  DR5016: .WORD   171001          ;ADDRESS OF DATA REGISTER

024036  000020                  OVFS:   .WORD   BIT4            ;OVERFLOW BIT FOR COUNTER 0
024040  000040                          .WORD   BIT5            ;OVERFLOW BIT FOR COUNTER 1
024042  000100                          .WORD   BIT6            ;OVERFLOW BIT FOR COUNTER 2
024044  000200                          .WORD   BIT7            ;OVERFLOW BIT FOR COUNTER 3


024046  000000                  DSWCH:  .WORD   0               ;COUNTER 0 DIP SWITCH SETTINGS
024050  000000                          .WORD   0               ;COUNTER 1 DIP SWITCH SETTINGS
024052  000000                          .WORD   0               ;COUNTER 2 DIP SWITCH SETTINGS
024054  000000                          .WORD   0               ;COUNTER 3 DIP SWITCH SETTINGS

024056      012     015     040 CTASCI: .ASCIZ  <12><15>/ CTN   1  2  3  4  5  6  7  8  9 10  ADDRESS   /
024061      103     124     116
024064      040     040     040
024067      061     040     040
024072      062     040     040
024075      063     040     040
024100      064     040     040
024103      065     040     040
024106      066     040     040
024111      067     040     040
024114      070     040     040
024117      071     040     061
024122      060     040     040
024125      101     104     104
024130      122     105     123
024133      123     040     075
024136      040     000
024140      040     040     000 BLANK2: .ASCIZ  /  /
```

```
3927                                    .EVEN
```

```
 3929 024144                                M5016:
                                            ;;*****************************************************************
      024144  000004                        TST24:  SCOPE
 3930 024146  012737  000024  001206               MOV     #24,$TESTN      ;LOAD THE TEST NUMBER
 3931
 3932                                               ;START WITH A KNOWN CONDITION -- CLEAR THE MODULE
 3933
 3934 024154  004737  032024                        JSR     PC,KLEER        ;CLEAR MODULE.
 3935 024160  152777  000020  156074               BISB    #DBIT,@CSR      ;SHUT OFF WORLD
 3936 024166  013737  002244  024032               MOV     IADDR,SR5016    ;INITIALIZE STATUS REG ADDRESS
 3937 024174  013737  002244  024034               MOV     IADDR,DR5016    ;INITIALIZE DATA REG ADDRESS
 3938 024202  052737  000001  024034               BIS     #BIT0,DR5016
 3939 024210  104411                               SAVREG                  ;SAVE ALL REGISTERS
 3940
 3941
 3942                                               ;STATUS OF M5016/PRESCALER AT THIS TIME SHOULD BE --
 3943
 3944                                               ;       STATUS REGISTER = 0     DATA REGISTER - 0
 3945
 3946                                               ;       -CONTENTS OF COUNTERS ARE-
 3947
 3948                                               ;       COUNTER 0       CONTAINS 0
 3949                                               ;       COUNTER 1       CONTAINS 0
 3950                                               ;       COUNTER 2       CONTAINS 0
 3951                                               ;       COUNTER 3       CONTAINS 0
 3952
 3953 024212  005037  001140                        CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
 3954
 3955                                               ;CHECK THAT STATUS REGISTER IS CLEAR.
 3956
 3957 024216  117737  177610  001142               MOVB    @SR5016,$BDDAT  ;READ STATUS REGISTER
 3958 024224  123737  001140  001142               CMPB    $GDDAT,$BDDAT   ;IF STATUS REGISTER IS CLEAR
 3959 024232  001401                               BEQ     CK4CLX          ;-THEN CHECK COUNTERS ARE CLEAR
 3960
 3961 024234  104075                               EMT     75
```

```
                                        ;CHECK THAT ALL COUNTERS ARE CLEAR
                                        
024236  005037  003146          CK4CLX: CLR     CNUM            ;INITIALIZE COUNTER NUMBER
024242  004737  040646                  JSR     PC,CNTRC        ;CONTROL C ??????
                                        
                                ;DO 1$ WHILE (COUNTER # "CNUM" IS GREATER THAN OR EQUAL TO 3)
                                        
024246  022737  000003  003146  1$:     CMP     #3,CNUM         ;IF COUNTER # IS NOT LESS THAN 3
024254  002416                          BLT     CKRWSX          ;-THEN CHECK R/W OF STAT REG
024256  113777  003146  177546          MOVB    CNUM,@SR5016    ;-ELSE LOAD STATUS REG WITH COUNTER #
024264  117737  177544  001142          MOVB    @DR5016,$BDDAT  ;READ DATA REG
024272  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;IF COUNTER 'CNUM' IS CLEAR
024300  001401                          BEQ     2$              ;-THEN SKIP ERROR
                                        
024302  104116                          EMT     116
                                        
024304  005237  003146          2$:     INC     CNUM            ;INCREMENT COUNTER NUMBER
024310  000756                          BR      1$              ;SEE IF DONE
                                        
                                ;-END 1$
                                        
                                        ;TEST PRESCALER STATUS REGISTER LOAD & READ
                                        
024312  112737  177777  001140  CKRWSX: MOVB    #-1,$GDDAT      ;INIITIALIZE LOAD PATTERN.
024320  005037  001142                  CLR     $BDDAT          ;INITIALIZE BAD DATA STORAGE LOCATION
024324  004737  040646                  JSR     PC,CNTRC        ;CONTROL C ??????
                                        
                                        ;CHECK R/W BITS BY LOADING COUNTS FROM 1 TO 4
                                ;DO 1$ WHILE (PATTERN "$GDDAT" IS LESS THAN OR EQUAL TO 4)
                                        
024330  105237  001140          1$:     INCB    $GDDAT          ;CHANGE LOAD PATTERN.
024334  122737  000004  001140          CMPB    #4,$GDDAT       ;IF DONE CHECKING R/W
024342  100414                          BMI     CKSRCX          ;-THEN CHECK SELF CLEAR ENABLE.
                                        
                                        ;TEST LOAD & READ
                                        
024344  113777  001140  177460          MOVB    $GDDAT,@SR5016  ;LOAD PRESCALER STATUS REGISTER.
024352  117737  177454  001142          MOVB    @SR5016,$BDDAT  ;GET CONTENTS OF STATUS REGISTER
024360  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;IF STAT REG LOADED CORRECTLY
024366  001760                          BEQ     1$              ;-THEN CHANGE LOAD PATTERN
                                        
024370  104111                          EMT     111
                                        
024372  000756                          BR      1$              ;  CONTINUE READ/WRITE TESTING.
                                        
                                .-END 1$
```

```
                                        ;STATUS OF M5016/PRESCALER AT THIS TIME SHOULD BE --
                                        ;       STATUS REGISTER = 4     DATA REGISTER   0
                                        ;       -CONTENTS OF COUNTERS ARE-
                                        ;       COUNTER 0       CONTAINS 0
                                        ;       COUNTER 1       CONTAINS 0
                                        ;       COUNTER 2       CONTAINS 0
                                        ;       COUNTER 3       CONTAINS 0
                                        ;TEST THAT CLEAR ENABLE BIT SELF CLEARS
024374  005037  001140          CKSRCX: CLR     $GDDAT          ;PLACE EXPECTED DATA IN $GDDAT
024400  004737  040646                  JSR     PC,CNTRC        ;CONTROL C ???????
024404  105777  177424                  TSTB    @DR5016         ;CLEAR STATUS REG CLEAR BIT
024410  117737  177416  001142          MOVB    @SR5016,$BDDAT  ;READ STATUS REGISTER
024416  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF STAT REG CLEARS SELF
024424  001401                          BEQ     CCNTCX          ;-THEN CHECK THAT COUNTERS CLEAR
024426  104112                          EMT     112
                                        ;CHECK THAT STATUS REG CLEARS EACH COUNTER INDIVIDUALLY
024430                          CCNTCX:
024430  012746  000340                  MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
024434  012746  024442                  MOV     #64$,-(SP)
024440  000002                          RTI
024442  000240                  64$:    NOP
024444  005037  003146                  CLR     CNUM            ;INITIALIZE COUNTER NUMBER
024450  142777  000010  155604          BICB    #TBIT,@CSR      ;ASSURE TBIT IS CLEAR
024456  152777  000010  155576          BISB    #TBIT,@CSR      ;TOGGLE THE COUNTERS
024464  142777  000010  155570          BICB    #TBIT,@CSR
```

```
                                        ;DO 1$ WHILE (THE NUMBER OF COUNTERS "CNUM" <= 3 )
                                        
024472  023727  003146  000003  1$:     CMP     CNUM,#3         ;IF DONE CLEARING ALL COUNTERS
024500  003036                          BGT     BINCNX          ;-THEN CHECK FOR BINARY COUNT
                                        
024502  113777  003146  177322          MOVB    CNUM,@SR5016    ;LOAD COUNTER NUMBER TO TEST
024510  152777  000004  177314          BISB    #4,@SR5016      ;CLEAR COUNTER AFTER READ
024516  012737  000001  001140          MOV     #1,$GDDAT       ;INITIALIZE EXPECTED DATA
024524  117737  177304  001142          MOVB    @DR5016,$BDDAT  ;READ COUNTER
024532  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
024540  001401                          BEQ     2$              ;-THEN SKIP ERROR
                                        
024542  104113                          EMT     113
                                        
024544  005037  001140          2$:     CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
024550  117737  177260  001142          MOVB    @DR5016,$BDDAT  ;READ COUNTER CNUM
024556  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
024564  001401                          BEQ     3$              ;-THEN SKIP ERROR
                                        
024566  104112                          EMT     112 -
                                        
024570  005237  003146          3$:     INC     CNUM            ;INCREMENT VARIANT
024574  000736                          BR      1$              ;SEE IF DONE
                                        
                                        ;-END 1$
```

```
4070 024576                                BINCNX:
     024576  012746  000340                        MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
     024602  012746  024610                        MOV     #64$,-(SP)
     024606  000002                                RTI
     024610  000240                        64$:    NOP
4071 024612  004737  040646                        JSR     PC,CNTRC        ;CONTROL C ??????
4072
4073 024616  012737  000010  003150                MOV     #10,NUM         ;INITIALIZE OVERFLOW POINTER
4074 024624  012737  177777  003146                MOV     #-1,CNUM        ;INITIALIZE COUNTER NUMBER
4075
4076                                               ;INITIALIZE COUNTER DIP SWITCH SETTINGS
4077
4078 024632  005005                                CLR     R5              ;INITIALIZE POINTER REGISTER
4079 024634  005065  024046                10$:    CLR     DSWCH(R5)       ;CLEAR CONTENTS OF DIP TABLE
4080 024640  005725                                TST     (R5)+           ;ADVANCE POINTER
4081 024642  022705  000010                        CMP     #10,R5          ;IF NOT FINISHED
4082 024646  001372                                BNE     10$             ;-THEN CONTINUE
4083
4084                                       ;DO 1$ WHILE ( CNUM<=3 )
4085
4086 024650  005237  003146                1$:     INC     CNUM            ;DETERMINE COUNTER NUMBER
4087 024654  023727  003146  000003                CMP     CNUM,#3         ;IF FINISHED BINARY COUNTS
4088 024662  003141                                BGT     CK4INX          ;-THEN CHECK FOR INTERRUPTS
4089
4090 024664  113777  003146  177140                MOVB    CNUM,@SR5016    ;LOAD COUNTER NUMBER TO TEST
4091 024672  152777  000004  177132                BISB    #BIT2,@SR5016   ;PREPARE TO CLEAR COUNTER CNUM
4092 024700  105777  177130                        TSTB    @DR5016         ;CLEAR COUNTER CNUM
4093 024704  152777  000001  155350                BISB    #RBIT,@CSR      ;PREPARE TO CLEAR OVERFLOW
4094 024712  105777  177114                        TSTB    @SR5016         ;CLEAR POSSIBLE OVERFLOW
4095
4096 024716  005037  001140                        CLR     $GDDAT          ;INITIALIZE BINARY COUNT
4097 024722  006337  003150                        ASL     NUM             ;INITIALIZE OVERFLOW POINTER
4098
4099                                       ;-DO 2$ WHILE ( NO OVERFLOW AND CORRECT COUNT )
4100 024726  142777  000010  155326        2$:     BICB    #TBIT,@CSR      ;ASSURE TBIT IS CLEAR
4101 024734  152777  000010  155320                BISB    #TBIT,@CSR      ;CLOCK THE COUNTER
4102 024742  142777  000010  155312                BICB    #TBIT,@CSR      ;DON'T LEAVE BITS HANGING
4103
4104 024750  005237  001140                        INC     $GDDAT          ;TALLY THE COUNTS
4105 024754  133777  003150  177050                BITB    NUM,@SR5016     ;IF COUNTER CNUM OVERFLOWS
4106 024762  001021                                BNE     4$              ;-DETERMINE IF LEGAL RADIX
4107
4108 024764  117737  177044  001142                MOVB    @DR5016,$BDDAT  ;READ COUNTER CNUM
4109 024772  105037  001143                        CLRB    $BDDAT+1        ;CLEAR EXTRANEOUS DATA
4110 024776  023737  001140  001142                CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
4111 025004  001750                                BEQ     2$              ;-THEN DO NEXT COUNT
4112                                       ;--END 2$
4113
4114 025006  023727  001140  000400                CMP     $GDDAT,#400     ;-ELSE IF COUNT REACHES 400
4115 025014  001402                                BEQ     3$              ;--THEN NOTE OVERFLOW ERROR
4116
4117 025016  104113                                EMT     113
4118 025020  000713                                BR      1$              ;---DO NEXT COUNTER
4119
4120 025022                                3$:
     025022  104114                                EMT     114
4121 025024  000711                                BR      1$              ;--DO NEXT COUNTER
```

```
                                        ;DETERMINE IF LEGAL RADIX
                                
025026  012737  000001  002304  4$:     MOV     #BIT0,TEMP      ;INITIALIZE RADIX LOOP VARIANT
                                
                                ;-DO 40$ WHILE ( NOT ALL LEGAL RADII TESTED )
                                
025034  022737  000400  002304  40$:    CMP     #BIT8,TEMP      ;IF ALL LEGAL RADII NOT TESTED
025042  001002                          BNE     5$              ;-THEN SKIP ERROR
                                
025044  104117                          EMT     117
025046  000700                          BR      1$              ;--DO NEXT COUNTER
                                
025050  006337  002304          5$:     ASL     TEMP            ;COMPUTE LEGAL RADIX TO TEST
025054  023737  002304  001140          CMP     TEMP,$GDDAT     ;IF NOT CORRECT RADIX
025062  001364                          BNE     40$             ;-THEN SEE IF ALL RADII TESTED
                                
                                ;--END 40$
                                
025064  006237  002304                  ASR     TEMP            ;-ELSE COMPUTE CORRECT SWITCH
025070  013705  003146                  MOV     CNUM,R5         ;--INITIALIZE OFFSET
025074  006305                          ASL     R5
025076  053765  002304  024046          BIS     TEMP,DSWCH(R5)  ;--SET CORRESPONDING DIP SWITCH
                                
                                        ;CHECK THAT DATA IS RESET ON OVERFLOW
                                
025104  117737  176724  001142          MOVB    @DR5016,$BDDAT  ;--IF COUNTER CNUM NOT ZERO
025112  001003                          BNE     6$              ;---THEN SKIP DIP SWITCH SET
                                
025114  052765  001000  024046          BIS     #BIT9,DSWCH(R5) ;---ELSE COUNTER RESETS ON OVF
                                
                                        ;CHECK THAT OVERFLOW CLEARS
                                
025122  152777  000001  155132  6$:     BISB    #RBIT,@CSR
025130  105777  176676                  TSTB    @SR5016         ;CLEAR THE OVERFLOW
025134  005037  001142                  CLR     $GDDAT          ;INITIALIZE DATA WORD
025140  117737  176666  001142          MOVB    @SR5016,$BDDAT  ;READ STATUS REGISTER
025146  005037  001140                  CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
025152  033737  003150  001142          BIT     NUM,$BDDAT      ;IF DATA IS AS EXPECTED
025160  001633                          BEQ     1$              ;-THEN TEST NEXT COUNTER
                                
025162  104115                          EMT     115
025164  000631                          BR      1$              ;--TEST NEXT COUNTER
                                
                                ;-END 1$
```

```
                                        ;CHECK COUNTER FOR INTERRUPTS
                                        ;
025166  012777  025576  155072  CK4INX: MOV     #6$,@VECTO      ;LOAD INTERRUPT VECTOR
025174  012777  000340  155066          MOV     #PR7,@VECTOA    ;LOCK OUT OTHER INTERRUPTS
025202  004737  040646                  JSR     PC,CNTRC        ;CONTROL C ??????
025206  012737  177777  003146          MOV     #-1,CNUM        ;INITIALIZE COUNTER NUMBER
025214  005037  003156                  CLR     TEMPB           ;INITIALIZE INTERRUPT FLAG
                                        ;
                                    ;DO 1$ WHILE (CNUM <= 3)
                                        ;
025220  005237  003146          1$:     INC     CNUM            ;ADVANCE COUNTER NUMBER
025224  022737  000003  003146          CMP     #3,CNUM         ;IF NOT FINISHED TESTING INTERRUPTS
025232  002002                          BGE     11$             ;-THEN CONTINUE
025234  000137  025770                  JMP     9$              ;-ELSE CHECK INTERRUPT ERROR
                                        ;
025240  013705  003146          11$:    MOV     CNUM,R5         ;INITIALIZE OFFSET
025244  006305                          ASL     R5
025246  016537  024046  003150          MOV     DSWCH(R5),NUM   ;COMPUTE FULL COUNT
025254  042737  177400  003150          BIC     #177400,NUM     ;-CLEAR EXTRANEOUS BITS
025262  006337  003150                  ASL     NUM             ;-GET RADIX
025266  005337  003150                  DEC     NUM             ;-GET FULL COUNT
                                        ;
                                        ;START COUNTER AT 0
                                        ;
025272  113777  003146  176532          MOVB    CNUM,@SR5016    ;LOAD COUNTER NUMBER TO CLEAR
025300  152777  000004  176524          BISB    #4,@SR5016      ;PREPARE TO CLEAR COUNTER
025306  105777  176522                  TSTB    @DR5016         ;CLEAR THE COUNTER
025312  005037  001140                  CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
025316  117737  176512  001142          MOVB    @DR5016,$BDDAT  ;IF THE COUNTER CNUM CLEARS
025324  001401                          BEQ     20$             ;-THEN SKIP THE ERROR
                                        ;
025326  104112                          EMT     112
                                        ;
025330  005037  002304          20$:    CLR     TEMP            ;INITIALIZE LOOP VARIANT
                                    ;-DO 2$ WHILE ( TEMP <- NUM )
                                        ;
025334  142777  000010  154720  2$:     BICB    #TBIT,@CSR      ;ASSURE TBIT IS CLEAR
025342  152777  000010  154712          BISB    #TBIT,@CSR      ;CLOCK COUNTER
025350  142777  000010  154704          BICB    #TBIT,@CSR      ;DON'T LEAVE BITS HANGING
025356  005237  002304                  INC     TEMP            ;TALLY THE COUNTS
025362  023737  003150  002304          CMP     NUM,TEMP        ;IF NOT FULL COUNT
025370  001361                          BNE     2$              ;-THEN CONTINUE COUNTING
                                    ;--END 2$
                                        ;
025372  005037  002304                  CLR     TEMP            ;INITIALIZE LOOP VARIANT
                                        ;
                                        ;CLEAR ALL OTHER COUNTERS
                                    ;-DO 3$ WHILE (TEMP < 4)
                                        ;
025376  023737  003146  002304  3$:     CMP     CNUM,TEMP       ;IF CNUM - TEMP
025404  001421                          BEQ     4$              ;-THEN ADVANCE VARIANT
                                        ;
025406  022737  000004  002304          CMP     #4,TEMP         ;-ELSE IF 4 <= TEMP
025414  003420                          BLE     5$              ;--THEN FORCE THE INTERRUPT
                                        ;
025416  112777  000004  176406          MOVB    #4,@SR5016      ;--ELSE PREPARE TO CLEAR COUNTER
025424  153777  002304  176400          BISB    TEMP,@SR5016    ;--LOAD COUNTER NUMBER "TEMP"
```

```
4226 025432  105777  176376           TSTB    @DR5016         ;--CLEAR COUNTER
4227 025436  152777  000001  154616   BISB    #RBIT,@CSR      ;PREPARE TO CLEAR ALL OVERFLOWS
4228 025444  105777  176362           TSTB    @SR5016         ;CLEAR THE OVERFLOW
4229
4230 025450  005237  002304   4$:     INC     TEMP            ;ADVANCE COUNTER NUMBER
4231 025454  000750                   BR      3$              ;CHECK IF DONE
4232
4233                          ;--END 3$
4234
4235 025456  004737  040564   5$:     JSR     PC,CLRINT       ;CLEAR ALL POSSIBLE INTERRUPTS
4236
4237                          ;FORCE AN INTERRUPT
4238
4239 025462  012746  000000           MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     025466  012746  025474           MOV     #64$,-(SP)
     025472  000002                   RTI
     025474  000240           64$:    NOP
4240 025476  142777  000010  154556   BICB    #TBIT,@CSR      ;ASSURE TBIT IS CLEAR
4241 025504  152777  000010  154550   BISB    #TBIT,@CSR      ;CLOCK THE COUNTERS
4242 025512  142777  000010  154542   BICB    #TBIT,@CSR      ;DON'T LEAVE BITS HANGING
4243 025520  152777  000100  154534   BISB    #EBIT,@CSR      ;ENABLE INTERRUPTS
4244
4245 025526  005037  002222           CLR     YLOOP           ;WAIT
     025532  005237  002222   65$:    INC     YLOOP
     025536  023727  002222  000077   CMP     YLOOP,#77
     025544  001372                   BNE     65$
4246
4247 025546  142777  000100  154506   BICB    #EBIT,@CSR      ;DON'T ALLOW INTERRUPTS
4248 025554  012746  000340           MOV     #PR7,-(SP)      ;SET PSW TO PRIORITY 7
     025560  012746  025566           MOV     #66$,-(SP)
     025564  000002                   RTI
     025566  000240           66$:    NOP
4249 025570  000613                   BR      1$              ;TRY NEXT COUNTER
4250
4251                          ;ON TIMEOUT RESTORE STACK
4252
4253 025572  022626           17$:    CMP     (SP)+,(SP)+     ;RESTORE STACK.
4254 025574  000432                   BR      16$             ;ADD TO INTERRUPT COUNT
4255
4256                          ;ON INTERRUPT DO 6$
4257
4258 025576  022626           6$:     CMP     (SP)+,(SP)+     ;RESTORE STACK.
4259 025600  013704  000004           MOV     ERRVEC,R4       ;SAVE ERROR VECTOR.
4260 025604  012737  025572  000004   MOV     #17$,ERRVEC     ;ON TIMEOUT GO TO 17$.
4261 025612  142777  000010  154442   BICB    #TBIT,@CSR      ;ASSURE THAT TBIT IS CLEARED.
4262 025620  005037  002314           CLR     TEMP1           ;INITIALIZE INTERRUPT TALLY.
4263 025624  013737  002276  002304   MOV     BASE,TEMP       ;INITIALIZE TEMP
4264 025632  152777  000001  154422   7$:     BISB    #RBIT,@CSR      ;PREPARE TO RESET INTERRUPT.
4265
4266                          ;CHECK IF PRESCALER INTERRUPTED
4267
4268 025640  117737  154420  002304   MOVB    @IAR,TEMP       ;ASSEMBLE INTERRUPTING ADDRESS.
4269 025646  123737  002304  024032   CMPB    TEMP,SR5016     ;IF PRESCALER INTERRUPTS
4270 025654  001417                   BEQ     8$              ;-THEN SET APPROPRIATE DSWTCH
4271
4272                          ;CLEAR THE UNWANTED INTERRUPT
4273
```

```
4274 025656  105777  154422            TSTB    @TEMP           ;-ELSE CLEAR INTERRUPT.
4275 025662  005237  002314      16$:  INC     TEMP1           ;COUNT NUMBER OF INTERRUPTS.
4276 025666  022737  000400  002314    CMP     #400,TEMP1      ;IF 400 INTERRUPTS NOT COUNTED
4277 025674  001356                    BNE     7$              ;-THEN TRY AGAIN.
4278
4279 025676  142777  000100  154356    BICB    #EBIT,@CSR      ;DISABLE INTERRUPTS
4280 025704  010437  000004            MOV     R4,ERRVEC       ;RESTORE ERROR VECTOR
4281 025710  000137  025220            JMP     1$              ;TRY THE NEXT COUNTER
4282
4283 025714  052765  000400  024046 8$:  BIS     #BIT8,DSWCH(R5) ;SET APPROPRIATE DSWTCH
4284 025722  012737  000001  003156    MOV     #1,TEMPB        ;SET INTERRUPT OCCURED FLAG
4285
4286                                    ;CHECK FOR ILLEGAL OVERFLOWS
4287
4288 025730  016537  024036  001140    MOV     OVFS(R5),$GDDAT ;INITIALIZE EXPECTED OVF DATA
4289 025736  117737  176070  001142    MOVB    @SR5016,$BDDAT  ;GET STATUS REGISTER DATA
4290 025744  042737  177417  001142    BIC     #177417,$BDDAT  ;CLEAR EXTRANEOUS DATA
4291 025752  023737  001140  001142    CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
4292 025760  001401                    BEQ     18$             ;-THEN SKIP OVERFLOW ERROR
4293
4294 025762  104114                    EMT     114
4295
4296 025764  000137  025220      18$:  JMP     1$              ;TRY NEXT COUNTER
4297
4298                              ;--END 4$
4299
4300
4301 025770  005737  003156      9$:   TST     TEMPB           ;IF THERE WERE NO INTERRUPTS
4302 025774  001001                    BNE     10$             ;-THEN CHECK IF UNDER APT
4303
4304 025776  104103                    EMT     103
4305
4306 026000  032737  000001  001222 10$:  BIT     #BIT0,$ENV      ;IF UNDER APT
4307 026006  001074                    BNE     FHOUSX          ;-THEN SKIP TYPEOUT
4308
```

```
4310 026010  005737  001210                TST     $PASS           ;IF NOT FIRST PASS
4311 026014  001071                        BNE     FHOUSX          ;-THEN SKIP TYPEOUT
4312
4313                                       ;TYPE THE DIP SWITCH SETTINGS IF IN STAND ALONE MODE
4314
4315 026016  005037  003154                CLR     XLOOP           ;INITIALIZE TYPE LOOP VARIANT
4316
4317
4318 026022  104401  024056          TYPDSW: TYPE    ,CTASCI         ;PRINT A HEADING
4319 026026  013746  002244                MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
4320 026032  104402                        TYPOC                   ;TYPE THE ADDRESS
4321 026034  104401  001177                TYPE    ,$CRLF          ;TYPE  CARRIAGE RETURN LINE FEED
4322
4323                                       ;DO 5$ WHILE ( DIP SWITCH SETTINGS NOT ALL PRINTED )
4324
4325 026040  013737  003154  002304  5$:   MOV     XLOOP,TEMP      ;SETUP FOR TYPING COUNTER NUM
4326 026046  062737  000060  002304        ADD     #60,TEMP        ;CONVERT TO ASCII NUMBER
4327 026054  104401  024140                TYPE    ,BLANK2         ;TYPE TWO BLANKS
4328 026060  104401  002304                TYPE    ,TEMP           ;TYPE COUNTER NUMBER
4329 026064  104401  024140                TYPE    ,BLANK2         ;TYPE ANOTHER TWO BLANKS
4330 026070  013705  003154                MOV     XLOOP,R5        ;COMPUTE DSWCH OFFSET
4331 026074  006305                        ASL     R5              ;MULTIPLY COUNTER NUMBER BY 2
4332
4333 026076  012737  000001  002222        MOV     #BIT0,YLOOP     ;INITIALIZE SECOND LOOP VARIANT
4334
4335                                       ;-DO 1$ WHILE (YLOOP < #BIT10 )
4336
4337 026104  104401  024140          1$:   TYPE    ,BLANK2         ;TYPE 2 BLANKS
4338 026110  004737  040646                JSR     PC,CNTRC        ;CONTROL C ????????
4339 026114  033765  002222  024046        BIT     YLOOP,DSWCH(R5) ;IF THIS SWITCH IS ON
4340 026122  001005                        BNE     2$              ;-THEN TYPE AN ASCII 1
4341
4342 026124  104401  026132                TYPE    ,10$            ;-ELSE TYPE AN ASCII 0
4343 026130  000404                        BR      3$              ;-SKIP TYPING THE ASCII 1
4344
4345 026132  000060                10$:    .WORD   '0              ;AN ASCII ZERO
4346 026134  000061                11$:    .WORD   '1              ;AN ASCII ONE
4347
4348 026136  104401  026134          2$:   TYPE    ,11$            ;TYPE AN ASCII 1
4349
4350 026142  006337  002222          3$:   ASL     YLOOP           ;DETERMINE NEXT SWITCH
4351 026146  032737  002000  002222        BIT     #BIT10,YLOOP    ;IF FINISHED THIS SWITCH PAK
4352 026154  001001                        BNE     4$              ;-THEN TEST THE NEXT
4353 026156  000752                        BR      1$              ;-ELSE DO NEXT SWITCH
4354
4355                                       ;-END 1$
4356
4357 026160  104401  001177          4$:   TYPE    ,$CRLF          ;TYPE CARRIAGE RETURN LINE FEED
4358 026164  005237  003154                INC     XLOOP           ;INCREMENT LOOP VARIANT
4359 026170  022737  000004  003154        CMP     #4,XLOOP        ;IF NOT FINISHED ALL COUNTERS
4360 026176  001320                        BNE     5$              ;-THEN DO NEXT SWITCH PAK
4361
4362                                       ;-END 5$
```

```
4364
4365 026200  004737  032024          FMOUSX: JSR     PC,KLEER        ;CLEAR MODULE.
4366 026204  152777  000020  154050          BISB    #DBIT,@CSR      ;SHUT OFF WORLD
4367
4368                                         ;STATUS OF M5016/PRESCALER AT THIS TIME SHOULD BE --
4369
4370                                         ;       STATUS REGISTER  0     DATA REGISTER   0
4371
4372                                         ;       -CONTENTS OF COUNTERS ARE-
4373
4374                                         ;       COUNTER 0       CONTAINS 0
4375                                         ;       COUNTER 1       CONTAINS 0
4376                                         ;       COUNTER 2       CONTAINS 0
4377                                         ;       COUNTER 3       CONTAINS 0
4378
4379 026212  005037  001140                  CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
4380
4381                                         ;CHECK THAT STATUS REGISTER IS CLEAR.
4382
4383 026216  117737  175610  001142          MOVB    @SR5016,$BDDAT  ;READ STATUS REGISTER
4384 026224  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;IF STATUS REGISTER IS CLEAR
4385 026232  001401                          BEQ     CK4CXX          ;-THEN CHECK COUNTERS ARE CLEAR
4386
4387 026234  104075                          EMT     75
```

```
4389
4390                                            ;CHECK THAT ALL COUNTERS ARE CLEAR
4391
4392 026236  005037  003146          CK4CXX: CLR     CNUM            ;INITIALIZE COUNTER NUMBER
4393
4394                                    ;DO 1$ WHILE (COUNTER # "CNUM" IS GREATER THAN OR EQUAL TO 3)
4395
4396 026242  022737  000003  003146  1$:     CMP     #3,CNUM         ;IF COUNTER # IS NOT LESS THAN 3
4397 026250  002416                          BLT     MAXRDX          ;-THEN GET MAXIMUM RADIX
4398 026252  113777  003146  175552          MOVB    CNUM,@SR5016    ;-ELSE LOAD STATUS REG WITH COUNTER #
4399 026260  117737  175550  001142          MOVB    @DR5016,$BDDAT  ;READ DATA REG
4400 026266  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;IF COUNTER 'CNUM' IS CLEAR
4401 026274  001401                          BEQ     2$              ;-THEN SKIP ERROR
4402
4403 026276  104116                          EMT     116
4404
4405 026300  005237  003146          2$:     INC     CNUM            ;INCREMENT COUNTER NUMBER
4406 026304  000756                          BR      1$              ;SEE IF DONE
4407
4408                                    ;-END 1$
4409
```

```
4411                                            ;GET MAXIMUM RADIX
4412
4413 026306  012737  000400  002304  MAXRDX: MOV     #BIT8,TEMP      ;INITIALIZE LOOP VARIANT
4414
4415                                            ;TRY EACH SWITCH SETTING UNTIL ONE AGREES STARTING AT 8
4416
4417                                    ;DO 3$ WHILE (TEMP IS NOT MAXIMUM OR NO SWITCHES SET )
4418 026314  006237  002304          3$:     ASR     TEMP            ;GET BIT TO TEST
4419 026320  001463                          BEQ     5$              ;ESCAPE IF NO SWITCHES SET
4420
4421 026322  012737  177777  003146          MOV     #-1,CNUM        ;INITIALIZE COUNTER VARIANT
4422
4423                                            ;SEE IF ANY COUNTERS ARE SET TO SWITCH SETTING TEMP
4424
4425                                    ;-DO 4$ WHILE   ( COUNTERS NOT ALL DONE )
4426 026330  005237  003146          4$:     INC     CNUM            ;ADVANCE COUNTER NUMBER
4427 026334  022737  000004  003146          CMP     #4,CNUM         ;IF DONE ALL COUNTERS
4428 026342  001764                          BEQ     3$              ;-THEN SEE IF DONE SWITCHES
4429
4430 026344  013705  003146                  MOV     CNUM,R5         ;CONSTRUCT THE OFFSET
4431 026350  006305                          ASL     R5
4432 026352  033765  002304  024046          BIT     TEMP,DSWCH(R5)  ;IF A SWITCH IS DETECTED NOT SET
4433 026360  001763                          BEQ     4$              ;-THEN TRY THE NEXT COUNTER
4434
4435                                    ;--END 4$
4436                                    ;-END 3$
4437
4438 026362  006337  002304                  ASL     TEMP            ;GET MAXIMUM RADIX
4439
4440 026366  006337  002304                  ASL     TEMP            ;GET TWICE MAXIMUM
4441 026372  005337  002304                  DEC     TEMP            ;COMPUTE NUMBER TO MAX BIT SETS
4442
4443                                            ;CLEAR ALL THE COUNTERS BEFORE MAXING THE COUNTS
4444
4445 026376  005037  003150                  CLR     NUM             ;INITIALIZE LOOP COUNT
4446
4447                                    ;DO 50$ WHILE ( NOT ALL COUNTERS HAVE BEEN CLEARED )
4448
4449 026402  022737  000004  003146  50$:     CMP     #4,CNUM         ;IF ALL COUNTERS CLEARED
4450 026410  001425                          BEQ     52$             ;-THEN FILL COUNTS TO MAXIMUM
4451
4452 026412  113777  003146  175412          MOVB    CNUM,@SR5016    ;LOAD COUNTER NUMBER TO CLEAR
4453 026420  152777  000004  175404          BISB    #4,@SR5016      ;PREPARE TO CLEAR COUNTER
4454 026426  105777  175402                  TSTB    @DR5016         ;CLEAR THE COUNTER
4455 026432  005037  001140                  CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
4456 026436  117737  175372  001142          MOVB    @DR5016,$BDDAT  ;READ COUNTER
4457 026444  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IF DATA IS AS EXPECTED
4458 026452  001401                          BEQ     51$             ;-THEN SKIP ERROR
4459
4460 026454  104112                          EMT     112
4461
4462 026456  005237  003146          51$:    INC     CNUM            ;PREPARE TO DO NEXT COUNTER
4463 026462  000747                          BR      50$             ;DO NEXT COUNTER
4464
4465                                    ;-END 50$
4466
4467                                            ;TOGGLE THE COUNTERS TO FULL COUNT TWICE TO GET OVF AND FULL COUNT
```

```
4468
4469 026464  005037  003150          52$:    CLR     NUM             ;INITIALIZE LOOP COUNT
4470
4471                                 ;DO 5$ WHILE (NUM IS LESS THAN TEMP )
4472 026470  023737  003150  002304  5$:     CMP     NUM,TEMP        ;IF DONE
4473 026476  001414                         BEQ     6$              ;-THEN LOAD SP5016
4474
4475 026500  142777  000010  153554         BICB    #TBIT,@CSR      ;ASSURE TBIT IS CLEAR
4476 026506  152777  000010  153546         BISB    #TBIT,@CSR      ;CLOCK THE COUNTERS
4477 026514  142777  000010  153540         BICB    #TBIT,@CSR      ;DON'T LEAVE BITS HANGING
4478
4479 026522  005237  003150                 INC     NUM             ;TALLY THE TOGGLES
4480 026526  000760                         BR      5$              ;CHECK IF DONE
4481
4482                                 ;-END 5$
4483
4484 026530  112777  000007  175274  6$:     MOVB    #7,@SR5016      ;COMPLETE THE FULL HOUSE
4485
4486 026536  004737  032024                 JSR     PC,KLEER        ;CLEAR THE WORLD
4487 026542  152777  000020  153512         BISB    #DBIT,@CSR      ;SHUT OUT WORLD
4488
4489 026550  005037  001140                 CLR     $GDDAT          ;INITIALIZE EXPECTED DATA
4490
4491                                 ;CHECK THAT STATUS REGISTER IS CLEAR
4492
4493 026554  117737  175252  001142         MOVB    @SR5016,$BDDAT  ;READ STATUS REGISTER
4494 026562  123737  001140  001142         CMPB    $GDDAT,$BDDAT   ;IF STATUS REGISTER IS CLEAR
4495 026570  001401                         BEQ     ALLCLX          ;-THEN CHECK COUNTERS ARE CLEAR
4496
4497 026572  104075                         EMT     75
```

```
4499                                            ;CHECK THAT ALL COUNTERS ARE CLEAR
4500
4501 026574  005037  003146          ALLCLX: CLR     CNUM            ;INITIALIZE COUNTER NUMBER
4502
4503                                            ;DO 2$ WHILE (3 IS GREATER THAN OR EQUAL TO COUNTER# ''CNUM'')
4504
4505 026600  022737  000003  003146  2$:     CMP     #3,CNUM         ;IF COUNTER # IS NOT LESS THAN 3
4506 026606  002416                          BLT     4$              ;-THEN CLEAN UP AND EXIT
4507 026610  113777  003146  175214          MOVB    CNUM,@SR5016    ;-ELSE LOAD STATUS REG WITH COUNTER #
4508 026616  117737  175212  001142          MOVB    @DR5016,$BDDAT  ;READ DATA REG
4509 026624  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;IF COUNTER 'CNUM' IS CLEAR
4510 026632  001401                          BEQ     3$              ;-THEN SKIP ERROR
4511
4512 026634  104116                          EMT     116
4513
4514 026636  005237  003146          3$:     INC     CNUM            ;INCREMENT COUNTER NUMBER
4515 026642  000756                          BR      2$              ;SEE IF DONE
4516
4517                                    ;-END 2$
4518
4519 026644  104412                  4$:     RESREG                  ;RESTORE THE REGISTERS
4520 026646  000207                          RTS     PC
```

SEQ 0118

```
   4522 026650  012537  003154          SDELAY: MOV     (R5)+,XLOOP
   4523 026654  005037  002222                  CLR     YLOOP
   4524 026660  005237  002222          1$:     INC     YLOOP
   4525 026664  023737  003154  002222          CMP     XLOOP,YLOOP
   4526 026672  001372                          BNE     1$
   4527 026674  000205                          RTS     R5
```

```
                                        .SBTTL  MONITOR TO SELECT DAC TESTS
026676                                  DCAMON:
026676  012777  040722  153450                MOV     #KBINT,@KBVFC
026704  152777  000100  152246                BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
026712  004737  027210                        JSR     PC,GTADRS               ;SET UP ADDRESSES FOR DAC
026716  004737  027304                        JSR     PC,TSTADR               ;CHECK PRESENCE OF ADDRESSES
026722  000207                              RTS     PC
026724  004737  027252                        JSR     PC,DDBIT                ;GO DO D-BIT TEST
026730  000207                              RTS     PC                      ;RETURN TO AUTO MONITOR
                                        
