LA180

PRINTER DIAGNOSTIC

MD-11-DZLAE-A

EP DZLAE A DL A

OCT 1976

digital

Made in U.S.A.

FICHE 1 OF 1

IDENTIFICATION

SEQ 0001

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

PRODUCT NAME: LA180 PRINTER DIAGNOSTIC

DATE CREATED: NOVEMBER 1975

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: ROBERT BAKER

TABLE OF CONTENTS

1.0 ABSTRACT

2.0 REQUIREMENTS

2.1 EQUIPMENT
2.2 STORAGE
2.3 PRELIMINARY PROGRAMS

3.0 LOADING PROCEDURE & INITIALIZATION

4.0 STARTING PROCEDURES

5.0 OPERATING PROCEDURES

5.1 SWITCH REGISTER CONTROLS
5.2 CONSOLE TERMINAL KEYBOARD CONTROL
5.3 DYNAMIC SOFTWARE SWITCH REGISTER CONTROL
5.4 ERROR REPORTING

6.0 TEST DESCRIPTIONS

6.1 OPERATOR INTERVENTION TESTS
6.1.1 TEST 00 - INTERFACE & CONTROL TESTS
6.1.2 TEST 01 - TOP OF FORM SWITCH TEST
6.1.3 TEST 02 - PRINT SPEED TIMING TEST

6.2 PRINTING TESTS
6.2.1 TEST 20 - DATA TRANSFER PATHS TEST
6.2.2 TEST 21 - HEAD POSITIONING TEST
6.2.3 TEST 22 - BACKSPACE TEST
6.2.4 TEST 23 - CHARACTER GENERATOR TEST
6.2.5 TEST 24 - NON-PRINTABLE CHARACTER TEST
6.2.6 TEST 25 - BUFFER TEST
6.2.7 TEST 26 - OVERPRINT TEST
6.2.8 TEST 27 - MULTIPLE LINE FEED TEST
6.2.9 TEST 30 - RIBBON FEED TEST
6.2.10 TEST 31 - BELL TEST

6.3 MAINTENANCE AIDS
6.3.1 TEST 60 - LIFE TEST
6.3.2 TEST 61 - SCOPE DRIVE ROUTINE
6.3.3 TEST 62 - LINE PRINT TEST
6.3.4 TEST 63 - CHARACTER PRINT TEST

1.0 ABSTRACT

THE DIAGNOSTICS FOR THE LA180 PRINTER ARE DESIGNED TO EXERCISE ALL AREAS OF THE PRINTER, SIMULATING WORSE CASE CONDITIONS TO DETECT BOTH MECHANICAL AND ELECTRICAL FAULTS. ADDITIONAL FACILITIES WITHIN THE DIAGNOSTIC PROGRAM WILL AID IN ISOLATION OF ANY FAULT CONDITIONS DETECTED.

OPERATION OF THE DIAGNOSTIC PROGRAM WILL BE CONTROLLED FROM THE PROCESSOR SWITCH REGISTER OR FROM AN AVAILABLE CONSOLE DEVICE. THE OPERATOR WILL BE GIVEN AS MUCH CONTROL OVER THE OPERATION OF THE PROGRAM AS POSSIBLE WHILE TRYING TO KEEP THE CONTROL SCHEME SIMPLE.

THIS DIAGNOSTIC PROGRAM WAS DESIGNED TO RUN IN 4K OR LESS OF MEMORY AND BE COMPATIBLE WITH ACT AND XXDP.

2.0 REQUIREMENTS

2.1 EQUIPMENT

THIS DIAGNOSTIC WAS WRITTEN TO RUN ON ALL MODELS OF THE PDP-11 PROCESSOR WITH A LA180 PRINTER USING A STANDARD PARALLEL INTERFACE. THE PROGRAM WILL USE A STANDARD CONSOLE DEVICE, IF AVAILABLE, FOR OPERATOR INSTRUCTIONS AND ERROR REPORTING. IT IS SUGGESTED THAT A CONSOLE DEVICE BE USED WHEN RUNNING THIS DIAGNOSTIC BUT IT IS NOT REQUIRED IF THE CPU HAS A HARDWARE SWITCH REGISTER. IF ANY NON-STANDARD ADDRESSES ARE USED FOR EITHER THE LA180 OR THE CONSOLE DEVICE, CHANGE THE ADDRESSES IN THE COMMON TAG AREA OF THE PROGRAM (STARTING AT LOCATION 1100).

2.2 STORAGE

THIS PROGRAM USES MOST OF 4K OF MEMORY WITHOUT AFFECTING THE AREA USED BY THE ABSOLUTE LOADER.

2.3 PRELIMINARY PROGRAMS

ALL APPLICABLE PDP-11 DIAGNOSTICS SHOULD BE RUN SUCCESSFULLY ON THE PROCESSOR.

3.0 LOADING PROCEDURE & INITIALIZATION

LOAD THE LA180 DIAGNOSTIC PROGRAM FOLLOWING NORMAL PROCEDURES.

IF A HARDWARE SWITCH REGISTER DOES NOT EXIST OR ALL SWITCHES ARE PLACED UP BEFORE STARTING THE DIAGNOSTIC, THE PROGRAM WILL USE THE CONTENTS OF LOCATION 176 AS THE VALUE OF THE SWITCHES. THEREFORE, WHEN USING THE SOFTWARE SWITCH REGISTER BE SURE TO LOAD LOCATION 176 WITH THE DESIRED SWITCH VALUE BEFORE STARTING THE PROGRAM. REFER TO SECTION 5.3 FOR DETAILS ON DYNAMIC SOFTWARE SWITCH REGISTER CONTROL.

REFER TO THE TEST ADDRESS TABLE IN THE PROGRAM LISTING FOR DETAILS ON CHANGING THE PRINTING TEST SEQUENCE OR DELETING TESTS FROM THE DIAGNOSTIC.

4.0 STARTING PROCEDURES

STARTING ADDRESSES:

200 = GENERAL START:
RUN OPERATOR INTERVENTION TESTS THEN ENTER PRINTING TEST SEQUENCE.

600 = RESTART:
ENTER PRINTING TEST SEQUENCE DIRECTLY SKIPPING OPERATOR INTERVENTION TESTS.

604 = GO DIRECTLY TO CONSOLE TERMINAL KEYBOARD CONTROL - SELECT TEST.

STARTING AT 200 WILL RUN THE ENTIRE DIAGNOSTIC PACKAGE. THE PROGRAM WILL FIRST EXECUTE THE OPERATOR INTERVENTION TESTS AND THEN ENTER THE PRINTING TEST SEQUENCE WHERE IT WILL LOOP CONTINUOUSLY. STARTING AT 600 (THE RESTART) WILL SKIP THE OPERATOR INTERVENTION TESTS AND ENTER THE PRINTING TEST SEQUENCE DIRECTLY. STARTING AT 604 WILL CAUSE THE PROGRAM TO GO DIRECTLY TO CONSOLE KEYBOARD CONTROL IF A CONSOLE DEVICE EXISTS, OTHERWISE, THE PROGRAM WILL HALT WAITING FOR A TEST SELECTION FROM THE SWITCH REGISTER. ALSO, BY PLACING THE HALT AND SELECT TEST SWITCH UP (1) BEFORE STARTING THE DIAGNOSTIC, THE DIAGNOSTIC WILL HALT WAITING FOR A TEST SELECTION FROM THE SWITCH REGISTER AFTER INITIALIZATION OF THE PROGRAM.

TO START THE DIAGNOSTIC PROGRAM; SET THE DESIRED STARTING ADDRESS IN THE SWITCH REGISTER AND DEPRESS LOAD ADDRESS, SET THE SWITCH REGISTER OPTIONS AS DESIRED (SEE SECTION 5.1), AND DEPRESS START. THE DIAGNOSTIC PROGRAM WILL NOW RUN IN THE MANNER SELECTED.

5.0 OPERATING PROCEDURES

5.1 SWITCH REGISTER CONTROLS

THE FOLLOWING, BASIC CONTROL FUNCTIONS ARE AVAILABLE THROUGH THE USE OF THE SWITCH REGISTER. IF A HARDWARE SWITCH REGISTER DOES NOT EXIST OR ALL SWITCHES WERE SET UP BEFORE STARTING THE DIAGNOSTIC, THE PROGRAM WILL USE THE CONTENTS OF LOCATION 176 AS THE VALUE OF THE SWITCH REGISTER. REFER TO SECTION 5.3 FOR DETAILS ON SSR CONTROL.

SWITCH	POSITION	FUNCTION
15	1 (UP)	STOP ON ERROR
	0 (DWN)	CONTINUE ON ERROR
14	1 (UP)	LOOP ON TEST
	0 (DWN)	NORMAL OPERATION
13	1 (UP)	INHIBIT ERROR TYPEOUT
	0 (DWN)	NORMAL OPERATION
12		MANUAL TIMING - OVER ALL PRINT SPEED TIMING
10	1 (UP)	BELL ON ERROR
	0 (DWN)	NORMAL OPERATION
09	1 (UP)	SINGLE CHAR - SCOPE ROUTINE
	0 (DWN)	FULL LINES
08	1 (UP)	HALT & SELECT TEST
	0 (DWN)	NORMAL OPERATION
07-00	# COLUMNS AT START UP.	
05-00	TEST SELECTION DURING DIAG.	
06-00	CHAR SELECTION FOR SCOPE ROUTINE	

5.1.1 SWITCH 15 - STOP ON ERROR

WITH THIS SWITCH UP (1), THE PROGRAM WILL HALT OR WAIT FOR A KEYBOARD ON ANY DETECTED ERROR. WHEN DOWN (0), THE PROGRAM WILL CONTINUE ON ERROR IF POSSIBLE.

5.1.2 SWITCH 14 - LOOP ON TEST

WITH THIS SWITCH UP (1), THE PROGRAM WILL CONTINUE TO LOOP ON THE CURRENT TEST UNTIL THIS SWITCH IS PLACED DOWN (0). AFTER RETURNING THIS SWITCH TO THE DOWN (0) POSITION, THE TEST WILL CONTINUE NORMAL OPERATION AT THE COMPLETION OF THE CURRENT TEST. THUS, WHENEVER THIS SWITCH IS DOWN (0), THE PROGRAM WILL CONTINUE NORMAL OPERATION.

5.1.3 SWITCH 13 - INHIBIT ERROR TYPEOUT

WHENEVER THIS SWITCH IS IN THE UP (1) POSITION, ERROR TYPEOUTS WILL NOT OCCUR.

5.1.4 SWITCH 11 - MANUAL TIMING

THIS SWITCH WILL BE USED TO MANUALLY TIME THE OVERALL PRINT SPEED OF THE LA180 PRINTER IF A CLOCK OPTION DOES NOT EXIST.

5.1.5 SWITCH 10 - BELL ON ERROR

PLACING THIS SWITCH UP (1) WILL CAUSE THE CONSOLE (IF AVAILABLE) TO RING A BELL WHENEVER AN ERROR CONDITION IS DETECTED IN THE LA180 PRINTER.

5.1.6 SWITCH 9 - SINGLE CHAR/FULL LINES CHAR

THIS SWITCH WILL BE USED TO SELECT WHETHER TO SEND ONLY A SINGLE CHARACTER OR FULL LINES OF CHARACTERS TO THE LA180 PRINTER DURING TEST 61 ONLY.

5.1.7 SWITCH 8 - HALT & SELECT TEST

THE PROGRAM WILL HALT WHENEVER THIS SWITCH IS PLACED IN THE UP (1) POSITION. AT THAT TIME, SET THE DESIRED TEST NUMBER IN THE PROPER POSITION IN THE PROCESSOR SWITCH REGISTER.

TO START THE NORMAL TEST SEQUENCE WITH THE SELECTED TEST, PLACE THE HALT AND SELECT TEST SWITCH DOWN (0) THEN DEPRESS THE CONTINUE SWITCH.

TO RUN A SELECTED TEST ONCE AND HALT, LEAVE THE HALT AND SELECT TEST SWITCH UP (1) AND DEPRESS CONTINUE. THE PROGRAM WILL EXECUTE ONE COMPLETE PASS OF THE SELECTED TEST, THEN HALT WAITING FOR ANOTHER TEST SELECTION. TO HALT THE PROGRAM DURING EXECUTION OF THE SELECTED TEST, PLACE THE HALT & SELECT TEST SWITCH DOWN (0) AT ANY TIME. THE PROGRAM WILL HALT AT THE COMPLETION OF THE CURRENT OPERATION AND WAIT FOR ANOTHER TEST SELECTION.

5.1.8 SELECTION OF NUMBER OF COLUMNS

THESE SWITCHES WILL BE USED WHEN THE PROGRAM IS FIRST STARTED TO INPUT THE DESIRED, MAXIMUM NUMBER OF COLUMNS THE DIAGNOSTIC IS TO TEST. THE NUMBER SET MUST BE IN OCTAL AND BE EQUAL TO OR GREATER THAN 2 AND LESS THAN OR EQUAL TO 132(10). IF THE SWITCHES ARE NOT SET WITHIN THESE SET LIMITS, THE PROGRAM WILL DEFAULT TO TESTING 132(10) COLUMNS. THUS, LEAVING THESE SWITCHES DOWN (000) THE PROGRAM WILL AUTOMATICALLY TEST THE FULL 132(10) COLUMNS.

5.1.9 TEST SELECTION

THESE SWITCHES WILL BE USED TO SELECT A DESIRED TEST WHENEVER THE HALT AND SELECT TEST SWITCH IS USED TO HALT THE DIAGNOSTIC PROGRAM.

5.2 CONSOLE TERMINAL - KEYBOARD CONTROL

WHENEVER A CONSOLE TERMINAL IS DETERMINED TO BE AVAILABLE BY THE PROGRAM, THE DIAGNOSTIC WILL BE CAPABLE OF BEING CONTROLLED FROM THE KEYBOARD OF THE CONSOLE DEVICE. TYPING A RUBOUT (DEL) ON THE CONSOLE KEYBOARD AT ANY TIME WILL CAUSE THE PROGRAM TO STOP AND PRINT THE FOLLOWING MESSAGE ON THE CONSOLE DEVICE:

```
SELECT TEST #:
```

TYPE ANY LEGAL TEST NUMBER FOLLOWED BY ONE OF THE FOLLOWING CONTROL CHARACTERS AND A CARRIAGE RETURN:

CHARACTER	FUNCTION
. (PERIOD)	RUN TEST ONCE & RETURN TO TEST SELECTION
L	LOOP ON SELECTED TEST
S	START SEQUENCE WITH SELECTED TEST

THE L AND S MAY BE EITHER UPPER OR LOWER CASE BUT TEST NUMBERS MUST ALWAYS BE ENTERED AS 2 DIGIT NUMBERS.

TO RESET THE DESIRED MAXIMUM NUMBER OF COLUMNS, TYPE A CONTROL-C (↑C) ON THE CONSOLE TERMINAL KEYBOARD AT ANY TIME, THE FOLLOWING MESSAGE WILL BE TYPED ON THE CONSOLE DEVICE:

```
# COLUMNS =
```

TYPE IN THE DESIRED NUMBER OF COLUMNS (IN DECIMAL) ON THE CONSOLE KEYBOARD FOLLOWED BY A CARRIAGE-RETURN. IF THE SELECTED NUMBER IS LESS THAN 2 OR GREATER THAN 132(10) THE MESSAGE WILL BE REPEATED AND YOU MUST RENTER THE NUMBER OF COLUMNS. WHEN A CORRECT NUMBER IS ENTERED, THE PROGRAM WILL THEN ASK FOR A TEST SELECTION AS DESCRIBED PREVIOUSLY IN THIS SECTION.

TO CHANGE THE NUMBER OF COLUMNS WHEN WAITING FOR A TEST SELECTION, TYPE A CONTROL-C FOLLOWED BY A CARRIAGE RETURN. WHILE INPUTTING A TEST SELECTION OR COLUMN NUMBER THE RUBOUT (DEL) KEY MAY BE USED TO DELETE INCORRECT ENTRIES. AT ALL TIMES SWITCH REGISTER CONTROL WILL STILL BE EFFECTIVE, EVEN IF USING CONSOLE TERMINAL KEYBOARD CONTROL.

5.3 DYNAMIC SOFTWARE SWITCH REGISTER CONTROL

WHENEVER A CONSOLE TERMINAL IS AVAILABLE AND A HARDWARE SWITCH REGISTER IS NOT AVAILABLE (OR ALL SWITCHES WERE UP WHEN THE PROGRAM WAS STARTED), THE PROGRAM WILL RECOGNIZE THE FOLLOWING DYNAMIC SOFTWARE SWITCH REGISTER CONTROL:

TYPING A CONTROL-G (BEL) AT ANY TIME DURING PROGRAM EXECUTION, EXCEPT WHEN WAITING FOR A TEST OR COLUMN NUMBER SELECTION, WILL CAUSE THE DIAGNOSTIC TO STOP THE CURRENT TEST AND TYPE THE FOLLOWING MESSAGE ON THE CONSOLE DEVICE:

SWR = XXXXXX NEW =

WHERE XXXXXX IS THE CURRENT CONTENTS OF THE SOFTWARE SWITCH REGISTER (SSR) IN OCTAL. THE SOFTWARE CONTROL ROUTINE WILL THEN AWAIT OPERATOR ACTION. THE OPERATOR IS THEN REQUIRED TO TYPE ONE OR MORE OF THE LEGAL CHARACTERS 1) 0-7, 2) LINE FEED <LF>, 3)CARRIAGE RETURN <CR>, OR 4) CONTROL-U (↑U). NO CHECK IS MADE FOR CHARACTER LEGALITY. IF THE INPUT CHARACTER IS NOT A LF, CR, OR ↑U IT IS ASSUMED TO BE AN OCTAL DIGIT AND WILL BE ECHOED AS THE DIGIT THAT IS GOING TO BE STORED IN THE SWITCH SETTING.

TO CHANGE THE CONTENTS OF THE SSR THE OPERATOR SIMPLY TYPES THE NEW DESIRED VALUE IN OCTAL, LEADING ZEROS NEED NOT BE TYPED. AND TERMINATES THE INPUT STRING WITH A <CR> OR <LF> DEPENDING ON THE PROGRAM ACTION DESIRED AS DESCRIBED BELOW. THE INPUT VALUE WILL BE TRUNCATED TO THE LAST 6 DIGITS TYPED. AT LEAST ONE DIGIT MUST BE TYPED PRIOR TO THE TERMINATOR BEFORE A CHANGE TO THE SSR WILL OCCUR.

WHEN THE INPUT STRING IS TERMINATED WITH A <CR> THE DIAGNOSTIC WILL CONTINUE EXECUTION FROM THE POINT AT WHICH IT WAS INTERRUPTED. IF A <CR> IS THE ONLY THING TYPED, THE PROGRAM WILL CONTINUE WITHOUT CHANGING THE SSR. IF A <LF> IS USED TO TERMINATE THE INPUT STRING, THE PROGRAM WILL THEN ASK FOR A TEST SELECTION AS DESCRIBED IN SECTION 5.2.

IF A ↑U IS TYPED AT ANY POINT IN THE INPUT STRING PRIOR TO THE TERMINATOR, THE INPUT VALUE WILL BE DISREGARDED AND THE PROMPT MESSAGE WILL BE REPRINTED.

5.4 ERROR REPORTING

IF A CONSOLE TERMINAL EXISTS AND THE INHIBIT ERROR TYPEOUT SWITCH IS DOWN (0), WHENEVER AN ERRO ERROR IS DETECTED THE FOLLOWING ERROR MESSAGE WILL BE PRINTED ON THE CONSOLE DEVICE:

TEST #XX, PC=XXXXXX, ERROR #XXX, MESSAGE ››››››››››

THE ERROR MESSAGE INDICATES THE TEST NUMBER, THE LOCATION WHERE THE ERROR OCCURRED, THE ERROR NUMBER, AND THE TYPE OF ERROR THAT OCCURRED. FOR ADDITIONAL INFORMATION ON ANY ERROR CONDITION, REFER TO THE PROGRAM LISTING.

WHENEVER A CONSOLE TERMINAL IS NOT AVAILABLE THE HALT ON ERROR SWITCH SHOULD BE USED. AFTER AN ERROR OCCURS AND THE PROGRAM HALTS, EXAMINE THE CONTENTS OF $ERRPC TO FIND THE ADDRESS WHERE THE ERROR OCCURRED AND $ITEMB TO FIND THE ERROR NUMBER. THE TEST NUMBER WILL BE LOCATED IN EITHER THE HARDWARE OR SOFTWARE DISPLAY DEPENDING ON CPU TYPE. THEN REFER TO THE PROGRAM LISTING TO DETERMINE THE TYPE OF ERROR THAT OCCURRED AND TO FIND ANY ADDITIONAL INFORMATION REGARDING THAT ERROR. IF NEEDED, THE ERROR MESSAGES ARE LOCATED NEAR THE END OF THE PROGRAM LISTING.

6.0 TEST DESCRIPTIONS

6.1 OPERATOR INTERVENTION TESTS

THIS SERIES OF TESTS CONSISTS OF ALL TESTS NORMALLY EXECUTED WHICH COULD POSSIBLE REQUIRE OPERATOR INTERVENTION. THESE TESTS ARE EXECUTED ONLY ONCE EACH WHEN THE DIAGNOSTIC IS FIRST STARTED UP. A DETAILED DESCRIPTION OF EACH TEST FOLLOWS:

6.1.1 TEST 00 - INTERFACE & CONTROL TESTS

THIS TEST IS DESIGNED AS A COMMAND DECODE AND CONTROL INTERFACE TEST AND INCLUDES CHECKOUT OF THE PRINTER INTERRUPT FACILITY. MANUAL INTERVENTION IS REQUIRED TO TEST THE VARIOUS TESTABLE ERROR (NON-READY) CONDITIONS OF THE PRINTER. OPERATOR INSTRUCTIONS WILL BE PRINTED ON THE CONSOLE DEVICE IF AVAILABLE THEN THE PROGRAM WILL WAIT FOR THE OPERATOR TO COMPLETE THE ACTION. DEPRESS THE SPACE BAR ON THE CONSOLE KEYBOARD OR THE CONTINUE SWITCH ON THE CPU IF NO CONSOLE DEVICE IS AVAILABLE TO TEST THE NEXT CONDITION WHEN READY. IF ANY ERROR CONDITION EXISTS OR ANY NON-EXPECTED RESULTS ARE ENCOUNTERED, AN ERROR MESSAGE WILL BE PRINTED ON THE CONSOLE DEVICE IF AVAILABLE. (REFER TO SECTION 5.3 ON ERROR REPORTING.)

POWER SHOULD BE OFF ON THE LA180 BEFORE STARTING THIS TEST. THE PROGRAM WILL FIRST TEST THAT THE ERROR BIT IS SET AND THE PRINTER IS NOT READY WITH POWER OFF. AN INSTRUCTION WILL THEN ASK FOR THE PRINTER POWER TO BE TURNED ON. TURN POWER ON AND MAKE SURE THERE IS PAPER IN THE PRINTER AND THE PRINTER IS OFF LINE. THE DIAGNOSTIC WILL AGAIN CHECK THAT THE ERROR BIT IS SET AND THE PRINTER IS NOT READY. AN INSTRUCTION ON THE CONSOLE DEVIDE WILL NEXT INFORM THE OPERATOR TO TURN THE LA180 ON LINE. THE PROGRAM WILL NOW CHECK THAT THE ERROR BIT IS CLEAR AND THE PRINTER IS READY. THE NEXT PRINTED INSTRUCTION WILL HAVE THE OPERATOR FORCE A PAPER OUT CONDITION BY OPENING THE PAPER FEED TRACTORS AND REMOVING THE PAPER FROM THE PRINTER. THE DIAGNOSTIC WILL CHECK THAT THE ERROR BIT IS SET AND PRINTER IS NOT READY. THE LAST INSTRUCTION WILL ASK TO RESTORE THE PRINTER TO ON-LINE BY RE-INSERTING PAPER AND CLEARING THE ERROR CONDITION. MAKE SURE THE PRINTER IS SET TO ON-LINE BEFORE CONTINUING. THE PROGRAM WILL TEST TO SEE IF THE ERROR BIT IS CLEARED AND THE PRINTER IS READY.

THE LAST HALF OF THIS TEST WILL BE PERFORMED AUTOMATICALLY WITHOUT FURTHER MANUAL INTERVENTION REQUIRED. THE DIAGNOSTIC WILL ISSUE A RESET INSTRUCTION AND SEE THAT THE ERROR BIT IS CLEAR AND THE PRINTER IS READY. A CARRIAGE RETURN WILL BE LOADED TO THE PRINTER TO SEE IF LOADING THE CHARACTER BUFFER WILL CLEAR THE READY BIT. THE TEST WILL THEN CHECK THAT THE ERROR BIT IS CLEAR AND THE PRINTER READY BIT DOES SET WITHIN A REASONABLE AMOUNT OF TIME. THE FINAL TEST WILL CHECK THAT THE PRINTER WILL NOT INTERRUPT ABOVE PRIORITY LEVEL 3 AND WILL INTERRUPT AT ALL PRIORITY LEVELS BELOW LEVEL 4.

