

The microfiche card displays a grid of 100 frames, arranged in 10 rows and 10 columns. Each frame contains a small table or chart with various data points, likely representing test results for different components or conditions. The data is too small to read clearly but appears to be organized in a structured manner.

100

B01

**IDENTIFICATION**

**PRODUCT CODE:** MAINDEC-11-DZICA-C-D  
**PRODUCT NAME:** ICS-11 FIELD TEST PROGRAM  
**DATE CREATED:** FEBRUARY 1977  
**MAINTAINER:** DIAGNOSTIC ENGINEERING  
**AUTHOR:** DAN DEKNIS

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1975, 1977, BY DIGITAL EQUIPMENT CORPORATION

## 0.0 TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
3.0	LOADING PROCEDURE
4.0	STARTING PROCEDURE
5.0	SWITCH REGISTER SETTINGS
6.0	PROGRAM QUESTIONS
7.0	TEST DESCRIPTIONS
8.0	SUMMARY
9.0	MISCELLANEOUS
10.0	LISTING

1.0 ABSTRACT  
\*\*\*\*\*

THIS PROGRAM ALLOWS THE USER TO CHECKOUT, DEBUG OR DEMONSTRATE THE INDUSTRIAL CONTROL SUBSYSTEM (ICS-11). THE PROGRAM IS DIVIDED INTO 8 DIFFERENT TESTS AIMED AT EXERCISING THE OPTIONS ON THE ICS-11 BUS. THE TESTS ARE SELECTED BY THE OPERATOR; ANY ADDITIONAL INFORMATION NEEDED TO RUN A PARTICULAR TEST IS REQUESTED BY THE PROGRAM.

2.0 REQUIREMENTS  
\*\*\*\*\*

2.1 EQUIPMENT  
-----

- A. PDP-11 COMPUTER WITH 8K OF MEMORY (OR MORE).
  - B. I/O TERMINAL (I.E. ASR33 TTY)
  - C. ICS-11 CONTROLLER
  - D. TO TEST INPUT MODULES; SOME FORM OF GENERATING AN INPUT (SUCH AS SWITCHES).
  - E. TO TEST OUTPUT MODULES; SOME FORM OF DETECTING AN OUTPUT (SUCH AS LIGHTS).
- NOTE: INPUT MODULES MAY BE CONNECTED TO OUTPUT MODULES
- F. TO TEST D/A MODULES; A MEANS OF MEASURING D/A OUTPUT (SUCH AS FLUKE METER) AND AN OCILLISCOPE.
  - G. TO TEST A/D MODULES; A PRECISION VOLTAGE STANDARD (SUCH AS AN EDC) IS NEEDED.

3.0 LOADING PROCEDURE  
\*\*\*\*\*

NORMAL PROCEDURE FOR LOADING AN OBJECT PROGRAM INTO CORE SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE  
\*\*\*\*\*

4.1 START ADDRESS 200  
-----

LOADING ADDRESS 200 AND STARTING, CLEARS ALL FORMERLY ENTERED INFORMATION (IF ANY) AND TYPES OUT "6.1 FILE BOX TO BE TESTED?" AND WAITS FOR A RESPONSE FROM THE OPERATOR (SEE SEC. 6.1) THEN IT WILL TYPE "6.2 ICS VECTOR ADDRESS?" (SEE SECTION 6.2). NEXT IT WILL TYPE OUT "6.3 TEST NO.?" AND WAITS FOR THE OPERATOR TO SELECT THE TEST HE WISHES TO RUN. WHEN A TEST IS SELECTED, ONE OR MORE ADDITIONAL QUESTIONS WILL BE ASKED (SEE SECTION 6.0).

4.2 RESTART ADDRESS 210  
-----

LOADING ADDRESS 210 AND STARTING, RETAINS ANY INFORMATION (IF ANY) THAT WAS PREVIOUSLY TYPED. ANY DEFAULT (OR STANDARD) ANSWERS THAT WERE OVERRIDEN BY THE OPERATOR, WILL RETAIN THE OVERRIDEN VALUE. THE PROGRAM WILL TYPE OUT "6.3" TEST NO.? AND WAIT FOR THE OPERATOR

TO SELECT A TEST TO RUN. ONCE A TEST IS SELECTED, ONE OR MORE ADDITIONAL QUESTIONS WILL BE ASKED (SEE SECTION 6.0).

## 5.0 SWITCH REGISTER SETTINGS \*\*\*\*\*

IF THE PROGRAM IS BEING RUN FROM A SWITCHLESS PROCESSOR (I.E. AN 11/34), THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SMR = NNNNN NEM ='

EACH TIME THE SETTINGS ARE ENTERED, THE ENTIRE REGISTER IMAGE MUST BE ENTERED. LEADING ZEROES ARE NOT REQUIRED. THE 'CONTROL U' FUNCTION MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY. ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE MADE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

## 5.1 SWITCH OPTIONS -----

TEST NO. -----	SMR ---	FUNCTION -----
TEST 0	SM13=1	INHIBIT TYPEOUT
TEST 1	SM00=1	RETURN TO MONITOR
TEST 2	SM04-SM13	DAC OUTPUT LEVEL
TEST 3	NONE	
TEST 4+5	SM15=1	HALT ON ERROR
	SM14=1	LOOP ON TEST
	SM13=1	INHIBIT ERROR TYPEOUT
	SM11=1	INHIBIT ITERATIONS
TEST 6	SM14-12	SELECT GAIN
	SM11=1	TYPEOUT RESULTS
	SM11=0	DISPLAY RESULTS
	SM07-10	SELECT MUX.
	SM04-07	SELECT CHAN.
TEST 7	SM00=1	RETURN TO MONITOR
	NONE	

ALL TESTS "IC" (CONTROL AND LETTER C) TYPED WILL RETURN YOU TO MONITOR. RUBOUT DELETES ENTIRE LINE THAT WAS TYPED. SWITCH REGISTER SETTINGS WILL USE EITHER THE HARDWARE SWITCH REGISTER (WHEN AVAILABLE) OR THE SOFTWARE SWITCH REGISTER (LOC. 176).

## 6.0 PROGRAM QUESTIONS \*\*\*\*\*

SINCE THIS PROGRAM HAS NO WAY OF DETERMINING THE PARTICULAR CONFIGURATION OF YOUR ICS-11, VARIOUS QUESTIONS WILL BE ASKED WHEN THE ANSWERS ARE REQUIRED BY THE PROGRAM. SOME QUESTIONS DO HAVE DEFAULT ANSWERS, IF IN DOUBT REFER TO

THE SECTION OF THE DOCUMENT THAT EXPLAINS THAT QUESTION.  
NOTE: SECTIONAL HEADERS (I.E. "6.3" TEST NO.?) REFER YOU TO SECTION OF THE DOCUMENT THAT DESCRIBES THAT QUESTION. TO UTILIZE DEFAULT PARAMETERS, TYPE A CARRIAGE RETURN.

6.1 FILE BOX TO BE TESTED?  
-----

RESPOND BY TYPING AN OCTAL NUMBER FROM 0 TO 13 FOLLOWED BY A CARRIAGE RETURN. THIS NUMBER REPRESENTS THE FILE BOX WHICH YOU WANT TO TEST

6.1.1 ERROR MESSAGE

- (1) "ILLEGAL NUMBER" - "A NON-OCTAL OR NUMBER GREATER THAN 13 WAS TYPED"
- (2) "NON-EXISTENT FILE BOX" - NO SLAVE-SYNC RESPONSE CAME FROM FILE BOX REQUESTED

6.2 ICS VECTOR ADDRESS?  
-----

RESPOND WITH THE INTERRUPT VECTOR ADDRESS OF THE FILE BOX UNDER TEST FOLLOWED BY A CARRIAGE RETURN.

6.2.1 ERROR MESSAGE

- (1) "ILLEGAL NUMBER" IS TYPED IF THE ADDRESS YOU TYPED IS NOT WITHIN THE VECTOR RANGE "234-774"
- (2) "FILE BOX INTERRUPTED AT XXXXXX -- CHECK JUMPERS"  
A MAINTENANCE INTERRUPT WAS FORCED AND THE INTERRUPT OCCURRED THRU XXXXXX. CHECK JUMPERS TO INSURE PROPER VECTOR WAS GIVEN. CONTINUE WILL RE-ASK THE QUESTION
- (3) "FILE BOX DID NOT INTERRUPT - FATAL"  
AN INTERRUPT WAS TRIED TO CHECK VECTOR ADDRESS, AND FILE BOX DID NOT INTERRUPT. CANNOT CONTINUE FROM HERE UNTIL PROBLEM IS RESOLVED

6.3 TEST NO.?  
-----

RESPOND BY TYPING A NUMBER FROM 0 TO 7 FOLLOWED BY A CARRIAGE RETURN. SEE SECTION 7 FOR MORE DETAILED INFORMATION ABOUT TESTS. ALL TESTS MAY BE TERMINATED BY TYPING "↑C" (CONTROL AND LETTER C DEPRESSED TOGETHER) OR BY SETTING SWR BIT 00=1.

TEST NO -----	TEST -----	EXIT METHOD -----
0	INPUT AND OUTPUT MODULE EXERCISER	↑C
1	INPUT OR OUTPUT MODULE EXERCISER	SW0=1, ↑C
2	DAC CALIBRATION	↑C
3	DAC INTERACTION	↑C
4	COUNTER MODULE TEST	↑C
5	A/D LOGIC TEST	↑C
6	A/D CALIBRATION	SW0=1

## 6.3.1 DEFAULT

THERE IS NO DEFAULT ANSWER FOR THIS QUESTION.

## 6.3.2 ERROR MESSAGE

"NO SUCH TEST" IS TYPED IF AN ILLEGAL TEST NUMBER IS TYPED.

6.4 INPUT MODULE ADDRESS?

RESPOND TO THIS QUESTION BY TYPING THE ADDRESS(ES)  
OF THE INPUT MODULE(S) YOU WISH TO EXERCISE  
FOLLOWED BY A CARRIAGE RETURN.  
INPUT EXPECTED IS IN OCTAL FORM.  
EXAMPLE:

20<CR>	INPUT DATA FROM MODULE IN ADDRESS 20, FILE 0
20,24<CR>	INPUT DATA FROM MODULES IN ADDRESS 20 AND 24, FILE 0
20:26,34<CR>	INPUT DATA FROM MODULES IN ADDRESSES 20,22,24,26, AND 34, FILE 0

## 6.4.1 DEFAULT

NO DEFAULTS.

## 6.4.2 ERROR MESSAGES

- (1) "TOO MANY ADDRESSES - RETYPE" IS TYPED IF YOU ENTERED MORE THAN 16 INPUT MODULE ADDRESSES (IN SAME FILE) FOR TEST 0 OR MORE THAN ONE ADDRESS FOR TEST 1.
- (2) "LAST CHARACTER TYPED NOT AN OCTAL DIGIT - RETYPE" INPUT MUST BE AN OCTAL DIGIT.
- (3) "INVALID ADDRESS: XXXXXX"  
THE PROGRAM ADDED THE BASE ADDRESS 171000 TO THE ADDRESS YOU TYPED AND TRIED TO ADDRESS IT BUT GOT NO SLAVE-SYNC RESPONSE FROM IT AND TRAPPED. YOU WILL BE ASKED TO RE-TYPE ALL ADDRESSES.
- (4) "ADDRESS XXXXXX NOT WITHIN FILE BOX RANGE" - THE ADDRESS MAY WELL EXIST BUT IT IS NOT WITHIN THE MODULE RANGE FOR THE FILE UNDER TEST YOU WILL BE ASKED TO RE-ENTER ALL ADDRESSES (REFERENCE 9.1 FOR ADDRESS RANGES).

6.5 OUTPUT MODULE ADDRESS?

RESPOND BY TYPING THE ADDRESS(ES) OF THE OUTPUT  
MODULE(S) YOU WISH TO EXERCISE FOLLOWED BY  
A CARRIAGE RETURN. EXAMPLE: SEE 6.4 FOR EXAMPLES.  
INPUT EXPECTED IS IN OCTAL FORM.

## 6.5.1 DEFAULT

NO DEFAULTS.

## 6.5.2 ERROR MESSAGES

- (1) "TOO MANY ADDRESSES - RETYPE" IS TYPED IF YOU

ENTERED MORE THAN 16 INPUT MODULE ADDRESSES (IN SAME <sup>HQ1</sup> FILE) FOR TEST 0, OR MORE THAN ONE ADDRESS FOR TEST 1.

- (2) "LAST CHARACTER TYPED NOT AN OCTAL DIGIT - RETYPE"  
INPUT MUST BE AN OCTAL DIGIT.
- (3) "INVALID ADDRESS: XXXXXX"  
THE PROGRAM ADDED THE BASE ADDRESS 171000 TO THE ADDRESS YOU TYPED AND TRIED TO ADDRESS IT BUT GOT NO SLAVE-SYNC RESPONSE FROM IT AND TRAPPED. YOU WILL BE ASKED TO RE-TYPE ALL ADDRESSES.
- (4) "ADDRESS XXXXXX NOT WITHIN FILE BOX RANGE" - THE ADDRESS MAY WELL EXIST BUT IT IS NOT WITHIN THE MODULE RANGE FOR THE FILE UNDER TEST YOU WILL BE ASKED TO RE-ENTER ALL ADDRESSES (REFERENCE 9.1 FOR ADDRESS RANGES).

## 6.6 COUNTER MODULE ADDRESSES?

RESPOND BY THE ADDRESS(ES) OF THE COUNTER MODULE(S) TO BE TESTED, FOLLOWED BY A CARRIAGE RETURN. EXAMPLE: SEE 6.4 FOR EXAMPLES. INPUT EXPECTED IS IN OCTAL FORM.

### 6.6.1 DEFAULT

NO DEFAULTS.

### 6.6.2 ERROR MESSAGES

- (1) "TOO MANY ADDRESSES - RETYPE" IS TYPED IF YOU ENTER MORE THAN 16 ADDRS.
- (2) "LAST CHARACTER TYPED NOT AN OCTAL DIGIT - RETYPE"  
INPUT MUST BE AN OCTAL DIGIT.
- (3) "INVALID ADDRESS: XXXXXX"  
THE PROGRAM ADDED THE BASE ADDRESS 171000 TO THE ADDRESS YOU TYPED AND TRIED TO ADDRESS IT BUT GOT NO SLAVE-SYNC RESPONSE FROM IT AND TRAPPED. YOU WILL BE ASKED TO RE-TYPE ALL ADDRESSES.
- (4) "ADDRESS XXXXXX NOT WITHIN FILE BOX RANGE" - THE ADDRESS MAY WELL EXIST BUT IT IS NOT WITHIN THE MODULE RANGE FOR THE FILE UNDER TEST YOU WILL BE ASKED TO RE-ENTER ALL ADDRESSES (REFERENCE 9.1 FOR ADDRESS RANGES).

## 6.7 DAC ADDRESS?

RESPOND BY TYPING THE ADDRESS OF THE DAC YOU WISH TO EXERCISE FOLLOWED BY A CARRIAGE RETURN. INPUT EXPECTED IS IN OCTAL FORM.

### 6.7.1 DEFAULT

NO DEFAULTS.

### 6.7.2 ERROR MESSAGES

- (1) "TOO MANY ADDRESSES - RETYPE" IS TYPED IF YOU



TRY TO ENTER MORE THAN ONE ADDRESS.

- (2) "LAST CHARACTER TYPED NOT AN OCTAL DIGIT - RETYPE"  
INPUT MUST BE AN OCTAL DIGIT.
- (3) "INVALID ADDRESS: XXXXXX"  
THE PROGRAM ADDED THE BASE ADDRESS 171000 TO THE  
ADDRESS YOU TYPED AND TRIED TO ADDRESS IT BUT GOT  
NO SLAVE-SYNC RESPONSE FROM IT AND TRAPPED. YOU  
WILL BE ASKED TO RE-TYPE ALL ADDRESSES.
- (4) "ADDRESS XXXXXX NOT WITHIN FILE BOX RANGE" - THE  
ADDRESS MAY WELL EXIST BUT IT IS NOT WITHIN  
THE MODULE RANGE FOR THE FILE UNDER TEST  
YOU WILL BE ASKED TO RE-ENTER ALL ADDRESSES  
(REFERENCE 9.1 FOR ADDRESS RANGES).

## 6.8 ADD5 ADDRESS?

RESPOND BY TYPING THE ADDRESS OF THE ADD5, OR THE ADDRESS  
OF THE ADD5 CONNECTED TO THE ADD7 YOU WISH TO EXERCISE,  
FOLLOWED BY A CARRIAGE RETURN.  
INPUT EXPECTED IS IN OCTAL FORM.

### 6.8.1 DEFAULT

NO DEFAULTS.

### 6.8.2 ERROR MESSAGES

- (1) "TOO MANY ADDRESSES - RETYPE" IS TYPED IF YOU  
TRIED TO ENTER MORE THAN ONE ADDRESS.
- (2) "LAST CHARACTER TYPED NOT AN OCTAL DIGIT - RETYPE"  
INPUT MUST BE AN OCTAL DIGIT.
- (3) "INVALID ADDRESS: XXXXXX"  
THE PROGRAM ADDED THE BASE ADDRESS 171000 TO THE  
ADDRESS YOU TYPED AND TRIED TO ADDRESS IT BUT GOT  
NO SLAVE-SYNC RESPONSE FROM IT AND TRAPPED. YOU  
WILL BE ASKED TO RE-TYPE ALL ADDRESSES.

## 6.9 DELAY TIME (IN MILLISE)?

RESPOND BY TYPING THE DELAY TIME YOU WISH THE PROGRAM  
TO USE IN BETWEEN OUTPUTTING DATA TO OUTPUT MODULES,  
FOLLOWED BY A CARRIAGE RETURN. FRACTIONAL TIMES NOT  
ALLOWED. NOTE 1 SECOND EQUALS 1000 MILLISEC.  
A SECONDARY DELAY TIME IS AVAILABLE FOR USE IN TEST 0.  
SEE SECTION 8 FLAG MODE OPERATION.  
INPUT EXPECTED IS IN DECIMAL FORM, MAXIMUM DELAY TIME=10  
SECONDS.

### 6.9.1 DEFAULT

- (1) FOR ANY START OR RESTART - 3 MILLISEC IS DEFAULT SINCE  
MOST (BUT NOT ALL) OUTPUT MODULES HAVE A RESPONSE  
TIME OF APPROXIMATELY 3 MILLISECONDS.  
NOTE: DELAY TIMES ARE CALCULATED USING MACHINE  
INSTRUCTION TIME LOOPS - THEY MAY VARY BETWEEN  
PROCESSORS.

6.10 PATTERN MODIFIER, PATTERN?

THIS IS A TWO PART QUESTION, RESPOND BY TYPING A SINGLE DIGIT REPRESENTING THE PATTERN MODIFIER, FOLLOWED BY A "," (COMMA), FOLLOWED BY THE STARTING PATTERN YOU WISH TO USE (0 TO 6 DIGITS) FOLLOWED BY A CARRIAGE RETURN. INPUTS EXPECTED ARE IN OCTAL FORM.

EXAMPLE:  
 6.10 PATTERN MODIFIER, PATTERN ? 0,100000<CR>  
 WOULD GIVE AN INCREMENTING PATTERN, WITH STARTING PATTERN OF "100000".

PATTERN MODIFIER	FUNCTION
0	INCREMENT PATTERN
1	DECREMENT PATTERN
2	NO CHANGED OF PATTERN
3	ROTATE LEFT PATTERN
4	ROTATE RIGHT PATTERN
5	RANDOM PATTERN
6	ARITH. SHIFT LEFT PATTERN
7	COMPLEMENT PATTERN

6.10.1 DEFAULT

(1) FOR ANY START OR RESTART - STARTING PATTERN OF ALL ZEROS, PATTERN MODIFIER OF ZERO.

6.11 INPUT OR OUTPUT MODULE (I OR O)?

THIS QUESTION IS ASKED BY TEST 1 WHERE ONLY AN INPUT OR AN OUTPUT MODULE IS EXERCISED. RESPOND BY TYPING AN "I" FOR INPUT MODULE, OR "O" FOR OUTPUT MODULE. A CARRIAGE RETURN IS REQUIRED AFTER THE I OR O.

6.11.1 DEFAULTS

NONE

6.11.2 ERROR MESSAGES

IF NEITHER AN "I" NOR AN "O" WAS TYPED THAN THE QUESTION WILL BE RETYPED.

6.12 UNIPOLAR OR BIPOLAR (U OR B)

RESPOND BY TYPING A "U" OR A "B" FOLLOWED BY A CARRIAGE RETURN. UNIPOLAR REFERS TO AN UNSIGNED A/D. THE STANDARD A/D IS BIPOLAR, THAT IS, ITS RESULTS ARE SIGNED.

6.12.1 DEFAULTS

ON CARRIAGE RETURN, IT IS ASSUMED BIPOLAR.

6.12.2 ERROR MESSAGES

NONE

6.13 GAIN?

THIS QUESTION IS ASKED BY TEST 7. RESPOND BY TYPING THE GAIN YOU WISH THE SAMPLES TO BE TAKEN AT. LEGAL GAINS ARE: 1,2,10,20,50,100,200, AND 1000

K01

6.13.1 DEFAULTS

NO DEFAULTS.

6.13.2 ERROR MESSAGES

"NO SUCH GAIN" IS TYPED IF THE GAIN YOU TYPED ISN'T LEGAL.

6.14 CHANS (SC,EC)?

THIS QUESTION IS ASKING YOU FOR THE CHANNELS YOU WISH THE SAMPLES TO BE TAKEN ON. SC REPRESENTS THE STARTING CHANNEL AND EC REPRESENT THE END CHANNEL. CHANNELS MUST BE SAMPLED IN CONSECUTIVE ORDER. CPU BASED ON CORE AVAILABLE TO STORE SAMPLES AWAY IN. RESPOND BY TYPING THE STARTING CHANNEL NUMBER, FOLLOWED BY A COMMA, FOLLOWED BY THE END CHANNEL NUMBER, FOLLOWED BY A CARRIAGE RETURN. TO SAMPLE ONLY ONE CHANNEL, SIMPLY TYPE THAT CHANNEL NUMBER FOLLOWED BY A CARRIAGE RETURN.

6.14.1 DEFAULTS

- (1) AT LOAD AND START AT ADDR. 200, CHAN. 0 WILL BE SELECTED.
- (2) AT ANY OTHER TIME, PREVIOUSLY TYPED CHANNEL(S).

6.14.2 ERROR MESSAGES

"ERROR! STARTING CHAN. GREATER THAN END CHAN." IS TYPED WHEN THAT CONDITION IS TRUE.

"ERROR! NO SUCH CHAN" IS TYPED IF THE CHAN NUMBER IS TOO LARGE.

6.15 EXPECTED AVERAGE?

RESPOND BY TYPING THE AVERAGE (IN OCTAL) WHICH YOU EXPECT THE SAMPLES TO AVERAGE, FOLLOWED BY A CARRIAGE RETURN. THIS QUESTION MAY NEED NOT TO BE ANSWERED - SEE WRITE FOR TEST-7.

6.15.1 DEFAULT

DEFAULT OF 0000 OR PREVIOUSLY TYPED DATA.

6.15.2 ERROR MESSAGES

"NUMBER TOO LARGE-MAX=7777" SELF EXPLAINATORY THE A005 IS ONLY A 12 BIT CONVERTER.

6.16 TOLERANCE?

RESPOND BY TYPING THE TOLERANCE FOLLOWED BY A CARRIAGE RETURN. A TOLERANCE OF ZERO WILL FORCE TYPEOUT OF THE RESULTS OF TEST 7. TOLERANCE MAY BE ANY NUMBER, HOWEVER, IT IS RECOMMENDED YOU READ TEST 7, AND EXAMINE

THE SPECIFICATIONS FOR THE ADOCS TO DETERMINE THIS NUMBER. L01

6.16.1 DEFAULTS

ZERO OR PREVIOUSLY TYPED TOLERANCE.

6.16.2 ERROR MESSAGES

NONE

7.0 TEST DESCRIPTIONS

\*\*\*\* \*\*\*\*\*

7.1 TEST 0

INPUT AND OUTPUT MODULE EXERCISER

THIS TEST IS DESIGNED TO EXERCISE UP TO 16 OUTPUT MODULES AND 16 INPUT MODULES. IT (1) OUTPUTS THE PATTERN TO ALL OUTPUT MODULES, (2) DELAYS THE SPECIFIED DELAY TIME, (3) MODIFIES THE PATTERN, (4) SAMPLES THE INPUT MODULES FOR CHANGE OF DATA. IF A CHANGE OF DATA HAS OCCURRED, IT STARTS TYPEOUT OF THE CHANGE. IF AN INPUT MODULE INTERRUPTS, ITS CHANGE OF DATA IS TYPED ALONG WITH ITS GENERIC CODE. THE GENERIC CODE FOR A STANDARD INPUT MODULE IS "3". (5) IT NEXT DELAYS A SECOND TIME SPECIFIED BY THE SECONDARY DELAY TIME (IF ANY). SECONDARY DELAY TIME SET BY "10" USED FOR EXERCISING THE M6870 SINGLE SHOT OUTPUT MODULE. (6) REPEAT STEPS 1-5 IF "1J" WAS TYPED (SEE SECTION 8.5) INPUT AND OUTPUT MODULES WILL BE ASSUMED CONNECTED. THE ONLY TYPEOUT WILL BE IF THE DATA SENT OUT DOESN'T MATCH THE DATA RECEIVED.

7.1.1 RUN TIME TEST 0

INDEFINITE - RUN TERMINATED BY OPERATOR.

7.2 TEST 1

INPUT OR OUTPUT MODULE EXERCISER

(1) FOR INPUT MODULES:  
READS THE INPUT MODULE CONTINUOUSLY AND DISPLAYS ITS CONTENTS IN RO WITH USE OF THE RESET INSTRUCTION. IF YOU HAVE AN 11/05 PROCESSOR YOU MUST USE TEST 0.

(2) FOR OUTPUT MODULES:  
TAKES THE CONTENTS OF THE SWITCH REGISTER AND SENDS IT TO THE OUTPUT MODULE, DELAYS SPECIFIED DELAY TIME AND REPEATS.

7.2.1 RUN TIME TEST 1

INDEFINITE - RUN TERMINATED BY OPERATOR.

# MO1

- 7.3 TEST 2 DAC CALIBRATION  
OUTPUTS THE CONTENTS OF THE SWITCH REGISTER TO ALL FOUR CHANNELS OF THE DAC SPECIFIED IN ORDER TO MAINTAIN A CALIBRATION LEVEL.
- 7.3.1 RUN TIME TEST 2  
INDEFINITE - RUN TERMINATED BY OPERATOR.
- 7.4 TEST 3 DAC INTERACTION  
OUTPUTS A RAMP TO ALL FOUR CHANNELS TO THE SPECIFIED DAC. THESE RAMPs ARE "OUT OF PHASE" WITH EACH OTHER SO THAT INTERACTION AND DUAL ADDRESSING CAN BE TESTED.
- 7.4.1 RUN TIME TEST 3  
INDEFINITE - RUN TERMINATED BY OPERATOR.
- 7.5 TEST 4 COUNTER MODULE TEST  
THIS TEST CHECKS OUT BASIC LOGIC FUNCTIONS OF THE COUNTER MODULE. THE USER MUST JUMPER TP3 AND TP4 ON THE M7440.
- 7.5.1 RUN TIME TEST 4  
SHORT PASS (SW11=1) APP. 15 SEC.  
LONG PASS (SW11=0) APP. 5 MIN.  
FIRST PASS IS ALWAYS A SHORT PASS.
- 7.6 TEST 5 A/D LOGIC TEST  
THIS TEST CHECKS OUT BASIC LOGIC FUNCTIONS OF THE A/D.
- 7.6.1 RUN TIME TEST 5  
SHORT PASS (SW11=1) APP. 15 SEC.  
LONG PASS (SW11=0) APP. 1 MIN.  
FIRST PASS IS ALWAYS A SHORT PASS.
- 7.7 TEST 6 A/D CALIBRATION  
THIS TEST ALLOWS THE USER TO CALIBRATE THE A/Ds.
- 7.7.1 RUN TIME TEST 6  
INDEFINITE - RUN TERMINATED BY OPERATOR.
- 7.8 TEST 7  
(1) REPEATIBILITY  
THIS TEST ALLOWS THE USER TO TEST THE REPEATIBILITY OF NUMBER OF CHANNELS AT ANY GAIN AND INPUT VOLTAGE. THE TEST MAY OR MAY NOT PRINT OUT A TABLE

NO 1

(SEE EXAMPLE OF PRINT-OUT) DEPENDENT ON WHETHER AN ERROR OCCURRED OR IF FORCED TYPEOUT IS DESIRED. 256 SAMPLES ARE TAKEN PER CHANNEL PER PASS. "REPEAT" IS TYPED AT THE BEGINNING OF EACH PASS. WHEN THIS TEST IS SELECTED, THE ADDR ADDR OF THE A/D SUBSYSTEM IS REQUESTED ALONG WITH WHAT CHANNELS YOU WISH TO SAMPLE, THE GAIN YOU WISH TO USE, FOLLOWED BY THE EXPECTED AVERAGE (OPTIONAL) AND THE TOLERANCE.

IF FORCED TYPEOUT IS DESIRED, A TOLERANCE OF ZERO SHOULD BE TYPED. IF THE EXPECTED AVERAGE IS KNOWN RUN ONE PASS AT ANY EXPECTED AVERAGE AND THE CURRENT AVERAGE WILL BE TYPED OUT.

- THE TEST OPERATES IN THE FOLLOWING MANNER:
- (1) IT TAKES 256 SAMPLES ON EACH CHANNEL SPECIFIED.
  - (2) IT COMPUTES HIGH, LOW, AND AVERAGE OF SAMPLES FOR EACH CHANNEL.
  - (3) IT COMPARES THE HIGH, LOW, AND AVERAGE AGAINST THE EXPECTED AVERAGE YOU TYPED. IF THERE IS ANY ERROR THE ERROR WILL BE TYPED FOR THAT PARTICULAR CHANNEL, IF THERE ARE NO ERRORS, THERE WILL BE NO TYPEOUT EXCEPT FOR "REPEAT" AT THE BEGINNING OF THE NEXT PASS.
  - (4) IF A TOLERANCE OF ZERO IS SPECIFIED, A FORCED TYPEOUT WILL OCCUR OF THE RESULTS OF ALL CHANNELS SPECIFIED.

7.8.1 RUN TIME TEST 7

DEPENDENT ON NUMBER OF CHANNELS TO BE SAMPLED. 1 PASS FOR 1 CHANNEL TAKES APPROXIMATELY 2 MINUTES. IF MULTIPLE CHANNELS, TIME BETWEEN TYPEOUTS SHOULD NOT EXCEED 2 MINUTES.

7.8.2 EXAMPLE OF TYPEOUT

LO	-5	-4	-3	-2	-1	AV	+1	+2	+3	+4	+5	HI
0000	0000	0000	0000	0004	0006	0240	0006	0000	0000	0000	0000	0000

8.0 FLAG MODE OPERATION

8.01 SUMMARY

FLAG	TESTS AFFECTED	SECTION
↑E	ALL	8.1
↑N	ALL	8.2
↑D	0	8.3
↑L	7	8.4
↑J	0	8.5

ALL FLAG MODES ARE ENABLED BY TYPING THE ASSOCIATED LETTER AND THE CONTROL KEY TOGETHER AT "6.3 TEST NO?". AFTER THE FLAG IS ACTED UPON, PROGRAM CONTROL WILL RETURN TO "6.3 TEST NO?".

8.1 ↑E EXPERT MODE

WHEN "↑E" IS TYPED EXPERT MODE WILL BE ENABLED. WHEN THE PROGRAM IS OPERATING IN "EXPERT MODE" NO QUESTIONS WILL BE TYPED, ONLY THE QUESTION MARKS.

8.2 IN NOVICE MODE

WHEN "IN" IS TYPED, "EXPERT MODE" WILL BE DISABLED. ALL QUESTIONS WILL BE TYPED.

8.3 ID SECONDARY DELAY

"ID" IS TYPED IN ORDER TO ENTER A SECONDARY DELAY USED IN TEST 0. THIS DELAY IS ONLY NEEDED FOR TESTING A SINGLE SHOT MODULE WHEN RUNNING TEST0 WITH "IJ" OPTION.

8.4 IL LINE PRINTER OPTION

WHEN A LINE PRINTER IS AVAILABLE FOR USE AS AN OUTPUT DEVICE FOR TEST 7, YOU MAY SELECT OUTPUT TO GO TO IT BY TYPING "IL".

WHEN THE TEST IS STARTED AT LOC 200, THE PROGRAM WILL TYPE "LINE PRINTER OPTION AVAILABLE" IF IT DETECTS A LINE PRINTER ON THE SYSTEM. IF "IL" IS TYPED, AND NO LINE PRINTER IS AVAILABLE, THE COMMAND WILL BE IGNORED. IF "IL" IS HONORED, THE PROGRAM WILL TELL YOU TO MAKE THE PRINTER READY. TO DO THIS, MAKE SURE ITS POWER IS ON AND IT IS SELECTED.

8.5 IJ CONNECTED MODE FOR TEST 0

"IJ" INDICATES TO THE PROGRAM THAT ALL INPUT AND OUTPUT MODULES EXERCISED BY TEST 0 ARE CONNECTED TO EACH OTHER FOR TEST. NO TYPE OUT WILL OCCUR IF THE DATA SENT OUT MATCHES THE DATA RECEIVED.