026732  012737  026732  003160  DACSTR:  MOV     #DACSTR,RETURN          ;SET RETURN FOR CONTROL A
026740  104401  052744                        TYPE    ,MASS13                 ;ASK ADDRESS
026744  104410                              RDOCT
026746  012637  002244                        MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
026752  023737  002244  002276                CMP     TADDR,BASE
026760  103764                              BLO     DACSTR
026762  004737  027210          DACMON:  JSR     PC,GTADRS               ;SET UP ADDRESSES
026766  004737  027304                        JSR     PC,TSTADR               ;CHECK RESPONSE OF ADDRESSES
026772  000207                  DCOUT:   RTS     PC                      ;EXIT TO MONITOR ON ERROR
026774  004737  031656                        JSR     PC,DAGTST               ;CHECK THAT GENERIC CODE OF
027000  104401  057272                        TYPE    ,DANGC                  ;THE DAC IS 261 IF IT IS GO TEST
027004  000207                              RTS     PC                      ;ELSE RETURN TO MAIN MON$PASS
027006  104401  057426          DCMN:    TYPE    ,DANSEL                 ;SELECT TEST A, T,  D,
027012  004737  032024                        JSR     PC,KLEER
027016  104406                              RDCHR                           ;-?-FOR MEANING OF SELECTION
027020  012637  002432                        MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
027024  104401  002432                        TYPE    ,ANSW
027030  122737  000110  002432              CMPB    # 'H,ANSW               ;IF QUESTION THE ELIGHTEN
027036  001003                              BNE     1$                      ;USER BY TYPING TEXT  OF TEST
027040  104401  057534                        TYPE    ,TEXT                   ;SELECTION DESCRIPTORS
027044  000760                              PR      DCMN                    ;NOW SELEC TEST
027046  122737  000101  002432  1$:      CMPB    # 'A,ANSW                ;SELECT DAC CALIBRATION TEST
027054  001015                              BNE     2$                      ;IF NOT THIS TEST THEN CHECK NEXT
027056  104401  047611                        TYPE    ,M30                    ;DISCONNECT CUSTOMER WIRES
027062  104406                  5$:      RDCHR
027064  012637  002432                        MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
027070  122737  000015  002432              CMPB    #15,ANSW
027076  001371                              BNE     5$
027100  004737  027670                        JSR     PC,DACTST               ;ELSE DO THIS TEST
027104  000137  027006                        JMP     DCMN                    ;
027110  122737  000124  002432  2$:      CMPB    #'T,ANSW                ;SELECT TBIT (STATUS BITS) TEST
027116  001016                              BNE     3$                      ;IF NOT THIS TEST THEN CHECK NEXT
027120  104401  047611                        TYPE    ,M30
027124  104406                  6$:      RDCHR
027126  012637  002432                        MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
027132  122737  000015  002432              CMPB    #15,ANSW
027140  001371                              BNE     6$
027142  004737  042224          14$:     JSR     PC,SWLOOP
027146  030430                              INTCOM                          ;DO INTCOM TEST
027150  000137  027006                        JMP     DCMN
027154  122737  000104  002432  3$:      CMPB    #'D,ANSW                ;SELECT DBIT TEST
027162  001004                              BNE     4$                      ;IF NOT THIS TEST THEN CHECK NEXT
027164  004737  027252          9$:      JSR     PC,DDBIT                        ;DO DDBIT TEST
027170  000137  027006                        JMP     DCMN
027174  004737  040646          4$:      JSR     PC,CNTRC                ;CHECK FOR EXIT BY CONTROL "C"
027200  104401  047136                        TYPE    ,M20                    ;TYPE ? IF NO TST SELECTION FOUND
027204  000137  027006                        JMP     DCMN                    ;GO TO START OF DAC MONITOR
```

```
4584
4585
4586
4587 027210  005000                    GTADRS: CLR     R0                      ;CLEAR ADDRESS POINTER
4588 027212  005001                            CLR     R1                      ;CLEAR ADDRESS COUNTER
4589 027214  013746  002244                    MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
4590 027220  013760  002244  002402    1$:     MOV     TADDR,LDATA0(R0)        ;WRITE ADDRESSES
4591 027226  105237  002244                    INCB    TADDR                   ;UPDATE TO NEXT REGISTER
4592 027232  005720                            TST     (R0)+                   ;UPDATE TO NEXT LOCATION
4593 027234  005201                            INC     R1                      ;UPDATE REGISTER COUNTER
4594 027236  022701  000010                    CMP     #10,R1                  ;CHECK FOR EIGHT REGISTERS
4595 027242  001366                            BNE     1$                      ;IF NOT LAST GET NEXT
4596 027244  012637  002244                    MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
4597 027250  000207                            RTS     PC                      ;AND RETURN
4598
4599                                    .SBTTL  D-BIT   TEST
4600
4601 027252  004737  032024            DDBIT:  JSR     PC,KLEER
4602 027256  152777  000020  152776            BISB    #DBIT,@CSR              ;SET THE D-BIT
4603 027264  004737  027376                    JSR     PC,TSTRGS               ;GO CHECK REGISTERS
4604 027270  142777  000020  152764            BICB    #DBIT,@CSR              ;CLEAR D-BIT WHEN DONE
4605 027276  004737  032024                    JSR     PC,KLEER                ;AND CLEAR THE WORLD
4606 027302  000207                            RTS     PC                      ;AND RETURN
4607
```

```
4609                                    .SBTTL  REGISTER VERIFICATION
4610 027304  013703  000004             TSTADR: MOV     @ERRVEC,R3              ;SAVE PRESENT LOC 4
4611 027310  012737  0273 2  000004             MOV     #2$,ERRVEC              ;SET UP TIME OUT VECTOR
4612 027316  004737  032024                     JSR     PC,KLEER                ;INIT THE SYSTEM
4613 027322  005000                             CLR     R0                      ;CLR ADDRESS LOC
4614 027324  005001                             CLR     R1                      ;CLR ADDRESS COUNTER
4615 027326  013700  002244                     MOV     TADDR, R0               ;GET FIRST REG ADDRESS
4616 027332  105720                     1$:     TSTB    (R0)+                   ;IS IT THERE AND CLEAR
4617 027334  001407                             BEQ     3$                      ;IF IT IS GET NEXT ADDRESS
4618 027336  104027                             EMT     27
4619 027340  000405                             BR      3$                      ;AND THEN GET NEXT ADDRESS
4620 027342                             2$:
     027342  104026                             EMT     26
4621 027344  004737  040646                     JSR     PC,CNTRC                ;CHECK FOR CONTROL "C" EXIT
4622 027350  022626                             CMP     (SP)+,(SP)+             ;ADJUST THE STACK AFTER TIME OUT
4623 027352  000406                             BR      4$                      ;RETURN MAIN MONITOR
4624 027354  005201                     3$:     INC     R1                      ;UPDATE ADDRESS COUNTER
4625 027356  022701  00001C                     CMP     #10,R1                  ;CHECK FOR LAST REG
4626 027362  001363                             BNE     1$                      ;IF IT IS THEN
4627 027364  062716  000002                     ADD     #2,(SP)                 ;CONTINUE WITH TEST
4628 027370  010337  000004             4$:     MOV     R3,ERRVEC               ;RESTORE LOC4
4629 027374  000207                             RTS     PC                      ;RETURN
4630
4631 027376                             TSTRGS:
                                        ;;*************************************************************
     027376  000004                     TST25:  SCOPE
4632 027400  012737  000025  001206             MOV     #25,$TESTN
4633 027406  005037  003034                     CLR     @ROTPAT                 ;CLEAR THE PATTRN SPOT
4634 027412  005000                             CLR     R0                      ;CLEAR THE INDEX REGISTER
4635 027414  012737  000007  002422             MOV     #7,KOUNT                ;DO SEVEN FOR NOW
4636 027422  004737  031624             1$:     JSR     PC,ROTDAT               ;NOW GO ROTATE A BIT
4637 027426  113770  003034  002402     2$:     MOVB    @ROTPAT,@LDATA0(R0)     ;NOW GO WRITE IT
4638 027434  004737  027650                     JSR     PC,UPREG                ;NOW GET THE NEXT REGISTER
4639 027440  000772                             BR      2$                      ;AND WRITE IN (4 IN ALL)
4640 027442  013737  003034  001140             MOV     @ROTPAT,$GDDAT          ;SAVE PATTERN FOR CHECKING
4641 027450  117037  002402  001142     3$:     MOVB    @LDATA0(R0),$BDDAT      ;READ THE RSGISTER
4642 027456  016037  002402  002304             MOV     LDATA0(R0),TEMP         ;AND STORE ITS ADDRESS
4643 027464  023737  001140  001142             CMP     $GDDAT,$BDDAT           ;SEE IF IT IS GOOD
4644 027472  001401                             BEQ     4$                      ;IF GOOD CONTINUE
4645 027474  104030                             EMT     30
4646 027476  004737  040646             4$:     JSR     PC,CNTRC                ;CHECK FOR CONTROL "C" EXIT
4647 027502  004737  027650                     JSR     PC,UPREG                ;READ NEXT REGISTER
4648 027506  000760                             BP      3$                      ;UNTIL 4 DONE
4649 027510  005337  002422                     DEC     KOUNT                   ;AND UNTIL7 ROTATES
4650 027514  100342                             BPL     1$                      ;NOW GO BACK AND DO IT
4651 027516  005037  003034                     CLR     @ROTPAT                 ;CLEAR PATTERN SPOT
4652 027522  012737  000001  002422             MOV     #1,KOUNT                ;DO ONLY 2 THIS TIME
4653 027530  004737  031624             5$:     JSR     PC,ROTDAT               ;GO ROTATE A BIT NOW
4654 027534  113770  003034  002404     6$:     MOVB    @ROTPAT,@HDATA0(R0)     ;GO WRITE IT NOW
4655 027542  004737  027650                     JSR     PC,UPREG                ;GET THE NEXT REGISTER
4656 027546  000772                             BR      6$                      ;AND WRITE IN IT (4 IN ALL)
4657 027550  013737  003034  001140             MOV     @ROTPAT,$GDDAT          ;SAVE PATTERN FOR CHECKING
4658 027556  112770  000000  002402     7$:     MOVB    #0,@LDATA0(R0)          ;NOW  LET IT TRANSFER
4659 027564  117037  002404  001142             MOVB    @HDATA0(R0),$BDDAT      ;READ THE REGISTER
4660 027572  016037  002404  002304             MOV     HDATA0(R0),TEMP         ;AND STORE ITS ADDRESS
4661 027600  123737  001140  001142             CMPB    $GDDAT,$BDDAT           ;SEE IF IT IS GOOD
4662 027606  001401                             BEQ     8$                      ;IF GOOD CONTINUE
```

```
4663 027610  104030                    EMT     30
4664 027612  004737  040646    8$:     JSR     PC,CNTRC        ;CHECK FOR CONTROL"C" EXIT
4665 027616  004737  027650            JSR     PC,UPREG        ;READ NEXT REGISTER
4666 027622  000755                    BR      7$              ;UNTIL 4 ARE DONE
4667 027624  005337  002422            DEC     KOUNT           ;AND FOR 2 ROTATES
4668 027630  100337                    BPL     5$              ;GO BACK AND DO IT
4669 027632  000004                    SCOPE
4670 027634  005737  002312            TST     AFLAG
4671 027640  001002                    BNE     9$
4672 027642  104401  053023            TYPE    ,MASS15         ;END OF TEST
4673 027646  000207            9$:     RTS     PC
4674
4675
4676
4677
4678 027650  022020            UPREG:  CMP     (R0)+,(R0)+             ;UPDATE INDEX
4679 027652  022700  000020            CMP     #20,R0          ;UNTIL 4 REGISTERS
4680 027656  001003                    BNE     1$              ;ARE DONE
4681 027660  005000                    CLR     R0              ;CLR THE INDEX REG
4682 027662  062716  000002            ADD     #2,(SP)         ;UPDATE THE STACK
4683 027666  000207            1$:     RTS     PC              ;CONTINUE TEST
4684
4685
4686
```

```
4688                                    .SBTTL  DAC CALIBRATION TEST
4689
4690
4691 027670  004737  032024          DACTST: JSR     PC,KLEER
4692
4693 027674  004737  031722          DATST:  JSR     PC,DECCHN               ;GET CHANEL # AND DECODE IT
4694 027700  104401  057642                  TYPE    ,CTXTV                  ;DO REFER VOLTAGE CAL Y-N
4695 027704  104406                          RDCHR                           ;GET Y-N ANSW
4696 027706  012637  002432                  MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
4697 027712  104401  002432                  TYPE    ,ANSW                   ;ECHO IT
4698 027716  122737  000116  002432          CMPB    #'N,ANSW                ;IF IT IS A NO
4699 027724  001413                          BEQ     5$                      ;GO TO CHAN AND
4700 027726  122737  000131  002432          CMPB    #'Y,ANSW                ;IS IT A YES?
4701 027734  001403                          BEQ     20$
4702 027736  104401  047136                  TYPE    ,M20                    ;?
4703 027742  000754                          BR      DATST
4704 027744  004737  040646          20$:    JSR     PC,CNTRC
4705 027750  004737  027766                  JSR     PC,DCTST                ;PASS REFER VOLT CAL TST
4706 027754  004737  040646          5$:     JSR     PC,CNTRC
4707 027760  104401  057076                  TYPE    ,M81
4708 027764  000411                          BR      DTST
4709
4710 027766  104401  060036          DCTST:  TYPE    ,CM1                    ;ADJ R135 @ PIN 6 OF E82 FOR -0.24
4711 027772  004737  041026                  JSR     PC,CRTST                ;USING SWITCHES AT E79 IF NEEDED
4712                                                                          ;WHEN DONE HIT <CR>
4713 027776  104401  060105                  TYPE    ,CM2                    ;ADJ R134 @ PIN 60 OF E81 FOR 5.12
4714 030002  004737  041026                  JSR     PC,CRTST                ;WHEN DONE HIT <CR>
4715 030006  000207                          RTS     PC
4716 030010  013700  002374          DTST:   MOV     XCHAN,R0                ;GET REG INDEX TO CHAN
4717 030014  104401  060155                  TYPE    ,CM3                    ;ADJ VOLTAGE OFFSET ON CHAN 1 TO 0.000
4718 030020  004737  041026          21$:    JSR     PC,CRTST                ;WAIT FOR <LR>
4719 030024  112770  000003  002404  1$:     MOVB    #3,@HDATA0 (R0)         ;LOAD ALL ONES INTO
4720 030032  112770  000377  002402          MOVB    #377,@LDATA0 (R0)       ;CHAN SELECTED
4721 030040  104401  060225                  TYPE    ,CM4                    ;CHECK VOLTAGE LEVEL ON CHANNEL SELECTED
4722 030044  004737  041026          22$:    JSR     PC,CRTST                ;FOR 10.23V + OR - 10 MIN
4723 030050  104401  057135                  TYPE    ,M82
4724 030054  104406                  41$:    RDCHR
4725 030056  012637  002432                  MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
4726 030062  104401  002432                  TYPE    ,ANSW
4727 030066  004737  040646                  JSR     PC,CNTRC
4728 030072  122737  000116  002432          CMPB    #'N,ANSW
4729 030100  001002                          BNE     40$
4730 030102  000137  030416                  JMP     7$
4731 030106  122737  000131  002432  40$:    CMPB    #'Y,ANSW
4732 030114  001357                          BNE     41$
4733 030116  104401  061024                  TYPE    ,CM12                   ;ASK IF 4 TO 20 MA RANGE WANTED
4734 030122  104406                  23$:    RDCHR                           ;READ ANSWER
4735 030124  012637  002432                  MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
4736 030130  104401  002432                  TYPE    ,ANSW                   ;ECHO ANSWER
4737 030134  122737  000131  002432          CMPB    #'Y,ANSW                ;IF YES GO TO 4 TO 20 MA TEST
4738 030142  001455                          BEQ     5$                      ;ELSE DO 0 TO 20 MA TEST
4739 030144  122737  000116  002432          CMPB    #'N,ANSW
4740 030152  001403                          BEQ     4$
4741 030154  004737  040646                  JSR     PC,CNTRC
4742 030160  000760                          BR      23$
4743 030162  112770  000003  002404  4$:     MOVB    #3,@HDATA0(R0)          ;RELOAD UPPER BYTE OF CHANEL
4744 030170  112770  000377  002402          MOVB    #377,@LDATA0(R0)        ;SELECTED AND LET IT GO (CONV)
```

```
 4745 030176  104401  060452                    TYPE    ,CM9                    ;ADJ CURRENT GAIN TO 10.000V
 4746 030202  004737  041026          25$:      JSR     PC,CRTST                ;OR 20MA
 4747 030206  112770  000000  002404            MOVB    #0,@HDATA0(R0)          ;CLEAR UPPER BYTE OF CHANSELECTED
 4748 030214  112770  000001  002402            MOVB    #1,@LDATA0(R0)          ;SET 1 TO LOWER BYTE OF
 4749                                                                            ;CHANEL SELECTED
 4750 030222  104401  060300                    TYPE    ,CM7                    ;ADJ CURRENT OFFSET TO 9.8 MV
 4751 030226  004737  041026          26$:      JSR     PC,CRTST                ;OR 19.5 MICRO-AMPS
 4752
 4753 030232  104401  060760                    TYPE    ,CM11                   ;ASK TO REVERIFY
 4754 030236  104406                  27$:      RDCHR
 4755 030240  012637  002432                    MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
 4756 030244  104401  002432                    TYPE    ,ANSW                   ;ECHO ANSWER
 4757 030250  004737  040646                    JSR     PC,CNTRC
 4758 030254  122737  000116  002432            CMPB    #'N,ANSW
 4759 030262  001737                            BEQ     4$
 4760 030264  122737  000131  002432            CMPB    #'Y,ANSW
 4761 030272  001451                            BEQ     7$
 4762 030274  000760                            BR      27$                     ;EXIT AFTER ALINEMENT
 4763 030276  112770  000003  002404  5$:       MOVB    #3,@HDATA0(R0)          ;LOAD ALL ONES
 4764 030304  112770  000377  002402            MOVB    #377,@LDATA0(R0)        ;INTO SELECTED CHANEL
 4765 030312  104401  060452                    TYPE    ,CM9                    ;ADJ CURRENT GAIN TO
 4766 030316  104406                  31$:      RDCHR                           ;10.000 V OR 20 MA
 4767 030320  012637  002432                    MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
 4768 030324  004737  040646                    JSR     PC,CNTRC
 4769 030330  122737  000015  002432            CMPB    #15,ANSW
 4770 030336  001367                            BNE     31$
 4771 030340  004737  032024                    JSR     PC,KLEER                ;CLEAR ALL REGS TO ZERO
 4772 030344  104401  060614                    TYPE    ,CM10                   ;ADJ CURRENT OFFSET
 4773 030350  004737  041026          32$:      JSR     PC,CRTST                ;FOR 2.00V OR 4.00MA
 4774 030354  104401  060760                    TYPE    ,CM11
 4775 030360  104406                  33$:      RDCHR
 4776 030362  012637  002432                    MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
 4777 030366  104401  002432                    TYPE    ,ANSW                   ;ECHO ANSWER
 4778 030372  004737  040646                    JSR     PC,CNTRC
 4779 030376  122737  000116  002432            CMPB    #'N,ANSW
 4780 030404  001734                            BEQ     5$
 4781 030406  122737  000131  002432            CMPB    #'Y,ANSW
 4782 030414  001361                            BNE     33$
 4783 030416  104401  057215          7$:       TYPE    ,M83                    ;END OF CALIBRATION
 4784 030422  004737  032024                    JSR     PC,KLEER
 4785 030426  000207                            RTS     PC
 4786
 4787                                  .SBTTL  INTERNAL COMPARATOR CHECK
 4788
 4789 030430                          INTCOM:
                                      ;;*****************************************************************
      030430  000004                  TST26:  SCOPE
 4790 030432  012737  000026  001206            MOV     #26,$TESTN
 4791 030440  152777  000010  151614            BISB    #TBIT,@CSR              ;SET THE TBIT INTO THE CSR
 4792 030446  112777  000000  151726            MOVB    #0,@LDATA0              ;START CH 0 CONVERSIONS
 4793 030454  112777  000100  151724            MOVB    #100,@LDATA1            ;START CH 1 CONVERSIONS
 4794 030462  112777  000200  151722            MOVB    #200,@LDATA2            ;START CH 2 CONVERSIONS
 4795 030470  112777  000300  151720            MOVB    #300,@LDATA3            ;START CH 3 CONVERSION
 4796 030476  004737  032102                    JSR     PC,DALLY                ;WAIT FOR ALL CHANELS TO FINISH
 4797
 4798 030502  013700  002404                    MOV     HDATA0,R0               ;GET FIRST REG TO CHECK
 4799 030506  111001                  4$:       MOVB    (R0),R1                 ;GET HIGH DATA BYTE
```

```
4800 030510  042701  177703                BIC     #177703,R1              ;CLEAR UNWANTED BITS
4801 030514  022701  000074                CMP     #74,R1                  ;IF ALL BIT ARE SET
4802 030520  001417                        BEQ       5$                    ;CONTINUE WITH TEST
4803
4804 030522  010137  002304                MOV     R1,TEMP
4805 030526  006237  002304                ASR     TEMP
4806 030532  006237  002304                ASR     TEMP
4807 030536  012737  000017  003042        MOV     #17,TEMP3
4808 030544  104031                        EMT     31
4809 030546  004737  032024                JSR     PC,KLEER
4810 030552  000207                        RTS     PC
4811 030554  004737  040646                JSR     PC,CNTRC                ;CHECK FOR CONTROL "C" EXIT
4812 030560  062700  000002          5$:   ADD     #2,R0                   ;UPDATE TO NEXT HIGH DATA REG
4813 030564  023700  002420                CMP     HDATA3,R0               ;AND CHECK TO SEE IF ALL STATUS
4814 030570  003746                        BLE     4$                      ;BITS ARE SET WHEN DONE
4815 030572  005000                        CLR     R0                      ;CLEAR R0 AND SYSTEM
4816 030574  004737  032024                JSR     PC,KLEER                ;RIGHT HERE
4817 030600  152777  000010  151454        BISB    #TBIT,@CSR              ;RELOAD THE TBIT
4818 030606  112777  000000  151566        MOVB    #0,@LDATA0              ;CHECK ALL STATUS
4819 030614  112777  000004  151564        MOVB    #4,@LDATA1              ;BITS AT THE 3-LSB
4820 030622  112777  000010  151562        MOVB    #10,@LDATA2             ;DIFFERANCE MARGIN
4821 030630  112777  000014  151560        MOVB    #14,@LDATA3
4822 030636  004737  032102                JSR     PC,DALLY                ;WAIT FOR IT TO HAPPEN
4823 030642  117702  151536                MOVB    @HDATA0,R2              ;GET THE STATUS INFO
4824 030646  042702  177703                BIC     #177703,R2              ;CLEAR UNWANTED BITS
4825 030652  022702  000074                CMP     #74,    R2              ;ALL FOUR STATUS BITS
4826 030656  001415                        BEQ     6$                      ;SHOULD BE SET ELSE ERROR
4827 030660  010237  002304                MOV     R2,TEMP
4828 030664  006237  002304                ASR     TEMP
4829 030670  006237  002304                ASR     TEMP
4830 030674  012737  000017  003042        MOV     #17,TEMP3
4831 030702  104031                        EMT     31
4832 030704  004737  032024                JSR     PC,KLEER
4833 030710  000207                        RTS     PC
4834 030712  004737  040646          6$:   JSR     PC,CNTRC                ;CHECK FOR CONTROL "C" EXIT
4835 030716  004737  032024                JSR     PC,KLEER                ;CLEAR THE SYSTEM
4836 030722  152777  000010  151332        BISB    #TBIT,  @CSR            ;RELOAD THE TBIT
4837 030730  112777  000014  151444        MOVB    #14,@LDATA0             ;CHECK ALL STATUS BIT AT
4838 030736  112777  000010  151442        MOVB    #10,@LDATA1             ;THE DESCENDING 3-LSB
4839 030744  112777  000004  151440        MOVB    #4,@LDATA2              ;DIFFERANCE MARGIN
4840 030752  112777  000000  151436        MOVB    #0,@LDATA3
4841 030760  004737  032102                JSR     PC,DALLY                ;WAIT FOR IT TO HAPPEN
4842 030764  117702  151414                MOVB    @HDATA0,R2              ;GET THE STATUS INFO
4843 030770  042702  177703                BIC     #177703,R2              ;CLEAR UNWANTED BITS
4844 030774  005702                        TST     R2                      ;ALL FOUR STATUS BITS
4845 030776  001414                        BEQ     7$                      ;SHOULD BE ZERO ELSE ERROR
4846 031000  010237  002304                MOV     R2,TEMP
4847 031004  006237  002304                ASR     TEMP
4848 031010  006237  002304                ASR     TEMP
4849 031014  005037  003042                CLR     TEMP3
4850 031020  104032                        EMT     32
4851 031022  004737  032024                JSR     PC,KLEER
4852 031026  000207                        RTS     PC
4853 031030  004737  040646          7$:   JSR     PC,CNTRC                ;CHECK FOR CONTROL "C" EXIT
4854
4855 031034  012737  000000  002424        MOV     #0,ACOUNT       ;SET UP OCT0 INPUT TO
4856 031042  012737  000020  002426        MOV     #20,BCOUNT      ;CHANELS 0 TO 3 I.E. 0,20,0,20
```

```
4857 031050  012737  000064  002360          MOV     #64,CH0         ;GET GOOD STATUS OUTPUT
4858 031056  012737  000040  002362          MOV     #40,CH1         ;GET ERROR
4859 031064  012737  000070  002364          MOV     #70,CH2         ;STATUS OUTPUTS
4860 031072  012737  000054  002366          MOV     #54,CH3         ;TOTAL OF FOUR
4861 031100  012737  000004  002370          MOV     #4,CH4          ;TO BE LOOKED AT
4862 031106  152777  000010  151146          BISB    #IBIT,@CSR      ;SET THE IBIT IN THE CSR
4863 031114  004737  031366          8$:     JSR     PC,INTLD        ;LOAD INPUT REGS
4864 031120  117737  151260  002304          MOVB    @HDATA0,TEMP    ;GET STATUS BITS
4865 031126  042737  177703  002304          BIC     #177703,TEMP    ;CLEAR UNWANTED BITS
4866 031134  013737  002304  003042          MOV     TEMP,TEMP3
4867 031142  006237  003042                  ASR     TEMP3
4868 031146  006237  003042                  ASR     TEMP3
4869 031152  004737  031454                  JSR     PC,CKICT        ;GO CHECK STATUS BITS
4870 031156  104033                          EMT     33
4871 031160  004737  040646                  JSR     PC,CNTRC        ;CHECK OFR CONTROL "C" EXIT
4872 031164  004737  031570                  JSR     PC,GOUP         ;GET NEXT INPUT VALUES
4873 031170  022737  001750  002424          CMP     #1750,ACOUNT    ;THIS IS THE LAST INPUT
4874 031176  003746                          BLE     8$              ;IF NOT GET NEXT INPUT
4875
4876 031200  012737  001770  002424          MOV     #1770,ACOUNT    ;SET UP UPPER LIMIT TO DO
4877 031206  012737  001760  002426          MOV     #1760,BCOUNT    ;STATUS BIT COMPLIMENT DOWN
4878 031214  012737  000010  002360          MOV     #10,CH0         ;GET GOOD STATUS OUTPUT
4879 031222  012737  000034  002362          MOV     #34,CH1         ;GET STATUS
4880 031230  012737  000004  002364          MOV     #4,CH2          ;ERROR OUTPUTS
4881 031236  012737  000020  002366          MOV     #20,CH3         ;TOTAL OF FUUR
4882 031244  012737  000070  002370          MOV     #70,CH4         ;TO BE LOOKED AT
4883 031252  004737  031366          9$:     JSR     PC,INTLD        ;LOAD INPUT REGS
4884 031256  117737  151122  002304          MOVB    @HDATA0,TEMP    ;GET STATUS BITS
4885 031264  042737  177703  002304          BIC     #177703,TEMP    ;CLEAR UNWANTED BITS
4886 031272  013737  002304  003042          MOV     TEMP,TEMP3
4887 031300  006237  003042                  ASR     TEMP3
4888 031304  006237  003042                  ASR     TEMP3
4889 031310  004737  031454                  JSR     PC,CKICT        ;GO VERIFY DATA
4890 031314  104033                          EMT     33
4891 031316  004737  040646                  JSR     PC,CNTRC        ;CHECK CONTROL "C" EXIT
4892 031322  004737  031606                  JSR     PC,GODN         ;GO COUNT DOWN
4893 031326  005737  002426                  TST     BCOUNT          ;CHECK FOR LAST INPUT
4894 031332  001347                          BNE     9$              ;IF NOT LAST GET NEXT INPUT
4895 031334  000004                          SCOPE
4896 031336  142777  000010  150716          BICB    #IBIT,@CSR      ,CLEAR THE IBIT AND
4897 031344  004737  032024                  JSR     PC,KLEER                ;CLEAR THE EMPIRE
4898 031350  032777  020000  147576          BIT     #BIT13,@SWR             ;CHECK INHIBIT PRINTOUT
4899 031356  001002                          BNE     10$
4900 031360  104401  053023                  TYPE    ,MASS15                 ;TYPE FINISHED
4901 031364  000207                  10$:    RTS     PC
4902
4903
4904 031366  113777  002425  151010  INTLD:  MOVB    ACOUNT+1,@HDATA0        ;LOAD UPPER CH0 REG
4905 031374  113777  002424  151000          MOVB    ACOUNT,@LDATA0  ;LOAD LOWER CH0 REG
4906 031402  113777  002427  151000          MOVB    BCOUNT+1,@HDATA1        ;LOAD UPPER CH1 REG
4907 031410  113777  002426  150770          MOVB    BCOUNT,@LDATA1  ;LOAD LOWER CH1 REG
4908 031416  113777  002425  150770          MOVB    ACOUNT+1,@HDATA2        ;LOAD UPPER CH2 REG
4909 031424  113777  002424  150760          MOVB    ACOUNT,@LDATA2  ;LOAD LOWER CH2 REG
4910 031432  113777  002427  150760          MOVB    BCOUNT+1,@HDATA3        ;LOAD UPPER CH3 REG
4911 031440  113777  002426  150750          MOVB    BCOUNT,@LDATA3  ;LOAD LOWER CH3 REG
4912 031446  004737  032102                  JSR     PC,DALLY        ;WAIT FOR CONVERSION
4913 031452  000207                          RTS     PC                      ;RETURN
```

```
4914
4915 031454  023737  002360  002304  CKICT:  CMP     CH0,TEMP                ;CHECK GOOD DATA
4916 031462  001003                          BNE     1$              ;IF BAD CHECK WHIC CHANEL
4917 031464  062716  000002                  ADD     #2,(SP)         ;ELSE BY PASS ERROR
4918 031470  000207                          RTS     PC                      ;RETURN
4919 031472  005037  002356          1$:     CLR     CHNUM           ;GET CH0
4920 031476  023737  002362  002304          CMP     CH1,TEMP                ;SEE IF THIS CHANEL IS BAD
4921 031504  001430                          BEQ     2$              ;IF YES GO REPORT IT ELSE
4922 031506  012737  000001  002356          MOV     #1,CHNUM        ;GET CH1
4923 031514  023737  002364  002304          CMP     CH2,TEMP                ;SEE IF THIS CHANEL IS BAD
4924 031522  001421                          BEQ     2$              ;IF YES GO REPORT IT ELSE
4925 031524  012737  000002  002356          MOV     #2,CHNUM        ;GET CH2
4926 031532  023737  002366  002304          CMP     CH3,TEMP                ;SEE IF THIS CHANEL IS BAD
4927 031540  001412                          BEQ     2$              ;IF YES GO REPORT IT ELSE
4928 031542  012737  000003  002356          MOV     #3,CHNUM        ;GET CH3
4929 031550  023737  002370  002304          CMP     CH4,TEMP                ;SEE IF THIS CHANEL IS BAD
4930 031556  001403                          BEQ     2$              ;IF YES GO REPORT IT ELSE
4931 031560  104034                          EMT     34
4932 031562  062716  000002                  ADD     #2,(SP)         ;UPDATE STACK AND RETURN
4933 031566  000207                  2$:     RTS     PC              ;TO CONTINUE TEST
4934
4935 031570  062737  000010  002424  GOUP:   ADD     #10,ACOUNT      ;GO UP TO 1760 BY
4936 031576  062737  000010  002426          ADD     #10,BCOUNT      ;INCREMENT OF TEN
4937 031604  000207                          RTS     PC
4938
4939 031606  162737  000010  002424  GODN:   SUB     #10,ACOUNT      ;GO DOWN TO ZERO BY
4940 031614  162737  000010  002426          SUB     #10,BCOUNT      ;INVERSE INCREMENT OF TEN
4941 031622  000207                          RTS     PC              ;RETURN
4942
4943 031624  005237  003036          ROTDAT: INC     ROTFLG                  ;CLEAR THIS FLAG ON FIRST ENTRY
4944 031630  001001                          BNE     1$                      ;CHECK IF ZERO FIRST TIME
4945 031632  000261                          SEC                             ;FIRST TIME IN SET CARRY
4946 031634  106137  003034          1$:     ROLB    ROTPAT                  ;ROTATE PATTERN ONCE
4947 031640  005737  002422                  TST     KOUNT                   ;DID IT GET DONE 8 TIMES
4948 031644  001003                          BNE     2$                      ;IF WE HAVE SET ROTFLG
4949 031646  012737  177777  003036          MOV     #-1,ROTFLG              ;BACK TO MINUS ONE
4950 031654  000207                  2$:     RTS     PC                      ;NOW USE DATA
4951
4952 031656  004737  032024          DAGTST: JSR     PC,KLEER                ;CLEAR IT ALL
4953 031662  152777  000004  150372          BISB    #GBIT,  @CSR            ;SET THE GENERIC BIT IN THE CSR
4954 031670  117700  150506                  MOVB    @LDATA0,R0              ;READ THE GENERIC CODE IN DAC
4955 031674  042700  177400                  BIC     #177400,R0              ;CLEAR THE UPPER BYTE
4956 031700  022700  000261                  CMP     #261,R0                 ;CHECK THAT GENERIC CODE IS 261
4957 031704  001002                          BNE       1$                    ;IF NOT RETURN ERR MESS AND EXIT
4958 031706  012716  027006                  MOV     #DCMN,  (SP)            ;ELSE CONTINUE TO TEST
4959 031712  142777  000004  150342  1$:     BICB    #GBIT,  @CSR            ;BUT FIRST CLEAR GBIT
4960 031720  000207                          RTS     PC
4961
4962
4963 031722  104401  057466          DECCHN: TYPE    ,DACHN                  ;ASK FOR CHANEL NUMBER (0,1,2,3)
4964 031726  104410                          RDOCT                           ;GET CHANEL NUMBER
4965 031730  012637  002432                  MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
4966 031734  022737  000004  002432          CMP     #4,ANSW         ;CHECK FOR RIGHT CHAN
4967 031742  003003                          BGT     1$
4968 031744  104401  047136                  TYPE    ,M20                    ;?
4969 031750  000764                          BR      DECCHN
4970 031752  013737  002432  002372  1$:     MOV     ANSW,DCHAN              ;GET DAC CHANEL SELECTED
```

```
4971 031760  000241                          CLC                         ;NO END AROUND CARRY WANTED
4972 031762  006337  002432                  ASL     ANSW                ;CONVERT TO VCHAN SELECTION
4973 031766  013737  002432  002374          MOV     ANSW,XCHAN          ;GET ANSW TO INDEX
4974 031774  006337  002374                  ASL     XCHAN               ;MULTIPLY BY 2 AGAIN
4975 032000  013737  002432  002376          MOV     ANSW,VCHAN          ;STORE SELECTED VCHAN
4976 032006  062737  000001  002432          ADD     #1,ANSW             ;CONVERT TO ICHAN SELECTED
4977 032014  013737  002432  002400          MOV     ANSW,ICHAN          ;STORE SELECTE ICHAN
4978 032022  000207                          RTS     PC                          ;RETURN
4979
4980 032024                          KLFER:
     032024  152777  000002  150230          BISB    #CBIT,@CSR          ;SET C BIT
     032032  005037  002222                  CLR     YLOOP               ;WAIT
     032036  005237  002222          64$:    INC     YLOOP
     032042  023727  002222  000007          CMP     YLOOP,#7
     032050  001372                          BNE     64$
     032052  152777  000002  150202          BISB    #CBIT,@CSR          ;DO IT AGAIN
     032060  005037  002222                  CLR     YLOOP               ;WAIT
     032064  005237  002222          65$:    INC     YLOOP
     032070  023727  002222  000007          CMP     YLOOP,#7
     032076  001372                          BNE     65$
4981 032100  000207                          RTS     PC                  ;AND RETURN
4982
4983 032102                          DALLY:
     032102  005037  002222                  CLR     YLOOP               ;WAIT
     032106  005237  002222          64$:    INC     YLOOP
     032112  023727  002222  000200          CMP     YLOOP,#200
     032120  001372                          BNE     64$
4984 032122  000207                          RTS     PC
4985
4986 032124  000137  005106          EXERC:  JMP     MONIT
```

```
4988 032130                                F5010:
4989 032130                                F5011:
4990 032130  012737  000004  002242              MOV     #4,BYTNUM       ;# OF BYTES PER MODULE
4991 032136  004737  041122                      JSR     PC,MONDAT       ;SUBROUTINE TO MONITOR DATA CONTINUOUSLY
4992
4993 032142                                F5012:
4994 032142  012737  000002  002242              MOV     #2,BYTNUM       ;# OF BYTES PER MODULE
4995 032150  004737  041122                      JSR     PC,MONDAT
4996
4997 032154                                F5013:
4998 032154  012737  000001  002242              MOV     #1,BYTNUM       ;# OF BYTES PER MODULE
4999 032162  004737  041122                      JSR     PC,MONDAT
5000
5001 032166                                F6010:
5002 032166  012737  000004  002242              MOV     #4,BYTNUM;MODULE IS 4 BYTE LONG
5003 032174  012700  000004                      MOV     #4,R0   ;SET A BYTE COUNTER
5004 032200  004737  041052                      JSR     PC,SETPTN
5005 032204  000207                              RTS     PC
5006
5007 032206                                F6012:
5008 032206                                F6013:
5009 032206  012737  000001  002242              MOV     #1,BYTNUM
5010 032214  004737  041052                      JSR     PC,SETPTN       ;ROUTINE TO OUTPUT ANY PATTERN TOOUT MODULE
5011 032220  000207                              RTS     PC
5012
5013 032222                                F6011:
5014 032222  012737  000002  002242              MOV     #2,BYTNUM       ;MODULE IS 2 BYTE LONG
5015 032230  004737  041052                      JSR     PC,SETPTN
5016 032234  000207                              RTS     PC
```

```
                                        .SBTTL A/D MONITOR
                                        ;THIS IS MONITOR FOR FOR AUTO MODE FOR A/D
032236                                  ADTST:
032236  012777  040722  150110                  MOV     #KBINT,@KBVEC
032244  152777  000100  146706                  BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
032252  005005                                  CLR     R5
032254  004737  042016                          JSR     PC,ADADDR       ;SET ADDRESSES AND CHECK REG.
032260  005705                                  TST     R5              ;ADDRESS ERROR?
032262  001017                                  BNE     1$
032264  152777  000001  147770                  BISB    #RBIT,@CSR
032272  105077  150554                          CLRB    @STAT1
032276  105077  150552                          CLRB    @STAT2
032302  004737  042172                          JSR     PC,GCODE        ;FIND IF SE/DIFF MODE
032306  004737  033624                          JSR     PC,ADLOG        ;TEST LOGIC
032312  004737  035252                          JSR     PC,LINEAR       ;TEST LINEARITY
032316  004737  036322                          JSR     PC,RUMP         ;TEST MONOTINICITY
032322  000207                          1$:     RTS     PC

                                        ;THIS IS MONITOR FOR OPERATOR INTERVENTION TESTS

032324                                  ATOD:
032324  012777  040722  150022                  MOV     #KBINT,@KBVEC
032332  152777  000100  146620                  BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
032340  004737  032024                          JSR     PC,KLEER
032344  012737  032374  003160                  MOV     #ADRET,RETURN   ;SET RETURN ADDRESS FOR CONTROL A
032352  104401  052744                          TYPE    ,MASS13         ;WHICH ADDRESS
032356  104410                                  RDOCT
032360  012637  002244                          MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
032364  023737  002244  002276                  CMP     TADDR,BASE
032372  103754                                  BLO     ATOD
032374  005005                          ADRET:  CLR     R5
032376  004737  042016                          JSR     PC,ADADDR       ;SET ADDRESSES
032402  005705                                  TST     R5              ;ADDRESS ERROR?
032404  001347                                  BNE     ATOD
032406  152777  000001  147646                  BISB    #RBIT,@CSR
032414  105077  150432                          CLRB    @STAT1          ;SET CHANNEL ZERO-NO MAX
032420  004737  042172                          JSR     PC,GCODE                ;FIND GENERIC CODE

032424  104401  054700                  ADMON:  TYPE    ,MASS46         ;WHICH TEST TO RUN?
032430  012737  000014  002240                  MOV     #14,$MUT        ;SET MODULE TYPE FOR ERRORS
032436  105077  150410                          CLRB    @STAT1
032442  105077  150406                          CLRB    @STAT2
032446  104406                                  RDCHR
032450  012637  002432                          MOV     (SP)+,ANSW      ;;POP STACK INTO ANSW
032454  104401  002432                          TYPE    ,ANSW           ;ECHO
032460  104401  052175                          TYPE    ,MASS0          ;CR,LF
032464  004737  040646                          JSR     PC,CNTRC                ;CHECK FOR CONTROL
032470  122737  000103  002432                  CMPB    #'C,ANSW        ;IS IT CALIBRATION-C

032476  001003                                  BNE     1$              ;NO
032500  004737  032732                          JSR     PC,ADCALB       ;EXECUTE CALIBRATION
```