6.1.2 TEST 01 - TOP OF FORM SWITCH TEST

THIS TEST CHECKS ALL POSITIONS OF THE TOP OF FORM SWITCH. THE PROGRAM WILL PRINT INSTRUCTIONS FOR THE NEXT SETTING OF THE TOP OF FORM SWITCH ON THE CONSOLE TERMINAL (IF AVAILABLE) AND THEN WAIT FOR THE OPERATOR TO COMPLETE THE ACTION. AFTER SETTING THE SWITCH, DEPRESS THE SPACE BAR OF THE CONSOLE DEVICE (OR CONTINUE ON THE PROCESSOR IF NO CONSOLE DEVICE EXISTS) TO TEST THAT SWITCH POSITION. AFTER CHECKING ALL POSITIONS, THE PRINTER OUTPUT CAN BE VISUALLY VERIFIED. A LINE OF ALL DASHES IS PRINTED AS A STARTING POINT AND THEN LINES ARE PRINTED TO INDICATE THE PROPER SPACING (IN INCHES) FROM THE PREVIOUS LINE TO THAT LINE.

EXAMPLE:

```
------------------------------

----- 4.0 INCH FORM FEED -----
```

6.1.3 TEST 02 - PRINT SPEED TIMING TEST

THIS TEST IS DESIGNED TO TIME THE LA180 FOR ONE FULL MINUTE WHILE A SWIRL PATTERN IS PRINTED TO THE SELECTED MAXIMUM NUMBER OF COLUMNS. IF A LINE CLOCK OR A PROGRAMMABLE CLOCK OPTION IS DETERMINED TO BE AVAILABLE BY THE PROGRAM, IT WILL BE USED TO AUTOMATICALLY TIME THE PRINTER. WHEN NEITHER CLOCK OPTION IS AVAILABLE, MANUAL TIMING WILL BE USED AND OPERATING INSTRUCTIONS WILL BE TYPED ON THE CONSOLE DEVICE IF IT IS AVAILABLE. WHICHEVER METHOD OF TIMING IS USED, AT THE END OF ONE FULL MINUTE THE APPROXIMATE PRINT SPEED WILL BE PRINTED ON THE LA180 AND ALSO ON THE CONSOLE DEVICE (IF AVAILABLE). REMEMBER, THE PRINT SPEED IS DIRECTLY RELATED TO THE NUMBER OF COLUMNS BEING PRINTED. ALSO, THE CONTENTS OF ONE LOCATION IN MEMORY WILL HAVE TO BE CHANGED IF THE LINE FREQUENCY IS 50 HZ. AND A CLOCK OPTION IS BEING USED FOR TIMING.

6.2 PRINTING TESTS

THESE TESTS ARE DESIGNED AS A TEST OF THE PRINTING MECHANISM AND THE ASSOCIATED CONTROL LOGIC. AT THE BEGINNING OF EACH TEST, A TEST HEADER WILL INDICATE THE TEST NUMBER BEING EXECUTED. THE TEST PROGRAM CONTINUALLY MONITORS FOR PROPER OPERATION OF THE LINE PRINTER AFTER EACH PRINTER OPERATION HAS BEEN COMPLETED, THROUGH THE PRINTER "READY" LINE AND THE SETTING OF THE "DEMAND" FLAG. IT SHOULD BE NOTED, HOWEVER, THAT THE "DEMAND" RETURN FROM THE PRINTER IS CONDITIONAL UPON THE PRINTER "READY". SINCE THE PROCESSOR CAN ONLY DETECT THE CURRENT CONDITION OF THE "READY" AND "DEMAND" RETURN LINES IT IS NECESSARY TO EXAMIME THE PRINT PATTERNS PRODUCED BY THE VARIOUS TEST ROUTINES. EACH PATTERN HAS BEEN CHOSEN FOR EASE OF VISUAL VERIFICATION. DETAILED DESCRIPTIONS OF EACH TEST PATTERN APPEARS IN THE DESCRIPTION OF THE FOLLOWING TEST ROUTINES.

6.2.1 TEST 20 - DATA TRANSFER PATHS TEST

THIS TEST IS DESIGNED TO TEST THE DATA LINES TO AND THROUGH THE INTERFACE AND TO THE LA180 PRINTER. AN ALTERNATING BIT PATTERN IS SENT WHICH WILL PRINT ALTERNATING *'S AND U'S IN A CHECKERBOARD PATTERN TO THE MAXIMUM COLUMN WIDTH. THE STARTING CHARACTER FOR EACH LINE IS ALTERNATED AND A TOTAL OF 16 LINES ARE PRINTED.

EXAMPLE:

```
     *U*U*U*U  .....        *U*U
     U*U*U*U*  .....        U*U*
     *U*U*U*U  .....        *U*U
     U*U*U*U*  .....        U*U*
```

6.2.2 TEST 21 - HEAD POSITIONING TEST

THIS TEST CHECKS THE CARRIAGE RETURN FROM ALL EVEN NUMBERED COLUMNS AND THE SPACING OF THE SOLENOID HEAD FROM THE LEFT MARGIN. HOWEVER, THE PRIMARY PURPOSE OF THIS TEST IS TO TEST THE SOLENOID HEAD POSITION DECODER FOR PROPER OPERATION.

THE TEST PRINTS A FULL LINE OF ALTERNATING 0'S AND SPACES, STARTING WITH A 0. AT THE END OF THE LINE THE PRINT HEAD IS RETURNED TO THE LEFT MARGIN WITH A CARRIAGE RETURN. THE SPACES ARE THEN FILLED IN BY SPACING THE PRINT HEAD OUT FROM THE LEFT MARGIN TO THE FIRST SPACE, PRINTING AN "X", AND EXECUTING A CARRIAGE RETURN. THIS PATTERN IS REPEATED UNTIL THE LINE IS COMPLETED. CHECK TO SEE THAT ALL X'S ARE IN THE MIDDLE OF THE SPACE BETWEEN THE TWO ADJACENT ZEROES.

EXAMPLE:

```
0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X
```

6.2.3 TEST 22 - BACKSPACE TEST

THIS TEST IS DESIGNED TO CHECK THE BACKSPACE FEATURE OF THE LA180 PRINTER. TWO LINES OF X'S INTERSPACED WITH DASHES WILL BE PRINTED BY PRINTING A SLASH, EXECUTING A BACKSPACE, AND THEN PRINTING A BACKSLASH TO COMPLETE EACH X CHARACTER. A MAXIMUM OF 127 COLUMNS WILL BE PRINTED BY THIS TEST.

EXAMPLE:

```
     X-X-X-X-X-X-X-X
     X-X-X-X-X-X-X-X
```

6.2.4 TEST 23 - CHARACTER GENERATOR TEST

THIS TEST CHECKS THE SPACE ALL 94 PRINTABLE CHARACTERS (ASCII CODES 040 TO 176) BY PRINTING A SINGLE LINE, 30 CHARACTERS LONG, OF EACH CHARACTER.

EXAMPLE:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
   .
   .
   .
""#""#""#""#""#""#""#""#""#""#""#""#""#""#""#""
#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$#$
```

6.2.5 TEST 24 - NON-PRINTABLE CHARACTER TEST

THIS TEST IS DESIGNED TO TEST THE LA180 HANDLING OF NON-PRINTABLE CHARACTERS AND TO EXERCISE THE FULL RANGE OF THE CHARACTER STORAGE BUFFER. THE TEST PATTERN PRODUCED WILL BE A 30 LINE SWIRL PATTERN, CONSISTING OF FULL LINES OF THE ENTIRE PRINTABLE CHARACTER SET. IF THIS TEST IS LOOPED ON, THE PATTERN WILL CONTINUE A FULL SWIRL, RATHER THAN ONLY 30 LINES AND THEN REPEATING. AS THE SWIRL PATTERN IS PRODUCED, THE GROUP OF PRINTABLE CHARACTERS WILL BE SHIFTED (IN INCREMENTS DEPENDING ON THE NUMBER OF COLUMNS BEING TESTED) THROUGH THE FULL RANGE OF THE CHARACTER BUFFER, STARTING AT THE END OF THE BUFFER. NON-PRINTABLE CHARACTERS WILL BE USED TO FILL THE CHARACTER BUFFER BEFORE AND AFTER THE GROUP OF PRINTABLE CHARACTERS, FOR EACH PRINTED LINE. ALL NON-PRINTABLE CHARACTERS HAVING NO CONTROL FUNCTION WITHIN THE LA180 WILL BE USED.

EXAMPLE:

```
!"#$%&'()*+,-./0123456789:;<=>?@ABC....
"#$%&'()*+,-./0123456789:;<=>?@ABCD....
#$%&'()*+,-./0122456789:;<=>?@ABCDE....
```

6.2.6 TEST 25 - BUFFER TEST

THIS TEST IS DESIGNED TO TEST THE CHARACTER STORAGE BUFFER IN THE LA180 FOR PROPER OPERATION. THIS TEST WILL PRODUCE FOUR LINES OF PRINT WITH 2 BLANK LINES BETWEEN THE FIRST AND SECOND LINES. THE LINES PRINTED WILL ALSO SERVE AS A CHECK OF PRINTING THE CORRECT COLUMN WIDTH. THE PATTERNS ARE DESCRIBED FOR 132 COLUMNS BUT WILL BE SHORTENED ACCORDINGLY FOR NARROWER TEST WIDTHS. BEFORE THE FIRST LINE IS STORED, 16 E'S WILL BE LOADED INTO THE BUFFER. THEN A RUBOUT (177) WILL BE SENT TO CHECK THAT A RUBOUT WILL CLEAR THE BUFFER. BEFORE EACH OF THE LAST THREE LINES IS PRINTED AND BEFORE THE BLANK LINES BETWEEN THE FIRST AND SECOND PRINTED LINES, THE CHARACTER BUFFER WILL BE FILLED WITH ALL E'S. THUS, AN E PRINTED ANYWHERE IN THE TEST PATTERN INDICATES AN ERROR.

THE FIRST LINE WILL CONTAIN 100 ONES, 30 THREES, AND 2 TWOS. THE SECOND PRINTED LINE WILL CONTAIN 99 ZEROES AND 33 ONES. THE THIRD LINE WILL CONSIST OF THE NUMBERS 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, AND 3 IN GROUPS OF 10 CHARACTERS EACH (EXCEPT THE FIRST GROUP OF ZEROES WILL CONTAIN ONLY 9 CHARACTERS). THE LAST LINE WILL CONTAIN THE NUMBERS 1 TO 9 THEN 0 IN SUCCESSION, REPEATED TO THE MAXIMUM COLUMN.

THUS, THE COLUMN NUMBER MAY BE READ DIRECTLY BY READING THE NUMBERS IN ANY GIVEN COLUMN ON THE LAST THREE LINES, FROM TOP TO BOTTOM.

COLUMN 30 WOULD BE 0
3
0

COLUMN 132 WOULD BE 1
3
2

EXAMPLE:

```
111111111111111111111111111111.....322

000000000000000000000000000000.....111
000000000111111111222222222333.....333
123456789012345678901234567890.....012
```

6.2.7 TEST 26 - OVERPRINT TEST

THIS TEST IS DESIGNED TO CHECK THE SPACING AND REPEATABLE PRINTING CHARACTERISTICS OF THE PRINTER. FOUR LINES OF CHARACTERS ARE EACH OVERPRINTED TWO TIMES. THE ROWS CONSIST OF THE FOLLOWING CHARACTERS ALTERNATED ACROSS THE LINE.

ROW 1 E - SP
ROW 2 SP - a
ROW 3 M - SP
ROW 4 SP - #

THE RESULTING PATTERN WILL BE A CHECKERBOARD PATTERN AND THE OVERPRINTED CHARACTERS SHOULD BE ALIGNED PROPERLY WITH THE INITIAL CHARACTERS.

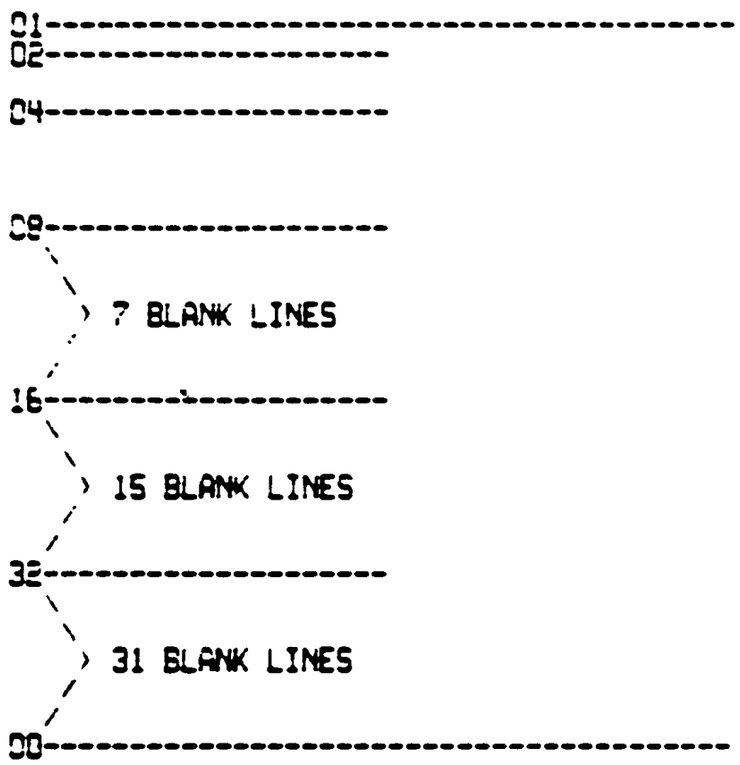
EXAMPLE:

```
E E E E E E E E E E
 a a a a a a a a a a
M M M M M M M M M M
 # # # # # # # # # #
```

6.2.8 TEST 27 - MULTIPLE LINE FEED TEST

THIS TEST CHECKS THE LINE FEED CAPABILITY OF THE PRINTER BY SENDING VARIOUS GROUPS OF LINE FEEDS INTERSPACED WITH REFERENCE LINES. THE NUMBER PRINTED AT THE LEFT MARGIN OF THE REFERENCE LINE INDICATES THE NUMBER OF LINE FEEDS THAT FOLLOW. EACH LINE WILL CONTAIN A STRING OF DASHES AS REFERENCE POINTS FOR MEASURING, THE FIRST AND LAST BEING 132 CHARACTERS LONG (MAXIMUM) AND THE MIDDLE LINES BEING 30 CHARACTERS LONG.

EXAMPLE:



6.2.9 TEST 30 - RIBBON FEED TEST

THIS TEST CHECKS THE RIBBON FEED MECHANISM BY PRINTING A SINGLE COLUMN OF 24 LINES OF X'S DOWN THE LEFT HAND MARGIN OF THE PAGE. VISUALLY CHECK FOR PROPER OPERATION OF THE RIBBON FEED MECHANISM DURING THIS TEST.

EXAMPLE:

X
X
X
.
.
.
X
X
X

6.2.10 TEST 31 - BELL TEST

THIS TEST IS DESIGNED TO CHECK THE BELL CODE LOGIC AND THE TIMING SEQUENCE OF THE MICRO LOGIC. THE TEST WILL PRINT "BELL TEST" INTERSPACED WITH BELL CODES BETWEEN CHARACTERS AND THE FOLLOWING CARRIAGE RETURN AND LINE FEED FUNCTIONS. A TOTAL OF FIVE BELLS WILL BE SOUNDED. THIS TEST WILL ALSO AUDIBLY INDICATE AN END OF A COMPLETE PASS THROUGH THE PRINTING TEST SEQUENCE.

EXAMPLE:

‹BEL› BELL ‹BEL› ‹SP› TEST ‹BEL› ‹CR›

‹BEL› ‹LF› ‹BEL› ‹CR›

6.3 MAINTANANCE AIDS

THESE TESTS ARE PROVIDED AS ADDITIONAL DEBUGGING AND EXERCISING AIDS FOR THE LA180 PRINTER. A DETAILED DESCRIPTION OF EACH TEST FOLLOWS.

6.3.1 TEST 60 - LIFE TEST

THIS TEST RUNS CONTINUOUSLY AND IS RUN AS AN INDIVIDUAL, SPECIAL TEST, AND IS NOT PART OF THE STANDARD PRINTING TEST SEQUENCE. THIS TEST PRINTS 2 LINES OF EACH PRINTABLE CHARACTER AND THEN REPEATS CONTINUOUSLY. THE SECOND LINE OF EACH CHARACTER IS OVERPRINTED 4 TIMES TO CONSERVE PAPER. AT THE COMPLETION OF EACH PASS THROUGH THE ENTIRE PRINTABLE CHARACTER SET, THE PASS COUNT WILL BE PRINTED ON THE LA180.

TIME FOR A COMPLETE PASS, WITH 132 COLUMNS IS APPROXIMATELY 10 MINUTES.

EXAMPLE:

```
AAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBB
```

6.3.2 TEST 61 - SCOPE DRIVE ROUTINE

THE PURPOSE OF THIS TEST IS TO PROVIDE THE OPERATOR WITH A SHORT BUT COMPREHENSIVE SCOPE DRIVER ROUTINE FOR USE IN TROUBLE SHOOTING THE PRINTER AND INTERFACE CONTROL LOGIC WITH AN OSCILLISCOPE.

DEPENDING ON THE SETTING OF THE SINGLE CHAR/FULL LINE SWITCH OF THE SWITCH REGISTER (SWITCH 09) THIS TEST WILL EITHER CONTINUALLY SEND WHATEVER CHARACTER IS SET IN THE SWITCH REGISTER TO THE LINE PRINTER, OR ONLY SEND IT ONCE AND HALT. WHEN CONTINUOUSLY SENDING CHARACTERS, A LINE FEED WILL BE INSERTED AFTER THE MAXIMUM COLUMN COUNT IS REACHED TO PRINT THE LINE. WHEN SENDING SINGLE CHARACTERS, DEPRESS CONTINUE TO SEND THE CHARACTER SET IN THE SWITCH REGISTER. TO RESUME SENDING CONTINUOUS CHARACTERS, PLACE THE SINGLE CHAR/FULL LINE CONTROL SWITCH DOWN, SET THE DESIRED CHARACTER, AND DEPRESS CONTINUE. TO STOP SENDING CONTINUOUSLY PLACE THE SINGLE CHAR/FULL LINE SWITCH UP AND THE PROGRAM WILL HALT WAITING FOR A CHARACTER SELECTION. WHEN SENDING INDIVIDUAL CHARACTERS OR IF SENDING NON-PRINTABLE CHARACTERS, NO LINE FEEDS OR CARRIAGE RETURNS WILL BE INSERTED BY THE PROGRAM.

6.3.3 TEST 62 - LINE PRINT TEST

THIS TEST CONTINUOUSLY PRINTS FULL LINES OF WHATEVER CHARACTER IS TYPED ON THE CONSOLE KEYBOARD. TO CHANGE CHARACTERS, RESELECT THIS TEST AND TYPE ANOTHER CHARACTER. AN ERROR MESSAGE WILL BE PRINTED ON THE LA180 IF THIS TEST IS SELECTED AND A CONSOLE TERMINAL DOES NOT EXIST.

6.3.4 TEST 63 - CHARACTER PRINT TEST

THIS TEST LOADS WHATEVER CHARACTER IS TYPED ON THE CONSOLE KEYBOARD TO THE LA180, CHARACTER BY CHARACTER ALL TYPED CHARACTERS ARE ECHOED TO THE CONSOLE DEVICE AS THEY ARE LOADED TO THE LA180. EXTRA CARRIAGE RETURNS OR LINE FEEDS ARE ECHOED TO THE CONSOLE DEVICE TO AVOID OVERPRINTING LINES. IF THIS TEST IS SELECTED AND A CONSOLE TERMINAL DOES NOT EXIST AN ERROR MESSAGE WILL BE PRINTED ON THE LA180.

H02

17	OPERATIONAL SWITCH SETTINGS
41	BASIC DEFINITIONS
152	TRAP CATCHER
164	ACT11 HOOKS
190	STARTING ADDRESSES
206	COMMON TAGS
272	PROGRAM INITIALIZATION
366	OPERATOR INTERVENTION TESTS
715	PRINTING TESTS
1114	END OF PASS ROUTINE
1145	MAINTENANCE AIDS
1291	CLOCK INTERRUPT SERVICE ROUTINES
1325	TEST EXIT ROUTINE
1351	ROUTINES TO SELECT DESIRED TEST
1564	ERROR HANDLER ROUTINE
1629	SAVE AND RESTORE R0-R5 ROUTINES
1675	TYPE ROUTINE
1718	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
1786	BINARY TO OCTAL (ASCII) AND TYPE
1864	ROUTINES TO LOAD CHARACTERS TO LA180
1980	PRINT TEST HEADER
2014	TTY INPUT ROUTINE
2092	READ A DECIMAL NUMBER FROM THE TTY
2153	DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
2217	SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
2235	DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
2275	SINGLE LENGTH BINARY TO OCTAL ASCIZ ROUTINE
2292	TRAP DECODER
2316	TRAP TABLE
2344	POWER DOWN AND UP ROUTINES
2381	TEST ADDRESS TABLE
2458	ERROR MESSAGE ADDRESS TABLE
2486	ERROR MESSAGES
2603	PROGRAM MESSAGES

```
                        .ENABLE ABS,AMA

                        .MCALL  .SWRHI,.EQUAT,.$CATCH,.$EOP,SETPRI
                        .MCALL  .$TRAP,.$TYPOCT,.$TYPDEC,.$POWER,.HEADER
                        .MCALL  .$DB2D,.$SB2D,.$DB20,.$SB20,.$READ
                        .MCALL  TYPNAM,.SETUP,TRMTRP,.$RDDEC,.$SAVE,.$ACT1.