9.0 MISCELLANEOUS

9.1 FILE BOX ASSIGNMENT

FILE BOX	ICSR	ICAR	MODULE RANGE	VECTOR ADDRESS
0	171776	171774	171000-171036	234
1	171766	171764	171040-171076	-----
2	171756	171754	171100-171136	↑
3	171746	171744	171140-171176	↑
4	171736	171734	171200-171236	↑
5	171726	171724	171240-171276	↑
6	171716	171714	171300-171336	↑
7	171706	171704	171340-171376	↑
10	171676	171674	171400-171436	↑
11	171666	171664	171440-171476	↑
12	171656	171654	171500-171536	↑
13	171646	171644	171540-171576	-----

↑  
↑  
↑  
↑  
↑  
↑  
↑  
↑  
↑  
↑  
↑

10. LISTING

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40  
DZICAC.NE2 21-MAR-77 15:28 TABLE OF CONTENTS

21	TRAP CATCHER
30	STARTING ADDRESS(ES)
37	BASIC DEFINITIONS
148	COMMON TAGS
192	ERROR POINTER TABLE
454	INITIALIZE THE COMMON TAGS
2791	SAVE AND RESTORE RD-R5 ROUTINES
2840	RANDOM NUMBER GENERATOR ROUTINE
2879	SINGLE LENGTH BINARY TO DECIMAL ASCII ROUTINE
2899	DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
2968	ERROR HANDLER ROUTINE
3002	ERROR MESSAGE TYPEOUT ROUTINE
3052	BINARY TO OCTAL (ASCII) AND TYPE
3131	POWER DOWN AND UP ROUTINES
3181	SCOPE HANDLER ROUTINE
3232	TTY INPUT ROUTINE
3377	READ A DECIMAL NUMBER FROM THE TTY
3442	TYPE ROUTINE
3518	TRAP DECODER
3544	TRAP TABLE



1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56

164000  
000000  
  
000001  
000000  
  
000000  
  
000174 000174  
000176 000000  
  
000200 000137 001704  
  
000210 000210 000137 003374  
  
000214  
  
001100  
  
000011  
000012  
000015  
000200  
177776  
  
177774  
177772  
177570  
177570  
  
000000  
000001

```
SSWR=164000
.HEADER /MAINDEC-11-DZICA-C/, 1977, /DAN DEKNIS/
.TITLE MAINDEC-11-DZICA-C
;#COPYRIGHT (C) 1977
;#DIGITAL EQUIPMENT CORP.
;#MAYNARD, MASS. 01754
;#
;#PROGRAM BY DAN DEKNIS
;#
;#THIS PROGRAM WAS ASSEMBLED USING THE POP-11 MAINDEC SYSMAC
;#PACKAGE (MAINDEC-11-DZICAC-C3), JAN 19, 1977.
;#
$TN=1

.SBTTL .SCATCH START
.SBTTL TRAP CATCHER

.=0
;#ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;#SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;#LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
.=174
DISPREG: .WORD 0 ;:SOFTWARE DISPLAY REGISTER
SWREG: .WORD 0 ;:SOFTWARE SWITCH REGISTER
.SBTTL STARTING ADDRESS(ES)
JMP @#START ;:JUMP TO STARTING ADDRESS OF PROGRAM

.=210
JMP @#START1

.EQUAT
.SBTTL BASIC DEFINITIONS
;#INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR ;:BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;:BASIC DEFINITION OF SCOPE CALL

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

;#GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0 ;:GENERAL REGISTER
R1= %1 ;:GENERAL REGISTER
```

57 000002  
58 000003  
59 000004  
60 000005  
61 000006  
62 000007  
63 000006  
64 000007  
65  
66  
67 000000  
68 000040  
69 000100  
70 000140  
71 000200  
72 000240  
73 000300  
74 000340  
75  
76  
77 100000  
78 040000  
79 020000  
80 010000  
81 004000  
82 002000  
83 001000  
84 000400  
85 000200  
86 000100  
87 000040  
88 000020  
89 000010  
90 000004  
91 000002  
92 000001  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105 100000  
106 040000  
107 020000  
108 010000  
109 004000  
110 002000  
111 001000  
112 000400

R2= %2  
R3= %3  
R4= %4  
R5= %5  
R6= %6  
R7= %7  
SP= %6  
PC= %7

::: GENERAL REGISTER  
::: GENERAL REGISTER  
::: GENERAL REGISTER  
::: GENERAL REGISTER  
::: GENERAL REGISTER  
::: GENERAL REGISTER  
::: STACK POINTER  
::: PROGRAM COUNTER

.\*PRIORITY LEVEL DEFINITIONS

PR0= 0  
PR1= 40  
PR2= 100  
PR3= 140  
PR4= 200  
PR5= 240  
PR6= 300  
PR7= 340

::: PRIORITY LEVEL 0  
::: PRIORITY LEVEL 1  
::: PRIORITY LEVEL 2  
::: PRIORITY LEVEL 3  
::: PRIORITY LEVEL 4  
::: PRIORITY LEVEL 5  
::: PRIORITY LEVEL 6  
::: PRIORITY LEVEL 7

.\*"SWITCH REGISTER" SWITCH DEFINITIONS

SW15= 100000  
SW14= 40000  
SW13= 20000  
SW12= 10000  
SW11= 4000  
SW10= 2000  
SW09= 1000  
SW08= 400  
SW07= 200  
SW06= 100  
SW05= 40  
SW04= 20  
SW03= 10  
SW02= 4  
SW01= 2  
SW00= 1

.EQUIV SW09, SW9  
.EQUIV SW08, SW8  
.EQUIV SW07, SW7  
.EQUIV SW06, SW6  
.EQUIV SW05, SW5  
.EQUIV SW04, SW4  
.EQUIV SW03, SW3  
.EQUIV SW02, SW2  
.EQUIV SW01, SW1  
.EQUIV SW00, SW0

.\*DATA BIT DEFINITIONS (BIT00 TO BIT15)

BIT15= 100000  
BIT14= 40000  
BIT13= 20000  
BIT12= 10000  
BIT11= 4000  
BIT10= 2000  
BIT09= 1000  
BIT08= 400

113 000200  
114 000100  
115 000040  
116 000020  
117 000010  
118 000004  
119 000002  
120 000001  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133 000004  
134 000010  
135 000014  
136 000014  
137 000014  
138 000020  
139 000024  
140 000030  
141 000034  
142 000060  
143 000064  
144 000240  
145 000214

BIT07= 200  
BIT06= 100  
BIT05= 40  
BIT04= 20  
BIT03= 10  
BIT02= 4  
BIT01= 2  
BIT00= 1  
.EQUIV BIT09,BIT9  
.EQUIV BIT08,BIT8  
.EQUIV BIT07,BIT7  
.EQUIV BIT06,BIT6  
.EQUIV BIT05,BIT5  
.EQUIV BIT04,BIT4  
.EQUIV BIT03,BIT3  
.EQUIV BIT02,BIT2  
.EQUIV BIT01,BIT1  
.EQUIV BIT00,BIT0

::BASIC "CPU" TRAP VECTOR ADDRESSES  
ERRVEC= 4 :: TIME OUT AND OTHER ERRORS  
RESVEC= 10 :: RESERVED AND ILLEGAL INSTRUCTIONS  
TBITVEC= 14 :: "T" BIT  
TRTVEC= 14 :: TRACE TRAP  
BPTVEC= 14 :: BREAKPOINT TRAP (BPT)  
IOTVEC= 20 :: INPUT/OUTPUT TRAP (IOT) \*\*SCOPE\*\*  
PMRVEC= 24 :: POWER FAIL  
EMTVEC= 30 :: EMULATOR TRAP (EMT) \*\*ERROR\*\*  
TRAPVEC= 34 :: "TRAP" TRAP  
TKVEC= 60 :: TTY KEYBOARD VECTOR  
TPVEC= 64 :: TTY PRINTER VECTOR  
PIRQVEC= 240 :: PROGRAM INTERRUPT REQUEST VECTOR  
.SCMTAG

146  
147  
148 000214  
149  
150  
151  
152  
153 001100  
154 001100  
155 001100 000000  
156 001102 000  
157 001103 000  
158 001104 000000  
159 001106 000000  
160 001110 000000  
161 001112 000000  
162 001114 000  
163 001115 001  
164 001116 000000  
165 001120 000000  
166 001122 000000  
167 001124 000000  
168 001126 000000  
169 001130 000000  
170 001132 000000  
171 001134 000  
172 001135 000  
173 001136 000000  
174 001140 177570  
175 001142 177570  
176 001144 177560  
177 001146 177562  
178 001150 177564  
179 001152 177566  
180 001154 000  
181 001155 002  
182 001156 012  
183 001157 000  
184 001160 000000  
185 001162 077  
186 001163 015  
187 001164 000012  
188 001166  
189

.SBTTL COMMON TAGS

STARS

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

SCHTAG: .=1100

SPASS: .WORD 0  
STSTM: .BYTE 00  
SERFLG: .BYTE 00  
SICNT: .WORD 00  
SLPADR: .WORD 00  
SLPERR: .WORD 00  
SERTTL: .WORD 00  
SITEMB: .BYTE 0  
SERMAX: .BYTE 1  
SERRPC: .WORD 0  
SGOADR: .WORD 0  
SBDADR: .WORD 0  
SGDDAT: .WORD 0  
SBDAT: .WORD 0  
SAUTOB: .BYTE 0  
SINTAG: .BYTE 0  
SMR: .WORD DSWR  
DISPLAY: .WORD DDISP  
STKS: 177560  
STKB: 177562  
STPS: 177564  
STPB: 177566  
SNUL: .BYTE 0  
SFILLS: .BYTE 2  
SFILLC: .BYTE 12  
STPFLG: .BYTE 0  
STIMES: 0  
SQUES: .ASCII /?/  
SCRLF: .ASCII <15>  
SLF: .ASCIZ <12>

;; START OF COMMON TAGS  
;; CONTAINS PASS COUNT  
;; CONTAINS THE TEST NUMBER  
;; CONTAINS ERROR FLAG  
;; CONTAINS SUBTEST ITERATION COUNT  
;; CONTAINS SCOPE LOOP ADDRESS  
;; CONTAINS SCOPE RETURN FOR ERRORS  
;; CONTAINS TOTAL ERRORS DETECTED  
;; CONTAINS ITEM CONTROL BYTE  
;; CONTAINS MAX. ERRORS PER TEST  
;; CONTAINS PC OF LAST ERROR INSTRUCTION  
;; CONTAINS ADDRESS OF 'GOOD' DATA  
;; CONTAINS ADDRESS OF 'BAD' DATA  
;; CONTAINS 'GOOD' DATA  
;; CONTAINS 'BAD' DATA  
;; RESERVED--NOT TO BE USED  
;; AUTOMATIC MODE INDICATOR  
;; INTERRUPT MODE INDICATOR  
;; ADDRESS OF SWITCH REGISTER  
;; ADDRESS OF DISPLAY REGISTER  
;; TTY KBD STATUS  
;; TTY KBD BUFFER  
;; TTY PRINTER STATUS REG. ADDRESS  
;; TTY PRINTER BUFFER REG. ADDRESS  
;; CONTAINS NULL CHARACTER FOR FILLS  
;; CONTAINS # OF FILLER CHARACTERS REQUIRED  
;; INSERT FILL CHARS. AFTER A "LINE FEED"  
;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)  
;; MAX. NUMBER OF ITERATIONS  
;; QUESTION MARK  
;; CARRIAGE RETURN  
;; LINE FEED

\*\*\*\*\*

.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.  
;;NOTE1: IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC)  
;;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

SERRTB:

190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245

001166

;A005 ERRORS  
;ERROR 1 ADDRESS ABILITY  
EM1 ::"COULD NOT SEND-RECIEVE DATA"  
DH1 ::ERROR A005  
::PC ADDR  
DT1 ::SERRC, ADCSR,0  
DF0 ::0  
  
;ERROR 2 "BIT EXERCISER"  
EM2 ::"SEND-RECIEVE DATA ERROR"  
DH2 ::ERROR A/D GOOD BAD  
::PC ADDR DATA DATA  
DT2 ::SERRPC, ADCSR, SGDDAT, SBDDAT,0  
DF0 ::0  
  
;ERROR 3 CONVERT BIT  
EM3 ::CONVERT BIT FAILED TO SET  
DH1 ::ERROR A/D  
DT1 ::PC ADDR  
DF0 ::SERRPC, ADCSR,0  
  
;ERROR 4 CONVERT BIT  
EM4 ::CONVERT BIT FAILED TO CLEAR  
DH1 ::ERROR A/D  
DT1 ::PC ADDR  
DF0 ::SERRPC, ADCSR,0  
  
;ERROR 5 READ C/R BIT  
EM5 ::CAN'T READ DATA REGISTER  
DH1 ::ERROR A/D  
DT1 ::PC ADDR  
DF0 ::SERRPC, ADCSR,0  
  
;ERROR 6 A005 INTERRUPT  
EM6 ::A005 FAILED TO INTERRUPT  
DH1 ::ERROR A/D  
DT1 ::PC ADDR

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 6  
 DZICAC.NE2 21-MAR-77 15:28 ERROR POINTER TABLE

246	001244	001476	DF0	;SERRPC, ADCSR,0
247				
248				
249				
250	001246	023013	EM7	;ADDR. OR GENERIC CODE INCORRECT ON INTR.
251	001250	023267	DH3	;ERROR MODULE ICAR
252	001252	001506	DT2	; PC ADDR S/B WAS
253	001254	001476	DF0	;SERRPC, ADCSR, SGDDAT, SBDDAT
254				
255				
256				
257				
258	001256	023072	EM10	;RIF DID NOT CLEAR INTR. FLAG ON A005
259	001260	023140	DH1	;ERROR A/D
260	001262	001500	DT1	; PC ADDR
261	001264	001476	DF0	;SERRPC, ADCSR
262				
263				
264	001266	023417	EM11	;COUNTER MODULE COULD NOT READ-WRITE BUFFER
265	001270	023501	DH4	;ERROR COUNTER
266	001272	001520	DT3	;PC ADDR
267	001274	001476	DF0	;SERRPC SBDADR
268				
269				
270				
271	001276	022601	EM2	
272	001300	023173	DH2	
273	001302	001526	DT4	
274	001304	001476	DF0	
275				
276				
277				
278	001306	023542	EM12	;COUNTER MODULE MISSED A COUNT
279	001310	023173	DH2	
280	001312	001526	DT4	
281	001314	001476	DF0	
282				
283				
284				
285	001316	023606	EM13	;COUNTER FAILED TO INTERRUPT
286	001320	023501	DH4	
287	001322	001520	DT3	
288	001324	001476	DF0	
289				
290				
291				
292	001326	023653	EM14	
293	001330	023267	DH3	
294	001332	001526	DT4	
295	001334	001476	DF0	
296				
297				
298				
299	001336	023751	EM15	
300	001340	023173	DH2	
301	001342	001526	DT4	

302	001344	001476	DF0			
303						
304						
305						
306	001346	024033	EM16			
307	001350	023501	DH4			
308	001352	001520	DT3			
309	001354	001476	DF0			
310						
311						
312						
313	001356	024110	EM17			
314	001360	023173	DH2			
315	001362	001526	DT4			
316	001364	001476	DF0			
317						
318						
319						
320	001366	024164	EM20			
321	001370	023501	DH4			
322	001372	001520	DT3			
323	001374	001476	DF0			
324						
325						
326						
327	001376	024232	EM21			
328	001400	023501	DH4			
329	001402	001520	DT3			
330	001404	001476	DF0			
331						
332						
333						
334	001406	024270	EM22			
335	001410	023267	DH3			
336	001412	001526	DT4			
337	001414	001476	DF0			
338						
339						
340						
341	001416	024341	EM23			
342	001420	023501	DH4			
343	001422	001520	DT3			
344	001424	001476	DF0			
345						
346						
347						
348	001426	024421	EM24			
349	001430	023173	DH2			
350	001432	001526	DT4			
351	001434	001476	DF0			
352						
353						
354						
355	001436	024502	EM25			
356	001440	024621	DH5			
357	001442	001540	DT5			

;ERROR 17 RIF DIDN'T CLEAR INTERRUPT FLAG ON COUNTER

;ERROR 20 COUNTER MODULE INITIALIZATION PART 1  
 ;INIT FAILED TO CLEAR COUNTER BIT(S)

;ERROR 21 ILLEGAL INTERRUPT POSTED ON ICS BUS

;ERROR 22 ICS-11 FAILED TO INTERRUPT

;ERROR 23 ICAR NOT ZERO AFTER FORCED ICS INTERRUPT.

;ERROR 24 SYS INITIALIZE FAILED TO CLEAR COUNTER INTERRUPT FLAG

;ERROR 25 COUNTER STARTED COUNTING AFTER SYSTEM INIT.

;ERROR 26 A005 READ DUAL ADDR ERROR  
 ; ERROR A/D DUAL  
 ; PC ADDR ADDR

358	001444	001476				DF0	
359							
360							;ERROR 27 A005 WRITE DUAL ADDR. ERROR
361							
362	001446	024537				EM26	
363	001450	024621				DH5	
364	001452	001540				DT5	
365	001454	001476				DF0	
366							
367							
368							;ERROR 30 SEND -RECIEVE DATA ERROR
369							
370	001456	024567				EM27	
371	001460	023173				DH2	
372	001462	001526				DT4	
373	001464	001476				DF0	
374							
375							
376							;ERROR 31 SAME AS ERROR 30 ONLY NO HEADER TYPEOUT
377							
378	001466	001476				DF0	
379	001470	001163				\$CRLF	
380	001472	001526				DT4	
381	001474	001476				DF0	
382							
383							
384	001476	000000				DF0:	0
385	001500	001116	007144	000000		DT1:	.WORD SERRPC,ADCSR,0
386	001506	001116	007144	001124		DT2:	.WORD SERRPC,ADCSR,\$GDDAT,\$BDDAT,0
387	001514	001126	000000				
388	001520	001116	001122	000000		DT3:	.WORD SERRPC,\$BDADR,0
389	001526	001116	001122	001124		DT4:	.WORD SERRPC,\$BDADR,\$GDDAT,\$BDDAT,0
390	001534	001126	000000				
391	001540	001116	007144	001122		DT5:	.WORD SERRPC,ADCSR,\$BDADR,0
392	001546	000000					
393							
394							
395							
396							
397	001550	020320				FR110:	20320 ;TTY DELAY TIME
398	001552	006532				FR50:	6532
399	001554	005650				FR40:	5650
400	001556	002724				FR20:	2724
401	001560	002052				FR16:	2052 ;DELAY FOR 16 MILLSEC
402	001562	001024				FR5:	1024
403	001564	000500				FR3:	500 ;DELAY FOR 3 MILLI SEC.
404	001566	000000				FREQ:	0 ;DELAY TIME.
405	001570	000000				FREQ1:	0
406	001572	000000				FREQ2:	0
407	001574	000000				FREQ3:	0
408	001576	000152				FR1:	152 ;DELAY FOR 1 MILLI SEC.
409	001600	000220				FR1120:	220 ;ON AN 11/20 FR1 X 3 FOR 11/45.
410	001602	000000				PATRM:	0 ;PATTERN MODIFIER
411	001604	000000				PATRN:	0 ;PATTERN TO BE SENT TO OUTPUT MODULE
412	001606	000000				PATRNC:	0
413	001610	000000				PATJOY:	0



414	001612	171000	ICSMOD: 171000	; STARTING ADDRESS OF ICS MODULES.	
415	001614	171776	ICSR: 171776		
416	001616	171774	ICAR: 171774		
417	001620	000234	ICSVT: 234		
418	001622	000236	ICSVT2: 236		
419	001624	000064	TPVCT: 64		
420	001626	000066	TPVCT2: 66		
421	001630	000000	INCFLG: 0		
422	001632	000000	TPBSY: 0		
423	001634	000000	HEADER: 0		
424	001636	000000	TOADR: 0		
425	001640	000000	TODAT: 0		
426	001642	000000	TOGEN: 0		
427	001644	000000	TPBSYP: 0		
428	001646	000001	ST200: 1	; CLEARED ON PROGRAM START AT 200	
429	001650	000000	CORSIZ: 0	; END ADDR OF CORE.	
430	001652	000000	EXPERT: 0	; 0=NOVICE MODE--1=EXPERT MODE	
431	001654	000000	CONNT: 0	; 0=NORMAL; 1=MODS CONNECTED FOR TST 0	
432	001656	000000	LPAV: 0		
433	001660	000000	LINEPR: 0	; 0=TTY OUTPUT---1=LINE PRINTER	
434	001662	177514	LPCSR: 177514		
435	001664	177516	LPDBR: 177516		
436	001666	000000	TPCSR: 0		
437	001670	000000	TPDBR: 0		
438	001672	000000	TMPFIL: 0		
439	001674	000000	TMPVEC: 0		
440	001676	000000	ICSHGH: 0		
441	001700	000000	CTLLOC: 0		
442	001702	000000	ICSLMT: 0		
443					
444		000002	EXIT= 2	; RTI INSTRUCTION	
445		000207	RETURN=207	; RTS PC INSTRUCTION	
446		004737	GOSUB=4737	; JSR PC, INSTRUCTION	
447		022626	POPSP2=22626		
448					
449	001704		.SETUP < .STRAP, .SSCOPE, .SEERROR, .SPOWER, SOFTSWR >		
450					
451	001704		START: SETUP		
452			.SBTTL INITIALIZE THE COMMON TAGS		
453			;; CLEAR THE COMMON TAGS (SCHTAG) AREA		
454	001704	012706	001100	MOV #SCHTAG, R6 ;; FIRST LOCATION TO BE CLEARED	
455	001710	005026		CLR (R6)+ ;; CLEAR MEMORY LOCATION	
456	001712	022706	001140	CMP #SMR, R6 ;; DONE?	
457	001716	001374		BNE -6 ;; LOOP BACK IF NO	
458	001720	012706	001100	MOV #STACK, SP ;; SETUP THE STACK POINTER	
459				;; INITIALIZE A FEW VECTORS	
460	001724	012737	017522	000020	MOV #SSCOPE, #IOTVEC ;; IOT VECTOR FOR SCOPE ROUTINE
461	001732	012737	000340	000022	MOV #340, #IOTVEC+2 ;; LEVEL 7
462	001740	012737	016670	000030	MOV #SEERROR, #EMTVEC ;; EMT VECTOR FOR ERROR ROUTINE
463	001746	012737	000340	000032	MOV #340, #EMTVEC+2 ;; LEVEL 7
464	001754	012737	021056	000034	MOV #STRAP, #STRAPVEC ;; TRAP VECTOR FOR TRAP CALLS
465	001762	012737	000340	000036	MOV #340, #STRAPVEC+2 ;; LEVEL 7
466	001770	012737	017350	000024	MOV #SPWRDN, #PWRVEC ;; POWER FAILURE VECTOR
467	001776	012737	000340	000026	MOV #340, #PWRVEC+2 ;; LEVEL 7
468	002004	005037	001160		CLR STIMES ;; INITIALIZE NUMBER OF ITERATIONS
469	002010	012737	002010	001106	MOV #., SLPADR ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE

```

470 002016
471
472
473 002016 013746 000004
474 002022 012737 002056 000004
475 002030 012737 177570 001140
476 002036 012737 177570 001142
477 002044 022777 177777 177066
478 002052 001012
479
480 002054 000403
481 002056 012716 002064 64S:
482 002062 000002
483 002064 012737 000176 001140 65S:
484 002072 012737 000174 001142
485 002100 012637 000004 66S:
486
487 002104 012737 176543 016434
488 002112 012737 123456 016436
489 002120 005037 001654
490
491
492
493
494
495 002124 012700 000214
496 002130 012701 000216
497 002134 010120
498 002136 012720 000004
499 002142 022121
500 002144 022700 001000
501 002150 003371
502 002152 104401
503 002154 021176
504
505
506
507
508
509 002156 012737 002164 001700
510 002164 104401 026653 10S:
511 002170 012737 000001 002206
512 002176 000401
513 002200 000771
514 002202 104414
515 002204 001672
516 002206 000001 22S:
517 002210 022737 000013 001672
518 002216 002003
519 002220 104401 026740
520 002224 000757
521 002226 012737 171000 001702 11S:
522 002234 012737 171776 001614
523 002242 005237 001672
524 002246 005337 001672 19S:
525 002252 001407

```

```

SWSU
;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV #ERRVEC, -(SP) ;; SAVE ERROR VECTOR
MOV #64S, #ERRVEC ;; SET UP ERROR VECTOR
MOV #DSWR, SWR ;; SETUP FOR A HARDWARE SWICH REGISTER
MOV #DDISP, DISPLAY ;; AND A HARDWARE DISPLAY REGISTER
CMP #-1, #SWR ;; TRY TO REFERENCE HARDWARE SWR
BNE 66S ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
;; AND THE HARDWARE SWR IS NOT = -1
BR 65S ;; BRANCH IF NO TIMEOUT
MOV #65S, (SP) ;; SET UP FOR TRAP RETURN
MOV #SWREG, SWR ;; POINT TO SOFTWARE SWR
MOV #DISPREG, DISPLAY
MOV (SP)+, #ERRVEC ;; RESTORE ERROR VECTOR
MOV #176543, SHINUM
MOV #123456, SLONUM
CLR CONNT

;; FILL LOCATIONS "214-1000" WITH .+2, IOT
FILVEC: MOV #214, R0
MOV #216, R1
98S: MOV R1, (R0)+
MOV #4, (R0)+
CMP (R1)+, (R1)+
CMP #1000, R0
BGT 98S
TYPE MHEAD ;TYPE HEADER

**
** SINCE ISC-11 IS CAPABLE OF DIFFERENT CONTROL REGISTERS
** ASK OPERATOR FOR FILE BOX AND INTERRUPT VECTOR ADDRESS
**
MOV #10S, CTLLC
TYPE, MFILE ;ASK FOR FILE BOX
MOV #1, 22S
BR +4
BR 10S
INOC
TMPFIL
1
22S: CMP #13, TMPFIL
BGE 11S
TYPE, ILLEG ;ILLEGAL FILE BOX... RE-ASK
BR 10S
11S: MOV #171000, ICSLMT ;INIT. MODULE ADDRESSES
MOV #171776, ICSR ;INIT. ICSR
INC TMPFIL
19S: DEC TMPFIL ;DEC. FILE
BEQ 18S ;BRANCH IF FOUND

```

```

526 002254 162737 000010 001614 SUB #10,ICSR ;NEXT ICSR
527 002254 062737 000040 001702 ADD #40,ICSLMT ;NEXT GROUP OF MODULE ADDRESSES
528 002270 000766 BR 195 ;LOOP
529 002272 013737 001614 001616 185: MOV ICSR,ICAR ;CREATE ICAR
530 002300 162737 000002 001616 SUB #2,ICAR
531 002306 013737 001702 001676 MOV ICSLMT,ICSHGH ;SET UPPER ADDRESS LIMIT
532 002314 062737 000040 001676 ADD #40,ICSHGH
533 002322 012737 002336 000004 MOV #125,2#4
534 002330 017700 177260 MOV @ICSR,R0
535 002334 000404 BR 975
536 002336 022626 125: POPSP2
537 002340 104401 027154 TYPE, NONXST
538 002344 000707 BR 105
539 002346 012737 000006 000004 975: MOV #6,2#4
540 002354 012737 002362 001700 MOV #995,CTLLOC
541 002362 104401 026707 002404 995: TYPE, MVECT ;ASK FOR INTERRUPT VECTOR
542 002366 012737 000001 002404 MOV #1,235
543 002374 000401 BR +4
544 002376 000771 BR 995
545 002400 104414 INOCT
546 002402 001674 TMPVEC
547 002404 000001 235: I
548 002406 022737 000776 001674 CMP #776,TMPVEC
549 002414 002404 BLT 165
550 002416 022737 000234 001674 CMP #234,TMPVEC
551 002424 003403 BLE 175
552 002426 104401 026740 165: TYPE, ILLEG ;NOT IN FLOATING AREA
553 002432 000753 BR 995
554 002434 013737 001674 001620 175: MOV TMPVEC,ICSVT
555 002442 062737 000002 001674 ADD #2,TMPVEC
556 002450 013737 001674 001622 MOV TMPVEC,ICSVT2 ;STORE VECTOR ADDRESS
557
558 ;**
559 ;**VERIFY THAT VECTOR GIVEN IS ACTUAL VECTOR BEFORE GOING
560 ;**ANY FARTHER
561
562 002456 013737 000020 002624 MOV @#20,935 ;SAVE LOC. 20
563 002464 012737 002540 000020 MOV #955,@#20 ;SET UP IOT TRAP VECTOR
564 002472 012777 002606 177120 MOV #945,@ICSVT ;SET UP INTERRUPT ROUTINE
565 002500 012777 000340 177114 MOV #340,@ICSVT2 ;PS PICKUP
566 002506 012737 000240 177776 MOV #240,@#PS ;SET PRIORITY TO 5
567 002514 012777 000404 177072 MOV #404,@ICSR ;SET MAINT. TO INTERRUPT
568 002522 000240 NOP
569 002524 104401 027033 TYPE, NOINT ;NO INTERRUPT
570 002530 013737 002624 000020 MOV 935,@#20 ;RESTORE LOC 20
571 002536 000413 BR 965 ;RE-ASK QUESTION
572 002540 011605 955: MOV (SP),R5 ;INVALID VECTOR ADDRESS PRINT
573 002542 024545 CMP -(R5),-(R5) ;SUB 4 FROM R5
574 002544 022626 POPSP2 ;POP STACK
575 002546 000240 NOP
576 002550 104401 027076 TYPE, FILINT ;PRINT WHERE VECTOR
577 002554 TYPE, TYPOCT R5
578 002554 010546 MOV R5,-(SP) ;;SAVE R5 FOR TYPEOUT
579 002556 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
580 002560 104401 027131 TYPE, CKJMP
581 002564 022626 POPSP2

```

```

582 002566 005077 177022 96S: CLR @ICSR
583 002572 013737 002624 000020 MOV 93S,@#20
584 002600 000000 HALT
585 002602 000137 002362 JMP 99S
586 002606 013737 002624 000020 94S: MOV 93S,@#20 ; INTERRUPT OK, RESTORE 20
587 002614 022626 POPSP2 ; POP STACK
588 002616 005077 176772 CLR @ICSR
589 002622 000401 BR .+4
590 002624 000000 93S: 0
591
592 ; FILL LOCATIONS "214-1000" WITH .+2, HALT
593
594
595 002626 012737 003374 001700 MOV @START1,CTLLOC
596 002634 012700 000214 FILHLT: MOV @214,R0
597 002640 012701 000216 MOV @216,R1
598 002644 010120 90S: MOV R1,(R0)+
599 002646 012720 000000 MOV @0,(R0)+
600 002652 022121 CMP (R1)+,(R1)+
601 002654 022700 001000 CMP @1000,R0
602 002660 003371 BGT 90S
603 002662 012737 003672 000060 MOV @KEYSRV,@TKVEC
604 002670 012737 000340 000062 MOV @340,@TKVEC+2
605 002676 052777 000100 176240 BIS @100,@STKS
606 002704 012737 002734 0000G4 MOV @9S,@J4
607 002712 005237 001656 INC LPAV
608 002716 005777 176740 TST @LPCSR
609 002722 104401 TYPE
610 002724 001163 SCRLF
611 002726 104401 TYPE
612 002730 025664 MLPAV
613 002732 000403 BR 8S
614 002734 005037 001656 9S: CLR LPAV
615 002740 022626 POPSP2
616
617
618 002742 012737 002756 000004 8S: MOV @3S,@#4 ; DETERMINE CORE SIZE.
619 002750 005000 CLR R0
620 002752 005720 2S: TST (0)+
621 002754 000776 BR 2S
622 002756 005740 3S: TST -(0)
623 002760 010037 001650 MOV R0,CORSIZ
624 002764 100003 BPL 4S
625 002766 012737 077776 001650 MOV @077776,CORSIZ
626 002774 012737 000006 000004 4S: MOV @6,@#4
627 003002 013737 001600 001576 MOV FR1120,FR1 ; SET FREQ FOR 11/20
628 003010 012737 000002 000012 MOV @RTI,@#12 ; FIND OUT IF 11/20 OR 11/45
629 003016 000262 SEV ; IF 11/45 THEN DELAY ITERATIONS
630 003020 074101 XOR %1,%1 ; MUST BE INCREASED.
631 003022 102427 BVS 5S
632 003024 063737 001600 001576 ADD FR1120,FR1 ; FLOW FELL THUR TO HERE -
633 003032 063737 001600 001576 ADD FR1120,FR1 ; IT MUST BE AN 11/45 OR EQUIV.
634 003040 013746 000004 MOV @#4,-(SP) ; SAVE LOC 4
635 003044 013746 000006 MOV @#6,-(SP) ; SAVE LOC 6
636 003050 012737 003070 000004 MOV @99S,@#4 ; SETUP FOR TRAP
637 003056 005737 177760 TST @#177760 ; CHECK IF 11/70

```

```

638 003062 006337 001576          ASL      FR1
639 003066 000401          BR       .+4
640 003070 022626          99S:    POPSP2
641 003072 012637 000006          MOV     (SP)+,2#6
642 003076 012637 000004          MOV     (SP)+,2#4
643 003102 005037 000012          5S:    CLR      2#12
644 003106 013737 001576 001564          MOV     FR1,FR3
645 003114 006137 001564          ROL     FR3
646 003120 013737 001564 001562          MOV     FR3,FR5
647 003126 063737 001576 001564          ADD     FR1,FR3
648 003134 063737 001564 001562          ADD     FR3,FR5
649 003142 013737 001562 001560          MOV     FR5,FR16
650 003150 006137 001560          ROL     FR16
651 003154 063737 001562 001560          ADD     FR5,FR16
652 003162 013737 001560 001556          MOV     FR16,FR20
653 003170 063737 001576 001560          ADD     FR1,FR16
654 003176 063737 001562 001556          ADD     FR5,FR20
655 003204 013737 001556 001552          MOV     FR20,FR50
656 003212 006137 001552          ROL     FR50
657 003216 063737 001562 001552          ADD     FR5,FR50
658 003224 063737 001562 001552          ADD     FR5,FR50
659 003232 013737 001552 001550          MOV     FR50,FR110
660 003240 006137 001550          ROL     FR110
661 003244 063737 001562 001550          ADD     FR5,FR110
662 003252 063737 001562 001550          ADD     FR5,FR110
663 003260 013737 001556 001554          MOV     FR20,FR40
664 003266 006137 001554          ROL     FR40
665 003272 005037 001652          CLR     EXPERT
666
667
668 003276 012737 001564 001566          MOV     #FR3, FREQ
669 003304 005037 001604          CLR     PATRN
670 003310 005737 001646          TST     ST200
671 003314 001406          BEQ     7S
672 003316 013737 001150 001666          MOV     STPS,TPCSR
673 003324 013737 001152 001670          MOV     STPB,TPDBR
674 003332 012700 013176          7S:    MOV     #INADR, RO
675 003336 005037 001646          CLR     ST200
676 003342 005020          1S:    CLR     (0)+
677 003344 020027 013344          CMP     RO,#OUTS
678 003350 001374          BNE     1S
679 003352 005037 001574          CLR     FREQ3
680 003356 005037 001572          CLR     FREQ2
681 003362 005037 001570          CLR     FREQ1
682 003366 013737 001564 001566          MOV     FR3,FREQ
683
684 003374 012706 001100          START1: MOV    #1100,SP
685 003400 005077 176210          CLR    JICSR
686 003404 005037 001100          CLR    SPASS
687 003410 005037 001660          CLR    LINEPR
688 003414 013737 001566 013134          MOV    FREQ,RTEMP
689 003422 005737 001646          TST    ST200
690 003426 001411          BEQ    3S
691 003430 012737 020636 000034          MOV    #STYPE,2#34
692 003436 005037 000036          CLR    2#36
693 003442 104401          TYPE

```

```

;11/70, DOUBLE
;SKIP
;RESTORE STACK
;RESTORE LOC 6
;RESTORE LOC 4
;RESTORE LOC 12.
;NOW WE MUST SET THE REST OF
;OF THE DELAY TIMES UP.