```
5071 032504  000747                          BR      ADMON
5072
5073
5074 032506  122737  000104  002432  1$:    CMPB    #'D,ANSW        ;IS IT LOGIC TEST
5075 032514  001005                         BNE     2$              ;NO
5076 032516  004737  042224          5$:    JSR     PC,SWLOOP
5077 032522  033624                         ADLOG                   ;DO ADLOG TEST
5078 032524  000137  032424                 JMP     ADMON
5079
5080 032530  122737  000115  002432  2$:    CMPB    #'M,ANSW        ;IS IT MONOTONICITY TEST-M
5081 032536  001005                         BNE     9$
5082 032540  004737  042224          8$:    JSR     PC,SWLOOP
5083 032544  036322                         RUMP                    ;DO RUMP TEST
5084 032546  000137  032424                 JMP     ADMON
5085
5086 032552  122737  000114  002432  9$:    CMPB    #'L,ANSW        ;IS IT LINEARITY TEST-L
5087 032560  001005                         BNE     10$             ;NO
5088 032562  004737  042224          12$:   JSR     PC,SWLOOP
5089 032566  035252                         LINEAR                  ;DO LINEAR TEST
5090 032570  000137  032424                 JMP     ADMON
5091
5092 032574  022737  000130  002432  10$:   CMP     #'X,ANSW
5093 032602  001037                         BNE     15$
5094 032604  104401  054166          17$:   TYPE    ,MASS37         ;WHAT MUX
5095 032610  104410                         RDOCT
5096 032612  012637  002304                 MOV     (SP)+,TEMP      ;;POP STACK INTO TEMP
5097 032616  001404                         BEQ     20$             ;NO MUX 0
5098 032620  122737  000007  002304         CMPB    #7,TEMP         ;MUX TO HIGH?
5099 032626  002003                         BGE     16$
5100 032630  104401  047136          20$:   TYPE    ,M20            ;?
5101 032634  000763                         BR      17$
5102 032636  006337  002304          16$:   ASL     TEMP
5103 032642  006337  002304                 ASL     TEMP
5104 032646  006337  002304                 ASL     TEMP
5105 032652  006337  002304                 ASL     TEMP
5106 032656  006337  002304                 ASL     TEMP
5107 032662  013737  002304  002224         MOV     TEMP,MXNUM
5108 032670  004737  042224                 JSR     PC,SWLOOP
5109 032674  037060                         MUX1            ;DO THIS TEST
5110 032676  000137  032424                 JMP     ADMON
5111 032702  122737  000110  002432  15$:   CMPB    #'H,ANSW
5112 032710  001004                         BNE     14$
5113 032712  104401  054031                 TYPE    ,MASS36         ;TYPE ALL OPTIONS
5114 032716  000137  032424                 JMP     ADMON
5115
5116 032722  104401  047136          14$:   TYPE    ,M20            ;UNKNOWN CHARACTER.
5117 032726  000137  032424                 JMP     ADMON
5118
5119
5120
5121
5122                                 .SBTTL  CALIBRATION OF A14 AND MUX
```

```
5124
5125 032732  104401  054166          ADCALB: TYPE    ,MASS37         ;WHICH MUX NUMBER YOU WANT TO
5126 032736  004737  042172                  JSR     PC,GCODE
5127 032742  004737  040646                  JSR     PC,CNTRC
5128                                                                  ;TEST-ZERO FOR A/D
5129 032746  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
5130 032752  012737  033574  000004          MOV     #20$,ERRVEC
5131 032760  104410                          RDOCT
5132 032762  012637  002304                  MOV     (SP)+,TEMP      ;;POP STACK INTO TEMP
5133 032766  122737  000007  002304          CMPB    #7,TEMP
5134 032774  002003                          BGE     1$
5135 032776  104401  047136                  TYPE    ,M20            ;?
5136 033002  000753                          BR      ADCALB
5137
5138 033004  006337  002304          1$:     ASL     TEMP            ;SHIFT IT 5 TIMES TO SET MUX #
5139 033010  006337  002304                  ASL     TEMP
5140 033014  006337  002304                  ASL     TEMP
5141 033020  006337  002304                  ASL     TEMP
5142 033024  006337  002304                  ASL     TEMP
5143 033030  104401  054232          5$:     TYPE    ,MASS38         ;WHICH CHANNEL?-ONLY FOR A014
5144 033034  104410                          RDOCT
5145 033036  012637  003042                  MOV     (SP)+,TEMP3     ;;POP STACK INTO TEMP3
5146 033042  105737  002304                  TSTB    TEMP
5147 033046  001022                          BNE     2$
5148 033050  022737  000301  003050          CMP     #301,GBITE      ;IS IT DIFF MODE
5149 033056  001007                          BNE     3$
5150
5151 033060  122737  000007  003042          CMPB    #7,TEMP3        ;CHECK IF CHANNEL NOT TOO HIGH
5152 033066  002130                          BGE     4$              ;OK
5153 033070  104401  054306                  TYPE    ,MASS39         ;TOO HIGH CHANNEL FOR DIFF MODE
5154 033074  000755                          BR      5$
5155
5156 033076  122737  000017  003042  3$:     CMPB    #17,TEMP3       ;IS CHANNEL TOO HIGH?
5157 033104  002121                          BGE     4$              ;OK
5158 033106  104401  054351                  TYPE    ,MASS40         ;TOO HIGH CHANNEL FOR SINGLE END
5159 033112  000746                          BR      5$
5160
5161 033114  113777  002304  147730  2$:     MOVB    TEMP,@SIAT1     ;HERE ONLY IF MUX-SET MUX #
5162 033122  152777  000004  147132          BISB    #GBIT,@CSR      ;SET GBIT
5163 033130  117737  147716  003040          MOVB    @SIAT1,TEMP2
5164 033136  042737  177400  003040          BIC     #177400,TEMP2   ;FIND FROM GENERIC CODE WHAT KIND
5165 033144  142777  000004  147110          BICB    #GBIT,@CSR      ;OF MUX AND IN WHAT MODE
5166 033152  122737  000342  003040          CMPB    #342,TEMP2      ;IS IT A156 SE?
5167 033160  001007                          BNE     6$              ;NO
5168 033162  122737  000037  003042          CMPB    #37,TEMP3       ;IS CHANNEL TOO HIGH
5169 033170  002067                          BGE     4$              ;NO
5170 033172  104401  054351                  TYPE    ,MASS40         ;TOO HIGH
5171 033176  000714                          BR      5$
5172
5173 033200  005000                  6$:     CLR     R0              ;SET GAIN =1
5174 033202  122737  000322  003040          CMPB    #322,TEMP2      ;IS IT A156 DIFF
5175 033210  001007                          BNE     7$
5176 033212  122737  000017  003042          CMPB    #17,TEMP3       ;IS CHANNEL TOO HIGH?
5177 033220  002053                          BGE     4$              ;NO
5178 033222  104401  054306                  TYPE    ,MASS39         ;YES
5179 033226  000700                          BR      5$
5180
```

```
5181 033230  122737  000323  003040  7$:     CMPB    #323,TEMP2      ;IS IT A157
5182 033236  001031                          BNE     8$              ;NO, UNKNOWN GENERIC CODE.
5183 033240  122737  000017  003042          CMPB    #17,TEMP3       ;IS CHANNEL TOO HIGH?
5184 033246  002003                          BGE     9$              ;NO
5185 033250  104401  054306                  TYPE    ,MASS39         ;YES
5186 033254  000665                          BR      5$
5187                                                                  ;HERE ONLY IF A157
5188 033256  104401  054424          9$:     TYPE    ,MASS41         ;WHICH GAIN?
5189 033262  104410                          RDOCT
5190 033264  012637  003044                  MOV     (SP)+,TEMP4     ;;POP STACK INTO TEMP4
5191 033270  005000                          CLR     R0
5192 033272  023760  003044  003062  11$:    CMP     TEMP4,GAINTB(R0)        ;FIND IF GAIN IS CORRECT
5193 033300  001413                          BEQ     21$             ;DO THE CALIBRATION FOR A157
5194 033302  062700  000002                  ADD     #2,R0
5195 033306  005760  003062                  TST     GAINTB(R0)
5196
5197 033312  001367                          BNE     11$
5198 033314  104401  054475                  TYPE    ,MASS42         ;WRONG GAIN SELECTED
5199 033320  000756                          BR      9$
5200
5201
5202 033322  104401  052636          8$:     TYPE    ,MASS11         ;UNKNOWN GENERIC CODE
5203 033326  000601                          BR      ADCALB
5204
5205
5206 033330  116077  003104  147516  21$:    MOVB    GAIN(R0),@STAT2 ;SET GAIN
5207 033336  104401  057346                  TYPE    ,GAINMG
5208 033342  016046  003062                  MOV     GAINTB(R0),-(SP)        ;;SAVE GAINTB(R0) FOR TYPEOUT
     033346  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5209 033350  153777  003042  147474  4$:     BISB    TEMP3,@STAT1
5210 033356  104401  054523                  TYPE    ,MASS43         ;CONNECT VOLTAGE SOURCE, TYPE CR
5211 033362  004737  041026                  JSR     PC,CRTST        ;WAIT FOR CR
5212 033366  104401  054560                  TYPE    ,MASS44         ;TYPE HEADER    -- OCTAL  VOLTAGE--
5213 033372  004737  037776          14$:    JSR     PC,CONV7        ;DO 7 CONV & AVERAGE THEM
5214 033376  012701  000002                  MOV     #2,R1
5215 033402                          13$:
     033402  005037  002222                  CLR     YLOOP           ;WAIT
     033406  005237  002222          64$:    INC     YLOOP
     033412  023727  002222  177777          CMP     YLOOP,#-1
     033420  001372                          BNE     64$
5216 033422  005301                          DEC     R1
5217 033424  001366                          BNE     13$
5218 033426  032737  000004  003162          BIT     #BIT2,AVRSAV
5219 033434  001410                          BEQ     40$
5220 033436  005237  002316                  INC     INTDAT
5221 033442  013746  002316                  MOV     INTDAT,-(SP)    ;;SAVE INTDAT FOR TYPEOUT
     033446  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5222 033450  005337  002316                  DEC     INTDAT
5223 033454  000403                          BR      41$
5224
5225 033456                          40$:
     033456  013746  002316                  MOV     INTDAT,-(SP)    ;;SAVE INTDAT FOR TYPEOUT
     033462  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5226 033464  162737  004000  002316  41$:    SUB     #4000,INTDAT    ;CALCULATE VOLTAGE IN MILLIVOLTS
5227 033472  013737  002316  001140          MOV     INTDAT,$GDDAT
5228 033500  063737  001140  001140          ADD     $GDDAT,$GDDAT   ;MULTIPLY BY 5
5229 033506  063737  001140  001140          ADD     $GDDAT,$GDDAT
```

```
5230 033514  063737  002316  001140          ADD     INTDAT,$GDDAT
5231 033522  042737  177770  003162          BIC     #177770,AVRSAV
5232 033530  013704  003162                  MOV     AVRSAV,R4
5233 033534  116404  003164                  MOVB    AVRTBL(R4),R4
5234 033540  060437  001140                  ADD     R4,$GDDAT
5235
5236 033544  104401  046636                  TYPE    ,M10            ;TAB
5237 033550  104401  046636                  TYPE    ,M10
5238 033554  013746  001140                  MOV     $GDDAT,-(SP)    ;;SAVE $GDDAT FOR TYPEOUT
     033560  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
5239 033562  104401  052175                  TYPE    ,MASSO          ;CR,LF
5240 033566  004737  040646                  JSR     PC,CNTRC        ;CONTROL C?
5241 033572  000677                          BR      14$             ;DO IT AGAIN.
5242 033574  022626                  20$:    CMP     (SP)+,(SP)+
5243 033576  104401  047742                  TYPE    ,M84            ;SELECTED MUX DID NOT RESPOND
5244 033602  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
5245 033606  152777  000001  146446          BISB    #RBIT,@CSR      ;RESET TIME-OUT
5246 033614  105777  147236                  TSTB    @LBYTE
5247 033620  000137  032732                  JMP     ADCALB
5248
5249                                  .SBTTL  LOGIC TEST OF A014
```

```
5251
5252 033624                            ADLOG:
                                       ;;*****************************************************************
     033624  000004                    TST27:  SCOPE
5253 033626  112737  000030  001206            MOVB    #30,$TESTN
5254
5255 033634  004737  032024                    JSR       PC    ,KLEER          ;CLEAR THE IOCM WORLD
5256 033640  012746  000000                    MOV     #PR0,-(SP)        ;SET PSW TO PRIORITY 0
     033644  012746  033652                    MOV     #64$,-(SP)
     033650  000002                            RTI
     033652  000240                    64$:    NOP
5257 033654  152777  000001  146400            BISB    #RBIT   , @CSR  ;CLEAR RANDOM INTERUPT
5258 033662  005037  001142                    CLR     $BDDAT
5259 033666  117737  147162  001142            MOVB    @STAT2,$BDDAT           ;STORE DATA IN $BDAT
5260 033674  132777  000001  147152            BITB    #BIT0   , @STAT2        ;CHECK BUSY BIT CLEAR
5261 033702  001401                            BEQ       1$            ;IF NO ERROR CONTINUE
5262 033704  104035                            EMT     35
5263
5264 033706  132777  000002  147140  1$:       BITB    #BIT1,@STAT2    ;CHECK CONV DONE IS CLEAR
5265 033714  001401                            BEQ       2$            ;IF NO ERROR CONTINUE
5266 033716  104035                            EMT     35
5267
5268 033720  132777  000004  147126  2$:       BITB    #BIT2,@STAT2    ;CHECK ERROR BIT IS CLEAR
5269 033726  001401                            BEQ       3$            ;IF NO ERROR CONTINUE
5270 033730  104035                            EMT     35
5271
5272 033732  132777  000010  147114  3$:       BITB    #BIT3,@STAT2    ;CHECK MUX TIME OUT IS CLEAR
5273 033740  001401                            BEQ     4$              ;IF NO ERROR CONTINUE
5274 033742  104035                            EMT     35
5275
5276 033744  132777  000360  147102  4$:       BITB    #360,@STAT2     ;CHECK GAIN BITS
5277 033752  001401                            BEQ     BSYCON          ;IF NO ERROR CONTINUE
5278 033754  104035                            EMT     35
5279 033756  012777  034122  146302  BSYCON:   MOV     #10$,@VECT0
5280 033764  012777  000340  146276            MOV     #PR7,@VECT0A
5281 033772  152777  000001  146262            BISB    #RBIT,@CSR      ;CLEAR INTERRUPTS
5282 034000  105777  147046                    TSTB    @STAT1
5283 034004  012700  177777                    MOV     #-1     , R0    ;INITIALIZE COUNTER
5284 034010  152777  000001  147036            BISB    #BIT0   , @STAT2        ;START CONVERSION
5285 034016  005200                    9$:     INC     R0              ;SRATR COUNTER
5286 034020  022700  000300                    CMP     #300    , R0    ;CHECK FOR COUNT OF 100
5287 034024  001002                            BNE       8$            ;STILL WAIT FOR CONVERSION
5288 034026  104040                            EMT     40
5289 034030  000474                            BR      7$
5290 034032  132777  000002  147014  8$:       BITB    #BIT1   , @STAT2        ;CHECK FOR CONVERSION DONE
5291 034040  001766                            BEQ       1$            ;IF NOT DONE THEN LOOP
5292 034042  132777  000001  147004  2$:       BITB    #BIT0   , @STAT2        ;VERIFY BUSY CLEAR
5293 034050  001402                            BEQ       3$            ;IF CLEAR CONTINUE
5294 034052  104036                            EMT     36
5295 034054  000462                            BR      7$
5296 034056  132777  000004  146770  3$:       BITB    #BIT2,@STAT2    ;CHECK ERROR CLEAR
5297 034064  001402                            BEQ     4$              ;CONTINUE IF CLEAR
5298 034066  104037                            EMT     37
5299 034070  000454                            BR      7$
5300 034072  105777  146164            4$:     TSTB    @CSR            ;TST INTERRUPT FLAG
5301 034076  100402                            BMI     5$              ;IT IS SET
5302 034100  104041                            EMT     41
```

```
5303 034102  000447                          BR      7$
5304 034104  152777  000100  146150  5$:     BISB    #EBIT,@CSR      ;ENEBLE INTERRUPT
5305 034112  000240                          NOP
5306 034114  000240                          NOP
5307 034116  104041                          EMT     41
5308 034120  000440                          BR      7$
5309
5310
5311 034122  022626                  10$:    CMP     (SP)+,(SP)+     ;ADJUST STOCK POINTER
5312 034124  142777  000100  146130          BICB    #EBIT,@CSR
5313 034132  013737  002276  002304          MOV     BASE,TEMP       ;INITIALIZE TEMP
5314 034140  132777  000200  146114  13$:    BITB    #EBIT,@CSR      ;INTERRUPT SET?
5315 034146  001424                          BEQ     12$             ;NO
5316
5317 034150  005037  002314                  CLR     TEMP1
5318 034154  117737  146104  002304          MOVB    @IAR,TEMP       ;FIND WHICH I/O INTERRUPTED
5319 034162  123737  002304  002244          CMPB    TEMP,TADDR      ;IS IT MUT
5320 034170  001414                          BEQ     7$              ;YES
5321 034172  152777  000001  146062          BISB    #RBIT,@CSR      ;CLEAR INTERRUPT
5322 034200  105777  146100                  TSTB    @TEMP
5323 034204  005237  002314                  INC     TEMP1
5324 034210  022737  000400  002314          CMP     #400,TEMP1
5325 034216  001350                          BNE     13$
5326 034220                          12$:
     034220  104041                          EMT     41
5327 034222  152777  000001  146032  7$:     BISB    #RBIT,@CSR      ;CLEAR INTERRUPTS
5328 034230  105777  146622                  TSTB    @LBYTE
5329 034234  004737  040564                  JSR     PC,CLRINT
5330 034240  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     034244  012746  034252                  MOV     #64$,-(SP)
     034250  000002                          RTI
     034252  000240          64$:    NOP
5331
5332
5333 034254                  CHSEL:
5334 034254  005000                          CLR       R0            ;SET TO START WITH CHANEL ZERO
5335 034256  005001                          CLR       R1            ;CLEAR REFERENCE REGISTER
5336 034260  105077  146566                  CLRB    @STAT1          ;CLEAR CHANEL TO ZERO
5337 034264  105777  146562                  TSTB    @STAT1          ;VERIFY CHANEL IS ZERO
5338 034270  001406                          BEQ       1$            ;IF ZERO CONTINUE
5339 034272  005037  001140                  CLR     $GDDAT          ;CHECK FOR CHANEL 0
5340 034276  117737  146550  001142          MOVB    @STAT1,$BDDAT   ;GET ACTUAL CHANEL
5341 034304  104042                          EMT     42
5342 034306  005200                  1$:     INC       R0            ;GET NEXT CHANEL TO CHECK
5343 034310  110077  146536                  MOVB      R0    , @STAT1        ;WRITE CHANEL SELECTED
5344 034314  117701  146532                  MOVB    @STAT1  , R1    ;READ CHANEL SELECTED
5345 034320  020001                          CMP       R0    , R1    ;VERIFY CORRECT CHANEL
5346 034322  001406                          BEQ       2$            ;CONTINUE IF MATCH
5347 034324  010037  001140                  MOV     R0,$GDDAT       ;STORE CHANEL WRITTEN
5348 034330  010137  001142                  MOV     R1,$BDDAT       ;STORE CHANEL READ
5349 034334  104042                          EMT     42
5350 034336  000431                          BR      CHNSEL
5351 034340  022737  000301  003050  2$:     CMP     #301    ,GBITE  ;CHECK FOR DIFF MODE
5352 034346  001404                          BEQ       3$            ;IF NOT DIFF MODE
5353 034350  022700  000017                  CMP     #17     , R0    ;CHECK FOR 17 CHANELS
5354 034354  001354                          BNE       1$            ;GET NEXT CHANEL UNTIL 17 DONE
5355 034356  000403                          BR        4$            ;EXIT WHEN DONE
```

```
5356 034360  022700  000007          3$:     CMP     #7      , R0    ;ELSE CHECK 7 DIFF CHANELS
5357 034364  001350                          BNE      1$             ;GET NEXT CHANEL UNTIL 7 DONE
5358 034366  132777  000004  146460  4$:     BITB    #BIT2,@STAT2    ;CHECK IF ERROR BIT CLEAR
5359 034374  001401                          BEQ     5$
5360 034376  104055                          EMT     55
5361 034400  105077  146446          5$:     CLRB    @STAT1          ;SET CHANELS TO ZERO
5362 034404  001406                          BEQ      CHNSEL         ;IF ZERO - DONE
5363 034406  005037  001140                  CLR     $GDDAT          ;STORE CHANEL WRITTEN
5364 034412  117737  146434  001142          MOVB    @STAT1,$BDDAT   ;STORE CHANEL READ
5365 034420  104042                          EMT     42
5366
5367 034422                          CHNSEL:
5368 034422  022737  000301  003050          CMP     #301    ,GBITE  ;CHECK IF IN DIFF MODE
5369 034430  001003                          BNE      1$             ;IN NOT DO SING END MODE
5370 034432  012700  000010                  MOV     #10     , R0    ;START WITH CH10 IN DIFF MODE
5371 034436  000402                          BR       2$             ;AND START CHECK
5372 034440  012700  000020          1$:     MOV     #20     , R0    ;START WITH CH20 IN SING END MODE
5373 034444  152777  000001  145610  2$:     BISB    #RBIT   , @CSR  ;CLEAR ANY INTERRUPTS
5374 034452  105777  146400                  TSTB    @LBYTE          ;CLEAR RIF BIT
5375 034456  110077  146370          4$:     MOVB      R0    , @STAT1        ;LOAD NON-EXISTANT CHANEL
5376 034462  132777  000004  146364          BITB    #BIT2   , @STAT2        ;CHECK ERROR BIT SET
5377 034470  001002                          BNE       5$            ;CONTINUE IF SET
5378 034472  104043                          EMT     43
5379 034474  000412                          BR      ADGAN
5380
5381 034476  132777  000200  145556  5$:     BITB    #BIT7,@CSR      ;TEST IF ERROR INTERRUPTED
5382 034504  001002                          BNE     6$
5383 034506  104044                          EMT     44
5384 034510  000404                          BR      ADGAN
5385 034512  005200                  6$:     INC       R0            ;UPDATE TO NEXT CHANEL
5386 034514  022700  000040                  CMP      #40    , R0    ;CHECK FOR LAST CHANEL
5387 034520  001351                          BNE      2$             ;CONTINUE IF NOT DONE
5388
5389 034522  012700  000020          ADGAN:  MOV     #20,R0          ;GET FIRST GAIN VALUE
5390 034526  152777  000001  145526  1$:     BISB    #RBIT,@CSR      ;CLEAR INTERRUPTS
5391 034534  105077  146312                  CLRB    @STAT1
5392 034540  105777  146312                  TSTB    @LBYTE
5393 034544  110077  146304                  MOVB    R0,@STAT2               ;WRITE GAIN
5394 034550  132777  000004  146276          BITB    #BIT2,@STAT2    ;CHECK FOR ERROR
5395 034556  001002                          BNE     2$              ;BRANCH IF ERROR
5396 034560  104045                          EMT     45
5397 034562  000405                          BR      3$
5398 034564  062700  000020          2$:     ADD     #20,R0          ;GET NEXT GAIN
5399 034570  022700  000400                  CMP     #400,R0         ;CHECK FOR LAST
5400 034574  001354                          BNE     1$              ;IF NOT CONTINUF
5401 034576  152777  000001  145456  3$:     BISB    #RBIT,@CSP      ;CLEAR INTERRUPTS
5402 034604  105777  146246                  TSTB    @LBYTE
5403
5404 034610  005001                  ADTBIT: CLR     R1              ;SET DIFF CHANEL NUMBER
5405 034612  005003                          CLR     R3              ;SET SEND CHANEL NUMB
5406 034614  005037  002302                  CLR     NORAMP          ;CLEAR NO RAMP FLAG
5407 034620  152777  000010  145434          BISB    #TBIT , @CSR    ;SET T-BIT
5408 034626  122737  000321  003050  2$:     CMPB    #321,GBITE      ;CHECK  GEN CODE
5409 034634  001403                          BEQ     3$
5410 034636  110177  146210                  MOVB    R1,@STAT1               ;WRITE DIFF CHANEL NUMBER
5411 034642  000402                          BR      12$
5412 034644  110377  146202          3$:     MOVB    R3,@STAT1               ;SELECT SEND CHANEL NUMBER
```

```
5413 034650  004737  037776          12$:    JSR     PC,CONV7        ;GO READ A/D DATA
5414 034654  005701                          TST     R1
5415 034656  001011                          BNE     5$              ;IF NOT CHECK NEXT CHANEL
5416 034660  023727  002316  007400          CMP     INTDAT,#7400            ;ELSE CHECK +V OUTPUT ALL ONES
5417 034666  002042                          BGE     9$              ;IF OK CHECK NEXT CHANEL
5418 034670  012737  007400  001140          MOV     #7400,$GDDAT    ;STORE LOWER LIMIT
5419 034676  104046                          EMT     46
5420 034700  000435                  4$:     BR      9$
5421 034702  022701  000001          5$:     CMP     #1      , R1    ;CHECK FOR CHANEL #1
5422 034706  001011                          BNE     7$              ;ELSE GO TO NEXT CHANEL
5423 034710  023727  002316  000400          CMP     INTDAT,#400             ;CHECK IF ABS ZERO IS IN LIMIT
5424 034716  003426                          BLE     9$              ;BRANCH IF OK
5425 034720  012737  000400  001140          MOV     #400,$GDDAT     ;STORE UPPER LIMIT
5426 034726  104046                          EMT     46
5427 034730  000421                  6$:     BR      9$              ;GET NEXT CHANEL
5428 034732  023727  002316  004005  7$:     CMP     INTDAT,#4005    ;CHECK HIGH LIMIT OF ZERO VALUE
5429 034740  003404                          BLE     8$              ;CONTINUE IF OK
5430 034742  012737  004005  001140          MOV     #4005,$GDDAT
5431 034750  104046                          EMT     46
5432 034752  023727  002316  003773  8$:     CMP     INTDAT,#3773    ;CHECK LOW LIMIT OF ZERO VALUE
5433 034760  002023                          BGE     11$             ;CONTINUE IF OK
5434 034762  012737  003773  001140          MOV     #3773,$GDDAT
5435 034770  104046                          EMT     46
5436 034772  000416                          BR      11$
5437 034774  005701                  9$:     TST      R1             ;IS CHANEL #0 SELECTED
5438 034776  001004                          BNE     10$             ;IF NOT CHECK NEXT CHANEL
5439 035000  005201                          INC     R1              ;ELSE UPDATE TO NEXT CHANEL
5440 035002  062703  000002                  ADD     #2,R3           ;UPDATE SEND CHANEL
5441 035006  000707                          BR      2$              ;AND GO CHECK IT
5442 035010  022701  000001          10$:    CMP     #1      . R1    ;IS CHANEL #1 SELECTED
5443 035014  001005                          BNE     11$             ;IF NOT CONTINUE
5444 035016  062701  000002                  ADD     #2      , R1    ;ELSE UPDATE TO CHANEL #3
5445 035022  062703  000004                  ADD     #4,R3           ;UPDATE SEND CHANEL
5446 035026  000677                          BR      2$              ;AND GO CHECK IT
5447 035030  122737  000321  003050  11$:    CMPB    #321,GBITE              ;TEST IF RAMP IS WORKING
5448 035036  001404                          BEQ     14$
5449 035040  112777  000002  146004          MOVB    #2,@STAT1       ;SET RAMP FOR SE
5450 035046  000403                          BR      15$
5451 035050  112777  000004  145774  14$:    MOVB    #4,@STAT1       ;SET RAMP FOR DIFF
5452 035056  005000                  15$:    CLR     R0
5453 035060  005001                          CLR     R1
5454 035062  004737  037776          17$:    JSR     PC,CONV7        ;DO CONVERSIONS
5455 035066  022737  007776  002316          CMP     #7776,INTDAT    ;WAIT FOR TOP OF RAMP
5456 035074  003412                          BLE     16$
5457 035076  105200                          INCB    R0
5458 035100  001370                          BNE     17$
5459 035102  005201                          INC     R1
5460 035104  022701  000050                  CMP     #50,R1
5461 035110  002364                          BGE     17$
5462 035112  104056                          EMT     56
5463 035114  005237  002302                  INC     NORAMP
5464 035120  000421                          BR      18$
5465
5466 035122  005001                  16$:    CLR     R1
5467 035124  005000                          CLR     R0
5468 035126  004737  037776          20$:    JSR     PC,CONV7        ;WAIT FOR BOTTOM OF RUMP
5469 035132  022737  000002  002316          CMP     #2,INTDAT
```

```
 5470 035140  002011                          BGE     18$
 5471 035142  105200                          INCB    R0              ;WAIT UP TO 6SEC
 5472 035144  001370                          BNE     20$
 5473 035146  005201                          INC     R1
 5474 035150  022701  000050                  CMP     #50.R1
 5475 035154  002364                          BGE     20$
 5476 035156  104056                          EMT     56
 5477 035160  005237  002302                  INC     NORAMP
 5478 035164  105077  145662          18$:    CLRB    @STAT1          ;SELECT CHANNEL 0 ALL ONES
 5479 035170  152777  000020  145064          BISB    #DBIT , @CSR    ;SET THE D-BIT
 5480 035176  004737  037776                  JSR     PC,CONV7        ;GO DO CONVERSION
 5481 035202  012737  003774  001140          MOV     #3774,$GDDAT    ;GET LOW END TOLER
 5482 035210  023737  001140  001142          CMP     $GDDAT,$BDDAT
 5483 035216  003401                          BLE     43$
 5484 035220  104050                          EMT     50
 5485 035222  012737  004004  001140  43$:    MOV     #4004,$GDDAT    ;GET HIGH END TOLER
 5486 035230  023737  001140  001142          CMP     $GDDAT,$BDDAT
 5487 035236  002001                          BGE     44$
 5488 035240  104050                          EMT     50
 5489 035242  004737  032024          44$:    JSR     PC,KLEER        ;CLEAR THE WORLD
 5490 035246  000004                          SCOPE
 5491 035250  000207                          RTS     PC              ;EXIT
 5492
 5493                                 .SBTTL  LINEARITY OF A014 TEST
```

```
5495 035252  012737  000031  001206  LINEAR: MOV     #31,$TESTN
5496 035260  005037  002306                  CLR     RERROR
5497 035264  005737  002302                  TST     NORAMP
5498 035270  001407                          BEQ     40$
5499 035272  005737  002312                  TST     AFLAG
5500 035276  001003                          BNE     41$
5501 035300  104056                          EMT     56
5502 035302  104401  050421                  TYPE    ,MASS51
5503 035306  000207                  41$:    RTS     PC
5504 035310  152777  000001  144744  40$:    BISB    #RBIT,@CSR              ;CLEAR IOCM RIF FLAG
5505 035316  105777  145532                  TSTB    @STAT2
5506 035322  005037  002436                  CLR     CONV
5507 035326  005037  002424                  CLR     ACOUNT
5508 035332  005037  002426                  CLR     BCOUNT
5509 035336  152777  000010  144716          BISB    #TBIT,@CSR              ;SET T BIT
5510 035344  004737  040564                  JSR     PC,CLRIN'
5511 035350  004737  037746                  JSR     PC,SETRAM               ;SET RAMP
5512 035354  004737  037776          1$:     JSR     PC,CONV7
5513 035360  022737  007777  002316          CMP     #7777,INTDAT            ;WAIT FOR TOP OF RAMP
5514 035366  001372                          BNE     1$
5515 035370  005237  002436                  INC     CONV                    ;WAIT FOR 50 CONVERSIONS AT 7777
5516 035374  022737  000050  002436          CMP     #50,CONV
5517 035402  001364                          BNE     1$
5518 035404  152777  000001  145442  3$:     BISB    #BIT0,@STAT2
5519 035412  132777  000002  145434  16$:    BITB    #BIT1,@STAT2
5520 035420  001774                          BEQ     16$
5521 035422  152777  000001  144632          BISB    #RBIT,@CSR
5522 035430  117737  145422  002440          MOVB    @LBYTE,DATALO
5523 035436  117705  145416                  MOVB    @HBYTE,R5
5524 035442  110537  002441                  MOVB    R5,DATALO+1
5525 035446  022737  007776  002440          CMP     #7776,DATALO            ;IS RAMP GOING DOWN
5526 035454  003753                          BLE     3$
5527 035456  013705  002440                  MOV     DATALO,R5
5528 035462  152777  000001  144572  5$:     BISB    #RBIT,@CSR              ;CLEAR DONE FLAG
5529 035470  152777  000001  145356          BISB    #BIT0,@STAT2            ;START CONVERSION
5530 035476  117737  145354  002440          MOVB    @LBYTE,DATALO           ;READ LOW BYTE
5531 035504  117705  145350                  MOVB    @HBYTE,R5               ;READ HIGH BYTE
5532 035510  110537  002441                  MOVB    R5,DATALO+1
5533 035514  013705  002440                  MOV     DATALO,R5               ;STORE IN TEXR LOC
5534 035520  010500                          MOV     R5,R0
5535 035522  005237  002424                  INC     ACOUNT                  ;STORE # OF CONVERSIONS THRU RAMP
5536 035526  001401                          BEQ     7$
5537 035530  000402                          BR      8$
5538 035532  005237  002426          7$:     INC     BCOUNT                  ;INCREAM IF ACOUNT OVERFLOW >32564
5539 035536  005700                  8$:     TST     R0                      ;IS IT END OF RAMP ?
5540 035540  001405                          BEQ     10$
5541 035542  132777  000002  145304  2$:     BITB    #BIT1,@STAT2            ;CHECK CONVER DONE
5542 035550  001774                          BEQ     2$                      ;LOOP IF NOT DONE
5543 035552  000743                          BR      5$                      ;ENABLE INTERR,START CONV
5544 035554  005001                  10$:    CLR     R1
5545 035556  006237  002426          11$:    ASR     BCOUNT                  ;FIND AVERAGE OF CONVERSIONS PER POINT
5546 035562  006037  002424                  ROR     ACOUNT                  ;BY DIVIDING # OF CONVERSIONS BY 4096
5547 035566  005201                          INC     R1                      ;POINTS
5548 035570  022701  000014                  CMP     #12.,R1
5549 035574  001370                          BNE     11$
5550 035576  005337  002424                  DEC     ACOUNT
5551 035602  013703  002424                  MOV     ACOUNT,R3
```

```
5552 035606  013702  002424                MOV     ACOUNT,R2
5553 035612  006202                       ASR     R2              ;LOW LIMIT AVERAGE OF CONVERSIONS/POINT
5554 035614  005503                       ADC     R3
5555 035616  060203                       ADD     R2,R3           ;HIGH LIMIT AVERAGE   " "
5556 035620  005203                       INC     R3
5557 035622  005302                       DEC     R2
5558 035624  005037  002442        49$:    CLR     CONVCT
5559 035630  005037  002436                CLR     CONV
5560 035634  152777  000001  144420      BISB    #RBIT,@CSR      ;CLEAR INTERRUPTS
5561 035642  105777  145204              TSTB    @STAT1
5562 035646  004737  037776        13$:    JSR     PC,CONV7
5563 035652  022737  007777  002316      CMP     #7777,INTDAT    ;WAIT FOR TOP OF RAMP
5564 035660  001372                       BNE     13$
5565 035662  005237  002436              INC     CONV            ;WAIT 50 CONVERSIONS AT TOP OF RAMP
5566 035666  022737  000050  002436      CMP     #50,CONV
5567 035674  001364                       BNE     13$
5568 035676  152777  000001  145150  14$:    BISB    #BIT0,@STAT2
5569 035704  132777  000002  145142  17$:    BITB    #BIT1,@STAT2
5570 035712  001774                       BEQ     17$
5571 035714  152777  000001  144340      BISB    #RBIT,@CSR
5572 035722  117737  145130  002440      MOVB    @LBYTE,DATALO
5573 035730  117705  145124              MOVB    @HBYTE,R5
5574 035734  110537  002441              MOVB    R5,DATALO+1
5575 035740  022737  007776  002440      CMP     #7776,DATALO    ;IS RAMP GOING DOWN ?
5576 035746  001353                       BNE     14$
5577 035750  013705  002440              MOV     DATALO,R5
5578 035754  010500                       MOV     R5,R0           ;INITIALIZE LOCATIONS
5579 035756  010237  002424              MOV     R2,ACOUNT       ;SET THIS TWO COUNTERS INITIALLY TO MIN COUNT
5580 035762  010237  002426              MOV     R2,BCOUNT
5581 035766  005037  002430              CLR     CCOUNT
5582 035772  005037  002434              CLR     DCOUNT
5583 035776  005037  002444              CLR     ECOUNT
5584 036002  005004                       CLR     R4
5585 036004  152777  000001  144250  33$:    BISB    #RBIT,@CSR      ;CLEAR INTERRUPTS
5586 036012  152777  000001  145034      BISB    #BIT0,@STAT2    ;START CONVER
5587 036020  117737  145032  002440      MOVB    @LBYTE,DATALO   ;READ LOW BYTE
5588 036026  117705  145026              MOVB    @HBYTE,R5       ;READ HIGH BYTE
5589 036032  110537  002441              MOVB    R5,DATALO+1
5590 036036  013705  002440              MOV     DATALO,R5       ;STORE IN TEXR LOC
5591 036042  010501                       MOV     R5,R1           ;STORE LAST CONV
5592 036044  160001                       SUB     R0,R1           ;GET DIFFRENCE BETWEEN LAST AND PREVIOUSE ONE
5593 036046  005401                       NEG     R1
5594 036050  006301                       ASL     R1
5595 036052  005261  002430              INC     CCOUNT(R1)      ;INC APPROPRIATE COUNTER
5596 036056  005701                       TST     R1              ;CHECK IF LAST READING WAS LOWER THEN ONE BEFORE
5597 036060  003005                       BGT     31$
5598
5599 036062  132777  000002  144764  32$:    BITB    #BIT1,@STAT2    ;WAIT FOR END OF CONV
5600 036070  001774                       BEQ     32$
5601 036072  000744                       BR      33$             ;START NEXT CONV
5602
5603 036074  023702  002424        31$:    CMP     ACOUNT,R2       ;CHECK FOR MIN # OF CONV
5604 036100  103430                       BLO     34$             ;ERROR
5605 036102  023703  002424              CMP     ACOUNT,R3       ;CHECK FOR MAX
5606 036106  101025                       BHI     34$             ;ERROR
5607 036110  013737  002426  002424  35$:    MOV     BCOUNT,ACOUNT
5608 036116  013737  002430  002426      MOV     CCOUNT,BCOUNT
```

```
 5609 036124  013737  002434  002430          MOV     DCOUNT,CCOUNT
 5610 036132  013737  002444  002434          MOV     ECOUNT,DCOUNT
 5611 036140  005037  002444                  CLR     ECOUNT
 5612 036144  005300                          DEC     R0                      ;SET NEXT LOWER CONV FOR CCOUNT
 5613 036146  003345                          BGT     32$                     ;CHECK FOR END OF RAMP
 5614 036150  005300                          DEC     R0                      ;CHECK LAST TWO CONV
 5615 036152  022700  177775                  CMP     #-3,R0
 5616 036156  001346                          BNE     31$
 5617 036160  000414                          BR      22$
 5618 036162  012764  000002  002446  34$:    MOV     #2,RAMP1(R4)            ;STORE ERROR
 5619 036170  060064  002446                  ADD     R0,RAMP1(R4)
 5620 036174  013764  002424  002566          MOV     ACOUNT,RAMP2(R4)
 5621 036202  005724                          TST     (R4)+                   ;INC ERROR POINTER
 5622 036204  022704  000050                  CMP     #40.,R4
 5623 036210  001337                          BNE     35$
 5624
 5625 036212  005704                  22$:    TST     R4                      ;ANY ERRORS ?
 5626 036214  001435                          BEQ     24$                     ;NO,EXIT
 5627 036216  005237  002306                  INC     RERROR                  ;DO IT 3 TIMES UNTILL NO ERRORS
 5628 036222  022737  000003  002306          CMP     #3,RERROR
 5629 036230  001402                          BEQ     25$
 5630 036232  000137  035624                  JMP     49$
 5631 036236                          25$:
      036236  104053                          EMT     53
 5632 036240  005005                          CLR     R5
 5633 036242  016537  002446  003026  23$:    MOV     RAMP1(R5),BLAST
 5634 036250  016537  002566  001142          MOV     RAMP2(R5),$BDDAT
 5635 036256  010237  002426                  MOV     R2,BCOUNT               ;STORE FOR ERROR PRINTOUT
 5636 036262  010337  002430                  MOV     R3,CCOUNT
 5637 036266  104052                          EMT     52
 5638 036270  062705  000002                  ADD     #2,R5                   ;INC ERROR TABLE POINTER
 5639 036274  020405                          CMP     R4,R5                   ;LAST ERROR ?
 5640 036276  001361                          BNE     23$
 5641 036300  022704  000050                  CMP     #40.,R4                 ;TOO MANY ERRORS ?
 5642 036304  001001                          BNE     24$                     ;NO
 5643 036306  104054                          EMT     54
 5644 036310  004737  040564          24$:    JSR     PC,CLRINT
 5645 036314  004737  032024                  JSR     PC,KLEER
 5646 036320  000207                          RTS     PC
 5647
 5648
```