                    .TITLE  MAINDEC-11-DZLAE-A
                    ;*COPYRIGHT (C) 1975
                    ;*DIGITAL EQUIPMENT CORP.
                    ;*MAYNARD, MASS. 01754
                    ;*
                    ;*PROGRAM BY ROBERT BAKER
                    ;*
                    ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                    ;*PACKAGE (MAINDEC-11-DZQAC-B1),AUG 29,1975.
                    ;*


                    .SBTTL  OPERATIONAL SWITCH SETTINGS
                    ;*
                    ;*      SWITCH                  USE
                    ;*      ------          --------------------
                    ;*        15            HALT ON ERROR
                    ;*        14            LOOP ON TEST
                    ;*        13            INHIBIT ERROR TYPEOUTS
                    ;*        12            MANUAL TIMING
                    ;*        10            BELL ON ERROR
                    ;*         9            SNGL CHAR/FULL LINE - SCOPE ROUTINE
                    ;*         8            HALT & SELECT TEST

                    ;*      07--00          # COLUMNS AT START UP

                    ;*      05--00          TEST # SELECTION

                    ;*      06--00          CHAR SELECTION FOR SCOPE ROUTINE


                    ;***  SET ALL SWITCHES UP BEFORE STARTING THE PROGRAM TO USE THE
                    ;     SOFTWARE SWITCH REGISTER CONTROL.  MAKE SURE LOCATION 000176
                    ;     CONTAINS THE DESIRED SWITCH SETTINGS BEFORE STARTING.
```

