LA180

PRINTER DIAGNOSTIC

MD-11-DZLAE-B

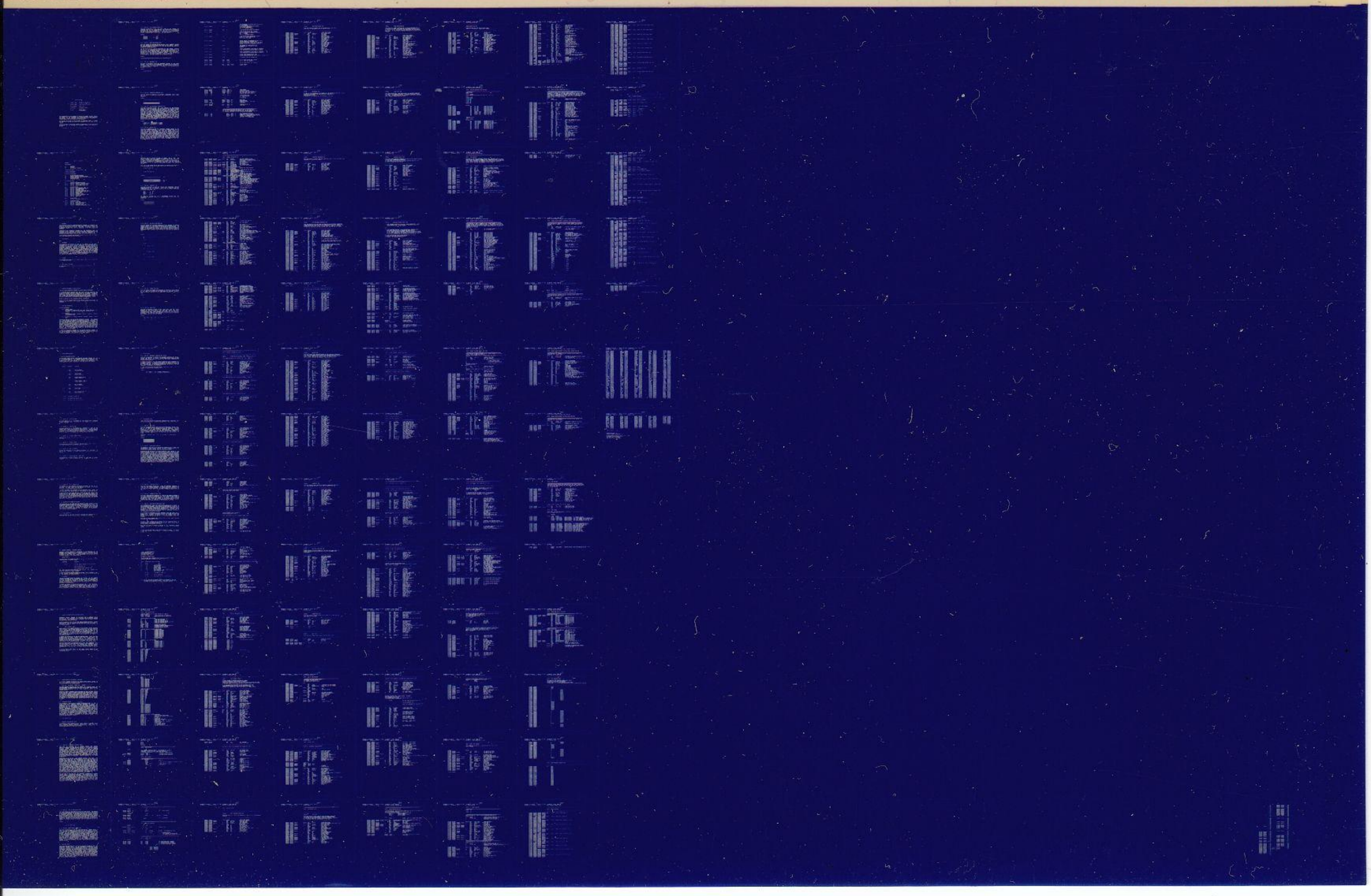
EP-DZLAE-B-DL-B

FICHE 1 OF 1

JUN 1977

digital

MADE IN USA



B01

.REM +

IDENTIFICATION

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

PRODUCT NAME: LA180 PRINTER DIAGNOSTIC

DATE CREATED: MARCH 1977

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: ROBERT BAKER
R. QUENNEVILLE