```

```

;CLEAR ADDR AREA
;INDICATE START AT LOC 200.

```

```

;HAS PROG. BEEN EVER STARTED
;AT ADDR. 200? IF SO CONTINUE.
;SET UP TO PRINT.

```

```

694 003444 022371          MSTERR
695 003446 000137 001704  JMP      START      ;PROGRAM MUST BE STARTED AT ADDR. 200.
696 003452 013737 001550 001566 3S:  MOV     FR110, FREQ  ;SET DELAY FOR TTY SETTLE TIME.
697 003460 005037 001570          CLR     FREQ1
698 003464 013737 001666 001150  MOV     TPCSR, $TPS
699 003472 013737 001670 001152  MOV     TPOBR, $TPB
700 003500 104422          DELAY
701 003502 000005          RESET
702 003504 104422          DELAY
703 003506 012737 000340 177776  MOV     #340, PS
704 003514 022737 000176 001140  CMP     #176, SWR
705 003522 001001          BNE     99S
706 003524 104405          GTSWR
707 003526 104401          99S:  TYPE
708 003530 022124          MTN
709 003532 012737 003650 003554  MOV     #TSTNO, 1S
710 003540 012737 000001 003556  MOV     #1, 1S+2
711 003546 000401          BR     +4
712 003550 000766          BR     99S
713 003552 104414          INOCT
714 003554 003650          1S:  TSTNO
715 003556 000001          1
716 003560 005737 003650          TST     TSTNO
717 003564 100404          BMI     4S
718 003566 023727 003650 000007  CMP     TSTNO, #7. ;LEGAL NUMBER?
719 003574 003403          BLE     2S
720 003576 104401          4S:  TYPE
721 003600 022147          MTNL
722 003602 000674          BR
723 003604 013737 003650 013134 2S:  MOV     START1
724 003612 006337 013134          MOV     TSTNO, RTEMP
725 003616 062737 003652 013134  ASL     RTEMP
726 003624 017737 007304 013134  ADD     #TSTLST, RTEMP
727 003632 012777 000100 175304  MOV     #RTEMP, RTEMP
728 003640 005037 177776          MOV     #100, $TKS ;ENABLE TTY TO INTERRUPT.
729 003644 000177 007264          CLR     PS
730          JMP     #RTEMP ;GOTO TEST
731 003650 000000          TSTNO: 0
732 003652 003744          TSTLST: TST0
733 003654 005322          TST1
734 003656 005604          TST2
735 003660 005676          TST3
736 003662 005746          TST4
737 003664 007146          TST5
738 003666 010026          TST6
739 003670 011000          TST7
740
741
742          ;#
743          ;#KEYBOARD INTERRUPT HANDLER
744          ;#
745 003672 017737 175250 003742  KEYSRV: MOV     #TKB, 2S
746 003700 042737 177600 003742  BIC     #177600, 2S
747 003706 122737 000003 003742  CMPB   #3, 2S
748 003714 001410          BEQ     1S
749 003716 122737 000007 003742  CMPB   #7, 2S ;CONTROL G TYPED?

```

E03

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 15  
DZICAC.NE2 21-MAR-77 15:28 INITIALIZE THE COMMON TAGS

750 003724 001003  
751 003726 104401  
752 003730 020431  
753 003732 104405  
754 003734 000002  
755 003736 000137 003374  
756 003742 000000  
757  
758

BNE 3S  
TYPE  
SCNTLG  
GTSMR  
3S: EXIT  
1S: JMP START1  
2S: 0

: IF NOT, THEN EXIT!  
: ALLOW {CONTROL G}  
: TO BE TYPED-OUT  
: GO GET SMR VALUE VIA TTY

```

759
760
761
762
763 003744 104401 TSTO: TYPE ;TYPE HEADER.
764 003746 025244 MHTO
765 003750 104415 INAR ;GET INPUT MODULE ADDRS.
766 003752 104416 OUTAR ;GET OUTPUT MODULE ADDRS.
767 003754 104417 PATAR ;GET PATTERN (OR USE DEFAULT).
768 003756 104420 DELAR ;GET DELAY TIME (OR USE DEFAULT).
769 003760 005037 001100 CLR SPASS
770 003764 005037 001634 CLR HEADER
771
772 003770 012777 004444 175622 MOV #TOINT, @ICSVT ;INITIAL SETUP.
773 003776 012777 004370 175620 MOV #TOPINT, @TPVCT ;SET UP FOR ICS INTERRUPTS.
774 004004 012777 000340 175610 MOV #340, @ICSVT2 ;SET UP FOR TTY INTERRUPTS.
775 004012 012777 000200 175606 MOV #200, @TPVCT2 ;SET CPU PRIORITY TO 6 ON ICS INTR.
776 004020 005037 001632 CLR TPBSY ;SET CPU PRIORITY TO 4 ON TTY INTR.
777 004024 012700 013344 MOV #OUTS, R0 ;SET NO TTY OUTPUT IN PROGRESS
778 004030 005020 013404 1S: CLR (0)+ ;CLEAR TEMP STORAGE AREA OF INPUT
779 004032 020027 013404 CMP RO, #OUTSE ;MODULE DATA CHANGE.
780 004036 001374 BNE 1S
781 004040 005737 001654 TST CONNT
782 004044 001002 BNE 2S
783 004046 104401 TYPE ;TYPE HEADER.
784 004050 021360 MTOH
785 004052 005037 001636 2S: CLR TOADR ;CLEAR TYPE OUT CONSTANTS.
786 004056 005037 001640 CLR TODAT
787 004062 005037 001642 CLR TOGEN
788
789
790 004066 104422 TOOUTR: DELAY ;TAKE CARE OF OUTPUT ADDRS.
791 004070 012737 004070 001106 MOV #TOOUTR, $LPADR
792 004076 104421 DELAY2
793 004100 012702 013176 MOV #INADR, R2
794 004104 052777 000004 175502 BIS #4, @ICSR ;ALLOW ICS TO INTR.
795 004112 005037 177776 CLR PS ;LET CPU ALLOW INTR.
796 004116 012704 013344 MOV #OUTS, R4
797 004122 012700 013236 TOOUT: MOV #OUTADR, R0 ;GET OUTPUT MODULE LIST.
798 004126 012001 1S: MOV (0)+, R1 ;GET FIRST ADDR
799 004130 001405 BEQ TOIN ;IF N ADDR. - EXIT.
800 004132 013711 001604 MOV PATRN, (1) ;SEND PATTERN TO OUTPUT MODULE.
801 004136 020027 013276 CMP RO, #CNTADR
802 004142 001371 BNE 1S
803
804 004144 005737 001654 TOIN: TST CONNT ;TAKE CARE OF INPUT ADDRS.
805 004150 001407 BEQ 2S ;MODULES CONNECTED?
806 004152 012701 013344 MOV #OUTS, R1 ;IF NOT NORMAL CONTINUE.
807 004156 013721 001604 3S: MOV PATRN, (1)+ ;IF SO COS AREA = CURRENT PATTERN.
808 004162 020127 013404 CMP R1, #OUTSE
809 004166 001373 BNE 3S
810 004170 013737 001604 2S: MOV PATRN, PATJOY
811 004176 104422 DELAY ;DELAY TIME
812 004200 000240 NOP
813 004202 005737 001632 TST TPBSY
814 004206 001374 BNE .-6
    
```



G03

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 17  
 DZICAC.NE2 21-MAR-77 15:28 INITIALIZE THE COMMON TAGS

815	004210	011201		1S:	MOV	(2),R1		;PICK UP FIRST ADDR.
816	004212	001002			BNE	+6		
817	004214	000137	004674		JMP	TOLOP		
818	004220	021114			CMP	(1),(4)		;DATA CHANGED?
819	004222	001007			BNE	TOIN2		;IF YES TAKE CARE OF IT
820		004224		TOIN1=.				
821	004224	005722			TST	(2)+		
822	004226	005724			TST	(4)+		;UPDATE COD POINTER
823	004230	020227	013236		CMP	R2,#OUTADR		;DONE ALL INPUT MODULES?
824	004234	001365			BNE	1S		
825	004236	000137	004674		JMP	TOLOP		
826								
827	004242	005737	001654		TOIN2:	TST	CONNT	
828	004246	001425			BEQ	1S		
829	004250	005737	001632		2S:	TST	TPBSY	
830	004254	001375			BNE	2S		
831	004256	010137	001122		MOV	R1,#BODADR		
832	004262	013737	001610	001124	MOV	PATJOY,\$GDDAT		
833	004270	011137	001126		MOV	(1),#BODAT		
834	004274	005737	001634		TST	HEADER		
835	004300	001403			BEQ	3S		
836	004302	104031			ERROR	31		
837	004304	000137	004674		JMP	TOLOP		
838								
839	004310	104030			3S:	ERROR	30	
840	004312	005237	001634		INC	HEADER		
841	004316	000137	004674		JMP	TOLOP		
842								
843	004322	005737	001632		1S:	TST	TPBSY	
844	004326	001375			BNE	1S		
845	004330	012737	000340	177776	MOV	#340,PS		
846	004336	010137	001636		MOV	R1,TOADR		;GET ADDR. OF INPUT MODULE
847	004342	011137	001640		MOV	(1),TODAT		;GET CHANGE DATA
848	004346	005037	001642		CLR	TOGEN		;NO GEN CODE (NO INTERRUPT)
849	004352	005237	001632		INC	TPBSY		;SET TTY BUSY.
850	004356	011114			MOV	(1),(4)		;RECORD NEW DATA
851	004360	104430			FOCTA			;FORM INFO INTO AN ASCIZ MESSAG.
852	004362	005046			CLR	-(6)		
853	004364	012746	004224		MOV	#TOIN1,-(6)		
854								
855	004370	105777	174554		TOPINT:	TSTB	#STPS	;PRINTER BUSY?
856	004374	100375			BPL	TOPINT		
857	004376	032777	020000	174534	BIT	#020000,#SWR		;INHIBIT TYPEOUT?
858	004404	001403			BEQ	3S		;NO CONTINUE.
859	004406	005037	001632		CLR	TPBSY		;YES-STOP TYPEOUT
860	004412	000002			EXIT			
861	004414				3S:			
862	004414	112377	174532		MOVB	(3)+,#STPB		;SEND CHAR.
863	004420	001404			BEQ	1S		;IF END GO TO END
864	004422	052777	000100	174520	BIS	#100,#STPS		;MAKE SURE TTY CAN INTR.
865	004430	000002			EXIT			;EXIT
866								
867	004432	005077	174512		1S:	CLR	#STPS	;TTY SETTLE DOWN TIME
868	004436	005037	001632		CLR	TPBSY		;CLEAR BUSY
869								
870	004442	000002			EXIT			

```

871
872
873
874
875
876
877 004444 017737 175146 004666 TOINT: MOV @ICAR,6S ;GET ADDR. OF INTRING MOD.
878 004452 006137 004666 ROL 6S ;FORM REAL ADDR.
879 004456 042737 177001 004666 BIC @177001,6S
880 004464 053737 001612 004666 BIS ICSMOD,6S
881 004472 017737 000170 004670 MOV @6S,7S ;FETCH DATA.
882 004500 005737 001632 10S: TST TPBSY ;TYPEOUT PENDING?
883 004504 001406 BEQ 5S ;IF NOT CONTINUE.
884 004506 042777 000004 175100 BIC @4,@ICSR ;STOP ICS FROM INTR.
885 004514 005037 177776 CLR PS ;ALLOW INTERRUPTS FROM TTY.
886 004520 000767 BR 10S ;LOOP.
887 004522 012737 000300 177776 5S: MOV @300,PS ;LOCK OUT INTR.S
888 004530 052777 000005 175056 BIS @5,@ICSR ;RE-ENABLE INTR.S AND SET RIF.
889 004536 017737 175054 001642 MOV @ICAR,TOGEN ;GET GENERIC CODE
890 004544 013737 004666 001636 MOV 6S,TOADR
891 004552 005777 175060 TST @TOADR
892 004556 013737 004670 001640 MOV 7S,TODAT
893 004564 005737 001654 TST COUNT
894 004570 001401 BEQ 9S
895 004572 000002 EXIT
896
897 004574 000337 001642 9S: SWAB TOGEN ;RIGHT JUSTIFY GEN CODE.
898 004600 042737 177760 001642 BIC @177760,TOGEN
899 004606 010037 004672 MOV RO,SAVO ;SAVE RO
900 004612 012700 013176 MOV @INADR,RO ;FIND OFFSET OF INPUT MODULE
901 004616 022037 001636 2S: CMP (0)+,TOADR ;THAT INTERRUPTED.
902 004622 001404 BEQ 3S
903 004624 020027 013236 CMP RO,@OUTADR ;IF NOT ENTERED INPUT MODULE THAT
904 004630 001407 BEQ 4S ;THAN ADDR WILL NOT BE IN TABLE
905 004632 000771 BR 2S ;IN THAT CASE DON'T WORRY ABOUT IT.
906 004634 162700 013200 3S: SUB @INADR+2,RO ;SUB TO GET OFFSET.
907 004640 062700 013344 ADD @OUTS,RO ;ADD STORAGE OF COS TO RECORD
908 004644 013710 001640 MOV TODAT,(0) ;NEW DATA FOR THAT MODULE.
909 004650 013700 004672 4S: MOV SAVO,RO ;RESTORE RO.
910 004654 104430 FOCTA ;NO-FORM ASCIZ STRING.
911 004656 005237 001632 INC TPBSY ;SET OUTPUT BUSY
912 004662 000642 BR TOPINT ;START OUTPUT
913 004664 1S:
914 004664 000002 EXIT
915
916 004666 000000 6S: 0
917 004670 000000 7S: 0
918 004672 000000 SAVO: 0
919 004674 005737 001632 TOLOP: TST TPBSY
920 004700 001375 BNE TOLOP
921 004702 000004 SCOPE
922 004704 104423 CPATR
923 004706 000137 004070 JMP TOOUTR
924
925
926 ;*
; *ROUTINE TO CONVERT 3 OCTAL NUMBERS TO AN ASCIZ STRING

```

```

;# CALL= FOCTA
927
928
929 004712 012703 027206 ROCTA: MOV #OUTBF,R3 ;SET UP BUFFER
930 004716 013737 001636 005004 MOV TOADR,PACK1
931 004724 004737 005010 JSR PC,PACK ;PACK ADDR.
932 004730 013737 001640 005004 MOV TODAT,PACK1
933 004736 004737 005010 JSR PC,PACK ;PACK DATA
934 004742 013737 001642 005004 MOV TOGEN,PACK1
935 004750 001402 BEQ IS
936 004752 004737 005010 JSR PC,PACK ;PACK GEN CODE (IF ANY)
937 004756 112723 000001 1S: MOVB #1,(3)+ ;FILLER CHARACTERS.
938 004762 112723 000001 MOVB #1,(3)+
939 004766 112723 000001 MOVB #1,(3)+
940 004772 105023 CLRB (3)+ ;STRING TERMINATOR.
941 004774 105023 CLRB (3)+
942 004776 012703 027204 MOV #OUTBF1,R3 ;RESET POINTER
943 005002 000002 EXIT
944
945 005004 000000 PACK1: 0
946 005006 000000 PACK2: 0
947
948 005010 012737 000260 005006 PACK: MOV #260,PACK2
949 005016 005737 005004 TST PACK1
950 005022 100002 BPL .+6
951 005024 005237 005006 INC PACK2
952 005030 113723 005006 MOVB PACK2,(3)+
953 005034 000337 005004 SWAB PACK1
954 005040 013737 005004 005006 MOV PACK1,PACK2
955 005046 006037 005006 ROR PACK2
956 005052 006037 005006 ROR PACK2
957 005056 006037 005006 ROR PACK2
958 005062 006037 005006 ROR PACK2
959 005066 042737 177770 005006 BIC #177770,PACK2
960 005074 052737 000260 005006 BIS #260,PACK2
961 005102 113723 005006 MOVB PACK2,(3)+
962 005106 013737 005004 005006 MOV PACK1,PACK2
963 005114 006037 005006 ROR PACK2
964 005120 042737 177770 005006 BIC #177770,PACK2
965 005126 052737 000260 005006 BIS #260,PACK2
966 005134 113723 005006 MOVB PACK2,(3)+
967 005140 000337 005004 SWAB PACK1
968 005144 013737 005004 005006 MOV PACK1,PACK2
969 005152 006037 005006 ROR PACK2
970 005156 006037 005006 ROR PACK2
971 005162 006037 005006 ROR PACK2
972 005166 006037 005006 ROR PACK2
973 005172 006037 005006 ROR PACK2
974 005176 006037 005006 ROR PACK2
975 005202 042737 177770 005006 BIC #177770,PACK2
976 005210 052737 000260 005006 BIS #260,PACK2
977 005216 113723 005006 MOVB PACK2,(3)+
978 005222 013737 005004 005006 MOV PACK1,PACK2
979 005230 006037 005006 ROR PACK2
980 005234 006037 005006 ROR PACK2
981 005240 006037 005006 ROR PACK2
982 005244 042737 177770 005006 BIC #177770,PACK2
  
```



K03

```

994
995
996
997
998 005322 104401 TST1: TYPE ;TYPE HEADER.
999 005324 025323 MHT1
1000 005326 005077 173612 CLR ;2STKS ;DON'T ALLOW TTY INTERRUPTS
1001 005332 005737 001652 5S: TST EXPERT ;EXPERT MODE?
1002 005336 001002 BNE .+6
1003 005340 104401 TYPE ;ASK "INPUT OR OUTPUT MODULE?"
1004 005342 022166 M100
1005 005344 104401 TYPE
1006 005346 022366 M0
1007 005350 005037 015046 CLR CHAR
1008 005354 105777 173564 1S: TSTB ;2STKS ;WAIT FOR RESPONSE
1009 005360 100375 BPL 1S
1010 005362 105777 173562 2S: TSTB ;2STPS ;PRINTER BUSY?
1011 005366 100375 BPL 2S
1012 005370 117777 173552 173554 MOVB ;2STKB,2STPB ;ECHO CHARACTER.
1013 005376 117737 173544 013134 MOVB ;2STKB,RTEMP
1014 005404 142737 000240 013134 BICB #240,RTEMP
1015 005412 123727 013134 000015 CMPB RTEMP,#15
1016 005420 001412 BEQ 4S
1017 005422 123727 013134 000003 CMPB RTEMP,#3 ;WAS TC TYPED?
1018 005430 001002 BNE .+6
1019 005432 000137 003374 JMP START1 ;IF SO GOTO MONITR.
1020 005436 113737 013134 015046 MOVB RTEMP,CHAR
1021 005444 000743 BR 1S
1022 005446 104401 4S: TYPE
1023 005450 001163 SCRLF
1024 005452 122737 000117 015046 CMPB #'0,CHAR ;DID HE TYPE "0"?
1025 005460 001433 BEQ TST10 ;IF SO-GOTO OUTPUT ROUTINE
1026 005462 122737 000111 015046 CMPB #'I,CHAR ;DID HE TYPE "I"?
1027 005470 001320 BNE 5S ;IF NOT RETYPE QUESTION
1028
1029 ;*ROUTINE TO HANDLE INPUTTING FROM INPUT MODULE TO DISPLAY
1030
1031 005472 104415 INAR ;GET INPUT MODULE ADDR.
1032 005474 104401 024775 TYPE,MSWD
1033 005500 104401 022004 TYPE,MWK
1034
1035 005504 012777 000100 173432 3S: MOV #100,2STKS ;ALLOW TTY INTERRUPTS
1036 005512 017700 005460 MOV ;2INADR,RO ;GET DATA FROM INPUT MODULE
1037 005516 010037 001142 MOV RO,DISPLAY ;PUT IN DISPLAY REGISTER IF 11/45.
1038 005522 022737 000176 001140 CMP #176,SWR ;SOFTWARE SWR IN USE?
1039 005530 001401 BEQ 6S ;BRANCH IF YES
1040 005532 104427 INTR ;RESET TO DISPLAY.
1041 005534 032777 000001 173376 6S: BIT #1,2SWR ;IF BIT1 SET THEN GOTO MONITR.
1042 005542 001763 BEQ 3S ;NOT SET-THEN LOOP.
1043 005544 000137 003374 JMP START1
1044
1045 ;*ROUTINE TO HANDLE OUTPUTTING FROM SWITCH REGISTER TO OUTPUT MODULE
1046
1047 005550 104416 TST10: OUTAR ;GET OUTPUT MODULE ADDR.
1048 005552 104420 DELAR ;GET DELAY TIME (DEFAULT=3MS).
1049 005554 005037 177776 CLR PS

```



```

1056                                     ;*
1057                                     ;*TEST 2 DAC CALIBRATION ROUTINE
1058                                     ;*
1059
1060 005604 104401 TST2: TYPE ;TYPE HEADER.
1061 005606 025401 MHT2
1062 005610 104424 IDAC ;INPUT DAC ADDR.
1063 005612 005737 013336 TST DACADR ;ANY DACS PRESENT
1064 005616 001005 BNE 4$
1065 005620 104401 TYPE ;MESSAGE "NO DAC ADDR. IN BUFFER.
1066 005622 021436 MND
1067 005624 000137 005604 JMP TST2
1068 005630 000000 2$: 0
1069 005632 104401 022004 4$: TYPE, MK
1070 005636 017737 173276 005630 1$: MOV @SWR,2$ ;GET VALUE OF SWR
1071 005644 042737 140017 005630 BIC #140017, 2$
1072 005652 013777 005630 005456 3$: MOV 2$, @DACADR ;SEND VALUE.
1073 005660 062737 040000 005630 ADD #40000, 2$ ;SET FOR NEXT CH.
1074 005666 103371 BCC 3$ ;DONE ALL CHS?
1075 005670 000762 BR 1$
1076
1077 005672 000137 003374 JMP START1
1078

```

```

1079
1080
1081
1082
1083
1084 005676 104401
1085 005700 025436
1086 005702 104424
1087 005704 005737 013336
1088 005710 001004
1089 005712 104401
1090 005714 021436
1091 005716 000137 005676
1092 005722 005000
1093 005724 104401 022004
1094 005730 010077 005402
1095 005734 062700 000020
1096 005740 000773
1097
1098 005742 000137 003374
1099

; *
; *TEST 3 DAC INTERACTION TEST
; *OUTPUTS RAMP TO 4 CHANNELS OF DAC THAT ARE "OUT OF PHASE"
; *
TST3: TYPE MHT3 ;TYPE HEADER.
      IDAC ;GET DAC ADDRESS
      TST DACADR ;DAC PRESENT?
      BNE IS
      TYPE ;NO-MESSAGE "NO DAC ADDR. IN BUFFER"
      MND
      JMP TST3
1S: CLR RO
     TYPE, MMK
2S: MOV RO, 2DACADR ;SEND WORD TO DAC
     ADD #20, RO
     BR 2S
      JMP START1
  
```



```

1100
1101
1102
1103
1104
1105 005746 104401 TST4: TYPE ;TYPE HEADER.
1106 005750 025473 MHT4
1107 005752 104425 CNTAR ;GET COUNTER MODULE ADDRS.
1108 005754 013737 001554 001566 MOV FR40,FREQ
1109 005762 104401 022004 TYPE, M4K
1110
1111 005766 012700 013276 TST4L1: MOV #CNTADR,RO ;GET LIST OF MODULES
1112 005772 010037 005004 MOV RO,PACK1
1113 005776 005710 TST (0) ;ANY ADDRS. ENTERED?
1114 006000 001004 BNE TST4L
1115 006002 104401 023355 TYPE M4AE ;"NO ADDRS ENTERED".
1116 006006 000137 005746 JMP TST4
1117
1118 006012 012777 000100 173124 TST4L: MOV #100,2STKS ;ALLOW TTY INTERRUPTS
1119 006020 013700 005004 MOV PACK1,RO
1120 006024 020027 013336 CMP RO,#0ACADR ;DONE ALL COUNTERS?
1121 006030 001005 BNE 2S
1122 006032 005237 001100 1S: INC SPASS
1123 006036 104401 TYPE
1124 006040 021465 MEND
1125 006042 000751 BR TST4L1
1126 006044 012037 001122 2S: MOV (0)+,#SBOADR ;GET ADDR. OF COUNTER MODULE.
1127 006050 001770 BEQ 1S ;NOTE: ZERO IF NO ADDR. ENTERED.
1128 006052 010037 005004 MOV RO,PACK1
1129
1130
1131
1132
1133
1134 006056 012737 006056 001106 TST4A: MOV #TST4A,SLPADR ;SET FOR SCOPE, ITERATIONS.
1135 006064 012777 077777 173030 MOV #77777,2SBOADR ;SEND PATTERN.
1136 006072 017737 173024 001126 MOV 2SBOADR,SBOADR ;GET IT BACK
1137 006100 001001 BNE TST4AL ;IF NON-ZERO - WE GOT SOMETHING BACK.
1138
1139 006102 104011 TST4AL: ERROR 11 ;COULD NOT MAKE ANY DATA XFER
1140 006104 000004 SCOPE
1141
1142
1143
1144
1145
1146 006106 012737 000001 013134 TST4B: MOV #1,RTEMP ;SET "GOOD DATA"
1147 006114 012737 000010 001160 MOV #10,STIMES
1148 006122 012737 006122 001106 1S: MOV #15,SLPADR ;SET FOR ITERATIONS
1149 006130 013737 013134 001124 MOV RTEMP,SGDDAT
1150 006136 013777 001124 172756 MOV SGDDAT,2SBOADR ;SEND PATTERN TO COUNTER MODULE.
1151 006144 017737 172752 001126 MOV 2SBOADR,SBOADR ;GET IT BACK.
1152 006152 023737 001124 001126 CMP SGDDAT,SBOADR ;DATA SENT=DATA RECEIVED?
1153 006160 001402 BEQ 2S
1154
1155 006162 104012 ERROR 12 ;NO - REPORT ERROR
    
```

```

1156 006164 000412 BR 3S ;LOOP
1157
1158 006166 043777 001124 172726 2S: BIC SGDDAT, 2SBADR ;TRY CLEARING BIT.
1159 006174 017737 172722 001126 MOV 2SBADR, SBDDAT ;DID IT CLEAR?
1160 006202 001403 BEQ 3S ;YES-LOOP.
1161 006204 005037 001124 CLR SGDDAT
1162 006210 104012 ERROR 12
1163
1164 006212 000004 3S: SCOPE
1165 006214 006137 013134 ROL RTEMP ;SET TO NEXT BIT.
1166 006220 103340 BCC 1S ;CONTINUE TESTING IF ALL BITS NOT DONE.
1167
1168 ;*
1169 ;*TEST THAT THE COUNTER MODULE CAN COUNT THRU
1170 ;*EACH STATE
1171 ;*
1172
1173 006222 012737 007076 013134 TST4C: MOV #CNTPAT, RTEMP ;GET ADDR. OF PATTERNS
1174 006230 012737 000010 001160 1S: MOV #10, STIMES
1175 006236 012737 006230 001106 MOV #1S, SLPADR
1176
1177 006244 017777 004664 172650 MOV 2RTEMP, 2SBADR ;SEND PATTERN.
1178 006252 005037 013136 CLR RTEMP1
1179 006256 017737 004652 001124 MOV 2RTEMP, SGDDAT
1180 006264 005237 001124 INC SGDDAT
1181 006270 005237 013136 4S: INC RTEMP1 ;WATCH TO SEE IF COUNTER COUNTS.
1182 006274 001413 BEQ 5S ;IF NO COUNT BY OVERFLOW-ERROR!
1183 006276 017737 172620 001126 MOV 2SBADR, SBDDAT
1184 006304 027737 004624 001126 CMP 2RTEMP, SBDDAT
1185 006312 001766 BEQ 4S
1186 006314 023737 001124 001126 CMP SGDDAT, SBDDAT
1187 006322 001401 BEQ 2S
1188
1189 006324 104013 5S: ERROR 13
1190 006326 000004 2S: SCOPE
1191 006330 062737 000002 013134 ADD #2, RTEMP
1192 006336 023727 013134 007142 CMP RTEMP, #CNTPAE
1193 006344 002731 BLT 1S
1194
1195 ;*
1196 ;*TEST TO SEE IF COUNTER INITIALIZES PROPERLY--PART 1
1197 ;*
1198
1199 006346 012737 000010 001160 TST4D: MOV #10, STIMES
1200 006354 012737 006354 001106 1S: MOV #1S, SLPADR
1201 006362 005077 172534 CLR 2SBADR
1202 006366 104427 INTR
1203 006370 104422 DELAY
1204 006372 005037 001124 CLR SGDDAT
1205 006376 017737 172520 001126 MOV 2SBADR, SBDDAT
1206 006404 001401 BEQ 2S
1207
1208 006406 104025 ERROR 25
1209
1210 006410 000004 2S: SCOPE
1211

```

```

1212
1213
1214
1215
1216 006412 012737 000010 001160 TST4E: MOV #10,STIMES ;SET ITERATION COUNT
1217 006420 012737 006432 001106 MOV #15,SLPADR ;SET LOOP ADDRESS.
1218 006426 005037 001124 CLR $GDDAT
1219 006432 012777 177777 172462 1S: MOV #177777,$SBDADR ;SET THE COUNTER.
1220 006440 000240 NOP
1221 006442 104427 INTR ;SYSTEM INITIALIZE.
1222 006444 017737 172452 001126 MOV $SBDADR,$SDDAT ;READ COUNTER.
1223 006452 001401 BEQ 2S
1224
1225 006454 104020 ERROR 20
1226
1227 006456 000004 2S: SCOPE
1228
1229
1230
1231

```

;\*TEST THAT THE COUNTER INITIALIZES PROPERLY PART 2  
;\*

;\*TEST THAT THE COUNTER HAS NO  
;\*INTERRUPTS POSTED ON THE BUS NOR ANY GENERIC CODE

1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287

006460 105777 173130  
006464 100001  
006466 104021  
006470 000004  
006472 012777 006520 173120  
006500 012777 000404 173106  
006506 005037 177776  
006512 000240  
006514 104022  
006516 000424  
006520 017737 173072 001126  
006526 022626  
006530 012777 000001 173056  
006536 005037 001124  
006542 005777 172354  
006546 005037 177776  
006552 042737 000360 001126  
006560 005737 001126  
006564 001401  
006566 104023  
006570 000004  
006572 012777 006660 173020  
006600 012777 000340 173014  
006606 012737 006614 001106  
006614 012737 000340 177776  
006622 012777 000004 172764

```

;*
TST4F: TSTB  @ICSR          ;ANY INTERRUPTS POSTED?
        BPL   15
        ERROR 21           ;ILLEGAL INTERRUPT ON FCS BUS
1S:     SCOPE
        MOV   #25,@ICSVT    ;SET FOR INTERRUPT
        MOV   #404,@ICSR   ;SET MAINT + INT ENABLE
        CLR   PS
        NOP
        ERROR 22           ;FATAL ERROR - ICS DID NOT INTERRUPT
        BR    3S
        ;ICS INTERRUPTS TO HERE
2S:     MOV   @ICAR,@SDDAT  ;READ ICAR.
        POPSP2             ;READJUST STACK
        MOV   #1,@ICSR
        CLR   @SDDAT
        TST  @SDDADR
        CLR   PS
        BIC  #360,@SDDAT   ;IGNORE FILE BOX ADDR.
        TST  @SDDAT        ;NO OTHER ADDR. OR GEN BITS
        BEQ  3S           ;SHOULD SHOW UP
        ERROR 23
3S:     SCOPE
        ;*
        ;*TEST THAT THE COUNTER MODULE WILL INTERRUPT ON OVERFLOW.
        ;*ON INTERRUPT CHECK ADDR AND GENERIC CODE, HALT ON OVERFLOW.
        ;*CHECK RIF EFFECT ON MODULE
        ;*
TST4G:  MOV   @TST4GI,@ICSVT ;SET UP FOR ICS INTERRUPT.
        MOV   #340,@ICSVT2  ;PRIORITY 7 ON INTERRUPT.
        MOV   @TST4GL,@SLPADR ;SET FOR ITERATIONS.
TST4GL: MOV   #340,PS        ;DON'T ALLOW INTERRUPTS.
        MOV   #4,@ICSR      ;ENABLE ICS TO INTERRUPT WHEN READY.