```
                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

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

                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
000300                          PR6=    300             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
000004                          SW02=   4
000002                          SW01=   2
000001                          SW00=   1
                                .EQUIV  SW09,SW9
```

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

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                          BIT15=  100000
040000                          BIT14=  40000
020000                          BIT13=  20000
010000                          BIT12=  10000
004000                          BIT11=  4000
002000                          BIT10=  2000
001000                          BIT09=  1000
000400                          BIT08=  400
000200                          BIT07=  200
000100                          BIT06=  100
000040                          BIT05=  40
000020                          BIT04=  20
000010                          BIT03=  10
000004                          BIT02=  4
000002                          BIT01=  2
000001                          BIT00=  1
                                .EQUIV  BIT09,BIT9
                                .EQUIV  BIT08,BIT8
                                .EQUIV  BIT07,BIT7
                                .EQUIV  BIT06,BIT6
                                .EQUIV  BIT05,BIT5
                                .EQUIV  BIT04,BIT4
                                .EQUIV  BIT03,BIT3
                                .EQUIV  BIT02,BIT2
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                          TBITVEC=14              ;;"T" BIT
000014                          TRTVEC= 14              ;;TRACE TRAP
000014                          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                          PWRVEC= 24              ;;POWER FAIL
000030                          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                          TRAPVEC=34              ;;"TRAP" TRAP
000060                          TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                          TPVEC=  64              ;;TTY PRINTER VECTOR
000240                          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
                        000015          CR=15
                        000012          LF=12
                        000014          FF=14
                        000200          CRLF=200
                                        ;*****************************************************************
                                        .SBTTL  TRAP CATCHER
                                                .=0
                        000000          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                  .=174
000174  000000                          DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                          SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                        ;*****************************************************************
                                        .SBTTL  ACT11 HOOKS
                                        ;HOOKS REQUIRED BY ACT11
                        000200                  $SVPC=.                 ;SAVE PC
                        000046                  .=46
000046  004756                          $ENDAD                          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                        000052                  .=52
000052  020000                          .WORD   20000                   ;;2)SET LOC.52 TO 20000
                        000200                  .=$SVPC                 ;; RESTORE PC
```

```
                                ;*****************************************************************

                        000030          .=30

000030  006572                  $ERROR                  ;EMT - ERROR HANDLER
000032  000000                  0

                        000100          .=100

000100  005440                  LKSRV                   ;KW11-L  INTERRUPT
000102  000000                  0

000104  005430                  DCI                     ;KW11-P  INTERRUPT
000106  000000                  0


                                ;*****************************************************************

                                .SBTTL  STARTING ADDRESSES

                        000200          .=200

000200  000137  001154          JMP     START           ;GENERAL START

                        000600          .=600

000600  000137  001170          JMP     RESTRT          ;RESTART - NO INTERACTIVE TESTS

000604  000137  001200          JMP     CNTRL           ;GO TO KEYBOARD CONTROL
                                                        ;WILL DEFAULT TO SWITCH REG
                                                        ;IF CONSOLE DEVICE DOES NOT EXIST


                                ;*****************************************************************

                                .SBTTL  COMMON TPGS

                        001100          .=1100

001100  177560          TKS:    177560                  ;TTY KYBD STATUS REG. ADDRESS
001102  177562          TKB:    177562                  ;TTY KYBD BUFFER REG. ADDRESS
001104  177564          TPS:    177564                  ;TTY PRINTER STATUS REG. ADDRESS
001106  177566          TPB:    177566                  ;TTY PRINTER BUFFER REG. ADDRESS


                                        .EQUIV  TKS,$TKS
                                        .EQUIV  TKB,$TKB
                                        .EQUIV  TPS,$TPS
                                        .EQUIV  TPB,$TPB

001110  177514          LPS:    177514                  ;LINE PRINTER STATUS REG. ADDRESS
                                                        ;BIT 15 = ERROR
                                                        ;BIT  7 = READY
                                                        ;BIT  6 = INTERRUPT ENABLE
```

```
001112  177516          LPB:    177516          ;LINE PRINTER DATA BUFFER REG. ADDRESS
                                                ;BITS 0-6 = ASCII CHAR BUFFER
                                                ;BITS 7-15 = NOT USED

001114  172540          PLKS:   172540          ;KL11-P CLOCK STATUS REG. ADDRESS
001116  172542          CSBR:   172542          ;KL11-P COUNT SET ADDRESS

001120  177546          LKS:    177546          ;KW11-L CLOCK STATUS REG. ADDRESS

001122  177570          SWR:    .WORD   177570  ;SW REG ADDRESS
001124  177570          DISPLAY: .WORD  177570  ;DISPLAY ADR

001126  000000          $TSTNM: .WORD   0       ;TEST NUMBER
001130  000204          WIDTH:  .WORD   132.    ;NUMBER OF COLUMNS
001132     000          STRONE: .BYTE   0       ;RUN TEST ONCE FLAG (SW REG CTL)
001133     000          TRONE:  .BYTE   0       ;RUN TEST ONCE FLAG (KYBD CTL)
001134     000          TLOOP:  .BYTE   0       ;LOOP ON TEST FLAG (KYBD CTL)

001135     000          CKFLAG: .BYTE   0       ;CLOCK OPTION FLAG
                                                ;0 = NONE AVAILABLE
                                                ;+1 = KL11-L
                                                ;-1 = KL11-P

001136  000207          $BELL:  .ASCIZ  <207>   ;BELL CODE
001140     077          $QUES:  .ASCII  /?/     ;QUESTION MARK
001141     015          $CRLF:  .ASCII  <15>    ;CARRIAGE RETURN - LINE FEED
001142  000012          $LF:    .ASCIZ  <12>    ;LINE FEED

001144  000015          $CR:    .ASCIZ  <15>    ;CARRIAGE RETURN ONLY
001146  000014          $FF:    .ASCIZ  <14>    ;FORM FEED

                        .EVEN

                        ;*****************************************************************
                        ;THE FOLLOWING LOCATIONS ARE USED BY THE TTY TYPE ROUTINES
                        ;SET THE FILL CHARACTERS AS REQUIRED FOR VARIOUS CONSOLE TERMINALS
                        ;THE TERMINAL AVAILABLE FLAG WILL BE SET BY THE PROGRAM.

001150     000          $NULL:  .BYTE   0       ;NULL CHARACTER FOR FILLS
001151     002          $FILLS: .BYTE   2       ;NUMBER OF FILLER CHARACTERS REQUIRED
001152     012          $FILLC: .BYTE   12      ;INSERT FILL CHARACTERS AFTER LINE FEED
001153     000          $TPFLG: .BYTE   0       ;TERMINAL AVAILABLE FLAG
                                                ;BIT <07> = 0 = YES
                        .EVEN
```

```
                                ;*****************************************************************
                                .SBTTL  PROGRAM INITIALIZATION
001154  005037  001126          START:  CLR     $TSTNM          ;SET TEST NUMBER TO ZERO
001160  023737  000042  000046          CMP     @#42,@#46       ;CHECK IF IN ACT QUICK VERIFY
                                                                ;SKIP MANUAL INTERVENTION TESTS IF YES
001166  001007                          BNE     STARTX          ;INITIALIZE
001170  012737  000020  001126  RESTRT: MOV     #20,$TSTNM      ;SET TEST NUMBER TO 20
001176  000403                          BR      STARTX          ;INITIALIZE
001200  013737  177777  001126  CONTRL: MOV     #-1,$TSTNM      ;SET CONTROL FLAG
001206  012737  177570  001122  STARTX: MOV     #177570,SWR     ;INITIALIZE SWR CONTROL
001214  013737  001122  001132          MOV     SWR,DISPLAY
001222  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
001226  012737  011320  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001334  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001342  012737  011434  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001350  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001356  005037  004772                  CLR     $PASS           ;;CLEAR THE PASS COUNT
001362  013737  004742  004734          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001370  013746  000004                  MOV     @#4,-(SP)       ;;SAVE ERROR VECTOR
001374  013746  000006                  MOV     @#6,-(SP)
001300  012737  001314  000004          MOV     #64$,4          ;;SET UP TIME OUT VECTOR
001306  005777  177610                  TST     @SWR            ;;TRY TO REFERENCE HARDWARE SWR
001312  000407                          BR      65$             ;;BRANCH IF NO TIMEOUT TRAP OCCURS
001314  012737  000176  001122  64$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001322  012737  000174  001124          MOV     #DISPREG,DISPLAY ;;POINT TO SOFTWARE DISPLAY REG
001330  022626                          CMP     (SP)+,(SP)+     ;;RESTORE STACK
001332  012637  000006          65$:    MOV     (SP)+,@#6       ;;RESTORE ERROR VECTOR
001336  012637  000004                  MOV     (SP)+,@#4
001342  017700  177554                  MOV     @SWR,R0         ;GET SWITCHES
001346  005200                          INC     R0              ;CHECK IF ALL SWITCHES ARE UP
001350  001006                          BNE     6$              ;CONTINUE IF NOT
001352  012737  000176  001122          MOV     #SWREG,SWR      ;IF UP, SET SOFTWARE SWITCH CONTROL
001360  012737  000174  001124          MOV     #DISPREG,DISPLAY
001366  017700  177530          6$:     MOV     @SWR,R0         ;GET SW REG
001372  042700  177400                  BIC     #177400,R0      ;SAVE BITS 0-7
001376  020027  000204                  CMP     R0,#132.        ;TEST # COLUMNS
001402  003003                          BGT     2$              ;TOO BIG, DEFAULT TO 132(10)
001404  020027  000002          1$:     CMP     R0,#2.          ;TEST # COLUMNS
001410  103002                          BHIS    3$              ;BRANCH IF OK
001412  012700  000204          2$:     MOV     #132.,R0        ;DEFAULT TO 132(10)
001416  010037  001130          3$:     MOV     R0,WIDTH        ;SAVE # COLUMNS
001422  105037  001134                  CLRB    TLOOP           ;RESET FLAGS
001426  105037  001132                  CLRB    STRONE
001432  105037  001133                  CLRB    TRONE
001436  005037  010242                  CLR     SVTST
001442  000401                          BR      10$             ;REPLACE THIS INSTRUCTION WITH NOP (204)
                                                                ;TO SKIP CLOCK TIMINGS
001444  000424                          BR      20$
001446  012737  000002  000006  10$:    MOV     #RTI,@#6        ;SET TRAP RETURN
001454  012737  000006  000004          MOV     #6,@#4
001462  112737  177777  001135          MOVB    #-1,CKFLAG      ;SET CLOCK FLAG FOR KW11-P
001470  000261                          SEC
```

```
327  001472  105777  177416                    TSTB    @PLKS           ;KW11-P AVAILABLE?
328  001476  103011                            BCC     30$             ;YES, CHECK FOR CONSOLE TERMINAL
329  001500  112737  000001  001135            MOVB    #1,CKFLAG       ;SET CLOCK FLAG FOR KW11-L
330  001506  000261                            SEC
331  001510  105777  177404                    TSTB    @LKS            ;KW11-L AVAILABLE?
332  001514  103002                            BCC     30$             ;YES, CHECK FOR CONSOLE TERMINAL
333  001516  105037  001135          20$:      CLRB    CKFLAG          ;CLEAR CLOCK AVAILABLE FLAG - NONE THERE
334  001522  112737  177777  001153  30$:      MOVB    #-1,$TPFLG      ;CLEAR TERMINAL AVAILABLE FLAG
335  001530  000261                            SEC
336  001532  105777  177342                    TSTB    @TKS            ;CONSOLE TERMINAL THERE?
337  001536  103402                            BCS     4$              ;NO, CONTINUE
338  001540  105037  001153                    CLRB    $TPFLG          ;YES, SET TERMINAL AVAILABLE FLAG
339  001544  103002                  4$:       BCC     5$              ;CONTINUE IF CONSOLE
340  001546  104413  013433                    PRINT   #NOMSG          ;PRINT NO CONSOLE MESSAGE
341  001552  005037  000006          5$:       CLR     @#6             ;RESET TRAP VECTOR HALTS
342  001556  005037  000004                    CLR     @#4
343  001562  005227  177777                    INC     #-1             ;;FIRST TIME?
344  001566  001022                            BNE     66$             ;;BRANCH IF NO
345  001570  022737  004756  000042            CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
346  001576  001416                            BEQ     66$             ;;BRANCH IF NO
347  001600  104400  001606                    TYPE    ,67$            ;;TYPE ASCIZ STRING
348  001604  000413                            BR      66$             ;;GET OVER THE ASCIZ
349                                  ;;67$:    .ASCIZ  <200>#MAINDEC-11-DZLAE-A#<200>
350  001634                          66$:
351  001634  005737  001126                    TST     $TSTNM          ;WANT CONTROL NOW?
352  001640  002005                            BGE     11$             ;NO, CONTINUE
353  001642  105737  001153          15$:      TSTB    $TPFLG          ;TERMINAL THERE?
354  001646  001006                            BNE     13$             ;NO, DEFAULT TO SW REG CONTROL
355  001650  000137  006270                    JMP     KYBDST          ;YES, GO TO KYBD CONTROL
356  001654  032777  000400  177240  11$:      BIT     #SW8,@SWR       ;WANT TEST SELECTION?
357  001662  001402                            BEQ     12$             ;NO, CHECK TEST #
358  001664  000137  005624          13$:      JMP     SELECT          ;YES, GO TO TEST SELECTION HALT
359  001670  013700  001126          12$:      MOV     $TSTNM,R0       ;GET TEST NUMBER
360  001674  006300                            ASL     R0              ;SET POINTER
361  001676  005760  011564                    TST     TAT(R0)         ;CHECK IF TEST IN TABLE
362  001702  003004                            BGT     14$             ;BRANCH IF IN TABLE
363  001704  002756                            BLT     15$             ;END OF SEQUENCE, SELECT TEST
364  001706  005237  001126                    INC     $TSTNM          ;INCREMENT TEST NUMBER
365  001712  000760                            BR      11$             ;CHECK NEXT TEST NUMBER IN TABLE
366  001714  000170  011564          14$:      JMP     @TAT(R0)        ;GO TO TEST
```

```
                                ;*****************************************************************

                                .SBTTL  OPERATOR INTERVENTION TESTS

                                ;/////////////////////////////////////////////////////////////////

                                ;TESTO  -       INTERFACE AND CONTROL TESTS
                                ;
                                ;TEST ERROR AND READY BITS, PRINTER OFF LINE - POWER OFF

                                ;/////////////////////////////////////////////////////////////////

001720  104400  013722        TESTO:  TYPE,   TOMSGO          ;TYPE INSTRUCTIONS
001724  104420                        HOLD                    ;WAIT FOR OPERATOR
001726  104417                28$:    CHECK                   ;CHECK FOR CONTROL
001730  005777  177154                TST     @LPS            ;CHECK FOR ERROR CONDITION
001734  100402                        BMI     1$              ;OK, ERROR SET
001736  104001                        ERROR   1               ;ERROR CLEAR, POWER OFF
001740  000772                        BR      28$             ;RETEST
001742  105777  177142        1$:     TSTB    @LPS            ;CHECK READY
001746  100002                        BPL     2$              ;OK, READY NOT SET
001750  104002                        ERROR   2               ;READY SET, POWER OFF
001752  000765                        BR      28$             ;RETEST

                                ;/////////////////////////////////////////////////////////////////

                                ;TEST ERROR AND READY BITS, PRINTER OFF LINE - POWER ON

                                ;/////////////////////////////////////////////////////////////////

001754  104400  013761        2$:     TYPE,   TOMSG1          ;TYPE INSTRUCTION - TURN POWER ON
001760  104420                        HOLD                    ;WAIT FOR OPERATOR
001762  104417                3$:     CHECK                   ;CHECK FOR CONTROL
001764  005777  177120                TST     @LPS            ;CHECK ERROR
001770  100402                        BMI     4$              ;OK, ERROR SET
001772  104003                        ERROR   3               ;ERROR CLEAR, PRINTER OFF LINE
001774  000772                        BR      3$              ;RETEST
001776  105777  177106        4$:     TSTB    @LPS            ;CHECK READY
002002  100002                        BPL     5$              ;OK, READY NOT SET
002004  104004                        ERROR   4               ;READY SET, PRINTER OFF LINE
002006  000765                        BR      3$              ;RETEST

                                ;/////////////////////////////////////////////////////////////////

                                ;TEST ERROR AND READY BITS, PRINTER ON LINE

                                ;/////////////////////////////////////////////////////////////////

002010  104400  014004        5$:     TYPE    ,TOMSG2         ;TYPE INSTRUCTION, TURN ON LINE
002014  104420                        HOLD                    ;WAIT FOR OPERATOR
002016  104417                6$:     CHECK                   ;CHECK FOR CONTROL
002020  005777  177064                TST     @LPS            ;CHECK ERROR
002024  100002                        BPL     7$              ;OK, ERROR CLEAR
002026  104005                        ERROR   5               ;ERROR SET, PRINTER ON LINE
```

```
                                ;////////////////////////////////////////////////////////////
                                ;TEST PAPER OUT SWITCH
                                ;////////////////////////////////////////////////////////////
002044  104400  014040    8$:   TYPE    .TOMSG3         ;TYPE INSTRUCTION, PAPER OUT
002050  104420                  HOLD                    ;WAIT FOR OPERATOR
002052  104417            9$:   CHECK                   ;CHECK CONTROL
002054  012777  000012  177030  MOV     #12,@LPB        ;SEND LF
002062  005777  177022          TST     @LPS            ;CHECK FOR ERROR CONDITION
002066  100402                  BMI     10$             ;OK, ERROR SET
002070  104007                  ERROR   7               ;ERROR CLEAR, PAPER OUT ERROR
002072  000767                  BR      9$              ;RETEST
002074  105777  177010    10$:  TSTB    @LPS            ;CHECK READY
002100  100002                  BPL     11$             ;OK, READY CLEAR
002102  104010                  ERROR   10              ;READY SET, PAPER OUT, ON LINE
002104  000762                  BR      9$              ;RETEST

                                ;////////////////////////////////////////////////////////////
                                ;TEST ABILITY TO CLEAR ERROR CONDITION
                                ;////////////////////////////////////////////////////////////
002106  104400  014072    11$:  TYPE    ,TOMSG4         ;TYPE INSTRUCTION, RESET 3 ON LINE
002112  104420                  HOLD                    ;WAIT FOR OPERATOR
002114  104417            12$:  CHECK                   ;CHECK FOR CONTROL
002116  005777  176766          TST     @LPS            ;CHECK ERROR
002122  100002                  BPL     13$             ;OK, ERROR CLEAR
002124  104011                  ERROR   11              ;ERROR DID NOT CLEAR
002126  000772                  BR      12$             ;RETEST
002130  105777  176754    13$:  TSTB    @LPS            ;CHECK READY
002134  100402                  BMI     14$             ;OK, READY SET
002136  104012                  ERROR   12              ;READY NOT SET
002140  000773                  BR      13$             ;RETEST

                                ;////////////////////////////////////////////////////////////
                                ;CHECK ERROR 3 READY BITS AFTER RESET INSTRUCTION
                                ;////////////////////////////////////////////////////////////
002142  104417            14$:  CHECK                   ;CHECK CONTROL
002144  000005                  RESET                   ;CLEAR WORLD
002146  005777  176736          TST     @LPS            ;CHECK ERROR
002152  100002                  BPL     15$             ;OK, ERROR CLEAR
002154  104013                  ERROR   13              ;ERROR BIT SET AFTER RESET INSTR.
```

```
002156  000771                          BR      14$             ;RETEST
002160  104417                  15$:    CHECK                   ;CHECK CONTROL
002162  000005                          RESET                   ;CLEAR WORLD
002164  105777  176720                  TSTB    @LPS            ;CHECK READY
002170  100402                          BMI     16$             ;OK, READY SET
002172  104014                          ERROR   14              ;READY BIT CLEAR AFTER RESET INSTR.
002174  000771                          BR      15$             ;RETEST

                                ;//////////////////////////////////////////////////////////////////////

                                ;CHECK THAT LOADING CHAR BUFFER RESETS READY BIT
                                ;AND PRINTER DOES GO BACK READY.

                                ;//////////////////////////////////////////////////////////////////////

002176  104417                  16$:    CHECK                   ;CHECK CONTROL
002200  005005                          CLR     R5              ;CLEAR TIME OUT COUNTER
002202  012777  000015  176702          MOV     #15,@LPB        ;LOAD CARRIAGE RETURN INTO BUFFER
002210  105777  176674                  TSTB    @LPS            ;CHECK READY
002214  100002                          BPL     17$             ;OK, READY CLEAR
002216  104015                          ERROR   15              ;READY BIT NOT CLEAR WHEN CHAR LOADED
002220  000766                          BR      16$             ;RETEST
002222  005777  176662          17$:    TST     @LPS            ;CHECK ERROR
002226  100002                          BPL     18$             ;OK, ERROR CLEAR
002230  104016                          ERROR   16              ;ERROR BIT SET AFTER CHAR LOAD
002232  000761                          BR      16$             ;RETEST
002234  105777  176650          18$:    TSTB    @LPS            ;PRINTER STILL BUSY
002240  100404                          BMI     19$             ;NO, NEXT TEST
002242  005205                          INC     R5              ;YES, INC TIMER
002244  001366                          BNE     17$             ;WAIT FOR FLAG
002246  104017                          ERROR   17              ;NO, TOO LONG
002250  000752                          BR      16$             ;RETEST

                                ;//////////////////////////////////////////////////////////////////////

                                ;CHECK INTERRUPT LEVEL OF PRINTER
                                ;PRINTER SHOULD BE AT LEVEL 4
                                ;
                                ;TEST THAT PRINTER WILL NOT INTERRUPT ABOVE LEVEL 3

                                ;//////////////////////////////////////////////////////////////////////

002252  012705  000340          19$:    MOV     #PR7,R5         ;SET FIRST LEVEL
002256  104417                  20$:    CHECK                   ;CHECK CONTROL
002260  012737  002362  000200          MOV     #23$,200        ;SET INTERRUPT RETURN
002266  012737  000340  000202          MOV     #PR7,202        ;SET PRIORITY 7 ON INTERRUPT
002274  005777  176610                  TST     @LPS            ;CHECK FOR ERROR
002300  100002                          BPL     21$             ;OK, ERROR CLEAR
002302  104020                          ERROR   20              ;ERROR BIT SET
002304  000764                          BR      20$             ;RETEST
002306  105777  176576          21$:    TSTB    @LPS            ;CHECK READY
002312  100402                          BMI     22$             ;OK, READY SET
002314  104021                          ERROR   21              ;READY BIT NOT SET
002316  000757                          BR      20$             ;RETEST
```

```
                                        22$:
002320  010546                                  MOV     R5,-(SP)        ;;PUT NEW PS ON STACK
002322  012746  002330                          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
002326  000002                                  RTI                     ;; POP NEW PC AND PS
002330                                  64$:
002330  052777  000100  176552                  BIS     #BIT6,@LPS      ;SET PRINTER INT. ENABLE
002336  000240                                  NOP                     ;DELAY
002340  042777  000100  176542                  BIC     #BIT6,@LPS      ;CLEAR PRINTER INT. ENABLE
002346  162705  000040                          SUB     #40,R5          ;SET NEXT LEVEL
002352  020527  000140                          CMP     R5,#PR3         ;LEVEL 3?
002356  001404                                  BEQ     24$             ;YES, CONTINUE NEXT TEST
002360  000736                                  BR      20$             ;NO, TEST THIS LEVEL

002362  022626                          23$:    CMP     (SP)+,(SP)+     ;RESTORE STACK
002364  104022                                  ERROR   22              ;INTERRUPT ABOVE LEVEL 3
002366  000733                                  BR      20$             ;RETEST

                                        ;//////////////////////////////////////////////////////////////////////////

                                        ;TEST ABILITY OF PRINTER TO INTERRUPT AT ALL PRIORITY LEVELS BELOW 4

                                        ;//////////////////////////////////////////////////////////////////////////

002370  104417                          24$:    CHECK                   ;CHECK FOR CONTROL
002372  012737  002456  000200                  MOV     #27$,200        ;SET INTERRUPT RETURN
002400  005777  176504                          TST     @LPS            ;CHECK FOR ERROR
002404  100002                                  BPL     25$             ;Ok, ERROR CLEAR
002406  104020                                  ERROR   20              ;ERROR BIT SET
002410  000767                                  BR      24$             ;RETEST
002412  105777  176472                  25$:    TSTB    @LPS            ;CHECK READY
002416  100402                                  BMI     26$             ;OK, READY SET
002420  104021                                  ERROR   21              ;READY CLEAR
002422  000762                                  BR      24$             ;RETEST
002424                                  26$:
002424  010546                                  MOV     R5,-(SP)        ;;PUT NEW PS ON STACK
002426  012746  002434                          MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
002432  000002                                  RTI                     ;; POP NEW PC AND PS
002434                                  65$:
002434  052777  000100  176446                  BIS     #BIT6,@LPS      ;SET PRINTER INTR. ENABLE
002442  000240                                  NOP                     ;DELAY
002444  042777  000100  176436                  BIC     #BIT6,@LPS      ;CLEAR PRINTER INTR. ENABLE
002452  104023                                  ERROR   23              ;NO INTERRUPT BELOW LEVEL 4
002454  000745                                  BR      24$             ;RETEST

002456  042777  000100  176424  27$:    BIC     #BIT6,@LPS      ;CLEAR PRINTER INTR. ENABLE
002464  022626                                  CMP     (SP)+,(SP)+     ;RESET STACK
002466  162705  000040                          SUB     #40,R5          ;SET NEXT LEVEL
002472  002336                                  BGE     24$             ;RETEST IF NOT DONE ALL LEVELS
002474  012737  000137  000200                  MOV     #137,200        ;RESET INSTRUCTIONS AT 200-202
002502  012737  001154  000202                  MOV     #START,202
002510  012746  000000                          MOV     #0,-(SP)        ;;PUT NEW PS ON STACK
002514  012746  002522                          MOV     #66$,-(SP)      ;;PUT NEW PC ON  STACK
002520  000002                                  RTI                     ;; POP NEW PC AND PS
002522                                  66$:
```

```
   583  002522  000137  005516                  JMP     EXIT            ;EXIT TEST
```

```
                                        ;////. .////////////////////////////////////////////////////////////
                                        ;
                                        ;TEST1  -       TOP OF FORM SWITCH TEST
                                        ;
                                        ;////////////.////////////////////////:////////////////////////////
                                        ;
002526  104412                  TEST1:  PRTHDR                  ;PRINT TEST HEADER
002530  012705  002640                  MOV     #4$,R5          ;SET TABLE POINTER
002534  012701  000036                  MOV     #30.,R1         ;SET CHAR COUNT
002540  012700  000055                  MOV     #55,R0          ;SET DASH CHAR
002544  104415                          MLOAD                   ;LOAD DASHED LINE
002546  104413  001144                  PRINT   ,$CR            ;PRINT LINE
002552  104400  014170          3$:     TYPE    ,T1MSG3         ;TYPE INSTRUCTIONS
002556  010537  002564                  MOV     R5,1$           ;SET SWITCH SETTING FOR MESG
002562  104400                          TYPE
002564  000000          1$:     .WORD   0
002566  104400  014222                  TYPE    ,T1MSG4         ;FINISH INSTRUCTIONS
002572  104420                          HOLD                    ;WAIT FOR OPERATOR
002574  104417                          CHECK                   ;CHECK FOR CONTROL
002576  104413  001146                  PRINT   ,$FF            ;ISSUE FORM FEED
002602  104413  014132                  PRINT   ,T1MSG1         ;PRINT REFERENCE LINE
002606  010537  002614                  MOV     R5,2$           ;SET FORM FEED LENGTH FOR MESG
002612  104413                          PRINT
002614  000000          2$:     .WORD   0
002616  104413  014141                  PRINT   ,T1MSG2         ;FINISH MESG
002622  022525                          CMP     (R5)+,(R5)+     ;INC TABLE POINTER
002624  005715                          TST     (R5)            ;CHECK TABLE TO SEE IF DONE TEST
002626  001351                          BNE     3$              ;FINISH TEST
002630  104413  001142                  PRINT   ,$LF            ;ADVANCE PAPER WHEN DONE
002634  000137  005516                  JMP     EXIT            ;EXIT TEST

002640  031440  000040          4$:     .ASCIZ  / 3 /           ;FORM FEED SWITCH SETTINGS
002644  027063  000065                  .ASCIZ  /3.5/
002650  032040  000040                  .ASCIZ  / 4 /
002654  027065  000065                  .ASCIZ  /5.5/
002660  033040  000040                  .ASCIZ  / 6 /
002664  033440  000040                  .ASCIZ  / 7 /
002670  034040  000040                  .ASCIZ  / 8 /
002674  027070  000065                  .ASCIZ  /8.5/
002700  030461  000040                  .ASCIZ  /11 /
002704  031061  000040                  .ASCIZ  /12 /
002710  032061  000040                  .ASCIZ  /14 /
002714  000000                          .WORD   0               ;END OF TABLE
```

```
                                    ;//////////////////////////////////////////////////////////////////////
                                    ;
                                    ;TEST2  -       PRINT SPEED TIMING
                                    ;
                                    ;A SWIRL PATTERN IS PRINTED FOR ONE FULL MINUTE
                                    ;WHILE THE NUMBER OF LINES PRINTED IS COUNTED.
                                    ;TIMING WILL BE DONE BY KW11-L/KW11-P CLOCK OTION IF EITHER AVAILABLE.
                                    ;OTHERWISE, MANUAL TIMING WILL BE USED TO OBTAIN APPROX. PRINT TIMINGS.
                                    ;
                                    ;IF A HARDWARE SWITCH REGISTER IS NOT AVAILABLE, THIS TEST
                                    ;CANNOT BE RUN WITHOUT A CLOCK OPTION BEING AVAILABLE.  THE
                                    ;PROGRAM WILL AUTOMATICALLY SKIP THIS TEST IF IT CANNOT BE RUN.
                                    ;
                                    ;//////////////////////////////////////////////////////////////////////

002716  104412                      TEST2:  PRTHDR                  ;PRINT TEST HEADER
002720  012737  003134  005462              MOV     #SPD,TORTN      ;SET TIME OUT RETURN
002726  005004                      1$:     CLR     R4              ;CLEAR LINE COUNT
002730  105737  001135              2$:     TSTB    CKFLAG          ;TEST CLOCK FLAG
002734  001417                              BEQ     4$              ;NONE THERE, MANUAL TIMING
002736  002407                              BLT     1$              ;KW11-P
002740  013737  005464  005460              MOV     MINCNT,CNTR     ;SET KW11-L CLOCK COUNT
002746  012777  000100  176144              MOV     #BIT6,@LKS      ;ENABLE CLOCK INTERRUPT
002754  000427                              BR      T2SP            ;START PRINTING
002756  013777  005464  176132  1$:     MOV     MINCNT,@CSBR    ;SET CLOCK COUNT
002764  012777  000105  176122              MOV     #105,@PLKS      ;START CLOCK
002772  000420                              BR      T2SP            ;START PRINTING
002774  023727  001122  177570  4$:     CMP     SWR,#177570     ;CLECK SW REG ADR
003002  001406                              BEQ     2$
003004  104413  014267                      PRINT   ,T2EM           ;CONTINUE IF HARDWARE
003010  104400  014267                      TYPE    ,T2EM           ;PRINT ERRORS MESG
003014  000137  005516                      JMP     EXIT            ;EXIT TEST
003020  104400  013461          2$:     TYPE    ,MANMSG         ;PRINT INSTRUCTIONS
003024  032777  010000  176070  3$:     BIT     #SW12,@SWR      ;SW12 UP?
003032  001774                              BEQ     3$              ;NO, WAIT FOR START

003034  012702  000041          T2SP:   MOV     #41,R2          ;SET START CHAR
003040  010200                  1$:     MOV     R2,R0           ;SET CHAR
003042  013701  001130          2$:     MOV     WIDTH,R1        ;SET COLUMN COUNT
003046  104414                  3$:     LOAD                    ;LOAD CHAR
003050  005301                          DEC     R1              ;DEC CHAR COUNT
003052  001407                          BEQ     4$              ;BRANCH IF DONE LINE
003054  005200                          INC     R0              ;NEXT CHAR
003056  020027  000177                  CMP     R0,#177         ;CHECK CHAR
003062  001371                          BNE     3$              ;OK, CONTINUE
003064  012700  000040                  MOV     #40,R0          ;SET CHAR = SPACE
003070  000766                          BR      3$              ;CONTINUE
003072  104413  001142          4$:     PRINT   ,$LF            ;PRINT LINE
003076  005204                          INC     R4              ;INC LINE COUNT
003100  105737  001135                  TSTB    CKFLAG          ;USING CLOCK TIMING?
003104  001006                          BNE     5$              ;YES, BYPASS MANUAL TIMING
003106  032777  010000  176006          BIT     #SW12,@SWR      ;SW12 DOWN?
003114  001002                          BNE     5$              ;NO, CONTINUE
003116  000137  003134                  JMP     SPD             ;YES, EXIT PRINTING ROUTINE
```

```
                                ;//////////////////////////////////////////////////////////////////
                                ;
                                ;ROUTINE TO PRINT/TYPE MEASURED PRINT SPEED FOR TEST 2.
                                ;
                                ;//////////////////////////////////////////////////////////////////
                                ;
003122  005202          5$:     INC     R2              ;INC START CHAR
003124  020227  000177          CMP     R2,#177         ;CHECK IT
003130  001343                  BNE     1$              ;OK, CONTINUE
003132  000740                  BR      T2SP            ;RESET START CHAR