TABLE OF CONTENTS

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 TO 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 EITHER A STANDARD PARALLEL INTERFACE OR A KL11/DL11 SERIAL INTERFACE. MULTIPLE LA180'S, ON INDIVIDUAL INTERFACES, CAN BE RUN TOGETHER PROVIDING THEY ARE ON EITHER ALL SERIAL OR ALL PARALLEL INTERFACES, AND PROVIDING THEIR ADDRESS ASSIGNMENTS ARE CONTIGUOUS. 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 SECTION 5.5 FOR INTERFACE CONTROL INFORMATION.

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. FOR USE ON SINGLE-PARALLEL INTERFACE ONLY.

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
11		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 REENTER 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 ON PARALLEL INTERFACE

IF A CONSOLE TERMINAL EXISTS AND THE INHIBIT ERROR TYPEOUT SWITCH IS DOWN (0), WHENEVER AN 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.

5.5 INTERFACE CONTROL

THIS DIAGNOSTIC, WHEN INITIALLY LOADED, IS PRECONDITIONED TO TEST A SINGLE LA180 ON A PARALLEL INTERFACE ADDRESSED AS 177510. THE DIAGNOSTIC CAN BE CONDITIONED TO RUN ON SERIAL INTERFACES BY DEPOSITING ANY NON-ZERO QUANTITY INTO LOCATION SERSW OR BY SELECTING AND RUNNING TEST 77. THE DIAGNOSTIC CAN BE RECONDITIONED TO RUN ON PARALLEL INTERFACES BY DEPOSITING ALL ZEROS INTO LOCATION SERSW OR BY SELECTING AND RUNNING TEST 76. THE NUMBER OF INTERFACES TO BE DRIVEN IS CONTROLLED BY THE QUANTITY IN LOCATION NUMLP. THE VALUE IN THIS LOCATION IS PRESET TO 1 AND SHOULD BE CHANGED IF MORE THAN 1 UNIT IS TO BE TESTED. THE ADDRESS OF THE LOWEST ADDRESSED PARALLEL INTERFACE (IF ANY) SHOULD BE DEPOSITED INTO LOCATION FSTPAD. THIS VALUE HAS BEEN PRESET TO 177510. THE ADDRESS OF THE LOWEST ADDRESSED SERIAL INTERFACE (IF ANY) SHOULD BE DEPOSITED INTO LOCATION FSTSAD. THIS VALUE HAS BEEN PRESET TO 175610.

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 PARALLEL-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 DEVICE 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 EXAMINE 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:

```
0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X0X
```

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:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
   .

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
```

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:

```
!"#$%&'()*+,-./0123456789:;<=>?@ABC....
"#$%&'()*+,-./01234567 9:;<=>?@ABCD....
#$%&'()*+,-./012456789:;<=>?@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:

```
11111111111111111111111111111111.....322
00000000000000000000000000000000.....111
00000000011111111112222222222333.....333
12345678901234567890123456789901.....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 - @
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
 @ @ @ @ @ @ @ @ @ @
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:

```
01----------------------------------------
02--------------------

04--------------------


08--------------------
  \
   > 7 BLANK LINES
  /
16--------------------
  \
   \
    > 15 BLANK LINES
   /
32--------------------
  \
   \
    > 31 BLANK LINES
   /
00----------------------------------------
```

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 OPTION TESTS

6.3.1 TEST 50 - SECONDARY CHARACTER SET OPTION

THIS TEST IS DESIGNED TO EXERCISE THE SECONDARY CHARACTER SET OPTION, TESTING THE ABILITY TO SELECT EITHER CHARACTER SET UNDER SOFTWARE CONTROL FROM THE CPU, AND ALSO TESTING THAT THE CORRECT CHARACTERS ARE PRINTED WITHIN EACH SET.

A NUMBER IS PRINTED AT THE LEFT MARGIN INDICATING WHICH CHARACTER SET IS BEING PRINTED. #1 INDICATES THE PRIMARY SET AND #2 INDICATES THE SECONDARY (APL) SET. FOLLOWING THE NUMBER THE APPROPRIATE CHARACTER SET, DISTINGUISHED BY THE ABSENCE OR PRESENCE OF THE 8TH BIT IN EACH CHARACTER , IS PRINTED IN ITS ENTIRETY.

IF LESS THAN 98 COLUMNS ARE BEING TESTED, ADDITIONAL LINE(S) WILL BE PRINTED TO COMPLETE EACH CHARACTER SET.

EXAMPLE:

```
#1= !"#$%&'()...ETC..PRIMARY CHARACTER SET...
#2= !"#$%&'()...ETC..SECONDARY CHARACTER SET...
```

6.4 MAINTENANCE AIDS

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

6.4.1 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
BBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBB
```

6.4.2 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.

6.4.3 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.

6.4.4 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.

6.4.5 TEST 64 - SELECTED PATTERN PRINT TEST

THIS TEST REPEATEDLY LOADS THE LA180 PRINTER BUFFER WITH A STRING OF DATA RESIDING IN STORAGE. THIS STRING IS ESTABLISHED BY TYPING ON THE CONSOLE KEYBOARD AND TERMINATING THE STRING BY DEPRESSING EITHER CTL-A OR CTL-B. PRINTING THEN BEGINS AUTOMATICALLY. PRINTING CAN BE INTERRUPTED BY DEPRESSING CTL-SPACE WHEREUPON A NEW PATTERN CAN BE ENTERED. IF CTL-A OR CTL-B IS THE FIRST KEY DEPRESSED, PRINTING OF THE INTERRUPTED PATTERN WILL RESUME.

IF A PATTERN OF CHARACTERS IS TO BE REPEATED OVER AND OVER (ON A SINGLE LINE) IT IS ONLY NECESSARY TO ENTER THE PATTERN ONCE AND THEN DEPRESS CTL-B. EXAMPLE: IF ABC(SP)(CTL-B) IS ENTERED, EACH LINE PRINTED WILL BE ABC ABC ABC ABC ETC., (TO MAXIMUM NUMBER OF SPECIFIED COLUMNS).

NOTE THAT NO PRINT COMMAND CODE (LF,CR, OR FF) IS USED WHEN ENTERING PATTERNS TO BE REPEATED BY CTL-B.

OPTIONALLY PRINT COMMANDS CAN BE ENTERED INTO THE CHARACTER STRING TO OBTAIN ALMOST ANY DESIRED SEQUENCE OF ACTIONS. SUCH A STRING SHOULD BE TERMINATED BY DEPRESSING CTL-A ONLY.

NOTE THAT THE STRING IS STORED AT THE END OF ALL DIAGNOSTIC CODING THEREBY ALLOWING IT TO BE OPEN-ENDED IN LENGTH (UP TO THE MEMORY LIMIT).