```
5651                                    .SBTTL  A/D MONOTONICITY TEST
5652
5653                                    ;*************************************************************
5654                                    ;                   MONTONICITY TEST
5655                                    ;*************************************************************
5656 036322  012737  000032  001206  RUMP:   MOV     #32,$TESTN
5657 036330  012746  000000            MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
     036334  012746  036342            MOV     #64$,-(SP)
     036340  000002                    RTI
     036342  000240            64$:    NOP
5658 036344  005737  002302            TST     NORAMP
5659 036350  001407                    BEQ     1$
5660 036352  005737  002312            TST     AFLAG
5661 036356  001003                    BNE     2$
5662 036360  104056                    EMT     56
5663 036362  104401  050421            TYPE    ,MASS51
5664 036366  000207            2$:     RTS     PC
5665 036370                     1$:
5666 036370  152777  000001  143664    BISB    #RBIT,@CSR              ;CLEAR DONE FLAG
5667 036376  105777  144452            TSTB    @STAT2
5668 036402  152777  000010  143652    BISB    #TBIT,@CSR              ;SET T BIT
5669 036410  004737  040564            JSR     PC,CLRINT
5670 036414  005004            RAMPST: CLR     R4                      ;CLEAR ERROR POINTER
5671 036416  004737  037746            JSR     PC,SETRAM               ;SET RAMP
5672 036422  152777  000001  144424  1$:     BISB    #BIT0,@STAT2
5673 036430  132777  000002  144416  2$:     BITB    #BIT1,@STAT2
5674 036436  001774                    BEQ     2$
5675 036440  152777  000001  143614    BISB    #RBIT,@CSR
5676 036446  117737  144404  002440    MOVB    @LBYTE,DATALO
5677 036454  117737  144400  002441    MOVB    @HBYTE,DATALO+1
5678 036462  005737  002440            TST     DATALO                  ;WAIT FOR -10.240 VOLTS
5679 036466  001355                    BNE     1$
5680 036470  005037  003026      BEG:  CLR     BLAST                   ;STORE FIRST CONVERSION
5681 036474  005037  003030            CLR     LAST
5682 036500  005037  003032            CLR     LASTCN
5683 036504  152777  000001  144342  STCO:   BISB    #BIT0,@STAT2            ;START CONVERSION
5684 036512  132777  000002  144334  13$:    BITB    #BIT1,@STAT2
5685 036520  001774                    BEQ     13$
5686 036522  152777  000001  143532    BISB    #RBIT,@CSR
5687 036530  117737  144322  002440    MOVB    @LBYTE,DATALO
5688 036536  117737  144316  002441    MOVB    @HBYTE,DATALO+1
5689 036544  013737  002440  003030    MOV     DATALO,LAST
5690 036552  013737  002440  001142    MOV     DATALO,$BDDAT
5691 036560  023737  003026  003030    CMP     BLAST,  LAST
5692 036566  001431                    BEQ     3$
5693 036570  005337  003030            DEC     LAST
5694 036574  023737  003026  003030    CMP     BLAST,  LAST            ;COMPARE BLAST+1WITH LAST
5695 036602  001416                    BEQ     2$
5696 036604  062737  000002  003030    ADD     #2,     LAST
5697 036612  023737  003026  003030    CMP     BLAST,  LAST
5698 036620  001401                    BEQ     1$
5699 036622  000431                    BR      4$                      ;ERROR MESSAGE
5700 036624  005737  003032      1$:   TST     LASTCN                  ;IS LASTCN=0 ?
5701 036630  001410                    BEQ     3$                      ;YES
5702 036632  005337  003026            DEC     BLAST                   ;MOVE LAST CONVERSION BEFORE  A' '  A' ' N
5703 036636  000405                    BR      3$
5704 036640  005737  003032      2$:   TST     LASTCN                  ;IS LASTCN
```

```
5705 036644  001002                        BNE     3$
5706 036646  005237  003026                INC     BLAST
5707 036652  023727  003030  007777  3$:   CMP     LAST,   #7777           ;IS LAST=10.240 VOLTS ?
5708 036660  001003                        BNE     5$
5709 036662  012737  000001  003032        MOV     #1,     LASTCN          ;SET FLAG AT TOP OF RAMP
5710 036670  005737  003030          5$:   TST     LAST                    ;ARE WE STILL AT MINUS 10 VOLTS ?
5711 036674  001303                        BNE     STCO
5712 036676  005737  003032                TST     LASTCN                  ;IS RAMP GOING UP ?
5713 036702  001700                        BEQ     STCO
5714 036704  000431                        BR      27$                     ;CHECK FOR ERRORS
5715 036706  013737  003026  001140  4$:   MOV     BLAST,  $GDDAT          ;GOOD DATA FOR ERROR MESSAGE
5716 036714  005737  003032                TST     LASTCN                  ;IS RAMP GOING UP ?
5717 036720  001003                        BNE     7$                      ;YES, DECREMENT LAST CONVERSION
5718 036722  062737  000002  003026        ADD     #2,     BLAST           ;RAMP GOING DOWN
5719 036730  005337  003026          7$:   DEC     BLAST                   ;ADJUST BLAST AFTER ERROR
5720 036734  013764  001140  002446        MOV     $GDDAT,RAMP1(R4)
5721 036742  013764  001142  002566        MOV     $BDDAT,RAMP2(R4)
5722 036750  013764  003032  002706        MOV     LASTCN,RAMP3(R4)        ;STORE RAMP SLOPE IN TABLE
5723 036756  062704  000002                ADD     #2,R4                   ;INCREMENT TABLE POINTER
5724 036762  022704  000050                CMP     #40.,R4                 ;IS TABLE FULL
5725 036766  001331                        BNE     3$                      ;NO
5726 036770  005704          27$:  TST     R4                      ;ANY ERRORS ?
5727 036772  001425                        BEQ     OUT                     ;NO ,EXIT
5728 036774  104401  050154                TYPE    ,MASS55                 ;OUT OF RANGE BY TWO BITS
5729 037000  005003                        CLR     R3                      ;POINTER FOR ERROR TABLE
5730 037002  016337  002446  001140  28$:   MOV     RAMP1(R3),$GDDAT
5731 037010  016337  002566  001142        MOV     RAMP2(R3),$BDDAT
5732 037016  016337  002706  003032        MOV     RAMP3(R3),LASTCN
5733 037024  104051                        EMT     51
5734 037026  062703  000002                ADD     #2,R3                   ;INCREMENT DATA POINTER
5735 037032  020403                        CMP     R4,R3                   ;LAST ERROR ?
5736 037034  001362                        BNE     28$                     ;NO
5737 037036  020427  000050                CMP     R4,#40.                 ;TOO MANY ERRORS ?
5738 037042  001001                        BNE     OUT
5739 037044  104054                        EMT     54
5740
5741 037046  004737  040564          OUT:  JSR     PC,CLRINT
5742 037052  004737  032024                JSR     PC,KLEER
5743 037056  000207                        RTS     PC
5744
```

```
5746
5747                                    .SBTTL  LOGIC TEST OF A156 AND A157
5748
5749 037060  012737  000033  001206  MUX1:   MOV     #33,$TESTN
5750 037066  012737  000156  002240          MOV     #156,$MUT
5751 037074  013737  003052  002250          MOV     STAT1,TBADDR
5752 037102  105077  143746                  CLRB    @STAT2
5753 037106  005037  001142                  CLR     $BDDAT
5754 037112  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
5755 037116  012737  037740  000004          MOV     #30$,ERRVEC
5756 037124  113777  002224  143720          MOVB    MXNUM,@STAT1            ;SET MUX #
5757 037132  013737  002224  001140          MOV     MXNUM,$GDDAT
5758 037140  117737  143706  001142          MOVB    @STAT1,$BDDAT           ;CHECK IF MUX RESPONDS
5759 037146  123737  001140  001142          CMPB    $GDDAT,$BDDAT
5760 037154  001403                          BEQ     1$
5761 037156  104060                          EMT     60
5762 037160  000137  037724                  JMP     13$
5763 037164  113777  002224  143660  1$:     MOVB    MXNUM,@STAT1
5764 037172  004737  042172                  JSR     PC,GCODE
5765 037176  122737  000342  003050          CMPB    #342,GBITE              ;SE GENERIC CODE?
5766 037204  001455                          BEQ     2$
5767 037206  122737  000322  003050          CMPB    #322,GBITE              ;DIF MODE A156
5768 037214  001412                          BEQ     20$
5769 037216  122737  000323  003050          CMPB    #323,GBITE              ;A157?
5770 037224  001406                          BEQ     20$                     ;YES, DIFFERENTIAL MODE
5771 037226  013737  003050  003042          MOV     GBITE,TEMP3
5772 037234  104061                          EMT     61
5773 037236  000137  037724                  JMP     13$
5774 037242  012705  000017          20$:    MOV     #17,R5
5775 037246  005004                          CLR     R4
5776 037250  012737  000040  001140          MOV     #40,$GDDAT
5777 037256  113777  002224  143566          MOVB    MXNUM,@STAT1
5778 037264  113777  001140  143562  43$:    MOVB    $GDDAT,@STAT2           ;SET TOO HIGH CHANNEL
5779 037272  132777  000004  143554          BITB    #BIT2,@STAT2            ;CHECK IF ERROR SET
5780 037300  001001                          BNE     42$
5781 037302  104062                          EMT     62
5782 037304  004737  040646          42$:    JSR     PC,CNTRC
5783 037310  152777  000001  142744          BISB    #RBIT,@CSR              ;CLEAR ERROR BIT
5784 037316  105777  143530                  TSTB    @STAT1
5785 037322  005237  001140                  INC     $GDDAT
5786 037326  005204                          INC     R4
5787 037330  022704  000020                  CMP     #20,R4
5788 037334  001353                          BNE     43$
5789 037336  000402                          BR      21$
5790 037340  012705  000037          2$:     MOV     #37,R5
5791 037344  010537  001140          21$:    MOV     R5,$GDDAT               ;CHECK ALL CHANNELS
5792 037350  053737  002224  001140          BIS     MXNUM,$GDDAT
5793 037356  113777  001140  143466  4$:     MOVB    $GDDAT,@STAT1
5794 037364  117737  143462  001142          MOVB    @STAT1,$BDDAT
5795 037372  123737  001140  001142          CMPB    $GDDAT,$BDDAT
5796 037400  001403                          BEQ     3$
5797 037402  104063                          EMT     63
5798 037404  004737  040646                  JSR     PC,CNTRC
5799 037410  005337  001140          3$:     DEC     $GDDAT
5800 037414  023737  002224  001140          CMP     MXNUM,$GDDAT
5801 037422  001355                          BNE     4$
5802
```

```
5803 037424  005000                          CLR     R0
5804 037426  152777  000001  142626          BISB    #RBIT,@CSR
5805 037434  005037  001140                  CLR     $GDDAT
5806 037440  117737  143410  001142          MOVB    @STAT2,$BDDAT           ;CHECK IF ERROR,GAIN & DONE CLEAR
5807 037446  105737  001142                  TSTB    $BDDAT
5808 037452  001001                          BNE     5$
5809 037454  104064                          EMT     64
5810 037456  152777  000360  143370  5$:     BISB    #360,@STAT2             ;SET WRONG GAIN
5811 037464  132777  000004  143362          BITB    #BIT2,@STAT2
5812 037472  001001                          BNE     6$
5813 037474  104066                          EMT     66
5814 037476  152777  000001  142556  6$:     BISB    #RBIT,@CSR
5815 037504  105777  143344                  TSTB    @STAT2                  ;CLEAR ERROR
5816 037510  001001                          BNE     7$
5817 037512  104067                          EMT     67
5818 037514  005001                  7$:     CLR     R1
5819 037516  156177  003104  143330  22$:    BISB    GAIN(R1),@STAT2
5820 037524  152777  000010  142530          BISB    #^BIT,@CSR
5821 037532  113777  002224  143312          MOVB    MXNUM,@STAT1
5822 037540  152777  000001  143306          BISB    #BIT0,@STAT2            ;START CONVERSION
5823 037546  005200                  9$:     INC     R0
5824 037550  001002                          BNE     8$
5825 037552  104070                          EMT     70
5826 037554  000463                          BR      13$
5827 037556  132777  000002  143270  8$:     BITB    #BIT1,@STAT2            ;WAIT FOR CONV DONE
5828 037564  001770                          BEQ     9$
5829 037566  132777  000004  143260          BITB    #BIT2,@STAT2
5830 037574  001002                          BNE     10$
5831 037576  000452                          BR      13$
5832 037600  104071                          EMT     71
5833 037602  012737  003774  001140  10$:    MOV     #3774,$GDDAT
5834 037610  117737  143242  001142          MOVB    @LBYTE,$BDDAT
5835 037616  117737  143236  001143          MOVB    @HBYTE,$BDDAT+1
5836 037624  042737  170000  001142          BIC     #170000,$BDDAT          ;GET CONVERSION RESULTS
5837 037632  023737  001140  001142          CMP     $GDDAT,$BDDAT
5838 037640  003401                          BLE     11$
5839 037642  104072                          EMT     72
5840 037644  012737  004004  001140  11$:    MOV     #4004,$GDDAT
5841 037652  023737  001140  001142          CMP     $GDDAT,$BDDAT
5842 037660  002001                          BGE     12$
5843 037662  104072                          EMT     72
5844 037664  117737  143162  001142  12$:    MOVB    @STAT1,$BDDAT
5845 037672  001403                          BEQ     23$
5846 037674  005037  001140                  CLR     $GDDAT
5847 037700  104073                          EMT     73
5848 037702  122737  000323  003050  23$:    CMPB    #323,GBITE
5849 037710  001005                          BNE     13$
5850 037712  062701  000002                  ADD     #2,R1
5851 037716  005761  003104                  TST     GAIN(R1)
5852 037722  001275                          BNE     22$
5853 037724  004737  032024          13$:    JSR     PC,KLEER
5854 037730  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
5855 037734  000004                          SCOPE
5856 037736  000207                          RTS     PC
5857 037740  022626                  30$:    CMP     (SP)+,(SP)+
5858 037742  104065                          EMT     65
5859 037744  000767                          BR      13$
```

5860

```
                                        .SBTTL  SUBROUTINES
                                
                                
                                
                                
                                
037746  022737  000321  003050  SETRAM: CMP     #321,GBITE
037754  001004                          BNE     1$
037756  112777  000004  143066          MOVB    #4,@STAT1
037764  000403                          BR      2$
037766  112777  000002  143056  1$:     MOVB    #2,@STAT1
037774  000207                  2$:     RTS     PC
                                
                                        ;       THIS SUBROUTINE DOES 8 CONVERSIONS AND STORES AVERAGE IN INTDAT
037776  117737  143050  003126  CONV7:  MOVB    @STAT1,ST1SAV           ;SAVE STAT1
040004  117737  143044  003130          MOVB    @STAT2,ST2SAV           ;SAVE STAT2
040012  005037  002316                  CLR     INTDAT
040016  005037  003046                  CLR     TEMP5
040022  152777  000001  143024  2$:     BISB    #BIT0,@STAT2    ;START CONVERSION
040030  132777  000002  143016  1$:     BITB    #BIT1,@STAT2    ;WAIT FOR CONVERSION DONE
040036  001774                          BEQ     1$
040040  152777  000001  142214          BISB    #RBIT,@CSR
040046  117737  143004  002304          MOVB    @LBYTE,TEMP     ;ASSEMBLE DATA
040054  117737  143000  002305          MOVB    @HBYTE,TEMP+1
040062  042737  170000  002304          BIC     #170000,TEMP
040070  063737  002304  002316          ADD     TEMP,INTDAT     ;ADD TO PREVIOUS DATA
040076  005237  003046                  INC     TEMP5
040102  113777  003126  142742          MOVB    ST1SAV,@STAT1
040110  113777  003130  142736          MOVB    ST2SAV,@STAT2
040116  022737  000010  003046          CMP     #10,TEMP5       ;IS IT LAST ONE
040124  001336                          BNE     2$              ;NO
040126  013737  002316  003162          MOV     INTDAT,AVRSAV   ;SAVE TOTAL
040134  006237  002316                  ASR     INTDAT          ;FIND AVERAGE
040140  006237  002316                  ASR     INTDAT
040144  006237  002316                  ASR     INTDAT
040150  042737  170000  002316          BIC     #170000,INTDAT
040156  013737  002316  001142          MOV     INTDAT,$BDDAT
040164  000207                          RTS     PC
                                
                                        ;THIS SUBROUTINE IS USED TO CHECK IF
                                        ;A BITE OF DATA IN $GDDAT-$BDDAT.
                                        ;IT CHECKS AS MANY BYTES AS SPECIFIED
                                        ;BY     BYTNUM
                                
040166                          TSTBYT:
040166  013746  002242                  MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
040172  013746  002244                  MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
040176  117737  142042  001142  2$:     MOVB    @TADDR,$BDDAT   ;MOV DATA TO BDDAT
040204  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;IS DATA OK
040212  001404                          BEQ     1$
040214  013737  002244  002250          MOV     TADDR,TBADDR
040222  104007                          EMT     7
040224  005237  002244          1$:     INC     TADDR           ;NEXT BYTE
040230  005337  002242                  DEC     BYTNUM
040234  001360                          BNE     2$
040236  012637  002244                  MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
040242  012637  002242                  MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
040246  000207                          RTS     PC
```

```
 5919
 5920 040250                                TSTONE:
      040250  013746  002242                        MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
 5921 040254  013746  002244                        MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
 5922 040260  117737  141760  001142  2$:     MOVB    @TADDR,$BDDAT   ;MOV DATA TO BDDAT
 5923 040266  123737  001140  001142          CMPB    $GDDAT,$BDDAT   ;IS DATA OK
 5924 040274  001404                          BEQ     1$
 5925 040276  013737  002244  002250          MOV     TADDR,TBADDR
 5926 040304  104011                          EMT     11
 5927 040306  005237  002244          1$:     INC     TADDR           ;NEXT BYTE
 5928 040312  005337  002242                  DEC     BYTNUM
 5929 040316  001360                          BNE     2$
 5930 040320  012637  002244                  MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
 5931 040324  012637  002242                  MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
 5932 040330  000207                          RTS     PC
```

```
                                        ;THIS SUBROUTINE IS USED TO SET AS MANY
                                        ;BYTES AS SPECIFIED BY BYTNUM IN MUT

                                SETBYT:
040332  013746  002242                  MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
040336  013746  002244                  MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
040342  113777  001140  141674  1$:     MOVB    $GDDAT,@TADDR   ;SET DATA IN MUT
040350  005237  002244                  INC     TADDR
040354  005337  002242                  DEC     BYTNUM
040360  001370                          BNE     1$
040362  012637  002244                  MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
040366  012637  002242                  MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
040372  000207                          RTS     PC
```

```
5947
5948                                            ;THESE TWO SUBROUTINES ARE USED IN MAPPING THE SYSTEM
5949
5950 040374  005004                    GENCOD: CLR     R4
5951 040376  012703  002026                    MOV     #GENER,R3
5952 040402  022301                    3$:     CMP     (R3)+,R1
5953 040404  001404                            BEQ     1$
5954 040406  062704  000006                    ADD     #6,R4
5955 040412  005713                            TST     (R3)
5956 040414  001372                            BNE     3$
5957 040416  000207                    1$:     RTS     PC
5958
5959
5960
5961
5962 040420  005004                    TABLE:  CLR     R4
5963 040422  012703  002026                    MOV     #GENER,R3
5964 040426  022301                    3$:     CMP     (R3)+,R1
5965 040430  001404                            BEQ     1$
5966 040432  062704  000002                    ADD     #2,R4
5967 040436  005713                            TST     (R3)
5968 040440  001372                            BNE     3$
5969 040442  010022                    1$:     MOV     R0,(R2)+
5970 040444  010125                            MOV     R1,(R5)+
5971 040446  000207                            RTS     PC
```

```
 5973                                                   ;THIS SUBROUTINE CHECKS IF A BIT GET SET AND CLEAR IN CSR
 5974
 5975 040450  113777  001140  141604  BITSET: MOVB    $GDDAT,@CSR     ;LOAD TESTED BIT
 5976 040456  005037  001142                 CLR     $BDDAT
 5977 040462  005037  002222                 CLR     YLOOP           ;WAIT
      040466  005237  002222          64$:   INC     YLOOP
      040472  023727  002222  000007        CMP     YLOOP,#7
      040500  001372                        BNE     64$
 5978 040502  117737  141554  001142        MOVB    @CSR,$BDDAT     ;READ CSR
 5979 040510  023727  001206  000005        CMP     $TESTN,#5
 5980 040516  001403                        BEQ     3$
 5981 040520  023727  001206  000004        CMP     $TESTN,#4
 5982 040526  001003                3$:     BNE     2$
 5983 040530  142737  000200  001142        BICB    #FBIT,$BDDAT
 5984 040536  123737  001140  001142  2$:   CMPB    $GDDAT,$BDDAT   ;COMPARE THEM
 5985 040544  001002                        BNE     1$
 5986 040546  062716  000002                ADD     #2,(SP)         ;NO ERROR
 5987 040552  000207                1$:     RTS     PC
 5988
 5989                                                   ;THIS IS INTERRUPT RUTINE FOR LINE CLOCK
 5990                                                   ;WHEN DIAGNOSTIC DOES NOT USE IT
 5991
 5992 040554                          NOCLK:
 5993 040554  000002                        RTI
 5994
 5995                                                   ;THIS IS INTERRUPT RUTINE FOR CLOCK TO COUNT TICKS
 5996
 5997 040556  005237  002300          COUNT: INC     CLK
 5998 040562  000002                        RTI
 5999
```

```
6001                                                    ;THIS SUBROUTINE CLEARS UNEXPECTED INTERRUPTS FROM DBUS
6002 040564  013737  002276  003042  CLRINT: MOV     BASE,TEMP3      ;INITIALIZE BASE ADDRESS
6003 040572  005037  003044                  CLR     TEMP4
6004 040576  132777  000200  141456  2$:     BITB    #FBIT,@CSR      ;IS INTERRUPT PENDING?
6005 040604  001417                         BEQ     1$              ;NO
6006 040606  117737  141452  003042          MOVB    @IAR,TEMP3
6007 040614  152777  000001  141440          BISB    #RBIT,@CSR
6008 040622  105777  142214                  TSTB    @TEMP3
6009 040626  005237  003044                  INC     TEMP4
6010 040632  032737  000400  003044          BIT     #BIT8,TEMP4
6011 040640  001756                         BEQ     2$
6012 040642  104023                         EMT     23
6013 040644  000207                 1$:     RTS     PC
6014
6015
6016                                                    ;THIS SUBROUTINE CHECKS FOR CONTROL C
6017
6018
6019 040646  117746  140310          CNTRC:  MOVB    @$TKB,-(SP)
6020 040652  042716  000200                  BIC     #BIT7,(SP)      ;CLEAR PARITY BIT
6021 040656  122716  000003                  CMPB    #3,(SP)         ;CONTROL C?
6022 040662  001007                         BNE     1$              ;NO
6023 040664  004737  032024                  JSR     PC,KLEER
6024 040670  012737  005404  000004          MOV     #TMOVEC,ERRVEC
6025 040676  000137  005106                  JMP     MONIT
6026 040702  122716  000001          1$:     CMPB    #1,(SP)
6027 040706  001003                         BNE     2$
6028 040710  022626                         CMP     (SP)+,(SP)+
6029 040712  000177  142242                  JMP     @RETURN
6030 040716  005726                 2$:     TST     (SP)+
6031 040720  000207                         RTS     PC
6032
6033
6034 040722  004737  040646          KBINT:  JSR     PC,CNTRC
6035 040726  000002                         RTI
```

```
                                                ;THIS SUBROUTINE IS USED IN THE LOOP TEST
                                                ;AND IT IS SIMILAR TO THE TSTBYT ROUTINE

040730                                  CHKBYT:
040730  013746  002314                          MOV     TEMP1,-(SP)     ;;PUSH TEMP1 ON STACK
040734  013746  002242                          MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
040740  117737  141350  001142          2$:     MOVB    @TEMP1,$BDDAT   ;READ DATA FROM INPUT MODULE
040746  123737  001140  001142                  CMPB    $GDDAT,$BDDAT
040754  001410                                  BEQ     1$
040756  013737  002244  002250                  MOV     TADDR,TBADDR
040764  012637  002242                          MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
040770  012637  002314                          MOV     (SP)+,TEMP1     ;;POP STACK INTO TEMP1
040774  000207                                  RTS     PC              ;ERROR
040776  005237  002314                  1$:     INC     TEMP1           ;GO TO NEXT BYTE
041002  005337  002242                          DEC     BYTNUM
041006  001354                                  BNE     2$
041010  012637  002242                          MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
041014  012637  002314                          MOV     (SP)+,TEMP1     ;;POP STACK INTO TEMP1
041020  062716  000002                          ADD     #2,(R6)         ;BYPASS ERROR IN MAINLINE CODE
041024  000207                                  RTS     PC


                                                ;THIS SUBROUTINE STARTS CONVERSION
                                                ;THIS SUBROUTINE WAITS FOR A <CR> AND
                                                ;MONITORS CNTRLC
041026  104406                          CRTST:  RDCHR
041030  012637  002432                          MOV (SP)+,ANSW          ;SAVE ANSWER
041034  004737  040646                          JSR     PC,CNTRC                ;CONTROL C ?
041040  122737  000015  002432                  CMPB    #15,ANSW        ;A <CR> ?
041046  001367                                  BNE     CRTST           ;WRONG CHARACTER, JUST WAIT
041050  000207                                  RTS PC
```

```
                                                ;THIS SUBROUTINE ALLOWS FIELD ENGINEER TO SELECT ANY
                                                ;DATA PATTERN FOR OUTPUT MODULES


041052  142777  000020  141202  SETPIN: BICB    #DBIT,@CSR
041060  142777  000004  141174          BICB    #GBIT,@CSR
041066  104401  053127          1$:     TYPE    ,MASS19         ;SELECT A DATA PATTERN
041072  104410                          RDOCT
041074  012637  001140                  MOV     (SP)+,$GDDAT    ;;POP STACK INTO $GDDAT
041100  113777  001140  141136          MOVB    $GDDAT,@TADDR   ;OUTPUT PATTERN TO MODULE
041106  005237  002244                  INC     TADDR
041112  005337  002242                  DEC     BYTNUM
041116  001363                          BNE     1$
041120  000207                          RTS     PC


                                                ;THIS SUBROUTINE IS USED IN FIELD TEST TO CONTINUOUSLY
                                                ;MONITOR DATA FROM INPUT MODULE.DATA IS PRINTED ONLY
                                                ;DURING FIRST PASS UNLESS THERE IS A CHANGE IN THE DATA
041122  005001                  MONDAT: CLR     R1              ;CLEAR PASS REGISTER
041124  013700  002242          6$:     MOV     BYTNUM,R0       ;# OF BYTES PER MODULE
041130  005300                  5$:     DEC     R0
041132  117737  141106  001140  1$:     MOVB    @TADDR,$GDDAT   ;STORE DATA
041140  005701                          TST     R1              ;IS IT FIRST PASS?
041142  001016                          BNE     4$              ;NO
041144  117760  141074  002226  2$:     MOVB    @TADDR,DBUFF(R0)        ;STORE DATA IN TABLE
041152  013746  002244                  MOV     TADDR,-(SP)     ;;SAVE TADDR FOR TYPEOUT
041156  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
041160  104401  053451                  TYPE    ,MASS27         ;COLON & TAB
041164  013746  001140                  MOV     $GDDAT,-(SP)    ;;SAVE $GDDAT FOR TYPEOUT
041170  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(AL. DIGITS)
041172  104401  052175                  TYPE    ,MASSO          ;CR & LF
041176  000404                          BR      3$
041200  123760  001140  002226  4$:     CMPB    $GDDAT,DBUFF(R0)        ;ANY CHANGE IN DATA ?
041206  001356                          BNE     2$              ;YES
041210  005237  002244          3$:     INC     TADDR           ;GO TO NEXT BYTE IN MODULE
041214  005700                          TST     R0              ;LAST BYTE IN MODULE ?
041216  001344                          BNE     5$              ;NO
041220  163737  002242  002244          SUB     BYTNUM,TADDR    ;RESTORE ADDRESS OF MUT
041226  004737  040646                  JSR     PC,CNTRC        ;TYPE CONTROL C TO RETURN TO MONITOR
041232  012701  000001                  MOV     #1,R1           ;NEW PASS
041236  000732                          BR      6$
```

```
6110
6111                                    ;THIS SUBROUTINE IS USED TO WRAP AROUND DATA "LOOP TEST"
6112
6113
6114 041240  004737  032024          LOPTST: JSR     PC,KLEER
6115 041244  020127  000141                  CMP     R1,#141 ;IS IT M5010?
6116 041250  001131                          BNE     1$              ;YES
6117 041252  012737  000125  001140          MOV     #125,$GDDAT
     041260  012737  000004  002242          MOV     #4,BYTNUM
     041266  004737  040332                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
     041272  005037  002222                  CLR     YLOOP           ;WAIT
     041276  005237  002222          64$:    INC     YLOOP
     041302  023727  002222  001777          CMP     YLOOP,#1777
     041310  001372                          BNE     64$
     041312  012737  000004  002242          MOV     #4,BYTNUM
     041320  004737  040730                  JSR     PC,CHKBYT       ;COMPARE DATA
     041324  104021                          EMT     21
6118 041326  012737  000252  001140          MOV     #252,$GDDAT
     041334  012737  000004  002242          MOV     #4,BYTNUM
     041342  004737  040332                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
     041346  005037  002222                  CLR     YLOOP           ;WAIT
     041352  005237  002222          65$:    INC     YLOOP
     041356  023727  002222  001777          CMP     YLOOP,#1777
     041364  001372                          BNE     65$
     041366  012737  000004  002242          MOV     #4,BYTNUM
     041374  004737  040730                  JSR     PC,CHKBYT       ;COMPARE DATA
     041400  104021                          EMT     21
6119 041402  012737  000377  001140          MOV     #377,$GDDAT
     041410  012737  000004  002242          MOV     #4,BYTNUM
     041416  004737  040332                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
     041422  005037  002222                  CLR     YLOOP           ;WAIT
     041426  005237  002222          66$:    INC     YLOOP
     041432  023727  002222  001777          CMP     YLOOP,#1777
     041440  001372                          BNE     66$
     041442  012737  000004  002242          MOV     #4,BYTNUM
     041450  004737  040730                  JSR     PC,CHKBYT       ;COMPARE DATA
     041454  104021                          EMT     21
6120 041456  012737  000000  001140          MOV     #0,$GDDAT
     041464  012737  000004  002242          MOV     #4,BYTNUM
     041472  004737  040332                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
     041476  005037  002222                  CLR     YLOOP           ;WAIT
     041502  005237  002222          67$:    INC     YLOOP
     041506  023727  002222  001777          CMP     YLOOP,#1777
     041514  001372                          BNE     67$
     041516  012737  000004  002242          MOV     #4,BYTNUM
     041524  004737  040730                  JSR     PC,CHKBYT       ;COMPARE DATA
     041530  104021                          EMT     21
6121 041532  000530                          BR      2$
6122 041534                          1$:
     041534  012737  000125  001140          MOV     #125,$GDDAT
     041542  012737  000002  002242          MOV     #2,BYTNUM
     041550  004737  040332                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
     041554  005037  002222                  CLR     YLOOP           ;WAIT
     041560  005237  002222          68$:    INC     YLOOP
     041564  023727  002222  001777          CMP     YLOOP,#1777
     041572  001372                          BNE     68$
     041574  012737  000002  002242          MOV     #2,BYTNUM
```

```
                041602  004737  040730                  JSR     PC,CHKBYT       ;COMPARE DATA
                041606  104021                          EMT     21
6123            041610  012737  000252  001140          MOV     #252,$GDDAT
                041616  012737  000002  002242          MOV     #2,BYTNUM
                041624  004737  040332                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
                041630  005037  002222                  CLR     YLOOP           ;WAIT
                041634  005237  002222          69$:    INC     YLOOP
                041640  023727  002222  001777          CMP     YLOOP,#1777
                041646  001372                          BNE     69$
                041650  012737  000002  002242          MOV     #2,BYTNUM
                041656  004737  040730                  JSR     PC,CHKBYT       ;COMPARE DATA
                041662  104021                          EMT     21
6124            041664  012737  000377  001140          MOV     #377,$GDDAT
                041672  012737  000002  002242          MOV     #2,BYTNUM
                041700  004737  040332                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
                041704  005037  002222                  CLR     YLOOP           ;WAIT
                041710  005237  002222          70$:    INC     YLOOP
                041714  023727  002222  001777          CMP     YLOOP,#1777
                041722  001372                          BNE     70$
                041724  012737  000002  002242          MOV     #2,BYTNUM
                041732  004737  040730                  JSR     PC,CHKBYT       ;COMPARE DATA
                041736  104021                          EMT     21
6125            041740  012737  000000  001140          MOV     #0,$GDDAT
                041746  012737  000002  002242          MOV     #2,BYTNUM
                041754  004737  040332                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
                041760  005037  002222                  CLR     YLOOP           ;WAIT
                041764  005237  002222          71$:    INC     YLOOP
                041770  023727  002222  001777          CMP     YLOOP,#1777
                041776  001372                          BNE     71$
                042000  012737  000002  002242          MOV     #2,BYTNUM
                042006  004737  040730                  JSR     PC,CHKBYT       ;COMPARE DATA
                042012  104021                          EMT     21
6126            042014  000207          2$:             RTS     PC
6127
6128
```

```
                                        ;THIS SUBROUTINE SETS UP ADDRESSES FOR A/D A014
                                        
042016  013737  002244  003056  ADADDR: MOV     TADDR,LBYTE
042024  013746  002244                  MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
042030  013737  002244  003060          MOV     TADDR,HBYTE
042036  005237  003060                  INC     HBYTE
042042  013737  003060  003052          MOV     HBYTE,STAT1
042050  005237  003052                  INC     STAT1
042054  013737  003052  003054          MOV     STAT1,STAT2
042062  005237  003054                  INC     STAT2
042066  013737  002244  002250          MOV     TADDR,TBADDR
042074  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
042100  012737  042154  000004          MOV     #1$,ERRVEC
042106  105777  140744                  TSTB    @LBYTE
042112  005237  002244                  INC     TADDR
042116  105777  140736                  TSTB    @HBYTE
042122  005237  002244                  INC     TADDR
042126  105777  140720                  TSTB    @STAT1
042132  005237  002244                  INC     TADDR
042136  105777  140712                  TSTB    @STAT2
042142                          2$:
042142  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
042146  012637  002244                  MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
042152  000207                          RTS     PC
                                        
042154  022626                  1$:     CMP     (SP)+,(SP)+
042156  104026                          EMT     26
042160  012737  005404  000004          MOV     #TMOVEC,ERRVEC
042166  005205                          INC     R5
042170  000764                          BR      2$
                                        
                                        
042172  152777  000004  140062  GCODE:  BISB    #GBIT,@CSR              ;LOAD GBITE WITH GENERIC CODE
042200  117737  140646  003050          MOVB    @STAT1,GBITE
042206  042737  177400  003050          BIC     #177400,GBITE
042214  142777  000004  140040          BICB    #GBIT,@CSR
042222  000207                          RTS     PC
                                        ;THIS SUBROUTINE RUNS A TEST AND CHECKS SW13 AND SW14
042224  005037  001210          SWLOOP: CLR $PASS
042230  011637  002310                  MOV     (SP),XXX                ;GET SUBR ADDR
042234  017737  140050  002310          MOV     @XXX,XXX                ;ANOTHER LEVEL OF DEFERRED
042242  004777  140042          1$:     JSR     PC,@XXX
042246  005237  001210                  INC     $PASS
042252  023737  001210  002320          CMP     $PASS,PASCNT
042260  001370                          BNE     1$
042262  032737  020000  000176          BIT     #BIT13,SWREG
042270  001005                          BNE     2$
042272  013746  002320                  MOV     PASCNT,-(SP)    ;;SAVE PASCNT FOR TYPEOUT
042276  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
042300  104401  046640                  TYPE    ,M11
042304  032737  040000  000176  2$:     BIT     #BIT14,SWREG
042312  001344                          BNE     SWLOOP
042314  062716  000002                  ADD     #2,(R6)         ;ADVANCE RETURN PC
042320  000207                          RTS     PC
```

```
 6183                                   .SBTTL  SCOPE HANDLER ROUTINE
                                        ;;**************************************************************
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      SCOPE             ;;SCOPE=IOT
042322                                  $SCOPE:
                                        ;#####START OF CODE FOR THE XOR TESTER#####
042322  000416                 $XTSTR: BR      6$                ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP-240)
042324  013746  000004                 MOV     @#ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
042330  012737  042350  000004         MOV     #5$,@#ERRVEC      ;;SET FOR TIMEOUT
042336  005737  177060                 TST     @#177060          ;;TIME OUT ON XOR?
042342  012637  000004                 MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
042346  000421                         BR      $SVLAD            ;;GO TO THE NEXT TEST
042350  022626                 5$:     CMP     (SP)+,(SP)+       ;;CLEAR THE STACK AFTER A TIME OUT
042352  012637  000004                 MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
042356  000407                         BR      7$                ;;LOOP ON THE PRESENT TEST
042360                                 6$:;#####END OF CODE FOR THE XOR TESTER#####
042360  105737  001117                 2$:     TSTB    $ERFLG            ;;HAS AN ERROR OCCURRED?
042364  001412                         BEQ     $SVLAD            ;;BR IF NO
042366  032777  001000  136560         BIT     #BIT09,@SWR       ;;LOOP ON ERROR?
042374  001404                         BEQ     4$                ;;BR IF NO
042376  013737  001124  001122  7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
042404  000420                         BR      $OVER
042406  105037  001117          4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
042412  105237  001116          $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
042416  113737  001116  001206         MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
042424  011637  001122                 MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
042430  011637  001124                 MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
042434  005037  001174                 CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
042440  112737  000001  001131         MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
042446  013777  001116  136502  $OVER: MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
042454  013716  001122                 MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
042460  000002                         RTI                       ;;FIXES PS
042462                                 $SW08TBL:
        000027                         .REPT   $TN-1
042462  010212                         .WORD   TST1+2            ;;STARTING ADDRESS OF TEST 1
042464  010300                         .WORD   TST2+2            ;;STARTING ADDRESS OF TEST 2
042466  010344                         .WORD   TST3+2            ;;STARTING ADDRESS OF TEST 3
042470  010410                         .WORD   TST4+2            ;;STARTING ADDRESS OF TEST 4
042472  010540                         .WORD   TST5+2            ;;STARTING ADDRESS OF TEST 5
042474  010610                         .WORD   TST6+2            ;;STARTING ADDRESS OF TEST 6
042476  010654                         .WORD   TST7+2            ;;STARTING ADDRESS OF TEST 7
042500  010720                         .WORD   TST10+2           ;;START'' S 'DDRESS OF TEST 10
042502  011014                         .WORD   TST11+2           ;;STAR  . ADDRESS OF TEST 11
042504  011244                         .WORD   TST12+2           ;;STARTING ADDRESS OF TEST 12
042506  011346                         .WORD   TST13+2           ;;STARTING ADDRESS OF TEST 13
042510  012010                         .WORD   TST14+2           ;;STARTING ADDRESS OF TEST 14
042512  012202                         .WORD   TST15+2           ;;STARTING ADDRESS OF TEST 15
042514  012374                         .WORD   TST16+2           ;;STARTING ADDRESS OF TEST 16
042516  012672                         .WORD   TST17+2           ;;STARTING ADDRESS OF TEST 17
042520  013522                         .WORD   TST20+2           ;;STARTING ADDRESS OF TEST 20
042522  014200                         .WORD   TST21+2           ;;STARTING ADDRESS OF TEST 21
```

```
042524  016404                          .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
042526  021550                          .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
042530  024146                          .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
042532  027400                          .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
042534  030432                          .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
042536  033626                          .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
 6184                           .SBTTL  TTY INPUT ROUTINE
                                ;;**************************************************************
                                .ENABL  LSB
                                .DSABL  LSB
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;
042540  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
042542  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
042550  105777  136404          1$:     TSTB    @$TKS           ;;WAIT FOR
042554  100375                          BPL     1$              ;;A CHARACTER
042556  117766  136400  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
042564  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
042572  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
042600  001013                          BNE     3$              ;;BRANCH IF NO
042602  105777  136352          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
042606  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
042610  117746  136346                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
042614  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
042620  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
042624  001366                          BNE     2$              ;;IF NOT DISCARD IT
042626  000750                          BR      1$              ;;YES, RESUME
042630  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
042636  002407                          BLT     4$              ;;BRANCH IF YES
042640  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
042646  003003                          BGT     4$              ;;BRANCH IF YES
042650  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
042656  000002                  4$:     RTI                     ;;GO BACK TO USER
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
042660  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
042662  012703  042766          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
042666  022703  042776          2$:     CMP     #$TTYIN+80.,R3  ;;BUFFER FULL?
042672  101405                          BLOS    4$              ;;BR IF YES
042674  104406                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
042676  112613                          MOVB    (SP)+,(R3)      ;;G T CHARACTER
042700  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
042704  001003                          BNE     3$              ;;SKIP IF NOT
042706  104401  001176          4$:     TYPE    ,$QUES          ;;TYPE A '?'
042712  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
042714  111337  042764          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
042720  104401  042764                  TYPE    ,9$
042724  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
```