                                ;
003134  012700  000177  SPD:    MOV     #177,R0         ;SET RUBOUT CHAR
003140  104414                  LOAD                    ;CLEAR CHAR BUFFER
003142  104400  013623          TYPE    ,PRSP1          ;START MESG
003146  104413  013623          PRINT   ,PRSP1
003152  105737  001135          TSTB    CKFLAG          ;USING CLOCK?
003156  001004                  BNE     1$              ;YES, BRANCH
003160  104400  013644          TYPE    ,PRSP2          ;ADD WORD "APPROXIMATELY" TO MESG
003164  104413  013644          PRINT   ,PRSP2
003170                  1$:
003170  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
003172  004737  011120          JSR     PC,$SB2D        ;CONVERT #
003176  104416                  CNLOAD                  ;LOAD #
003200  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
003202  104404                  TYPDS                   ;TYPE #
003204  104413  013655          PRINT   ,PRSP3          ;LOAD MORE OF MESG
003210  104400  013655          TYPE    ,PRSP3
003214  013746  001130          MOV     WIDTH,-(SP)     ;#COLUMNS ON STACK
003220  104404                  TYPDS                   ;TYPE #
003222  104400  013704          TYPE    ,PRSP4          ;TYPE END OF MESG
003226  013746  001130          MOV     WIDTH,-(SP)     ;# COLUMNS ON STACK AGAIN
003232  004737  011120          JSR     PC,$SB2D        ;CONVERT #
003236  104416                  CNLOAD                  ;LOAD #
003240  104413  013704          PRINT   ,PRSP4          ;PRINT MESG ON LA180
003244  000137  005516          JMP     EXIT            ;EXIT TEST
```

```
                                ;*****************************************************************

                                .SBTTL  PRINTING TESTS



                                ;/////////////////////////////////////////////////////////////////

                                ;TEST20 -       DATA TRANSFER PATHS TEST
                                ;
                                ;THIS TEST PRINTS 16 LINES OF ALTERNATING *'S AND U'S IN A CHECKERBOARD
                                ;PATTERN

                                ;/////////////////////////////////////////////////////////////////

003250  104412          TEST20: PRTHDR                  ;PRINT TEST HEADER
003252  012703  052452          MOV     #"*U,R3         ;SET FIRST CHAR PAIR
003256  012702  000020          MOV     #16.,R2         ;SET LINE COUNT
003262  010300          1$:     MOV     R3,R0           ;SET CHAR PAIR
003264  013701  001130          MOV     WIDTH,R1        ;SET COLUMN COUNT
003270  104414          2$:     LOAD                    ;LOAD CHAR
003272  000300                  SWAB    R0              ;SET NEXT CHAR
003274  005301                  DEC     R1              ;DEC COLUMN COUNT
003276  001374                  BNE     2$              ;FINISH LINE
003300  000303                  SWAB    R3              ;SET NEXT LINE START CHAR
003302  104413  001142          PRINT   $LF             ;PRINT LINE
003306  005302                  DEC     R2              ;DEC LINE COUNT
003310  001364                  BNE     1$              ;FINISH TEST
003312  000137  005516          JMP     EXIT            ;EXIT WHEN DONE
```

```
                                ;//////////////////////////////////////////////////////////////////////////////
                                ;
                                ;TEST21 -       HEAD POSITIONING TEST
                                ;
                                ;THIS TEST PRINTS A SINGLE LINE OF ALTERNATING O'S AND SPACES
                                ;THEN FILLS IN THE SPACES WITH X'S ONE AT A TIME.
                                ;
                                ;//////////////////////////////////////////////////////////////////////////////

003316  104412          TEST21: PRTHDR                  ;PRINT TEST HEADER
003320  013701  001130          MOV     WIDTH,R1        ;SET COLUMN COUNT
003324  012700  000060  1$:     MOV     #60,R0          ;SET CHAR
003330  104414                  LOAD                    ;LOAD CHAR
003332  005301                  DEC     R1              ;DEC CHAR COUNT
003334  001405                  BEQ     2$              ;PRINT LINE WHEN LOADED
003336  012700  000040          MOV     #40,R0          ;SET SPACE CHAR
003342  104414                  LOAD                    ;LOAD SPACE
003344  005301                  DEC     R1              ;DEC CHAR COUNT
003346  001366                  BNE     1$              ;FINISH LINE
003350  104413  001144  2$:     PRINT   .$CR            ;PRINT LINE
003354  012702  000002          MOV     #2,R2           ;SET FIRST CHAR COUNT
003360  012700  000040  3$:     MOV     #40,R0          ;SET SPACE CHAR
003364  010201                  MOV     R2,R1           ;GET CHAR COUNT
003366  005301                  DEC     R1              ;SUBTRACT ONE
003370  104415                  MLOAD                   ;LOAD SPACES
003372  012700  000130          MOV     #'X,R0          ;SET X CHAR
003376  104414                  LOAD                    ;LOAD X
003400  104413  001144          PRINT   .$CR            ;PRINT LINE
003404  062702  000002          ADD     #2,R2           ;ADD 2 TO CHAR COUNT
003410  020237  001130          CMP     R2,WIDTH        ;DONE LINE?
003414  101761                  BLOS    3$              ;NO, FINISH LINE
003416  104413  001142          PRINT   .$LF            ;YES, ADVANCE PAPER
003422  000137  005516          JMP     EXIT            ;EXIT TEST
```

```
                                ;//////////////////////////////////////////////////////////////////////////////
                                ;
                                ;TEST22 -       BACKSPACE TEST
                                ;
                                ;2 LINES OF X'S INTERSPACED WITH DASHES WILL BE PRINTED BY PRINTING A SLASH,
                                ;EXECUTING A BACKSPACE, AND THEN PRINTING A BACKSLASH TO
                                ;COMPLETE EACH X CHAR.
                                ;A MAXIMUM OF 127 COLUMNS WILL BE PRINTED BY THIS TEST.
                                ;
                                ;//////////////////////////////////////////////////////////////////////////////

003426  104412                  TEST22: PRTHDR                  ;PRINT TEST HEADER
003430  012702  000002                  MOV     #2,R2           ;SET LINE COUNT
003434  013701  001130          1$:     MOV     WIDTH,R1        ;SET COLUMN COUNT
003440  020127  000177                  CMP     R1,#127.        ;CHECK # COLUMNS
003444  003402                          BLE     2$              ;GREATER THAN 127?
003446  012701  000177                  MOV     #127.,R1        ;YES, SET TO 127
003452  104413  003510          2$:     PRINT   ,10$            ;LOAD SLASH-BS-BACKSLASH
003456  005301                          DEC     R1              ;DEC CLOUMN COUNT
003460  001405                          BEQ     3$              ;PRINT LINE IF DONE
003462  012700  000055                  MOV     #55,R0          ;SET ASCII OF DASH
003466  104414                          LOAD                    ;LOAD DASH CHAR
003470  005301                          DEC     R1              ;DEC COLUMN COUNT
003472  001367                          BNE     2$              ;FINISH LINE
003474  104413  001142          3$:     PRINT   ,$LF            ;PRINT LINE
003500  005302                          DEC     R2              ;DEC LINE COUNT
003502  001354                          BNE     1$              ;FINISH TEST
003504  000137  005516                  JMP     EXIT            ;EXIT TEST

003510  004057  000134          10$:    .ASCIZ  <57><10><134>

                                        .EVEN
```

```
                                ;*****************************************************
                                ;
                                ;TEST23 -       CHARACTER GENERATOR TEST
                                ;
                                ;THIS TEST PRINTS A SINGLE LINE (30 CHARACTERS LONG) OF EACH PRINTABLE CHAR
                                ;PRECEDED BY A LINE OF ALL SPACES
                                ;
                                ;*****************************************************

003514  104412                  TEST23: PRTHDR                  ;PRINT TEST HEADER
003516  012700  000040                  MOV     #40,R0          ;SET FIRST CHAR
003522  012701  000036          1$:     MOV     #30.,R1         ;SET CHAR COUNT
003526  104415                          MLOAD                   ;LOAD LINE
003530  104413  001142                  PRINT   $LF             ;PRINT LINE
003534  005200                          INC     R0              ;SET NEXT CHAR
003536  020027  000177                  CMP     R0,#177         ;CHAR = RUBOUT?
003542  001367                          BNE     1$              ;NO, FINISH TEST
003544  000137  005516                  JMP     EXIT            ;EXIT TEST
```

```
                                ;*****************************************************************
                                ;
                                ;TEST24 -       NON PRINTABLE CHARACTER TEST
                                ;
                                ;THIS TEST PRINTS A 30 LINE SWIRL PATTERN WITH NON-PRINTABLE CHARACTERS
                                ;LOADED BEFORE AND AFTER THE PRINTING CHARACTERS TO TEST ALL AREAS OF THE
                                ;CHARACTER BUFFER IN THE LA180.  IF THIS TEST IS LOOPED ON, THE SWIRL
                                ;PATTERN WILL CONTINUE, 30 LINES PRINTED EACH TIME THE TEST IS LOOPED.
                                ;
                                ;*****************************************************************

003550  104412                  TEST24: PRTHDR                  ;PRINT TEST HEADER
003552  012702  000041                  MOV     #41,R2          ;SET START CHAR
003556  012705  000036                  MOV     #30.,R5         ;SET LINE COUNT
003562  013701  001130                  MOV     WIDTH,R1        ;GET COLUMN COUNT
003566  012703  000377                  MOV     #255.,R3        ;SET BUFFER SIZE
003572  160103                        SUB     R1,R3           ;SUBTRACT COLUMNS COUNT
003574  005004                        CLR     R4              ;CLEAR CHAR INC COUNT
003576  162703  000035          2$:     SUB     #29.,R3         ;DIVIDE NON-PRINT CHAR COUNT BY 29
003602  002402                        BLT     3$
003604  005204                        INC     R4              ;R4 = NON-PRINT CHAR INC COUNT
003606  000773                        BR      2$
003610  005003                  3$:     CLR     R3              ;CLEAR NON-PRINT CHAR COUNT 2ND BLOCK
003612  162701  000377          4$:     SUB     #255.,R1        ;CALCULATE # NON-PRINT CHARS, 1ST BLOCK
003616  005401                        NEG     R1
003620  160301                        SUB     R3,R1
003622  004737  003754                JSR     PC,10$          ;LOAD 1ST BLOCK OF NON-PRINT CHARS
003626  013701  001130                MOV     WIDTH,R1        ;SET # PRINTABLE CHARS (COLUMN COUNT)
003632  010200                        MOV     R2,R0           ;SET FIRST PRINT CHAR
003634  104414                  5$:     LOAD                    ;LOAD PRINTABLE CHAR
003636  005301                        DEC     R1              ;DEC CHAR COUNT
003640  001407                        BEQ     6$              ;BRANCH IF DONE PRINTABLE CHARS
003642  005200                        INC     R0              ;NEXT CHAR
003644  020027  000177                CMP     R0,#177         ;CHECK CHAR
003650  001371                        BNE     5$              ;OK - CONTINUE
003652  012700  000040                MOV     #40,R0          ;RESET CHAR = SPACE
003656  000766                        BR      5$              ;CONTINUE
003660  010301                  6$:     MOV     R3,R1           ;SET # NON-PRINT CHARS, 2ND BLOCK
003662  004737  003754                JSR     PC,10$          ;LOAD 2ND BLOCK NON-PRINT CHARS
003666  104413  001142                PRINT   $LF             ;PRINT LINE
003672  005305                        DEC     R5              ;DEC LINE COUNT
003674  003015                        BGT     8$              ;CONTINUE TEST
003676  105737  001134                TSTB    TLOOP           ;LOOP ON TEST?
003702  001006                        BNE     7$              ;YES, INITIALIZE
003704  032777  040000  175210        BIT     #SW14,@SWR      ;LOOP ON TEST?
003712  001002                        BNE     7$              ;YES, INITIALIZE
003714  000137  005516                JMP     EXIT            ;EXIT TEST
003720  012705  000036          7$:     MOV     #30.,R5         ;RESET LINE COUNT
003724  005003                        CLR     R3              ;CLEAR NON-PRINT CHAR COUNT
003726  000401                        BR      9$              ;CONTINUE SWIRL
003730  060403                  8$:     ADD     R4,R3           ;INC NON-PRINT CHAR COUNT, 2ND BLOCK
003732  013701  001130          9$:     MOV     WIDTH,R1        ;RESET COLUMN COUNT
003736  005202                        INC     R2              ;INC START CHAR
003740  020227  000177                CMP     R2,#177         ;CHECK START CHAR
```

```
003744  001322                          BNE     4$              ;OK, CONTINUE
003746  012702  000041                  MOV     #41,R2          ;RESET START CHAR
003752  000717                          BR      4$              ;CONTINUE

                                ;////////////////////////////////////////////////////////////

                                ;ROUTINE TO LOAD NON-PRINTABLE CHARACTERS FOR TEST 24.

                                ;////////////////////////////////////////////////////////////

003754  005701                  10$:    TST     R1              ;TEST CHAR COUNT
003756  001430                          BEQ     14$             ;RETURN IF ZERO
003760  005000                  11$:    CLR     R0              ;SET FIRST NON-PRINT CHAR
003762  104414                  12$:    LOAD                    ;LOAD CHAR
003764  005301                          DEC     R1              ;DEC CHAR COUNT
003766  003424                          BLE     14$             ;RETURN IF DONE
003770  005200                  13$:    INC     R0              ;NEXT CHAR
003772  020027  000007                  CMP     R0,#7           ;CHAR = BELL ?
003776  001774                          BEQ     13$             ;YES, NEXT CHAR
004000  020027  000010                  CMP     R0,#10          ;CHAR = BS?
004004  001771                          BEQ     13$             ;YES, NEXT CHAR
004006  020027  000012                  CMP     R0,#12          ;CHAR = LF?
004012  001766                          BEQ     13$             ;YES, NEXT CHAR
004014  020027  000014                  CMP     R0,#14          ;CHAR = FF?
004020  001763                          BEQ     13$             ;YES, NEXT CHAR
004022  020027  000015                  CMP     R0,#15          ;CHAR = CR?
004026  001760                          BEQ     13$             ;YES, NEXT CHAR
004030  020027  000040                  CMP     R0,#40          ;CHAR = SPACE?
004034  001751                          BEQ     11$             ;YES, RESET CHAR
004036  000751                          BR      12$             ;CONTINUE
004040  000207                  14$:    RTS     PC              ;RETURN
```

```
                                ;//////////////////////////////////////////////////////////////////////////////
                                ;
                                ;TEST25 -       BUFFER TEST
                                ;
                                ;THIS TEST CHECKS THE CHARACTER BUFFER OF THE LA180 WHILE PRINTING
                                ;FOUR LINES OF NUMBERS (WITH 2 BLANK LINES BETWEEN THE FIRST AND SECOND
                                ;LINE).  THESE LINES CAN BE USED TO CHECK THE PROPER PRINTING WIDTH.
                                ;ANY E PRINTED INDICATES AN INCORRECT LOAD OR BUFFER ACTION.
                                ;
                                ;//////////////////////////////////////////////////////////////////////////////

004042  104412          TEST25: PRTHDR                  ;PRINT TEST HEADER
004044  012701  000020          MOV     #16.,   R1      ;SET CHAR COUNT
004050  012700  000105          MOV     #'E,R0          ;SET E CHAR
004054  104415                  MLOAD                   ;LOAD BUFFER
004056  012700  000177          MOV     #177,R0         ;SET RUBOUT CHAR
004062  104414                  LOAD                    ;CLEAR BUFFER
004064  005002                  CLR     R2              ;CLEAR CHAR COUNT
004066  013701  001130          MOV     WIDTH,R1        ;SET COLUMN COUNT
004072  012700  000061          MOV     #61,R0          ;SET CHAR
004076  104414          1$:     LOAD                    ;LOAD CHAR
004100  005301                  DEC     R1              ;DEC COLUMN COUNT
004102  001414                  BEQ     2$              ;PRINT LINE WHEN LOADED
004104  005202                  INC     R2              ;INC CHAR COUNT
004106  020227  000144          CMP     R2,#100.        ;DONE ONES?
004112  002771                  BLT     1$              ;NO, CONTINUE
004114  012700  000063          MOV     #63,R0          ;SET NEXT CHAR
004120  020227  000202          CMP     R2,#130.        ;DONE 3'S?
004124  002764                  BLT     1$              ;NO, CONTINUE
004126  012700  000062          MOV     #62,R0          ;YES, SET NEXT CHAR
004132  000761                  BR      1$              ;CONTINUE LOADING CHARACTERS
004134  104413  001142  2$:     PRINT   ,$LF            ;PRINT LINE
004140  012701  000400          MOV     #256.,R1        ;SET CHAR COUNT
004144  012700  000105          MOV     #'E,R0          ;SET E CHAR
004150  104415                  MLOAD                   ;LOAD BUFFER
004152  104413  001142          PRINT   ,$LF            ;PRINT BLANK LINE
004156  012701  000376          MOV     #254.,R1        ;SET CHAR COUNT
004162  012700  000105          MOV     #'E,R0          ;SET E CHAR
004166  104415                  MLOAD                   ;LOAD BUFFER
004170  012700  000177          MOV     #177,R0         ;SET RUBOUT CHAR
004174  104414                  LOAD                    ;CLEAR BUFFER
004176  104413  001142          PRINT   ,$LF            ;BLANK LINE
004202  012701  000400          MOV     #256.,R1        ;SET CHAR COUNT
004206  012700  000105          MOV     #'E,R0          ;SET E CHAR
004212  104415                  MLOAD                   ;LOAD BUFFER
004214  013701  001130          MOV     WIDTH,R1        ;SET COLUMN COUNT
004220  012700  000060          MOV     #60,R0          ;SET CHAR
004224  005002                  CLR     R2              ;CLEAR CHAR COUNT
004226  104414          3$:     LOAD                    ;LOAD CHAR
004230  005301                  DEC     R1              ;DEC COLUMN COUNT
004232  001407                  BEQ     4$              ;PRINT LINE WHEN LOADED
004234  005202                  INC     R2              ;INC CHAR COUNT
004236  020227  000143          CMP     R2,#99.         ;DONE ZEROS?
004242  002771                  BLT     3$              ;NO, CONTINUE
```

```
  967  004244  012700  000061              MOV     #61,R0          ;YES, SET NEXT CHAR
  968  004250  000766                      BR      3$              ;CONTINUE
  969  004252  104413  001142      4$:     PRINT   .$LF            ;PRINT LINE
  970  004256  012701  000400              MOV     #256.,R1        ;SET CHAR COUNT
  971  004262  012700  000105              MOV     #'E,R0          ;SET E CHAR
  972  004266  104415                      MLOAD                   ;LOAD BUFFER
  973  004270  012700  000177              MOV     #177,R0         ;SET RUBOUT CHAR
  974  004274  104414                      LOAD                    ;CLEAR BUFFER
  975  004276  012700  000060              MOV     #60,R0          ;SET CHAR
  976  004302  012702  000011              MOV     #9.,R2          ;SET GROUP COUNT
  977  004306  013701  001130              MOV     WIDTH,R1        ;SET CHAR COUNT
  978  004312  104414              5$:     LOAD                    ;LOAD CHAR
  979  004314  005302                      DEC     R2              ;DEC GROUP COUNT
  980  004316  001010                      BNE     7$              ;FINISH GROUP
  981  004320  005200                      INC     R0              ;SET NEXT CHAR
  982  004322  020027  000072              CMP     R0,#72          ;GOOD CHAR?
  983  004326  103402                      BLO     6$              ;YES, CONTINUE
  984  004330  012700  000060              MOV     #60,R0          ;NO, RESET CHAR TO ZERO
  985  004334  012702  000012      6$:     MOV     #10.,R2         ;RESET GROUP COUNT
  986  004340  005301              7$:     DEC     R1              ;DEC CHAR COUNT
  987  004342  001363                      BNE     5$              ;FINISH LINE
  988  004344  104413  001142              PRINT   .$LF            ;PRINT LINE
  989  004350  012701  000400              MOV     #256.,R1        ;SET CHAR COUNT
  990  004354  012700  000105              MOV     #'E,R0          ;SET E CHAR
  991  004360  104415                      MLOAD                   ;LOAD BUFFER
  992  004362  012700  000061              MOV     #61,R0          ;SET CHAR
  993  004366  013701  001130              MOV     WIDTH,R1        ;SET CHAR COUNT
  994  004372  104414              8$:     LOAD                    ;LOAD CHAR
  995  004374  005200                      INC     R0              ;SET NEXT CHAR
  996  004376  020027  000072              CMP     R0,#72          ;GOOD CHAR?
  997  004402  103402                      BLO     9$              ;YES, CONTINUE
  998  004404  012700  000060              MOV     #60,R0          ;RESET CHAR
  999  004410  005301              9$:     DEC     R1              ;DEC CHAR COUNT
 1000  004412  001367                      BNE     8$              ;FINISH LINE
 1001  004414  104413  001142              PRINT   .$LF            ;PRINT LINE
 1002  004420  000137  005516              JMP     EXIT            ;EXIT TEST
```

```
                                ;////////////////////////////////////////////////////////////////
                                ;
                                ;TEST26 -       OVERPRINT TEST
                                ;
                                ;THIS TEST PRINTS FOUR LINES OF ALTERNATING CHARACTERS AND SPACES
                                ;IN A CHECKERBOARD PATTERN.  EACH LINE IS OVERPRINTED TWICE.
                                ;
                                ;////////////////////////////////////////////////////////////////

004424  104412                  TEST26: PRTHDR                  ;PRINT TEST HEADER
004426  012703  004504                  MOV     #6$,R3          ;SET TABLE POINTER
004432  012702  000003          1$:     MOV     #3,R2           ;SET OVERPRINT COUNT
004436  011300                  2$:     MOV     (R3),R0         ;GET CHAR PAIR
004440  001417                          BEQ     5$              ;EXIT IF DONE TEST
004442  013701  001130                  MOV     WIDTH,R1        ;SET COLUMN COUNT
004446  104414                  3$:     LOAD                    ;LOAD CHAR
004450  000300                          SWAB    R0              ;SET NEXT CHAR
004452  005301                          DEC     R1              ;DEC COLUMN COUNT
004454  001374                          BNE     3$              ;FINISH LINE
004456  005302                          DEC     R2              ;DEC OVERPRINT COUNT
004460  001403                          BEQ     4$              ;BRANCH IF DONE OVERPRINT
004462  104413  001144                  PRINT   ,$CR            ;PRINT LINE
004466  000763                          BR      2$              ;CONTINUE
004470  104413  001142          4$:     PRINT   ,$LF            ;PRINT LINE
004474  005723                          TST     (R3)+           ;INC TABLE POINTER
004476  000755                          BR      1$              ;PRINT NEXT LINE
004500  000137  005516          5$:     JMP     EXIT            ;EXIT TEST

004504  020105                  6$:     .ASCII  /E /
004506  040040                          .ASCII  / /
004510  020115                          .ASCII  /M /
004512  021440                          .ASCII  / */
004514  000000                          .WORD   0               ;END OF TABLE
```

```
1036                                    ;////////////////////////////////////////////////////////////////////////
1037
1038                                    ;TEST27 -       MULTIPLE LINE FEED TEST
1039                                    ;
1040                                    ;NUMBER PRINTED INDICATES NUMBER OF LINE FEEDS FOLLOWING THAT LINE.
1041                                    ;DASHED REFERENCE LINES ARE PRINTED TO AID IN CHECKING PROPER
1042                                    ;LINE FEEDS.
1043
1044                                    ;////////////////////////////////////////////////////////////////////////
1045
1046  004516  104412                    TEST27: PRTHDR                  ;PRINT TEST HEADER
1047  004520  012703  004624                    MOV     #5$,R3          ;SET TABLE POINTER
1048  004524  111346                    1$:     MOVB    (R3),-(SP)      ;NUMBER ONTO STACK
1049  004526  004737  011120                    JSR     PC,$$B2D        ;CONVERT #
1050  004532  104416                            CNLOAD                  ;LOAD NUMBER
1051  004534  121327  000001                    CMPB    (R3),#1         ;TEST #
1052  004540  101406                            BLOS    2$              ;PRINT FULL DASH LINE IF 0 OR 1
1053  004542  012701  000035                    MOV     #29.,R1         ;SET DASH LENGTH
1054  004546  121327  000010                    CMPB    (R3),#8.        ;CHECK #
1055  004552  101411                            BLOS    4$              ;2,4 OR 8 - PRINT 29 DASHES
1056  004554  000407                            BR      3$              ;16 OR 32 - PRINT 28 DASHES
1057  004556  013701  001130            2$:     MOV     WIDTH,R1        ;SET CHAR COUNT
1058  004562  020127  000036                    CMP     R1,#30.         ;CHECK IT
1059  004566  103002                            BHIS    3$              ;OK, CONTINUE
1060  004570  012701  000036                    MOV     #30.,R1         ;SET TO 30
1061  004574  005301                    3$:     DEC     R1              ;SUBTRACT ONE
1062  004576  012700  000055            4$:     MOV     #55,R0          ;SET DASH CHAR
1063  004602  104415                            MLOAD                   ;LOAD DASH LINE
1064  004604  111301                            MOVB    (R3),R1         ;SET LF COUNT
1065  004606  113700  001142                    MOVB    $LF,R0          ;SET LF CHAR
1066  004612  104415                            MLOAD                   ;LOAD LF'S
1067  004614  105723                            TSTB    (R3)+           ;INC TABLE POINTER
1068  004616  003342                            BGT     1$              ;FINISH TEST
1069  004620  000137  005516                    JMP     EXIT            ;EXIT TEST
1070
1071  004624     001     002     004    5$:     .BYTE   1,2,4.,8.,16.,32.,0
1072  004627     010     020     040
1073  004632     000
1074
1075          004634                            .EVEN
```

```
1076                                    ;////////////////////////////////////////////////////////////////////////
1077
1078                                    ;TEST30 -       RIBBON FEED TEST
1079                                    ;
1080                                    ;THIS TEST PRINTS A SINGLE COLUMN OF 24 LINES OF X'S DOWN THE LEFT
1081                                    ;HAND MARGIN OF THE PAGE.
1082
1083                                    ;////////////////////////////////////////////////////////////////////////
1084
1085  004634  104412                    TEST30: PRTHDR                  ;PRINT TEST HEADER
1086  004636  012701  000030                    MOV     #24.,R1         ;SET LINE COUNT
1087  004642  104413  004656            1$:     PRINT   ,2$             ;PRINT X - LF
1088  004646  005301                            DEC     R1              ;DEC LINE COUNT
1089  004650  001374                            BNE     1$              ;FINISH TEST
1090  004652  000137  005516                    JMP     EXIT            ;EXIT TEST
1091
1092  004656  005130     000            2$:     .ASCIZ  /X/<12>
1093
1094          004662                            .EVEN
1095
1096
1097
1098                                    ;////////////////////////////////////////////////////////////////////////
1099
1100                                    ;TEST31 -       BELL TEST
1101                                    ;
1102                                    ;THIS TEST WILL SOUND 5 BELLS BETWEEN PRINTING "BELL TEST"
1103
1104                                    ;////////////////////////////////////////////////////////////////////////
1105
1106  004662  104412                    TEST31: PRTHDR                  ;PRINT TEST HEADER
1107  004664  104413  004674                    PRINT   ,1$             ;DO TEST
1108  004670  000137  004716                    JMP     $EOP            ;EXIT TEST
1109
1110  004674  041007  046105  003514     1$:     .ASCIZ  <7>/BELL/<7>. TEST/<7><15><7><12><7><15>
1111  004702  052040  051505  003524
1112  004710  003415  003412  000015
1113
1114                                            .EVEN
```

```
                                        ;*****************************************************************

                                        .SBTTL  END OF PASS ROUTINE

                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO EXIT

004716                                  $EOP:
004716  000240                                  NOP
004720  005237  004772                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
004724  042737  100000  004772                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
004732  005327                                  DEC     (PC)+           ;;LOOP?
004734  000001                          $EOPCT: .WORD   1
004736  003013                                  BGT     $DOAGN          ;;YES
004740  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
004742  000001                          $ENDCT: .WORD   1
004744  004734                                  $EOPCT
004746                                  $GET42:
                                        
004746  013700  000042                        MOV     @#42,R0         ;;GET MONITOR ADDRESS
004752  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
004754  000005                                  RESET                   ;;CLEAR THE WORLD
004756  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
004760  000240                                  NOP                     ;;SAVE ROOM
004762  000240                                  NOP                     ;;FOR
004764  000240                                  NOP                     ;;ACT11
004766                                  $DOAGN:
004766  000137  005516                        JMP     @#EXIT          ;;RETURN
004772  000000                          $PASS:  .WORD   0               ;;NUMBER OF PASSES
```

```
                                        ;*****************************************************************

                                        .SBTTL  MAINTENANCE AIDS



                                        ;/////////////////////////////////////////////////////////////////

                                        ;TEST60 -       LIFE TEST
                                        ;
                                        ;THIS TEST PRINTS 2 FULL LINES OF EACH PRINTABLE CHARACTER
                                        ;THE SECOND LINE IS OVERPRINTED 4 TIMES TO CONSERVE PAPER
                                        ;AT THE END OF EACH PASS THROUGH THE ENTIRE PRINTABLE CHARACTER
                                        ;SET, THE PASS COUNT WILL BE PRINTED ON THE LA180.

                                        ;/////////////////////////////////////////////////////////////////

004774  005037  005106          TEST60: CLR     PASCNT          ;CLEAR PASS COUNT
005000  104412                  1$:     PRTHDR                  ;PRINT TEST HEADER, FEED BLANK LINES
005002  012700  000041                  MOV     #41,R0          ;SET FIRST CHAR
005006  013701  001130          2$:     MOV     WIDTH,R1        ;SET COLUMN COUNT
005012  104415                          MLOAD                   ;LOAD LINE
005014  104413  001142                  PRINT   ,$LF            ;PRINT LINE
005020  012702  000005                  MOV     #5,R2           ;SET OVERPRINT COUNT
005024  013701  001130          3$:     MOV     WIDTH,R1        ;SET COLUMN COUNT
005030  104415                          MLOAD                   ;LOAD LINE
005032  104413  001144                  PRINT   ,$CR            ;PRINT LINE
005036  005302                          DEC     R2              ;DEC OVERPRINT COUNT
005040  001371                          BNE     3$              ;FINISH OVERPRINT
005042  104413  001142                  PRINT   ,$LF            ;ADVANCE PAPER
005046  005200                          INC     R0              ;SET NEXT CHAR
005050  020027  000177                  CMP     R0,#177         ;TEST CHAR
005054  001354                          BNE     2$              ;OK, CONTINUE
005056  005237  005106                  INC     PASCNT          ;INC PASS COUNT
005062  104413  013310                  PRINT   ,PASMSG         ;LOAD PASS COUNT MESG
005066  013746  005106                  MOV     PASCNT,-(SP)    ;;PUSH PASCNT ON STACK
005072  004737  011120                  JSR     PC,$SB2D        ;CONVERT #
005076  104416                          CNLOAD                  ;LOAD #
005100  104413  001142                  PRINT   ,$LF            ;PRINT MESG
005104  000735                          BR      1$              ;START NEXT PASS

005106  000000          PASCNT: .WORD   0               ;PASS COUNT
```

```
                                ;//////////////////////////////////////////////////////////////////////////
                                ;
                                ;TEST61 -       SCOPE DRIVE ROUTINE
                                ;
                                ;THIS TEST WILL LOAD A CHARACTER SET IN SWITCH REGISTER BITS 0-6
                                ;IF SWITCH 9 IS DOWN, FULL LINES WILL BE LOADED & PRINTED (LF).
                                ;IF SWITCH 9 IS UP, THE CHAR WILL BE LOADED ONCE AND THE PROGRAM
                                ;WILL HALT.  NO LINE FEEDS OR CARRIAGE RETURNS WILL BE SENT BY THE
                                ;PROGRAM.
                                ;
                                ;//////////////////////////////////////////////////////////////////////////
                                ;
005110  104412          TEST61: PRTHDR                  ;PRINT TEST HEADER
005112  000422                  BR      11$             ;CHECK SWITCH REG FIRST
005114  013701  001130    10$:  MOV     WIDTH,R1        ;SET COLUMN COUNT
005120  017700  173776    1$:   MOV     @SWR,R0         ;GET CHARACTER
005124  042700  177600          BIC     #^C177,R0       ;MASK UNWANTED BITS
005130  104414                  LOAD                    ;LOAD CHAR
005132  020027  000015          CMP     R0,#15          ;CHAR = CR?
005136  001410                  BEQ     11$             ;YES, RESET COLUMN COUNT
005140  020027  000012          CMP     R0,#12          ;CHAR = LF?
005144  001405                  BEQ     11$             ;YES, RESET COLUMN COUNT
005146  020027  000040          CMP     R0,#40          ;NON-PRINTABLE CHAR?
005152  103404                  BLO     12$             ;YES, DON'T DEC COLUMN COUNT
005154  005301                  DEC     R1              ;DEC COLUMN COUNT
005156  000402                  BR      12$             ;CONTINUE
005160  013701  001130    11$:  MOV     WIDTH,R1        ;RESET COLUMN COUNT
005164  032777  001000  173730 12$:  BIT     #SW9,@SWR       ;TEST SWITCH 9
005172  001402                  BEQ     2$              ;PRINT FULL LINE
005174  000000                  HALT                    ;ONE CHAR - HALT
005176  000750                  BR      1$              ;GET NEXT CHAR
005200  005701            2$:   TST     R1              ;TEST COLUMN COUNT
005202  003346                  BGT     1$              ;CONTINUE IF NOT DONE LINE
005204  001403                  BEQ     3$              ;LOADED MORE THAN WIDTH?
005206  012700  000177          MOV     #177,R0         ;YES, CLEAR BUFFER
005212  104414                  LOAD
005214  104413  001142    3$:   PRINT   ,$LF            ;PRINT LINE, ADVANCE PAPER
005220  000735                  BR      10$             ;CONTINUE
```

```
1225                                ;////////////////////////////////////////////////////////////////////////
1226
1227                                ;TEST62 -       LINE PRINT TEST
1228                                ;
1229                                ;THIS TEST PRINTS FULL LINES CONTINUOUSLY OF WHATEVER CHARACTER
1230                                ;IS TYPED ON THE CONSOLE KEYBOARD.  TO CHANGE CHARACTERS, RESELECT
1231                                ;THIS TEST.  AN ERROR MESG WILL BE PRINTED IF THIS TEST IS SELECTED
1232                                ;AND A CONSOLE TERMINAL DOES NOT EXIST.
1233
1234                                ;////////////////////////////////////////////////////////////////////////
1235
1236  005222  104412                TEST62: PRTHDR                  ;PRINT TEST HEADER
1237  005224  105737  001153                TSTB    $TPFLG          ;CHECK IF TERMINAL EXISTS
1238  005230  001073                BNE     TERR                    ;EXIT IF NONE
1239  005232  104400  013413        TYPE    ,TCHAR                  ;TYPE INSTR
1240  005236  105777  173636        2$:     TSTB    @TKS            ;WAIT FOR KYBD FLAG
1241  005242  100375                BPL     2$
1242  005244  104417                CHECK                           ;CHECK CHAR FOR CONTROL
1243  005246  017700  173630        MOV     @TKB,R0                 ;GET CHAR