TO STOP PRINTING AND ENTER A NEW STRING DEPRESS CTL-SPACE. TO SELECT A NEW TEST DEPRESS DELETE/RUBOUT. TO CHANGE THE NUMBER OF COLUMNS DEPRESS CTL-C.

```
        .ENABLE ABS,AMA


.TITLE  MAINDEC-11-DZLAE-B
;*COPYRIGHT (C) 1977
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY ROBERT BAKER
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*

.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

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

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

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

```
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
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

                000000                          .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                          .=174
000174  000000                 DISPREG: .WORD  0              ;;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:   .WORD  0              ;;SOFTWARE SWITCH REGISTER

                                        .SBTTL  ACT11 HOOKS

                                        ;;*****************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                000200                          $SVPC=.                 ;SAVE PC
                000046                          .=46
000046  005322                          $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  007744                          $ERROR                  ;EMT - ERROR HANDLER
000032  000000                          0

                000100                  .=100

000100  006566                          LKSRV                   ;KW11-L  INTERRUPT
000102  000000                          0

000104  006556                          DCI                     ;KW11-P  INTERRUPT
000106  000000                          0


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

                                .SBTTL  STARTING ADDRESSES

                000200                  .=200

000200  000137  001174                  JMP     START           ;GENERAL START

                000600                  .=600

000600  000137  001210                  JMP     RESTRT          ;RESTART - NO INTERACTIVE TESTS

000604  000137  001220                  JMP     CNTRL           ;GO TO KEYBOARD CONTROL
                                                                ;WILL DEFAULT TO SWITCH REG
                                                                ;IF CONSOLE DEVICE DOES NOT EXIST

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

                                .SBTTL  COMMON TAGS

                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  000000                  LPKS:   0               ;THE FOLLOWING 4 LOCATIONS ARE INITIALIZED
                                                        ;BY THE PROGRAM
                                                        ;LINE PRINTER KBD STATUS REG ADDRESS
                                                        ;(IF SERIAL INTERFACE)
                                                        ;BIT 6 = INTERRUPT ENABLE

001112  000000                  LPKB:   0               ;LINE PRINTER KBD BUFFER REG ADDRESS

001114  000000                  LPS:    0               ;LINE PRINTER STATUS REG. ADDRESS
                                                        ;BIT 15 = ERROR (IF PARALLEL INTERFACE)
                                                        ;BIT  7 = READY
                                                        ;BIT  6 = INTERRUPT ENABLE

001116  000000                  LPB:    0               ;LINE PRINTER DATA BUFFER REG. ADDRESS
                                                        ;BITS 0-6 = ASCII CHAR BUFFER
                                                        ;BITS 7-15 = NOT USED

001120  000000                  SERSW:  0               ;SERIAL SWITCH. THE PRESENCE OF ANY 1-BITS
                                                        ;IN THIS WORD WILL INDICATE THAT
                                                        ;SERIAL-INTERFACED LA180'S ARE TO BE TESTED.
                                                        ;DEFAULT IS ALL 0'S = PARALLEL INTERFACE.

001122  000001                  NUMLP:  1               ;THE NUMBER OF LA180'S WHICH THIS
                                                        ;DIAGNOSTIC WILL ATTEMPT TO TEST.
                                                        ;DEFAULT = 1

001124  175610                  FSTSAD: 175610          ;FIRST LA180 ADDRESS TO BE USED IF INTERFACE-
                                                        ;TYPE INDICATED IN LOCATION SERSW IS SERIAL.

001126  177510                  FSTPAD: 177510          ;FIRST LA180 ADDRESS TO BE USED IF INTERFACE-
                                                        ;TYPE INDICATED IN LOCATION SERSW IS PARALLEL.

001130  000000                  ACTFST: 0               ;ACTUAL FIRST ADDRESS BEING USED
                                                        ;DURING PROGRAM EXECUTION. INITILIZED BY PROGRAM ITSELF

001132  000000                  LPCTR:  0               ;LP COUNTER (USED BY PROGRAM ONLY)

001134  172540                  PLKS:   172540          ;KL11-P CLOCK STATUS REG. ADDRESS
001136  172542                  CSBR:   172542          ;KL11-P COUNT SET ADDRESS

001140  177546                  LKS:    177546          ;KW11-L CLOCK STATUS REG. ADDRESS

001142  177570                  SWR:    .WORD   177570  ;SW REG ADDRESS
001144  177570                  DISPLAY: .WORD  177570  ;DISPLAY ADR
```