```

F04

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 29  
 DZICAC.NE2 21-MAR-77 15:28 INITIALIZE THE COMMON TAGS

```

1288 006630 012777 177777 172264 MOV #177777,2$B0ADR ;SET COUNTER TO ALL ONES.
1289 006636 104422 DELAY ;DELAY 40 MS.
1290 006640 005037 177776 CLR PS ;ALLOW INTERRUPTS.
1291 006644 000240 NOP
1292 006646 005077 172742 CLR 2ICSR
1293 006652 104014 ERROR 14 ;REPORT ERROR - COUNTER MODULE DIDN'T INTR.
1294 006654 104427 INTR
1295 006656 000457 BR TST4GE ;LOOP.
1296
1297 ;#RECEIVE ICS INTERRUPT HERE
1298
1299 006660 012706 001100 TST4GI: MOV #1100,SP ;RESET THE STACK POINTER.
1300 006664 017737 172726 001126 MOV 2ICAR,$B0DAT ;GET ADDR & GENERIC CODE.
1301 006672 013737 001122 001124 MOV $B0ADR,$G0DAT ;GET REAL ADDR.
1302 006700 006037 001124 ROR $G0DAT ;FORM ADDR. AS IT WOULD LOOK IN
1303 006704 142737 177777 001125 BICB #-1,$G0DAT+1 ;ICAR.
1304 006712 005077 172676 CLR 2ICSR
1305 006716 005037 177776 CLR PS
1306 006722 052737 003400 001124 BIS #003400,$G0DAT ;ADD GENERIC CODE.
1307 006730 023737 001124 001126 CMP $G0DAT,$B0DAT ;CHECK ADDR. + GEN. CODE.
1308 006736 001403 BEQ 15
1309
1310 006740 104015 ERROR 15 ;ADDR OR GENERIC CODE INCORRECT.
1311 006742 104427 INTR
1312 006744 000424 BR TST4GE ;LOOP.
1313
1314 006746 104422 15: DELAY ;DELAY 40 MS.
1315 006750 005037 001124 CLR $G0DAT
1316 006754 017737 172142 001126 MOV 2$B0ADR,$B0DAT
1317 006762 001402 BEQ 25
1318
1319 006764 104016 ERROR 16 ;COUNTER MODULE DIDN'T HALT ON OVERFLOW.
1320 006766 000413 BR TST4GE ;LOOP.
1321
1322 006770 052777 000001 172616 25: BIS #1,2ICSR ;SET RIF BIT IN ICS-11.
1323 006776 005777 172120 TST 2$B0ADR ;INITIATE RIF ON COUNTER MODULE.
1324 007002 000240 NOP
1325 007004 105777 172604 TSTB 2ICSR ;ANY INTR. PENDING ON ICS BUS?
1326 007010 100002 BPL TST4GE ;NO-THEN LOOP.
1327
1328 007012 104017 ERROR 17 ;RIF DIDN'T CLEAR INTERRUPT FLAG
1329 007014 104427 INTR
1330 007016 012777 000001 172570 TST4GE: MOV #1,2ICSR
1331 007024 005777 172072 TST 2$B0ADR
1332 007030 000004 SCOPE
1333
1334 ;*
1335 ;*TEST TO SEE IF RESET CLEARS INTERRUPT FLAG ON COUNTER
1336 ;*
1337
1338 007032 012737 000010 001160 TST4H: MOV #10,$TIMES
1339 007040 012777 177777 172054 15: MOV #-1,2$B0ADR ;START COUNTER COUNTING
1340 007046 012737 007040 001106 MOV #15,$LPADR
1341 007054 104422 DELAY ;DELAY FOR INTR.
1342 007056 104427 INTR ;SYSTEM INITIALIZE
1343 007060 105777 172530 TSTB 2ICSR ;INTERRUPT POSTED?

```

G04

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 30  
DZICAC.NE2 21-MAR-77 15:28 INITIALIZE THE COMMON TAGS

1344 007064 100001  
1345  
1346 007066 104024  
1347 007070 000004  
1348 007072 000137 006012  
1349  
1350 007076 000000  
1351 007100 000001  
1352 007102 000003  
1353 007104 000007  
1354 007106 000017  
1355 007110 000037  
1356 007112 000077  
1357 007114 000177  
1358 007116 000377  
1359 007120 000777  
1360 007122 001777  
1361 007124 003777  
1362 007126 007777  
1363 007130 017777  
1364 007132 037777  
1365 007134 077777  
1366 007136 177777  
1367 007140 100000  
1368 007142 000000  
1369  
1370  
1371  
1372  
1373  
1374  
1375

BPL 25  
25: ERROR 24  
SCOPE  
JMP TST4L  
CNTPAT: 0  
1  
3  
7  
17  
37  
77  
177  
377  
777  
1777  
3777  
7777  
17777  
37777  
77777  
177777  
100000  
CNTPAE: 0

; INIT FAILED TO CLEAR INTR. FLAG

; PATTERNS USED BY TST4C TO  
; DETERMINE IF COUNTER MODULE  
; CAN COUNT THRU EACH STATE.

```
1376                                     ;*  
1377                                     ;*TEST 5 A/D LOGIC TEST  
1378                                     ;*  
1379  
1380 007144 000000 ADCSR: 0 ;ADDR OF A005 UNDER TEST  
1381 007146 104401 TST5: TYPE ;TYPE HEADER.  
1382 007150 025534 MHT5  
1383 007152 104426 ADAR ;GET A005 ADDR.  
1384 007154 012746 007762 ADLOGL: MOV #ADLOGE,-(6)  
1385 007160 104401 022004 TYPE, MHK  
1386  
1387 007164 013737 013340 007144 ADLOG: MOV ADADR,ADCSR ;*GET A005'S ADDR.  
1388 007172 052777 000100 171744 BIS #100,#STKS  
1389 007200 005037 177776 CLR PS  
1390 007204 012737 007224 001106 MOV #1$,SLPADR  
1391 007212 012703 007776 MOV #ADPATP,R3 ;*GET PATTERN POINTERS.  
1392 007216 012737 077770 001124 MOV #077770,$GDDAT ;*DATA TO BE SENT  
1393 007224 013777 001124 177712 1$: MOV $GDDAT,#ADCSR ;*SEND DATA TO BE SENT  
1394 007232 017737 177706 001126 MOV #ADCSR,$BDDAT ;*GET DATA BACK.  
1395 007240 001001 BNE #2$ ;*IF DATA PRESENT GO AHEAD  
1396  
1397 007242 104001 ERROR 1 ;*ERROR "COULD NOT SEND /RECEIVE DATA"  
1398 ;*FROM A005  
1399 007244 000004 2$: SCOPE  
1400 ;*BASIC "BIT BANG" OF A005  
1401  
1402 007246 011337 001124 ADLOG2: MOV (3),$GDDAT ;*GET PATTERN TO BE SENT.  
1403 007252 013777 001124 177664 MOV $GDDAT,#ADCSR ;*SEND PATTERN TO ADCSR.  
1404 007260 017737 177660 001126 MOV #ADCSR,$BDDAT ;*GET TO BACK.  
1405 007266 023737 001124 001126 CMP $GDDAT,$BDDAT ;*DATA SENT=DATE RECIEVED?  
1406 007274 001401 BEQ #2$ ;*IF SO-CONTINUE  
1407  
1408 007276 104002 ERROR 2 ;*REPORT ERROR: "CSR READ/WRITE ERROR"  
1409  
1410 007300 000004 2$: SCOPE ;*LOOP  
1411  
1412 007302 062703 000002 1$: ADD #2,R3 ;*UPDATE PATTERN POINTERS  
1413 007306 020327 010020 CMP R3,#ADPATE ;*DONE ALL PATTERNS?  
1414 007312 003755 BLE ADLOG2 ;*IF NOT-CONTINUE NEXT PATTERN.  
1415  
1416 ;*WILL CONVERTING SET AND THEN CLEAR?  
1417  
1418 007314 012737 104000 001124 ADLOG3: MOV #104000,$GDDAT ;*SET CONVERT,READ CSR  
1419 007322 012737 007322 001106 1$: MOV #1$,SLPADR  
1420 007330 013777 001124 177606 MOV $GDDAT,#ADCSR ;*SEND TO ADCSR  
1421 007336 017737 177602 001126 MOV #ADCSR,$BDDAT ;*GET A005 CSR  
1422 007344 023737 001124 001126 CMP $GDDAT,$BDDAT ;*DID CONVERT BIT SET?  
1423 007352 001402 BEQ ADL03A  
1424 007354 104003 ERROR 3 ;*ERROR: "CONVERT BIT FAILED TO SET"  
1425 007356 000411 BR ADLOG4  
1426  
1427 007360 013737 001560 001566 ADL03A: MOV FR16,FREQ ;*SET TO DELAY 16 MILLISEC.  
1428 007366 104422 DELAY  
1429 007370 005777 177550 TST #ADCSR ;*DID CONVERT BIT CLEAR?  
1430 007374 100002 BPL 1$+2  
1431
```

```

1432 007376 104004          ERROR 4          ;*ERROR "CONVERT BIT FAILED TO CLEAR"
1433
1434
1435
1436 007400 104427          1$:  INTR
1437 007402 000004          ADLOG4: SCOPE          ;*LOOP.
1438
1439
1440
1441 007404 012777 052770 177532 ADLOG5: MOV #52770, @ADCSR ;*LOAD CSR WITH ALL BUT READ CR BIT
1442 007412 022777 052770 177524      CMP #52770, @ADCSR ;*DID WE READ THE CR?
1443 007420 001001          BNE 1$ ;*IF NOT NO ERROR.
1444
1445 007422 104005          ERROR 5          ;*REPORT ERROR "CANNOT READ A005 DATA
1446
1447
1448 007424 000004          1$:  SCOPE          ;*REGISTER WITH READ BIT CLEARED."
1449
1450
1451
1452 007426 012777 000340 172166 ADLOG6: MOV #340, @ICSVT2
1453 007434 012777 007506 172156      MOV @ADLOG7, @ICSVT ;*SET VECTOR FOR INTER.
1454 007442 012777 000004 172144      MOV #4, @ICSR ;*ALLOW ICS TO INT.
1455 007450 012777 104000 177466      MOV #104000, @ADCSR ;*SET A005 TO CONVERT.
1456 007456 013737 001560 001566      MOV FR16, FREQ ;*ALLOW UP TO 16 MILLI SEC FOR NTO.
1457 007464 005037 177776          CLR PS
1458 007470 104422          DELAY
1459 007472 000240          NOP
1460 007474 104006          ERROR 6          ;*REPORT ERROR "A005 FAILED TO INTER. AT
1461 007476 005077 172112          ADL6L: CLR @ICSR
1462 007502 000004          SCOPE
1463 007504 000445          BR ADLOG9
1464
1465
1466
1467 007506 022626          ADLOG7: POPSP2 ;*A005 INTR TO HERE RESET SP.
1468 007510 017737 172102 001126      MOV @ICAR, $BDDAT ;*GET ADDR. AND GEN CODE
1469 007516 013737 007144 001124      MOV @ADCSR, $GDDAT ;*FORM GOOD ADDR.
1470 007524 042737 177000 001124      BIC #177000, $GDDAT
1471 007532 006237 001124          ASR $GDDAT
1472 007536 052737 003400 001124      BIS #003400, $GDDAT ;*ADD GENERIC CODE
1473 007544 023737 001126 001124      CMP $BDDAT, $GDDAT ;*IS ADDR + GENERIC CODE OK?
1474 007552 001402          BEQ ADLOG8
1475
1476 007554 104007          ERROR 7          ;*REPORT ERROR "A005 ADDR. OR GENERIC
1477
1478
1479 007556 000747          BR ADL6L          ;*CODE INCORRECT.
1480
1481
1482
1483 007560 052777 000001 172026 ADLOG8: BIS #1, @ICSR ;*SET RIF BIT.
1484 007566 005777 177352          TST @ADCSR ;*PUT IT TO WORK ON A005.
1485 007572 012777 007612 172020      MOV #1$, @ICSVT ;*SET INTERRUPT VECTORS.
1486 007600 005037 177776          CLR PS ;*ALLOW INTERRUPTS
1487 007604 104422          DELAY
    
```



```

1488 007606 000240      NOP
1489 007610 000732      BR      ADL6L      ;#EXIT A/D TESTS.
1490
1491 007612 022626      1S:    POPSP2      ;#RESET SP.
1492 007614 104010      ERROR  10      ;#REPORT ERROR" RIF DID NOT CLEAR INTR.
1493                                ;#FLAG ON A005".
1494 007616 000727      BR      ADL6L
1495
1496                                ;*
1497                                ;#A005 DUAL ADDRESSING TEST
1498                                ;*
1499
1500 007620 012737 007710 001106 ADLOG9: MOV      #2S, $LPADR
1501 007626 005037 001122      CLR      $BDADR      ;SET FIRST ADDR.
1502 007632 053737 001612 001122      BIS      ICSMOD, $BDADR
1503 007640 162737 000002 001122      SUB      #2, $BDADR
1504 007646 013737 001612 013134      MOV      ICSMOD, RTEMP ;FIX STOP ADDR.
1505 007654 062737 000040 013134      ADD      #40, RTEMP
1506 007662 062737 000002 001122      1S:    ADD      #2, $BDADR ;UPDATE ADDR.
1507 007670 023737 001122 013134      CMP      $BDADR, RTEMP ;DONE ALL ADDRS?
1508 007676 001427      BEQ      6S
1509 007700 023737 001122 007144      CMP      $BDADR, ADCSR ;SAME ADDR AS A005 UNDER TEST?
1510 007706 001765      BEQ      1S
1511 007710 012777 004100 177226      2S:    MOV      #4100, $ADCSR ;SEND DATA TO A/D.
1512 007716 022777 004100 171176      CMP      #4100, $BDADR ;DUAL ADDR. READ ERROR?
1513 007724 001001      BNE      3S
1514
1515 007726 104026      ERROR  26      ;DUAL ADDR. READ ERROR
1516 007730 000004      3S:    SCOPE
1517
1518 007732 012777 004200 171162      4S:    MOV      #4200, $BDADR ;CHECK FOR WRITE ERROR
1519 007740 022777 004200 177176      CMP      #4200, $ADCSR ;BAD WRITE?
1520 007746 001001      BNE      5S
1521
1522 007750 104027      ERROR  27      ;DUAL ADDR. WRITE ERROR
1523
1524 007752 000004      5S:    SCOPE
1525 007754 000742      BR      1S
1526 007756 104427      6S:    INTR
1527 007760 000207      RTS      PC      ;RETURN TO
;ADLOGE
1528
1529
1530                                ;#END OF A/D TESTS
1531
1532 007762 005237 001100      ADLOGE: INC      $PASS
1533 007766 104401      1S:    TYPE
1534 007770 021465      MEND
1535 007772 000137 007154      JMP      ADLOGL ;#RETURN TO MONITOR.
1536
1537 007776 004000      ADPATP: 004000
1538 010000 004020      004020
1539 010002 004040      004040
1540 010004 004100      004100
1541 010006 004200      004200
1542 010010 004400      004400
1543 010012 005000      005000

```

K04

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 34  
DZICAC.NE2 21-MAR-77 15:28 INITIALIZE THE COMMON TAGS

1544	010014	006000		006000	
1545	010016	004000		004000	
1546	010020	004000		ADPATE: 004000	
1547					
1548	010022	000137	003374	JMP	START1
1549					

L04

```

1550                                     ;*
1551                                     ;*TEST 6 A/D CALIBRATION TEST
1552                                     ;*
1553
1554 010026 104401 TST6: TYPE ;TYPE HEADER.
1555 010030 025570 MMT6
1556 010032 104426 ADAR ;GET A005 ADDR.
1557 010034 104431 QUBR ;UNI OR BI-POLAR?
1558                                     ;TBCA AUDIO1 CALIBRATION ROUTINE
1559                                     ;SWR0=1 RETURN TO MONITR
1560                                     ;SWR4-6 SELECT CHANNEL
1561                                     ;SWR7-10 SELECT MUX.
1562                                     ;SWR11=1 TYPE OUT CONVERSIONS
1563                                     ;SWR11=0 DISPLAY CONVERSIONS
1564                                     ;SWR12-14 SELECT GAIN (1 TO 1000)
1565
1566 010036 104401 024775 TYPE, MSWD
1567 010042 104401 022004 TYPE, MUX
1568 010046 052777 000100 171070 TBCAA: BIS #100,2SWR ;ALLOW TTY INTERRUPT
1569 010054 013737 013340 007144 MOV ADADR,ADCSR
1570 010062 001004 BNE 15
1571 010064 104401 TYPE
1572 010066 024746 MNAO
1573 010070 000137 003374 JMP START1
1574 010074 032777 000001 171036 15: BIT #1,2SWR ;CHECK THE SWITCH REGISTER
1575 010102 001402 BEQ .+6 ;FOR RETURN TO MONITOR
1576 010104 000137 003374 JMP START1
1577 010110 004737 010740 JSR PC,CONVER ;START CONVERSION
1578 010114 017700 177024 MOV 2ADCSR,%0 ;GET RESULTS
1579 010120 004737 010726 JSR PC,REPET7 ;RIGHT JUSTIFY
1580 010124 032777 004000 171006 BIT #4000,2SWR ;DISPLAY OR TYPE DATA?
1581 010132 001016 BNE CALTP ;TYPE IT
1582 010134 023737 001600 001576 CMP FR1120,FR1
1583 010142 001411 BEQ 25
1584 010144 022737 000176 001140 CMP #176,SWR ;DISPLAY AVAILABLE?
1585 010152 001405 BEQ 25 ;IF NOT, THEN BRANCH
1586 010154 000005 RESET
1587 010156 000005 RESET
1588 010160 000005 RESET
1589 010162 000005 RESET
1590 010164 000005 RESET
1591 010166 25:
1592 010166 000727 CALTP: BR TBCAA
1593 010170 104401 TYPE
1594 010172 001163 SCRLF
1595 010174 004737 010240 JSR %7,CALIT ;CONVERT DATA TO BINARY
1596 010200 013737 001550 001566 MOV FR110,FREQ ;LET TTY SETTLE DOWN.
1597 010206 104422 DELAY
1598 010210 104401 TYPE
1599 010212 025146 MCALT1
1600 010214 017700 176724 MOV 2ADCSR,%0 ;GET DATA
1601 010220 004737 010726 JSR PC,REPET7
1602 010224 RO TYOCT
1603 010224 010046 MOV RO,-(SP) ;;SAVE RO FOR TYPEOUT
1604 010226 104402 TYPC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1605 010230 104401 TYPE

```

1606	010232	025125			MCALOT			
1607	010234	104422			DELAY			
1608	010236	000703			BR	TBCAA		
1609	010240	005737	013342		TST	BIPOL		
1610	010244	001406			BEQ	35		
1611	010246	022700	007777		CMP	#7777,RO		
1612	010252	001017			BNE	25		
1613	010254	104401			TYPE			
1614	010256	025065			MPOVFL			
1615	010260	000207			RETURN			
1616	010262							
1617	010262	022700	004000		35: CMP	#4000, %0		;ROUTINE TO CONVERT BINARY
1618	010266	001003			BNE	15		;READ FROM A/D BUFFER TO SIX
1619	010270	104401			TYPE			;PLACE DECIMAL VOLTAGE
1620	010272	025036			MMOVFL			
1621	010274	000207			RTS	PC		
1622	010276	022700	003777		15: CMP	#3777, %0		;POSITIVE OR NEGATIVE
1623	010302	001003			BNE	25		
1624	010304	104401			TYPE			
1625	010306	025065			MPOVFL			
1626	010310	000207			RTS	PC		
1627	010312	012703	010610		25: MOV	#A1, %3		;CLEAR STORAGE AREA
1628	010316	005023			CLR	(3)+		
1629	010320	020327	010624		CMP	%3, #A1+14		;DETERMINE IF
1630	010324	001374			BNE	.-6		
1631	010326	012703	010610		MOV	#A1, %3		;POSITIVE OR
1632	010332	012702	002000		MOV	#2000, %2		
1633	010336	005737	013342		TST	BIPOL		
1634	010342	001401			BEQ	+.4		
1635	010344	006302			ASL	%2		
1636	010346	012701	010632		MOV	#HOLDIT,%1		;NEGATIVE VOLTAGE
1637	010352	005737	013342		TST	BIPOL		
1638	010356	001012			BNE	CALIT1		
1639	010360	032700	004000		BIT	#4000,RO		
1640	010364	001405			BEQ	CALITP		
1641	010366	104401			TYPE			;IF MINUS TYPE "--"
1642	010370	025172			MMINUS			
1643	010372	005100			COM	%0		
1644	010374	005200			INC	%0		
1645	010376	000402			BR	CALIT1		;IF POSITIVE
1646	010400	104401			CALITP: TYPE			;TYPE "+"
1647	010402	025174			MPLUS			
1648	010404	030200			CALIT1: BIT	%2, %0		
1649	010406	001044			BNE	CALIT2		;DO OCTAL TO
1650	010410	062701	000005		ADD	#5, %1		
1651	010414	006002			CALITE: ROR	%2		;DECIMAL CONVERSIONS
1652	010416	001372			BNE	CALIT1		
1653	010420	005737	010606		TST	HOLDTM		;BY METHOD OF INCREMENTING
1654	010424	001403			BEQ	+.10		
1655	010426	112737	000001 010610		MOV8	#1, A1		
1656	010434	005037	010606		CLR	HOLDTM		;DECIMAL COUNTER AND
1657	010440	012703	010610		MOV	#A1, %3		
1658	010444	052723	000260		BIS	#260, (3)+		
1659	010450	022703	010624		CMP	#A1+14, %3		;DECREMENTING OCTAL DATA
1660	010454	001373			BNE	.-10		
1661	010456	012703	010611		MOV	#A1+1, %3		;UNTIL ZERO

1662	010462	112723	000001		MOVW	#1,	(3)+	
1663	010466	105203			INCB	%3		
1664	010470	112723	000256		MOVW	#256,	(3)+	;IF DECIMAL COUNTER = 12 (OCTAL)
1665	010474	105203			INCB	%3		
1666	010476	112723	000001		MOVW	#1	(3)+	
1667	010502	022703	010624		CMP	#A1+14,	%3	;CLEAR IT AND INCREMENT
1668	010506	001372			BNE	.-12		;NEXT COUNTER
1669	010510	105013			CLRB	(3)		
1670	010512	104401			TYPE			;TYPE DECIMAL
1671	010514	010610			AI			
1672	010516	000207			RTS	%7		
1673	010520	005037	010606	CALIT2:	CLR	HOLDTM		
1674	010524	012703	010624		MOV	#A1+14,	%3	;FIRST USE TABLE HOLDIT
1675	010530	112137	010604	CALIT3:	MOVW	(1)+,	SCAN	;TO GET DECIMAL VALUE FOR DATA
1676	010534	063743	010604		ADD	SCAN,	-(3)	;COMPUTED FROM ADDS
1677	010540	005737	010606		TST	HOLDTM		
1678	010544	001404			BEQ	CALIT4		
1679	010546	005037	010606		CLR	HOLDTM		;ADD THESE VALUES TOGETHER
1680	010552	062713	000001		ADD	#1,	(3)	
1681	010556	122713	000012	CALIT4:	CMPB	#12,	(3)	;HOLDIT CONTAINS VALUES FOR DATA
1682	010562	101004			BHI	CALIT5		
1683	010564	162713	000012		SUB	#12,	(3)	;BASED ON GAIN OF A/D
1684	010570	005237	010606		INC	HOLDTM		
1685	010574	022703	010612	CALIT5:	CMP	#A1+2,	%3	;AT TIME DATA WAS TAKEN
1686	010600	001353			BNE	CALIT3		
1687	010602	000704			BR	CALITE		
1688	010604	000000		SCAN:	0			
1689	010606	000000		HOLDTM:	000000			
1690	010610	000000		AI:	000000			
1691		010632		.=. +20				
1692	010632	000	000	000	HOLDIT:	.BYTE	0,0,0,0,5	;TABLE OF VOLTAGE REPRESENTATION
1693	010635	000	005					
1694								
1695	010637	000	000	000	.BYTE	0,0,0,5,2		;OF BINARY INPUT.
1696	010642	005	002					
1697								
1698	010644	000	000	005	.BYTE	0,0,5,2,1		
1699	010647	002	001					
1700								
1701	010651	000	005	002	.BYTE	0,5,2,6,0		
1702	010654	006	000					
1703								
1704	010656	005	002	001	.BYTE	5,2,1,3,0		
1705	010661	003	000					
1706								
1707	010663	003	006	005	.BYTE	3,6,5,1,0		
1708	010666	001	000					
1709								
1710	010670	001	010	007	.BYTE	1,8.,7,0,0		
1711	010673	000	000					
1712								
1713	010675	001	011	003	.BYTE	1,9.,3,0,0		
1714	010700	000	000					
1715								
1716	010702	005	011	001	.BYTE	5,9.,1,0,0		
1717	010705	000	000					

```

1718
1719 010707      010      011      000      .BYTE    8.,9.,0,0,0
1720 010712      000      000
1721
1722 010714      011      004      000      .BYTE    9.,4,0,0,0
1723 010717      000      000
1724
1725 010721      004      002      000      .BYTE    4,2,0,0,0
1726 010724      000      000
1727
1728                      .EVEN
1729
1730 010726      006000                      REPET7: ROR      %0
1731 010730      006000                      ROR      %0                      ;ROUTINE TO RIGHT
1732 010732      006000                      ROR      %0                      ;JUSTIFY DATA BY
1733 010734      006000                      ROR      %0                      ;ROTATING IT 4 PLACE TO
1734 010736      000207                      RTS      PC                      ;THE RIGHT
1735 010740      017702      170174                      CONVER: MOV      @SWR,R2
1736 010744      042702      000017                      BIC      #17,%2                      ;THISROUTINE SAMPLES SWR
1737 010750      052702      104000                      BIS      @104000,%2                      ;AND SENDS CHANNEL GAIN
1738 010754      010277      176164                      MOV      %2,@ADCSR                      ;CONVERT AND READ BITS TO ADU01
1739 010760      005777      176160                      CONVED: TST      @ADCSR
1740 010764      100775                      BMI      -4                      ;TO START CONVERSION
1741 010766      005077      176152                      CLR      @ADCSR                      ;AND WAIT FOR CONVERSION COMPLETE
1742 010772      000207                      RTS      PC
1743
1744 010774      000137      003374                      JMP      START1
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
  
```

```

1755
1756
1757
1758
1759
1760
1761 011000 104401
1762 011002 025625
1763 011004 104426
1764 011006 104431
1765 011010 013737 013340 007144
1766 011016 013737 001550 001566
1767 011024 104422
1768 011026 004737 007164
1769 011032 005737 001652
1770 011036 001002
1771 011040 104401
1772 011042 025763
1773 011044 012737 013100 011066
1774 011052 012737 000001 011070
1775 011060 000401
1776 011062 000763
1777 011064 104414
1778 011066 013100
1779 011070 000001
1780
1781 011072 012700 013154
1782 011076 005710
1783 011100 001003
1784 011102 104401
1785 011104 026333
1786 011106 000751
1787 011110 012001
1788 011112 042701 170000
1789 011116 020137 013100
1790 011122 001365
1791 011124 014037 013100
1792
1793 011130 005737 001652
1794 011134 001002
1795 011136 104401
1796 011140 026000
1797 011142 005037 013104
1798 011146 005037 013106
1799 011152 005037 013110
1800 011156 005037 013112
1801 011162 013737 001650 013134
1802 011170 162737 027240 013134
1803 011176 005037 013136
1804 011202 005237 013136
1805
1806 011206 162737 001000 013134
1807 011214 100372
1808 011216 005337 013136
1809 011222 013737 013136 013114
1810 011230 013746 013136

```

```

; *
; *TEST 7 ROUTINE TO MANUALLY TAKE SAMPLES, COMPARE RESULTS
; *PRINT ERROR IF ANY
; *CALL=JMP TST8 FROM MONITOR
; *
TST7:  TYPE
      MHT8
      ADAR
      QUBR
      MOV  ADADR,ADCSR
      MOV  FR110,FREQ
      DELAY
      JSR  PC,ADLOG
      TST  EXPERT
      BNE  .+6
      TYPE
      MGIN
      MOV  #GAIN,10S
      MOV  #1,10S+2
      BR   99S
      BR   22S
      INOCT
      GAIN
      I
      MOV  #GLIST,RO
      TST  (0)
      BNE  21S
      TYPE
      MSG
      BR   22S
      MOV  (0)+,R1
      BIC  #170000,R1
      CMP  R1,GAIN
      BNE  20S
      MOV  -(0),GAIN
      TST  EXPERT
      BNE  .+6
      TYPE
      MCHAN
      CLR  CHANS
      CLR  CHANF
      CLR  CHANSR
      CLR  CHANFR
      MOV  CORSIZ,RTEMP
      SUB  #SENDAD,RTEMP
      CLR  RTEMP1
      INC  RTEMP1
      SUB  #1000,RTEMP
      BPL  15S
      DEC  RTEMP1
      MOV  RTEMP1,CHANNO
      MOV  RTEMP1,-(SP)

```

```

;TYPE HEADER
;ALLOW TTY TO SETTLE BEFORE CONT.
;BASIC LOGIS CHECK
;ASK FOR GAIN
;GET GAIN
;GET LIST OF LEGAL GAINS.
;AT END OF LIST?
;NO-CONTINUE.
;GAIN HE TYPED IS KNOWN.
;TELL HIM.
;REASK QUESTION.
;GET GAIN FROM GAIN LIST.
;MASK OUT REAL GAIN BITS.
;GAIN HE TYPED MATCH ONE IN GAIN LIST?
;NO-CHECK NEXT IN LIST.
;YES-REPLACE TYPED GAIN BY REAL GAIN.
;ASK FOR CHANS:

```

1811	011234	004737	016440			GOSUB	SSB20		
1812	011240	062716	000010			ADD	#10,(SP)		
1813	011244	005726				TST	(SP)+		
1814	011246	012737	013110	011270		MOV	#CHANSR,115		
1815	011254	012737	000002	011272		MOV	#2,115+2		
1816	011262	000401				BR	+4		
1817	011264	000721				BR	35		
1818	011266	104414				INOC	T		;GET CHANS
1819	011270	013110			115:	CHANSR	2		
1820	011272	000002				TST	CHANFR		
1821	011274	005737	013112						
1822									
1823	011300	001003				BNE	.+10		
1824	011302	013737	013110	013112		MOV	CHANSR,CHANFR		
1825	011310	023737	013110	013112		CMP	CHANSR,CHANFR		;CHAN. S+F?
1826	011316	003403				BLE	45		
1827	011320	104401				TYPE			
1828	011322	026047				MCHER1			
1829	011324	000701				BR	35		
1830	011326	023727	013112	000177	45:	CMP	CHANFR,#177		;CHAN WITHIN LEGAL BOUNDS?
1831	011334	003403				BLE	305		
1832	011336	104401				TYPE			;NO-THEN TELL HIM
1833	011340	026253				MCHANH			
1834	011342	000672				BR	35		
1835	011344	005737	001652		305:	TST	EXPERT		
1836	011350	001002				BNE	.+6		
1837	011352	104401				TYPE			;ASK FOR EXPECTED AVERAGE.
1838	011354	026302				MAVEQ			
1839	011356	012737	013124	011400		MOV	#AVEXP,125		
1840	011364	012737	000001	011402		MOV	#1,125+2		
1841	011372	000401				BR	+4		
1842	011374	000763				BR	305		
1843	011376	104414				INOC	T		;GET AVERAGE.
1844	011400	013124			125:	AVEXP	1		
1845	011402	000001				BIT	#170000,AVEXP		;LEGAL AVERAGE?
1846	011404	032737	170000	013124		BEQ	65		
1847	011412	001403				TYPE			;NO TELL HIM(OR HER) ASK:
1848	011414	104401				MNTL			;QUESTION AGAIN.
1849	011416	026353				BR	305		
1850	011420	000751				MOV	#TOLER,135		
1851	011422	012737	013122	011502	65:	MOV	#1,135+2		
1852	011430	012737	000001	011504		BIT	#4000,AVEXP		
1853	011436	032737	004000	013124		BEQ	605		
1854	011444	001406				TST	BIPOL		
1855	011446	005737	013342			BNE	605		
1856	011452	001003				BIS	#170000,AVEXP		
1857	011454	052737	170000	013124	605:	TST	EXPERT		
1858	011462	005737	001652			BNE	.+6		
1859	011466	001002				TYPE			;ASK FOR TOLERANCE.
1860	011470	104401				MTOL			
1861	011472	026025				BR	.+4		
1862	011474	000401				BR	605		
1863	011476	000771				INOC	T		
1864	011500	104414			135:	TOLER	1		
1865	011502	013122							
1866	011504	000001							



```

1867 011506 005737 001660          TST      LINEPR
1868 011512 001411          BEQ      14$
1869 011514 013737 001662 001150      MOV     LPCSR,STPS
1870 011522 013737 001664 001152      MOV     LPDBR,STPB
1871 011530 104401 022004          TYPE,   MNK
1872 011534 000402          BR      .+6
1873 011536 104401          14$:   TYPE "REPEAT"
1874 011540 025114          MREP
1875 011542 013737 013110 013106      MOV     CHANSR,CHANF
1876 011550 005337 013106          DEC     CHANF
1877 011554 032777 000001 167356      BIT     #1,25MR
1878 011562 001402          BEQ     .+6
1879 011564 000137 003374          JMP     START1
1880 011570 013737 013106 013104 5$:   MOV     CHANF,CHANS
1881 011576 005237 013104          INC     CHANS
1882 011602 063737 013114 013106      ADD     CHANNO,CHANF
1883 011610 023737 013104 013112      CMP     CHANS,CHANFR
1884 011616 003347          BGT     14$
1885 011620 023737 013106 013112      CMP     CHANF,CHANFR
1886 011626 003403          BLE     .+10
1887 011630 013737 013112 013106      MOV     CHANFR,CHANF
1888 011636 004737 011650          GOSUB  ,SAMPR          ;TAKE CONVERSIONS
1889 011642 004737 012152          GOSUB  ,AVERR         ;TAKE AVERAGES
1890 011646 000750          BR      5$
1891
1892
1893          ;*
1894          ;*ROUTINE TO TAKE 256 SAMPLES OF AL CHANNELS SPECIFIED
1895          ;*CALL=GOSUB,SAMPR
1896 011650 013737 001552 001566 SAMPR: MOV     FRSD,FREQ          ;SUB PART TO SET UP DELAY BASEDON
1897 011656 013737 013106 013076      MOV     CHANF,CHAN
1898 011664 163737 013104 013076      SUB     CHANS,CHAN
1899 011672 013737 013076 013120      MOV     CHAN,SAMOFF
1900 011700 001416          BEQ     2$
1901 011702 006337 013120          ASL     SAMOFF
1902 011706 163737 001562 001566 1$:   SUB     FRS,FREQ
1903 011714 005337 013076          DEC     CHAN
1904 011720 001372          BNE     1$
1905 011722 005737 001566          TST     FREQ
1906 011726 100003          BPL     2$
1907 011730 012737 000001 001566      MOV     #1,FREQ
1908 011736 012737 177400 013116 2$:   MOV     #256,SAMCNT      ;SET SAMPLE COUNT.
1909 011744 012701 027240          MOV     #BUFFER,R1      ;SET FOR STORAGE.
1910 011750 013737 013104 013076 3$:   MOV     CHANS,CHAN      ;GET STARTING CHANNEL
1911 011756 104422          DELAY
1912 011760 013737 013076 013074 4$:   MOV     CHAN,CHAN1      ;SET TO RIGHT JUSTIFY CHAN.
1913 011766 006337 013074          ASL     CHAN1           ;FIX TO LOAD INTO AID WORD.
1914 011772 006337 013074          ASL     CHAN1
1915 011776 006337 013074          ASL     CHAN1
1916 012002 006337 013074          ASL     CHAN1
1917 012006 004737 012044          GOSUB  ,CONVT          ;TAKE CONVERSION.
1918 012012 010021          MOV     RD,(1)+        ;STORE RESULT
1919 012014 005237 013076          INC     CHAN           ;READY FOR NEXT CHAN.
1920 012020 023737 013076 013106      CMP     CHAN,CHANF      ;DONE ALL CHANNELS
1921 012026 003754          BLE     4$
1922 012030 005237 013116          INC     SAMCNT         ;DONE 256 SAMPLES?