1244  005252  010046                MOV     R0,-(SP)                ;CHAR ONTO STACK
1245  005254  004737  007172        JSR     PC,$TYPEC               ;ECHO CHAR
1246  005260  012746  000200        MOV     #CRLF,-(SP)             ;SEND CR-LF
1247  005264  004737  007172        JSR     PC,$TYPEC
1248  005270  013701  001130        1$:     MOV     WIDTH,R1        ;SET COLUMN COUNT
1249  005274  104415                MLOAD                           ;LOAD LINE
1250  005276  104413  001142        PRINT   ,$LF                    ;PRINT LINE
1251  005302  000772                BR      1$                      ;CONTINUE
```

```
1252                                    ;//////////////////////////////////////////////////////////////////////
1253
1254                                    ;TEST63 -       CHARACTER PRINT TEST
1255                                    ;
1256                                    ;THIS TEST LOADS WHATEVER CHARACTER IS TYPED ON THE CONSOLE KEYBOARD
1257                                    ;TO THE LA180, CHARACTER BY CHARACTER.
1258                                    ;IF THIS TEST IS SELECTED AND A CONSOLE TERMINAL DOES NOT EXIST,
1259                                    ;AN ERROR MESSAGE WILL BE PRINTED.
1260
1261                                    ;//////////////////////////////////////////////////////////////////////
1262
1263  005304  104412                   TEST63: PRTHDR                  ;PRINT TEST HEADER
1264  005306  105737  001153                   TSTB    $TPFLG          ;CHECK IF TERMINAL EXISTS
1265  005312  001042                           BNE     TERR            ;EXIT IF NONE
1266  005314  104400  013413                   TYPE    ,TCHAR          ;TYPE INSTR
1267  005320  104400  001141                   TYPE    ,$CRLF          ;SEND CR-LF
1268  005324  105777  173550           1$:     TSTB    @TKS            ;WAIT FOR KYBD FLAG
1269  005330  100375                           BPL     1$              ;WAIT FOR FLAG
1270  005332  104417                           CHECK                   ;CHECK CHAR FOR CONTROL
1271  005334  017700  173542                   MOV     @TKB,R0         ;GET CHAR
1272  005340  010046                           MOV     R0,-(SP)        ;CHAR ONTO STACK
1273  005342  004737  007172                   JSR     PC,$TYPEC       ;ECHO CHAR
1274  005346  104414                           LOAD                    ;LOAD CHAR
1275  005350  020027  000015                   CMP     R0,#CR          ;SEND LF AFTER CR
1276  005354  001003                           BNE     2$
1277  005356  012746  000012                   MOV     #LF,-(SP)
1278  005362  000413                           BR      4$
1279  005364  020027  000012           2$:     CMP     R0,#LF          ;SEND CR AFTER LF
1280  005370  001003                           BNE     3$
1281  005372  012746  000015                   MOV     #CR,-(SP)
1282  005376  000405                           BR      4$
1283  005400  020027  000014           3$:     CMP     R0,#FF          ;SND CRLF AFTER FF
1284  005404  001347                           BNE     1$
1285  005406  012746  000200                   MOV     #CRLF,-(SP)
1286  005412  004737  007172           4$:     JSR     PC,$TYPEC
1287  005416  000742                           BR      1$              ;CONTINUE
1288
1289  005420  104413  013433           TERR:   PRINT   ,NCMSG          ;PRINT ERROR MESG
1290  005424  000137  005516                   JMP     EXIT            ;EXIT TEST
```

```
                                        ;//////////////////////////////////////////////////////////////

                                        .SBTTL  CLOCK INTERRUPT SERVICE ROUTINES

                                        ;//////////////////////////////////////////////////////////////

005430  004737  005466                  DCI:    JSR     PC,SRDCI                ;DISABLE KW11-P INTERRUPT
005434  000177  000022                          JMP     @TORTN                  ;TIME OUT RETURN

005440  005337  005460                  LKSRV:  DEC     CNTR                    ;DEC TIME COUNT
005444  001401                                  BEQ     1$                      ;TIME OUT?
005446  000002                                  RTI                             ;NO, RETURN
005450  004737  005466                  1$:     JSR     PC,SRDCI                ;DISABLE KW11-L INTERRUPT
005454  000177  000002                          JMP     @TORTN                  ;TIME OUT RETURN

005460  000000                          CNTR:   .WORD   0                       ;KW11-L TIMING COUNT
005462  000000                          TORTN:  .WORD   0                       ;TIME OUT RETURN ADDRESS

005464  007020                          MINCNT: .WORD   7020                    ;60 HZ LINE FREQ. MINUTE COUNT
                                                                                ;SET TO 5670(8) FOR 50 HZ. LINE FREQ.

                                        ;//////////////////////////////////////////////////////////////

                                        ;SUBROUTINE TO DISABLE CLOCK OPTION INTERRUPTS

                                        ;//////////////////////////////////////////////////////////////

005466  105737  001135                  SRDCI:  TSTB    CKFLAG                  ;CHECK CLOCK OPTION FLAG
005472  001410                                  BEQ     2$                      ;RETURN IF NONE
005474  002404                                  BLT     1$                      ;BRANCH IF KW11-P
005476  042777  000100  173414                  BIC     #BIT6,@LKS              ;DISABLE KW11-L INTERRUPT
005504  000403                                  BR      2$                      ;RETURN
005506  042777  000100  173400          1$:     BIC     #BIT6,@PLKS             ;DISABLE KW11-P INTERRUPT
005514  000207                          2$:     RTS     PC
```

```
                                    ;//////////////////////////////////////////////////////////////////

                                    .SBTTL  TEST EXIT ROUTINE

                                    ;//////////////////////////////////////////////////////////////////

005516  004737  006012              EXIT:   JSR     PC,KYBDF        ;CHECK FOR KYBD FLAG
005522  032777  040000  173372              BIT     #SW14,@SWR      ;LOOP ON TEST SWITCH?
005530  001016                              BNE     3$              ;YES, RETURN TO TEST
005532  032777  000400  173362              BIT     #SW8,@SWR       ;WANT SW REG TEST SELECTION?
005540  001031                              BNE     SELECT          ;YES, SELECT TEST
005542  105737  001134                      TSTB    TLOOP           ;KYBD CTRL - LOOP ON TEST?
005546  001007                              BNE     3$              ;YES, RETURN TO TEST
005550  105737  001133                      TSTB    TRONE           ;KYBD CTRL - RUN TEST ONCE?
005554  001402                              BEQ     2$              ;NO, CONTINUE
005556  000137  006336                      JMP     TSEL            ;YES, SELECT TEST
005562  005237  001126              2$:     INC     $TSTNM          ;INCREMENT TEST NUMBER
005566  013700  001126              3$:     MOV     $TSTNM,R0       ;GET NEW TEST NUMBER
005572  006300                              ASL     R0              ;SET TABLE POINTER
005574  005760  011564                      TST     TAT(R0)         ;CHECK ADDRESS IN TABLE
005600  003005                              BGT     4$              ;OK, GO TO TEST
005602  001767                              BEQ     2$              ;ZERO, SKIP TEST
005604  012737  000020  001126              MOV     #20,$TSTNM      ;END OF SEQ. - RESET TEST # TO 20
005612  000765                              BR      3$              ;TEST NEW NUMBER
005614  012706  001100              4$:     MOV     #STACK,SP       ;RESET STACK
005620  000170  011564                      JMP     @TAT(R0)        ;GO TO TEST
```

```
                                        ;*************************************************************

                                        .SBTTL  ROUTINES TO SELECT DESIRED TEST



                                        ;//////////////////////////////////////////////////////////////

                                        ;ROUTINE TO SELECT TEST FROM SW REG, BITS 0-5

                                        ;//////////////////////////////////////////////////////////////

        005624  004737  005466          SELECT: JSR     PC,SRDCI        ;CLEAR CLOCK INTER
        005630  105037  001132                  CLRB    STRONE          ;CLEAR PROGRAM CONTROL FLAGS
        005634  105037  001133                  CLRB    TRONE
        005640  105037  001134                  CLRB    TLOOP
        005644  004737  006012                  JSR     PC,KYBDF        ;CHECK IF KYBD FLAG
        005650  000000                  1$:     HALT                    ;NO KYBD FLAG - HALT
                                                                        ;WAIT FOR OPERATOR TO SELECT TEST
                                                                        ;PRESS CONTINUE WHEN READY
        005652  032777  000400  173242          BIT     #SW8,@SWR       ;CHECK SW8
        005660  001403                          BEQ     2$              ;WANT TO RUN TEST ONCE & HALT
        005662  112737  177777  001132          MOVB    #-1,STRONE      ;YES, SET FLAG
        005670  017700  173226          2$:     MOV     @SWR,R0         ;NO, CHECK SW REG
        005674  042700  177700                  BIC     #↑C77,R0        ;MASK BITS 0-5
        005700  010037  001126                  MOV     R0,$TSTNM       ;SAVE TEST NUMBER
        005704  006300                          ASL     R0              ;SET TABLE POINTER
        005706  005760  011564                  TST     TAT(R0)         ;CHECK IF TEST IS IN TABLE
        005712  003744                          BLE     SELECT          ;NOT THERE, GET NEW SELECTION
        005714  012706  001100                  MOV     #STACK,SP       ;RESET STACK
        005720  000170  011564                  JMP     @TAT(R0)        ;GO TO SELECTED TEST

                                        ;//////////////////////////////////////////////////////////////

                                        ;ROUTINE TO CHECK FOR KYBD OR SW REG CONTROL
                                        ;CALL:  CHECK

                                        ;//////////////////////////////////////////////////////////////

        005724  004737  006012          $CHECK: JSR     PC,KYBDF        ;CHECK FOR KYBD FLAG
        005730  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
        005732  005000                          CLR     R0              ;CLEAR R0
        005734  032777  000400  173160          BIT     #SW8,@SWR       ;CHECK SW8
        005742  001401                          BEQ     1$              ;BRANCH IF SW DOWN
        005744  005300                          DEC     R0              ;SET FLAG IF UP
        005746  123700  001132          1$:     CMPB    STRONE,R0       ;CHANGE IN SWITCH?
        005752  001402                          BEQ     2$              ;NO, RETURN
        005754  000137  005624                  JMP     SELECT          ;YES,  SELECT TEST
        005760                          2$:
        005760  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
        005762  000002                          RTI                     ;RETURN
```

```
1402                                    ;////////////////////////////////////////////////////////////////
1403
1404                                    ;ROUTINE TO WAIT FOR OPERATOR ACTION
1405
1406                                    ;////////////////////////////////////////////////////////////////
1407
1408  005764  105737  001153    $HOLD:  TSTB    $TPFLG          ;TERMINAL THERE?
1409  005770  100002                    BPL     1$              ;BRANCH IF YES
1410  005772  000000                    HALT                    ;HALT IF NO
1411  005774  000435                    BR      2$              ;RETURN ON CONTINUE SWITCH
1412  005776  104400  013352    1$:     TYPE    ,WTMSG          ;TYPE WAIT MESG
1413  006002  105777  173072    3$:     TSTB    @TKS            ;KYBD FLAG?
1414  006006  100375                    BPL     3$              ;NO, WAIT FOR FLAG
1415  006010  000002            2$:     RTI                     ;RETURN
1416
1417
1418                                    ;////////////////////////////////////////////////////////////////
1419
1420                                    ;ROUTINE TO CHECK FOR KEYBOARD FLAGS
1421                                    ;WHEN LOOKING FOR CONTROL FROM THE CONSOLE DEVICE KEYBOARD
1422                                    ;CALL:  JSR     PC,KYBDF
1423
1424                                    ;////////////////////////////////////////////////////////////////
1425
1426  006012  105737  001153    KYBDF:  TSTB    $TPFLG          ;TERMINAL THERE?
1427  006016  001121                    BNE     1$              ;NO, EXIT
1428  006020  105777  173054            TSTB    @TKS            ;FLAG SET?
1429  006024  100116                    BPL     1$              ;NO - EXIT
1430  006026  104405            2$:     RDCHR                   ;FLAG SET - READ CHAR
1431  006030  023727  001122  000176    CMP     SWR,#SWREG      ;USING HARDWARE SW REG?
1432  006036  001101                    BNE     4$              ;BRANCH IF YES
1433  006040  022716  000007            CMP     #7,(SP)         ;CHAR = BEL <007>?
1434  006044  001076                    BNE     4$              ;BRANCH IF NOT
1435  006046  005726            5$:     TST     (SP)+           ;RESET STACK
1436  006050  005037  006264            CLR     20$             ;CLEAR NEW SW SETTING
1437  006054  005037  006266            CLR     30$             ;CLEAR INPUT FLAG
1438  006060  104400  013330            TYPE    ,DSMSG1         ;TYPE MESG
1439  006064  013746  000176            MOV     $WREG,-(SP)     ;;PUSH SWREG ON STACK
1440  006070  104401                    TYPOC                   ;TYPE IT
1441  006072  104400  013340            TYPE    ,DSMSG2         ;TYPE MORE OF MESG
1442  006076  104405            9$:     RDCHR                   ;READ CHAR
1443  006100  021627  000025            CMP     (SP),#25        ;CHAR = CONTROL-U ?
1444  006104  001003                    BNE     10$             ;BRANCH IF NOT
1445  006106  104400  010534            TYPE    ,$CNTLU         ;ECHO CONTROL-U
1446  006112  000755                    BR      5$              ;RESTART ROUTINE
1447  006114  021627  000015    10$:    CMP     (SP),#CR        ;CHAR = CR?
1448  006120  001011                    BNE     7$              ;BRANCH IF NOT
1449  006122  104400  001141            TYPE    ,$CRLF          ;ECHO CR-LF
1450  006126  005737  006266            TST     30$             ;CHECK INPUT FLAG
1451  006132  001452                    BEQ     6$              ;LEAVE SW SETTINGS ALONE IF NO INPUT
1452  006134  013737  006264  000176    MOV     20$,SWREG       ;SET NEW SW REG
1453  006142  000446                    BR      6$              ;RETURN TO TEST
1454  006144  021627  000012    7$:     CMP     (SP),#LF        ;CHAR = LF?
1455  006150  001011                    BNE     8$              ;BRANCH IF NOT
```

```
1456  006152  104400  001141                    TYPE    ,$CRLF          ;ECHO CR-LF
1457  006156  005737  006266                    TST     30$             ;CHECK INPUT FLAG
1458  006162  001465                           BEQ     TSEL            ;LEAVE SW SETTINGS ALONE IF NO INPUT
1459  006164  013737  006264  000176           MOV     20$,SWREG       ;SET NEW SW REG
1460  006172  000461                           BR      TSEL            ;GO TO TEST SELECT
1461  006174  042716  177770          8$:     BIC     #^C7,(SP)       ;MASK DIGIT
1462  006200  062716  000060                   ADD     #60,(SP)        ;MAKE ASCII
1463  006204  004737  007172                   JSR     PC,$TYPEC       ;PRINT DIGIT
1464  006210  042716  177770                   BIC     #^C7,(SP)       ;MASK DIGIT
1465  006214  006337  006264                   ASL     20$             ;SHIFT SWTICH SETTINGS FOR NEW ONE
1466  006220  006337  006264                   ASL     20$
1467  006224  006337  006264                   ASL     20$
1468  006230  062637  006264                   ADD     (SP)+,20$       ;ADD NEW SWITCH
1469  006234  005237  006266                   INC     30$             ;SET INPUT FLAG
1470  006240  000716                           BR      9$              ;CONTINUE
1471  006242  022716  000177          4$:     CMP     #177,(SP)       ;CHAR = RUBOUT?
1472  006246  001001                           BNE     3$              ;NO, CHECK AGAIN
1473  006250  000432                           BR      TSEL            ;YES, GET TEST SELECTION
1474  006252  022716  000003          3$:     CMP     #3,(SP)         ;CHAR = CNTL C ?
1475  006256  001404                           BEQ     KYBDST          ;YES, GET # COLUMNS
1476  006260  005726                   6$:     TST     (SP)+           ;RESET STACK
1477  006262  000207                   1$:     RTS     PC              ;NO, RETURN
1478
1479  006264  000000                   20$:    .WORD   0               ;SW SETTING INPUT
1480  006266  000000                   30$:    .WORD   0               ;INPUT FLAG
```

```
1481                                    ;////////////////////////////////////////////////////////////////
1482
1483                                    ;ROUTINE TO SET NUMBER OF COLUMNS FROM CONSOLE DEVICE KYBD
1484
1485                                    ;////////////////////////////////////////////////////////////////
1486
1487  006270  004737  005466    KYBDST: JSR     PC,SRDCI        ;CLEAR CLOCK INTER
1488  006274  104400  013157            TYPE    ,COLUMN         ;TYPE COLUMNS MESSAGE
1489  006300  104407            2$:     RDDEC                   ;GET # COLUMNS
1490  006302  012605                    MOV     (SP)+,R5        ;;POP STACK INTO R5
1491  006304  020527  000204            CMP     R5,#132.        ;TEST SIZE OF NUMBER
1492  006310  003003                    BGT     1$              ;TOO BIG, GET AGAIN
1493  006312  020527  000002            CMP     R5,#2           ;TEST SIZE AGAIN
1494  006316  103005                    BHIS    3$              ;BRANCH IF OK
1495  006320  104400  001140    1$:     TYPE    ,$QUES          ;INPUT ERROR - TYPE QUESTION MARK
1496  006324  104400  001141            TYPE    ,$CRLF
1497  006330  000763                    BR      2$              ;GET INPUT AGAIN
1498  006332  010537  001130    3$:     MOV     R5,WIDTH        ;OK, SAVE # COLUMNS
1499
1500                                    ;////////////////////////////////////////////////////////////////
1501
1502                                    ;ROUTINE TO SELECT TEST FROM CONSOLE DEVICE KEYBOARD
1503                                    ;AND DETERMINE TEST ACTION BY INPUT CONTROL CHARACTER.
1504                                    ;TEST NUMBER MUST BE OCTAL, FOLLOWED BY ONE OF THE
1505                                    ;FOLLOWING CONTROL CHARACTERS:
1506                                    ;
1507                                    ;       PERIOD  .       =       RUN TEST ONCE AND SELECT NEXT TEST
1508                                    ;
1509                                    ;               L       =       LOOP ON SELECTED TEST
1510                                    ;
1511                                    ;               S       =       START TEST SEQUENCE WITH SELECTED TEST
1512
1513                                    ;////////////////////////////////////////////////////////////////
1514
1515  006336  004737  005466    TSEL:   JSR     PC,SRDCI        ;CLEAR CLOCK INTER
1516  006342  105037  001133            CLRB    TRONE           ;CLEAR PROGRAM CONTROL FLAGS
1517  006346  105037  001134            CLRB    TLOOP
1518  006352  105037  001132            CLRB    STRONE
1519  006356  104400  013175            TYPE    ,SELTST         ;TYPE SELECT TEST MESG
1520  006362  104406            5$:     RDLIN                   ;GET TEST SELECTION
1521  006364  012605                    MOV     (SP)+,R5        ;;POP STACK INTO R5
1522  006366  112500                    MOVB    (R5)+,R0        ;SET TEST # IN R0
1523  006370  042700  177600            BIC     #^C177,R0
1524  006374  022700  000003            CMP     #3,R0           ;CHECK IF CHAR = CNTL-C
1525  006400  001733                    BEQ     KYBDST          ;GET # COLUMNS IF CNTL-C
1526  006402  020027  000060            CMP     R0,#60          ;CHECK IF OCTAL NUMBER
1527  006406  103464                    BLO     4$              ;NOT OCTAL - INPUT ERRO
1528  006410  020027  000067            CMP     R0,#67
1529  006414  101061                    BHI     4$              ;NOT OCTAL - INPUT ERROR
1530  006416  042700  177770            BIC     #^C7,R0         ;OK - STORE DIGIT
1531  006422  006300                    ASL     R0
1532  006424  006300                    ASL     R0
1533  006426  006300                    ASL     R0
1534  006430  112501                    MOVB    (R5)+,R1        ;GET SECOND DIGIT
```

```
1535  006432  042701  177600              BIC     #^C177,R1       ;CHECK IF AN OCTAL DIGIT
1536  006436  020127  000060              CMP     R1,#60
1537  006442  103446                      BLO     4$              ;NOT OCTAL - INPUT ERPOR
1538  006444  020127  000067              CMP     R1,#67
1539  006450  101043                      BHI     4$              ;NOT OCTAL - INPUT ERROR
1540  006452  042701  177770              BIC     #^C7,R1         ;OK - MASK DIGIT
1541  006456  060100                      ADD     R1,R0           ;MAKE COMPLETE TEST NUMBER
1542  006460  010037  001126              MOV     R0,$TSTNM       ;SAVE TEST NUMBER
1543  006464  006300                      ASL     R0              ;SET TABLE POINTER
1544  006466  005760  011564              TST     TAT(R0)         ;CHECK IF TEST IS IN TABLE
1545  006472  003432                      BLE     4$              ;NOT IN TABLE, GET NEW SELECTION
1546  006474  112501                      MOVB    (R5)+,R1        ;GET CONTROL CHAR
1547  006476  042701  177600              BIC     #^C177,R1       ;MASK BITS
1548  006502  020127  000056              CMP     R1,#56          ;CHAR = PERIOD?
1549  006506  001004                      BNE     1$              ;NO, CONTINUE CHECK
1550  006510  112737  177777  001133      MOVB    #-1,TRONE       ;YES, SET FLAG
1551  006516  000414                      BR      3$              ;CONTINUE
1552  006520  042701  000040      1$:     BIC     #BIT5,R1                ;MASK BIT 5 (ALLOW UPPER OR LOWER CASE)
1553  006524  022701  000114              CMP     #'L,R1          ;CHAR = L?
1554  006530  001004                      BNE     2$              ;NO, CONTINUE
1555  006532  112737  177777  001134      MOVB    #-1,TLOOP       ;YES, SET FLAG
1556  006540  000403                      BR      3$              ;CONTINUE
1557  006542  022701  000123      2$:     CMP     #'S,R1          ;CHAR = S?
1558  006546  001004                      BNE     4$              ;NO, INPUT ERROR
1559  006550  012706  001100      3$:     MOV     #STACK,SP       ;RESET STACK
1560  006554  000170  011564              JMP     @TAT(R0)        ;ALL OK - GO TO SELECTED TEST
1561  006560  104400  001140      4$:     TYPE    .$QUES          ;INPUT ERROR
1562  006564  104400  001141              TYPE    .$CRLF          ;PRINT QUESTION MARK
1563  006570  000674                      BR      5$              ;GET NEW INPUT
```

```
1564                                    ;*****************************************************************
1565
1566                                    .SBTTL  ERROR HANDLER ROUTINE
1567
1568                                    ;THIS ROUTINE WILL SAVE THE ERROR NUMBER AND THE ADR OF THE ERROR CALL
1569                                    ;AND GO TO REPORT ON ERROR
1570                                    ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1571                                    ;       SW15 = 1        HALT ON ERROR
1572                                    ;       SW13 = 1        INHIBIT ERROR TYPEOUTS
1573                                    ;       SW10 = 1        BELL ON ERROR
1574                                    ;
1575                                    ;CALL
1576                                    ;       ERROR   N               ;ERROR = EMT & N = ERROR ITEM NUMBER
1577
1578  006572  013777  001126  172324  $ERROR: MOV     $TSTNM,@DISPLAY;DISPLAY TEST NUMBER AND ERROR FLAG
1579  006600  032777  002000  172314          BIT     #BIT10,@SWR     ;BELL ON ERROR?
1580  006606  001402                          BEQ     1$              ;NO - SKIP
1581  006610  104400  001136                  TYPE    ,$BELL          ;RING BELL
1582  006614  011637  006666          1$:     MOV     (SP),$ERRPC     ;GET ADDRESS OF ERROR INSTRUCTION
1583  006620  162737  000002  006666          SUB     #2,$ERRPC
1584  006626  117737  000034  006670          MOVB    @$ERRPC,$ITEMB  ;STRIP AND SAVE THE ERROR ITEM CODE
1585  006634  032777  020000  172260          BIT     #BIT13,@SWR     ;SKIP TYPEOUTS IF SET
1586  006642  001004                          BNE     2$              ;SKIP TYPEOUTS
1587  006644  004737  006672                  JSR     PC,REPORT       ;GO TO REPORT ROUTINE
1588  006650  104400  001141                  TYPE    ,$CRLF
1589  006654  005777  172242          2$:     TST     @SWR            ;HALT ON ERROR IF SET
1590  006660  100001                          BPL     3$              ;SKIP IS CONTINUE
1591  006662  104420                          HOLD                    ;DYNAMIC HALT
1592  006664  000002          3$:     RTI                             ;RETURN
1593
1594  006666  000000          $ERRPC: .WORD   0               ;LAST ERROR INSTRUCTION EXECUTED
1595  006670  000000          $ITEMB: .WORD   0               ;ITEM CODE
```

```
1596                                    ;////////////////////////////////////////////////////////////////////////
1597
1598                                    ;ERROR REPORT ROUTINE
1599                                    ;
1600                                    ;ERROR MESSAGE WILL USE THE FOLLOWING FORM:
1601                                    ;TEST #XX,  PC=XXXXXX,  ERROR #XXX,  MESSAGE >>>>>>>>>>>>>
1602
1603                                    ;////////////////////////////////////////////////////////////////////////
1604
1605  006672  105737  001153          REPORT: TSTB    $TPFLG          ;TERMINAL EXIT?
1606  006676  001035                          BNE     2$              ;NO, EXIT
1607  006700  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1608  006702  104400  013252                  TYPE    ,ETSTNO         ;TYPE FIRST PART OF ERROR MESG
1609  006706  013746  001126                  MOV     $TSTNM,-(SP)    ;;PUSH $TSTNM ON STACK
1610  006712  104402                          TYPOS                   ;TYPE TEST NUMBER
1611  006714     002                          .BYTE   2               ;TWO DIGITS MAX.
1612  006715     001                          .BYTE   1               ;TYPE LEADING ZEROS
1613  006716  104400  013262                  TYPE    ,PCMSG          ;TYPE PART OF ERROR MESG
1614  006722  013746  006666                  MOV     $ERRPC,-(SP)    ;;PUSH $ERRPC ON STACK
1615  006726  104401                          TYPOC                   ;TYPE ERROR PC
1616  006730  104400  013271                  TYPE    ,ERR            ;TYPE MORE OF ERROR MESG
1617  006734  013746  006670                  MOV     $ITEMB,-(SP)    ;;PUSH $ITEMB ON STACK
1618  006740  104402                          TYPOS                   ;TYPE ERROR NUMBER
1619  006742     003                          .BYTE   3               ;3 DIGITS MAX.
1620  006743     001                          .BYTE   1               ;TYPE LEADING ZEROS
1621  006744  104400  013304                  TYPE    ,ERRS           ;TYP SPACES
1622  006750  013700  006670                  MOV     $ITEMB,R0       ;GET ERROR NUMBER
1623  006754  006300                          ASL     R0              ;SET TABLE POINTER
1624  006756  016037  011762  006766          MOV     EMAT-2(R0),1$   ;SET ERROR MESSAGE ADR
1625  006764  104400                          TYPE                    ;TYPE ERROR MESG
1626  006766  000000                  1$:     .WORD   0
1627  006770  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1628  006772  000207                  2$:     RTS     PC              ;RETURN
```

```
                                ;*****************************************************************

                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

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

006774                          $SAVREG:
006774  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
006776  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
007000  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
007002  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
007004  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
007006  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
007010  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
007014  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
007020  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
007024  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
007030  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
007032                          $RESREG:
007032  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
007036  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
007042  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
007046  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
007052  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
007054  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
007056  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
007060  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
007062  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
007064  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
007066  000002                          RTI
```

```
                                        ;*****************************************************

                                        .SBTTL  TYPE ROUTINE

                                        ;ROUTINE TO TYPE ASCIZ MESSAGES, MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER
                                        ;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;
                                        ;CALL:
                                        ;
                                        ;       TYPE    ,MESADR         ;MESADR IS ADDRESS OF FIRST CHAR IN ASCIZ STRING

007070  105737  001153          $TYPE:  TSTB    $TPFLG          ;IS THERE A TERMINAL?
007074  100407                          BMI     3$              ;LEAVE IF NO TERMINAL
007076  010046                  1$:     MOV     R0,-(SP)        ;SAVE R0
007100  017600  000002                  MOV     @2(SP),R0       ;GET ADR OF ASCIZ STRING
007104  112046                  2$:     MOVB    (R0)+,-(SP)     ;PUSH CHARACTER TO BE TYPED ONTO STACK
007106  001005                          BNE     4$              ;BR IF IT ISN'T THE TERMINATOR
007110  005726                          TST     (SP)+           ;IF TERMINATOR, POP IT OFF STACK
007112  012600                          MOV     (SP)+,R0        ;RESTORE R0
007114  062716  000002          3$:     ADD     #2,(SP)         ;ADJUST RETURN PC
007120  000002                          RTI                     ;RETURN
007122  122716  000200          4$:     CMPB    #CRLF,(SP)      ;BRANCH IF NOT <CRLF>
007126  001004                          BNE     5$
007130  005726                          TST     (SP)+           ;POP <CR><LF> EQUIV
007132  104400  001141                  TYPE,   $CRLF           ;TYPE CR-LF
007136  000762                          BR      2$              ;GET NEXT CHAR
007140  004737  007172          5$:     JSR     PC,$TYPEC       ;GO TYPE CHAR
007144  123726  001152          6$:     CMPB    $FILLC,(SP)+    ;IS IT TIME FOR FILLER CHARS?
007150  001355                          BNE     2$              ;IF NO GO GET NEXT CHAR
007152  013746  001150                  MOV     $NULL,-(SP)     ;GET # FILLER CHARS NEEDED
                                                                ;AND THE NULL CHAR
007156  105366  000001          7$:     DECB    1(SP)           ;DOES A NULL NEED TO BE TYPED?
007162  002770                          BLT     6$              ;BR IF NO - GO POP THE NULL OFF OF STACK
007164  004737  007172                  JSR     PC,$TYPEC       ;GO TYPE NULL
007170  000772                          BR      7$              ;LOOP

007172  105777  171706          $TYPEC: TSTB    @$TPS           ;WAIT UNTIL PRINTER IS READY
007176  100375                          BPL     $TYPEC
007200  116677  000002  171700          MOVB    2(SP),@$TPB     ;LOAD CHAR TO BE TYPED INTO DATA REG.
007206  000207                          RTS     PC              ;RETURN
```

```
                                ;*****************************************************************

                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

007210                          $TYPDS:
007210  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
007212  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
007214  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
007216  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
007220  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
007222  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
007226  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
007232  100004                          BPL     1$              ;;BR IF INPUT IS POS.
007234  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
007236  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
007244  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
007246  012703  007424                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
007252  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
007256  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
007260  016001  007414                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
007264  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
007266  002402                          BLT     4$              ;;BR IF DONE
007270  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
007272  000774                          BR      3$
007274  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
007276  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
007300  001002                          BNE     5$              ;;FALL THROUGH IF 0
007302  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
007304  100407                          BMI     7$              ;;BR IF YES
007306  106316                  5$:     ASLB    (SP)            ;;MSD?
007310  103003                          BCC     6$              ;;BR IF NO
007312  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
007320  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
007324  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
007330  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
007332  005720                          TST     (R0)+           ;;JUST INCREMENTING
007334  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
007340  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
007342  003002                          BGT     8$              ;;GO TO EXIT
007344  010502                          MOV     R5,R2           ;;GET THE LSD
007346  000764                          BR      6$              ;;GO CHANGE TO ASCII
007350  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
007352  100003                          BPL     9$              ;;BR IF NO
007354  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
007362  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
```

```
1772  007364  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
1773  007366  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
1774  007370  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
1775  007372  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
1776  007374  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1777  007376  104400  007424                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
1778  007402  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
1779  007410  012616                          MOV     (SP)+,(SP)
1780  007412  000002                          RTI                     ;;RETURN TO USER
1781  007414  023420                  $DTBL:  10000.
1782  007416  001750                          1000.
1783  007420  000144                          100.
1784  007422  000012                          10.
1785  007424  000004                  $DBLK:  .BLKW   4
```

```
                        ;*********************************************************
                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                        ;*      .BYTE   M               ;;M=1 OR 0
                        ;*                                      ;;1=TYPE LEADING ZEROS
                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                        ;*
                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                        ;*$TYPOS OR $TYPOC
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                        ;*
                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
                        
007434  017646  000000  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
007440  116637  000001  007657  MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
007446  112637  007661          MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
007452  062716  000002          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
007456  000406                  BR      $TYPON
007460  112737  000001  007657  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
007466  112737  000006  007661          MOVB    #6,$OMODE+1     ;;SET FOR SIX.6) DIGITS
007474  112737  000005  007656  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