```
042730  001356                          BNE     2$              ;;LOOP IF NOT RETURN
042732  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
042736  104401  001200                  TYPE    ,$LF            ;;TYPE A LINE FEED
042742  012603                          MOV     (SP)+,R3        ;;RESTORE R3
042744  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
042746  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
042754  012766  042766  000004          MOV     #$TTYIN,4(SP)
042762  000002                          RTI                     ;;RETURN
042764     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
042765     000                          .BYTE   0               ;;TERMINATOR
042766                          $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
042776     136     125     015  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
043001     012     000
043003     136     107     015  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
043006     012     000
043010     015     012     123  $MSWR:  .ASCIZ  <15><12>/SWR - /
043013     127     122     040
043016     075     040     000
043021     040     040     116  $MNEW:  .ASCIZ  /  NEW - /
043024     105     127     040
043027     075     040     000

6185                    .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
                        ;;**************************************************************
                        ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                        ;*CHANGE IT TO BINARY.
                        ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
                        ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                        ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                        ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                        ;*CALL:
                        ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                        ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                        ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
043032  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
043034  016666  000004  000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
043042  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
043044  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
043046  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
043050  104407          1$:     RDLIN                   ;;READ AN ASCIZ LINE
043052  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
043054  010037  043160          MOV     R0,5$           ;;AND SAVE IT
043060  005001                  CLR     R1              ;;CLEAR DATA WORD
043062  005002                  CLR     R2
043064  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
043066  001420                  BEQ     3$              ;;IF ZERO GET OUT
043070  122716  000060          CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
043074  003026                  BGT     4$              ;;IS AN OCTAL DIGIT
043076  122716  000067          CMPB    #'7,(SP)
043102  002423                  BLT     4$
043104  006301                  ASL     R1              ;;*2
043106  006102                  ROL     R2
043110  006301                  ASL     R1              ;;*4
043112  006102                  ROL     R2
043114  006301                  ASL     R1              ;;*8
043116  006102                  ROL     R2
043120  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
043124  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
```

```
043126  000756                          BR      2$              ;;LOOP
043130  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
043132  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
043136  010237  043170                  MOV     R2,$HIOCT
043142  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
043144  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
043146  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
043150  000002                          RTI                     ;;RETURN
043152  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
043154  105010                          CLRB    (R0)            ;;SET A TERMINATOR
043156  104401                          TYPE                    ;;TYPE UP THRU THE BAD CHAR.
043160  000000                  5$:     .WORD   0
043162  104401  001176                  TYPE    ,$QUES          ;;"?" "CR" & "LF"
043166  000730                          BR      1$              ;;TRY AGAIN
043170  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
  6186                          .SBTTL  ERROR HANDLER ROUTINE
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15-1         HALT ON ERROR
                                ;*SW13-1         INHIBIT ERROR TYPEOUTS
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR EMT AND N=ERROR ITEM NUMBER
043172                          $ERROR:
043172  105237  001117          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
043176  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
043200  013777  001116  135750          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
043206  005237  001126                  INC     $ERTTL          ;;INC THE ERROR COUNT
043212  011637  001132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
043216  162737  000002  001132          SUB     #2,$ERRPC
043224  117737  135702  001130          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
043232  032777  020000  135714          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
043240  001004                          BNE     20$             ;;SKIP TYPEOUTS
043242  004737  043352                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
043246  104401  001177                  TYPE    ,$CRLF
043252                          20$:
043252  122737  000001  001222          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
043260  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
043262  113737  001130  043274          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
043270  004737  044236                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
043274     000                  21$:    .BYTE   0
043275     000                          .BYTE   0
043276  000777                  22$:    BR      22$             ;;APT ERROR LOOP
043300  005777  135650          2$:     TST     @SWR            ;;HALT ON ERROR
043304  100001                          BPL     3$              ;;SKIP IF CONTINUE
043306  000000                          HALT                    ;;HALT ON ERROR.
043310  032777  001000  135636  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
043316  001402                          BEQ     4$              ;;BR IF NO
043320  013716  001124                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
043324  005737  001174          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
043330  001402                          BEQ     5$              ;;BR IF NONE
043332  013716  001174                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
043336                          5$:
043336  022737  006724  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
```

```
        043344  001001                  BNE     6$              ;;BRANCH IF NO
        043346  000000                  HALT                    ;;YES
        043350                  6$:
        043350  000002                  RTI                     ;;RETURN
  6187                          .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
                                ;;*****************************************************************
                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
        043352                  $ERRTYP:
        043352  104401  001177          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
        043356  010046                  MOV     R0,-(SP)        ;;SAVE R0
        043360  005000                  CLR     R0              ;;PICKUP THE ITEM INDEX
        043362  153700  001130          BISB    @#$ITEMB,R0
        043366  001004                  BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
        043370  013746  001132          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
        043374  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        043376  000426                  BR      6$              ;;GET OUT
        043400  005300          1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
        043402  006300                  ASL     R0              ;;      WORK FOR THE ERROR TABLE
        043404  006300                  ASL     R0
        043406  006300                  ASL     R0
        043410  062700  003174          ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
        043414  012037  043424          MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
        043420  001404                  BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
        043422  104401                  TYPE                    ;;TYPE THE "ERROR MESSAGE"
        043424  000000          2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
        043426  104401  001177          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
        043432  012037  043442  3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
        043436  001404                  BEQ     5$              ;;SKIP TYPEOUT IF 0
        043440  104401                  TYPE                    ;;TYPE THE "DATA HEADER"
        043442  000000          4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
        043444  104401  001177          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
        043450  011000          5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
        043452  001004                  BNE     7$              ;;GO TYPE THE DATA
        043454  012600          6$:     MOV     (SP)+,R0        ;;RESTORE R0
        043456  104401  001177          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
        043462  000207                  RTS     PC              ;;RETURN
        043464                  7$:
        043464  013046                  MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
        043466  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        043470  005710                  TST     (R0)            ;;IS THERE ANOTHER NUMBER?
        043472  001770                  BEQ     6$              ;;BR IF NO
        043474  104401  043502          TYPE    ,8$             ;;TYPE TWO(2) SPACES
        043500  000771                  BR      7$              ;;LOOP
        043502     040     040     000  8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
                                        .EVEN
  6188                          .SBTTL  TYPE ROUTINE
                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
```

```
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*
043506  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
043512  100002                          BPL     1$              ;;BR IF YES
043514  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
043516  000430                          BR      3$              ;;LEAVE
043520  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
043522  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
043526  122737  000001  001222          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
043534  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
043536  132737  000100  001223          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
043544  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
043546  010037  043556                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
043552  004737  044226                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
043556  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
043560  132737  000040  001223  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
043566  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
043570  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
043572  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
043574  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
043576  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
043600  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
043604  000002                          RTI                     ;;RETURN
043606  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
043612  001430                          BEQ     8$
043614  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
043620  001006                          BNE     5$
043622  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
043624  104401                          TYPE                    ;;TYPE A CR AND LF
043626  001177                          $CRLF
043630  105037  043764                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
043634  000755                          BR      2$              ;;GET NEXT CHARACTER
043636  004737  043720          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
043642  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
043646  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
043650  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
043654  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
043660  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
043662  004737  043720                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
043666  105337  043764                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
043672  000770                          BR      7$              ;;LOOP
                                ;HORIZONTAL TAB PROCESSOR
043674  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
043700  004737  043720          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
043704  132737  000007  043764          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
043712  001372                          BNE     9$              ;;TAB STOP
043714  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
043716  000724                          BR      2$              ;;GET NEXT CHARACTER
043720  105777  135240          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
043724  100375                          BPL     $TYPEC
043726  116677  000002  135232          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
```

```
043734  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
043742  001003                          BNE     1$              ;;BRANCH IF NO
043744  105037  043764                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
043750  000406                          BR      $TYPEX          ;;EXIT
043752  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
043760  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
043762  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
043764  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
043766  000207                  $TYPEX: RTS     PC
  6189                          .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
043770                          $TYPDS:
043770  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
043772  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
043774  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
043776  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
044000  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
044002  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
044006  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
044012  100004                          BPL     1$              ;;BR IF INPUT IS POS.
044014  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
044016  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
044024  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
044026  012703  044204                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
044032  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
044036  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
044040  016001  044174                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
044044  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
044046  002402                          BLT     4$              ;;BR IF DONE
044050  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
044052  000774                          BR      3$
044054  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
044056  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
044060  001002                          BNE     5$              ;;FALL THROUGH IF 0
044062  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
044064  100407                          BMI     7$              ;;BR IF YES
044066  106316                  5$:     ASLB    (SP)            ;;MSD?
044070  103003                          BCC     6$              ;;BR IF NO
044072  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
044100  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
044104  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
044110  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
044112  005720                          TST     (R0)+           ;;JUST INCREMENTING
044114  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
044120  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
044122  003002                          BGT     8$              ;;GO TO EXIT
044124  010502                          MOV     R5,R2           ;;GET THE LSD
044126  000764                          BR      6$              ;;GO CHANGE TO ASCII
044130  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
```

```
        044132  100003                          BPL     9$                      ;;BR IF NO
        044134  116663  177777  177776          MOVB    -1(SP),-2(R3)           ;;YES--SET THE SIGN FOR TYPING
        044142  105013                  9$:     CLRB    (R3)                    ;;SET THE TERMINATOR
        044144  012605                          MOV     (SP)+,R5                ;;POP STACK INTO R5
        044146  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
        044150  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
        044152  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
        044154  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
        044156  104401  044204                  TYPE    ,$DBLK                  ;;NOW TYPE THE NUMBER
        044162  016666  000002  000004          MOV     2(SP),4(SP)             ;;ADJUST THE STACK
        044170  012616                          MOV     (SP)+,(SP)
        044172  000002                          RTI                             ;;RETURN TO USER
        044174  023420                  $DTBL:  10000.
        044176  001750                          1000.
        044200  000144                          100.
        044202  000012                          10.
        044204                          $DBLK:  .BLKW   4
  6190  044214  004737  044236                  JSR     PC,$ATY4                        ;;ONLY REPORT A FATAL ERROR
  6191                                  .SBTTL  APT COMMUNICATIONS ROUTINE
                                        ;;**************************************************************
        044220  112737  000001  044464  $ATY1:  MOVB    #1,$FFLG                ;;TO REPORT FATAL ERROR
        044226  112737  000001  044462  $ATY3:  MOVB    #1,$MFLG                ;;TO TYPE A MESSAGE
        044234  000403                          BR      $ATYC
        044236  112737  000001  044464  $ATY4:  MOVB    #1,$FFLG                ;;TO ONLY REPORT FATAL ERROR
        044244                          $ATYC:
        044244  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
        044246  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
        044250  105737  044462                  TSTB    $MFLG                   ;;SHOULD TYPE A MESSAGE?
        044254  001450                          BEQ     5$                      ;;IF NOT:  BR
        044256  122737  000001  001222          CMPB    #APTENV,$ENV            ;;OPERATING UNDER APT?
        044264  001031                          BNE     3$                      ;;IF NOT:  BR
        044266  132737  000100  001223          BITB    #APTSPOOL,$ENVM         ;;SHOULD SPOOL MESSAGES?
        044274  001425                          BEQ     3$                      ;;IF NOT:  BR
        044276  017600  000004                  MOV     @4(SP),R0               ;;GET MESSAGE ADDR.
        044302  062766  000002  000004          ADD     #2,4(SP)                        ;;BUMP RETURN ADDR.
        044310  005737  001202          1$:     TST     $MSGTYPE                ;;SEE IF DONE W/ LAST XMISSION?
        044314  001375                          BNE     1$                      ;;IF NOT:  WAIT
        044316  010037  001216                  MOV     R0,$MSGAD               ;;PUT ADDR IN MAILBOX
        044322  105720                  2$:     TSTB    (R0)+                   ;;FIND END OF MESSAGE
        044324  001376                          BNE     2$
        044326  163700  001216                  SUB     $MSGAD,R0               ;;SUB START OF MESSAGE
        044332  006200                          ASR     R0                      ;;GET MESSAGE LNGTH IN WORDS
        044334  010037  001220                  MOV     R0,$MSGLGT              ;;PUT LENGTH IN MAILBOX
        044340  012737  000004  001202          MOV     #4,$MSGTYPE             ;;TELL APT TO TAKE MSG.
        044346  000413                          BR      5$
        044350  017637  000004  044374  3$:     MOV     @4(SP),4$               ;;PUT MSG ADDR IN JSR LINKAGE
        044356  062766  000002  000004          ADD     #2,4(SP)                        ;;BUMP RETURN ADDRESS
        044364  013746  177776                  MOV     177776,-(SP)            ;;PUSH 177776 ON STACK
        044370  004737  043506                  JSR     PC,$TYPE                ;;CALL TYPE MACRO
        044374  000000                  4$:     .WORD   0
        044376                          5$:
        044376  105737  044464          10$:    TSTB    $FFLG                   ;;SHOULD REPORT FATAL ERROR?
        044402  001416                          BEQ     12$                     ;;IF NOT:  BR
        044404  005737  001222                  TST     $ENV                    ;;RUNNING UNDER APT?
        044410  001413                          BEQ     12$                     ;;IF NOT:  BR
        044412  005737  001202          11$:    TST     $MSGTYPE                ;;FINISHED LAST MESSAGE?
        044416  001375                          BNE     11$                     ;;IF NOT:  WAIT
```

```
044420  017637  000004  001204          MOV     @4(SP),$FATAL   ;;GET ERROR #
044426  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
044434  005237  001202                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
044440  105037  044464          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
044444  105037  044463                  CLRB    $LFLG           ;;CLEAR LOG FLAG
044450  105037  044462                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
044454  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
044456  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
044460  000207                          RTS     PC              ;;RETURN
044462     000                  $MFLG:  .BYTE   0               ;;MESSG. FLAG
044463     000                  $LFLG:  .BYTE   0               ;;LOG FLAG
044464     000                  $FFLG:  .BYTE   0               ;;FATAL FLAG
                                        .EVEN
             000200                     APTSIZE=200
             000001                     APTENV=001
             000100                     APTSPOOL=100
             000040                     APTCSUP=040
6192                                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
                                        ;;**************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$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
044466  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
044472  116637  000001  044711          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
044500  112637  044713                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
044504  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
044510  000406                          BR      $TYPON
044512  112737  000001  044711  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
044520  112737  000006  044713          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
044526  112737  000005  044710  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
044534  010346                          MOV     R3,-(SP)        ;;SAVE R3
044536  010446                          MOV     R4,-(SP)        ;;SAVE R4
044540  010546                          MOV     R5,-(SP)        ;;SAVE R5
044542  113704  044713                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
044546  005404                          NEG     R4
044550  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
044554  110437  044712                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
044560  113704  044711                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
044564  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
```

```
044570  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
044572  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
044574  000404                          BR      3$              ;;GO DO MSB
044576  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
044600  006105                          ROL     R5
044602  006105                          ROL     R5
044604  010503                          MOV     R5,R3
044606  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
044610  105337  044712                  DECB    $OMODE          ;;TYPE THIS DIGIT?
044614  100016                          BPL     7$              ;;BR IF NO
044616  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
044622  001002                          BNE     4$              ;;TEST FOR 0
044624  005704                          TST     R4              ;;SUPPRESS THIS 0?
044626  001403                          BEQ     5$              ;;BR IF YES
044630  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
044632  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
044636  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
044642  110337  044706                  MOVB    R3,8$           ;;SAVE FOR TYPING
044646  104401  044706                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
044652  105337  044710          7$:     DECB    $OCNT           ;;COUNT BY 1
044656  003347                          BGT     2$              ;;BR IF MORE TO DO
044660  002402                          BLT     6$              ;;BR IF DONE
044662  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
044664  000744                          BR      2$              ;;GO DO THE LAST DIGIT
044666  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
044670  012604                          MOV     (SP)+,R4        ;;RESTORE R4
044672  012603                          MOV     (SP)+,R3        ;;RESTORE R3
044674  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
044702  012616                          MOV     (SP)+,(SP)
044704  000002                          RTI                     ;;RETURN
044706     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
044707     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
044710     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
044711     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
044712  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
6193                                    .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
044714                                  $SAVREG:
044714  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
044716  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
044720  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
044722  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
044724  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
044726  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
```

```
        044730  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
        044734  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
        044740  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
        044744  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
        044750  000002                          RTI
                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
        044752                          $RESREG:
        044752  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
        044756  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
        044762  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
        044766  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
        044772  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
        044774  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
        044776  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
        045000  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
        045002  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
        045004  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
        045006  000002                          RTI
  6194                                  .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.
        045010  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
        045012  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
        045016  005740                          TST     -(R0)           ;;BACKUP BY 2
        045020  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
        045022  006300                          ASL     R0              ;;POSITION FOR INDEXING
        045024  016000  045044                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
        045030  000200                          RTS     R0              ;;GO TO ROUTINE
                                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
        045032  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
        045034  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
        045042  000007                          RTI                     ;;RESTORE THE PSW
                                        .SBTTL  TRAP TABLE
                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE "TRAP" INSTRUCTION.
                                        ;       ROUTINE
                                        ;       -------
        045044  045032                  $TRPAD: .WORD   $TRAP2
        045046  043506                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
        045050  044512                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        045052  044466                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
        045054  044526                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
        045056  043770                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
        045060  042540                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
        045062  042660                          $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
        045064  043032                          $RDOCT  ;;CALL=RDOCT    TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
        045066  044714                          $SAVREG ;;CALL=SAVREG   TRAP+11(104411) SAVE R0-R5 ROUTINE
        045070  044752                          $RESREG ;;CALL=RESREG   TRAP+12(104412) RESTORE R0-R5 ROUTINE
  6145                                  .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;;**************************************************************
                                        ;POWER DOWN ROUTINE
        045[illegible]                  $PWRDN: MOV     #$[illegible],@#P[illegible] ;;SET FOR FAST UP
```

```
.MAIN.  MACRO M1110  20-FEB-79 11:22  PAGE 98-1
POWER DOWN AND UP ROUTINES                                                    SEQ 0170

045100  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
045106  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
045110  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
045112  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
045114  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
045116  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
045120  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
045122  017746  134026                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
045126  010637  045236                  MOV     SP,$SAVR6       ;;SAVE SP
045132  012737  045144  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
045140  000000                          HALT
045142  000776                          BR      .-2             ;;HANG UP
                        ;;**************************************************************
                        ;POWER UP ROUTINE
045144  012737  045232  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
045152  013706  045236                  MOV     $SAVR6,SP       ;;GET SP
045156  005037  045236                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
045162  005237  045236          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
045166  001375                          BNE     1$              ;;OF  WORD
045170  012677  133760                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
045174  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
045176  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
045200  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
045202  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
045204  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
045206  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
045210  012737  045072  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
045216  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
045224  104401                          TYPE                    ;REPORT THE POWER FAILURE
045226  045240          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
045230  000002                          RTI
045232  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
045234  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
045236  000000          $SAVR6: 0                               ;;PUT THE SP HERE
045240     015     012     120  $POWER: .ASCIZ  <15><12>"POWER"
045243     117     127     105
045246     122     000

                                .EVEN
```

```
                                        .NLIST BEX
6199 045250  103  102  111  EM1:    .ASCIZ  /CBIT NOT CLEARING IOCM/
6200 045277  111  117  103  EM2:    .ASCII  /IOCM TEST/<15><12>
6201 045312  103  123  122  EM2Y:   .ASCII  /CSR DATA ERROR WHEN TESTING /
6202 045346  040  040  102  EM2X:   .ASCIZ  /  BIT  /<15><12>
6203 045360  124  123  124  DH100:  .ASCIZ  /TST#   PC      MODULE  ADDR    GDDAT   BDDAT   ITER/
6204 045425  124  123  124  DH101:  .ASCIZ  /TST#   PC      MODULE  ADDRESS ITER/
6205 045461  120  103  011  DH1:    .ASCIZ  /PC     TST#    GDDAT   BDDAT   PASS/
6206 045512  111  117  103  EM3:    .ASCIZ  /IOCM IS NOT RESPONDING/<15><12>
6207 045543  104  102  125  EM4:    .ASCIZ  /DBUS BIT IS FAILING/<15><12>
6208 045571  116  117  040  EM5:    .ASCIZ  /NO INTERRUPT /<15><12>
6209 045611  120  103  011  DH5:    .ASCIZ  /PC     TST#    MODULE  ADDR    PASS/
6210 045644  127  122  117  EM6:    .ASCIZ  /WRONG PATTERN IN IAR AFTER INTERRUPT/<15><12>
6211 045713                  EM17:
6212 045713  104  101  124  EM7:    .ASCIZ  /DATA ERROR/<15><12>
6213 045730  120  103  011  DH7:    .ASCIZ  /PC     MODULE  ADDRESS TST#    GDDAT   BDDAT   PASS/
6214 046000  124  105  123  EM10:   .ASCIZ  /TEST ABORTED, IOCM ERROR/<15><12>
6215 046033  104  101  124  EM11:   .ASCIZ  /DATA NOT CLEAR AFTER 10 SEC/<15><12>
6216 046071  111  116  124  EM12:   .ASCIZ  /INTERRUPT TOO LATE   /<15><12>
6217 046121  122  111  106  EM13:   .ASCIZ  /RIF BIT NOT CLEARING COS REGISTERS /<15><12>
6218 046167  116  117  040  EM14:   .ASCIZ  /NO INTERRUPT IN MAINTENANCE MODE/<15><12>
6219 046232  122  111  106  EM15:   .ASCIZ  /RIF BIT ISN'T CLEARING INTERRUPT/<15><12>
6220 046275  103  114  105  EM16:   .ASCIZ  /CLEARING DBIT ISN'T CLEARING CSR/<15><12>
6221 046340  103  117  123  EM20:   .ASCIZ  /COS REGISTER DATA ERROR/<15><12>
6222 046372  114  117  117  EM21:   .ASCIZ  /LOOP ERROR ON 6010-5010 OR 5011/<15><12>
6223 046434  120  103  011  DH21:   .ASCIZ  /PC     GDDAT   BDDAT   PASS/
6224 046460  101  104  104  EM22:   .ASCIZ  /ADDRESS TEST WITH MBIT SET ISN'T WORKING /<15><12>
6225 046534  125  116  101  EM23:   .ASCIZ  /UNABLE TO CLEAR INTERRUPT /<15><12>
6226 046571  120  103  011  DH23:   .ASCIZ  /PC     TST#    IAR/
6227 046605  116  015  012  M1:     .ASCIZ  /N/<15><12>
6228 046611  131  015  012  M0:     .ASCIZ  /Y/<15><12>
6229 046615  122  000       M2:     .ASCIZ  /R/
6230 046617  107  040  000  M3:     .ASCIZ  /G /
6231 046622  124  040  000  M4:     .ASCIZ  /T /
6232 046625  104  040  000  M5:     .ASCIZ  /D /
6233 046630  115  040  000  M6:     .ASCIZ  /M /
6234 046633  105  040  000  M7:     .ASCIZ  /E /
6235 046636  011  000       M10:    .ASCIZ  /       /
6236 046640  040  120  101  M11:    .ASCIZ  / PASSES COMPLETED/<15><12>
6237 046664  115  117  104  EM24:   .ASCIZ  /MODULE UNDER TEST IS NOT RESPONDING/<15><12>
6238 046732  015  012  124  M13:    .ASCIZ  <15><12>/TEST OPTIONS:/<15><12>
6239 046754  123  040  040  M14:    .ASCIZ  /S  SYSTEM TEST/<15><12>
6240 046775  104  040  040  M15:    .ASCIZ  /D  DIGITAL MODULE/<15><12>
6241 047021  101  040  040  M16:    .ASCIZ  /A  ANALOG MODULE/<15><12>
6242 047044  115  040  040  M17:    .ASCIZ  /M  MAP OF DBUS INTERFACES/<15><12>
6243 047100  130  040  040  M18:    .ASCIZ  /X  EXERCIZER/<15><12>
6244 047117  111  040  040  M19:    .ASCIZ  /I  IOCM TEST/<15><12>
6245 047136  040  040  040  M20:    .ASCIZ  /   ?/<15><12>
6246 047145  124  040  040  M21:    .ASCIZ  /T  SET SWREG/<15><12>
6247 047164  061  060  060  M22:    .ASCII  /100000   HALT ON ERROR/<15><12>
6248 047214  064  060  060         .ASCII  /40000    LOOP ON TEST/<15><12>
6249 047243  062  060  060         .ASCIZ  /20000    INHIBIT ERROR & EOP PRINT/<15><12>
6250 047310  061  060  060         .ASCIZ  /10000    LOOP ON ERROR/<15><12>
6251 047341  015  012  123  M23:    .ASCIZ  <15><12>/SWREG =/
6252 047353  106  040  040  M24:    .ASCIZ  /F  FIELD TEST/<15><12>
6253 047373  127  040  040  M25:    .ASCIZ  /W  LOOP TEST/<15><12>
6254 047412  114  040  040  M26:    .ASCIZ  /L  LOAD ITERATION COUNT/<15><12>
```

```
6255 047444  116  125  115  M27:    .ASCIZ  /NUMBER OF ITERATIONS(OCT) - /
6256 047501  101  130  130  M28:    .ASCIZ  /AXXX  ANALOG MODULE TEST (A014,A630)/<15><12>
6257 047550  114  111  116  EM25:   .ASCIZ  /LINE CLOCK IS NOT INTERRUPTING/<15><12>
6258 047611  015  012  102  M30:    .ASCII  <15><12>/BEFORE RUNNING THIS TEST MAKE SURE CUSTOMER WIRES ARE DISCONNECTED/
6259 047715  015  012  127          .ASCIZ  <15><12>/WHEN READY TYPE CR/
6260 047742  015  012  123  M84:    .ASCIZ <15><12> /SELECTED MUX DIDN'T RESPOND/
6261 050000  015  012  124  EM54:   .ASCIZ  <15><12>/TOO MANY ERRORS, TEST ABORTED/<15><12>
6262 050042  015  012  114  EM53:   .ASCII  <15><12>/LINEARITY TEST ERROR/<15><12>
6263 050072  105  122  122          .ASCIZ  /ERRPC  ADDRESS TSTPNT  MIN     MAX     WAS (# OF CONVERS)/<15><12>
6264 050154  015  012  115  MASS55: .ASCII  <15><12>/MONOTONICITY ERROR BY TWO BITS OR MORE/<15><12>
6265 050226  105  122  122          .ASCIZ  /ERRPC  ADDRESS GDDAT   BDDAT   RAMP    (UP=0,DOWN=1)/<15><12>
6266 050305  015  012  105  EM55:   .ASCIZ  <15><12>/ERROR BIT SET AFTER SETTING CHANNEL REG./
6267 050360  015  012  124  EM56:   .ASCIZ  <15><12>%TEST RAMP IN A/D ISN'T WORKING%
6268 050421  126  105  122  MASS51: .ASCIZ  /VERIFY IT BY RUNNING LOGIC TEST/<15><12>
6269 050463  015  012  125  EM57:   .ASCIZ  <15><12>/UNKNOWN GENERIC CODE/
6270 050512  105  122  122  DH57:   .ASCIZ/ERRPC    ADDRESS GCODE/
6271 050536  015  012  115  EM60:   .ASCIZ  <15><12>/MUX SELECT REGISTER ERROR/
6272 050572  015  012  127  EM61:   .ASCIZ  <15><12>/WRONG GENERIC CODE WITH MUX SELECTED/
6273 050641  015  012  123  EM63:   .ASCIZ  <15><12>/SELECTED MUX DID'T RESPOND/
6274 050676  105  122  122  DH61:   .ASCIZ/ERRPC    MODULE  ADDRESS MUX#    TST#    GDDAT   BDDAT/
6275 050751  105  122  122  DH62:   .ASCIZ/ERRPC    MODULE  ADDRESS MUX#    TST#/
6276 051010  015  012  122  EM62:   .ASCIZ  <15><12>/RIF BIT NOT CLEARING ERROR BIT/
6277 051051  015  012  115  EM73:   .ASCIZ  <15><12>/MUX REGISTER NOT CLEAR AFTER CONVERSION/
6278 051123  111  116  124  EM74:   .ASCIZ  /INTERRUPT WON'T CLEAR/<12><15>
6279 051153  103  102  111  EM75:   .ASCIZ  /CBIT FAILS TO CLEAR MODULE/<12><15>
6280 051210  103  117  125  EM76:   .ASCIZ  %COUNTER R/W FAILURE%<12><15>
6281 051236  124  102  111  EM77:   .ASCIZ  /TBIT FAILS TO INITIALIZE MODULE/<12><15>
6282 051300  103   '7  125  EM100:  .ASCIZ  /COUNTER FAILS TO HALT AFTER INTERRUPT/<12><15>
6283 051350  103  117  125  EM101:  .ASCIZ  /COUNTER FAILS TO COUNT/<12><15>
6284 051401  127  122  117  EM102:  .ASCIZ  /WRONG COUNT IN COUNTER AFTER TRANSITION/<12><15>
6285 051453  116  117  040  EM103:  .ASCIZ  /NO INTERRUPT/<12><15>
6286 051472  122  111  106  EM104:  .ASCIZ  /RIF BIT NOT CLEARING INTERRUPT/<12><15>
6287 051533  105  122  122  EM105:  .ASCIZ  /ERROR MESSAGE STUB/<15><12>
6288 051560  103  117  125  EM106:  .ASCIZ  /COUNTER NOT AT INITIALIZED FREQ/<12><15>
6289 051622  124  111  115  EM107:  .ASCIZ  /TIME BASE ERROR/<15><12>
6290 051644  125  116  105  EM110:  .ASCIZ /UNEXPECTED COUNTER INTERRUPT/ <15><12>
6291 051703  115  117  104  EM111:  .ASCIZ  /MODULE READ-WRITE FAILURE/<12><15>
6292 051737  115  117  104  EM112:  .ASCIZ  /MODULE SELF-CLEAR ERROR/<12><15>
6293 051771  111  116  103  EM113:  .ASCIZ  /INCORRECT COUNT IN COUNTER/<12><15>
6294 052026  115  065  060  EM114:  .ASCIZ  /M5016 OVERFLOW FAILURE/<12><15>
6295 052057  117  126  105  EM115:  .ASCIZ  /OVERFLOW FAILS TO CLEAR/<12><15>
6296 052111  103  123  122  EM116:  .ASCIZ  /CSR CBIT FAILS TO CLEAR COUNTER/<12><15>
6297 052153  012  015  111  EM117:  .ASCIZ  <12><15>/ILLEGAL RADIX/<12><15>
6298 052175  015  012  000  MASS0:  .ASCIZ  <15><12>
6299 052200  101  104  104  MASS2:  .ASCIZ  /ADDRESS  /
6300 052212  040  040  040  MASS3:  .ASCIZ  /    M5010 NONISOLATED 32 BIT DC INPUT/<15><12>
6301 052261  040  040  040  MASS4:  .ASCIZ  /    M5011 NONISOLATED 16 BIT DC INPUT/<15><12>
6302 052330  040  040  040  MASS5:  .ASCIZ  /    M5012 ISOLATED 16 BIT DC INPUT/<15><12>
6303 052374  040  040  040  MASS6:  .ASCIZ  /    M5013 8 BIT AC INPUT/<15><12>
6304 052426  040  040  040  MASS7:  .ASCIZ  /    M6010 NONISOLATED 32 BIT DC OUTPUT/<15><12>
6305 052476  040  040  040  MASS8:  .ASCIZ  /    M6011 16 BIT ONESHOT OUTPUT/<15><12>
6306 052537  040  040  040  MASS9:  .ASCIZ  /    M6012 ISOLATED 8 BIT DC OUTPUT/<15><12>
6307 052603  040  040  040  MASS10: .ASCIZ  /    M6013 8 BIT AC OUTPUT/<15><12>
6308 052636  040  040  040  MASS11: .ASCIZ  /    UNKNOWN GENERIC CODE =  /
6309 052672  015  012  127  MASS12: .ASCIZ  <15><12>/WHICH MODULE OR TEST OPTION (H=HELP)?  /
6310 052744  015  012  124  MASS13: .ASCIZ  <15><12>/TYPE ADDRESS OF MUT    /
6311 052776  111  116  124  MASS14: .ASCIZ  /INTERRUPT TOO LATE/<15><12>
```

```
6312 053023  015  012  105  MASS15: .ASCIZ  <15><12>/END OF MODULE TEST/<15><12>
6313 053052  105  116  104  MASS16: .ASCIZ  /END OF LOOP TEST/<15><12>
6314 053075  105  116  104  MASS17: .ASCIZ  /END OF  AUTO TEST/<15><12>
6315 053121  040  040  040  MASS18: .ASCIZ  /   /<15><12>
6316 053127  124  131  120  MASS19: .ASCIZ  /TYPE ONE BYTE DATA PATTERN    /
6317 053166  124  131  120  MASS20: .ASCIZ  /TYPE CONTROL-C TO RETURN TO MONITOR/<15><12>
6318 053234  124  131  120  MASS21: .ASCIZ  /TYPE ADDRESS OF OUTPUT MUT/
6319 053267  124  131  120  MASS22: .ASCIZ  /TYPE ADDRESS OF INPUT MUT/
6320 053321  103  117  116  MASS23: .ASCIZ  /CONNECT OUTPUT TO INPUT MODULE/<15><12>
6321 053362  105  116  104  MASS25: .ASCIZ  /END OF LOOP TEST/<15><12>
6322 053405  127  122  117  MASS26: .ASCIZ  /WRONG MODULE-OUTPUT MUST BE M6010/<15><12>
6323 053451  072  011  000  MASS27: .ASCIZ  /:      /
6324 053454  127  122  117  MASS28: .ASCIZ  /WRONG MODULE-INPUT MUST BE EITHER M5010 OR M5011/<15><12>
6325 053537  040  040  040  MASS29: .ASCIZ  /   A630 FOUR CHANNEL DAC MODULE /<15><12>
6326 053602  040  040  040  MASS30: .ASCIZ  %   AD14 A/D CONVERTER -SINGLE ENDED%<15><12>
6327 053650  040  040  040  MASS31: .ASCIZ  %   AD14 A/D CONVERTER - DIFFERENTIAL MODE%<15><12>
6328 053724  011  115  125  MASS32: .ASCIZ  /       MUX# /
6329 053733  011  011  101  MASS33: .ASCIZ  /                A156 - SINGLE-ENDED/<15><12>
6330 053763  011  011  101  MASS34: .ASCIZ  /                A156 - DIFFERENTIAL MODE/<15><12>
6331 054020  011  011  101  MASS35: .ASCIZ  /                A157/<15><12>
6332 054031  015  012  104  MASS36: .ASCII  <15><12> /D - LOGIC TEST/<15><12>
6333 054053  103  040  055          .ASCII  /C - CALIBRATION/<15><12>
6334 054074  115  040  055          .ASCII  /M - MONOTONICITY TEST/<15><12>
6335 054123  130  040  055          .ASCII  /X - MUX TEST/<15><12>
6336 054141  114  040  055          .ASCIZ  /L - LINEARITY TEST/<15><12>
6337 054166  015  012  127  MASS37: .ASCIZ  <15><12>        /WHICH MUX # DO YOU WANT TO TEST? /
6338 054232  015  012  127  MASS38: .ASCIZ  <15><12>        %WHICH A/D CHANNEL YOU WANT TO CALIBRATE? %
6339 054306  015  012  124  MASS39: .ASCIZ  <15><12>        /TOO HIGH CHANNEL FOR DIFF MODE/<15><12>
6340 054351  015  012  124  MASS40: .ASCIZ  <15><12>        /TOO HIGH CHANNEL FOR SINGLE-ENDED MODE/<15><12>
6341 054424  015  012  127  MASS41: .ASCIZ  <15><12>/WHICH GAIN (1,10,20,50,100,200,1000) ?/
6342 054475  127  122  117  MASS42: .ASCIZ  /WRONG GAIN SELECTED/<15><12>
6343 054523  103  117  116  MASS43: .ASCIZ  /CONNECT VOLTAGE SOURCE  (CR)/
6344 054560  015  012  117  MASS44: .ASCIZ  <15><12>  /OCTAL    AVERAGE VOLTAGE(MV)/<15><12>
6345 054620  125  116  105  MASS45: .ASCIZ  /UNEXPECTED TIME-OUT TRAP-LAST PC BEFORE TRAP    /
6346 054700  015  012  127  MASS46: .ASCIZ  <15><12>        /WHICH TEST (D,C,M,L,X,H-HELP)? /
6347 054742  040  040  040  MASS48: .ASCIZ  /   M5012-YA 16 BIT TTL INPUT/<15><12>
6348 055001  040  040  040  MASS49: .ASCIZ  /   M6010-YA 32 BIT TTL OUTPUT/<15><12>
6349 055041  116  117  040  MASS50: .ASCIZ  %NO I/O MODULE PRESENT%<15><12>
6350 055071  040  040  040  MASS56: .ASCIZ   /   M5014 INPUT COUNTER/<15><12>
6351 055122  040  040  040  MASS52: .ASCIZ   /   M5016 QUAD 8 BIT COUNTER/<15><12>
6352 055160  040  040  040  MASS53: .ASCIZ   /   M6014 DUAL 16 BIT COUNTER/<15><12>
6353 055217  040  040  040  MASS54: .ASCIZ   /   G670 MOTHER BOARD/<15><12>
6354 055246  015  012  101  EM27:   .ASCIZ  <15><12>/A REGISTER DID NOT CLEAR ON INIT /
6355 055312  015  012  101  EM26:   .ASCIZ  <15><12>/A REGISTER DID NOT RESPOND WHEN ADDRESSED/
6356 055366  015  012  040  EM33:   .ASCIZ  <15><12>/ CHANNEL COMPARATOR ERROR/
6357 055422  124  123  124  DH110:  .ASCIZ  /TST#   PC       ADDRESS COUNTER MODULE  ITER/
6358 055466  124  123  124  DH102:  .ASCIZ  /TST#   PC       ADDRESS COUNTER GDDAT   BDDAT   ITER/
6359 055537  124  123  124  DH114:  .ASCIZ  /TST#   PC       ADDRESS COUNTER GDOVF   BDOVF   ITER/
6360 055610  015  012  101  DH33:   .ASCIZ  <15><12>/ADDRESS        CHNUM   STBITS  CH0     CH1     CH2     CH3/
6361 055657  015  012  115  EM34:   .ASCIZ  <15><12>/MORE THEN ONE CHANNEL FAILS DURING COMPARATOR TEST/
6362 055744  015  012  105  DH34:   .ASCIZ  <15><12>/ERRPC  MODULE  ADDRESS STBITS  CH0     CH1     CH2     CH3/
6363 056022  015  012  105  DH40:   .ASCIZ  <15><12>/ERRPC  TST#    GENERIC /
6364 056050  015  012  104  EM30:   .ASCIZ  <15><12>/DATA ERROR DURING REGISTER TEST/
6365 056112  015  012  105  DH30:   .ASCIZ  <15><12>/ERRPC  MODULE  GDDATA  BDDATA  REGADR /
6366 056157  123  124  101  EM35:   .ASCIZ      /STATUS REGISTER ERROR AFTER INITIALIZE/<15><12>
6367 056230  102  125  123  EM36:   .ASCIZ      /BUSY BIT NOT CLEAR AFTER CONVERSION/<15><12>
6368 056276  105  122  122  EM37:   .ASCIZ      /ERROR BIT SET AFTER CONVERSION/<15><12>
```