```

```

1923 012034 001345      BNE      3$
1924 012036 012701 027240  MOV      #BUFFER,R1
1925 012042 000207      RETURN
1926
1927      ;*ROUTINE TO FORM A AID WORD FOR CONVERSIONS, START A CONVERSION,
1928      ;*WAIT FOR DONE, AND ENABLE READING OF THE AID DBR.
1929
1930      ;*CALL=GOSUB,CONVT,RESULT FOUND IN RO,RIGHT JUSTIFIED
1931 012044 013737 013100 013102  CONV:  MOV      GAIN,ADWD      ;SET GAIN IN WORD
1932 012052 042737 007777 013102      BIC      #7777,ADWD
1933 012060 053737 013074 013102      BIS      CHAN1,ADWD      ;SET CHAN.
1934 012066 052737 104000 013102      BIS      #104000,ADWD      ;SET READ BIT, CONVERT BIT.
1935 012074 013777 013102 175042      MOV      ADWD,ADCSR      ;START CONVERSION.
1936 012102 005777 175036      1$:      TST      ADCSR      ;WAIT FOR CONVERSION COMPLETE.
1937 012106 100775      BMI      1$
1938 012110 005077 175030      CLR      ADCSR      ;ENABLE READING OF DBR.
1939 012114 017700 175024      MOV      ADCSR,RO      ;READ AID RESULTS PUT IN RO
1940 012120 005737 013342      TST      BIPOL
1941 012124 001405      BEQ      2$
1942 012126 006000      ROR      RO
1943 012130 006000      ROR      RO
1944 012132 006000      ROR      RO
1945 012134 006000      ROR      RO
1946 012136 000207      RETURN
1947 012140 006200      2$:      ASR      RO      ;RIGHT JUSTIFY, REMEMBERING SIGN.
1948 012142 006200      ASR      RO
1949 012144 006200      ASR      RO
1950 012146 006200      ASR      RO
1951 012150 000207      RETURN      ;RETURN
1952
1953      ;*
1954      ;*AVERAGING ROUTINE USED BE TEST 7
1955      ;*AT THIS POINT IN TIME ALL SAMPLES FOR ALL CHANNELS HAVE
1956      ;*BEEN TAKEN AND STORED IN "BUFFER" IN A
1957      ;*SEQUENTIAL INTERLEAVED BUFFER FORM" GIVEN BY THE FOLLOWING FORMULA:
1958      ;*"L=2*N+R1" WHERE L EQUALS THE LOCATION OF A SAMPLE,
1959      ;*N EQUALS THE NUMBER OF CHANNELS SAMPLES WERE TAKEN ON, AND R1
1960      ;*IS THE BUFFER POINTER SET TO "BUFFER" INITIALLY.
1961      ;*CALL=GOSUB,AVERR
1962 012152 013737 013104 013076  AVERR:  MOV      CHANS,CHAN      ;SET TO FIRST CHAN.
1963 012160 005037 013074      CLR      CHAN1      ;1ST CHAN OFFSET
1964 012164 012737 000400 013116  AVERRL:  MOV      #256.,SAMCNT
1965 012172 013701 013074      MOV      CHAN1,R1      ;PUT CHAN OFFSET IN R1.
1966 012176 062701 027240      ADD      #BUFFER,R1      ;ADD BUFFER POINTER TO R1.
1967 012202 005037 013126      CLR      AVTKN      ;SET INITIAL CONDITIONS FOR THIS CHAN.
1968 012206 005037 013134      CLR      RTEMP
1969 012212 011137 013130      MOV      (1),RLOW
1970 012216 011137 013132      MOV      (1),RHIGH
1971 012222 023711 013132      2$:      CMP      RHIGH,(1)      ;FIND REAL HIGH VALUE.
1972 012226 003002      BGT      3$
1973 012230 011137 013132      MOV      (1),RHIGH
1974 012234 021137 013130      3$:      CMP      (1),RLOW      ;FIND REAL LOW VALUE.
1975 012240 003002      BGT      4$
1976 012242 011137 013130      MOV      (1),RLOW
1977 012246 012137 013136      4$:      MOV      (1),RTEMP1      ;GET CURRENT SAMPLE.
1978 012252 005737 013342      TST      BIPOL
    
```

```

1979 012256 001003      BNE      .+10
1980 012260 062737 004000 013136  ADD      #4000,RTEMP1 ;ADD CONSTANT TO SAMPLE.
1981 012266 063737 013136 013126  ADD      RTEMP1,AVTKN ;"BOOT" ADD ALL SAMPLES.
1982 012274 005537 013134      ADC      RTEMP
1983 012300 063701 013120      ADD      SAMOFF,R1 ;UPDATE TO LOOK AT NEXT CHAN SAMPLE
1984 012304 005337 013116      DEC      SAMCNT ;DONE ALL SAMPLES PER THIS CHAN?
1985 012310 001344      BNE      2$ ;NO-DO NEXT SAMPLE.
1986 012312 013737 013124 013136  MOV      AVEXP,RTEMP1 ;YES! SEE IF RHIGH OK:
1987 012320 063737 013122 013136  ADD      TOLER,RTEMP1
1988 012326 023737 013136 013132  CMP      RTEMP1,RHIGH
1989 012334 002427      BLT      ERAV1 ;NO-THEN REPORT ERROR.
1990 012336 163737 013122 013136  SUB      TOLER,RTEMP1 ;YES-OK-CHECK LOWEST READING.
1991 012344 163737 013122 013136  SUB      TOLER,RTEMP1
1992 012352 023737 013130 013136  CMP      RLOW,RTEMP1
1993 012360 002415      BLT      ERAV1 ;NO-REPORT ERROR.
1994 012362 005737 013122      TST      TOLER ;DOES OPERATOR WISH "FORCED" TYPEOUT?
1995 012366 001415      BEQ      ERAV2 ;IF SO-DO IT
1996 012370 062737 000002 013074  AVERRN: ADD      #2,CHAN1 ;SET TO DO NEXT CHAN-BUT
1997 012376 005237 013076      INC      CHAN ;IF DONE ALL CHANS-EXIT-
1998 012402 023737 013076 013106  CMP      CHAN,CHANF ;OTHERWISE LOOP.
1999 012410 003665      BLE      AVERRL
2000 012412 000207      RETURN
2001
2002 ;*ERROR REPORTER
2003
2004 012414 104401      ERAV1: TYPE ;TYPE "REPEATIBILITY ERROR"
2005 012416 026447      MREPER
2006 012420 000402      BR ;
2007 012422 104401      ERAV2: TYPE .+6 ;TYPE "REPEATIBILITY FORCED TYPEOUT"
2008 012424 026411      MREPFT
2009 012426 012702 000007      1$: MOV      #7,R2 ;FIND AVERAGE.
2010 012432 006237 013134      10$: ASR      RTEMP ;
2011 012436 006037 013126      ROR      AVTKN ;AVERAGE=TOTAL OF SAMPLES-
2012 012442 005302      DEC      R2 ;DIVIDED BY 256.
2013 012444 001372      BNE      10$
2014 012446 006237 013134      ASR      RTEMP
2015 012452 006037 013126      ROR      AVTKN
2016 012456 005537 013126      ADC      AVTKN
2017 012462 005737 013342      TST      BIPOL
2018 012466 001003      BNE      .+10
2019
2020 012470 162737 004000 013126  SUB      #4000,AVTKN ;SUBTRACT CONSTANT.
2021 012476 104401      TYPE ;TYPE HEADER 1
2022 012500 026476      MREPT1
2023 012502      TYPOCT CHAN ;TYPE CHAN.
2024 012502 013746 013076  MOV      CHAN,-(SP) ;SAVE CHAN FOR TYPEOUT
2025 012506 104402      TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2026 012510 104401      TYPE
2027 012512 026406      M2SP
2028 012514 013702 013100  MOV      GAIN,R2 ;TYPE GAIN.
2029 012520 042702 170000  BIC      #170000,R2
2030 012524      TYPOCT R2
2031 012524 010246  MOV      R2,-(SP) ;SAVE R2 FOR TYPEOUT
2032 012526 104402      TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2033 012530 104401      TYPE
2034 012532 026406      M2SP

```

H05

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 44  
DZICAC.NE2 21-MAR-77 15:28 INITIALIZE THE COMMON TAGS

2035	012534	013702	013130		MOV	RLOW,R2		
2036	012540	042702	170000		BIC	#170000,R2		
2037	012544				TYPOCT	R2		;TYPE LOWEST SAMPLE TAKEN.
2038	012544	010246			MOV	R2,-(SP)		;SAVE R2 FOR TYPEOUT
2039	012546	104402			TYPOC			;GO TYPE--OCTAL ASCII(ALL DIGITS)
2040	012550	104401			TYPE			
2041	012552	026406			M2SP			
2042								
2043								
2044								
2045								
2046								
2047								
2048								
2049	012554	013702	013126		MOV	AVTKN,R2		
2050	012560	042702	170000		BIC	#170000,R2		
2051	012564				TYPOCT	R2		;TYPE AVERAGE OF SAMPLES TAKEN.
2052	012564	010246			MOV	R2,-(SP)		;SAVE R2 FOR TYPEOUT
2053	012566	104402			TYPOC			;GO TYPE--OCTAL ASCII(ALL DIGITS)
2054	012570	104401			TYPE			
2055	012572	026406			M2SP			
2056	012574	013702	013132		MOV	RHIGH,R2		
2057	012600	042702	170000		BIC	#170000,R2		
2058	012604				TYPOCT	R2		;TYPE HIGHEST SAMPLE TAKEN.
2059	012604	010246			MOV	R2,-(SP)		;SAVE R2 FOR TYPEOUT
2060	012606	104402			TYPOC			;GO TYPE--OCTAL ASCII(ALL DIGITS)
2061	012610	104401			TYPE			;TYPE SECOND HEADER.
2062	012612	026550			MREPT2			
2063	012614	013737	013126	013134	MOV	AVTKN,RTEMP		;GET AV.
2064	012622	162737	000005	013134	SUB	#5,RTEMP		;MAKE -5 POINT TO GET LOW POINT.
2065	012630	012737	000400	013116	MOV	#256.,SAMCNT		;SET # OF SAMPLES.
2066	012636	013701	013074		MOV	CHAN1,R1		
2067	012642	062701	027240		ADD	#BUFFER,R1		
2068	012646	005002			CLR	R2		
2069	012650	022137	013134	2\$:	CMP	(1)+,RTEMP		
2070	012654	002001			BGE	3\$		
2071	012656	005202			INC	R2		
2072	012660	063701	013120	3\$:	ADD	SAMOFF,R1		
2073	012664	005337	013116		DEC	SAMCNT		
2074	012670	001367			BNE	2\$		
2075	012672	004737	013036		GOSUB	AVTYP		;TYPE RLOW # OF SAMPLES.
2076	012676	012703	177765		MOV	#-11.,R3		;SET TO DO 11 TIMES.
2077	012702	012737	000400	013116	4\$:	MOV	#256.,SAMCNT	;SET SAMPLE COUNT AT 256.
2078	012710	005002			CLR	R2		
2079	012712	013701	013074		MOV	CHAN1,R1		;GET BUFFER POINTER.
2080	012716	062701	027240		ADD	#BUFFER,R1		
2081	012722	022137	013134	5\$:	CMP	(1)+,RTEMP		;SAMPLE=COUNT POINTER?
2082	012726	001001			BNE	6\$		
2083	012730	005202			INC	R2		
2084	012732	063701	013120	6\$:	ADD	SAMOFF,R1		
2085	012736	005337	013116		DEC	SAMCNT		;CHECKED ALL SAMPLES?
2086	012742	001367			BNE	5\$		
2087	012744	004737	013036		GOSUB	AVTYP		;TYPE # OF SAMPLES AT THIS POINT.
2088	012750	005237	013134		INC	RTEMP		;MOVE TO NEXT POINT.
2089	012754	005203			INC	R3		;DONE ALL POINTS?
2090	012756	001351			BNE	4\$		;NO-THEN LOOP

```

2091 012760 005002          CLR      R2          ;FIND ALL OVERSCALE POINTS.
2092 012762 012737 000400 013116  MOV      #256.,SAMCNT
2093 012770 013701 013074      MOV      CHAN1,R1
2094 012774 062701 027240      ADD      #BUFFER,R1
2095 013000 022137 013134      7S:    CMP      (1)+,RTEMP
2096 013004 003401          BLE
2097 013006 005202          INC      R2
2098 013010 063701 013120      8S:    ADD      SAMOFF,R1
2099 013014 005337 013116      DEC      SAMCNT
2100 013020 001367          BNE      7S
2101 013022 004737 013036      GOSUB   ,AVTYP
2102 013026 104401          TYPE
2103 013030 001163          SCRLF
2104 013032 000137 012370      JMP      AVERRN
2105 013036 010246      AVTYP: MOV      R2,-(SP)    ;PUT # ON STACK
2106 013040 004737 016440      GOSUB   ,SSB2D      ;GOTO OCTAL-BCD ROUTINE.
2107 013044 062716 000007      ADD      #7,(SP)    ;GET RID OF 1ST.2 DIGITS
2108 013050 012637 013056      MOV      (SP)+,1S  ;TYPE STRING.
2109 013054 104401          TYPE
2110 013056 000000      1S:    0
2111 013060 104401          TYPE          ;TYPE 2 SPACES.
2112 013062 026406          M2SP
2113 013064 000207          RETURN
2114
2115
2116          ;*
2117          ;*POINTERS USED BY TEST 7 REPEATIBILITY TEST
2118          ;*
2119 013066 000000      CHAN7:0
2120 013070 000000      NA07:0
2121 013072 000000      NA17:0
2122 013074 000000      CHAN1: 0
2123 013076 000000      CHAN:  0
2124 013100 000000      GAIN:  0
2125 013102 000000      ADWD:  0
2126 013104 000000      CHANS: 0
2127 013106 000000      CHANF: 0
2128 013110 000000      CHANSR:0
2129 013112 000000      CHANFR:0
2130 013114 000000      CHANNO:0
2131 013116 000000      SAMCNT:0
2132 013120 000000      SAMOFF:0
2133 013122 000000      TOLER: 0
2134 013124 000000      AVEXP: 0
2135 013126 000000      AVTKN: 0
2136 013130 000000      RLOW:  0
2137 013132 000000      RHIGH: 0
2138 013134 000000      RTEMP: 0
2139 013136 000000      RTEMP1:0
2140 013140 000000      REPMAN:0
2141 013142 000000
2142 013144 000000
2143 013146 000000
2144 013150 000 000 000 0
2145 013152 000000          0
2146

```

```

;LEFT JUSTIFIED CURRENT CHANNELL
;CURRENT CHANNELL
;CURRENT GAIN
;WORD SENT TO A/D
;STARTING CHANNEL
;LAST CHANNEL

;SAMPLE COUNT
;SAMPLE OFFSET
;TOLERANCE BEFORE ERROR IS REPORTED
;EXPECTED AVERAGE
;AVERAGE OF SAMPLES TAKEN
;LOWEST SAMPLE TAKEN
;HIGHEST SAMPLE TAKEN

;SECTION RESERVED FOR
;MANUAL REPEATIBILITY
;WHERE OPERATOR ENTERS
;IN INFORMATION

```

```

2147
2148
2149
2150
2151 013154 071000
2152 013156 060200
2153 013160 050100
2154 013162 040050
2155 013164 030020
2156 013166 020010
2157 013170 010002
2158 013172 000001
2159 013174 000000
2160
2161
2162 013176 000000
2163 013200 000000
2164 013202 000000
2165 013204 000000
2166 013206 000000
2167 013210 000000
2168 013212 000000
2169 013214 000000
2170 013216 000000
2171 013220 000000
2172 013222 000000
2173 013224 000000
2174 013226 000000
2175 013230 000000
2176 013232 000000
2177 013234 000000
2178
2179 013236 000000
2180 013240 000000
2181 013242 000000
2182 013244 000000
2183 013246 000000
2184 013250 000000
2185 013252 000000
2186 013254 000000
2187 013256 000000
2188 013260 000000
2189 013262 000000
2190 013264 000000
2191 013266 000000
2192 013270 000000
2193 013272 000000
2194 013274 000000
2195
2196 013276 000000
2197 013300 000000
2198 013302 000000
2199 013304 000000
2200 013306 000000
2201 013310 000000
2202 013312 000000

```

.\*  
.\* GAIN LIST  
.\*

```

GLIST: 071000      :GAIN OF 1000.
        060200      :GAIN OF 200.
        050100      :GAIN OF 100.
        040050      :GAIN OF 50.
        030020      :GAIN OF 20.
        020010      :GAIN OF 10.
        010002      :GAIN OF 2.
        000001      :GAIN OF 1.
        000000      :ILLEGAL GAIN.

```

.\*INPUT MODULE ADDRESSES

```

INADR: 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0

```

.\*OUTPUT MODULE ADDRESSES

```

OUTADR: 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0

```

.\*COUNTER MODULE ADDRESSES

```

CNTADR: 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0

```

K05

```
2203 013314 000000 .WORD 0
2204 013316 000000 .WORD 0
2205 013320 000000 .WORD 0
2206 013322 000000 .WORD 0
2207 013324 000000 .WORD 0
2208 013326 000000 .WORD 0
2209 013330 000000 .WORD 0
2210 013332 000000 .WORD 0
2211 013334 000000 .WORD 0
2212 .;* DAC MODULE ADDRESS
2213 013336 000000 DACADR: 0
2214 .;* ADOS MODULE ADDRESS
2215 013340 000000 ADADR: 0
2216 013342 000000 BIPOL: 0 ;0=BIPOLAR-1=UNI
2217 .;* TEMP STORAGE OF INPUT MODULE DATA
2218 013344 000000 OUTS: 0
2219 013346 000000 .WORD 0
2220 013350 000000 .WORD 0
2221 013352 000000 .WORD 0
2222 013354 000000 .WORD 0
2223 013356 000000 .WORD 0
2224 013360 000000 .WORD 0
2225 013362 000000 .WORD 0
2226 013364 000000 .WORD 0
2227 013366 000000 .WORD 0
2228 013370 000000 .WORD 0
2229 013372 000000 .WORD 0
2230 013374 000000 .WORD 0
2231 013376 000000 .WORD 0
2232 013400 000000 .WORD 0
2233 013402 000000 .WORD 0
2234 013404 000000 OUTSE: 0
2235
2236 .;*
2237 .;*ROUTINE TO INPUT INPUT MODULE ADDRESS.
2238 .;*
2239
2240 013406 005737 001652 RINA: TST EXPERT
2241 013412 001002 BNE .+6
2242 013414 104401 TYPE ;ASK OPERATOR "INPUT MODULE ADDR(S)?"
2243 013416 021263 MIPA
2244 013420 012737 013176 013442 MOV #INADR, RINA1
2245 013426 012737 000020 013444 MOV #16., RINA1+2
2246 013434 000401 BR .+4
2247 013436 000763 BR RINA
2248 013440 104414 INOCT ;CALL TO INPUT OCTAL ROUTINE
2249 013442 013176 RINA1: INADR ;ADDR TO STORE INPUT ADDRESS AT (VARIES).
2250 013444 000020 16. ;NUMBER OF ADDRESS ALLOWED.
2251 013446 005737 015046 TST CHAR
2252 013452 001411 BEQ 2$
2253 013454 012700 013176 MOV #INADR, RD ;NOW FORM REAL ICS ADDRESS.
2254 013460 053710 001612 1$: BIS ICSMOD, (0) ;NORMAL-ICS MOD=171000 UNLESS PATCHED
2255 013464 004737 015510 JSR PC,CKADR
2256 013470 020037 013442 CMP RD,RINA1 ;DONE ALL ADDRS?
2257 013474 101771 BLOS 1$ ;IF NOT CONTINUE.
2258 013476 000002 2$: EXIT ;IF SO-EXIT
```

```

2259
2260
2261
2262
2263
2264 013500 005737 001652 ROUTA: TST EXPERT
2265 013504 001002 BNE .+6
2266 013506 104401 TYPE ;ASK OPERATOR "OUTPUT MODULE ADDRESSES?"
2267 013510 021321 MOPA
2268 013512 012737 013236 013534 MOV #OUTADR, ROUTA1
2269 013520 012737 000020 013536 MOV #16., ROUTA1+2
2270 013526 000401 BR .+4
2271 013530 000763 BR ROUTA
2272 013532 104414 INOCT ;CALL TO INPUT OCTAL ROUTINE
2273 013534 013236 ROUTA1: OUTADR ;ADDR TO STORE OUTPUT ADDRS.
2274 013536 000020 16.
2275 013540 005737 015046 TST CHAR
2276 013544 001411 BEQ 25
2277 013546 012700 013236 MOV #OUTADR, RO ;
2278 013552 053710 001612 15: BIS ICSMOD, (0)
2279 013556 004737 015510 JSR PC,CKADR
2280 013562 020037 013534 CMP RO,ROUTA1
2281 013566 101771 BLOS 15
2282 013570 000002 25: EXIT
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304

```

```

;*
;*ROUTINE TO INPUT COUNTER MODULE ADDR
;*

```

```

RCNTA: TST EXPERT
BNE .+6
TYPE ;ASK OPERATOR FOR COUNTER MODULE
MCNT ;ADDRS.
MOV #CNTADR, RCNTA1
MOV #16., RCNTA1+2
BR .+4
BR RCNTA
RCNTA1: INOCT ;GET THEM
16.
MOV #CNTADR, RO
TST CHAR
BEQ 25
15: BIS ICSMOD, (0)
JSR PC,CKADR
CMP RO,RCNTA1

```



```

2305 013660 003771 BLE IS
2306
2307 013662 000002 2S: EXIT
2308
2309
2310
2311
2312 013664 005737 001652 RDACA: TST EXPERT
2313 013670 001002 BNE .+6
2314 013672 104401 TYPE
2315 013674 022021 MDAC
2316 013676 012737 013336 013720 MOV #DACADR, RDAC1
2317 013704 012737 000001 013722 MOV #1, RDAC1+2
2318 013712 000401 BR .+4
2319 013714 000763 BR RDACA
2320 013716 104414
2321 013720 013336 RDAC1: INOCT
2322 013722 000001 DACADR
2323 013724 005737 015046 TST CHAR
2324 013730 001407 BEQ IS
2325 013732 053737 001612 013336 BIS ICSMOD, DACADR
2326 013740 012700 013336 MOV #DACADR, RO
2327 013744 004737 015510 JSR PC, CKADR
2328 013750 000002 1S: EXIT
2329
2330
2331
2332 013752 005737 001652 RADA: TST EXPERT
2333 013756 001002 BNE .+6
2334 013760 104401 TYPE
2335 013762 022044 MADU
2336 013764 012737 013340 014006 MOV #ADADR, RADA1
2337 013772 012737 000001 014010 MOV #1, RADA1+2
2338 014000 000401 BR .+4
2339 014002 000763 BR RADA
2340 014004 104414
2341 014006 013340 RADA1: INOCT
2342 014010 000001 ADADR
2343 014012 005737 015046 TST CHAR
2344 014016 001407 BEQ IS
2345 014020 053737 001612 013340 BIS ICSMOD, ADADR
2346 014026 012700 013340 MOV #ADADR, RO
2347 014032 004737 015510 JSR PC, CKADR
2348 014036 000002 1S: EXIT
2349
2350
2351
2352
2353
2354
2355
2356 014040 104401 INOCTR: TYPE ;TYPE A "?".
2357 014042 022366 MQ
2358 014044 017601 000000 MOV #2(6), R1 ;PICK UP STORAGE ADDRESS.
2359 014050 062716 000002 ADD #2, (6) ;PICK UP # OF WORDS THAT-
2360 014054 017637 000000 015060 MOV #2(6), NINC1 ;IS THE MAX # TO BE INPUTED.
    
```

2361	014062	162716	000002			SUB	#2	(6)		:CLEAR COLON TYPED FLAG.
2362	014066	005037	001630			CLR	INCFLG			:CLEAR ANY CHAR. TYPED FLAG.
2363	014072	005037	013134			CLR	RTEMP			:POINT TO TEMP STORAGE AREA.
2364	014076	012700	013344			MOV	#OUTS,RO			:CLEAR THE AREA.
2365	014102	005020			20\$:	CLR	(0)+			
2366	014104	020027	013404			CMP	RO,#OUTSE			
2367	014110	003774				BLE	20\$			
2368	014112	012700	013344			MOV	#OUTS,RO			:POINT TO AREA AGAIN.
2369	014116	005037	015056			CLR	NINC			:CLEAR CHAR. COUNT
2370	014122	005037	015050		1\$:	CLR	NIN			:CLEAR NUMBER TO BE FORMED
2371	014126	005037	015054			CLR	CHARC			:INPUTED CHAR. COUNT
2372	014132	105777	165006		2\$:	TSTB	#STKS			:KEY TYPED?
2373	014136	100375				BPL	2\$			:NO - THEN WAIT.
2374	014140	117737	165002	015046		MOV	#STKB, CHAR			:YES - READ CHAR.
2375	014146	042737	177600	015046		BIC	#177600, CHAR			:STRIP CHAR PARITY BIT - IF ANY.
2376	014154	023727	015046	000177		CMP	CHAR,#177			:WAS IT A RUBOUT?
2377	014162	001002				BNE	.+6			:NO - CONTINUE
2378	014164	000137	015006			JMP	10\$			:YES - TYPE "?" - REINITIALIZE.
2379	014170	105777	164754			TSTB	#STPS			:PRINTER BUSY?
2380	014174	100375				BPL	.-4			:YES - THEN WAIT TILL NOT
2381	014176	013777	015046	164746		MOV	CHAR, #STPB			:NO - ECHO CHAR.
2382	014204	023727	015046	000015		CMP	CHAR, #15			:CHAR. = <CR>?
2383	014212	001514				BEQ	3\$			:YES - THEN TERMINATE INPUT.
2384	014214	023727	015046	000003		CMP	CHAR, #3			:WAS IT A IC?
2385	014222	001002				BNE	.+6			
2386	014224	000177	165450			JMP	#CTLLOC			:IF YES GOTO MONITR.
2387	014230	023727	015046	000014		CMP	CHAR, #14			:CHAR = "IL"?
2388	014236	001002				BNE	.+6			
2389	014240	000137	015112			JMP	LPSET			:YES - THEN SET LINEPRINTER MODE.
2390	014244	122737	000012	015046		CMPB	#12, CHAR			:CHAR = "IJ"?
2391	014252	001002				BNE	.+6			
2392	014254	000137	015134			JMP	CONNTR			:YES - THEN SET JOINED MODE.
2393	014260	122737	000004	015046		CMPB	#4, CHAR			:CHAR = "ID"?
2394	014266	001002				BNE	.+6			
2395	014270	000137	015260			JMP	RDELA3			:YES - GET SECONDARY DELAY.
2396	014274	123727	015046	000005		CMPB	CHAR, #5			:CHAR = "IE"?
2397	014302	001002				BNE	.+6			
2398	014304	000137	015062			JMP	EXSET			:YES - SET EXPERT MODE.
2399	014310	123727	015046	000016		CMPB	CHAR, #16			:CHAR = "IN"?
2400	014316	001002				BNE	.+6			
2401	014320	000137	015076			JMP	NOSET			:YES - SET NOVICE MODE
2402	014324	023727	015046	000072		CMP	CHAR, #':			:CHAR = COLON?
2403	014332	001522				BEQ	4\$			:YES SET FOR "THROUGH" ENTRY.
2404	014334	023727	015046	000054		CMP	CHAR, #',			:CHAR = A COMMA?
2405	014342	001575				BEQ	7\$			
2406	014344	023727	015046	000060		CMP	CHAR, #60			:CHAR TYPED ? ASCIZ?
2407	014352	002002				BGE	.+6			
2408	014354	000137	015016			JMP	11\$			:NO - REPORT ERROR.
2409										
2410	014360	023727	015046	000067		CMP	CHAR, #67			:CHAR TYPED <ASCIZ ?>
2411	014366	003402				BLE	.+6			
2412	014370	000137	015016			JMP	11\$			:NO - REPORT ERROR.
2413	014374	005237	013134			INC	RTEMP			:YES - INCREMENT CHAR. COUNT.
2414	014400	006137	015050			ROL	NIN			:LEFT JUSTIFY CURRENT NUMBER.
2415	014404	006137	015050			ROL	NIN			
2416	014410	006137	015050			ROL	NIN			