007502  010346                  MOV     R3,-(SP)        ;;SAVE R3
007504  010446                  MOV     R4,-(SP)        ;;SAVE R4
007506  010546                  MOV     R5,-(SP)        ;;SAVE R5
007510  113704  007661          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
007514  005404                  NEG     R4
007516  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
007522  110437  007660          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
007526  113704  007657          MOVB    $OFILL,R4       ;;GET THE ZERO FIL_ SWITCH
007532  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
007536  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
007540  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
007542  000404                  BR      3$              ;;GO DO MSB
007544  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
007546  006105                  ROL     R5
007550  006105                  ROL     R5
007552  010503                  MOV     R5,R3
007554  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
007556  105337  007660          DECB    $OMODE          ;;TYPE THIS DIGIT?
007562  100016                  BPL     7$              ;;BR IF NO
007564  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
```

```
 1940  007570  001002                        BNE     4$              ;;TEST FOR 0
 1941  007572  005704                        TST     R4              ;;SUPPRESS THIS 0?
 1942  007574  001403                        BEQ     5$              ;;BR IF YES
 1943  007576  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 1944  007600  052703  000060              5$:     BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 1945  007604  052703  000040                      BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREAD
 1946  007610  110337  007654                      MOVB    R3,8$           ;;SAVE FOR TYPING
 1947  007614  104430  007654                      TYPE    ,8$             ;;GO TYPE THIS DIGIT
 1948  007620  105337  007656                7$:     DECB    $CONT           ;;COUNT BY 1
 1949  007624  003347                        BGT     2$              ;;BR IF MORE TO DO
 1950  007626  002402                        BLT     6$              ;;BR IF DONE
 1951  007630  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 1952  007632  000744                        BR      2$              ;;GO DO THE LAST DIGIT
 1953  007634  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
 1954  007636  012604                        MOV     (SP)+,R4        ;;RESTORE P4
 1955  007640  012603                        MOV     (SP)+,R3        ;;RESTORE R3
 1956  007642  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 1957  007650  012616                        MOV     (SP)+,(SP)
 1958  007652  000002                        RTI                     ;;RETURN
 1959  007654  000000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 1960  007655  000000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 1961  007656  000000                $CONT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 1962  007657  000000                $CFILL: .BYTE   0               ;;ZERO FILL SWITCH
 1963  007660  000000                $CMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                ;*****************************************************************

                                .SBTTL  ROUTINES TO LOAD CHARACTERS TO LA180

                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

                                ;ROUTINE TO LOAD SINGLE CHARACTERS TO LA180 PRINTER
                                ;CALL:   MOV     CHAR,R0         ;PUT CHAR IN R0
                                ;       LOAD
                                ;
                                ;IF PRINTER DOES NOT GO READY WITHIN 4.4 TO 16.2 SECONDS
                                ;(DEPENDING ON CPU AND MEMORY TYPE) THE DIAGNOSTIC WILL
                                ;INDICATE AN ERROR - ERROR #24 - NOT READY.

                                ;-------------------------------------------------------------

007662  104417                  $LOAD:  CHECK                   ;CHECK FOR CONTROL
007664  012737  000020  007756          MOV     #16.,6$         ;SET DELAY LOOP COUNT
007672  005777  171212                  TST     @LPS            ;CHECK FOR ERROR CONDX
007676  100002                          BPL     1$              ;BRANCH IF OK
007700  104024                          ERROR   24              ;PRINTER ERROR BEFORE LOAD
007702  000423                          BR      4$              ;EXIT
007704  005037  007754          1$:     CLR     5$              ;CLEAR DELAY COUNT
007710  005237  007754          2$:     INC     5$              ;INC COUNT
007714  001005                          BNE     3$              ;COUNTINUE
007716  005337  007756                  DEC     6$              ;DEC DELAY LOOP COUNT
007722  001370                          BNE     1$              ;WAIT IF NOT READY
007724  104025                          ERROR   25              ;ERROR, PRINTER NOT READY
007726  000411                          BR      4$              ;EXIT
007730  105777  171154          3$:     TSTB    @LPS            ;CHECK FOR PRINTER READY
007734  100365                          BPL     2$              ;WAIT FOR READY
007736  110077  171150                  MOVB    R0,@LPB         ;LOAD CHAR
007742  005777  171142                  TST     @LPS            ;TEST FOR ERROR CONDX AGAIN
007746  100001                          BPL     4$              ;BRANCH IF OK
007750  104026                          ERROR   26              ;PRINTER ERROR AFTER LOAD
007752  000002                  4$:     RTI                     ;RETURN

007754  000000                  5$:     .WORD   0               ;DELAY COUNTER
007756  000020                  6$:     .WORD   16.             ;DELAY LOOP COUNTER
```

```
                                ;//////////////////////////////////////////////////////////////////////////////

                                ;ROUTINE TO LOAD MULTIPLE CHARACTERS (NOT ASCIZ/ASCII STRINGS)
                                ;WILL LOAD CHAR ONCE IF COUNT = 0
                                ;PUT CHARACTER COUNT IN R1 (POSITIVE)
                                ;PUT ASCII CHARACTER IN R0
                                ;CALL:  MLOAD

                                ;//////////////////////////////////////////////////////////////////////////////

007760  104414                  $MLOAD: LOAD                    ;LOAD CHAR
007762  005301                          DEC     R1              ;DEC COUNT
007764  003375                          BGT     $MLOAD          ;FINISH LOAD
007766  000002                          RTI                     ;RETURN


                                ;//////////////////////////////////////////////////////////////////////////////

                                ;ROUTINE TO LOAD CONVERTED NUMBERS AND SUPPRESS LEADING ZEROS
                                ;IF ALL DIGITS ARE ZERO, ROUTINE WILL PRINT A SINGLE ZERO
                                ;ENTER WITH ADDRESS OF ASCIZ STRING ON STACK
                                ;CALL:  CNLOAD

                                ;//////////////////////////////////////////////////////////////////////////////

007770                          $CNLD:
007770  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
007772  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
007774  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
007776  005002                          CLR     R2
010000  016601  000012                  MOV     12(SP),R1       ;GET ADR
010004  112100                  1$:     MOVB    (R1)+,R0        ;GET CHAR
010006  001410                          BEQ     3$              ;RETURN WHEN DONE
010010  005702                          TST     R2              ;DONE LEADING ZEROS?
010012  001004                          BNE     2$              ;YES, LOAD CHAR
010014  020027  000060                  CMP     R0,#60          ;NO, CHECK THIS CHAR
010020  001771                          BEQ     1$              ;SKIP IF ZERO
010022  005202                          INC     R2              ;SET FLAG IF NON-ZERO
010024  104414                  2$:     LOAD                    ;LOAD CHAR
010026  000766                          BR      1$              ;CONTINUE
010030  005702                  3$:     TST     R2              ;ALL CHARS ZERO?
010032  001003                          BNE     4$              ;NO, EXIT
010034  012700  000060                  MOV     #60,R0          ;YES, LOAD ZERO
010040  104414                          LOAD
010042                          4$:
010042  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
010044  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
010046  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
010050  016666  000002  000004          MOV     2(SP),4(SP)     ;ADJUST STACK
010056  012616                          MOV     (SP)+,(SP)
010060  000002                          RTI                     ;RETURN
```

```
                                        ;////////////////////////////////////////////////////////////////////////
                                        ;
                                        ;ROUTINE TO PRINT ASCIZ STRINGS ON LA180 PRINTER
                                        ;STRING MUST BE TERMINATED BY NULL BYTE
                                        ;CALL:          PRINT
                                        ;               MESADR
                                        ;
                                        ;////////////////////////////////////////////////////////////////////////
                                        ;
010062                                  $PRINT:
010062  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
010064  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
010066  017601  000004                          MOV     @4(SP),R1       ;GET ADDRESS OF ASCIZ STRING
010072  112100                          1$:     MOVB    (R1)+,R0        ;GET CHAR
010074  001407                                  BEQ     2$              ;BRANCH IF TERMINATOR (NULL)
010076  120027  000200                          CMPB    R0,#CRLF        ;CHECK IF CRLF CODE
010102  001002                                  BNE     3$              ;BRANCH IF NOT
010104  012700  000012                          MOV     #LF,R0          ;YES, DO LF ONLY
010110  104414                          3$:     LOAD                    ;LOAD CHAR
010112  000767                                  BR      1$              ;GET NEXT CHAR
010114                                  2$:
010114  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
010116  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
010120  062716  000002                          ADD     #2,(SP)         ;ADJUST RETURN PC
010124  000002                                  RTI                     ;RETURN
```

```
1980                                    ;////////////////////////////////////////////////////////////////////
1981
1982                                    .SBTTL  PRINT TEST HEADER
1983
1984                                    ;THE NUMBER OF COLUMNS WILL ALSO BE PRINTED FOR TEST 25 ONLY
1985                                    ;CALL:  PRTHDR
1986
1987                                    ;////////////////////////////////////////////////////////////////////
1988
1989  010126                            $PRHDR:
1990  010126  012746  000000                    MOV     #0,-(SP)        ;;PUT NEW PS ON STACK
1991  010132  012746  010140                    MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
1992  010136  000002                            RTI                     ;; POP NEW PC AND PS
1993  010140                            64$:
1994  010140  012700  000177                    MOV     #177,R0         ;SET RUBOUT CHAR
1995  010144  104414                            LOAD                    ;CLEAR LA180 CHAR BUFFER
1996  010146  023737  001126  010242            CMP     $TSTNM,SVTST    ;CHECK IF PRINTED THIS # LAST
1997  010154  001427                            BEQ     3$              ;YES,  PRINT BLANK LINE & EXI
1998  010156  013737  001126  010242            MOV     $TSTNM,SVTST    ;NO, STORE NEW #
1999  010164  104413  013217                    PRINT   ,TSTNO          ;LOAD TEST # MESG
2000  010170  013746  001126                    MOV     $TSTNM,-(SP)    ;;PUSH $TSTNM ON STACK
2001  010174  004737  011270                    JSR     PC,$SB20        ;CONVERT #
2002  010200  104416                            CNLOAD                  ;LOAD NUMBER
2003  010202  104413  001142                    PRINT   ,$LF            ;PRINT LINE
2004  010206  023727  001126  000025            CMP     $TSTNM,#25      ;IS THIS TEST 25?
2005  010214  001007                            BNE     3$              ;NO, SKIP COLUMN MESG
2006  010216  013746  001130                    MOV     WIDTH,-(SP)     ;PUT # COLUMNS ON STACK
2007  010222  004737  011120                    JSR     PC,$SB2D        ;CONVERT #
2008  010226  104416                            CNLOAD                  ;LOAD NUMBER
2009  010230  104413  013237                    PRINT   ,COLMN          ;LOAD CLOUMN MESG
2010  010234  104413  001142            3$:     PRINT   ,$LF            ;BLANK LINE
2011  010240  000002                    4$:     RTI                     ;RETURN
2012
2013  010242  000000            SVTST:  .WORD   0               ;SAVE TEST NUMBER
```

```
2014                                    ;*****************************************************************
2015
2016                                    .SBTTL  TTY INPUT ROUTINE
2017
2018                                    ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2019                                    ;*CALL:
2020                                    ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
2021                                    ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2022                                    ;
2023
2024  010244  011646                    $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
2025  010246  016666  000004  000002            MOV     4(SP),2(SP)     ;;SAVE THE PS
2026  010254  105777  170620            1$:     TSTB    @$TKS           ;;WAIT FOR
2027  010260  100375                            BPL     1$              ;;A CHARACTER
2028  010262  117766  170614  000004            MOVB    @$TKB,4(SP)     ;;READ THE TTY
2029  010270  042766  177600  000004            BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
2030  010276  000002                            RTI                     ;;GO BACK TO USER
2031                                    ;*****************************************************************
2032                                    ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2033                                    ;*CALL:
2034                                    ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
2035                                    ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2036                                    ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2037
2038  010300  010346                    $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
2039  010302  005046                            CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
2040  010304  012703  010541            1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
2041  010310  022703  010545            2$:     CMP     #$TTYIN+4,R3    ;;BUFFER FULL?
2042  010314  101456                            BLOS    4$              ;;BR IF YES
2043  010316  104405                            RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2044  010320  112613                            MOVB    (SP)+,(R3)      ;;GET CHARACTER
2045  010322  122713  000177                   CMPB    #177,(R3)       ;;IS IT A RUBOUT
2046  010326  001022                            BNE     5$              ;;BR IF NO
2047  010330  005716                            TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
2048  010332  001007                            BNE     6$              ;;BR IF NO
2049  010334  112737  000134  010532           MOVB    #'\,9$          ;;TYPE A BACK SLASH
2050  010342  104400  010532                    TYPE    ,9$
2051  010346  012716  177777                   MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
2052  010352  005303                    6$:     DEC     R3              ;;BACKUP BY ONE
2053  010354  020327  010541                   CMP     R3,#$TTYIN      ;;STACK EMPTY?
2054  010360  103434                            BLO     4$              ;;BR IF YES
2055  010362  111337  010532                   MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
2056  010366  104400  010532                    TYPE    ,9$             ;;GO TYPE
2057  010372  000746                            BR      2$              ;;GO READ ANOTHER CHAR.
2058  010374  005716                    5$:     TST     (SP)            ;;RUBOUT KEY SET?
2059  010376  001406                            BEQ     7$              ;;BR IF NO
2060  010400  112737  000134  010532           MOVB    #'\,9$          ;;TYPE A BACK SLASH
2061  010406  104400  010532                    TYPE    ,9$
2062  010412  005016                            CLR     (SP)            ;;CLEAR THE RUBOUT KEY
2063  010414  122713  000025            7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
2064  010420  001003                            BNE     8$              ;;BR IF NO
2065  010422  104400  010534                    TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
2066  010426  000726                            BR      1$              ;;GO START OVER
2067  010430  122713  000012            8$:     CMPB    #12,(R3)        ;;IS CHARACTER A "LF"?
```

```
2068  010434  001011                          BNE     3$              ;;BRANCH IF NO
2069  010436  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
2070  010440  104400  001141                  TYPE    .$CRLF          ;;TYPE A "CR" & "LF"
2071  010444  104400  010541                  TYPE    $TTYIN          ;;TYPE THE INPUT STRING
2072  010450  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
2073  010452  104400  001140          4$:     TYPE    $QUES           ;;TYPE A '?'
2074  010456  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
2075  010460  111337  010532          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2076  010464  104400  010532                  TYPE    9$
2077  010470  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2078  010474  001305                          BNE     2$              ;;LOOP IF NOT RETURN
2079  010476  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2080  010502  104400  001142                  TYPE    $LF             ;;TYPE A LINE FEED
2081  010506  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
2082  010510  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2083  010512  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2084  010514  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2085  010522  012766  010541  000004          MOV     #$TTYIN,4(SP)
2086  010530  000002                          RTI                     ;;RETURN
2087  010532     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
2088  010533     000                  .BYTE   0                       ;;TERMINATOR
2089  010534  052536  005015     000  $CNTLU: .ASCIZ   /↑U/<15><12>   ;;CONTROL "U"
2090  010541  000004                  $TTYIN: .BLKB    4              ;;RESERVE 4 BYTES FOR TTY INPUT
2091          010546                  .EVEN
```

```
                                ;*****************************************************************

                                .SBTTL  READ A DECIMAL NUMBER FROM THE TTY

                                ;*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:
                                ;*      RDDEC                   ;;READ A DECIMAL NUMBER
                                ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
                                ;
010546  011646          $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
010550  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
010556  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
010560  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
010562  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
010564  104406          1$:     RDLIN                   ;;READ AN ASCIZ LINE
010566  012600                  MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
010570  010037  010714          MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
010574  005046                  CLR     -(SP)           ;;CLEAR DATA WORD
010576  005002                  CLR     R2              ;;SIGN SET POSITIVE
010600  122710  000055          CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
010604  001001                  BNE     2$              ;;BR IF NO MINUS SIGN
010606  112002                  MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
010610  112001          2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
010612  001424                  BEQ     3$              ;;GET OUT IF ZERO
010614  122701  000060          CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
010620  003032                  BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
010622  122701  000071          CMPB    #'9,R1
010626  002427                  BLT     5$
010630  032716  170000          BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
010634  001024                  BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
010636  006316                  ASL     (SP)            ;;*2
010640  011646                  MOV     (SP),-(SP)      ;;SAVE FOR LATER
010642  006316                  ASL     (SP)            ;;*4
010644  006316                  ASL     (SP)            ;;*8.
010646  062616                  ADD     (SP)+,(SP)      ;;*10.
010650  102416                  BVS     5$              ;;OVERFLOW ISN'T ALLOWED
010652  162701  000060          SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
010656  060116                  ADD     R1,(SP)         ;;ADD IN THIS DIGIT
010660  102412                  BVS     5$              ;;OVERFLOW ISN'T ALLOWED
010662  000752                  BR      2$              ;;LOOP
010664  005702          3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
010666  001401                  BEQ     4$              ;;BR IF NO
010670  005416                  NEG     (SP)            ;;YES--NEGATE THE NUMBER
010672  012666  000012  4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
010676  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
010700  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
010702  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
010704  000002                  RTI                     ;;RETURN
```

```
 2146
 2147  010706  005726                  5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
 2148  010710  105010                          CLRB    (R0)            ;;SET A TERMINATOR
 2149  010712  104400                          TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
 2150  010714  000000                   $:     .WORD   0               ;;POINTER GOES HERE
 2151  010716  104400  001140                  TYPE    ,$QUES          ;;"?" "CR" &"LF"
 2152  010722  000720                          BR      1$              ;;TRY AGAIN
```

```
                                        ;*****************************************************************

                                        .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

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


010724  104410                  $DB2D:  SAVREG                  ;;SAVE REGISTERS
010726  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
010732  012700  011104                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
010736  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
010742  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
010744  012202                          MOV     (R2)+,R2
010746  012737  000012  011022          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
010754  012704  011034                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
010760  012705  011036                  MOV     #$TNPWR+2,R5
010764  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
010766  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
010770  005602                          SBC     R2
010772  161502                          SUB     (R5),R2
010774  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
010776  005203                          INC     R3              ;;ADD 1 TO PARTIAL
011000  000772                          BR      2$              ;;LOOP
011002  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
011004  005502                          ADC     R2
011006  062402                          ADD     (R4)+,R2
011010  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
011012  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
011016  110320                          MOVB    R3,(R0)+        ;;SAVE IT
011020  005327                          DEC     (PC)+           ;;DONE?
011022  000000                  4$:     .WORD   0
011024  001357                          BNE     1$              ;;BR IF NO
011026  105020                          CLRB    (R0)+           ;;TERMINATOR
011030  104411                          RESREG                  ;;RESTORE REGISTERS
011032  000207                          RTS     PC              ;;RETURN
011034  145000                  $TNPWR: 145000                  ;;1.0E09
011036  035632                          35632
011040  160400                          160400                  ;;1.0E08
011042  002765                          2765
011044  113200                          113200                  ;;1.0E07
011046  000230                          230
011050  041100                          041100                  ;;1.0E06
011052  000017                          17
011054  103240                          103240                  ;;1.0E05
011056  000001                          1
011060  023420                          23420                   ;;1.0E04
011062  000000                          0
```

```
2207  011064  001750                        1750                    ;;1.0E03
2208  011066  000000                        0
2209  011070  000144                        144                     ;;1.0E02
2210  011072  000000                        0
2211  011074  000012                        12                      ;;1.0E01
2212  011076  000000                        0
2213  011100  000001                        1                       ;;1.0E00
2214  011102  000000                        0
2215  011104  000014             $DECVL: .BLKB   12.                ;;RESERVE STORAGE FOR ASCIZ STRING
2216
2217                              ;*****************************************************************
2218
2219                              .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
2220
2221                              ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
2222                              ;*UNSIGNED DECIMAL ASCIZ NUMBER.
2223                              ;*CALL
2224                              ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
2225                              ;*      JSR     PC,@#$SB2D      ;;CALL
2226                              ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK
2227
2228
2229  011120  016637  000002  011144 $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
2230  011126  012746  011144                 MOV     #1$,-(SP)       ;;SET POINTER
2231  011132  004737  010724                 JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
2232  011136  012666  000002                 MOV     (SP)+,2(SP)     ;;PICKUP POINTER
2233  011142  000207                         RTS     PC              ;;RETURN
2234  011144  000000  000000          1$:    .WORD   0,0
```

```
2235                                    ;*****************************************************************
2236
2237                                    .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
2238
2239                                    ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
2240                                    ;*UNSIGNED OCTAL ASCIZ NUMBER.
2241                                    ;*CALL
2242                                    ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
2243                                    ;*      JSR     PC,@#$DB20      ;;CALL THE ROUTINE
2244                                    ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
2245
2246
2247  011150  104410                    $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
2248  011152  016501  000002                   MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
2249  011156  012705  011267                   MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
2250  011162  012704  000014                   MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
2251  011166  012703  177770                   MOV     #^C7,R3         ;;MASK
2252  011172  012100                           MOV     (R1)+,R0        ;;LOWER WORD
2253  011174  012101                           MOV     (R1)+,R1        ;;HIGH WORD
2254  011176  005002                           CLR     R2              ;;TERMINATOR
2255  011200  110245                    1$:    MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
2256  011202  010002                           MOV     R0,R2           ;;GET THIS DIGIT
2257  011204  005304                           DEC     R4              ;;COUNT THIS CHARACTER
2258  011206  003007                           BGT     3$              ;;BR IF NOT THE LAST DIGIT
2259  011210  001405                           BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
2260  011212  005205                           INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
2261  011214  010566  000002                   MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
2262  011220  104411                           RESREG                  ;;RESTORE ALL REGISTERS
2263  011222  000207                           RTS     PC              ;;RETURN TO USER
2264  011224  006203                    2$:    ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
2265  011226  006001                    3$:    ROR     R1              ;;POSITION THE BINARY NUMBER FOR
2266  011230  006000                           ROR     R0              ;;      THE NEXT OCTAL DIGIT
2267  011232  006001                           ROR     R1
2268  011234  006000                           ROR     R0
2269  011236  006001                           ROR     R1
2270  011240  006000                           ROR     R0
2271  011242  040302                           BIC     R3,R2           ;;MASK OUT ALL JUNK
2272  011244  062702  000060                   ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
2273  011250  000753                           BR      1$              ;;GO PUT IT IN THE DATA TABLE
2274  011252  000016                    $OCTVL: .BLKB   14.             ;;RESERVE DATA TABLE
```

```
                                        ;*****************************************************************
                                        .SBTTL  SINGLE LENGTH BINARY TO OCTAL ASCIZ ROUTINE
                                        ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                        ;*UNSIGNED OCTAL ASCIZ NUMBER.
                                        ;*CALL
                                        ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
                                        ;*      JSR     PC,@#$$B20      ;;CALL
                                        ;*      RETURN                  ;;ADDRESS OF 1ST ASCIZ CHAR. IS ON THE STACK

011270  016637  000002  011314  $$B20:  MOV     2(SP),I$        ;;SAVE THE BINARY NUMBER
011276  012746  011314                  MOV     #I$,-(SP)       ;;SET POINTER
011302  004737  011150                  JSR     PC,@#$$D20      ;;CALL DOUBLE LENGTH CONVERT ROUTINE
011306  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
011312  000207                          RTS     PC              ;;RETURN
011314  000000  000000          I$:     .WORD   0,0
```

```
                        ;*****************************************************************

                        .SBTTL  TRAP DECODER

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

011320  016646  000002  $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
011324  042716  000020          BIC     #20,(SP)        ;;  THE CALLER--DONOT ALLOW
011330  012746  011336          MOV     #1$,-(SP)       ;;  T-BIT TRAPS
011334  000002                  RTI                     ;;SET THE NEW STATUS
011336  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
011340  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
011344  005740                  TST     -(R0)           ;;BACKUP BY 2
011346  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
011350  022700  000042          CMP     #$TERM,R0       ;;CHECK FOR OUT OF BOUNDS
011354  003002                  BGT     .+6             ;;BR IF OK
011356  000000                  HALT                    ;;OUT OF BOUNDS
011360  000776                  BR      .-2             ;;HANGUP
011362  006300                  ASL     R0              ;;POSITION FOR INDEXING
011364  016000  011372          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
011370  000200                  RTS     R0              ;;GO TO ROUTINE


                        .SBTTL  TRAP TABLE

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

                        ;       ROUTINE
                        ;       -------
011372                  $TRPAD:
011372  007070                  $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPECUT ROUTINE
011374  007460                  $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
011376  007434                  $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
011400  007474                  $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
011402  007210                  $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
011404  010244                  $RDCHR  ;;CALL=RDCHR    TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
011406  010300                  $RDLIN  ;;CALL=RDLIN    TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
011410  010546                  $RDDEC  ;;CALL=RDDEC    TRAP+7(104407)  READ A DECIMAL NUMBER FROM TTY
011412  006774                  $SAVREG ;;CALL=SAVREG   TRAP+10(104410) SAVE R0-R5 ROUTINE
011414  007032                  $RESREG ;;CALL=RESREG   TRAP+11(104411) RESTORE R0-R5 ROUTINE
011416  010126                  $PRHDR  ;;CALL=PRTHDR   TRAP+12(104412) PRINT TEST HEADER
011420  010062                  $PRINT  ;;CALL=PRINT    TRAP+13(104413) PRINT ROUTINE
011422  007662                  $LOAD   ;;CALL=LOAD     TRAP+14(104414) LOAD CHAR ROUTINE
011424  007760                  $MLOAD  ;;CALL=MLOAD    TRAP+15(104415) MULTIPLE CHAR LOAD
011426  007770                  $CNLD   ;;CALL=CNLOAD   TRAP+16(104416) LOAD CONVERTED NUMBER
011430  005724                  $CHECK  ;;CALL=CHECK    TRAP+17(104417) CHECK FOR KYBD OR SW REG CONTROL
011432  005764                  $HOLD   ;;CALL=HOLD     TRAP+20(104420) WAIT FOR OPERATOR ACTION
000042          $TERM=.-$TRPAD
```

```
                                        ;*****************************************************************
                                        .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;POWER DOWN ROUTINE
011434  012737  011556  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
011442  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
011450  010046                          MOV     R0,-(SP)         ;;PUSH R0 ON STACK
011452  010146                          MOV     R1,-(SP)         ;;PUSH R1 ON STACK
011454  010246                          MOV     R2,-(SP)         ;;PUSH R2 ON STACK
011456  010346                          MOV     R3,-(SP)         ;;PUSH R3 ON STACK
011460  010446                          MOV     R4,-(SP)         ;;PUSH R4 ON STACK
011462  010546                          MOV     R5,-(SP)         ;;PUSH R5 ON STACK
011464  010637  011562                  MOV     SP,$SAVR6        ;;SAVE SP
011470  012737  011502  000024          MOV     #$PWRLP,@#PWRVEC ;;SET UP VECTOR
011476  000000                          HALT
011500  000776                          BR      .-2              ;;HANG UP
                                        ;POWER UP ROUTINE
011502  013706  011562          $PWRUP: MOV     $SAVR6,SP        ;;GET SP
011506  005037  011562                  CLR     $SAVR5           ;;WAIT LOOP FOR THE TTY
011512  005237  011562          1$:     INC     $SAVR6           ;;WAIT FOR THE INC
011516  001375                          BNE     1$               ;;OF  WORD
011520  012605                          MOV     (SP)+,R5         ;;POP STACK INTO R5
011522  012604                          MOV     (SP)+,R4         ;;POP STACK INTO R4
011524  012603                          MOV     (SP)+,R3         ;;POP STACK INTO R3
011526  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
011530  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
011532  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
011534  012737  011434  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
011542  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
011550  104400                          TYPE                     ;;REPORT THE POWER FAILURE
011552  013423          $PWRMG: .WORD   PWRMSG           ;;POWER FAIL MESSAGE POINTER
011554  000002                  RTI
011556  000000          $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
011560  000776                  BR      .-2              ;; BEFORE THE POWER DOWN WAS COMPLETE
011562  000000          $SAVR6: 0                        ;;PUT THE SP HERE
```