```
6369 056337      103      117      116  EM40:   .ASCIZ  /CONVERSION NOT DONE AFTER .5MSEC/<15><12>
6370 056402      116      117      040  EM41:   .ASCIZ  /NO INTERRUPT AFTER CONVERSION DONE/
6371 056445      103      110      101  EM42:   .ASCIZ  /CHANNEL SELECTION REGISTER ERROR/
6372 056506      105      122      122  EM43:   .ASCIZ  /ERROR BIT NOT SET AFTER SETTING WRONG GAIN/<15><12>
6373 056563      105      122      122  EM44:   .ASCIZ  /ERROR BIT WILL NOT INTERRUPT/<15><12>
6374 056622      105      122      122  EM45:   .ASCIZ  /ERROR BIT NOT SET AFTER WRONG GAIN/<15><12>
6375 056667      115      101      111  EM46:   .ASCIZ      /MAINTANCE REFERENCE VOLTAGE TEST FAILED/<15><12>
6376 056741      122      101      115  EM47:   .ASCIZ  /RAMP IS NOT WORKING/<15><12>
6377 056767      104      101      124  EM50:   .ASCIZ  /DATA IS NOT ZERO WITH D BIT SET/<15><12>
6378 057031      015      012      104  M80:    .ASCIZ  <15><12>/DBIT DOES NOT DISABLE ALL OUTPUTS /
6379 057076      015      012      101  M81:    .ASCIZ  <15><12>/ADJUSTMENT OF VOLTAGE OUTPUT/
6380 057135      015      012      104  M82:    .ASCIZ  <15><12>/DO YOU WANT TO ADJUST CURRENT OUTPUT (Y-N)?  /
6381 057215      015      012      105  M83:    .ASCIZ  <15><12>/END OF CALIBRATION/
6382 057242      012      015      125  M85:    .ASCIZ  <12><15>/UNEXPECTED TIME OUT/<12><15>
6383 057272      015      012      104  DANGC:  .ASCIZ <15><12> /DAC DID NOT RESPOND WITH GENERIC CODE 261/
6384 057346      015      012      107  GAINMG: .ASCII <15><12> /GAIN X INPUT VOLTAGE=VOLTAGE READING/
6385 057414      015      012      107          .ASCIZ <15><12> /GAIN = /
6386 057426      015      012      123  DANSEL: .ASCIZ <15><12> /SELECT TEST (A,T,D OR H=HELP)/
6387 057466      015      012      124  DACHN:  .ASCIZ <15><12> /TYPE CHANNEL # DESIRED (0,1,2,3)   /
6388 057534      015      012      101  TEXT:   .ASCII <15><12> /A - CALIBRATION TEST/
6389 057562      015      012      124          .ASCII <15><12> /T - INTERNAL COMPARATOR TEST/
6390 057620      015      012      104          .ASCIZ <15><12> /D - D-BIT TEST /
6391
6392 057642      015      012      104  CTXTV:  .ASCIZ  <15><12>/DO YOU WANT TO ADJ REFER. VOLTAGE (Y-N)?  /
6393
6394
6395 057717      015      012      122          .ASCIZ <15><12> /R66 - CALIB POT FOR CH1 CURRENT ZERO/<15><12>
6396 057770      015      012      122          .ASCII <15><12> /R80 - CALIB POT FOR CH2 CURRENT GAIN/
6397
6398 060036      015      012      123  CM1:    .ASCIZ <15><12> /STEP 1:        +10.240V  TOLERANCE 2MV (CR)/
6399 060105      015      012      123  CM2:    .ASCIZ <15><12> /STEP 2:   5.120V   TOLERANCE 2MV (CR)/
6400 060155      015      012      123  CM3:    .ASCIZ <15><12> /STEP 1:   0.000V   TOLERANCE 5MV (CR)/
6401 060225      015      012      123  CM4:    .ASCIZ <15><12> /STEP 2:   +10.230V   TOLERANCE 40MV (CR)/
6402
6403
6404 060300      015      012      123  CM7:    .ASCII <15><12> /STEP 2: CURRENT OFFSET ADJ.  9.8MV  TOLER. 5MV/
6405 060360      015      012      040          .ASCIZ <15><12> /        OR 19.5 MICRO-AMPS. TOLER. 10 MICRO-AMPS.  (CR)/
6406
6407 060452      015      012      123  CM9:    .ASCII <15><12> /STEP 1: CURRENT GAIN ADJ. TO +10.000V TOLER. 5MV/
6408 060534      015      012      040          .ASCIZ <15><12> /        OR 20.000MA TOLER. 10 MICRO-AMPS (CR)/
6409 060614      015      012      123  CM10:   .ASCII <15><12> /STEP 2: CURRENT OFFSET ADJ. TO +2.000V TOLER. 5MV/
6410 060677      015      012      040          .ASCIZ <15><12> /        OR 4.000 MA.TOLER.  10 MICRO AMPS (CR)/
6411 060760      015      012      110  CM11:   .ASCIZ  <15><12>/HAS CALIB BEEN RE-VERIFIED (Y-N) /
6412 061024      015      012      104  CM12:   .ASCIZ  <15><12>/DO YOU WISH TO CALIB THE 4-20 MA RANGE (Y-N)? /
6413 061105      015      012      101  EM31:   .ASCIZ  <15><12>/ALL COMPARATOR STATUS BITS SHOULD BE SET/
6414 061160      015      012      101  EM32:   .ASCIZ  <15><12>/ALL COMPARATOR STATUS BITS SHOULD BE CLEAR/
6415 061235      015      012      120  DH31:   .ASCIZ  <15><12>/PC#    ADDRESS EXPECT  WAS/
6416                                    .EVEN
6417 061266  001206   001132   002244  DT110:  .WORD   $TESTN, $ERRPC, TADDR,  CNUM,   $MUT,   $PASS,
6418 061304  001206   001132   002244  DT102:  .WORD   $TESTN, $ERRPC, TADDR,  CNUM,   $GDDAT, $BDDAT, $PASS,
6419 061324  001206   001132   002240  DT101:  .WORD   $TESTN, $ERRPC, $MUT,   TADDR,  $PASS,
6420 061340  001206   001132   002240  DT100:  .WORD   $TESTN, $ERRPC, $MUT,   TADDR,  $GDDAT, $BDDAT, $PASS,
6421 061360  001132   002240   002314  DT17:   .WORD   $ERRPC, $MUT,   TEMP1,  $TESTN, $GDDAT, $BDDAT, $PASS,
6422 061400  001132   002240   002252  DT20:   .WORD   $ERRPC, $MUT,   COSADR, $TESTN, $GDDAT, $BDDAT, $PASS,
6423 061420  001132   001206   003042  DT23:   .WORD   $ERRPC, $TESTN, TEMP3,
6424 061430  001132   002240   002250  DT7:    .WORD   $ERRPC, $MUT,   TBADDR, $TESTN, $GDDAT, $BDDAT, $PASS,
6425 061450  001132   001206   001140  DT1:    .WORD   $ERRPC, $TESTN, $GDDAT, $BDDAT, $PASS,
```

```
6426 061464  001132  001206  002240  DT5:    .WORD   $ERRPC, $TESTN, $MUT,   TADDR,  $PASS,
6427 061500  001132  001140  001142  DT21:   .WORD   $ERRPC, $GDDAT, $BDDAT, $PASS,
6428 061512  001132  002240  001140  DT30:   .WORD   $ERRPC, $MUT,   $GDDAT, $BDDAT, TEMP,
6429 061526  001132  002244  002356  DT33:   .WORD   $ERRPC, TADDR,  CHNUM,  TEMP3,  ACOUNT, BCOUNT, ACOUNT, BCOUNT,
6430 061550  001132  002240  002244  DT34:   .WORD   $ERRPC, $MUT,   TADDR,  TEMP3,  ACOUNT, BCOUNT, ACOUNT, BCOUNT,
6431 061572  001132  002244  001140  DT51:   .WORD   $ERRPC, TADDR,  $GDDAT, $BDDAT, LASTCN,
6432 061606  001132  002244  003026  DT52:   .WORD   $ERRPC, TADDR,  BLAST,  BCOUNT, CCOUNT, $BDDAT,
6433 061624  001132  003042  003040  DT57:   .WORD   $ERRPC, TEMP3,  TEMP2,
6434 061634  001132  002240  002250  DT61:   .WORD   $ERRPC, $MUT,   TBADDR, MXNUM,  $TESTN, $GDDAT, $BDDAT,
6435 061654  001132  002240  002250  DT62:   .WORD   $ERRPC, $MUT,   TBADDR, MXNUM,  $TESTN,
6436 061670  001132  002244  003042  DT31:   .WORD   $ERRPC, TADDR,  TEMP3,  TEMP,
6437 061702     000     000     000  DF102:  .BYTE   0,0,0,0,0,0,1
6438 061711     000     000     000  DF100:  .BYTE   0,0,0,0,0,1
6439 061717     000     000     000  DF101:  .BYTE   0,0,0,0,1
6440 061724     000     000     000  DF51:   .BYTE   0,0,0,0
6441 061730     000     000     001  DF52:   .BYTE   0,0,1,1,1
6442         000001                  .END
```

SYMBOL TABLE

SEQ 0176

```
ABASE = 000000        APTENV= 000001        CCOUNT  002430        DCAMON  026676        DT61    061634
ACDW1 = 000000        APTSIZ= 000200        CHKBYT  040730        DCHAN   002372        DT62    061654
ACDW2 = 000000        APTSPO= 000100        CHNSEL  034422        DCMN    027006        DT7     061430
ACNTH   003132        ASWREG= 000000        CHNUM   002356        DCOUNT  002434        EBIT  = 000100
ACNTL   003134        ATABL   001326        CHSEL   034254        DCOUT   026772        ECOUNT  002444
ACOUNT  002424        ATESTN= 000000        CH0     002360        DCTST   027766        EMTVEC= 000030
ACPUOP= 000000        ATOD    032324        CH1     002362        DDBIT   027252        EM1     045250
ADADDR  042016        AUNIT = 000000        CH2     002364        DDISP = 177570        EM10    046000
ADCALB  032732        AUSWR = 000000        CH3     002366        DECCHN  031722        EM100   051300
ADDW0 = 000000        AUTO    006002        CH4     002370        DF100   061711        EM101   051350
ADDW1 = 000000        AVECT1= 000000        CKICT   031454        DF101   061717        EM102   051401
ADDW10= 000000        AVECT2= 000000        CKRWSX  024312        DF102   061702        EM103   051453
ADDW11= 000000        AVRSAV  003162        CKSRCX  024374        DF51    061724        EM104   051472
ADDW12= 000000        AVRTBL  003164        CK4CLX  024236        DF52    061730        EM105   051533
ADDW13= 000000        BASE    002276        CK4CXX  026236        DH1     045461        EM106   051560
ADDW14= 000000        BCNTH   003136        CK4INX  025166        DH100   045360        EM107   051622
ADDW15= 000000        BCNTL   003140        CLK     002300        DH101   045425        EM11    046033
ADDW2 = 000000        BCOUNT  002426        CLKADR  002260        DH102   055466        EM110   051644
ADDW3 = 000000        BEG     036470        CLKVC   002322        DH110   055422        EM111   051703
ADDW4 = 000000        BINCNX  024576        CLKVCA  002324        DH114   055537        EM112   051737
ADDW5 = 000000        BITSET  040450        CLRINT  040564        DH21    046434        EM113   051771
ADDW6 = 000000        BIT0  = 000001        CM1     060036        DH23    046571        EM114   052026
ADDW7 = 000000        BIT00 = 000001        CM10    060614        DH30    056112        EM115   052057
ADDW8 = 000000        BIT01 = 000002        CM11    060760        DH31    061235        EM116   052111
ADDW9 = 000000        BIT02 = 000004        CM12    061024        DH33    055610        EM117   052153
ADEVCT= 000000        BIT03 = 000010        CM2     060105        DH34    055744        EM12    046071
ADEVM = 000000        BIT04 = 000020        CM3     060155        DH40    056022        EM13    046121
ADGAN   034522        BIT05 = 000040        CM4     060225        DH5     045611        EM14    046167
ADLOG   033624        BIT06 = 000100        CM7     060300        DH57    050512        EM15    046232
ADMON   032424        BIT07 = 000200        CM9     060452        DH61    050676        EM16    046275
ADRET   032374        BIT08 = 000400        CNTRC   040646        DH62    050751        EM17    045713
ADTBIT  034610        BIT09 = 001000        CNUM    003146        DH7     045730        EM2     045277
ADTST   032236        BIT1  = 000002        CONV    002436        DIGIT   005416        EM2X    045546
AENV  = 000000        BIT10 = 002000        CONVCT  002442        DISPLA  001156        EM2Y    045552
AENVM = 000000        BIT11 = 004000        CONV?   037776        DISPRE  000174        EM20    046340
AFATAL= 000000        BIT12 = 010000        COSADR  002252        DMUT    002254        EM21    046372
AFLAG   002312        BIT13 = 020000        COUNT   040556        DR5016  024034        EM22    046460
ALLCLX  026574        BIT14 = 040000        CR    = 000015        DSWCH   024046        EM23    046534
AMADR1= 000000        BIT15 = 100000        CRLF  = 000200        DSWR  = 177570        EM24    046664
AMADR2= 000000        BIT2  = 000004        CRTST   041026        DTST    030010        EM25    047550
AMADR3= 000000        BIT3  = 000010        CSR     002262        DT1     061450        EM26    055312
AMADR4= 000000        BIT4  = 000020        CTASCI  024056        DT100   061340        EM27    055246
AMAMS1= 000000        BIT5  = 000040        CTRSH   003142        DT101   061324        EM3     045512
AMAMS2= 000000        BIT6  = 000100        CTRSL   003144        DT102   061304        EM30    056050
AMAMS3= 000000        BIT7  = 000200        CTXTV   057642        DT110   061266        EM31    061105
AMAMS4= 000000        BIT8  = 000400        DACHN   057466        DT17    061360        EM32    061160
AMSGAD= 000000        BIT9  = 001000        DACMON  026762        DT20    061400        EM33    055366
AMSGLG= 000000        BLANK2  024140        DACSTR  026732        DT21    061500        EM34    055657
AMSGTY= 000000        BLAST   003026        DACTST  027670        DT23    061420        EM35    056157
AMTYP1= 000000        BPTVEC= 000014        DAGTST  031656        DT30    061512        EM36    056230
AMTYP2= 000000        BSYCON  033756        DALLY   032102        DT31    061670        EM37    056276
AMTYP3= 000000        BTABL   001566        DANGC   057272        DT33    061526        EM4     045543
AMTYP4= 000000        BYTNUM  002242        DANSEL  057426        DT34    061550        EM40    056337
ANSW    002432        CBIT  = 000002        DATALO  002440        DT5     061464        EM41    056402
APASS = 000000        CBITCL  023672        DATST   027674        DT51    061572        EM42    056445
APRIOR= 000000        CBITCM  021370        DBIT  = 000020        DT52    061606        EM43    056506
APTCSU= 000040        CCNTCX  024430        DBUFF   002226        DT57    061624        EM44    056563
```

.MAIN. MACRO M1110 20-FEB-79 11:22 PAGE 100-6
SYMBOL TABLE

SEQ 0177

```
EM45    056622    INTHL1  022310    MASS41  054424    M5013   011544    SAVREG= 104411
EM46    056667    INTHL2  017176    MASS42  054475    M5014   016402    SCOPE = 000004
EM47    056741    INTLD   031366    MASS43  054523    M5014R  021534    SDELAY  026650
EM5     045571    IOCM    010142    MASS44  054560    M5016   024144    SETBYT  040332
EM50    056767    IOTVEC= 000020    MASS45  054620    M6      046630    SETCLK  004676
EM53    050042    KBINT   040722    MASS46  054700    M6010   012006    SETPTN  041052
EM54    050000    KBVEC   002354    MASS48  054742    M6011   012372    SETRAM  037746
EM55    050305    KLEER   032024    MASS49  055001    M6011A  012564    SR5016  024032
EM56    050360    KOUNT   002422    MASS5   052330    M6011B  012656    STACK = 001100
EM57    050463    LAST    003030    MASS50  055041    M6012   012200    START   004364
EM6     045644    LASTCN  003032    MASS51  050421    M6013   012200    STAT1   003052
EM60    050536    LBYTE   003056    MASS52  055122    M6014   021546    STAT2   003054
EM61    050572    LDATA0  002402    MASS53  055160    M7      046633    STCO    036504
EM62    051010    LDATA1  002406    MASS54  055217    M80     057031    STKLMT= 177774
EM63    050641    LDATA2  002412    MASS55  050154    M81     057076    STSTIH  023210
EM7     045713    LDATA3  002416    MASS56  055071    M82     057135    STSTI2  020112
EM73    051051    LF    = 000012    MASS6   052374    M83     057215    ST1SAV  003126
EM74    051123    LINEAR  035252    MASS7   052426    M84     047742    ST2SAV  003130
EM75    051153    LOGIC   006660    MASS8   052476    M85     057242    SWLOOP  042224
EM76    051210    LONGIN  002256    MASS9   052537    NOCLK   040554    SWR     001154
EM77    051236    LOOP    006740    MAXRDX  026306    NORAMP  002302    SWREG   000176
ERRVEC= 000004    LOPTST  041240    MBIT  = 000040    NUM     003150    SWREGS  003754
EXERC   032124    MAPE    007370    MOD     002326    OUT     037046    SW0   = 000001
FBIT  = 000200    MASS0   052175    MODUL   002150    OVFS    024036    SW00  = 000001
FHOUSX  026200    MASS10  052603    MONDAT  041122    PASCNT  002320    SW01  = 000002
FIELD   007170    MASS11  052636    MONIT   005106    PATT    002232    SW02  = 000004
F5010   032130    MASS12  052672    MUT     002100    PIRQ  = 177772    SW03  = 000010
F5011   032130    MASS13  052744    MUX1    037060    PIRQVE= 000240    SW04  = 000020
F5012   032142    MASS14  052776    MXNUM   002224    PR0   = 000000    SW05  = 000040
F5013   032154    MASS15  053023    M0      046611    PR1   = 000040    SW06  = 000100
F6010   032166    MASS16  053052    M1      046605    PR2   = 000100    SW07  = 000200
F6011   032222    MASS17  053075    M10     046636    PR3   = 000140    SW08  = 000400
F6012   032206    MASS18  053121    M11     046640    PR4   = 000200    SW09  = 001000
F6013   032206    MASS19  053127    M13     046732    PR5   = 000240    SW1   = 000002
GAIN    003104    MASS2   052200    M14     046754    PR6   = 000300    SW10  = 002000
GAINMG  057346    MASS20  053166    M15     046775    PR7   = 000340    SW11  = 004000
GAINTB  003062    MASS21  053234    M16     047021    PS    = 177776    SW12  = 010000
GBIT  = 000004    MASS22  053267    M17     047044    PSW   = 177776    SW13  = 020000
GBITE   003050    MASS23  053321    M18     047100    PWRVEC= 000024    SW14  = 040000
GCODE   042172    MASS25  053362    M19     047117    RAMPST  036414    SW15  = 100000
GENCOD  04037     MASS26  053405    M2      046615    RAMP1   002446    SW2   = 000004
GENER   002026    MASS27  053451    M20     047136    RAMP2   002566    SW3   = 000010
GODN    031606    MASS28  053454    M21     047145    RAMP3   002706    SW4   = 000020
GOUP    031570    MASS29  053537    M22     047164    RBIT  = 000001    SW5   = 000040
GTADRS  027210    MASS3   052212    M23     047341    RDCHR = 104406    SW6   = 000100
G670    014176    MASS30  053602    M24     047353    RDLIN = 104407    SW7   = 000200
HBYTE   003060    MASS31  053650    M25     047373    RDOCT = 104410    SW8   = 000400
HDATA0  002404    MASS32  053724    M26     047412    RERROR  002306    SW9   = 001000
HDATA1  002410    MASS33  053733    M27     047444    RESREG= 104412    TABLE   040420
HDATA2  002414    MASS34  053763    M28     047501    RESVEC= 000010    TADDR   002244
HDATA3  002420    MASS35  054020    M3      046617    RETURN  003160    TADDR1  002246
HT    = 000011    MASS36  054031    M30     047611    ROTDAT  031624    TBADDR  002250
IAR     002264    MASS37  054166    M4      046622    ROTFLG  003036    TBIT  = 000010
ICHAN   002400    MASS38  054232    M5      046625    ROTPAT  003034    TBITVE= 000014
INTCOM  030430    MASS39  054306    M5010   011242    RUMP    036322    TEMP    002304
INIDAT  002316    MASS4   052261    M5011   012670    R6    =%000006    TEMPB   003156
INTHLT  014724    MASS40  054351    M5012   013520    R7    =%000007    TEMP1   002314
```

.MAIN. MACRO M1110 20-FEB-79 11:22 PAGE 100-7
SYMBOL TABLE

SEQ 0178

```
TEMP2   003040        TST4    010406        $DDW10  001312        $ILLUP  045232        $RESRE  044752
TEMP3   003042        TST5    010536        $DDW11  001314        $INTAG  001151        $RTNAD  006736
TEMP4   003044        TST6    010606        $DDW12  001316        $ITEMB  001130        $SAVRE  044714
TEMP5   003046        TST7    010652        $DDW13  001320        $LF     001200        $SAVR6  045236
TEXT    057534        TYPDS = 104405        $DDW14  001322        $LFLG   044463        $SCOPE  042322
TINIMD  017022        TYPDSW  026022        $DDW15  001324        $LPADR  001122        $SETUP= 000027
TINIMO  022150        TYPE  = 104401        $DDW2   001272        $LPERR  001124        $STUP = 177777
TINIZ   014564        TYPOC = 104402        $DDW3   001274        $MADR1  001234        $SVLAD  042412
TIOCM   010210        TYPON = 104404        $DDW4   001276        $MADR2  001240        $SWR  = 121000
TKVEC = 000060        TYPOPT  005036        $DDW5   001300        $MADR3  001244        $SWREG  001224
TMOVEC  005404        TYPOS = 104403        $DDW6   001302        $MADR4  001250        $SWRMK= 000000
TPVEC = 000064        UPREG   027650        $DDW7   001304        $MAII   001202        $SWO8T  042462
TRANPT  015424        VAR1    003152        $DDW8   001306        $MAMS1  001232        $TESTN  001206
TRANP1  023002        VCHAN   002376        $DDW9   001310        $MAMS2  001236        $TKB    001162
TRANP2  017714        VECTO   002266        $DEVCT  001212        $MAMS3  001242        $TKS    001160
TRAPVE= 000034        VECTOA  002270        $DEVM   001260        $MAMS4  001246        $TN   = 000030
TRTVEC= 000014        VECT1   002272        $DOAGN  006734        $MBADR  001102        $TPB    001166
TSECTB  020762        VECT1A  002274        $DTBL   044174        $MFLG   044462        $TPFLG  001173
TSTADR  027304        XCHAN   002374        $ENDAD  006724        $MNEW   043021        $TPS    001164
TSTBYT  040166        XLOOP   003154        $ENDCT  006710        $MSGAD  001216        $TRAP   045010
TSTONE  040250        XXDP    002236        $ENV    001222        $MSGLG  001220        $TRAP2  045032
TSTPAT  014410        XXX     002310        $ENVM   001223        $MSGTY  001202        $TRP  = 000013
TSTPT1  021760        YLOOP   002222        $EOP    006660        $MSWR   043010        $TRPAD  045044
TSTPT2  016624        $APTHD  001100        $EOPCT  006702        $MTYP1  001233        $TSTM   001104
TSTRGS  027376        $ATYC   044244        $ERFLG  001117        $MTYP2  001237        $TSTNM  001116
TST1    010210        $ATY1   044220        $ERMAX  001131        $MTYP3  001243        $TTYIN  042766
TST10   010716        $ATY3   044226        $ERROR  043172        $MTYP4  001247        $TYPDS  043770
TST11   011012        $ATY4   044236        $ERRPC  001132        $MUT    002240        $TYPE   043506
TST12   011242        $AUTOB  001150        $ERRTB  003174        $NULL   001170        $TYPEC  043720
TST13   011344        $BASE   001256        $ERRTY  043352        $NWTST= 000000        $TYPEX  043766
TST14   012006        $BDADR  001136        $ERTTL  001126        $OCNT   044710        $TYPOC  044512
TST15   012200        $BDDAT  001142        $ESCAP  001174        $OMODE  044712        $TYPON  044526
TST16   012372        $CDW1   001262        $ETABL  001222        $OVER   042446        $TYPOS  044466
TST17   012670        $CDW2   001264        $ETEND  001326        $PASS   001210        $UNIT   001214
TST2    010276        $CHARC  043764        $FATAL  001204        $PASTM  001106        $UNITM  001110
TST20   013520        $CMTAG  001114        $FFLG   044464        $POWER  045240        $USWR   001226
TST21   014176        $CM3  = 000000        $FILLC  001172        $PWRDN  045072        $VECT1  001252
TST22   016402        $CNTLG  043003        $FILLS  001171        $PWRMG  045226        $VECT2  001254
TST23   021546        $CNTLU  042776        $GDADR  001134        $PWRUP  045144        $XTSTR  042322
TST24   024144        $CPUOP  001230        $GDDAT  001140        $QUES   001176        $$GET4= 000000
TST25   027376        $CRLF   001177        $GET42  006714        $RDCHR  042540        $$SW08= 000030
TST26   030430        $DBLK   044204        $HIBTS  001100        $RDLIN  042660        $OFILL  044711
TST27   033624        $DDW0   001266        $HIOCT  043170        $RDOCT  043052        .$X   = 001100
TST3    010342        $DDW1   001270        $ICNT   001120        $RDSZ = 000010

. ABS.  061735      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  48928 WORDS  ( 192 PAGES)
DYNAMIC MEMORY:  20508 WORDS  ( 78 PAGES)
ELAPSED TIME:  00:08:18
(VPCAC/ENABLE:AMA,(VPCAC/-SP/CR=LB1:[1,1]SYSMAC/ML,DB1:[204,41](VPCAC
```

```
SYMBOL  VALUE          REFERENCES
ABASE   = 000000        8-1241    8-1241
ACDW1   = 000000        8-1241    8-1241
ACDW2   = 000000        8-1241    8-1241
ACNTH     003132       #8-1241   *37-2691  *37-2692   37-2711   38-2735   39-2766   39-2792   42-2884   43-2939
                        43-2943   43-2991   43-2997   44-3017   44-3022  *45-3052  *45-3053   45-3073   46-3096
                        47-3127   47-3136   47-3159   50-3255   51-3319   51-3339   51-3359   53-3432   53-3455
                        54-3487   54-3496  *55-3524  *55-3525   55-3544   56-3567   57-3599   57-3606   57-3628
                        60-3719   61-3764   61-3786   61-3807   63-3861   63-3869
ACNTL     003134       #8-1241   *37-2690   37-2710   38-2733   38-2734   39-2764   39-2765   39-2790   39-2791
                        41-2831   41-2834   42-2882   42-2883   43-2934   43-2940   43-2944   43-2969   43-2992
                        43-2998   44-3018   44-3023   44-3028   44-3039  *45-3051   45-3071   45-3072   46-3094
                        46-3095   47-3128   47-3134   47-3135   47-3157   47-3158   49-3200   49-3203   50-3254
                        50-3256   51-3320   51-3340   51-3360   51-3371   52-3393   52-3396   53-3433   53-3456
                        54-3488   54-3495   54-3512  *55-3523   55-3543   56-3566   57-3600   57-3605   59-3668
                        59-3671   61-3765   61-3789   61-3808   62-3833   62-3836   63-3862   63-3883
ACOUNT    002424       #8-1241   *79-4855   79-4873  *79-4876   79-4904   79-4905   79-4908   79-4909  *79-4935
                       *79-4939  *84-5507  *84-5535  *84-5546  *84-5550   84-5551   84-5552  *84-5579   84-5603
                        84-5605  *84-5607   84-5620   100-6429  100-6429  100-6430  100-6430
ACPUOP  = 000000        8-1241    8-1241
ADADDR    042016       81-5027   81-5050  #97-6132
ADCALB    032732       81-5070  #82-5125   82-5136   82-5203   82-5247
ADDW0   = 000000        8-1241    8-1241
ADDW1   = 000000        8-1241    8-1241
ADDW10  = 000000        8-1241    8-1241
ADDW11  = 000000        8-1241    8-1241
ADDW12  = 000000        8-1241    8-1241
ADDW13  = 000000        8-1241    8-1241
ADDW14  = 000000        8-1241    8-1241
ADDW15  = 000000        8-1241    8-1241
ADDW2   = 000000        8-1241    8-1241
ADDW3   = 000000        8-1241    8-1241
ADDW4   = 000000        8-1241    8-1241
ADDW5   = 000000        8-1241    8-1241
ADDW6   = 000000        8-1241    8-1241
ADDW7   = 000000        8-1241    8-1241
ADDW8   = 000000        8-1241    8-1241
ADDW9   = 000000        8-1241    8-1241
ADEVCT  = 000000        8-1241    8-1241
ADEVM   = 000000        8-1241    8-1241
ADGAN     034522       83-5379   83-5384  #83-5389
ADLOG     033624       81-5034   81-5077  #83-5252
ADMON     032424      #81-5058   81-5071   81-5078   81-5084   81-5090   81-5110   81-5114   81-5117
ADRET     032374       81-5043  #81-5049
ADTBIT    034610      #83-5404
ADTST     032236        8-1241    8-1241  #81-5025
AENV    - 000000        8-1241    8-1241
AENVM   = 000000        8-1241    8-1241
AFATAL  - 000000        8-1241    8-1241
AFLAG     002312       #8-1241   *14-1658  *14-1681  *19-1920   78-4670   84-5499   86-5660
ALCLK     026574        74-4495  #75-4501
AMADR1  - 000000        8-1241    8-1241
AMADR2  - 000000        8-1241    8-1241
```

SYMBOL VALUE REFERENCES