2417	014414	005237	015054			INC	CHARC			;	INCREMENT CHAR COUNT.
2418	014420	042737	000007	015050		BIC	87,	NIN		;	STRIP LOWER NUMBER.
2419	014426	042737	000260	015046		BIC	8260,	CHAR		;	STRIP CHAR. INPUTED.
2420	014434	053737	015046	015050		BIS	CHAR,	NIN		;	ADD TO CURRENT NUMBER.
2421	014442	000633				BR	2S			;	LOOP.
2422	014444	005737	001630		3S:	TST	INCFLG			;	COME HERE ON <CR>: COLON FLAG SET?
2423	014450	001066				BNE	5S			;	YES TAKE CARE OF IT.
2424	014452	023737	015056	015060		CMF	NINC,	NINC1		;	DID WE ASSEMBLE MORE NUMBERS THAN ALLOWED?
2425	014460	003146				BGT	8S			;	IF SO - REPORT ERROR.
2426	014462	005737	013134			TST	RTEMP			;	ANY CHAR. TYPED?
2427	014466	001005				BNE	9S			;	YES - FUDGE RETURN.
2428	014470	012737	000017	015056		MOV	815.,	NINC			
2429	014476	005037	015046			CLR	CHAR				
2430	014502	013720	015050		9S:	MOV	NIN,	(0)+		;	STORE NUMBER.
2431	014506	013737	015060	013134		MOV	NINC1,	RTEMP		;	FIX NUMBER INPUTED COUNT.
2432	014514	163737	015056	015060		SUB	NINC	NINC1		;	FIX EXACT COUNT.
2433	014522	005337	015060			DEC	NINC1			;	= COUNT -1
2434	014526	012700	013344			MOV	80UTS,RO			;	SET TO READ BACK NUMBERS.
2435	014532	012021			21S:	MOV	(0)+,(1)+			;	STORE IN CORRECT AREA.
2436	014534	005337	015056			DEC	NINC			;	DONE ALL NUMBERS?
2437	014540	001002				BNE	22S			;	NO - SKIP NEXT INSTR.
2438	014542	010176	000000			MOV	R1,2(6)			;	FIX CALL ON ADDR. OF NUMBERS TO INPUT.
2439	014546	005337	013134		22S:	DEC	RTEMP			;	DONE ALL NUMBERS.
2440	014552	001367				BNE	21S			;	NO - LOOP.
2441	014554	062716	000002			ADD	82,(6)			;	YES - UPDATE SP.
2442	014560	013776	015060	000000		MOV	NINC1,2(6)			;	STORE ON CALL # OF NUMBERS TO INPUT.
2443	014566	062716	000002			ADD	82,(6)			;	FIX SP.
2444	014572	104401	001163			TYPE,	SCALF				
2445	014576	000002				EXIT				;	EXIT.
2446											
2447	014600	013737	015050	015052	4S:	MOV	NIN,	NINI		;	ENTER HERE WHEN COLON TYPED. STORE CURRENT #.
2448	014606	005737	001630			TST	INCFLG			;	CHECK COLON FLAG.
2449	014612	001105				BNE	12S			;	IF SET UNKNOWN INPUT (2 COLONS!)
2450	014614	012737	000001	001630		MOV	81,INCFLG			;	SET COLON FLAG.
2451	014622	000137	014122			JMP	1S			;	EXIT TO LOOP
2452	014626	162737	000002	015052	5S:	SUB	82,	NINI		;	COME HERE ON COLON FLAG SET + ANOTHER # FORMED.
2453	014634	062737	000002	015052	6S:	ADD	82,	NINI		;	UPDATE FORMER NUMBER.
2454	014642	023737	015056	015060		CMF	NINC,	NINC1		;	NUMBERS OVERFLOW BUFFER AREA?
2455	014650	003052				BGT	8S			;	YES - REPORT ERROR.
2456	014652	013720	015052			MOV	NIN1,	(0)+		;	NO - STORE NEW NUMBER.
2457	014656	005237	015056			INC	NINC			;	UPDATE NUMBER COUNT.
2458	014662	023737	015050	015052		CMF	NIN,	NINI		;	UPDATED NUMBER = CURRENT NUMBER?
2459	014670	001361				BNE	6S			;	NO - LOOP.
2460	014672	005037	001630			CLR	INCFLG			;	YES - CLEAR COLON FLAG
2461	014676	005337	015056			DEC	NINC				
2462	014702	162700	000002			SUB	82,RO			;	FIX POINTER (STORAGE).
2463	014706	022737	000015	015046		CMF	81S,	CHAR		;	LAST CHAR TYPED A <CR>?
2464	014714	001653				BEQ	3S			;	YES - EXIT.
2465	014716	022737	000054	015046		CMF	8',,CHAR			;	LAST CHAR TYPED A COMMA?
2466	014724	001404				BEQ	7S			;	YES - TAKE CARE OF IT.
2467											
2468	014726	062700	000002			ADD	82,	RO		;	FIX STORAGE POINTER.
2469	014732	000137	014122			JMP	1S			;	LOOP.
2470	014736	005737	001630		7S:	TST	INCFLG			;	ENTER HERE ON COMMA TYPED. WAS COLON-
2471	014742	001331				BNE	5S			;	PREVIOUSLY TYPED? IF SO TAKE CARE OF IT.
2472	014744	023737	015056	015060		CMF	NINC,	NINC1		;	BUFFER OVERFLOW?

```

2473 014752 003011
2474 014754 005737 015054
2475 014760 001422
2476 014762 013720 015050
2477 014766 005237 015056
2478 014772 000137 014122
2479
2480 014776 104401
2481 015000 022070
2482 015002 000137 014040
2483
2484 015006 104401
2485 015010 022360
2486 015012 000137 014040
2487
2488 015016 104401
2489 015020 026126
2490 015022 000137 014132
2491 015026 104401
2492 015030 026207
2493 015032 000137 014040
2494
2495
2496
2497
2498
2499 015036 000240
2500 015040 162716 000004
2501 015044 000002
2502
2503 015046 000000
2504 015050 000000
2505 015052 000000
2506 015054 000000
2507
2508 015056 000000
2509 015060 000000
2510 015062 005237 001652
2511 015066 104401
2512 015070 021540
2513 015072 000137 015036
2514
2515 015076 005037 001652
2516 015102 104401
2517 015104 021570
2518 015106 000137 015036
2519
2520
2521
2522 015112 005737 001656
2523 015116 001404
2524 015120 104401
2525 015122 021621
2526 015124 005237 001660
2527 015130 000137 015036
2528

```

```

      BGT      BS
      TST      CHARC
      BEQ      12S
      MOV      NIN, (0)+
      INC      NINC
      JMP      IS
BS:    TYPE
      MABOV
      JMP      INOCTR
10S:   TYPE
      MAMARK
      JMP      INOCTR
11S:   TYPE
      MININ
      JMP      2S
12S:   TYPE
      MINKN
      JMP      INOCTR
;
; RETURN FOR ↑D, ↑J, ↑L, ↑N, ↑E
;
RASK:  NOP
      SUB      #4, (6)
      EXIT
CHAR:  0
NIN:   0
NINI:  0
CHARC: 0
NINC:  0
NINC1: 0
EXSET: INC      EXPERT
      TYPE
      MEXEN
      JMP      RASK
NOSET: CLR      EXPERT
      TYPE
      MNOEN
      JMP      RASK
LPSET: TST      LPAV
      BEQ      IS
      TYPE
      MLEN
      INC      LINEPR
IS:    JMP      RASK

```

```

; IF YES - REPORT ERROR.
; WHY CHAR. TYPED?
; IF NO - REPORT ERROR.
; OTHERWISE STORE CURRENT NUMBER.
; INCR. NUMBER COUNT.
; LOCP.

; "ADDR. BUFFER OVERFLOW ERROR
; RETYPE"

; TYPE A "?"

; TYPE "LAST CHAR TYPED NOT AN OCTAL DIGIT"

; CHARACTER INPUTTED
; ASSEMBLED NUMBER
; ASSEMBLED NUMBER, 1ST IF MULTIPLE
; USED TO COUNT CHARS. IN NUMBER.

; NUMBER OF ADDRESSED ASSEMBLED.
; MAXIMUM NUMBER TO BE ASSEMBLED.
; ENTER HER ON "↑E".
; SET EXPERT MODE.

; ENTER HERE ON "↑N".
; SET NOIVE MODE

; ENTER HERE ON "↑L".
; ANY LP AVAILABLE - IF NOT EXIT.
; TYPE "MAKE LP READY"

```

01  
02  
03  
04  
05  
06  
07  
08  
09  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84

015134 005137 001654  
015140 001402  
015142 104401  
015144 021707  
015146 000137 015036  
  
015152 005737 001652  
015156 001002  
015160 104401  
015162 022234  
015164 005037 001570  
015170 013737 001564 001566  
015176 104401  
015200 022366  
015202 104411  
015204 012637 013134  
015210 001417  
015212 005037 001570  
015216 005037 001566  
015222 063737 001576 001566 1S:  
015230 005537 001570  
015234 005737 001570  
015240 100404  
015242 005337 013134  
015246 001365  
015250 000002 2S:  
  
015252 104401 3S:  
015254 022274  
015256 000735

```

;*
;*ROUTINE TO CONNECT I/O MODULES (SOFT)
;*
CONNTR: COM      CONNT
        BEQ      .+6
        TYPE
        MAC
        JMP      RASK

;*
;*ROUTINE TO ASK FOR AND SET DELAY TIME
;* 0 DELAY OR CR BY OPERATOR - USER DEFAULT OF 3 MILLISEC.
RDELA: TST      EXPERT
        BNE      .+6
        TYPE
;ASK OPERATOR -
; "DELAY TIME IN MS (DEFAULT=3MS)?"
        MDEL
        CLR      FREQ1
        MOV      FR3, FREQ
        TYPE
        M0
        RDEEC
        MOV      (6)+, RTEMP ;GET NUMBER.
        BEQ      2S ;IF ZERRO, USE DEFAULT
        CLR      FREQ1
        CLR      FREQ
        ADD      FR1, FREQ ;SET FREQUENCY (OR DELAY).
        ADC      FREQ1
        TST      FREQ1
        BMI      3S
        DEC      RTEMP
        BNE      1S
        EXIT
        TYPE ;TYPE" TIME TOO LONG".
        MTL
        BR      RDELA

;*
;*ROUTINE TO FORM SECONDARY DELAY
;*
RDELA3: TYPE, MDEL1
RDELA4: TYPE
        MDEL2
        RDEEC
        MOV      (6)+, RTEMP
        BEQ      2S
        CLR      FREQ2
        ADD      FR1, FREQ
        ADC      FREQ3
        TST      FREQ3
        BMI      3S
        DEC      RTEMP
        BNE      1S
        JMP      RASK

```

```

2585 015336 104401
2586 015340 022274
2587 015342 000750
2588
2589
2590
2591
2592
2593
2594
2595
2596 015344 005737 001652
2597 015350 001002
2598 015352 104401
2599 015354 024700
2600 015356 104401
2601 015360 022366
2602 015362 005037 015046
2603 015366 105777 163552
2604 015372 100375
2605 015374 105777 163550
2606 015400 100375
2607 015402 017737 163540 013134
2608 015410 013777 013134 163534
2609 015416 042737 000240 013134
2610 015424 023727 013134 000003
2611 015432 001002
2612 015434 000137 003374
2613 015440 023727 013134 000015
2614 015446 001404
2615 015450 013737 013134 015046
2616 015456 000743
2617 015460 104401 001163
2618 015464 023727 015046 000125
2619 015472 001003
2620 015474 005237 013342
2621 015500 000002
2622 015502 005037 013342
2623 015506 000002
2624
2625
2626
2627
2628
2629
2630 015510 013737 000004 015610
2631 015516 012737 015560 000004
2632 015524 005770 000000
2633 015530 000240
2634
2635
2636
2637
2638
2639 015532 013737 015610 000004
2640 015540 023710 001702

```

```

3S:  TYPE
      MTTL
      BR      RDELAY

      ;*
      ;*ROUTINE TO SET UNI OR BIPOLAR A/D
      ;*CALL = QUBR
      ;*

RQUBR: TST      EXPERT
        BNE     .+6
        TYPE    ;UNI OR BI-POLAR?
        MOURB
        TYPE
        MO
        CLR     CHAR
        TSTB   2STKS ;WAIT FOR RESPONSE
        BPL    1S
        TSTB   2STPS ;ECHO.
        BPL    2S
        MOV     2STKB,RTEMP
        MOV     RTEMP,2STPB
        BIC     240,RTEMP
        CMP     RTEMP,#3 ;IC TYPED?
        BNE     .+6
        JMP     START1
        CMP     RTEMP,#15 ;CR TYPED?
        BEQ     3S
        MOV     RTEMP,CHAR
        BR      1S
        TYPE,   SCRLF
        CMP     CHAR,#'U
        BNE     4S
        INC     BIPOL
        EXIT
4S:  CLR     BIPOL
      EXIT

      ;*
      ;*ROUTINE TO CHECK THAT AN ADDRESS TYPED IS A LEGAL ICS-ADDR.
      ;*TYPE ERROR IF NOT - ENTER WITH ADDR IN (0)
      ;*CALL = JSR PC,CKADR
      ;*

CKADR: MOV     2#4,3S ;STORE LOCATION 4.
        MOV     2#5,2#4 ;SET TIME-OUT LOCATION FOR TRAP IF ANY.
        TST    2(0) ;TEST MOD. ADDR. TYPED.
        NOP

      ;**
      ;** MODULE ADDRESS DOES EXIST BUT MAY NOT BE ASSOCIATED WITH THIS
      ;** FILE BOX. THEREFORE, CHECK ADDRESS FOR WITHIN FILE BOX LIMITS
      ;**

        MOV     3S,2#4 ;RESTORE TIME OUT VECTOR
        CMP     ICSLMT,(R0)

```

```

2641 015544 003022          BGT      4$
2642 015546 023710 001676  CMP      ICSHGH,(R0)
2643 015552 003417          BLE      4$
2644 015554 005720          1$:     TST      (0)+
2645 015556 000207          RTS      PC          ;EXIT - SUBROUTINE.
2646 015560 104401 022517  2$:     TYPE   MNRFN    ;TYPE ERROR MESSAGE.
2647 015564          TYPOCT  (0)    ;TYPE ADDR.
2648 015564 011046          MOV      (0),-(SP)  ;SAVE (0) FOR TYPEOUT
2649 015566 104402          TYPOC   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2650 015570 062706 000006  ADD      #6,R6
2651 015574 162716 000002  SUB      #2,(R6)    ;FIX RETURN ADDRESS ON STACK
2652 015600 013737 015610 000004  MOV      3$,#4     ;RESTORE VECTOR ADDRESS
2653 015606 000002          EXIT          ;SO THAT WE RETURN AND REASK
2654          ;QUESTION ON MODULE ADDR.
2655 015610 000000          3$:     .WORD   0    ;TEMP STORAGE OF LOC. 4.
2656
2657
2658 015612 104401 026763  4$:     TYPE   NRANG1   ;NOT WITHIN FILE
2659 015616          TYPOCT  (0)
2660 015616 011046          MOV      (0),-(SP)  ;SAVE (0) FOR TYPEOUT
2661 015620 104402          TYPOC   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
2662 015622 104401 026776  TYPE   NRANG2
2663 015626 005726          TST      (R6)+
2664 015630 162716 000002  SUB      #2,(R6)    ;POP STACK PAST JSR RETURN
2665 015634 000002          EXIT          ;BACK POINTER TO RE-ASK QUESTION
2666          ;RETURN
2667
2668          ;*
2669          ;*ROUTINE USED OFR GENERATION A DELAY
2670          ;*CALL=DELAT
2671          ;*
2672
2673 015636 013737 001566 013134  RDELAY: MOV      FREQ,  RTEMP ;GET DELAY TIME
2674 015644 001411          BEQ      2$
2675 015646 013737 001570 015672  MOV      FREQ1, RTEMP2
2676 015654 005337 013134          1$:     DEC      RTEMP    ;DELAY
2677 015660 001375          BNE     1$
2678 015662 005337 015672  DEC      RTEMP2
2679 015666 100372          BPL     1$
2680 015670 000002          2$:     EXIT          ;RETURN
2681 015672 000000          RTEMP2: 0
2682
2683          ;*
2684          ;*ROUTINE TO GENERATE SECONDARY DELAY
2685          ;*CALL =DELAY2
2686          ;*
2687 015674 013737 001572 013134  RDELA2: MOV      FREQ2,RTEMP
2688 015702 001413          BEQ      2$
2689 015704 013737 001574 015672  MOV      FREQ3,RTEMP2
2690 015712 005037 177776          CLR     PS
2691 015716 005337 013134          1$:     DEC      RTEMP
2692 015722 001375          BNE     1$
2693 015724 005337 015672  DEC      RTEMP2
2694 015730 100372          BPL     1$
2695 015732 000002          2$:     EXIT
2696

```

```

2697
2698
2699
2700
2701
2702
2703
2704 015734 RINIT:
2705 015734 013700 001104 MOV SICNT,RO
2706 015740 000005 RESET
2707 015742 032777 000001 163170 BIT #1,DSWR
2708 015750 001402 BEQ 1$
2709 015752 000137 003374 JMP START1
2710
2711 015756 052777 000100 163160 1$: BIS #100, D$TKS
2712 015764 000002 EXIT
2713
2714
2715
2716
2717
2718
2719
2720
2721 015766 023727 001602 000012 RCPAT: CMP PATRNM,#12
2722 015774 001442 BEQ RCPATR
2723 015776 000241 CLC ;CLEAR C BIT
2724 016000 005237 001604 RCPATI: INC D$PATRN ;MODIFY PATTERN
2725 016004 103412 BCS 1$
2726 016006 022737 005137 016000 CMP #5137,RCPATI ;DOING A COMPLEMENT PATERN?
2727 016014 001406 BEQ 1$ ;IF SO DON'T ADD CARRY IF ANY!
2728 016016 063737 001606 001604 ADD PATRNC,PATRN
2729 016024 005037 001606 CLR PATRNC
2730 016030 000002 EXIT
2731 016032 023727 016000 006037 1$: CMP RCPATI,#6037
2732 016040 001004 BNE 2$
2733 016042 012737 100000 001606 MOV #100000,PATRNC
2734 016050 000002 EXIT
2735 016052 012737 000001 001606 2$: MOV #1,PATRNC
2736 016060 000002 EXIT
2737
2738 016062 005237 RCPATL: ;#MODIFIER LIST
2739 016064 005337 5237 ;#INCREMENTING PATTERN
2740 016066 005737 5337 ;#DECREMENTING PATTERN
2741 016070 006137 5737 ;#NO CHANGE OF PATTERN
2742 016072 006037 6137 ;#ROTATE LEFT PATTERN
2743 016074 006237 6037 ;#ROTATE RIGHT PATTERN
2744 016076 006237 6237 ;#RANDOM NUMBER GENERATOR
2745 016100 005137 6337 ;#ARITH. SHIFT LEFT PATTERN
2746
2747
2748
2749
2750
2751
2752 016102 RCPATR: ;#COMPLEMENT PATTERN

```

016102 5 6 7



```

2753 016102 004737 016336 JSR PC SRAND ;GENERATE A RANDOM NUMBER.
2754 016106 013737 016436 001604 MOV SLONUM,PATRN ;PUT NUMBER IN PATTERN.
2755 016114 000002 EXIT
2756
2757
2758
2759 ;*
2760 ;*PATTERN INPUTTER AND MODIFIER ROUTINE
2761 ;*FORM PATTERN MODIFIER, PATTERN.
2762 ;*CALL=PATAR
2763 016116 005737 001652 RPATA: TST EXPERT
2764 016122 001002 BNE .+6
2765 016124 104401 TYPE ;ASK OPERATOR-"PATTERN MODIFIER AND
2766 ;PATTERN?"
2767 016126 022316 MPPM ;SET TO INPUT PATTERN + MODIFIER
2768 016130 012737 001602 016162 MOV #PATRN, 15
2769 016136 012737 000002 016164 MOV #2, 15+2
2770 016144 005037 001602 CLR PATRN ;SET DEFAULTS: 0 PATTERN
2771 016150 005037 001604 CLR PATRN ;PATTERN MOD. = INC.
2772 016154 000401 BR .+4
2773 016156 000757 BR RPATA
2774 016160 104414 INOCT ;GET THEM.
2775 016162 001602 15: PATRN 2
2776 016164 000002
2777 016166 012737 016062 013134 MOV #RCPATL,RTEMP ;POINT TO BEGINNING OF MODIFIER LIST.
2778 016174 032737 177770 001602 BIT #177770,PATRN ;LEGAL PATTERN MODIFIER?
2779 016202 001403 BEQ 25 ;IF YES 25
2780 016204 104401 025730 TYPE, MIVP ;NO TYPE ERROR MESSAGE.
2781 016210 000742 BR RPATA ;REASK QUESTION.
2782 016212
2783 016212 006337 001602 25: ASL PATRN
2784 016216 042737 177761 001602 BIC #177761, PATRN ;FIX ADDITIVE.
2785 016224 063737 001602 013134 ADD PATRN, RTEMP ;POINT TO MODIFIER.
2786 016232 017737 174676 016000 MOV @RTEMP, RCPAT+12 ;CHANGE PATTERN MODIFIER ROUTINE.
2787 016240 000002 EXIT ;RETURN.
2788 016242 .SSAVE
2789 .SBTTL SAVE AND RESTORE R0-R5 ROUTINES
2790
2791 016242 STARS
2792 ;*****
2793 ;SAVE R0-R5
2794 ;CALL:
2795 ; SAVREG
2796 ;UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
2797 ;
2798 ;TOP---(+16)
2799 ; +2---(+18)
2800 ; +4---R5
2801 ; +6---R4
2802 ; +8---R3
2803 ;+10---R2
2804 ;+12---R1
2805 ;+14---R0
2806
2807 016242 $SAVREG:
2808 016242 PUSH <R0,R1,R2,R3,R4,R5>

```

```

2809 016242 010046      MOV      RO,-(SP)      ;; PUSH RO ON STACK
2810 016244 010146      MOV      R1,-(SP)      ;; PUSH R1 ON STACK
2811 016246 010246      MOV      R2,-(SP)      ;; PUSH R2 ON STACK
2812 016250 010346      MOV      R3,-(SP)      ;; PUSH R3 ON STACK
2813 016252 010446      MOV      R4,-(SP)      ;; PUSH R4 ON STACK
2814 016254 010546      MOV      R5,-(SP)      ;; PUSH R5 ON STACK
2815 016256 016646 000022  MOV      22(SP),-(SP)  ;; SAVE PS OF MAIN FLOW
2816 016262 016646 000022  MOV      22(SP),-(SP)  ;; SAVE PC OF MAIN FLOW
2817 016266 016646 000022  MOV      22(SP),-(SP)  ;; SAVE PS OF CALL
2818 016272 016646 000022  MOV      22(SP),-(SP)  ;; SAVE PC OF CALL
2819 016276 000002      RTI
2820
2821      ;#RESTORE RO-R5
2822      ;#CALL:
2823      ;#
2824      ;# RESREG
2825      ;#RESREG:
2825 016300 012666 000022  MOV      (SP)+,22(SP)  ;; RESTORE PC OF CALL
2826 016304 012666 000022  MOV      (SP)+,22(SP)  ;; RESTORE PS OF CALL
2827 016310 012666 000022  MOV      (SP)+,22(SP)  ;; RESTORE PC OF MAIN FLOW
2828 016314 012666 000022  MOV      (SP)+,22(SP)  ;; RESTORE PS OF MAIN FLOW
2829 016320      POP      <R5,R4,R3,R2,R1,RO>
2830 016320 012605      MOV      (SP)+,R5      ;; POP STACK INTO R5
2831 016322 012604      MOV      (SP)+,R4      ;; POP STACK INTO R4
2832 016324 012603      MOV      (SP)+,R3      ;; POP STACK INTO R3
2833 016326 012602      MOV      (SP)+,R2      ;; POP STACK INTO R2
2834 016330 012601      MOV      (SP)+,R1      ;; POP STACK INTO R1
2835 016332 012600      MOV      (SP)+,RO      ;; POP STACK INTO RO
2836 016334 000002      RTI
2837 016336
2838      .SRAND
2839      .SBTTL RANDOM NUMBER GENERATOR ROUTINE
2840
2841 016336
2842      STARS
2843      ;*****
2844      ;THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
2845      ;WITH A RANGE OF 0 TO 2(+33)-1.
2846      ;CALL:
2847      ;# JSR      PC,SRAND      ;; CALL THE ROUTINE
2848      ;# RETURN      ;; RETURN HERE THE RANDOM
2849      ;#           ;; NUMBER WILL BE IN
2850      ;#           ;; SHINUM,SLONUM
2851 016336 010046      SRAND:  PUSH      <RO,R1,R2>
2852 016340 010146      MOV      RO,-(SP)      ;; PUSH RO ON STACK
2853 016342 010246      MOV      R1,-(SP)      ;; PUSH R1 ON STACK
2854 016344 013700 016436      MOV      R2,-(SP)      ;; PUSH R2 ON STACK
2855 016350 013701 016434      MOV      SLONUM,RO      ;; SET RO WITH LOW
2856 016354 012702 177771      MOV      SHINUM,R1      ;; SET R1 WITH HIGH
2857 016360 006300      MOV      #7,R2          ;; SET SHIFT COUNT
2858 016362 006101      1S:    ASL      RO            ;; SHIFT RO LEFT AND
2859 016364 005202      ROL      R1            ;; ROTATE CARRY INTO R1 AND
2860 016366 001374      INC      R2            ;; CHECK FOR DONE
2861 016370 063700 016436      BNE      1S           ;; CONTINUE SHIFT LOOP
2862 016374 005501      ADD      SLONUM,RO      ;; ADD NUMBER TO MAKE X 129
2863 016376 063701 016434      ADC      R1            ;; PROPOGATE CARRY
2864 016402 062700 001057      ADD      SHINUM,R1      ;; ADD NUMBER TO MAKE X 129
2864      ADD      #1057,RO      ;; ADD LOW CONSTANT
    
```

```

2865 016406 005501          ADC      R1          ;; PROPOGATE CARRY
2866 016410 062701 047401  ADD      #47401,R1  ;; ADD HIGH CONSTANT
2867 016414 010037 016436  MOV      R0,SLONUM  ;; SAVE R0
2868 016420 010137 016434  MOV      R1,SHINUM  ;; SAVE R1
2869 016424          POP      <R2,R1,R0>
2870 016424 012602          MOV      (SP)+,R2   ;; POP STACK INTO R2
2871 016426 012601          MOV      (SP)+,R1   ;; POP STACK INTO R1
2872 016430 012600          MOV      (SP)+,R0   ;; POP STACK INTO R0
2873 016432 000207          RTS      PC          ;; RETURN
2874 016434 176543  SHINUM: .WORD 176543
2875 016436 123456  SLONUM: .WORD 123456
2876 016440          .SSB2D
2877          .SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
2878
2879 016440          STARS
2880          ;; *****
2881          ;; *THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
2882          ;; *UNSIGNED DECIMAL ASCIZ NUMBER.
2883          ;; *CALL
2884          ;; *   MOV      NUMBER,-(SP)      ;; PUT BINARY NUMBER ON THE STACK
2885          ;; *   JSR      PC,@#SSB2D      ;; CALL
2886          ;; *   RETURN                      ;; ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK
2887
2888
2889 016440 016637 000002 016470 SSB2D: MOV      2(SP),1S      ;; SAVE BINARY NUMBER
2890 016446 012746 016470          MOV      #1S,-(SP)  ;; SET POINTER
2891 016452 004737 016474          JSR      PC,@#SDB2D ;; CALL DOUBLE LENGTH CONVERT
2892 016456 062716 000005          ADD      #5,(SP)    ;; ONLY ALLOW FIVE CHARACTERS
2893 016462 012666 000002          MOV      (SP)+,2(SP) ;; PICKUP POINTER
2894 016466 000207          RTS      PC          ;; RETURN
2895 016470 000000 000000  1S:      .WORD 0,0
2896 016474          .SDB2D
2897          .SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
2898
2899 016474          STARS
2900          ;; *****
2901          ;; *THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
2902          ;; *DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
2903          ;; *POSITIVE.
2904          ;; *CALL
2905          ;; *   MOV      #PNTR,-(SP)      ;; POINTER TO LOW WORD OF BINARY NUMBER
2906          ;; *   JSR      PC,@#SDB2D      ;; CALL
2907          ;; *   RETURN                      ;; THE FIRST ADDRESS OF ASCIZ
2908          ;; *                                     ;; IS ON THE STACK
2909
2910
2911 016474 104412          SDB2D: SAVREG          ;; SAVE REGISTERS
2912 016476 016602 000302          MOV      2(SP),R2   ;; PICKUP THE DATA POINTER
2913 016502 012700 016654          MOV      #SDECVL,R0 ;; GET ADDRESS OF "SDECVL" STRING
2914 016506 010066 000002          MOV      R0,2(SP)   ;; PUT ADDRESS OF ASCIZ STRING ON STACK
2915 016512 012201          MOV      (R2)+,R1   ;; PICKUP THE BINARY NUMBER
2916 016514 012202          MOV      (R2)+,R2
2917 016516 012737 000012 016572          MOV      #10,4S     ;; SET UP TO DO 10 CONVERSIONS
2918 016524 012704 016604          MOV      #STNPNR,R4 ;; ADDRESS OF TEN POWER
2919 016530 012705 016606          MOV      #STNPNR+2,R5
2920 016534 005003          1S:      CLR      R3      ;; CLEAR PARTIAL

```

```

2921 016536 161401      2S:  SUB      (R4),R1      ;;SUBTRACT TEN POWER
2922 016540 005602      SBC      R2
2923 016542 161502      SUB      (R5),R2
2924 016544 002402      BLT     3S      ;;BR IF TEN POWER TO LARGE
2925 016546 005203      INC     R3      ;;ADD 1 TO PARTIAL
2926 016550 000772      BR      2S      ;;LOOP
2927 016552 062401      3S:  ADD      (R4)+,R1    ;;RESTORE SUBTRACTED VALUE
2928 016554 005502      ADC     R2
2929 016556 062402      ADD     (R4)+,R2
2930 016560 022525      CMP     (R5)+,(R5)+  ;;MOVE TO NEXT TEN POWER
2931 016562 052703      BIS     #0,R3     ;;CHANGE PARTIAL TO ASCII
2932 016566 110320      MOVB   R3,(R0)+  ;;SAVE IT
2933 016570 005327      DEC     (PC)+    ;;DONE?
2934 016572 000000      4S:  .WORD   0
2935 016574 001357      BNE    1S      ;;BR IF NO
2936 016576 105020      CLRB   (R0)+    ;;TERMINATOR
2937 016600 104413      RESREG  ;;RESTORE REGISTERS
2938 016602 000207      RTS     PC      ;;RETURN
2939 016604 145000      STNPHR: 145000  ;;1.0E09
2940 016606 035632      35632
2941 016610 160400      160400      ;;1.0E08
2942 016612 002765      2765
2943 016614 113200      113200      ;;1.0E07
2944 016616 000230      230
2945 016620 041100      041100      ;;1.0E06
2946 016622 000017      17
2947 016624 103240      103240      ;;1.0E05
2948 016626 000001      1
2949 016630 023420      23420      ;;1.0E04
2950 016632 000000      0
2951 016634 001750      1750      ;;1.0E03
2952 016636 000000      0
2953 016640 000144      144      ;;1.0E02
2954 016642 000000      0
2955 016644 000012      12      ;;1.0E01
2956 016646 000000      0
2957 016650 000001      1      ;;1.0E00
2958 016652 000000      0
2959 016654 000014      SDECVL: .BLKB 12.      ;;RESERVE STORAGE FOR ASCII STRING

```

000060

```

2960
2961
2962
2963
2964
2965 016670
2966
2967
2968 016670
2969
2970
2971
2972
2973
2974
2975
2976

```

```

.SERRR SERRTYP
.SBTTL ERROR HANDLER ROUTINE

STARS
*****
;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 SERRTYP ON ERROR
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;SW15=1 HALT ON ERROR
;SW13=1 INHIBIT ERROR TYPEOUTS
;CALL