E07


```
                                        ;//////////////////////////////////////////////////////////////////

                                        .SBTTL  TEST ADDRESS TABLE

                                        ;THE PRINTING TEST SEQUENCE STARTS WITH TEST20
                                        ;TO REMOVE A TEST FROM THE PROGRAM -
                                        ;DEPOSIT ZERO FOR THE TEST ADDRESS IN THIS TABLE
                                        ;A MINUS ONE (-1) IN THE TABLE INDICATES THE END OF THE PRINTING SEQUENCE

                                        ;//////////////////////////////////////////////////////////////////

        011564  001720                  TAT:    TEST0
        011566  002526                          TEST1
        011570  002716                          TEST2
        011572  000000                          0               ;TEST3
        011574  000000                          0               ;TEST4
        011576  000000                          0               ;TEST5
        011600  000000                          0               ;TEST6
        011602  000000                          0               ;TEST7
        011604  000000                          0               ;TEST10
        011606  000000                          0               ;TEST11
        011610  000000                          0               ;TEST12
        011612  000000                          0               ;TEST13
        011614  000000                          0               ;TEST14
        011616  000000                          0               ;TEST15
        011620  000000                          0               ;TEST16
        011622  000000                          0               ;TEST17
        011624  003250                          TEST20
        011626  003316                          TEST21
        011630  003426                          TEST22
        011632  003514                          TEST23
        011634  003550                          TEST24
        011636  004042                          TEST25
        011640  004424                          TEST26
        011642  004516                          TEST27
        011644  004634                          TEST30
        011646  004662                          TEST31
        011650  177777                          -1              ;TEST32
        011652  000000                          0               ;TEST33
        011654  000000                          0               ;TEST34
        011656  000000                          0               ;TEST35
        011660  000000                          0               ;TEST36
        011662  000000                          0               ;TEST37
        011664  000000                          0               ;TEST40
        011666  000000                          0               ;TEST41
        011670  000000                          0               ;TEST42
        011672  000000                          0               ;TEST43
        011674  000000                          0               ;TEST44
        011676  000000                          0               ;TEST45
        011700  000000                          0               ;TEST46
        011702  000000                          0               ;TEST47
        011704  000000                          0               ;TEST50
        011706  000000                          0               ;TEST51
        011710  000000                          0               ;TEST52
```

```
2435  011712  000000                    0               ;TEST53
2436  011714  000000                    0               ;TEST54
2437  011716  000000                    0               ;TEST55
2438  011720  000000                    0               ;TEST56
2439  011722  000000                    0               ;TEST57
2440  011724  004774                    TEST60
2441  011726  005110                    TEST61
2442  011730  005222                    TEST62
2443  011732  005304                    TEST63
2444  011734  000000                    0               ;TEST64
2445  011736  000000                    0               ;TEST65
2446  011740  000000                    0               ;TEST66
2447  011742  000000                    0               ;TEST67
2448  011744  000000                    0               ;TEST70
2449  011746  000000                    0               ;TEST71
2450  011750  000000                    0               ;TEST72
2451  011752  000000                    0               ;TEST73
2452  011754  000000                    0               ;TEST74
2453  011756  000000                    0               ;TEST75
2454  011760  000000                    0               ;TEST76
2455  011762  000000                    0               ;TEST77
2456
2457
2458                          ;////////////////////////////////////////////////////////////////////////////
2459
2460                          .SBTTL  ERROR MESSAGE ADDRESS TABLE
2461:
2462                          ;////////////////////////////////////////////////////////////////////////////
2463
2464  011764  012040          EMAT:   ERR1
2465  011766  012067                  ERR2
2466  011770  012114                  ERR3
2467  011772  012152                  ERR4
2468  011774  012206                  ERR5
2469  011776  012241                  ERR6
2470  012000  012276                  ERR7
2471  012002  012325                  ERR10
2472  012004  012352                  ERR11
2473  012006  012376                  ERR12
2474  012010  012440                  ERR13
2475  012012  012474                  ERR14
2476  012014  012532                  ERR15
2477  012016  012566                  ERR16
2478  012020  012622                  ERR17
2479  012022  012657                  ERR20
2480  012024  012711                  ERR21
2481  012026  012745                  ERR22
2482  012030  013001                  ERR23
2483  012032  013040                  ERR24
2484  012034  013077                  ERR25
2485  012036  013121                  ERR26
```

```
                                        ;////////////////////////////////////////////////////////////////
                                        .SBTTL  ERROR MESSAGES
                                        ;////////////////////////////////////////////////////////////////

012040  051105  047522  020122  ERR1:   .ASCIZ  /ERROR CLEAR, POWER OFF/
012046  046103  040505  026122
012054  050040  053517  051105
012062  047440  043106     000
012067     122  040505  054504  ERR2:   .ASCIZ  /READY SET, POWER OFF/
012074  051440  052105  020054
012102  047520  042527  020122
012110  043117  000106
012114  051105  047522  020122  ERR3:   .ASCIZ  /ERROR CLEAR, PRINTER OFF LINE/
012122  046103  040505  026122
012130  050040  044522  052116
012136  051105  047440  043106
012144  046040  047111  000105
012152  042522  042101  020131  ERR4:   .ASCIZ  /READY SET, PRINTER OFF LINE/
012160  042523  026124  050040
012166  044522  052116  051105
012174  047440  043106  046040
012202  047111  000105
012206  051105  047522  020122  ERR5:   .ASCIZ  /ERROR SET, PRINTER ON LINE/
012214  042523  026124  050040
012222  044522  052116  051105
012230  047440  020116  044514
012236  042516     000
012241     122  040505  054504  ERR6:   .ASCIZ  /READY CLEAR, PRINTER ON LINE/
012246  041440  042514  051101
012254  020054  051120  047111
012262  042524  020122  047117
012270  046040  047111  000105
012276  051105  047522  020122  ERR7:   .ASCIZ  /ERROR CLEAR, PAPER OUT/
012304  046103  040505  026122
012312  050040  050101  051105
012320  047440  052125     000
012325     122  040505  054504  ERR10:  .ASCIZ  /READY SET, PAPER OUT/
012332  051440  052105  020054
012340  040520  042520  020122
012346  052517  000124
012352  051105  047522  020122  ERR11:  .ASCIZ  /ERROR DID NOT CLEAR/
012360  044504  020104  047516
012366  020124  046103  040505
012374  000122
012376  042522  042101  020131  ERR12:  .ASCIZ  /READY NOT SET AFTER ERROR CLEARED/
012404  047516  020124  042523
012412  020124  043101  042524
012420  020122  051105  047522
012426  020122  046103  040505
012434  042522  000104
012440  051105  047522  020122  ERR13:  .ASCIZ  /ERROR SET AFTER RESET INSTR/
012446  042523  020124  043101
```

```
2540  012454  042524  020122  042522
2541  012462  042523  020124  047111
2542  012470  052123  000122
2543  012474  042522  042101  020131  ERR14:  .ASCIZ  /READY CLEAR AFTER RESET INSTR/
2544  012502  046103  040505  020122
2545  012510  043101  042524  020122
2546  012516  042522  042523  020124
2547  012524  047111  052123  000122
2548  012532  042522  042101  020131  ERR15:  .ASCIZ  /READY SET AFTER CHAR LOADED/
2549  012540  042523  020124  043101
2550  012546  042524  020122  044103
2551  012554  051101  046040  040517
2552  012562  042504  000104
2553  012566  051105  047522  020122  ERR16:  .ASCIZ  /ERROR SET AFTER CHAR LOADED/
2554  012574  042523  020124  043101
2555  012602  042524  020122  044103
2556  012610  051101  046040  040517
2557  012616  042504  000104
2558  012622  042522  042101  020131  ERR17:  .ASCIZ  /READY NEVER SET, CHAR LOADED/
2559  012630  042516  042526  020122
2560  012636  042523  026124  041440
2561  012644  040510  020122  047514
2562  012652  042101  042105     000
2563  012657     105  051122  051117  ERR20:  .ASCIZ  /ERROR SET, INTERRUPT TEST/
2564  012664  051440  052105  020054
2565  012672  047111  042524  051122
2566  012700  050125  020124  042524
2567  012706  052123     000
2568  012711     122  040505  054504  ERR21:  .ASCIZ  /READY CLEAR, INTERRUPT TEST/
2569  012716  041440  042514  051101
2570  012724  020054  047111  042524
2571  012732  051122  050125  020124
2572  012740  042524  052123     000
2573  012745     120  044522  052116  ERR22:  .ASCIZ  /PRINTER INTER ABOVE LEVEL 3/
2574  012752  051105  044440  052116
2575  012760  051105  040440  047502
2576  012766  042526  046040  053105
2577  012774  046105  031440     000
2578  013001     116  020117  051120  ERR23:  .ASCIZ  /NO PRINTER INTER BELOW LEVEL 4/
2579  013006  047111  042524  020122
2580  013014  047111  042524  020122
2581  013022  042502  047514  020127
2582  013030  042514  042526  020114
2583  013036  000064
2584  013040  051120  047111  042524  ERR24:  .ASCIZ  /PRINTER ERROR BEFORE CHAR LOAD/
2585  013046  020122  051105  047522
2586  013054  020122  042502  047506
2587  013062  042522  041440  040510
2588  013070  020122  047514  042101
2589  013076     000
2590  013077     120  044522  052116  ERR25:  .ASCIZ  /PRINTER NOT READY/
2591  013104  051105  047040  052117
2592  013112  051040  040505  054504
2593  013120     000
```

```
2594  013121     120  044522  052116  ERR26:  .ASCIZ  /PRINTER ERROR AFTER CHAR LOAD/
2595  013126  051105  042440  051122
2596  013134  051117  040440  052106
2597  013142  051105  041440  040510
2598  013150  020122  047514  042101
2599  013156     000
2600
2601
2602
2603                          ;////////////////////////////////////////////////////////////////////////
2604
2605                          .SBTTL  PROGRAM MESSAGES
2606
2607                          ;////////////////////////////////////////////////////////////////////////
2608
2609  013157     200  020043  047503  COLUMN: .ASCIZ  <CRLF>/# COLUMNS = /
2610  013164  052514  047115  020123
2611  013172  020075     000
2612  013175     200  042523  042514  SELTST: .ASCIZ  <CRLF>/SELECT TEST #   /
2613  013202  052103  052040  051505
2614  013210  020124  020043  020040
2615  013216     000
2616  013217     012  052012  051505  TSTNO:  .ASCIZ  <LF><LF>/TEST NUMBER  /
2617  013224  020124  052516  041115
2618  013232  051105  020040     000
2619  013237     040  041440  046117  COLMN:  .ASCIZ  /  COLUMNS/<LF>
2620  013244  046525  051516  000012
2621  013252  052200  051505  020124  ETSTNO: .ASCIZ  <CRLF>/TEST #/
2622  013260  000043
2623  013262  020054  050040  036503  PCMSG:  .ASCIZ  /,  PC=/
2624  013270     000
2625  013271     054  020040  051105  ERR:    .ASCIZ  /,  ERROR #/
2626  013276  047522  020122  000043
2627  013304  020054  000040          ERRS:   .ASCIZ  /,  /
2628  013310  042412  042116  047440  PASMSG: .ASCIZ  <LF>/END OF PASS  #/
2629  013316  020106  040520  051523
2630  013324  020040  000043
2631  013330  051600  051127  036440  DSMSG1: .ASCIZ  <CRLF>/SWR = /
2632  013336  000040
2633  013340  020040  047040  053505  DSMSG2: .ASCIZ  /   NEW = /
2634  013346  036440  000040
2635  013352  040527  052111  047111  WTMSG:  .ASCIZ  /WAITING,  TYP SPACE TO CONTINUE/<CRLF>
2636  013360  026107  020040  054524
2637  013366  020120  050123  041501
2638  013374  020105  047524  041440
2639  013402  047117  044524  052516
2640  013410  100105     000
2641  013413     103  040510  020122  TCHAR:  .ASCIZ  /CHAR = /
2642  013420  020075     000
2643  013423     200  047520  042527  PWRMSG: .ASCIZ  <CRLF>/POWER/<CRLF>
2644  013430  100122     000
2645  013433     012  047516  041440  NCMSG:  .ASCIZ  <LF>/NO CONSOLE TERMINAL/<LF>
2646  013440  047117  047523  042514
2647  013446  052040  051105  044515
```

```
  2648  013454  040516  005114     000
  2649  013461     120  044522  052116  MANMSG: .ASCII  /PRINT SPEED MANUAL TIMING/<CRLF>
  2650  013466  051440  042520  042105
  2651  013474  046440  047101  040525
  2652  013502  020114  044524  044515
  2653  013510  043516     200
  2654  013513     120  052125  051440          .ASCII  /PUT SWITCH 12 UP TO START TIMING/<CRLF>
  2655  013520  044527  041524  020110
  2656  013526  031061  052440  020120
  2657  013534  047524  051440  040524
  2658  013542  052122  052040  046511
  2659  013550  047111  100107
  2660  013554  052520  020124  053523          .ASCIZ  /PUT SWITCH 12 DOWN AT END OF 1 MINUTE/<CRLF>
  2661  013562  052111  044103  030440
  2662  013570  020062  047504  047127
  2663  013576  040440  020124  047105
  2664  013604  020104  043117  030440
  2665  013612  046440  047111  052125
  2666  013620  100105     000
  2667  013623     200  051120  047111  PRSP1:  .ASCIZ  <CRLF>/PRINT SPEED IS /
  2668  013630  020124  050123  042505
  2669  013636  020104  051511  000040
  2670  013644  050101  051120  054117  PRSP2:  .ASCIZ  /APPROX. /
  2671  013652  020056     000
  2672  013655     040  046040  047111  PRSP3:  .ASCIZ  /  LINES/<57>/MINUTE , WITH /
  2673  013662  051505  046457  047111
  2674  013670  052125  020105  020054
  2675  013676  044527  044124  000040
  2676  013704  020040  044103  051101  PRSP4:  .ASCIZ  /  CHARS/<57>/LINE/<CRLF>
  2677  013712  027523  044514  042516
  2678  013720  000200
  2679  013722  052524  047122  050040  TOMSG0: .ASCIZ  /TURN POWER OFF & SET OFF LINE/<CRLF>
  2680  013730  053517  051105  047440
  2681  013736  043106  023040  051440
  2682  013744  052105  047440  043106
  2683  013752  046040  047111  100105
  2684  013760     000
  2685  013761     117  026113  052040  TOMSG1: .ASCIZ  /OK, TURN POWER ON/<CRLF>
  2686  013766  051125  020116  047520
  2687  013774  042527  020122  047117
  2688  014002  000200
  2689  014004  045517  020054  042523  TOMSG2: .ASCIZ  /OK, SET PRINTER TO ON-LINE/<CRLF>
  2690  014012  020124  051120  047111
  2691  014020  042524  020122  047524
  2692  014026  047440  026516  044514
  2693  014034  042516  000200
  2694  014040  045517  020054  051124  TOMSG3: .ASCIZ  /OK, TRY PAPER OUT SWITCH/<CRLF>
  2695  014046  020131  040520  042520
  2696  014054  020122  052517  020124
  2697  014062  053523  052111  044103
  2698  014070  000200
  2699  014072  045517  020054  042522  TOMSG4: .ASCIZ  /OK, RESTORE PRINTER TO ON-LINE/<CRLF>
  2700  014100  052123  051117  020105
  2701  014106  051120  047111  042524
```

```
2702  014114  020122  047524  047440
2703  014122  026516  044514  042516
2704  014130  000200
2705  014132  026455  026455  020055  T1MSG1: .ASCIZ  /----- /
2706  014140     000
2707  014141     045  047111  044103  T1MSG2: .ASCIZ  / INCH FORM FEED -----/<CR>
2708  014146  043040  051117  020115
2709  014154  042506  042105  026440
2710  014162  026455  026455  000015
2711  014170  042523  020124  047506  T1MSG3: .ASCIZ  /SET FORM FEED SWITCH TO  /
2712  014176  046522  043040  042505
2713  014204  020104  053523  052111
2714  014212  044103  052040  020117
2715  014220  000040
2716  014222  020040  047111  044103  T1MSG4: .ASCIZ  /  INCHES & DEPRESS TOF RESET SWITCH/<CRLF>
2717  014230  051505  023040  042040
2718  014236  050105  042522  051523
2719  014244  052040  043117  051040
2720  014252  051505  052105  051440
2721  014260  044527  041524  100110
2722  014266     000
2723  014267     116  020117  042515  T2EM:   .ASCIZ  /NO METHOD OF TIMING AVAILABLE/<CRLF>
2724  014274  044124  042117  047440
2725  014302  020106  044524  044515
2726  014310  043516  040440  040526
2727  014316  046111  041101  042514
2728  014324  000200
2729
2730
2731          000001                  .END
```

```
BIT0   = 000001          BIT00  = 000001          BIT01  = 000002          BIT02  = 000004
BIT03  = 000010          BIT04  = 000020          BIT05  = 000040          BIT06  = 000100
BIT07  = 000200          BIT08  = 000400          BIT09  = 001000          BIT1   = 000002
BIT10  = 002000          BIT11  = 004000          BIT12  = 010000          BIT13  = 020000
BIT14  = 040000          BIT15  = 100000          BIT2   = 000004          BIT3   = 000010
BIT4   = 000020          BIT5   = 000040          BIT6   = 000100          BIT7   = 000200
BIT8   = 000400          BIT9   = 001000          BPTVEC= 000014           CHECK  = 104417
CKFLAG  001135           CNLOAD= 104416           CNTR    005460           COLMN   013237
COLUMN  013157           CONTRL  001200           CR     = 000015          CRLF   = 000200
CSBR    001116           DCI     005430           DDISP  = 177570          DISPLA  001124
DISPRE  000174           DSMSG1  013330           DSMSG2  01334C           DSWR   = 177570
EMAT    011764           EMTVEC= 000030           ERR     013271           ERRS    013304
ERRVEC= 000004           ERR1    012040           ERR10   012325           ERR11   012352
ERR12   012376           ERR13   012440           ERR14   012474           ERR15   012532
ERR16   012566           ERR17   012622           ERR2    012067           ERR20   012657
ERR21   012711           ERR22   012745           ERR23   013001           ERR24   013040
ERR25   013077           ERR26   013121           ERR3    012114           ERR4    012152
ERR5    012206           ERR6    012241           ERR7    012276           ETSTNO  013252
EXIT    005516           FF     = 000014          HOLD   = 104420          IOTVEC= 000020
KYBDF   006012           KYBDST  006270           LF     = 000012          LKS     001120
LKSRV   005440           LOAD   = 104414          LPB     001112           LPS     001110
MANMSG  013461           MINCNT  005464           MLOAD  = 104415          NCMSG   013433
PASCNT  005106           PASMSG  013310           PC     =%000007          PCMSG   013262
PIRQ   = 177772          PIRQVE= 000240           PLKS    001114           PRINT  = 104413
PRSP1   013623           PRSP2   013644           PRSP3   013655           PRSP4   013704
PRTHDR= 104412           PR0    = 000000          PR1    = 000040          PR2    = 000100
PR3    = 000140          PR4    = 000200          PR5    = 000240          PR6    = 000300
PR7    = 000340          PS     = 177776          PSW    = 177776          PWRMSG  013423
PWRVEC= 000024           RDCHR  = 104405          RDDEC  = 104407          RDLIN  = 104406
REPORT  006672           RESREG= 104411           RESTRT  001170           RESVEC= 000010
R0     =%000000          R1     =%000001          R2     =%000002          R3     =%000003
R4     =%000004          R5     =%000005          R6     =%000006          R7     =%000007
SAVREG= 104410           SELECT  005624           SELTST  013175           SP     =%000006
SPD     003134           SRDCI   005466           STACK  = 001100          START   001154
STARTX  001206           STKLMT= 177774           STRONE  001132           SVTST   010242
SWR     001122           SWREG   000176           SW0    = 000001          SW00   = 000001
SW01   = 000002          SW02   = 000004          SW03   = 000010          SW04   = 000020
SW05   = 000040          SW06   = 000100          SW07   = 000200          SW08   = 000400
SW09   = 001000          SW1    = 000002          SW10   = 002000          SW11   = 004000
SW12   = 010000          SW13   = 020000          SW14   = 040000          SW15   = 100000
SW2    = 000004          SW3    = 000010          SW4    = 000020          SW5    = 000040
SW6    = 000100          SW7    = 000200          SW8    = 000400          SW9    = 001000
TAT     011564           TBITVE= 000014           TCHAR   013413           TERR    005420
TEST0   001720           TEST1   002526           TEST2   002716           TEST20  003250
TEST21  003316           TEST22  003426           TEST23  003514           TEST24  003550
TEST25  004042           TEST26  004424           TEST27  004516           TEST30  004634
TEST31  004662           TEST60  004774           TEST61  005110           TEST62  005222
TEST63  005304           TKB     001102           TKS     001100           TKVEC  = 000060
TLOOP   001134           TORTN   005462           TPB     001106           TPS     001104
TPVEC  = 000064          TRAPVE= 000034           TRONE   001133           TRTVEC= 000014
TSEL    006336           TSTNO   013217           TYPDS  = 104404          TYPE   = 104400
TYPOC  = 104401          TYPON  = 104403          TYPOS  = 104402          TOMSG0  013722
TOMSG1  013761           TOMSG2  014004           TOMSG3  014040           TOMSG4  014072
TIMSG1  014132           TIMSG2  014141           TIMSG3  014170           TIMSG4  014222
```

```
T2EM    014267          T2SP    003034          WIDTH   001130          WTMSG   013352
$BELL   001136          $CHECK  005724          $CNLD   007770          $CNTLU  010534
$CR     001144          $CRLF   001141          $DBLK   007424          $DB2D   010724
$DB20   011150          $DECVL  011104          $DOAGN  004766          $DTBL   007414
$ENDAD  004756          $ENDCT  004742          $EOP    004716          $EOPCT  004734
$ERROR  006572          $ERRPC  006666          $FF     001146          $FILLC  001152
$FILLS  001151          $GET42  004746          $HD   = 000000          $HOLD   005764
$ILLUP  011556          $ITEMB  006670          $LF     001142          $LOAD   007662
$MLOAD  007760          $NULL   001150          $OCNT   007656          $OCTVL  011252
$OMODE  007660          $PASS   004772          $PRHDR  010126          $PRINT  010062
$PWRDN  011434          $PWRMG  011552          $PWRUP  011502          $QUES   001140
$RDCHR  010244          $RDDEC  010546          $RDLIN  010300          $RDSZ = 000004
$RESRE  007032          $SAVRE  006774          $SAVR6  011562          $SB2D   011120
$SB20   011270          $SETUP= 000034          $STUP = 177777          $SVPC = 000200
$SWR  = 162000          $TERM = 000042          $TKB    001102          $TKS    001100
$TN   = 000000          $TNPWR  011034          $TPB    001106          $TPFLG  001153
$TPS    001104          $TRAP   011320          $TRP  = 000021          $TRPAD  011372
$TSTNM  001126          $TTYIN  010541          $TYPDS  007210          $TYPE   007070
$TYPEC  007172          $TYPOC  007460          $TYPON  007474          $TYPOS  007434
$CFILL  007657          .     = 014326
```

ERRORS DETECTED: 0

*DZLAEA,DZLAEA/SOL←DZLAEA
RUN-TIME: 51 23 0 SECONDS
CORE USED: 16K

Spooler runtime 10 Seconds, 304 KCS, 300 disk reads, 4 disk writes, 90 pages