```
AMADR3  = 000000   8-1241    8-1241
AMADR4  = 000000   8-1241    8-1241
AMAMS1  = 000000   8-1241    8-1241
AMAMS2  = 000000   8-1241    8-1241
AMAMS3  = 000000   8-1241    8-1241
AMAMS4  = 000000   8-1241    8-1241
AMSGAD  = 000000   8-1241    8-1241
AMSGLG  = 000000   8-1241    8-1241
AMSGTY  = 000000   8-1241    8-1241
AMTYP1  = 000000   8-1241    8-1241
AMTYP2  = 000000   8-1241    8-1241
AMTYP3  = 000000   8-1241    8-1241
AMTYP4  = 000000   8-1241    8-1241
ANSW       002432  #8-1241   *14-1663   14-1664   *77-4552   77-4553   77-4554   77-4558   *77-4562   77-4563
                   77-4567   *77-4571   77-4572    77-4577  *79-4696   79-4697   79-4698    79-4700  *79-4725
                   79-4726    79-4728   79-4731   *79-4735   79-4736   79-4737   79-4739   *79-4755   79-4756
                   79-4758    79-4760  *79-4767    79-4769  *79-4776   79-4777   79-4779    79-4781  *79-4965
                   79-4966    79-4970  *79-4972    79-4973   79-4975  *79-4976   79-4977   *81-5063   81-5064
                   81-5067    81-5074   81-5080    81-5086   81-5092   81-5111  *94-6063    94-6065
APASS   = 000000   8-1241    8-1241
APRIOR  = 000000   8-1241
APTCSU  = 000040   98-6188  #98-6191
APTENV  = 000001   98-6186    98-6188   98-6191  #98-6191
APTSIZ  = 000200   12-1620   #98-6191
APTSPO  = 000100   98-6188    98-6191  #98-6191
ASWREG  = 000000   8-1241    8-1241
ATABL      001326  #8-1241    19-1813   19-1848    19-1870   19-1874
ATESTN  = 000000   8-1241    8-1241
ATOD       032324   14-1671  #81-5041   81-5048    81-5052
AUNIT   = 000000   8-1241    8-1241
AUSWR   = 000000   8-1241    8-1241
AUTO       006002   13-1633   13-1641   14-1683   #19-1801   19-1930
AVECT1  = 000000   8-1241    8-1241
AVECT2  = 000000   8-1241    8-1241
AVRSAV     003162  #8-1241    82-5218  *82-5231    82-5232  *89-5893
AVRTBL     003164  #8-1241    82-5233
BASE       002276  #8-1241    16-1725   19-1811    22-2031   27-2271   30-2391   34-2553   36-2647   41-2824
                    49-3193    52-3390   53-3473    59-3661   62-3830   70-4263   77-4541   81-5047   83-5313
                    93-6002
BCNTH      003136  #8-1241   *37-2695  *37-2696    37-2724   38-2754   39-2783   41-2878   42-2896   43-2954
                    43-3005    44-3024  *45-3056   *45-3057   45-3086   46-3117   47-3149   49-3247   50-3267
                    51-3342    51-3361   53-3469    54-3509  *55-3528  *55-3529   55-3557   56-3588   57-3619
                    59-3715    60-3729   61-3809    63-3881
BCNTL      003140  #8-1241   *37-2693  *37-2694    37-2721   37-2723   38-2748   38-2752   38-2753   39-2779
                    39-2781    39-2782   41-2840    41-2874   41-2876   41-2877   42-2892   42-2894   42-2895
                    43-2941    43-2955   43-2964    43-2977   43-2993   43-3006   44-3019   44-3025   44-3035
                   *45-3054   *45-3055   45-3082    45-3084   45-3085   46-3111   46-3115   46-3116   47-3129
                    47-3145    47-3147   47-3148    49-3209   49-3243   49-3245   49-3246   50-3264   50-3266
                    50-3268    51-3321   51-3343    51-3362   51-3377   52-3400   52-3403   53-3434   53-3466
                    53-3468    53-3476   54-3489    54-3506   54-3508  *55-3526  *55-3527   55-3554   55-3556
                    56-3583    56-3587   57-3601    57-3616   57-3618   59-3677   59-3713   60-3727   61-3766
                    61-3790    61-3810   62-3840    62-3843   63-3863   63-3879
```

SYMBOL	VALUE	REFERENCES								
BCOUNT	002426	#8-1241	*79-4856	*79-4877	79-4893	79-4906	79-4907	79-4910	79-4911	*79-4936
		*79-4940	*84-5508	*84-5538	*84-5545	*84-5580	84-5607	*84-5608	*84-5635	100-6429
		100-6429	100-6430	100-6430	100-6432					
BEG	036470	#86-5680								
BINCNX	024576	68-4047	#69-4070							
BITSET	040450	25-2186	25-2189	25-2197	25-2200	25-2209	25-2212	26-2230	26-2233	26-2242
		26-2245	26-2254	26-2257	#92-5975					
BIT0	= 000001	#6-928	6-937	13-1635	19-1905	58-3639	59-3702	59-3704	60-3736	60-3738
		60-3741	60-3743	63-3872	65-3938	69-4125	70-4306	71-4333	83-5260	83-5284
		83-5292	84-5518	84-5529	84-5568	84-5586	86-5672	86-5683	87-5822	89-5880
BIT00	= 000001	#6-928	6-928							
BIT01	= 000002	#6-928	6-928							
BIT02	= 000004	#6-928	6-928							
BIT03	= 000010	#6-928	6-928							
BIT04	= 000020	#6-928	6-928							
BIT05	= 000040	#6-928	6-928							
BIT06	= 000100	#6-928	6-928							
BIT07	= 000200	#6-928	6-928							
BIT08	= 000400	#6-928	6-928							
BIT09	= 001000	#6-928	6-928	98-6183	98-6186					
BIT1	= 000002	#6-928	6-936	83-5264	83-5290	84-5519	84-5541	84-5569	84-5599	86-5673
		86-5684	87-5827	89-5881						
BIT10	= 002000	#6-928	71-4351							
BIT11	= 004000	#6-928								
BIT12	= 010000	#6-928								
BIT13	= 020000	#6-928	13-1639	16-1754	19-1913	20-1962	79-4898	97-6174	98-6186	
BIT14	= 040000	#6-928	16-1758	19-1917	20-1966	97-6178				
BIT15	= 100000	#6-928	42-2886	43-2928	43-2989	50-3258	60-3721			
BIT2	= 000004	#6-928	6-935	69-4091	82-5218	83-5268	83-5296	83-5358	83-5376	83-5394
		87-5779	87-5811	87-5829						
BIT3	= 000010	#6-928	6-934	22-2091	83-5272					
BIT4	= 000020	#6-928	6-933	64-3914						
BIT5	= 000040	#6-928	6-932	13-1637	64-3915					
BIT6	= 000100	#6-928	6-931	13-1625	13-1642	14-1659	16-1720	19-1802	20-1939	21-1984
		22-2030	24-2154	64-3916	77-4530	81-5025	81-5041			
BIT7	= 000200	#6-928	6-930	64-3917	83-5381	93-6020				
BIT8	= 000400	#6-928	69-4129	70-4283	74-4413	93-6010				
BIT9	= 001000	#6-928	69-4151							
BLANK2	024140	#64-3926	71-4327	71-4329	71-4337					
BLAST	003026	#8-1241	*84-5633	*86-5680	86-5691	86-5694	86-5697	*86-5702	*86-5706	86-5715
		*86-5718	*86-5719	100-6432						
BPTVEC	- 000014	#6-928								
BSYCON	033756	83-5277	#83-5279							
BTABL	001566	#8-1241	19-1814	19-1864	19-1869	19-1899				
BYTNUM	002242	#8-1241	*29-2342	*29-2345	*30-2364	*30-2368	*32-2458	*32-2458	*32-2461	*32-2461
		*32-2463	*32-2463	*32-2466	*32-2466	*33-2486	*33-2486	*33-2504	*33-2504	*34-2530
		*34-2533	*36-2623	*36-2626	*80-4990	*80-4994	*80-4998	*80-5002	*80-5009	*80-5014
		89-5906	*89-5914	*89-5917	89-5920	*89-5928	*89-5931	90-5937	*90-5941	*90-5944
		94-6041	*94-6046	*94-6050	*94-6052	*95-6081	95-6089	95-6105	*96-6117	*96-6117
		*96-6118	*96-6118	*96-6119	*96-6119	*96-6120	*96-6120	*96-6122	*96-6122	*96-6123
		*96-6123	*96-6124	*96-6124	*96-6125	*96-6125				
BIT	000002	#6-936	79-4980	79-4980						

SYMBOL	VALUE	REFERENCES								
CBITCL	023672	61-3796	61-3824	62-3855	#63-3860					
CBITCM	021370	51-3384	53-3446	53-3467	#54-3486					
CCNTCK	024430	67-4031	#67-4037							
CCOUNT	002430	#8-1241	*84-5581	*84-5595	84-5608	*84-5609	*84-5636	100-6432		
CHKBYT	040730	#94-6040	96-6117	96-6118	96-6119	96-6120	96-6122	96-6123	96-6124	96-6125
CHNSEL	034422	83-5350	83-5362	#83-5367						
CHNUM	002356	#8-1241	*79-4919	*79-4922	*79-4925	*79-4928	100-6429			
CHSEL	034254	#83-5333								
CH0	002360	#8-1241	*79-4857	*79-4878	79-4915					
CH1	002362	#8-1241	*79-4858	*79-4879	79-4920					
CH2	002364	#8-1241	*79-4859	*79-4880	79-4923					
CH3	002366	#8-1241	*79-4860	*79-4881	79-4926					
CH4	002370	#8-1241	*79-4861	*79-4882	79-4929					
CKICT	031454	79-4869	79-4889	#79-4915						
CKRWSK	024312	66-3972	#66-3987							
CKSRCK	024374	66-3996	#67-4026							
CK4CLK	024236	65-3959	#66-3966							
CK4CKK	026236	72-4385	#73-4392							
CK4INK	025166	69-4088	#70-4171							
CLK	002300	#8-1241	*33-2482	33-2490	33-2498	*33-2503	33-2509	*51-3325	51-3327	51-3345
		*51-3351	51-3353	52-3389	*53-3435	53-3440	53-3448	*61-3787	61-3798	61-3801
		62-3829	*92-5997							
CLKADR	002260	#8-1241	13-1625							
CLKVC	002322	#8-1241	33-2483	33-2514	51-3304	61-3772				
CLKVCA	002324	#8-1241	33-2484	51-3305	61-3773					
CLRINT	040564	20-1957	22-2033	26-2235	29-2347	30-2367	30-2370	30-2384	30-2410	34-2534
		34-2547	34-2597	36-2628	36-2641	36-2671	39-2788	41-2857	47-3155	49-3226
		51-3303	52-3418	53-3431	57-3626	59-3694	61-3767	61-3771	62-3853	70-4235
		83-5329	84-5510	84-5644	86-5669	86-5741	#93-6002			
CM1	060036	79-4710	#100-6398							
CM10	060614	79-4772	#100-6409							
CM11	060760	79-4753	79-4774	#100-6411						
CM12	061024	79-4733	#100-6412							
CM2	060105	79-4713	#100-6399							
CM3	060155	79-4717	#100-6400							
CM4	060225	79-4721	#100-6401							
CM7	060300	79-4750	#100-6404							
CM9	060452	79-4745	79-4765	#100-6407						
CNTRC	040646	16-1721	16-1748	19-1901	20-1958	21-2000	22-2129	22-2141	66-3967	66-3989
		67-4027	69-4071	70-4173	71-4338	77-4581	78-4621	78-4646	78-4664	79-4704
		79-4706	79-4727	79-4741	79-4757	79-4768	79-4778	79-4811	79-4834	79-4853
		79-4871	79-4891	81-5066	82-5127	82-5240	87-5782	87-5798	#93-6019	93-6034
		94-6064	95-6106							
CNUM	003146	#8-1241	*66-3966	66-3971	66-3973	*66-3980	*67-4038	68-4046	68-4049	*68-4065
		*69-4074	*69-4086	69-4087	69-4090	69-4142	*70-4174	*70-4179	70-4180	70-4184
		70-4193	70-4218	*73-4392	73-4396	73-4398	*73-4405	*74-4421	*74-4426	74-4427
		74-4430	74-4449	74-4452	*74-4462	*75-4501	75-4505	75-4507	*75-4514	100-6417
		100-6418								
CONV	002436	#8-1241	*84-5506	*84-5515	84-5516	*84-5559	*84-5565	84-5566		
CONVCT	002442	#8-1241	*84-5558							
CONV7	037776	82-5213	83-5413	83-5454	83-5468	83-5480	84-5512	84-5562	#89-5876	
COSADR	002252	#8-1241	*34-2527	*34-2528	34-2567	34-2572	34-2574	*34-2577	34-2585	34-2591

SYMBOL	VALUE	REFERENCES								
		34-2592	*34-2595	100-6422						
COUNT	040556	33-2483	51-3304	61-3772	#92-5997					
CR	000015	#6-928	98-6188	98-6188						
CRLF	000200	#6-928	13-1623	13-1623	98-6188	98-6188				
CRTST	041026	79-4711	79-4714	79-4718	79-4722	79-4746	79-4751	79-4773	82-5211	#94-6062
		94-6066								
CSR	002262	#8-1241	16-1730	16-1735	19-1805	19-1812	19-1851	19-1860	19-1888	19-1925
		20-1946	21-1989	22-2034	22-2140	24-2167	24-2171	25-2215	25-2216	25-2218
		27-2270	28-2302	28-2314	28-2315	28-2316	29-2341	29-2344	29-2346	30-2363
		30-2366	30-2369	30-2376	30-2377	30-2382	30-2383	30-2394	30-2403	30-2411
		30-2414	31-2433	32-2457	33-2481	34-2529	34-2532	34-2538	34-2539	34-2544
		34-2545	34-2558	34-2565	34-2571	34-2589	34-2598	34-2601	34-2602	36-2622
		36-2625	36-2627	36-2632	36-2633	36-2638	36-2639	36-2650	36-2661	36-2664
		36-2672	36-2675	37-2701	39-2770	39-2771	39-2786	40-2801	41-2825	41-2858
		42-2881	44-3021	45-3062	46-3092	46-3100	46-3110	47-3130	47-3132	47-3133
		47-3152	47-3153	48-3168	48-3173	49-3194	49-3227	50-3252	50-3253	51-3300
		51-3301	51-3316	51-3356	51-3382	52-3408	52-3419	53-3427	53-3430	53-3437
		53-3442	53-3452	53-3479	54-3490	54-3493	55-3534	56-3564	56-3571	56-3582
		57-3602	57-3604	57-3624	58-3637	59-3662	59-3695	60-3718	61-3769	61-3770
		61-3783	61-3805	62-3848	62-3854	63-3864	63-3867	65-3935	67-4039	67-4040
		67-4041	69-4093	69-4100	69-4101	69-4102	69-4155	70-4205	70-4206	70-4207
		70-4227	70-4240	70-4241	70-4242	70-4243	70-4247	70-4261	70-4264	70-4279
		72-4366	74-4475	74-4476	74-4477	74-4487	77-4602	77-4604	79-4791	79-4817
		79-4836	79-4862	79-4896	79-4953	79-4959	79-4980	79-4980	81-5030	81-5053
		82-5162	82-5165	82-5245	83-5257	83-5281	83-5300	83-5304	83-5312	83-5314
		83-5321	83-5327	83-5373	83-5381	83-5390	83-5401	83-5407	83-5479	84-5504
		84-5509	84-5521	84-5528	84-5560	84-5571	84-5585	86-5666	86-5668	86-5675
		86-5686	87-5783	87-5804	87-5814	87-5820	89-5883	92-5975	92-5978	93-6004
		93-6007	95-6074	95-6075	97-6161	97-6164				
CTASCI	024056	#64-3925	71-4318							
CTRSH	003142	#8-1241	*37-2711	37-2714	*37-2724	*38-2735	38-2739	38-2741	*38-2754	*39-2766
		39-2768	39-2772	*39-2783	*39-2792	40-2802	41-2865	*41-2878	*42-2884	*42-2896
		42-2903	42-2906	*45-3073	45-3075	*45-3086	*46-3096	46-3101	46-3103	*46-3117
		*47-3136	47-3138	*47-3149	*47-3159	48-3169	49-3234	*49-3247	*53-3455	53-3459
		*53-3469	*54-3496	54-3499	*54-3509	*55-3544	55-3547	*55-3557	*56-3567	56-3573
		56-3575	*56-3588	*57-3606	57-3609	*57-3619				
CTRSI	003144	#8-1241	*37-2710	37-2713	37-2715	*37-2723	*38-2734	38-2740	38-2742	*38-2753
		*39-2765	39-2769	39-2773	*39-2782	*39-2791	40-2803	41-2866	*41-2877	*42-2883
		*42-2895	42-2904	42-2907	*45-3072	45-3076	*45-3085	*46-3095	46-3102	46-3104
		*46-3116	*47-3135	47-3139	*47-3148	*47-3158	48-3170	49-3235	*49-3246	*53-3456
		53-3458	53-3460	*53-3468	*54-3495	54-3498	54-3500	*54-3508	*55-3543	55-3546
		55-3548	*55-3556	*56-3566	56-3572	56-3574	56-3576	*56-3587	*57-3605	57-3608
		57-3610	*57-3618							
CTXTV	057642	79-4694	#100-6392							
DACHN	057466	79-4963	#100-6387							
DACMON	026762	#77-4543								
DACSTR	026732	14-1674	#77-4537	77-4537	77-4542					
DACTST	027670	77-4565	#79-4691							
DAGTST	031656	77-4546	#79-4952							
DALLY	032102	79-4796	79-4822	79-4841	79-4912	#79-4983				
DANGC	057272	77-4547	#100-6383							
DANSEL	057426	77-4549	#100-6386							

SYMBOL	VALUE	REFERENCES								
DATALO	002440	#8-1241	*84-5522	*84-5524	84-5525	84-5527	*84-5530	*84-5532	84-5533	*84-5572
		*84-5574	84-5575	84-5577	*84-5587	*84-5589	84-5590	*86-5676	*86-5677	86-5678
		*86-5687	*86-5688	86-5689	86-5690					
DATST	027674	#79-4693	79-4703							
DBIT	= 000020	#6-933	25-2207	25-2216	29-2341	30-2363	31-2433	32-2457	33-2481	34-2529
		36-2622	37-2701	42-2881	44-3021	45-3062	47-3130	47-3152	50-3252	51-3300
		53-3442	53-3452	54-3490	54-3493	55-3534	57-3602	57-3624	60-3718	61-3770
		63-3864	63-3867	65-3935	72-4366	74-4487	77-4602	77-4604	83-5479	95-6074
DBUFF	002226	#8-1241	*31-2439	*31-2439	*31-2439	31-2439	31-2439	*95-6094	95-6100	
DCAMON	026676	8-1241	#77-4530							
DCHAN	002372	#8-1241	*79-4970							
DCMN	027006	#77-4549	77-4557	77-4566	77-4576	77-4580	77-4583	79-4958		
DCOUNT	002434	#8-1241	*84-5582	84-5609	*84-5610					
DCOUT	026772	#77-4545								
DCTST	027766	79-4705	#79-4710							
DDBIT	027252	77-4534	77-4579	#77-4601						
DDISP	= 177570	#6-928	8-1241	12-1620						
DECCHN	031722	79-4693	#79-4963	79-4969						
DF100	061711	10-1525	10-1530	10-1535	10-1540	10-1545	10-1550	10-1565	10-1570	10-1575
		10-1580	10-1585	10-1590	10-1610	10-1615	#100-6438			
DF101	061717	10-1555	#100-6439							
DF102	061702	10-1595	10-1600	10-1605	#100-6437					
DF51	061724	10-1424	#100-6440							
DF52	061730	10-1429	#100-6441							
DH1	045461	10-1243	10-1247	10-1255	10-1263	10-1295	#100-6205			
DH100	045360	10-1523	10-1528	10-1533	10-1538	10-1543	10-1548	10-1563	10-1568	10-1573
		10-1583	10-1588	10-1608	#100-6203					
DH101	045425	10-1553	#100-6204							
DH102	055466	10-1593	10-1603	#100-6358						
DH110	055422	10-1578	10-1613	#100-6357						
DH114	055537	10-1598	#100-6359							
DH21	046434	10-1307	#100-6223							
DH23	046571	10-1315	#100-6226							
DH30	056112	10-1337	#100-6365							
DH31	061235	10-1342	10-1347	#100-6415						
DH33	055610	10-1352	#100-6360							
DH34	055744	10-1357	#100-6362							
DH40	056022	10-1463	#100-6363							
DH5	045611	10-1259	10-1279	10-1327	10-1332	10-1367	10-1372	10-1377	10-1382	10-1392
		10-1397	10-1402	10-1412	10-1442	10-1447	#100-6209			
DH57	050512	10-1453	#100-6270							
DH61	050676	10-1458	10-1473	10-1478	10-1508	#100-6274				
DH62	050751	10-1468	10-1483	10-1488	10-1493	10-1498	10-1503	10-1513	#100-6275	
DH7	045730	10-1267	10-1275	10-1299	10-1303	10-1362	10-1387	10-1407	10-1417	#100-6213
DIGIT	005416	14-1686	#16-1720	16-1726						
DISPLA	001156	#8-1241	*12-1620	*12-1620	98-6183	98-6186				
DISPRE	000174	#6-917	12-1620							
DMUT	002254	#8-1241	*16-1745	16-1747	*19-1873	19-1878	*21-1997	21-1998		
DR5016	024034	#64-3912	*65-3937	*65-3938	66-3974	67-4028	68-4052	68-4059	69-4092	69-4108
		69-4148	70-4195	70-4197	70-4226	73-4399	74-4454	74-4456	75-4508	
DSWCH	024046	#64-3920	*69-4079	*69-4144	*69-4151	70-4186	*70-4283	71-4339	74-4432	
DSWR	- 177570	#6-928	8-1241	12-1620						

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES								
DTST	030010	79-4708	#79-4716							
DT1	061450	10-1244	10-1248	10-1256	10-1264	10-1296	#100-6425			
DT100	061340	10-1524	10-1529	10-1534	10-1539	10-1544	10-1549	10-1564	10-1569	10-1574
		10-1584	10-1589	10-1609	#100-6420					
DT101	061324	10-1554	#100-6419							
DT102	061304	10-1594	10-1599	10-1604	#100-6418					
DT110	061266	10-1579	10-1614	#100-6417						
DT17	061360	10-1300	#100-6421							
DT20	061400	10-1304	#100-6422							
DT21	061500	10-1308	#100-6427							
DT23	061420	10-1316	10-1464	#100-6423						
DT30	061512	10-1338	#100-6428							
DT31	061670	10-1343	10-1348	#100-6436						
DT33	061526	10-1353	#100-6429							
DT34	061550	10-1358	#100-6430							
DT5	061464	10-1260	10-1280	10-1328	10-1333	10-1368	10-1373	10-1378	10-1383	10-1393
		10-1398	10-1403	10-1413	10-1443	10-1448	#100-6426			
DT51	061572	10-1423	#100-6431							
DT52	061606	10-1428	#100-6432							
DT57	061624	10-1454	#100-6433							
DT61	061634	10-1459	10-1474	10-1479	10-1509	#100-6434				
DT62	061654	10-1469	10-1484	10-1489	10-1494	10-1499	10-1504	10-1514	#100-6435	
DT7	061430	10-1268	10-1276	10-1363	10-1388	10-1408	10-1418	#100-6424		
EBIT	= 000100	#6-931	25-2184	28-2302	30-2377	30-2382	34-2539	34-2544	34-2565	34-2602
		36-2632	36-2638	40-2801	48-3168	48-3173	51-3316	51-3356	51-3382	53-3427
		53-3437	58-3637	61-3783	61-3805	70-4243	70-4247	70-4279	83-5304	83-5312
ECOUNT	002444	#8-1241	*84-5583	84-5610	*84-5611					
EMTVEC	= 000030	#6-928	12-1620	12-1620						
EM1	045250	10-1242	#100-6199							
EM10	046000	10-1270	#100-6214							
EM100	051300	10-1537	#100-6282							
EM101	051350	10-1542	#100-6283							
EM102	051401	10-1547	#100-6284							
EM103	051453	10-1552	#100-6285							
EM104	051472	10-1557	#100-6286							
EM105	051533	10-1562	#100-6287							
EM106	051560	10-1567	#100-6288							
EM107	051622	10-1572	#100-6289							
EM11	046033	10-1274	#100-6215							
EM110	051644	10-1577	#100-6290							
EM111	051703	10-1582	#100-6291							
EM112	051737	10-1587	#100-6292							
EM113	051771	10-1592	#100-6293							
EM114	052026	10-1597	#100-6294							
EM115	052057	10-1602	#100-6295							
EM116	052111	10-1607	#100-6296							
EM117	052153	10-1612	#100-6297							
EM12	046071	10-1278	#100-6216							
EM13	046121	10-1282	#100-6217							
EM14	046167	10-1286	#100-6218							
EM15	046232	10-1290	#100-6219							
EM16	046275	10-1294	#100-6220							

SYMBOL	VALUE	REFERENCES								
EM17	045713	10-1298	#100-6211							
EM2	045277	10-1246	#100-6200							
EM2X	045346	*25-2185	*25-2196	*25-2208	*26-2229	*26-2241	*26-2253	#100-6202		
EM2Y	045312	#100-6201								
EM20	046340	10-1302	#100-6221							
EM21	046372	10-1306	#100-6222							
EM22	046460	10-1310	#100-6224							
EM23	046534	10-1314	#100-6225							
EM24	046664	10-1318	#100-6237							
EM25	047550	10-1322	#100-6257							
EM26	055312	10-1326	#100-6355							
EM27	055246	10-1331	#100-6354							
EM3	045512	10-1250	#100-6206							
EM30	056050	10-1336	10-1477	#100-6364						
EM31	061105	10-1341	#100-6413							
EM32	061160	10-1346	#100-6414							
EM33	055366	10-1351	#100-6356							
EM34	055657	10-1356	#100-6361							
EM35	056157	10-1361	#100-6366							
EM36	056230	10-1366	#100-6367							
EM37	056276	10-1371	10-1502	#100-6368						
EM4	045543	10-1254	#100-6207							
EM40	056337	10-1376	10-1497	#100-6369						
EM41	056402	10-1381	#100-6370							
EM42	056445	10-1386	10-1472	#100-6371						
EM43	056506	10-1391	10-1467	#100-6372						
EM44	056563	10-1396	#100-6373							
EM45	056622	10-1401	10-1487	#100-6374						
EM46	056667	10-1406	#100-6375							
EM47	056741	10-1411	#100-6376							
EM5	045571	10-1258	#100-6208							
EM50	056767	10-1416	10-1507	#100-6377						
EM53	050042	10-1431	#100-6262							
EM54	050000	10-1436	#100-6261							
EM55	050305	10-1441	#100-6266							
EM56	050360	10-1446	#100-6267							
EM57	050463	10-1452	#100-6269							
EM6	045644	10-1262	#100-6210							
EM60	050536	10-1457	#100-6271							
EM61	050572	10-1462	#100-6272							
EM62	051010	10-1492	#100-6276							
EM63	050641	10-1482	#100-6273							
EM7	045713	10-1266	#100-6212							
EM73	051051	10-1512	#100-6277							
EM74	051123	10-1517	#100-6278							
EM75	051153	10-1522	#100-6279							
EM76	051210	10-1527	#100-6280							
EM77	051236	10-1532	#100-6281							
ERRVEC =	000004	#6-928	12-1620	12-1620	12-1620	*13-1624	*13-1628	16-1727	*16-1728	*16-1733
		*16-1770	*16-1775	19-1809	*19-1810	*19-1850	19-1859	19-1885	*19-1886	*19-1895
		21-1987	*21-1988	*21-2009	*21-2014	22-2028	*22-2029	*22-2085	*22-2117	22-2134
		24-2153	*24-2158	*24-2166	28-2299	*28-2300	*28-2320	*30-2385	*30-2392	*54-2546

SYMBOL		VALUE	REFERENCES								
			*34-2554	*36-2640	*36-2648	41-2819	41-2820	*41-2821	*41-2822	*41-2855	*41-2856
			49-3188	49-3189	*49-3190	*49-3191	*49-3224	*49-3225	59-3656	59-3657	*59-3658
			*59-3659	*59-3692	*59-3693	70-4259	*70-4260	*70-4280	78-4610	*78-4611	*78-4628
			82-5129	*82-5130	*82-5244	87-5754	*87-5755	*87-5854	*93-6024	97-6141	*97-6142
			*97-6150	*97-6156	98-6183	98-6183	98-6183	98-6183			
EXERC		032124	14-1692	#79-4986							
FBIT	=	000200	#6-930	25-2195	28-2316	30-2403	36-2664	41-2858	49-3227	52-3419	59-3695
			62-3854	83-5314	92-5983	93-6004					
FHOUSX		026200	70-4307	71-4311	#72-4365						
FIELD		007170	14-1701	#21-1983							
F5010		032130	8-1241	#80-4988							
F5011		032130	8-1241	#80-4989							
F5012		032142	8-1241	8-1241	#80-4993						
F5013		032154	8-1241	#80-4997							
F6010		032166	8-1241	8-1241	#80-5001						
F6011		032222	8-1241	#80-5013							
F6012		032206	8-1241	#80-5007							
F6013		032206	8-1241	#80-5008							
GAIN		003104	#8-1241	82-5206	87-5819	87-5851					
GAINMG		057346	82-5207	#100-6384							
GAINTB		003062	#8-1241	82-5192	82-5195	82-5208					
GBIT	=	000004	#6-935	16-1735	19-1812	19-1851	19-1860	20-1946	21-1989	22-2034	26-2240
			79-4953	79-4959	82-5162	82-5165	95-6075	97-6161	97-6164		
GBITE		003050	#8-1241	82-5148	83-5351	83-5368	83-5408	83-5447	87-5765	87-5767	87-5769
			87-5771	87-5848	89-5868	*97-6162	*97-6163				
GCODE		042172	81-5033	81-5055	82-5126	87-5764	#97-6161				
GENCOD		040374	22-2037	#91-5950							
GENER		002026	#8-1241	16-1740	19-1864	19-1867	21-1993	91-5951	91-5963		
GNS	=	******	6-917	6-917	13-1623	98-6194	98-6194	98-6194	98-6194	98-6194	98-6194
			98-6194	98-6194	98-6194	98-6194	98-6194	98-6194	98-6194	98-6194	98-6194
			98-6194	98-6194	98-6194	98-6194	98-6194				
GODN		031606	79-4892	#79-4939							
GOUP		031570	79-4872	#79-4935							
GTADRS		027210	77-4531	77-4543	#77-4587						
G670		014176	8-1241	#37-2686							
HBYTE		003060	#8-1241	84-5523	84-5531	84-5573	84-5588	86-5677	86-5688	87-5835	89-5885
			*97-6134	*97-6135	97-6136	97-6145					
HDATA0		002404	#8-1241	78-4654	78-4659	78-4660	79-4719	79-4743	79-4747	79-4763	79-4798
			79-4823	79-4842	79-4864	79-4884	79-4904				
HDATA1		002410	#8-1241	79-4906							
HDATA2		002414	#8-1241	79-4908							
HDATA3		002420	#8-1241	79-4813	79-4910						
HT	=	000011	#6-928	98-6188	98-6188						
IAR		002264	#8-1241	19-1845	19-1856	22-2125	22-2131	28-2307	30-2393	30-2405	34-2555
			34-2579	36-2649	36-2666	41-2829	49-3198	52-3391	53-3474	59-3666	62-3851
			70-4268	83-5318	93-6006						
ICHAN		002400	#8-1241	*79-4977							
INTCOM		030430	77-4575	#79-4789							
INTDAT		002316	#8-1241	*82-5220	82-5221	*82-5222	82-5225	*82-5226	82-5227	82-5230	83-5416
			83-5423	83-5428	83-5432	83-5455	83-5469	84-5513	84-5563	*89-5878	*89-5887
			89-5893	*89-5894	*89-5895	*89-5896	*89-5897	89-5898			
INTHLT		014724	39-2780	#39-2786							

SYMBOL	VALUE	REFERENCES								
INTHL1	022310	57-3617	#57-3624							
INTHL2	017176	47-3146	#47-3152							
INTLD	031366	79-4863	79-4883	#79-4904						
IOCM	010142	14-1595	#24-2153							
IOTVEC	= 000020	#6-928	12-1620	12-1620						
KBINT	040722	13-1642	16-1720	19-1802	20-1939	21-1984	22-2030	24-2154	77-4530	81-5025
		81-5041	#93-6034							
KBVEC	002354	#8-1241	13-1642	16-1720	19-1802	20-1939	21-1984	22-2030	24-2154	77-4530
		81-5025	81-5041							
KLEER	032024	16-1729	16-1744	16-1761	16-1771	19-1804	19-1902	20-1969	20-1974	21-1996
		21-2010	22-2032	22-2135	24-2159	24-2168	25-2222	27-2285	28-2319	37-2700
		44-3020	45-3061	51-3299	54-3492	55-3533	63-3866	65-3934	72-4365	74-4486
		77-4550	77-4601	77-4605	78-4612	79-4691	79-4771	79-4784	79-4809	79-4816
		79-4832	79-4835	79-4851	79-4897	79-4952	#79-4980	81-5042	83-5255	83-5489
		84-5645	86-5742	87-5853	93-6023	96-6114				
KOUNT	002422	#8-1241	*78-4635	*78-4649	*78-4652	*78-4667	79-4947			
LAST	003030	#8-1241	*86-5681	*86-5689	86-5691	*86-5693	86-5694	*86-5696	86-5697	86-5707
		86-5710								
LASTCN	003032	#8-1241	*86-5682	86-5700	86-5704	*86-5709	86-5712	86-5716	86-5722	*86-5732
		100-6431								
LBYTE	003056	#8-1241	82-5246	83-5328	83-5374	83-5392	83-5402	84-5522	84-5530	84-5572
		84-5587	86-5676	86-5687	87-5834	89-5884	*97-6132	97-6143		
LDATA0	002402	#8-1241	*77-4590	78-4637	78-4641	78-4642	78-4658	79-4720	79-4744	79-4748
		79-4764	79-4792	79-4818	79-4837	79-4905	79-4954			
LDATA1	002406	#8-1241	79-4793	79-4819	79-4838	79-4907				
LDATA2	002412	#8-1241	79-4794	79-4820	79-4839	79-4909				
LDATA3	002416	#8-1241	79-4795	79-4821	79-4840	79-4911				
LF	= 000012	#6-928	98-6188	98-6188						
LINEAR	035252	81-5035	81-5089	#84-5495						
LOGIC	006660	6-925	19-1904	19-1906	#19-1930					
LONGIN	002256	#8-1241								
LOOP	006740	14-1704	#20-1938							
LOPTST	041240	20-1956	#96-6114							
MAPE	007370	14-1689	#22-2028							
MASS0	052175	14-1676	16-1769	17-1784	21-2008	22-2083	81-5065	82-5239	95-6098	#100-6298
MASS10	052603	22-2055	#100-6307							
MASS11	052636	16-1767	21-2006	22-2081	22-2114	82-5202	#100-6308			
MASS12	052672	14-1660	#100-6309							
MASS13	052744	16-1722	21-1983	77-4538	81-5044	#100-6310				
MASS14	052776	#100-6311								
MASS15	053023	16-1760	78-4672	79-4900	#100-6312					
MASS16	053052	24-2157	#100-6313							
MASS17	053075	19-1919	#100-6314							
MASS18	053121	#100-6315								
MASS19	053127	95-6076	#100-6316							
MASS2	052200	22-2038	#100-6299							
MASS20	053166	21-1999	#100-6317							
MASS21	053234	20-1940	#100-6318							
MASS22	053267	20-1943	#100-6319							
MASS23	053321	20-1938	#100-6320							
MASS25	053362	20-1968	#100-6321							
MASS26	053405	20-1971	#100-6322							

SYMBOL	VALUE	REFERENCES								
MASS27	053451	95-6096	#100-6323							
MASS28	053454	20-1973	#100-6324							
MASS29	053537	22-2075	#100-6325							
MASS3	052212	22-2041	#100-6300							
MASS30	053602	22-2077	#100-6326							
MASS31	053650	22-2079	#100-6327							
MASS32	053724	22-2096	#100-6328							
MASS33	053733	22-2100	#100-6329							
MASS34	053763	22-2105	#100-6330							
MASS35	054020	22-2110	#100-6331							
MASS36	054031	81-5113	#100-6332							
MASS37	054166	81-5094	82-5125	#100-6337						
MASS38	054232	82-5143	#100-6338							
MASS39	054306	82-5153	82-5178	82-5185	#100-6339					
MASS4	052261	22-2043	#100-6301							
MASS40	054351	82-5158	82-5170	#100-6340						
MASS41	054424	82-5188	#100-6341							
MASS42	054475	82-5198	#100-6342							
MASS43	054523	82-5210	#100-6343							
MASS44	054560	82-5212	#100-6344							
MASS45	054620	14-1714	#100-6345							
MASS46	054700	81-5058	#100-6346							
MASS48	054742	22-2057	#100-6347							
MASS49	055001	22-2059	#100-6348							
MASS5	052330	22-2045	#100-6302							
MASS50	055041	19-1852	#100-6349							
MASS51	050421	84-5502	86-5663	#100-6268						
MASS52	055122	22-2069	#100-6351							
MASS53	055160	22-2071	#100-6352							
MASS54	055217	22-2073	#100-6353							
MASS55	050154	86-5728	#100-6264							
MASS56	055071	22-2061	22-2063	22-2065	22-2067	#100-6350				
MASS6	052374	22-2047	#100-6303							
MASS7	052426	22-2049	#100-6304							
MASS8	052476	22-2051	#100-6305							
MASS9	052537	22-2053	#100-6306							
MAXPDX	026306	73-4397	#74-4413							
MBIT	= 000040	#6-932	25-2195	27-2270	28-2302					
MOD	002326	#8-1241	21-1992							
MODUL	002150	#8-1241	16-1738	19-1873						
MONDAY	041122	80-4991	80-4995	80-4999	#95-6088					
MONIT	005106	13-1644	#14-1657	14-1675	14-1710	14-1712	16-1762	16-1772	16-1776	17-1785
		19-1872	19-1921	20-1970	20-1975	21-2011	21-2015	22-2136	24-2160	79-4986
		93-6025								
MUT	002100	#8-1241	16-1739	19-1875	19-1881					
MUX1	037060	19-1891	81-5109	#87-5749						
MXNUM	002224	#8-1241	*19-1887	19-1889	*19-1892	19-1893	*81-5107	87-5756	87-5757	87-5763
		87-5777	87-5792	87-5800	87-5821	100-6434	100-6435			
M0	046611	#100-6228								
M1	046605	#100-6227								
M10	046636	82-5236	82-5237	#100-6235						
M11	046640	16-1757	19-1916	20-1965	97-6177	#100-6236				

SYMBOL	VALUE	REFERENCES								
M13	046732	13-1645	#100-6238							
M14	046754	13-1646	#100-6239							
M15	046775	13-1647	#100-6240							
M16	047021	#100-6241								
M17	047044	13-1648	#100-6242							
M18	047100	#100-6243								
M19	047117	13-1649	#100-6244							
M2	046615	26-2253	#100-6229							
M20	047136	14-1711	77-4582	79-4702	79-4968	81-5100	81-5116	82-5135	#100-6245	
M21	047145	13-1650	#100-6246							
M22	047164	17-1780	#100-6247							
M23	047341	17-1781	#100-6251							
M24	047353	13-1651	#100-6252							
M25	047373	13-1652	#100-6253							
M26	047412	13-1653	#100-625[illegible]							
M27	047444	14-1707	#100-6255							
M28	047501	13-1654	#100-6256							
M3	046617	26-2241	#100-6230							
M30	047611	77-4560	77-4569	#100-6258						
M4	046622	26-2229	#100-6231							
M5	046625	25-2208	#100-6232							
M5010	011242	8-1241	#29-2338							
M5011	012670	8-1241	#34-2525							
M5012	013520	8-1241	8-1241	#36-2619						
M5013	011344	8-1241	#30-2360							
M5014	016402	8-1241	8-1241	8-1241	8-1241	#45-3045				
M5014R	021534	54-3507	#54-3511							
M5016	024144	8-1241	#65-3929							
M6	046630	25-2196	#100-6233							
M6010	012006	8-1241	8-1241	#31-2430						
M6011	012372	8-1241	#33-2476							
M6011A	012564	33-2497	#33-2502							
M6011B	012656	33-2493	33-2508	#33-2513						
M6012	012200	8-1241	#32-2453							
M6013	012200	8-1241	#32-2454							
M6014	021546	8-1241	#55-3518							
M7	046633	25-2185	#100-6234							
M80	057051	#100-6378								
M81	057076	79-4707	#100-6379							
M82	057135	79-4723	#100-6380							
M83	057215	79-4783	#100-6381							
M84	047742	82-5243	#100-6260							
M85	057242	#100-6382								
NOCLK	040554	6-919	33-2514	#92-5992						
NORAMP	002302	#8-1241	*83-5406	*83-5463	*83-5477	84-5497	86-5658			
NUM	003150	#8-1241	*39-2795	40-2802	40-2803	*42-2886	*42-2889	42-2900	42-2903	42-2904
		42-2908	*43-2928	*43-2931	43-2936	[illegible]-2939	43-2940	43-2941	43-2945	43-2956
		*47-3162	48-3169	48-3170	*50-3258	*5[illegible]-3261	50-3272	50-3273	*51-33[illegible]2	*51-3335
		*51-3349	52-3411	52-3414	*57-3631	58-3638	58-3640	*60-3721	*60-[illegible]724	60-3733
		60-3737	60-3739	60-3745	*69-4073	*69-4097	69-4105	69-4160	*70-4186	*70-4187
		*70-4188	*70-4189	70-4209	*74-4445	*74-4469	74-4472	*74-44[illegible]9		
OUT	037046	86-5727	86-5738	#86-5741						

SYMBOL	VALUE	REFERENCES								
OVFS	024036	#64-3914	70-4288							
PASCNT	002320	#8-1241	*14-1709	16-1749	16-1752	16-1756	19-1907	19-1910	19-1915	20-1960
		20-1964	97-6172	97-6176						
PATT	002232	#8-1241	31-2439	31-2439	38-2736	38-2737	38-2758	46-3097	46-3098	46-3121
		56-3568	56-3569	56-3592						
PIRQ	= 177772	#6-928								
PIRQVE	= 000240	#6-928								
PR0	= 000000	#6-928	13-1643	28-2301	28-2321	30-2373	30-2413	33-2485	34-2535	34-2600
		36-2629	36-2674	40-2800	48-3167	51-3317	52-3410	53-3438	53-3481	58-3636
		61-3784	62-3850	70-4239	83-5256	83-5330	86-5657			
PR1	= 000040	#6-928								
PR2	= 000100	#6-928								
PR3	= 000140	#6-928								
PR4	= 000200	#6-928								
PR5	= 000240	#6-928								
PR6	= 000300	#6-928								
PR7	= 000340	#6-928	28-2298	30-2375	33-2484	34-2537	36-2631	37-2688	39-2794	40-2808
		41-2822	45-3049	47-3161	48-3174	49-3191	51-3305	51-3307	51-3357	51-3383
		53-3425	53-3429	53-3443	53-3451	55-3521	57-3630	58-3645	59-3659	61-3773
		61-3775	61-3804	67-4037	69-4070	70-4172	70-4248	83-5280		
PS	= 177776	#6-928	6-928							
PSW	= 177776	#6-928								
PWRVEC	= 000024	#6-928	98-6195	98-6195	98-6195	98-6195	98-6195	98-6195		
RAMPST	036414	#86-5670								
RAMP1	002446	#8-1241	*84-5618	*84-5619	84-5633	*86-5720	86-5730			
RAMP2	002566	#8-1241	*84-5620	84-5634	*86-5721	86-5731				
RAMP3	002706	#8-1241	*86-5722	86-5732						
RBIT	= 000001	#6-937	19-1888	19-1925	22-2140	26-2252	28-2314	30-2394	34-2558	34-2571
		34-2589	36-2650	36-2661	41-2825	49-3194	52-3408	53-3479	59-3662	62-3848
		69-4093	69-4155	70-4227	70-4264	81-5030	81-5053	82-5245	83-5257	83-5281
		83-5321	83-5327	83-5373	83-5390	83-5401	84-5504	84-5521	84-5528	84-5560
		84-5571	84-5585	86-5666	86-5675	86-5686	87-5783	87-5804	87-5814	89-5883
		93-6007								
RDCHR	= 104406	14-1661	77-4551	77-4561	77-4570	79-4695	79-4724	79-4734	79-4754	79-4766
		79-4775	81-5062	94-6062	98-6184	#98-6194				
RDLIN	= 104407	98-6185	#98-6194							
RDOCT	= 104410	14-1667	14-1708	16-1723	17-1782	20-1941	20-1944	21-1985	77-4539	79-4964
		81-5045	81-5095	82-5131	82-5144	82-5189	95-6077	#98-6194		
RERROR	002306	#8-1241	*84-5496	*84-5627	84-5628					
RESREG	= 104412	54-3511	75-4519	#98-6194						
RESVEC	= 000010	#6-928								
RETURN	003160	#8-1241	*77-4537	*81-5043	93-6029					
ROTDAT	031624	78-4636	78-4653	#79-4943						
ROTFLG	003036	#8-1241	*79-4943	*79-4949						
ROTPAT	003034	#8-1241	*78-4633	78-4637	78-4640	*78-4651	78-4654	78-4657	*79-4946	
RUMP	036322	81-5036	81-5083	#86-5656						
R6	=%000006	#6-928	*12-1620	*12-1620	12-1620	*13-1629	*14-1657	14-1715	94-6054	97-6180
R7	=%000007	#6-928								
SAVREG	= 104411	45-3048	65-3939	#98-6194						
SCOPE	= 000004	#6-928	19-1930	24-2164	25-2182	25-2193	25-2205	26-2226	26-2238	26-2250
		27-2268	28-2295	28-2322	29-2339	29-2348	30-2361	30-2386	31-2431	31-2440
		32-2455	32-2467	33-2477	33-2513	34-2525	34-2548	36-2620	36-2642	37-2686

SYMBOL	VALUE	REFERENCES								
		45-3045	55-3518	65-3929	78-4631	78-4669	79-4789	79-4895	83-5252	83-5490
		87-5855								
SDELAY	026650	#76-4522								
SETBYT	040332	32-2458	32-2461	32-2463	32-2466	33-2486	33-2504	#90-5937	96-6117	96-6118
		96-6119	96-6120	96-6122	96-6123	96-6124	96-6125			
SETCLK	004676	6-927	#13-1624							
SETPTN	041052	80-5004	80-5010	80-5015	#95-6074					
SETRAM	037746	84-5511	86-5671	#89-5868						
SR5016	024032	#64-3911	*65-3936	65-3957	66-3973	66-4000	66-4001	67-4029	68-4049	68-4050
		69-4090	69-4091	69-4094	69-4105	69-4156	69-4158	70-4193	70-4194	70-4224
		70-4225	70-4228	70-4269	70-4289	72-4383	73-4398	74-4452	74-4453	74-4484
		74-4493	75-4507							
STACK	= 001100	#6-928	12-1620							
START	004364	6-917	#12-1620							
STAT1	003052	#8-1241	19-1889	19-1890	19-1926	81-5031	81-5054	81-5060	82-5161	82-5163
		82-5209	83-5282	83-5336	83-5337	83-5340	83-5343	83-5344	83-5361	83-5364
		83-5375	83-5391	83-5410	83-5412	83-5449	83-5451	83-5478	84-5561	87-5751
		87-5756	87-5758	87-5763	87-5777	87-5784	87-5793	87-5794	87-5821	87-5844
		89-5870	89-5872	89-5876	89-5889	*97-6136	*97-6137	97-6138	97-6147	97-6162
STAT2	003054	#8-1241	81-5032	81-5061	82-5206	83-5259	83-5260	83-5264	83-5268	83-5272
		83-5276	83-5284	83-5290	83-5292	83-5296	83-5358	83-5376	83-5393	83-5394
		84-5505	84-5518	84-5519	84-5529	84-5541	84-5568	84-5569	84-5586	84-5599
		86-5667	86-5672	86-5673	86-5683	86-5684	87-5752	87-5778	87-5779	87-5806
		87-5810	87-5811	87-5815	87-5819	87-5822	87-5827	87-5829	89-5877	89-5880
		89-5881	89-5890	*97-6138	*97-6139	97-6149				
STCO	036504	#86-5683	86-5711	86-5713						
STKLMT	= 177774	#6-928								
STSTIM	023210	60-3728	#61-3764							
STST12	020112	50-3265	#51-3299							
ST1SAV	003126	#8-1241	*89-5876	89-5889						
ST2SAV	003130	#8-1241	*89-5877	89-5890						
SWLOOP	042224	24-2155	77-4574	81-5076	81-5082	81-5088	81-5108	#97-6167	97-6179	
SWR	001154	#8-1241	12-1620	*12-1620	12-1620	*12-1620	*12-1620	79-4898	98-6183	98-6186
		98-6186	98-6186	98-6192	98-6195					
SWREG	000176	#6-917	12-1620	*13-1639	16-1754	16-1758	*17-1783	19-1913	19-1917	20-1962
		20-1966	97-6174	97-6178						
SWREGS	005754	14-1698	#17-1780							
SW0	= 000001	#6-928								
SW00	= 000001	#6-928	6-928							
SW01	= 000002	#6-928	6-928							
SW02	= 000004	#6-928	6-928							
SW03	= 000010	#6-928	6-928							
SW04	= 000020	#6-928	6-928							
SW05	= 000040	#6-928	6-928							
SW06	= 000100	#6-928	6-928							
SW07	= 000200	#6-928	6-928							
SW08	= 000400	#6-928	6-928							
SW09	= 001000	#6-928	6-928							
SW1	= 000002	#6-928								
SW10	= 002000	#6-928								
SW11	= 004000	#6-928								
SW12	= 010000	#6-928								

SYMBOL	VALUE	REFERENCES
SW13	= 020000	#6-928
SW14	= 040000	#6-928
SW15	= 100000	#6-928
SW2	= 000004	#6-928
SW3	= 000010	#6-928
SW4	= 000020	#6-928
SW5	= 000040	#6-928
SW6	= 000100	#6-928
SW7	= 000200	#6-928
SW8	= 000400	#6-928
SW9	= 001000	#6-928
TABLE	040420	19-1817 #91-5962
TADDR	002244	#8-1241 *16-1724 16-1725 16-1736 *19-1874 *20-1942 20-1947 *21-1986 21-1990 30-2395 30-2402 30-2406 31-2434 31-2439 31-2439 33-2479 33-2494 33-2505 34-2527 34-2556 *34-2578 34-2580 *34-2596 36-2651 36-2659 *36-2660 36-2662 *36-2663 36-2667 37-2690 37-2691 37-2693 37-2695 *37-2713 37-2721 *38-2733 38-2748 *38-2752 *39-2764 39-2779 *39-2781 *39-2790 41-2831 41-2843 41-2874 *41-2876 *42-2882 42-2892 *42-2894 *43-2934 *43-2964 *43-2969 *43-2977 *44-3028 *44-3035 *44-3039 45-3051 45-3052 45-3054 45-3056 *45-3071 45-3082 *45-3084 *46-3094 46-3111 *46-3115 *47-3134 47-3145 *47-3147 *47-3157 49-3200 49-3212 49-3243 *49-3245 *50-3254 50-3264 *50-3266 *51-3371 *51-3377 *52-3396 *52-3403 *53-3458 53-3466 *54-3498 54-3506 *54-3512 55-3523 55-3524 55-3526 55-3528 *55-3546 55-3554 *56-3572 56-3583 *57-3608 57-3616 *57-3628 58-3638 *58-3639 58-3640 59-3668 59-3680 *59-3702 59-3703 *59-3704 59-3705 59-3713 *59-3715 *60-3719 60-3727 *60-3729 *60-3736 60-3737 *60-3738 60-3739 *60-3741 60-3742 *60-3743 60-3744 *62-3836 *62-3843 *63-3869 63-3871 *63-3872 63-3873 63-3879 *63-3881 *63-3883 65-3936 65-3937 71-4319 *77-4540 77-4541 77-4589 77-4590 *77-4591 *77-4596 78-4615 *81-5046 81-5047 83-5319 89-5907 89-5908 89-5911 *89-5913 *89-5916 89-5921 89-5922 89-5925 *89-5927 *89-5930 90-5938 90-5939 *90-5940 *90-5943 94-6045 95-6079 *95-6080 95-6091 95-6094 95-6095 *95-6102 *95-6105 97-6132 97-6133 97-6134 97-6140 *97-6144 *97-6146 *97-6148 *97-6151 100-6417 100-6418 100-6419 100-6420 100-6426 100-6429 100-6430 100-6431 100-6432 100-6436
TADDR1	002246	#8-1241 *33-2479 *33-2480 33-2496 33-2507
TBADDR	002250	#8-1241 *87-5751 *89-5911 *89-5925 *94-6045 *97-6140 100-6424 100-6434 100-6435
TBIT	= 000010	#6-934 26-2228 29-2344 29-2346 30-2366 30-2369 30-2376 30-2383 30-2411 30-2414 34-2529 34-2532 34-2538 34-2545 34-2598 34-2601 36-2625 36-2627 36-2633 36-2639 36-2672 36-2675 39-2770 39-2771 39-2786 42-2881 46-3092 46-3100 46-3110 47-3132 47-3133 47-3153 50-3253 51-3301 51-3356 51-3382 53-3427 53-3430 56-3564 56-3571 56-3582 57-3604 57-3624 60-3718 61-3769 61-3770 61-3805 67-4039 67-4040 67-4041 69-4100 69-4101 69-4102 70-4205 70-4206 70-4207 70-4240 70-4241 70-4242 70-4261 74-4475 74-4476 74-4477 79-4791 79-4817 79-4836 79-4862 79-4896 83-5407 84-5509 86-5668 87-5820
TBITVE	= 000014	#6-928
TEMP	002304	#8-1241 *28-2307 28-2308 28-2312 *30-2391 *30-2393 30-2395 30-2397 *30-2405 30-2406 *31-2439 *31-2439 31-2439 *34-2553 *34-2555 34-2556 34-2559 *34-2579 34-2580 *36-2647 *36-2649 36-2651 36-2653 *36-2666 36-2667 *41-2824 *41-2829 41-2834 41-2840 41-2843 41-2848 *43-2954 *43-2955 43-2956 43-2963 43-2976 *44-3024 *44-3025 44-3032 *49-3193 *49-3198 49-3203 49-3209 49-3212 49-3217 *50-3273 *50-3274 50-3276 50-3277 50-3281 *52-3390 *52-3391 52-3393 52-3400 52-3409 *53-3473 *53-3474 53-3476 53-3480 *59-3661 *59-3666 59-3671 59-3677 59-3680 59-3685 *62-3830 *62-3831 62-3833 62-3840 62-3849 *69-4125 69-4129

SYMBOL	VALUE	REFERENCES								
		*69-4135	69-4136	*69-4141	69-4144	*70-4202	*70-4208	70-4209	*70-4213	70-4218
		70-4221	70-4225	*70-4230	*70-4263	*70-4268	70-4269	70-4274	*71-4325	*71-4326
		71-4328	*74-4413	*74-4418	74-4432	*74-4438	*74-4440	*74-4441	74-4472	*78-4642
		*78-4660	*79-4804	*79-4805	*79-4806	*79-4827	*79-4828	*79-4829	*79-4846	*79-4847
		*79-4848	*79-4864	*79-4865	79-4866	*79-4884	*79-4885	79-4886	79-4915	79-4920
		79-4923	79-4926	79-4929	*81-5096	81-5098	*81-5102	*81-5103	*81-5104	*81-5105
		*81-5106	81-5107	*82-5132	82-5133	*82-5138	*82-5139	*82-5140	*82-5141	*82-5142
		82-5146	82-5161	*83-5313	*83-5318	83-5319	83-5322	*89-5884	*89-5885	*89-5886
		89-5887	100-6428	100-6436						
TEMPB	003156	#8-1241	*51-3361	*51-3362	51-3374	*61-3809	*61-3810	61-3819	*70-4175	*70-4284
		70-4301								
TEMP1	002314	#8-1241	*20-1945	20-1950	*30-2390	*30-2398	30-2399	*31-2439	*31-2439	31-2439
		*34-2552	*34-2560	34-2561	*36-2646	*36-2654	36-2655	*41-2823	*41-2849	41-2850
		*43-2962	43-2968	*49-3192	*49-3218	49-3219	*52-3389	*59-3660	*59-3686	59-3687
		*62-3829	*70-4262	*70-4275	70-4276	*83-5317	*83-5323	83-5324	94-6040	94-6042
		*94-6047	*94-6049	*94-6053	100-6421					
TEMP2	003040	#8-1241	*19-1869	*82-5163	*82-5164	82-5166	82-5174	82-5181	100-6433	
TEMP3	003042	#8-1241	*19-1870	*79-4807	*79-4830	*79-4849	*79-4866	*79-4867	*79-4868	*79-4886
		*79-4887	*79-4888	*82-5145	82-5151	82-5156	82-5168	82-5176	82-5183	82-5209
		*87-5771	*93-6002	*93-6006	93-6008	100-6423	100-6429	100-6430	100-6433	100-6436
TEMP4	003044	#8-1241	*82-5190	82-5192	*93-6003	*93-6009	93-6010			
TEMP5	003046	#8-1241	*89-5879	*89-5888	89-5891					
TEXT	057534	77-4556	#100-6388							
TINIMD	017022	#47-3126								
TINIMO	022150	#57-3598								
TINIZ	014564	#39-2763								
TIOCM	010210	19-1803	19-1861	24-2156	#24-2164					
TKVEC	= 000060	#6-928								
TMOVEC	005404	13-1628	#14-1714	30-2385	34-2546	36-2640	93-6024	97-6156		
TPVEC	= 000064	#6-928								
TRANPT	015424	41-2875	#42-2881							
TRANP1	023002	59-3714	#60-3718							
TRANP2	017714	49-3244	#50-3252							
TRAPVE	- 000034	#6-928	12-1620	12-1620						
TRTVEC	= 000014	#6-928								
TSECTB	020762	51-3380	52-3420	#53-3425						
TSTADR	027304	77-4532	77-4544	#78-4610						
TSTBYT	040166	29-2342	29-2345	30-2364	30-2368	32-2458	32-2461	32-2463	32-2466	33-248c
		33-2504	34-2530	34-2533	36-2623	36-2626	#89-5906			
TSTONE	040250	33-2501	33-2512	#89-5920						
TSTPAT	014410	37-2722	#38-2731							
TSTPT1	021760	55-3555	#56-3563							
TSTPT2	016624	45-3083	#46-3091							
TSTRGS	027376	77-4603	#78-4631							
TST1	010210	#24-2164	98-6183							
TST10	010716	#27-2268	98-6183							
TST11	011012	#28-2295	98-6183							
TST12	011242	#29-2339	98-6183							
TST13	011344	#30-2361	98-6183							
TST14	012006	#31-2431	98-6183							
TST15	012200	#32-2455	98-6183							
TST16	012372	#33-2477	98-6183							

SYMBOL	VALUE	REFERENCES								
TST17	012670	#34-2525	98-6183							
TST2	010276	#25-2182	98-6183							
TST20	013520	#36-2620	98-6183							
TST21	014176	#37-2686	98-6183							
TST22	016402	#45-3045	98-6183							
TST23	021546	#55-3518	98-6183							
TST24	024144	#65-3929	98-6183							
TST25	027376	#78-4631	98-6183							
TST26	030430	#79-4789	98-6183							
TST27	033624	#83-5252	98-6183							
TST3	010342	#25-2193	98-6183							
TST4	010406	#25-2205	98-6183							
TST5	010536	#26-2226	98-6183							
TST6	010606	#26-2238	98-6183							
TST7	010652	#26-2250	98-6183							
TYPDS	= 104405	82-5238	#98-6194							
TYPDSW	026022	#71-4318								
TYPE	- 104401	13-1623	13-1645	13-1646	13-1647	13-1648	13-1649	13-1650	13-1651	13-1652
		13-1653	13-1654	14-1660	14-1664	14-1676	14-1707	14-1711	14-1714	16-1722
		16-1757	16-1760	16-1767	16-1769	17-1780	17-1781	17-1784	19-1852	19-1916
		19-1919	20-1938	20-1940	20-1943	20-1965	20-1968	20-1971	20-1973	21-1983
		21-1999	21-2006	21-2008	22-2038	22-2041	22-2043	22-2045	22-2047	22-2049
		22-2051	22-2053	22-2055	22-2057	22-2059	22-2061	22-2063	22-2065	22-2067
		22-2069	22-2071	22-2073	22-2075	22-2077	22-2079	22-2081	22-2083	22-2096
		22-2100	22-2105	22-2110	22-2114	24-2157	71-4318	71-4321	71-4327	71-4328
		71-4329	71-4337	71-4342	71-4348	71-4357	77-4538	77-4547	77-4549	77-4553
		77-4556	77-4560	77-4569	77-4582	78-4672	79-4694	79-4697	79-4702	79-4707
		79-4710	79-4713	79-4717	79-4721	79-4723	79-4726	79-4733	79-4736	79-4745
		79-4750	79-4753	79-4756	79-4765	79-4772	79-4774	79-4777	79-4783	79-4900
		79-4963	79-4968	81-5044	81-5058	81-5064	81-5065	81-5094	81-5100	81-5113
		81-5116	82-5125	82-5135	82-5143	82-5153	82-5158	82-5170	82-5178	82-5185
		82-5188	82-5198	82-5202	82-5207	82-5210	82-5212	82-5236	82-5237	82-5239
		82-5243	84-5502	86-5663	86-5728	95-6076	95-6096	95-6098	97-6177	98-6184
		98-6184	98-6184	98-6185	98-6185	98-6186	98-6187	98-6187	98-6187	98-6187
		98-6187	98-6187	98-6187	98-6188	98-6189	98-6192	#98-6194	98-6195	
TYPOC	= 104402	14-1715	16-1756	16-1768	19-1915	20-1964	21-2007	22-2039	22-2082	22-2097
		22-2113	71-4320	82-5208	82-5221	82-5225	95-6095	95-6097	97-6176	98-6187
		98-6187	#98-6194							
TYPON	= 104404	#98-6194								
TYPOPT	005036	#13-1645	14-1678							
TYPOS	= 104403	#98-6194								
UPREG	027650	78-4638	78-4647	78-4655	78-4665	#78-4678				
ARI	003152	#8-1241	•51-3309	51-3311	51-3364	•51-3365	•51-3366	51-3368	51-3374	•61-3777
		61-3779	•61-3812	61-3814	61-3819					
VCHAN	002376	#8-1241	•79-4975							
VECTO	002266	#8-1241	28-2297	30-2374	34-2536	36-2630	39-2793	47-3160	51-3306	53-3428
		57-3629	61-3774	70-4171	83-5279					
VECTOA	002270	#8-1241	28-2298	30-2375	34-2537	36-2631	39-2794	47-3161	51-3307	53-3429
		57-3630	61-3775	70-4172	83-5280					
VECTT	002272	#8-1241								
VECTTA	002274	#8-1241								
XCHAN	002374	#8-1241	79-4716	•79-4973	•79-4974					

SYMBOL	VALUE	REFERENCES								
XLOOP	003154	#8-1241	*42-2899	*42-2910	*43-2935	*43-2947	*43-2959	*60-3732	*60-3747	*61-3788
		*61-3792	*71-4315	71-4325	71-4330	*71-4358	71-4359	*76-4522	76-4525	
XXDP	002236	#8-1241	13-1630	19-1903						
XXX	002310	#8-1241	*97-6168	97-6169	*97-6169	97-6170				
YLOOP	002222	#8-1241	*25-2217	*25-2217	25-2217	*28-2303	*28-2303	28-2303	*30-2378	*30-2378
		30-2378	*30-2415	*30-2415	30-2415	*34-2540	*34-2540	34-2540	*34-2603	*34-2603
		34-2603	*36-2634	*36-2634	36-2634	*36-2676	*36-2676	36-2676	*39-2787	*39-2787
		39-2787	*40-2804	*40-2804	40-2804	*41-2864	*41-2864	41-2864	*43-2995	*43-2995
		43-2995	*47-3154	*47-3154	47-3154	*48-3171	*48-3171	48-3171	*49-3233	*49-3233
		49-3233	*51-3302	*51-3302	51-3302	*51-3326	*51-3329	*53-3439	*53-3439	53-3439
		*57-3625	*57-3625	57-3625	*58-3641	*58-3641	58-3641	*59-3701	*59-3701	59-3701
		*70-4245	*70-4245	70-4245	*71-4333	71-4339	*71-4350	71-4351	*76-4523	*76-4524
		76-4525	*79-4980	*79-4980	79-4980	*79-4980	*79-4980	79-4980	*79-4983	*79-4983
		79-4983	*82-5215	*82-5215	82-5215	*92-5977	*92-5977	92-5977	*96-6117	*96-6117
		96-6117	*96-6118	*96-6118	96-6118	*96-6119	*96-6119	96-6119	*96-6120	*96-6120
		96-6120	*96-6122	*96-6122	96-6122	*96-6123	*96-6123	96-6123	*96-6124	*96-6124
		96-6124	*96-6125	*96-6125	96-6125					
$APTHD	001100	6-1040	#6-1040							
$ASTAT	= ******	98-6190	98-6191	98-6191						
$ATYC	044244	98-6191	#98-6191							
$ATY1	044220	#98-6191								
$ATY3	044226	98-6188	#98-6191							
$ATY4	044236	98-6186	98-6190	#98-6191						
$AUTOB	001150	#8-1241								
$BASE	001256	#8-1241								
$BDADR	001136	#8-1241								
$BDDAT	001142	#8-1241	*24-2169	*24-2171	24-2172	*25-2218	25-2219	*27-2274	*27-2280	*28-2311
		*28-2312	*31-2439	31-2439	*34-2567	34-2568	*34-2574	*34-2585	34-2586	*34-2592
		*37-2714	*37-2715	37-2716	*38-2741	*38-2742	38-2743	*39-2772	*39-2773	39-2774
		*41-2865	*41-2866	41-2867	*42-2906	*42-2907	42-2908	42-2917	*43-2943	*43-2944
		43-2945	43-2962	*43-2963	*43-2968	43-2971	*43-2976	43-2978	*43-2997	*43-2998
		43-3000	*43-3005	*43-3006	43-3008	*44-3022	*44-3023	44-3026	*44-3032	44-3033
		*45-3075	*45-3076	45-3077	*46-3103	*46-3104	46-3105	*47-3138	*47-3139	47-3140
		*49-3234	*49-3235	49-3236	*50-3279	*50-3280	50-3281	50-3290	*51-3359	*51-3360
		51-3368	*53-3459	*53-3460	53-3461	*54-3499	*54-3500	54-3501	*55-3547	*55-3548
		55-3549	*56-3575	*56-3576	56-3577	*57-3609	*57-3610	57-3611	*59-3703	*59-3705
		59-3706	*60-3742	*60-3744	60-3745	60-3754	*61-3807	*61-3808	61-3814	*63-3871
		*63-3873	63-3874	*65-3957	65-3958	*66-3974	66-3975	*66-3988	*66-4001	66-4002
		*67-4029	67-4030	*68-4052	68-4053	*68-4059	68-4060	*69-4108	*69-4109	69-4110
		*69-4148	*69-4157	*69-4158	69-4160	*70-4197	*70-4289	*70-4290	70-4291	*72-4383
		72-4384	*73-4399	73-4400	*74-4456	74-4457	*74-4493	74-4494	*75-4508	75-4509
		*78-4641	78-4643	*78-4659	78-4661	*83-5258	*83-5259	*83-5340	*83-5348	*83-5364
		83-5482	83-5486	*84-5634	*86-5690	86-5721	*86-5731	*87-5753	*87-5758	87-5759
		*87-5794	87-5795	*87-5806	87-5807	*87-5834	*87-5835	*87-5836	87-5837	87-5841
		*87-5844	*89-5898	*89-5908	89-5909	*89-5922	89-5923	*92-5976	*92-5978	*92-5983
		92-5984	*94-6042	94-6043	100-6418	100-6420	100-6421	100-6422	100-6424	100-6425
		100-6427	100-6428	100-6431	100-6432	100-6434				
$CDW1	001262	#8-1241								
$CDW2	001264	#8-1241								
$CMARC	043764	*98-6188	*98-6188	98-6188	*98-6188	#98-6188				
$CKSWR	******	98-6194								
$CMTAG	001114	#8-1241	12-1620	12-1620	12-1620	12-1620	12-1620	12-1620		

SYMBOL	VALUE	REFERENCES								
$CM3	= 000000	#8-1241	8-1241							
$CNTLG	043003	#98-6184								
$CNTLU	042776	#98-6184								
$CPUOP	001230	#8-1241								
$CRLF	001177	#8-1241	71-4321	71-4357	98-6184	98-6184	98-6185	98-6185	98-6186	98-6186
		98-6186	98-6187	98-6187	98-6187	98-6187	98-6188	98-6188	98-6188	
$DBLK	044204	98-6189	98-6189	#98-6189						
$DDW0	001266	#8-1241								
$DDW1	001270	#8-1241								
$DDW10	001312	#8-1241								
$DDW11	001314	#8-1241								
$DDW12	001316	#8-1241								
$DDW13	001320	#8-1241								
$DDW14	001322	#8-1241								
$DDW15	001324	#8-1241								
$DDW2	001272	#8-1241								
$DDW3	001274	#8-1241								
$DDW4	001276	#8-1241								
$DDW5	001300	#8-1241								
$DDW6	001302	#8-1241								
$DDW7	001304	#8-1241								
$DDW8	001306	#8-1241								
$DDW9	001310	#8-1241								
$DEVCT	001212	#8-1241								
$DEVM	001260	#8-1241								
$DOAGN	006734	19-1930	19-1930	#19-1930						
$DTBL	044174	98-6189	#98-6189							
$ENDAD	006724	13-1623	#19-1930	98-6186						
$ENDCT	006710	12-1620	#19-1930							
$ENV	001222	#8-1241	13-1635	13-1637	19-1905	70-4306	98-6186	98-6188	98-6191	98-6191
$ENVM	001223	#8-1241	12-1620	98-6188	98-6188	98-6191				
$EOP	006660	#19-1930								
$ECPCT	006702	*12-1620	#19-1930	19-1930						
$ERFLG	001117	#8-1241	98-6183	98-6185	*98-6183	98-6183	98-6183	*98-6186	98-6186	98-6186
$ERMAX	001131	#8-1241	*12-1620	*98-6183	98-6183	98-6183				
$ERROR	043172	12-1620	#98-6186							
$ERPPC	001132	#8-1241	*98-6186	*98-6186	98-6186	98-6186	98-6186	98-6187	100-6417	100-6418
		100-6419	100-6420	100-6421	100-6422	100-6423	100-6424	100-6425	100-6426	100-6427
		100-6428	100-6429	100-6430	100-6431	100-6432	100-6433	100-6434	100-6435	100-6436
$ERRTB	003174	#10-1241	98-6187							
$ERRTY	043352	98-6186	#98-6187							
$ERTTL	001126	#8-1241	*98-6186	98-6186	98-6186					
$ESCAP	001174	#8-1241	*12-1620	*98-6183	98-6186	98-6186	98-6186	98-6186		
$ETABL	001222	#8-1241								
$ETEND	001326	6-1040	#8-1241							
$FATAL	001204	#8-1241	*98-6191							
$FFLG	044464	*98-6191	*98-6191	98-6191	*98-6191	#98-6191				
$FILLC	001172	#8-1241	98-6188	98-6188	98-6188					
$FILLS	001171	#8-1241	98-6188	98-6188						
$GDADR	001134	#8-1241								
$GDDAT	001140	#8-1241	*24-2170	*25-2184	*25-2188	*25-2195	*25-2199	*25-2207	*25-2211	*25-2214
		*26-2228	*26-2232	*26-2240	*26-2244	*26-2252	*26-2256	*27-2273	*27-2277	*28-2310

SYMBOL	VALUE	REFERENCES								
		*29-2342	*29-2345	*30-2364	*30-2368	*31-2439	*32-2458	*32-2461	*32-2463	*32-2466
		*33-2486	*33-2500	*33-2504	*33-2511	*34-2530	*34-2533	*34-2566	34-2568	*34-2573
		*34-2584	34-2586	*34-2591	*36-2623	*36-2626	*37-2709	37-2716	*38-2736	*38-2737
		38-2739	38-2740	38-2743	*38-2751	*39-2763	39-2774	*41-2863	41-2867	*42-2900
		*42-2901	42-2917	*43-2936	*43-2937	43-2971	43-2978	*43-2989	43-2991	43-2992
		43-2993	43-3000	43-3008	*44-3016	44-3026	44-3033	*45-3070	45-3077	*46-3097
		*46-3098	46-3101	46-3102	46-3105	*46-3114	*47-3126	47-3140	*49-3232	49-3236
		*50-3272	50-3290	*51-3310	*51-3311	51-3319	51-3320	51-3321	51-3340	51-3343
		*51-3364	*53-3426	53-3461	*54-3486	54-3501	*55-3542	55-3549	*56-3568	*56-3569
		56-3573	56-3574	56-3577	*56-3586	*57-3598	57-3611	*59-3700	59-3706	*60-3733
		*60-3734	60-3754	*61-3778	*61-3779	61-3786	61-3789	61-3790	*63-3860	63-3874
		*65-3953	65-3958	66-3975	*66-3987	*66-3994	66-3995	66-4000	66-4002	*67-4026
		67-4030	*68-4051	68-4053	*68-4058	68-4060	*69-4096	*69-4104	69-4110	69-4114
		69-4136	*69-4159	*70-4196	*70-4288	70-4291	*72-4379	72-4384	73-4400	*74-4455
		74-4457	*74-4489	74-4494	75-4509	*78-4640	78-4643	*78-4657	78-4661	*82-5227
		82-5228	*82-5228	82-5229	*82-5229	*82-5230	*82-5234	82-5238	*83-5339	*83-5347
		*83-5363	*83-5418	*83-5425	*83-5430	*83-5434	*83-5481	83-5482	*83-5485	83-5486
		*86-5715	86-5720	*86-5730	*87-5757	87-5759	*87-5776	87-5778	*87-5785	*87-5791
		*87-5792	87-5793	87-5795	*87-5799	87-5800	*87-5805	*87-5833	87-5837	*87-5840
		87-5841	*87-5846	89-5909	89-5923	90-5939	92-5975	92-5984	94-6043	*95-6078
		95-6079	*95-6091	95-6097	95-6100	*96-6117	*96-6118	*96-6119	*96-6120	*96-6122
		*96-6123	*96-6124	*96-6125	100-6418	100-6420	100-6421	100-6422	100-6424	100-6425
		100-6427	100-6428	100-6431	100-6434					
$GET42	006714	#19-1930								
$GTSWR	= ******	98-6194								
$HIBTS	001100	#6-1040								
$HIOCT	043170	*98-6185	#98-6185							
$ICNT	001120	#8-1241								
$ILLUP	045232	98-6195	98-6195	#98-6195						
$INTAG	001151	#8-1241								
$ITEMB	001130	#8-1241	*98-6186	98-6186	98-6186	98-6186	98-6187			
$LF	001200	#8-1241	98-6184	98-6184	98-6184	98-6185	98-6185	98-6186	98-6186	98-6188
		98-6188								
$LFLG	044463	*98-6191	#98-6191							
$LPADR	001122	#8-1241	*98-6183	*98-6183	98-6183	98-6183	98-6183			
$LPERR	001124	#8-1241	*12-1620	98-6183	*98-6183	98-6183	98-6183	98-6186		
$MADR1	001234	#8-1241								
$MADR2	001240	#8-1241								
$MADR3	001244	#8-1241								
$MADR4	001250	#8-1241								
$MAIL	001202	6-1040	6-1040	#8-1241	12-1620	98-6183	98-6186	98-6188		
$MAMS1	001232	#8-1241								
$MAMS2	001236	#8-1241								
$MAMS3	001242	#8-1241								
$MAMS4	001246	#8-1241								
$MBADR	001102	#6-1040								
$MFLG	044462	*98-6191	98-6191	*98-6191	#98-6191					
$MNEW	043021	#98-6184								
$MSGAD	001216	#8-1241	*98-6191	98-6191						
$MSGLG	001220	#8-1241	*98-6191							
$MSGTY	001202	#8-1241	98-6191	*98-6191	98-6191	*98-6191				
$MSWR	043010	#98-6184								

SYMBOL	VALUE	REFERENCES
$MTYP1	001233	#8-1241
$MTYP2	001237	#8-1241
$MTYP3	001243	#8-1241
$MTYP4	001247	#8-1241
$MUT	002240	#8-1241 *16-1743 *19-1875 *81-5059 *87-5750 100-6417 100-6419 100-6420 100-6421 100-6422 100-6424 100-6426 100-6428 100-6430 100-6434 100-6435
$NULL	001170	#8-1241 98-6188 98-6188 98-6188
$NWTST	= 000000	#24-2164 #25-2182 #25-2193 #25-2205 #26-2226 #26-2238 #26-2250 #27-2268 #28-2295 #29-2339 #30-2361 #31-2431 #32-2455 #33-2477 #34-2525 #36-2620 #37-2686 #45-3045 #55-3518 #65-3929 #78-4631 #79-4789 #83-5252
$OCNT	044710	*98-6192 *98-6192 #98-6192
$OMODE	044712	*98-6192 *98-6192 98-6192 *98-6192 *98-6192 #98-6192
$OVER	042446	98-6183 #98-6183
$PASS	001210	#8-1241 *12-1620 *13-1632 *13-1640 *14-1682 *16-1746 *16-1751 16-1752 *19-1909 19-1910 *19-1912 *19-1930 *19-1930 19-1930 19-1930 *20-1955 *20-1959 20-1960 71-4310 *97-6167 *97-6171 97-6172 100-6417 100-6418 100-6419 100-6420 100-6421 100-6422 100-6424 100-6425 100-6426 100-6427
$PASTM	001106	#6-1040
$POWER	045240	98-6195 #98-6195
$PWRDN	045072	#98-6195 98-6195
$PWRMG	045226	#98-6195
$PWRUP	045144	98-6195 #98-6195
$QUES	001176	#8-1241 98-6184 98-6184 98-6184 98-6185 98-6185 98-6185 98-6186 98-6186 98-6188 98-6188
$RDCHR	042540	#98-6184 98-6194 98-6194
$RDDEC	= ******	98-6194
$RDLIN	042660	#98-6184 98-6194 98-6194
$RDOCT	043032	#98-6185 98-6194 98-6194
$RDSZ	= 000010	#98-6184 98-6184
$RESRE	044752	#98-6193 98-6194
$RTNAD	006736	#19-1930
$R2A	= ******	98-6194
$SAVRE	044714	#98-6193 98-6194 98-6194
$SAVR6	045236	*98-6195 98-6195 *98-6195 *98-6195 #98-6195
$SCOPE	042322	12-1620 #98-6183
$SETUP	= 000027	#11-1618 11-1618 #11-1618 11-1618 #11-1618 11-1618 #11-1618 11-1618 #11-1618 12-1620 12-1620 12-1620 12-1620 12-1620 12-1620 12-1620 12-1620 12-1620 12-1620 12-1620 13-1623 13-1623 13-1623 19-1930 19-1930 98-6183 98-6184 98-6184 98-6186 98-6186 98-6186 98-6186
$STUP	= 177777	#11-1618 #11-1618 11-1618 #11-1618 #11-1618 11-1618 #11-1618 #11-1618 11-1618 #11-1618 #11-1618 11-1618
$SVLAD	042412	98-6183 98-6183 #98-6183
$SWR	= 121000	#2-4 8-1241 8-1241 8-1241 12-1620 12-1620 12-1620 12-1620 12-1620 19-1930 19-1930 19-1930 19-1930 19-1930 24-2164 25-2182 25-2193 25-2205 26-2226 26-2238 26-2250 27-2268 28-2295 29-2339 30-2361 31-2431 32-2455 33-2477 34-2525 36-2620 37-2686 45-3045 55-3518 65-3929 78-4631 79-4789 83-5252 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6183 98-6186 98-6186 98-6186 98-6186 98-6186 98-6186 98-6186 98-6186 98-6186 98-6186 98-6195
$SWREG	001224	#8-1241 *2-1620
$SWRMK	000000	98-6183