```

```

2977 ;* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2978
2979 016670 $ERROR:
2980 016670 104406 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
2981 016672 105237 001103 7S: INCB SERFLG ;;SET THE ERROR FLAG
2982 016676 001775 BEQ 7S ;;DON'T LET THE FLAG GO TO ZERO
2983 016700 013777 001102 162234 MOV $STNM,2DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
2984 016706 005237 001112 INC SERTTL ;;INC THE ERROR COUNT
2985 016712 011637 001116 MOV (SP),SERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
2986 016716 162737 000002 001116 SUB 2,SERRPC
2987 016724 117737 162166 001114 MOVB 2SERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
2988 016732 032777 020000 162200 BIT 8BIT13,$SWR ;;SKIP TYPEOUT IF SET
2989 016740 001004 BNE 20S ;;SKIP TYPEOUTS
2990 016742 004737 016766 JSR PC,SERRTYP ;;GO TO USER ERROR ROUTINE
2991 016746 104401 001163 TYPE ,SCLRF
2992 016752
2993 016752 005777 162162 20S:
2994 016756 100002 2S: TST 2SWR ;;HALT ON ERROR
2995 016760 000000 BPL 3S ;;SKIP IF CONTINUE
2996 016762 104406 HALT ;;HALT ON ERROR!
2997 016764 3S: CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
2998 016764 000002 RTI ;;RETURN
2999 016766
3000 .SBTTL ERROR MESSAGE TYPEOUT ROUTINE
3001
3002 016766 STARS
3003 *****
3004 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
3005 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" (SERRTB),
3006 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
3007
3008 016766 SERRTYP:
3009 016766 104401 001163 TYPE SCLRF ;; "CARRIAGE RETURN" & "LINE FEED"
3010 016772 010046 MOV RO,-(SP) ;;SAVE RO
3011 016774 005000 CLR RO ;;PICKUP THE ITEM INDEX
3012 016776 153700 001114 BISB 2$ITEMB,RO
3013 017002 001004 BNE 1S ;; IF ITEM NUMBER IS ZERO, JUST
3014 3014 TYPE THE PC OF THE ERROR
3015 017004 TYPOCT SERRPC,<ERROR ADDRESS>
3016 017004 013746 001116 MOV SERRPC,-(SP) ;;SAVE SERRPC FOR TYPEOUT
3017 3017 ERROR ADDRESS
3018 017010 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3019 017012 000426 BR 6S ;;GET OUT
3020 017014 005300 1S: DEC RO ;;ADJUST THE INDEX SO THAT IT WILL
3021 017016 006300 ASL RO WORK FOR THE ERROR TABLE
3022 017020 006300 ASL RO
3023 017022 006300 ASL RO
3024 017024 062700 001166 ADD #SERRTB,RO ;;FORM TABLE POINTER
3025 017030 012037 017040 MOV (RO)+,2S ;;PICKUP "ERROR MESSAGE" POINTER
3026 017034 001404 BEQ 3S ;;SKIP TYPEOUT IF NO POINTER
3027 017036 104401 TYPE ;;TYPE THE "ERROR MESSAGE"
3028 017040 000000 2S: .WORD 0 ;;"ERROR MESSAGE" POINTER GOES HERE
3029 017042 104401 001163 TYPE SCLRF ;; "CARRIAGE RETURN" & "LINE FEED"
3030 017046 012037 017056 3S: MOV (RO)+,4S ;;PICKUP "DATA HEADER" POINTER
3031 017052 001404 BEQ 5S ;;SKIP TYPEOUT IF 0
3032 017054 104401 TYPE ;;TYPE THE "DATA HEADER"

```

```

3033 017056 000000
3034 017060 104401 001163
3035 017064 011000
3036 017066 001004
3037 017070 012600
3038 017072 104401 001163
3039 017076 000207
3040 017100
3041 017100 013046
3042 017102 104402
3043 017104 005710
3044 017106 001770
3045 017110 104401 017116
3046 017114 000771
3047 017116 020040 000
3048 017122 017122
3049 017122
3050
3051
3052 017122
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076 017122 017646 000000
3077 017126 116637 000001 017345
3078 017134 112637 017347
3079 017140 062716 000002
3080 017144 000406
3081 017146 112737 000001 017345
3082 017154 112737 000006 017347
3083 017162 112737 000005 017344
3084 017170 010346
3085 017172 010446
3086 017174 010546
3087 017176 113704 017347
3088 017202 005404

```

```

4S:  WORD 0
      TYPE SCRLF
5S:  MOV (R0),R0
      BNE 7S
6S:  MOV (SP)+,R0
      TYPE SCRLF
      RTS PC
7S:  TYPOCT 2(R0)+
      MOV 2(R0)+,-(SP)
      TYPOC
      TST (R0)
      BEQ 6S
      TYPE BS
      BR 7S
8S:  .ASCIZ / /
      .EVEN
      .STYPOCT

```

```

::"DATA HEADER" POINTER GOES HERE
::"CARRIAGE RETURN" & "LINE FEED"
::PICKUP "DATA TABLE" POINTER
::GO TYPE THE DATA
::RESTORE R0
::"CARRIAGE RETURN" & "LINE FEED"
::RETURN
::TYPE AN OCTAL NUMBER
::SAVE 2(R0)+ FOR TYPEOUT
::GO TYPE--OCTAL ASCII(ALL DIGITS)
::IS THERE ANOTHER NUMBER?
::BR IF NO
::TYPE TWO(2) SPACES
::LOOP
::TWO(2) SPACES

```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

STARS

```

*****
#THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
#OCTAL (ASCII) NUMBER AND TYPE IT.
#STYPOS---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
#STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
#STYPOS OR STYPOC
#CALL:
#      MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
#      TYPON  ;;CALL FOR TYPEOUT
#STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
#CALL:
#      MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
#      TYPOC  ;;CALL FOR TYPEOUT
STYPOS: MOV 2(SP),-(SP)      ;;PICKUP THE MODE
        MOV 1(SP),SOFILL    ;;LOAD ZERO FILL SWITCH
        MOV (SP)+,SOMODE+1  ;;NUMBER OF DIGITS TO TYPE
        ADD #2,(SP)        ;;ADJUST RETURN ADDRESS
        BR STYPON
STYPOC: MOV #1,SOFILL      ;;SET THE ZERO FILL SWITCH
        MOV #6,SOMODE+1    ;;SET FOR SIX(6) DIGITS
STYPON: MOV #5,SOCNT      ;;SET THE ITERATION COUNT
        MOV R3,-(SP)       ;;SAVE R3
        MOV R4,-(SP)       ;;SAVE R4
        MOV R5,-(SP)       ;;SAVE R5
        MOV SOMODE+1,R4    ;;GET THE NUMBER OF DIGITS TO TYPE
        NEG R4

```

```

3089 017204 062704 000006 ADD #6,R4 ;:SUBTRACT IT FOR MAX. ALLOWED
3090 017210 110437 017346 MOVB R4,SOMODE ;:SAVE IT FOR USE
3091 017214 113704 017345 MOVB $OFILL,R4 ;:GET THE ZERO FILL SWITCH
3092 017220 016605 000012 MOV 12(SP),R5 ;:PICKUP THE INPUT NUMBER
3093 017224 005003 CLR R3 ;:CLEAR THE OUTPUT WORD
3094 017226 006105 1S: ROL R5 ;:ROTATE MSB INTO "C"
3095 017230 000404 BR 3S ;:GO DO MSB
3096 017232 006105 2S: ROL R5 ;:FORM THIS DIGIT
3097 017234 006105 ROL R5
3098 017236 006105 ROL R5
3099 017240 010503 MOV R5,R3
3100 017242 006103 3S: ROL R3 ;:GET LSB OF THIS DIGIT
3101 017244 105337 017346 DECB $OMODE ;:TYPE THIS DIGIT?
3102 017250 100016 BPL 7S ;:BR IF NO
3103 017252 042703 177770 BIC #177770,R3 ;:GET RID OF JUNK
3104 017256 001002 BNE 4S ;:TEST FOR 0
3105 017260 005704 TST R4 ;:SUPPRESS THIS 0?
3106 017262 001403 BEQ 5S ;:BR IF YES
3107 017264 005204 4S: INC R4 ;:DON'T SUPPRESS ANYMORE 0'S
3108 017266 052703 000060 BIS #0,R3 ;:MAKE THIS DIGIT ASCII
3109 017272 052703 000040 5S: BIS #' ,R3 ;:MAKE ASCII IF NOT ALREADY
3110 017276 110337 017342 MOVB R3,$S ;:SAVE FOR TYPING
3111 017302 104401 017342 TYPE #S ;:GO TYPE THIS DIGIT
3112 017306 105337 017344 7S: DECB $OCNT ;:COUNT BY 1
3113 017312 003347 BGT 2S ;:BR IF MORE TO DO
3114 017314 002402 BLT 6S ;:BR IF DONE
3115 017316 005204 INC R4 ;:INSURE LAST DIGIT ISN'T A BLANK
3116 017320 000744 BR 2S ;:GO DO THE LAST DIGIT
3117 017322 012605 6S: MOV (SP)+,R5 ;:RESTORE R5
3118 017324 012604 MOV (SP)+,R4 ;:RESTORE R4
3119 017326 012603 MOV (SP)+,R3 ;:RESTORE R3
3120 017330 016666 000002 000004 MOV 2(SP),4(SP) ;:SET THE STACK FOR RETURNING
3121 017336 012616 MOV (SP)+,(SP)
3122 017340 000002 RTI ;:RETURN
3123 017342 000 8S: .BYTE 0 ;:STORAGE FOR ASCII DIGIT
3124 017343 000 .BYTE 0 ;:TERMINATOR FOR TYPE ROUTINE
3125 017344 000 SOCNT: .BYTE 0 ;:OCTAL DIGIT COUNTER
3126 017345 000 $OFILL: .BYTE 0 ;:ZERO FILL SWITCH
3127 017346 000000 $OMODE: .WORD 0 ;:NUMBER OF DIGITS TO TYPE
3128 017350 $POWER , $PWRMSG , $START
3129 .SBTTL POWER DOWN AND UP ROUTINES
3130 STARS
3131 017350 ;:*****
3132 ;:POWER DOWN ROUTINE
3133 $PWRDN: MOV #SILLUP,$PWRVEC ;:SET FOR FAST UP
3134 017350 012737 017514 000024 MOV #340,$PWRVEC+2 ;:PRIO:7
3135 017356 012737 000340 000026 PUSH <R0,R1,R2,R3,R4,R5>
3136 017364 MOV R0,-(SP) ;:PUSH R0 ON STACK
3137 017364 010046 MOV R1,-(SP) ;:PUSH R1 ON STACK
3138 017366 010146 MOV R2,-(SP) ;:PUSH R2 ON STACK
3139 017370 010246 MOV R3,-(SP) ;:PUSH R3 ON STACK
3140 017372 010346 MOV R4,-(SP) ;:PUSH R4 ON STACK
3141 017374 010446 MOV R5,-(SP) ;:PUSH R5 ON STACK
3142 017376 010546 PUSH @SWR ;:PUSH @SWR ON STACK
3143 017400 MOV @SWR,-(SP)
3144 017400 017746 161534

```

```

3145 017404 010637 017520      MOV      SP,SSAVR6      ;;SAVE SP
3146 017410 012737 017422 000024  MOV      @SPWRUP,@@PWVVEC ;;SET UP VECTOR
3147 017416 000000      HALT
3148 017420 000776      BR      -2      ;;HANG UP
3149
3150 017422      STARS
3151      ;*****
3152      :POWER UP ROUTINE
3153 017422 012737 017514 000024 SPWRUP: MOV      @SILLUP,@@PWVVEC ;;SET FOR FAST DOWN
3154 017430 013706 017520      MOV      SSAVR6,SP      ;;GET SP
3155 017434 005037 017520      CLR      SSAVR6      ;;WAIT LOOP FOR THE TTY
3156 017440 005237 017520 1S: INC      SSAVR6      ;;WAIT FOR THE INC
3157 017444 001375      BNE     1S      ;;OF WORD
3158 017446      POP     @SWR
3159 017446 012677 161466      MOV      (SP)+,@SWR      ;;POP STACK INTO @SWR
3160 017452      POP     <R5,R4,R3,R2,R1,R0>
3161 017452 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
3162 017454 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
3163 017456 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
3164 017460 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
3165 017462 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
3166 017464 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
3167 017466 012737 017350 000024  MOV      @SPWRDN,@@PWVVEC ;;SET UP THE POWER DOWN VECTOR
3168 017474 012737 000340 000026  MOV      @340,@@PWVVEC+2 ;;PRIO:7
3169 017502 104401      TYPE
3170 017504 025176      SPWRMG: .WORD  PWMSG      ;;REPORT THE POWER FAILURE
3171 017506 012716      MOV      (PC)+,(SP)      ;;POWER FAIL MESSAGE POINTER
3172 017510 001704      SPWRAD: .WORD  START      ;;RESTART AT START
3173 017512 000002      RTI
3174 017514 000000      SILLUP: HALT      ;;RESTART ADDRESS
3175 017516 000776      BR      -2      ;;THE POWER UP SEQUENCE WAS STARTED
3176 017520 000000      SSAVR6: 0      ;;BEFORE THE POWER DOWN WAS COMPLETE
3177      ;;PUT THE SP HERE
3178 017522      .SSCOPE
3179      .SBTTL SCOPE HANDLER ROUTINE
3180
3181 017522      STARS
3182      ;*****
3183      ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3184      ;AND LOAD THE TEST NUMBER(STSTN) INTO THE DISPLAY REG.(DISPLAY<7:0>)
3185      ;AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:08>
3186      ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3187      ;#SW14=1      LOOP ON TEST
3188      ;#SW11=1      INHIBIT ITERATIONS
3189      ;#CALL
3190      ;#      SCOPE      ;;SCOPE=IOT
3191
3192 017522      SSCOPE:
3193 017522 104406      1S: CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
3194 017524 032777 040000 161406  BIT      @BIT14,@SWR      ;;LOOP ON PRESENT TEST?
3195 017532 001055      BNE     $OVER      ;;YES IF SW14=1
3196      ;#####START OF CODE FOR THE XOR TESTER#####
3197 017534 000416      $XTSTR: BR      @S      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
3198      ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
3199 017536 013746 000004      MOV      @ERRVEC,-(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3200 017542 012737 017562 000004      MOV      @SS,@ERRVEC      ;;SET FOR TIMEOUT

```



```

3201 017550 005737 177060          TST      @177060          ;; TIME OUT ON XOR?
3202 017554 012637 000004          MOV      (SP)+, @ERRVEC ;; RESTORE THE ERROR VECTOR
3203 017560 000436                    BR      $SVLAD          ;; GO TO THE NEXT TEST
3204 017562 022626                    SS:     CMP      (SP)+, (SP)+ ;; CLEAR THE STACK AFTER A TIME OUT
3205 017564 012637 000004          MOV      (SP)+, @ERRVEC ;; RESTORE THE ERROR VECTOR
3206 017570 000436                    BR      $OVER          ;; LOOP ON THE PRESENT TEST
3207 017572                    6S: ; #####END OF CODE FOR THE XOR TESTER#####
3208 017572 105737 001103                    2S:     TSTB     $ERFLG   ;; HAS AN ERROR OCCURRED?
3209 017576 001404                    BEQ     $S          ;; BR IF NO
3210 017600 105037 001103                    4S:     CLRB     $ERFLG   ;; ZERO THE ERROR FLAG
3211 017604 005037 001160                    CLR     $TIMES        ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
3212 017610 032777 004000 161322 3S:     BIT      @BIT11, @SWR ;; INHIBIT ITERATIONS?
3213 017616 001011                    BNE     $S          ;; BR IF YES
3214 017620 005737 001100                    TST     $PASS        ;; IF FIRST PASS OF PROGRAM
3215 017624 001406                    BEQ     $S          ;; INHIBIT ITERATIONS
3216 017626 005237 001104                    INC     $ICNT        ;; INCREMENT ITERATION COUNT
3217 017632 023737 001160 001104                    CMP     $TIMES, $ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
3218 017640 002012                    BGE     $OVER        ;; BR IF MORE ITERATION REQUIRED
3219 017642 012737 000001 001104 1S:     MOV      @1, $ICNT    ;; REINITIALIZE THE ITERATION COUNTER
3220 017650 013737 017702 001160                    MOV     $SMXCNT, $TIMES ;; SET NUMBER OF ITERATIONS TO DO
3221 017656 105237 001102                    $SVLAD: INCB     $STSTM    ;; COUNT TEST NUMBERS
3222 017662 011637 001106                    MOV     (SP), $LPADR  ;; SAVE SCOPE LOOP ADDRESS
3223 017666 013777 001102 161246 $OVER:  MOV     $STSTM, @DISPLAY ;; DISPLAY TEST NUMBER
3224 017674 013716 001106                    MOV     $LPADR, (SP)  ;; FUDGE RETURN ADDRESS
3225 017700 000002                    RTI
3226 017702 003720                    $SMXCNT: 2000.        ;; MAX. NUMBER OF ITERATIONS
3227
3228 017704                    .SREAD
3229                    .SBTTL TTY INPUT ROUTINE
3230
3231 017704                    STARS
3232                    ; *****
3233                    .ENABL LSB
3234
3235 017704                    STARS
3236                    ; *****
3237                    ; #SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3238                    ; #ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
3239                    ; #SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
3240                    ; #WHEN OPERATING IN TTY FLAG MODE.
3241 017704 022737 000176 001140 $CKSMR: CMP     @SWREG, $SWR ;; IS THE SOFT-SWR SELECTED?
3242 017712 001074                    BNE     $S          ;; BRANCH IF NO
3243 017714 105777 161224                    TSTB   @STKS        ;; CHAR THERE?
3244 017720 100071                    BPL     $S          ;; IF NO, DON'T WAIT AROUND
3245 017722 117746 161220                    MOVB   @STKB, -(SP) ;; SAVE THE CHAR
3246 017726 042716 177600                    BIC    @C177, (SP)  ;; STRIP-OFF THE ASCII
3247 017732 022726 000007                    CMP     @7, (SP)+   ;; IS IT A CONTROL G?
3248 017736 001062                    BNE     $S          ;; NO, RETURN TO USER
3249 017740 123727 001134 000001                    CMPB   $AUTOB, @1   ;; ARE WE RUNNING IN AUTO-MODE?
3250 017746 001456                    BEQ     $S          ;; BRANCH IF YES
3251
3252 017750 104401 020431                    SGTSWR: TYPE    , $CNTLG ;; ECHO THE CONTROL-G (!G)
3253 017754 104401 020436                    TYPE    $MSWR       ;; TYPE CURRENT CONTENTS
3254 017760                    TYPOCT $SWREG      ;; OF THE SWR
3255 017760 013746 000176                    MOV     $SWREG, -(SP) ;; SAVE SWREG FOR TYPEOUT
3256 017764 104402                    TYPOC   ;; GO TYPE--OCTAL ASCII(ALL DIGITS)

```

3257	017766	104401	020447		TYPE	SMNEW	:: PROMPT FOR NEW SMR
3258	017772	005046		19S:	CLR	-(SP)	:: CLEAR COUNTER
3259	017774	005046			CLR	-(SP)	:: THE NEW SMR
3260	017776	105777	161142	7S:	TSTB	2STKS	:: CHAR THERE?
3261	020002	100375			BPL	7S	:: IF NOT TRY AGAIN
3262							
3263	020004	117746	161136		MOVB	2STKB, -(SP)	:: PICK UP CHAR
3264	020010	042716	177600		BIC	81C177, (SP)	:: MAKE IT 7-BIT ASCII
3265							
3266							
3267							
3268	020014	021627	000025	9S:	CMP	(SP), #25	:: IS IT A CONTROL-U?
3269	020020	001005			BNE	10S	:: BRANCH IF NOT
3270	020022	104401	020424		TYPE	SCNTLU	:: YES, ECHO CONTROL-U (↑U)
3271	020026	062706	000006	20S:	ADD	#6, SP	:: IGNORE PREVIOUS INPUT
3272	020032	000757			BR	19S	:: LET'S TRY IT AGAIN
3273							
3274							
3275	020034	021627	000015	10S:	CMP	(SP), #15	:: IS IT A <CR>?
3276	020040	001022			BNE	16S	:: BRANCH IF NO
3277	020042	005766	000004		TST	4(SP)	:: YES, IS IT THE FIRST CHAR?
3278	020046	001403			BEQ	11S	:: BRANCH IF YES
3279	020050	016677	000002	161062	MOV	2(SP), 2SMR	:: SAVE NEW SMR
3280	020056	062706	000006	11S:	ADD	#6, SP	:: CLEAR UP STACK
3281	020062	104401	001163	14S:	TYPE	SCRLF	:: ECHO <CR> AND <LF>
3282	020066	123727	001135	000001	CMPB	\$INTAG, #1	:: RE-ENABLE TTY KBD INTERRUPTS?
3283	020074	001003			BNE	15S	:: BRANCH IF NOT
3284	020076	012777	000100	161040	MOV	#100, 2STKS	:: RE-ENABLE TTY KBD INTERRUPTS
3285	020104	000002		15S:	RTI		:: RETURN
3286	020106	004737	021006	16S:	JSR	PC, STYPEC	:: ECHO CHAR
3287	020112	021627	000060		CMP	(SP), #60	:: CHAR < 0?
3288	020116	002420			BLT	18S	:: BRANCH IF YES
3289	020120	021627	000067		CMP	(SP), #67	:: CHAR > 7?
3290	020124	003015			BGT	18S	:: BRANCH IF YES
3291	020126	042726	000060		BIC	#60, (SP)+	:: STRIP-OFF ASCII
3292	020132	005766	000002		TST	2(SP)	:: IS THIS THE FIRST CHAR
3293	020136	001403			BEQ	17S	:: BRANCH IF YES
3294	020140	006316			ASL	(SP)	:: NO, SHIFT PRESENT
3295	020142	006316			ASL	(SP)	:: CHAR OVER TO MAKE
3296	020144	006316			ASL	(SP)	:: ROOM FOR NEW ONE.
3297	020146	005266	000002	17S:	INC	2(SP)	:: KEEP COUNT OF CHAR
3298	020152	056616	177776		BIS	-2(SP), (SP)	:: SET IN NEW CHAR
3299	020156	000707			BR	7S	:: GET THE NEXT ONE
3300	020160	104401	001162	18S:	TYPE	SQUES	:: TYPE ?<CR><LF>
3301	020164	000720			BR	20S	:: SIMULATE CONTROL-U
3302					.DSABL	LSB	

STARS

```

*****
*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
*****

```

```

3313
3314 020166 011646          SRDCHR: MOV      (SP),-(SP)      ;; PUSH DOWN THE PC
3315 020170 016666 000004 000002 1S:  MOV      4(SP),2(SP)      ;; SAVE THE PS
3316 020176 105777 160742 1S:  TSTB    2STKS          ;; WAIT FOR
3317 020202 100375          BPL      1S              ;; A CHARACTER
3318 020204 117766 160736 000004 1S:  MOVVB   2STKB,4(SP)      ;; READ THE TTY
3319 020212 042766 177600 000004 BIC      8(C177),4(SP)    ;; GET RID OF JUNK IF ANY
3320 020220 026627 000004 000023 CMP      4(SP),#23       ;; IS IT A CONTROL-S?
3321 020226 001013          BNE      3S              ;; BRANCH IF NO
3322 020230 105777 160710 2S:  TSTB    2STKS          ;; WAIT FOR A CHARACTER
3323 020234 100375          BPL      2S              ;; LOOP UNTIL ITS THERE
3324 020236 117746 160704 2S:  MOVVB   2STKB,-(SP)      ;; GET CHARACTER
3325 020242 042716 177600 BIC      8(C177),(SP)    ;; MAKE IT 7-BIT ASCII
3326 020246 022627 000021 CMP      (SP)+,#21       ;; IS IT A CONTROL-Q?
3327 020252 001366          BNE      2S              ;; IF NOT DISCARD IT
3328 020254 000750          BR       1S              ;; YES, RESUME
3329 020256 026627 000004 000140 3S:  CMP      4(SP),#140     ;; IS IT UPPER CASE?
3330 020264 002407          BLT      4S              ;; BRANCH IF YES
3331 020266 026627 000004 000175 CMP      4(SP),#175     ;; IS IT A SPECIAL CHAR?
3332 020274 003003          BGT      4S              ;; BRANCH IF YES
3333 020276 042766 000040 000004 BIC      #40,4(SP)      ;; MAKE IT UPPER CASE
3334 020304 000002 4S:  RTI                    ;; GO BACK TO USER
3335 020306
3336 *****
3337 *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3338 *CALL:
3339 *
3340 *      RDLIN
3341 *      RETURN HERE
3342 *
3343 SRDLIN: MOV      R3,-(SP)      ;; SAVE R3
3344 1S:  MOV      #STTYIN,R3      ;; GET ADDRESS
3345 2S:  CMP      #STTYIN+8.,R3    ;; BUFFER FULL?
3346 BLOS  4S              ;; BR IF YES
3347 RDCHR          ;; GO READ ONE CHARACTER FROM THE TTY
3348 MOVVB   (SP)+,(R3)          ;; GET CHARACTER
3349 10S: CMPB    #177,(R3)       ;; IS IT A RUBOUT
3350 BNE      3S              ;; SKIP IF NOT
3351 4S:  TYPE    'SQUES'         ;; TYPE A '?'
3352 BR       1S              ;; CLEAR THE BUFFER AND LOOP
3353 3S:  MOVVB   (R3),9S          ;; ECHO THE CHARACTER
3354 TYPE    '9S'
3355 CMPB    #15,(R3)+          ;; CHECK FOR RETURN
3356 BNE      2S              ;; LOOP IF NOT RETURN
3357 CLRB    -1(R3)            ;; CLEAR RETURN (THE 15)
3358 TYPE    'SLF'             ;; TYPE A LINE FEED
3359 MOV      (SP)+,R3          ;; RESTORE R3
3360 MOV      (SP),-(SP)        ;; ADJUST THE STACK AND PUT ADDRESS OF THE
3361 MOV      4(SP),2(SP)      ;; FIRST ASCII CHARACTER ON IT
3362 MOV      #STTYIN,4(SP)
3363 RTI
3364 9S:  .BYTE  0              ;; RETURN
3365 .BYTE  0              ;; STORAGE FOR ASCII CHAR. TO TYPE
3366 .BYTE  0              ;; TERMINATOR
3367 STTYIN: .BLKB  8          ;; RESERVE 8 BYTES FOR TTY INPUT
3368 SCNTLU: .ASCIZ /#U/<15><12> ;; CONTROL "U"
3369 SCNTLG: .ASCIZ /#G/<15><12> ;; CONTROL "G"

```

```

3369 020436 005015 053523 020122
3370 020444 020075 000
3371 020447 040 047040 053505
3372 020454 036440 000040
3373 020460
3374
3375
3376 020460
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389 020460 011646
3390 020462 016666 000004 000002
3391 020470
3392 020470 010046
3393 020472 010146
3394 020474 010246
3395 020476 104410
3396 020500 012600
3397 020502 010037 020626
3398 020506 005046
3399 020510 005002
3400 020512 122710 000055
3401 020516 001001
3402 020520 112002
3403 020522 112001
3404 020524 001424
3405 020526 122701 000060
3406 020532 003032
3407 020534 122701 000071
3408 020540 002427
3409 020542 032716 170000
3410 020546 001024
3411 020550 006316
3412 020552 011646
3413 020554 006316
3414 020556 006316
3415 020560 062616
3416 020562 102416
3417 020564 162701 000060
3418 020570 060116
3419 020572 102412
3420 020574 000752
3421 020576 005702
3422 020600 001401
3423 020602 005416
3424 020604 012666 000012

```

```

SMSWR: .ASCIZ <15><12>/SWR = /
SMNEW: .ASCIZ / NEW = /

.SRDEEC
.SBTTL READ A DECIMAL NUMBER FROM THE TTY

STARS
*****
*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
*POSITIVE 32767 TO NEGATIVE 32768.
*CALL:
*      RDEEC          ;; READ A DECIMAL NUMBER
*      RETURN HERE    ;; NUMBER IS ON TOP OF THE STACK
*
SRDEEC: MOV      (SP), -(SP)      ;; PROVIDE SPACE FOR
        MOV      4(SP), 2(SP)    ;; THE INPUT NUMBER
        PUSH     <R0,R1,R2>
        MOV      R0, -(SP)       ;; PUSH R0 ON STACK
        MOV      R1, -(SP)       ;; PUSH R1 ON STACK
        MOV      R2, -(SP)       ;; PUSH R2 ON STACK
1$:     RDLIN
        MOV      (SP)+, R0        ;; READ AN ASCIZ LINE
        MOV      R0, 6$          ;; ADDRESS OF 1ST CHAR.
        CLR      -(SP)           ;; SAVE INCASE OF BAD INPUT
        CLR      R2              ;; CLEAR DATA WORD
        CLR      R2              ;; SIGN SET POSITIVE
        CMPB     #'-, (R0)       ;; SEE IF A MINUS SIGN WAS TYPED
        BNE     2$              ;; BR IF NO MINUS SIGN
        MOVB     (R0)+, R2        ;; SAVE FOR LATER USE
2$:     MOVB     (R0)+, R1        ;; PICKUP THIS CHARACTER
        BEQ     3$              ;; GET OUT IF ZERO
        CMPB     #'0, R1         ;; MAKE SURE THIS CHARACTER
        BGT     5$              ;; IS A DIGIT BETWEEN 0 & 9
        CMPB     #'9, R1
        BLT     5$
        BIT     #'C7777, (SP)    ;; DON'T LET NUMBER GET TO BIG
        BNE     5$              ;; BR IF NUMBER WOULD OVERFLOW
        ASL     (SP)             ;; #2
        MOV     (SP), -(SP)      ;; SAVE FOR LATER
        ASL     (SP)             ;; #4
        ASL     (SP)             ;; #8
        ADD     (SP)+, (SP)      ;; #10.
        BVS     5$              ;; OVERFLOW ISN'T ALLOWED
        SUB     #'0, R1          ;; STRIP AWAY THE ASCII JUNK
        ADD     R1, (SP)         ;; ADD IN THIS DIGIT
        BVS     5$              ;; OVERFLOW ISN'T ALLOWED
        BR     2$              ;; LOOP
3$:     TST     R2              ;; CHECK IF NUMBER IS NEG
        BEQ     4$              ;; BR IF NO
        NEG     (SP)            ;; YES--NEGATE THE NUMBER
4$:     MOV     (SP)+, 12(SP)    ;; SAVE THE RESULT

```

```

3425 020610          POP      <R2,R1,RO>
3426 020610 012602  MOV      (SP)+,R2      ;; POP STACK INTO R2
3427 020612 012601  MOV      (SP)+,R1      ;; POP STACK INTO R1
3428 020614 012600  MOV      (SP)+,RO      ;; POP STACK INTO RO
3429 020616 000002  RTI                    ;; RETURN
3430
3431 020620 005726 5S:   TST      (SP)+      ;; CLEAN PARTIAL NUMBER FROM STACK
3432 020622 105010  CLRB     (RO)          ;; SET A TERMINATOR
3433 020624 104401  TYPE     TYPE          ;; TYPE THE INPUT UP TO BAD CHAR.
3434 020626 000000 6S:   .WORD    0          ;; POINTER GOES HERE
3435 020630 104401  TYPE     $QUES        ;; "?" "CR" & "LF"
3436 020634 000720  BR      1S           ;; TRY AGAIN
3437
3438 020636          .STYPE
3439          .SBTTL TYPE ROUTINE
3440
3441 020636          STARS
3442          ;; *****
3443          ;; *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
3444          ;; *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3445          ;; *NOTE1:          SFULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3446          ;; *NOTE2:          SFILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
3447          ;; *NOTE3:          SFILLC CONTAINS THE CHARACTER TO FILL AFTER.
3448          ;; *
3449          ;; *CALL:
3450          ;; *1) USING A TRAP INSTRUCTION
3451          ;; *   TYPE      ,MESADR      ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3452          ;; *OR
3453          ;; *   TYPE
3454          ;; *   MESADR
3455          ;; *
3456
3457 020636 105737 001157 STYPE: TSTB     STPFLG      ;; IS THERE A TERMINAL?
3458 020642 100002          BPL      1S           ;; BR IF YES
3459 020644 000000          HALT          ;; HALT HERE IF NO TERMINAL
3460 020646 000407          BR      3S           ;; LEAVE
3461 020650 010046 1S:   MOV      RO,-(SP)      ;; SAVE RO
3462 020652 017600 000002  MOV      @2(SP),RO      ;; GET ADDRESS OF ASCIZ STRING
3463 020656 112046 2S:   MOVB     (RO)+,-(SP)  ;; PUSH CHARACTER TO BE TYPED ONTO STACK
3464 020660 001005          BNE      4S           ;; BR IF IT ISN'T THE TERMINATOR
3465 020662 005726          TST      (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
3466 020664 012600 60S:  MOV      (SP)+,RO      ;; RESTORE RO
3467 020666 062716 000002 3S:   ADD      @2,(SP)      ;; ADJUST RETURN PC
3468 020672 000002          RTI                    ;; RETURN
3469 020674 122716 000011 4S:   CMPB     @HT,(SP)      ;; BRANCH IF <HT>
3470 020700 001430          BEQ      8S           ;;
3471 020702 122716 000200          CMPB     @CRLF,(SP)   ;; BRANCH IF NOT <CRLF>
3472 020706 001006          BNE      5S           ;;
3473 020710 005726          TST      (SP)+      ;; POP <CR><LF> EQUIV
3474 020712 104401          TYPE     TYPE          ;; TYPE A CR AND LF
3475 020714 001163          SCRLF
3476 020716 105037 021052          CLRB     $CHARCNT    ;; CLEAR CHARACTER COUNT
3477 020722 000755          BR      2S           ;; GET NEXT CHARACTER
3478 020724 004737 021006 5S:   JSR      PC,$STYPEC    ;; GO TYPE THIS CHARACTER
3479 020730 123726 001156 6S:   CMPB     $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
3480 020734 001350          BNE      2S           ;; IF NO GO GET NEXT CHAR.