```
SYMBOL  VALUE           REFERENCES
$SWDBT    042462        #98-6183
$TESTN    001206        #8-1241    *24-2165   *25-2183   *25-2194   *25-2206   *26-2227   *26-2239   *26-2251   *27-2269
                        *28-2296   *29-2340   *30-2362   *31-2432   *32-2456   *33-2478   *34-2526   *36-2621   *37-2687
                        *45-3046   *55-3519   *65-3930   *78-4632   *79-4790   *83-5253   *84-5495   *86-5656   *87-5749
                         92-5979    92-5981   *98-6183    100-6417   100-6418   100-6419   100-6420   100-6421   100-6422
                         100-6423   100-6424   100-6425   100-6426   100-6434   100-6435
$TKB      001162        #8-1241    93-6019    98-6184    98-6184    98-6184    98-6184
$TKS      001160        #8-1241    13-1642    14-1659    16-1720    19-1802    20-1939    21-1984    22-2030    24-2154
                         77-4530    81-5025    81-5041    98-6184    98-6184    98-6184    98-6184
$TN     = 000030        #2-5       24-2164    24-2164   #24-2164    25-2182    25-2182   #25-2182    25-2193    25-2193
                        #25-2193    25-2205    25-2205   #25-2205    26-2226    26-2226   #26-2226    26-2238    26-2238
                        #26-2238    26-2250    26-2250   #26-2250    27-2268    27-2268   #27-2268    28-2295    28-2295
                        #28-2295    29-2339    29-2339   #29-2339    30-2361    30-2361   #30-2361    31-2431    31-2431
                        #31-2431    32-2455    32-2455   #32-2455    33-2477    33-2477   #33-2477    34-2525    34-2525
                        #34-2525    36-2620    36-2620   #36-2620    37-2686    37-2686   #37-2686    45-3045    45-3045
                        #45-3045    55-3518    55-3518   #55-3518    65-3929    65-3929   #65-3929    78-4631    78-4631
                        #78-4631    79-4789    79-4789   #79-4789    83-5252    83-5252   #83-5252    98-6183
$TPB      001166        #8-1241    98-6188    98-6188    98-6188
$TPFLG    001173        #8-1241    98-6188    98-6188    98-6188
$TPS      001164        #8-1241    98-6188    98-6188    98-6188
$TRAP     045010         12-1620   #98-6194
$TRAP2    045032        #98-6194    98-6194
$TRP    = 000013        #98-6194    98-6194    98-6194    98-6194    98-6194   #98-6194    98-6194    98-6194    98-6194
                         98-6194   #98-6194    98-6194    98-6194    98-6194    98-6194   #98-6194    98-6194    98-6194
                         98-6194    98-6194   #98-6194    98-6194    98-6194    98-6194    98-6194   #98-6194    98-6194
                         98-6194    98-6194    98-6194   #98-6194    98-6194    98-6194    98-6194    98-6194   #98-6194
                         98-6194    98-6194    98-6194    98-6194   #98-6194    98-6194    98-6194    98-6194    98-6194
                        #98-6194    98-6194    98-6194    98-6194    98-6194   #98-6194
$TRPAD    045044         98-6194   #98-6194
$TSTM     001104        #6-1040
$TSTNM    001116        #8-1241    *19-1930    98-6183   *98-6183    98-6183    98-6183    98-6183    98-6183    98-6186
                         98-6186    98-6186
$TTYIN    042766         98-6184    98-6184    98-6184   #98-6184
$TYPBN  = ******         98-6194
$TYPDS    043770        #98-6189    98-6194    98-6194
$TYPE     043506        #98-6188    98-6191    98-6194    98-6194
$TYPFC    043720         98-6188    98-6188    98-6188   #98-6188    98-6188
$TYPEX    043766         98-6188    98-6188   #98-6188
$TYPOC    044512        #98-6192    98-6194    98-6194
$TYPON    044526         98-6192   #98-6192    98-6194
$TYPOS    044466        #98-6192    98-6194
$UNIT     001214        #8-1241
$UNITM    001110        #6-1040
$USWR     001226        #8-1241
$VECT1    001252        #8-1241
$VECT2    001254        #8-1241
$XTSTR    042322        #98-6183
$$GET4  = 000000        #19-1930    19-1930
$$SW08  = 000030        #98-6183    98-6183    98-6183   #98-6183    98-6183    98-6183   #98-6183    98-6183    98-6183
                        #98-6183    98-6183    98-6183   #98-6183    98-6183    98-6183   #98-6183    98-6183    98-6183
                        #98-6183    98-6183    98-6183   #98-6183    98-6183    98-6183   #98-6183    98-6183    98-6183
                        #98-6183    98-6183    98-6183   #98-6183    98-6183    98-6183   #98-6183    98-6183    98-6183
```

SYMBOL	VALUE	REFERENCES								
		#98-6183	98-6183	98-6183	#98-6183	98-6183	98-6183	#98-6183	98-6183	98-6183
		#98-6183	98-6183	98-6183	#98-6183	98-6183	98-6183	#98-6183	98-6183	98-6183
		#98-6183	98-6183	98-6183	#98-6183	98-6183	98-6183	#98-6183	98-6183	98-6183
		#98-6183	98-6183	98-6183	#98-6183	98-6183	98-6183	#98-6183		
$OFIL	044711	*98-6192	*98-6192	98-6192	#98-6192					
$40CAT	= ******	98-6186								
.$ASTA	= ******	98-6191	98-6191							
.$K	= 001100	#6-1040	6-1040							

```
MACRO NAME      REFERENCES
CCLEAR          #6-1026     79-4980
CLRPR           #6-951      13-1643     28-2301     28-2321     30-2373     30-2413     33-2485     34-2535     34-2600     36-2629
                 36-2674    40-2800     48-3167     51-3317     52-3410     53-3438     53-3481     58-3636     61-3784     62-3850
                 70-4239    83-5256     83-5330     86-5657
COMMEN          #6-928
DELAY           #6-1033     25-2217     28-2303     30-2378     30-2415     34-2540     34-2603     36-2634     36-2676     39-2787
                 40-2804    41-2864     43-2995     47-3154     48-3171     49-3233     51-3302     53-3439     57-3625     58-3641
                 59-3701    70-4245     79-4980     79-4980     79-4983     82-5215     92-5977     96-6117     96-6118     96-6119
                 96-6120    96-6122     96-6123     96-6124     96-6125
ENDCOM          #6-928
ERROR           #6-944      16-1732     16-1774     19-1807     19-1871     21-2013     24-2174     24-2176     25-2187     25-2190
                 25-2198    25-2201     25-2210     25-2213     25-2221     26-2231     26-2234     26-2243     26-2246     26-2255
                 26-2258    27-2281     28-2304     28-2313     28-2318     28-2326     30-2379     30-2401     30-2408     30-2416
                 30-2419    31-2439     33-2492     34-2541     34-2563     34-2570     34-2576     34-2582     34-2588     34-2594
                 34-2604    34-2608     36-2635     36-2657     36-2669     36-2677     37-2719     38-2746     39-2777     40-2806
                 41-2837    41-2853     41-2861     41-2870     42-2913     42-2920     43-2950     43-2966     43-2974     43-2981
                 43-3003    43-3011     44-3030     44-3037     45-3080     46-3108     47-3143     48-3176     49-3206     49-3222
                 49-3230    49-3239     50-3286     50-3293     51-3332     51-3372     51-3378     52-3397     52-3404     52-3421
                 53-3445    53-3453     53-3464     54-3504     55-3552     56-3580     57-3614     58-3643     59-3674     59-3690
                 59-3698    59-3709     60-3750     60-3757     61-3795     61-3817     61-3822     62-3837     62-3844     62-3856
                 63-3877    65-3961     66-3978     66-4005     67-4033     68-4056     68-4063     69-4117     69-4120     69-4132
                 69-4163    70-4200     70-4294     70-4304     72-4387     73-4403     74-4460     74-4497     75-4512     78-4618
                 78-4620    78-4645     78-4663     79-4808     79-4831     79-4850     79-4870     79-4890     79-4931     83-5262
                 83-5266    83-5270     83-5274     83-5278     83-5288     83-5294     83-5298     83-5302     83-5307     83-5326
                 83-5341    83-5349     83-5360     83-5365     83-5378     83-5383     83-5396     83-5419     83-5426     83-5431
                 83-5435    83-5462     83-5476     83-5484     83-5488     84-5501     84-5631     84-5637     84-5643     86-5662
                 86-5733    86-5739     87-5761     87-5772     87-5781     87-5797     87-5809     87-5813     87-5817     87-5825
                 87-5832    87-5839     87-5843     87-5847     87-5858     89-5912     89-5926     93-6012     96-6117     96-6118
                 96-6119    96-6120     96-6122     96-6123     96-6124     96-6125     97-6155
ESCAPE          #6-928
GETPRI          #6-928
GETSWR          #6-928      #13-1623    13-1623
KBDISA          #6-1001      14-1659
KBENAB          #6-996      #13-1642    #16-1720    #19-1802    #20-1939    #21-1984    #22-2030    #24-2154    #77-4530    #81-5025
                #81-5041
MORETA          #6-1041     #8-1241
MULT            #6-928
NEWTST          #6-928      #24-2164    #25-2182    #25-2193    #25-2205    #26-2226    #26-2238    #26-2250    #27-2268    #28-2295
                #29-2339    #30-2361    #31-2431    #32-2455    #33-2477    #34-2525    #36-2620    #37-2686    #45-3045    #55-3518
                #65-3929    #78-4631    #79-4789    #83-5252
PATTST          #6-969       31-2439
POP             #6-928      #14-1662    #14-1668    #14-1709    #16-1724    #16-1733    #17-1783    #19-1850    #19-1879    #19-1880
                #19-1895    #19-1896    #19-1897    #20-1942    #20-1945    #21-1986    #24-2158    #28-2320    #77-4540    #77-4552
                #77-4562    #77-4571    #77-4596    #79-4696    #79-4725    #79-4735    #79-4755    #79-4767    #79-4776    #79-4965
                #81-5046    #81-5063    #81-5096    #82-5132    #82-5145    #82-5190    #82-5244    #87-5854    #89-5916    #89-5917
                #89-5930    #89-5931    #90-5943    #90-5944    #94-6046    #94-6047    #94-6052    #94-6053    #95-6078    #97-6150
                #97-6151    #98-6185    #98-6189    #98-6191    #98-6191    #98-6193    #98-6195    #98-6195
PUSH            #6-928      #19-1876    #19-1877    #19-1883    #19-1884    #19-1885    #24-2153    #28-2299    #71-4319    #77-4589
                #82-5129    #87-5754    #89-5906    #89-5907    #89-5920    #89-5921    #90-5937    #90-5938    #94-6040    #94-6041
                #97-6133    #97-6141    #98-6185    #98-6189    #98-6191    #98-6191    #98-6191    #98-6193    #98-6195    #98-6195
READIN          #6-1013
REPORT          #2-12       #6-928      98-6190
```

```
MACRO NAME      REFERENCES
SETPR           #6-938      37-2688     40-2808     45-3049     48-3174     51-3357     51-3383     53-3425     53-3443     53-3451
                 55-3521    58-3645     61-3804     67-4037     69-4070     70-4248
SETPRI          #6-928
SETTRA          #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194
                #98-6194
SETUP           #6-928      #12-1620
SKIP            #6-928
SLASH           #6-928
STARS           #6-928      #6-1040     #6-1040     #6-1040     #8-1241     #8-1241     #8-1241     #19-1930    #24-2164    #25-2182
                #25-2193    #25-2205    #26-2226    #26-2238    #26-2250    #27-2268    #28-2295    #29-2339    #30-2361    #31-2431
                #32-2455    #33-2477    #34-2525    #36-2620    #37-2686    #45-3045    #55-3518    #65-3929    #78-4631    #79-4789
                #83-5252    #98-6183    #98-6184    #98-6184    #98-6184    #98-6185    #98-6186    #98-6187    #98-6188    #98-6189
                #98-6191    #98-6192    #98-6193    #98-6194    #98-6195    #98-6195
SWRSU           #6-928      #12-1620     12-1620
TRMTRP          #98-6194
TSTIN           #6-964       29-2342     29-2345     30-2364     30-2368     34-2530     34-2533     36-2623     36-2626
TSTOUT          #6-957      #32-2458    #32-2461    #32-2463    #32-2466    #33-2486    #33-2504
TYPBIN          #6-928
TYPDEC          #6-928      #82-5238
TYPNAM          #6-928      #13-1623
TYPNUM          #6-928
TYPOCS          #6-928
TYPOCT          #6-928       14-1715     16-1756     16-1768     19-1915     20-1964     21-2007     22-2039     22-2082     22-2097
                 22-2113     82-5208     82-5221     82-5225     95-6095     95-6097     97-6176     98-6187     98-6187
TYPTXT          #6-928
WRTOUT          #6-1004     96-6117     96-6118     96-6119     96-6120     96-6122     96-6123     96-6124     96-6125
$$CMRE          #7-1241
$$CMTM          #7-1241
$$ESCA          #6-928
$$NEWT          #6-928       24-2164     25-2182     25-2193     25-2205     26-2226     26-2238     26-2250     27-2268     28-2295
                 29-2339     30-2361     31-2431     32-2455     33-2477     34-2525     36-2620     37-2686     45-3045     55-3518
                 65-3929     78-4631     79-4789     83-5252
$$SET           #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194    #98-6194
                #98-6194
$$SETM          #12-1620    #12-1620
$$SKIP          #6-928
.EQUAT          #2-9         6-928
.SETUP          #2-9         11-1618
.$APTB          #2-11       #8-1241      8-1241
.$APTM          #2-11       #6-1040
.$APTY          #2-11       #98-6191
.$CATC          #2-9         6-917
.$CMTA          #2-10        7-1241
.$EOP           #2-10       #19-1930
.$ERRO          #2-9         98-6186
.$ERRT          #2-10        98-6187
.$POWE          #2-10       #98-6195
.$RDOC          #2-11        98-6185
.$READ          #2-11       #98-6184
.$SAVE          #2-11       #98-6193
.$SCOP          #2-9         98-6183
.$TRAP          #2-10       #98-6194
```

CVPCAC CREATED BY MACRO ON 20-FEB-79 AT 11:26 PAGE 26
MACRO CROSS REFERENCE CREF V01 SEQ 0204

MACRO NAME	REFERENCES	
.$TYPD	#2-11	98-6189
.$TYPE	#2-9	98-6188
.$TYPC	#2-10	#98-6192