```

```

3481 020736 013746 001154      MOV      $NULL,-(SP)      ;; GET # OF FILLER CHARS. NEEDED
3482                                     ;; AND THE NULL CHAR.
3483 020742 105366 000001      7S:     DECB      1(SP)      ;; DOES A NULL NEED TO BE TYPED?
3484 020746 002770                BLT      6S                ;; BR IF NO--GO POP THE NULL OFF OF STACK
3485 020750 004737 021006      JSR      PC,$TYPEC      ;; GO TYPE A NULL
3486 020754 105337 021052      DECB      $CHARCNT      ;; DO NOT COUNT AS A COUNT
3487 020760 000770                BR       7S                ;; LOOP

```

;;HORIZONTAL TAB PROCESSOR

```

3491 020762 112716 000040      8S:     MOVB      #' (SP)      ;; REPLACE TAB WITH SPACE
3492 020766 004737 021006      9S:     JSR      PC,$TYPEC      ;; TYPE A SPACE
3493 020772 132737 000007 021052  BITB      #7,$CHARCNT      ;; BRANCH IF NOT AT
3494 021000 001372                BNE      9S                ;; TAB STOP
3495 021002 005726                TST      (SP)+            ;; POP SPACE OFF STACK
3496 021004 000724                BR       2S                ;; GET NEXT CHARACTER
3497 021006 105777 160136      $TYPEC: TSTB      $STPS          ;; WAIT UNTIL PRINTER IS READY
3498 021012 100375                BPL      $TYPEC
3499 021014 116677 000002 160130  MOVB      2(SP),$STPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
3500 021022 122766 000015 000002  CMPB      #CR,2(SP)        ;; IS CHARACTER A CARRIAGE RETURN?
3501 021030 001003                BNE      1S                ;; BRANCH IF NO
3502 021032 105037 021052      CLRB      $CHARCNT        ;; YES--CLEAR CHARACTER COUNT
3503 021036 000406                BR       $TYPEX           ;; EXIT
3504 021040 122766 000012 000002  1S:     CMPB      #LF,2(SP)      ;; IS CHARACTER A LINE FEED?
3505 021046 001402                BEQ      $TYPEX           ;; BRANCH IF YES
3506 021050 105227                INCB      (PC)+            ;; COUNT THE CHARACTER
3507 021052 000000      $CHARCNT: .WORD      0      ;; CHARACTER COUNT STORAGE
3508 021054 000207      $TYPEX:  RTS      PC

```

;;STRAP  
;SBTTL TRAP DECODER

```

3514 021056
3515
3516
3517 021056
3518
3519
3520
3521
3522
3523
3524 021056 010046 000002      STRAP:  MOV      RO,-(SP)      ;; SAVE RO
3525 021060 016600                MOV      2(SP),RO          ;; GET TRAP ADDRESS
3526 021064 005740                TST      -(RO)            ;; BACKUP BY 2
3527 021066 111000                MOVB     (RO),RO          ;; GET RIGHT BYTE OF TRAP
3528 021070 006300                ASL      RO                ;; POSITION FOR INDEXING
3529 021072 016000 021112      MOV      $TRPAD(RO),RO    ;; INDEX TO TABLE
3530 021076 000200                RTS      RO                ;; GO TO ROUTINE

```

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

```

3535 021100 011646 000004 000002      STRAP2: MOV      (SP),-(SP)  ;; MOVE THE PC DOWN
3536 021102 016666                MOV      4(SP),2(SP)      ;; MOVE THE PSW DOWN

```

3537 021110 000002  
3538  
3539 021112  
3540 021112  
3541  
3542  
3543  
3544  
3545  
3546  
3547  
3548 021112 021100  
3549 021114 020636  
3550 021116  
3551 021116  
3552 021116 017146  
3553 021120  
3554 021120  
3555 021120 017122  
3556 021122  
3557 021122  
3558 021122 017162  
3559  
3560 021124  
3561 021124  
3562 021124 017754  
3563  
3564 021126  
3565 021126  
3566 021126 017704  
3567 021130  
3568 021130  
3569 021130 020166  
3570 021132  
3571 021132  
3572 021132 020306  
3573 021134  
3574 021134  
3575 021134 020460  
3576 021136  
3577 021136  
3578 021136 016242  
3579 021140  
3580 021140  
3581 021140 016300  
3582  
3583 021142  
3584 021142  
3585 021142 014040  
3586 021144  
3587 021144  
3588 021144 013406  
3589 021146  
3590 021146  
3591 021146 013500  
3592 021150

```
RTI ;;RESTORE THE PSW
SETTRAP TYPE,STYPE,↑/TTY TYPEOUT ROUTINE/
SSSET TYPE,STYPE,\(TRAP+STRP),\STRP,(TTY TYPEOUT ROUTINE)
.SBTTL TRAP TABLE
; *THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
; *BY THE "TRAP" INSTRUCTION.
;
; ROUTINE
; -----
;STRPAD: .WORD STRAP2
;STYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
SETTRAP TYPOC,STYPOC,↑/TYPE OCTAL NUMBER (WITH LEADING ZEROS)/
SSSET TYPOC,STYPOC,\(TRAP+STRP),\STRP,(TYPE OCTAL NUMBER (WITH LEADING ZEROS))
;STYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
SETTRAP TYPOS,STYPOS,↑/TYPE OCTAL NUMBER (NO LEADING ZEROS)/
SSSET TYPOS,STYPOS,\(TRAP+STRP),\STRP,(TYPE OCTAL NUMBER (NO LEADING ZEROS))
;STYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
SETTRAP TYPON,STYPON,↑/TYPE OCTAL NUMBER (AS PER LAST CALL)/
SSSET TYPON,STYPON,\(TRAP+STRP),\STRP,(TYPE OCTAL NUMBER (AS PER LAST CALL))
;STYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
SETTRAP GTSWR,SGTSWR,↑/GET SOFT-SWR SETTING/
SSSET GTSWR,SGTSWR,\(TRAP+STRP),\STRP,(GET SOFT-SWR SETTING)
;SGTSWR ;;CALL=GTSWR TRAP+5(104405) GET SOFT-SWR SETTING
SETTRAP CKSWR,SCKSWR,↑/TEST FOR CHANGE IN SOFT-SWR/
SSSET CKSWR,SCKSWR,\(TRAP+STRP),\STRP,(TEST FOR CHANGE IN SOFT-SWR)
;SCKSWR ;;CALL=CKSWR TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
SETTRAP RDCHR,SRDCHR,↑/TTY TYPEIN CHARACTER ROUTINE/
SSSET RDCHR,SRDCHR,\(TRAP+STRP),\STRP,(TTY TYPEIN CHARACTER ROUTINE)
;SRDCHR ;;CALL=RDCHR TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
SETTRAP RDLIN,SRDLIN,↑/TTY TYPEIN STRING ROUTINE/
SSSET RDLIN,SRDLIN,\(TRAP+STRP),\STRP,(TTY TYPEIN STRING ROUTINE)
;SRDLIN ;;CALL=RDLIN TRAP+10(104410) TTY TYPEIN STRING ROUTINE
SETTRAP RDDEC,SRDDEC,↑/READ A DECIMAL NUMBER FROM TTY/
SSSET RDDEC,SRDDEC,\(TRAP+STRP),\STRP,(READ A DECIMAL NUMBER FROM TTY)
;SRDDEC ;;CALL=RDDEC TRAP+11(104411) READ A DECIMAL NUMBER FROM TTY
SETTRAP SAVREG,SSAVREG,↑/SAVE RO-R5 ROUTINE/
SSSET SAVREG,SSAVREG,\(TRAP+STRP),\STRP,(SAVE RO-R5 ROUTINE)
;SSAVREG ;;CALL=SAVREG TRAP+12(104412) SAVE RO-R5 ROUTINE
SETTRAP RESREG,SRESREG,↑/RESTORE RO-R5 ROUTINE/
SSSET RESREG,SRESREG,\(TRAP+STRP),\STRP,(RESTORE RO-R5 ROUTINE)
;SRESREG ;;CALL=RESREG TRAP+13(104413) RESTORE RO-R5 ROUTINE
SETTRAP INOCT,INOCTR,↑/INPUT OCTAL ROUTINE/
SSSET INOCT,INOCTR,\(TRAP+STRP),\STRP,(INPUT OCTAL ROUTINE)
;INOCTR ;;CALL=INOCT TRAP+14(104414) INPUT OCTAL ROUTINE
SETTRAP INAR,RINAR,↑/INPUT ADDR. ROUTINE/
SSSET INAR,RINAR,\(TRAP+STRP),\STRP,(INPUT ADDR. ROUTINE)
;RINAR ;;CALL=INAR TRAP+15(104415) INPUT ADDR. ROUTINE
SETTRAP OUTAR,ROUTA,↑/INPUT OUTPUT ADDR. ROUTINE/
SSSET OUTAR,ROUTA,\(TRAP+STRP),\STRP,(INPUT OUTPUT ADDR. ROUTINE)
;ROUTA ;;CALL=OUTAR TRAP+16(104416) INPUT OUTPUT ADDR. ROUTINE
SETTRAP PATAR,RPATA,↑/INPUT PATTERN ROUTINE/
```

```

3593 021150          $$SET  PATAR,RPATA, \<TRAP+STRAP> \STRAP <INPUT PATTERN ROUTINE>
3594 021150 016116  RPATA  ;;CALL=PATAR TRAP+17(104417) INPUT PATTERN ROUTINE
3595 021152          SETTRAP DELAR,RDELA, <INPUT DELAY TIME ROUTINE>
3596 021152          $$SET  DELAR,RDELA, \<TRAP+STRAP> \STRAP <INPUT DELAY TIME ROUTINE>
3597 021152 015152  RDELA  ;;CALL=DELAR TRAP+20(104420) INPUT DELAY TIME ROUTINE
3598 021154          SETTRAP DELAY2,RDELA2, <SECONDARY DELAY ROUTINE>
3599 021154          $$SET  DELAY2,RDELA2, \<TRAP+STRAP> \STRAP <SECONDARY DELAY ROUTINE>
3600 021154 015674  RDELA2 ;;CALL=DELAY2 TRAP+21(104421) SECONDARY DELAY ROUTINE
3601 021156          SETTRAP DELAY,RDELAY, <ROUTINE TO DELAY XX MILLISEC>
3602 021156          $$SET  DELAY,RDELAY, \<TRAP+STRAP> \STRAP <ROUTINE TO DELAY XX MILLISEC>
3603 021156 015636  RDELAY ;;CALL=DELAY TRAP+22(104422) ROUTINE TO DELAY XX MILLISEC
3604 021160          SETTRAP CPATR,RCPAT, <ROUTINE TO CHANGE PATTERNS>
3605 021160          $$SET  CPATR,RCPAT, \<TRAP+STRAP> \STRAP <ROUTINE TO CHANGE PATTERNS>
3606 021160 015766  RCPAT  ;;CALL=CPATR TRAP+23(104423) ROUTINE TO CHANGE PATTERNS
3607 021162          SETTRAP IDAC,RDACA, <ROUTINE TO INPUT DAC ADDR>
3608 021162          $$SET  IDAC,RDACA, \<TRAP+STRAP> \STRAP <ROUTINE TO INPUT DAC ADDR>
3609 021162 013664  RDACA  ;;CALL=IDAC TRAP+24(104424) ROUTINE TO INPUT DAC ADDR
3610
3611 021164          SETTRAP CNTAR,RCNTA, <ROUTINE TO INPUT COUNTER MODULE ADDR>
3612 021164          $$SET  CNTAR,RCNTA, \<TRAP+STRAP> \STRAP <ROUTINE TO INPUT COUNTER MODULE ADDR>
3613 021164 013572  RCNTA  ;;CALL=CNTAR TRAP+25(104425) ROUTINE TO INPUT COUNTER MODULE ADDR
3614 021166          SETTRAP ADAR,RADA, <ROUTINE TO INPUT ADDS ADDR>
3615 021166          $$SET  ADAR,RADA, \<TRAP+STRAP> \STRAP <ROUTINE TO INPUT ADDS ADDR>
3616 021166 013752  RADA  ;;CALL=ADAR TRAP+26(104426) ROUTINE TO INPUT ADDS ADDR
3617 021170          SETTRAP INTR,RINIT, <ROUTINE TO ISSUE SYSTEM INITIALIZE>
3618 021170          $$SET  INTR,RINIT, \<TRAP+STRAP> \STRAP <ROUTINE TO ISSUE SYSTEM INITIALIZE>
3619 021170 015734  RINIT  ;;CALL=INTR TRAP+27(104427) ROUTINE TO ISSUE SYSTEM INITIALIZE
3620 021172          SETTRAP FOCTA,ROCTA, <ROUTINE TO CONVERT OCTAL NUMBERS TO ASCII>
3621 021172          $$SET  FOCTA,ROCTA, \<TRAP+STRAP> \STRAP <ROUTINE TO CONVERT OCTAL NUMBERS TO ASCII>
3622 021172 004712  ROCTA  ;;CALL=FOCTA TRAP+30(104430) ROUTINE TO CONVERT OCTAL NUMBERS TO ASCII
3623 021174          SETTRAP QUBR,RQUBR, <ROUTINE TO SET UNI OR BI-POLAR>
3624 021174          $$SET  QUBR,RQUBR, \<TRAP+STRAP> \STRAP <ROUTINE TO SET UNI OR BI-POLAR>
3625 021174 015344  RQUBR  ;;CALL=QUBR TRAP+31(104431) ROUTINE TO SET UNI OR BI-POLAR
3626
3627
3628
3629
3630
3631 021176 005015 041511 026523 MHEAD: .ASCIZ <15><12>:ICS-11 FIELD TEST PROGRAM - MAINDEC-11-DZICA-C :<15><12>
021263 015 033012 032056 MIPA: .ASCIZ <15><12>/6.4 INPUT MODULE ADDRESSES /
021321 015 033012 032456 MOPA: .ASCIZ <15><12>/6.5 OUTPUT MODULE ADDRESSES /
021360 005015 040440 042104 MTOH: .ASCIZ <15><12>/ ADDR DATA GENERIC CODE (IF INTERRUPT)/
021436 005015 047516 042040 MNDA: .ASCIZ <15><12>/NO DAC ADDR. ENTERED/

021465 015 042412 042116 MEND: .ASCIZ <15><12>/END PASS/

021500 005015 027066 020066 MCNT: .ASCIZ <15><12>/6.6 COUNTER MODULE ADDRESSES /
021540 042536 MEXEN: .ASCII /#E/
021542 005015 054105 042520 MNOEN: .ASCIZ <15><12>/EXPERT MODE ENABLED/
021570 047136 MNOEN: .ASCII /#N/
021572 005015 054105 042520 MLEN: .ASCIZ <15><12>/EXPERT MODE DISABLED/
021621 136 114 MLEN: .ASCII /#L/

```



K07

021623	015	034012	032056	.ASCII	<15><12>/8.4 LINE PRINTER MODE ENABLED/
021662	005015	040515	042513	.ASCIZ	<15><12>/MAKE PRINTER READY/
021707	136	112		MAC:	.ASCII /?J/
021711	015	044412	047457	.ASCIZ	<15><12>/I/O MODULES ASSUMED CONNECTED:
021751	136	000104		MDL1:	.ASCIZ /?D/
021754	005015	042523	047503	MDL2:	.ASCIZ <15><12>/SECONDARY DELAY TIME?/
022004	005015	047527	045522	MWK:	.ASCIZ <15><12>/WORKING.../
022021	015	033012	033456	MDAC:	.ASCIZ <15><12>/6.7 DAC ADDRESS /
022044	005015	027066	020070	MDAU:	.ASCIZ <15><12>/6.8 ADDS ADDRESS /
022070	005015	047524	020117	MABOV:	.ASCIZ <15><12>/100 MANY NUMBERS-RETYPE- /
022124	005015	027066	020063	MTN:	.ASCIZ <15><12>/6.3 TEST NUMBER./
022147	015	047012	020117	MTNL:	.ASCIZ <15><12>/NO SUCH TEST/
022166	005015	027066	030461	MIOO:	.ASCIZ <15><12>/6.11 INPUT OR OUTPUT MODULE(I OR O)/
022234	005015	027066	020071	MDL:	.ASCIZ <15><12>/6.9 DELAY TIME (IN MILLISEC) /
022274	005015	052516	041115	MTTL:	.ASCIZ <15><12>/NUMBER TOO BIG!/
022316	005015	027066	030061	MPPH:	.ASCIZ <15><12>/6.10 PATTERN MODIFIER, PATTERN /
022360	005015	037440	000040	MOMARK:	.ASCIZ <15><12>/ ? /
022366	020077	000		MO:	.ASCIZ /? /
022371	015	054412	052517	MSTERR:	.ASCII <15><12>/YOU MUST INITIALLY START PROGRAM AT ADDR. 200/
022450	005015	043101	042524	.ASCIZ	<15><12>/AFTER RESART AT ADDR. 210 IS ALLOWED/
022517	015	044412	053116	MNRFN:	.ASCIZ <15><12>/INVALID ADDRESS: /
022543	015	041412	052517	EM1:	.ASCIZ <15><12>/COULD NOT SEND-RECEIVE DATA/
022601	015	051412	047105	EM2:	.ASCIZ <15><12>/SEND-RECIEVE DATA ERROR/
022633	015	041412	047117	EM3:	.ASCIZ <15><12>/CONVERT BIT FAILED TO SET/
022667	015	041412	047117	EM4:	.ASCIZ <15><12>/CONVERT BIT FAILED TO CLEAR/
022725	015	041412	047101	EM5:	.ASCIZ <15><12>/CAN'T READ DATA REGISTER/
022760	005015	030101	032460	EM6:	.ASCIZ <15><12>/ADDS FAILED TO INTERRUPT/
023013	015	040412	042104	EM7:	.ASCIZ <15><12>/ADDR. OR GENERIC CODE INCORRECT ON INTERRUPT/
023072	005015	044522	020106	EM10:	.ASCIZ <15><12>/RIF DID NOT CLEAR INTR FLAG ON ADDS/
023140	051105	047522	020122	DH1:	.ASCII ?ERROR A/D?
023154	005015	020040	041520	.ASCIZ	<15><12>/ PC ADDR/
023173	105	051122	051117	DH2:	.ASCII /ERROR MODULE GOOD BAD/
023227	015	020012	050040	.ASCIZ	<15><12>/ PC ADDR DATA DATA/
023267	105	051122	051117	DH3:	.ASCII /ERROR MODULE ICAR/
023315	015	020012	050040	.ASCIZ	<15><12>/ PC ADDR EXP'D REC'D/
023355	015	047012	020117	MNAE:	.ASCIZ <15><12>/NO COUNTER MODULE ADDRS. ENTERED/
023417	015	041412	052517	EM11:	.ASCIZ <15><12>/COULD NOT READ-WRITE INTO COUNTER MODULE BUFFER/
023501	015	020012	051105	DH4:	.ASCII <15><12>/ ERROR COUNTER/
023522	005015	020040	041520	.ASCIZ	<15><12>/ PC ADDR/
023542	005015	047503	047125	EM12:	.ASCIZ <15><12>/COUNTER MODULE-UP COUNT INCORRECT/
023606	005015	047503	047125	EM13:	.ASCIZ <15><12>/COUNTER MODULE FAILED TO INTERRUPT/
023653	015	041412	052517	EM14:	.ASCIZ <15><12>/COUNTER MODULE-ADDR. OR GENERIC CODE INCORRECT ON INTERRUPT/

023751	015	041412	052517	EM15:	.ASCIZ	<15><12>/COUNTER MODULE DIDN'T STOP COUNTING ON OVERFLOW/
024033	015	051012	043111	EM16:	.ASCIZ	<15><12>/RIF DIDN'T CLEAR INTERRUPT FLAG ON COUNTER/
024110	005015	054523	027123	EM17:	.ASCIZ	<15><12>/SYS. INIT. FAILED TO CLEAR COUNTER BIT(S)/
024164	005015	046111	042514	EM20:	.ASCIZ	<15><12>/ILLEGAL INTERRUPT POSTED ON ICS BUS/
024232	005015	041511	020123	EM21:	.ASCIZ	<15><12>/ICS FAILED TO INTERRUPT CPU/
024270	005015	046111	042514	EM22:	.ASCIZ	<15><12>/ILLEGAL ADDR. OR GEN CODE BITS IN ICAR/
024341	015	051412	051531	EM23:	.ASCIZ	<15><12>/SYS. INIT. FAILED TO CLEAR COUNTER INTR. FLAG/
024421	015	041412	052517	EM24:	.ASCIZ	<15><12>/COUNTER STARTED COUNTING AFTER SYS. INITIALIZE/
024502	005015	030101	032460	EM25:	.ASCIZ	<15><12>/A005 READ DUAL ADDR. ERROR/
024537	015	040412	030060	EM26:	.ASCIZ	<15><12>/A005 DUAL ADDR. ERROR/
024567	015	051412	047105	EM27:	.ASCIZ	<15><12>/SEND-RECIEVE DATA ERROR/
024621	015	042412	051122	DMS:	.ASCII	<15><12>?ERROR A/D DUAL?
024647	015	020012	050040		.ASCIZ	<15><12>/ PC ADDR ADDR/
024700	005015	027066	031061	MGURB:	.ASCIZ	<15><12>/6.12 UNIPOLAR OR BIPOLAR ( U OR B ) /
024746	005015	047516	040440	MNAD:	.ASCIZ	<15><12>?NO A/D ADDR. ENTERED?
024775	015	051412	030127	MSMO:	.ASCIZ	<15><12>/SMO0=1 MEANS RETURN TO MONITOR/
025036	030455	027060	030060	MMOVFL:	.ASCIZ	/-10.0000 (OR OVERFLOW)/
025065	053	030061	030056	MPOVFL:	.ASCIZ	/+10.0000 (OR OVERFLOW)/
025114	005015	042522	042520	MREP:	.ASCIZ	<15><12>/REPEAT/
025125	040	042522	042101	MCALOT:	.ASCIZ	/ READING (OCTAL)/
025146	053040	046117	051524	MCALT1:	.ASCIZ	/ VOLTS (DECIMAL) = /
025172	000055			MMINUS:	.ASCIZ	/-/
025174	000053			MPLUS:	.ASCIZ	/+ /
025176	005015	042522	052123	PWRMSG:	.ASCIZ	<15><12>/RESTARTING AFTER A POWER FAILURE/<15><12><12>
025244	005001	042524	052123	MHT0:	.ASCIZ	<1><12>/TEST 0 - INPUT AND OUTPUT MODULE EXERCISER/<15><12>
025323	015	052012	051505	MHT1:	.ASCIZ	<15><12>/TEST 1 - INPUT OR OUTPUT MODULE EXERCISER/<15><12>
025401	015	052012	051505	MHT2:	.ASCIZ	<15><12>/TEST 2 - DAC CALIBRATION/<15><12>
025436	005015	042524	052123	MHT3:	.ASCIZ	<15><12>/TEST 3 - DAC INTERACTION/<15><12>
025473	015	052012	051505	MHT4:	.ASCIZ	<15><12>/TEST 4 - COUNTER MODULE TEST/<15><12>
025534	005015	042524	052123	MHT5:	.ASCIZ	<15><12>?TEST 5 - A/D LOGIC TEST?<15><12>

M07

```

025570 005015 042524 052123 MHT6: .ASCIZ <15><12>?TEST 6 - A/D CALIBRATION?<15><12>

025625 015 052012 051505 MHT8: .ASCIZ <15><12>?TEST 7 - A/D REPEATABILITY?<15><12>
025664 005015 027070 020064 MLPV: .ASCIZ <15><12>/8.4 LINE PRINTER OPTION AVAILABLE/
025730 005015 047111 040526 MIVP: .ASCIZ <15><12>/INVALID PATTERN MODIFIER/

025763 015 033012 030456 MGAIN: .ASCIZ <15><12>/6.13 GAIN /

026000 005015 027066 032061 MCHAN: .ASCIZ <15><12>/6.14 CHANS (SC,EC)/
026025 015 033012 030456 MTOL: .ASCIZ <15><12>/6.16 TOLERANCE /

026047 015 042412 051122 MCHER1: .ASCIZ <15><12>/ERROR! STARTING CHAN. GREATER THAN END CHAN./
026126 005015 040514 052123 MINNN: .ASCIZ <15><12>/LAST CHARACTER NOT A OCTAL DIGIT--RETYPE IT. /
026207 015 037412 047111 MINKN: .ASCIZ <15><12>?INPUT NOT UNDERSTOOD--RETYPE IT./

026253 015 042412 051122 MCHANH: .ASCIZ <15><12>/ERROR! NO SUCH CHAN./

026302 005015 027066 032461 MAVE0: .ASCIZ <15><12>/6.15 EXPECTED AVERAGE /
026333 015 047012 020117 MMSG: .ASCIZ <15><12>/NO SUCH GAIN!/

026353 015 047012 046525 MNTL: .ASCIZ <15><12>/NUMBER TOO LARGE-MAX=7777/
026406 020040 000 M2SP: .ASCIZ / /
026411 015 051012 050105 MREPFT: .ASCIZ <15><12>/REPEATIBILITY FORCED TYP0UT/

026447 015 051012 050105 MREPER: .ASCIZ <15><12>/REPEATABILITY ERROR!/

026476 005015 041440 040510 MREPT1: .ASCIZ <15><12>/ CHAN GAIN LOW AVER HIGH/<15><12>

026550 005015 046040 020117 MREPT2: .ASCII <15><12>/ LO -5 -4 -3 -2 -1 AV/
026613 040 025440 020061 .ASCIZ / +1 +2 +3 +4 +5 HI/<15><12>

026653 015 033012 030456 MFILE: .ASCIZ <15><12>/6.1 FILE BOX TO BE TESTED/
026707 015 033012 031056 MVECT: .ASCIZ <15><12>/6.2 ICS VECTOR ADDRESS/
026740 005015 046111 042514 ILLEG: .ASCIZ <15><12>/ILLEGAL NUMBER/<15><12>
026763 015 040412 042104 MRANG1: .ASCIZ <15><12>/ADDRESS /
026776 047040 052117 053440 MRANG2: .ASCIZ / NOT WITHIN FILE BOX RANGE/<15><12>
027033 015 043012 046111 NOINT: .ASCIZ <15><12>/FILE BOX DID NOT INTERRUPT-FATAL/
027076 005015 044506 042514 FILINT: .ASCIZ <15><12>/FILE BOX INTERRUPTED AT /
027131 040 026455 041440 CKJMP: .ASCIZ / -- CHECK JUMPERS /
027154 005015 047516 026516 NONXST: .ASCIZ <15><12>/NON-EXISTENT FILE BOX/
.EVEN

```

```

3632 027204 015 012
3633 027206 000000
3634 027240
3635 027240
3636
3637
3638 027240
3639 000001

```

```

OUTBF1: .BYTE 15,12
OUTBF: 0
      .=. +30
BUFFER:
:*END ADDRESS IS FUNCTION OF HOW MANY CHANS. ARE BEING EXERCISED
:*AT ONE TIME IN TEST 7.
$ENDAD=.
.END

```









M3INDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 81  
DZICAC.NE2 21-MAR-77 15:28 CROSS REFERENCE TABLE -- USER SYMBOLS

MHT2	025401	1061	3631#											
MHT3	025436	1085	3631#											
MHT4	025473	1106	3631#											
MHT5	025534	1382	3631#											
MHT6	025570	1555	3631#											
MHT8	025625	1762	3631#											
MINKN	026207	2492	3631#											
MINNN	026126	2489	3631#											
MIOO	022166	1004	3631#											
MIPA	021263	2243	3631#											
MIVP	025730	2780	3631#											
MLEN	021621	2525	3631#											
MLPAV	025664	612	3631#											
MMINUS	025172	1642	3631#											
MPOVFL	025036	1620	3631#											
MNAD	024746	1572	3631#											
MNAE	023355	1115	3631#											
MNDA	021436	1066	1090	3631#										
MNOEN	021570	2517	3631#											
MNRFN	022517	2646	3631#											
MNSG	026333	1785	3631#											
MNTL	026353	1849	3631#											
MOPA	021321	2267	3631#											
MPLUS	025174	1647	3631#											
MPOVFL	025065	1614	1625	3631#										
MPPM	022316	2767	3631#	3631#										
MO	022366	1006	2357	2550	2600	3631#								
MOMARK	022360	2485	3631#											
MOURB	024700	2598	3631#											
MREP	025114	1874	3631#											
MREPER	026447	2005	3631#											
MREPFT	026411	2008	3631#											
MREPT1	026476	2022	3631#											
MREPT2	026550	2052	3631#											
MSERR	022371	694	3631#											
MSWD	024775	1032	1566	3631#										
MTN	022124	708	3631#											
MTNL	022147	721	3631#											
MTOL	026025	1861	3631#											
MTTL	022274	2565	2586	3631#										
MTOH	021360	784	3631#											
MVECT	026707	541	3631#											
MVK	022004	1033	1050	1069	1093	1109	1385	1567	1871	3631#				
M2SP	026406	2027	2034	2041	2055	2112	3631#							
NA07	013070	2120#												
NA17	013072	2121#												
NIN	015050	2370#	2414#	2415#	2416#	2418#	2420#	2430	2447	2458	2476	2504#		
NINC	015056	2369#	2424	2428#	2432	2436#	2454	2457#	2461#	2472	2477#	2508#		
NINC1	015060	2360#	2424	2431	2432#	2433#	2442	2454	2472	2509#				
NINI	015052	2447#	2452#	2453#	2456	2458	2505#							
NOINT	027033	569	3631#											
NONXST	027154	537	3631#											
NOSET	015076	2401	2515#											
NRANG1	026763	2658	3631#											
NRANG2	026776	2662	3631#											
OUTADR	013236	797	823	903	1052#	2179#	2268	2273	2277					











JOB

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 86  
 DZICAC.NE2 21-MAR-77 15:28 CROSS REFERENCE TABLE -- USER SYMBOLS

SMSR	020436	3253	3369#											
SFXCNT	017702	3220	3226#											
SNLL	001154	180#	3481	3510										
SOCNT	017344	3083#	3112#	3125#										
SOMODE	017346	3078#	3082#	3087	3090#	3101#	3127#							
SOVER	017666	3195	3206	3218	3223#									
SPASS	001100	155#	686#	769#	1122#	1532#	3214	3227						
SPARD	017510	3172#												
SPARDN	017350	466	3134#	3167										
SPRING	017504	3170#												
SPRUP	017422	3146	3153#											
SQLS	001162	185#	2999	3300	3351	3367	3435	3437	3510					
SRAND	016336	2753	2850#											
S-DOCHR	020166	3314#	3567	3569										
SRODEC	020460	3389#	3573	3575										
SROLIN	020306	3343#	3570	3572										
SROCT=	#####	3573												
SROSZ =	000010	3335#												
SRESAE	016300	2824#	3581											
SR2A =	#####	3582												
SSAVRE	016242	2807#	3576	3578										
SSAVR6	017520	3145#	3154	3155#	3156#	3176#								
SSB2D	016440	1811	2106	2889#										
SSCOPE	017522	460	3192#											
SSETUP=	000117	450#	459	460	462	464	466	468	469	2980	2996	2998	3193	3235
		3373												
SSTUP =	177777	450#												
SSVLAD	017656	3203	3221#											
SSWR =	164000	3#	16	184	185	468	469	470	2973	2974	2975	2976	2984	2988
		2993	2997	2999	3173	3186	3187	3188	3189	3194	3206	3208	3209	3210
		3211	3212	3223	3226									
SSWRNK=	000000	3189												
STIMES	001160	184#	468#	1147#	1174#	1199#	1216#	1338#	3211#	3217	3220#	3226		
STKB	001146	177#	745	1012	1013	2374	2606	3233	3245	3263	3318	3324		
STKS	001144	176#	605#	727#	1000#	1008	1035#	1051#	1118#	1388#	1568#	2372	2602	2711#
		3233	3243	3260	3284#	3316	3322							
STN =	000001	16#												
STNPR	016604	2918	2919	2933#										
STPB	001152	179#	673	699#	862#	1012#	1870#	2381#	2607#	3499#	3510			
STPFLC	001157	183#	3457	3510										
STPS	001150	178#	672	698#	855	864#	867#	1010	1869#	2379	2604	3497	3510	
STRAP	021056	464	3524#											
STRAP2	021100	3535#	3548											
STRP =	000032	3539#	3540	3541	3550#	3551	3552	3553#	3554	3555	3556#	3557	3558	3559#
		3561	3562	3563#	3565	3566	3567#	3568	3569	3570#	3571	3572	3573#	3574
		3575	3576#	3577	3578	3579#	3580	3581	3582#	3584	3585	3586#	3587	3588
		3589#	3590	3591	3592#	3593	3594	3595#	3596	3597	3598#	3599	3600	3601#
		3602	3603	3604#	3605	3606	3607#	3608	3609	3610#	3612	3613	3614#	3615
		3616	3617#	3618	3619	3620#	3621	3622	3623#	3624	3625	3626#		
STRPAD	021112	3529	3548#											
STSTN	001102	156#	2983	2999	3185	3221#	3223	3227						
STTYIN	020414	3344	3345	3362	3366#									
STYPB=	#####	3559												
STYPDS=	#####	3559												
STYPE	020636	691	3457#	3539	3549									
STYPEC	021006	3286	3478	3485	3492	3497#	3498							





M08

MAINDEC-11-DZICA-C MACY11 27(1006) 22-MAR-77 09:40 PAGE 90  
DZICAC.NE2 21-MAR-77 15:28 CROSS REFERENCE TABLE -- MACRO NAMES

.STYPE 38 3438  
.STYPO 38 3049

.ABS. 027240 000

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

DSKZ:DZICAC.BIN, DSKZ:DZICAC.LST/CRF/SOL=DSKZ:DZICAC.NE2  
RUN-TIME: 14 9 1 SECONDS  
RUN-TIME RATIO: 88/25=3.4  
CORE USED: 23K (46 PAGES)

EOF1DZICACSEQ 00010000 770610 PDP10 411 0