```
001146  000000                  $TSTNM: .WORD   0               ;TEST NUMBER
001150  000204                  WIDTH:  .WORD   132.            ;NUMBER OF COLUMNS
001152     000                  STRONE: .BYTE   0               ;RUN TEST ONCE FLAG (SW REG CTL)
001153     000                  TRONE:  .BYTE   0               ;RUN TEST ONCE FLAG (KYBD CTL)
001154     000                  TLOOP:  .BYTE   0               ;LOOP ON TEST FLAG (KYBD CTL)

001155     000                  CKFLAG: .BYTE   0               ;CLOCK OPTION FLAG
                                                                ;0 = NONE AVAILABLE
                                                                ;+1 = KL11-L
                                                                ;-1 = KL11-P

001156  000207                  $BELL:  .ASCIZ  <207>           ;BELL CODE
001160     077                  $QUES:  .ASCII  /?/             ;QUESTION MARK
001161     015                  $CRLF:  .ASCII  <15>            ;CARRIAGE RETURN - LINE FEED
001162  000012                  $LF:    .ASCIZ  <12>            ;LINE FEED

001164  000015                  $CR:    .ASCIZ  <15>            ;CARRIAGE RETURN ONLY
001166  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.
001170     000          $NULL:  .BYTE   0               ;NULL CHARACTER FOR FILLS
001171     002          $FILLS: .BYTE   2               ;NUMBER OF FILLER CHARACTERS REQUIRED
001172     012          $FILLC: .BYTE   12              ;INSERT FILL CHARACTERS AFTER LINE FEED
001173     000          $TPFLG: .BYTE   0               ;TERMINAL UNAVAILABLE FLAG
                                                        ;BIT <07> = 1 = UNAVAILABLE
                        .EVEN
```

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

                                .SBTTL  PROGRAM INITIALIZATION

001174  005037  001146          START:  CLR     $TSTNM          ;SET TEST NUMBER TO ZERO
001200  023737  000042  000046          CMP     @#42,@#46       ;CHECK IF IN ACT QUICK VERIFY
                                                                ;SKIP MANUAL INTERVENTION TESTS IF YES
001206  001007                         BNE     STARTX          ;INITIALIZE
001210  012737  000020  001146  RESTRT: MOV     #20,$TSTNM      ;SET TEST NUMBER TO 20
001216  000403                         BR      STARTX          ;INITIALIZE
001220  012737  177777  001146  CONTRL: MOV     #-1,$TSTNM      ;SET CONTROL FLAG
001226  012737  177570  001142  STARTX: MOV     #177570,SWR     ;INITIALIZE SWR CONTROL
001234  013737  001142  001144          MOV     SWR,DISPLAY
                                .SBTTL  INITIALIZE THE COMMON TAGS
001242  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001246  012737  013060  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001254  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001262  012737  013210  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001270  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001276  005037  005336                  CLR     $PASS           ;;CLEAR THE PASS COUNT
001302  013737  005306  005300          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1"  SETUP FOR A SOFTWARE SWITCH REGISTER.
001310  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001314  012737  001350  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001322  012737  177570  001142          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001330  012737  177570  001144          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001336  022777  177777  177576          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001344  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
001346  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
001350  012716  001356          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001354  000002                          RTI
001356  012737  000176  001142  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001364  012737  000174  001144          MOV     #DISPREG,DISPLAY
001372  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001376  017700  177540                  MOV     @SWR,R0         ;GET SWITCHES
001402  005200                          INC     R0              ;CHECK IF ALL SWITCHES ARE UP
001404  001006                          BNE     6$              ;CONTINUE IF NOT
001406  012737  000176  001142          MOV     #SWREG,SWR      ;IF UP, SET SOFTWARE SWITCH CONTROL
001414  012737  000174  001144          MOV     #DISPREG,DISPLAY
001422  017700  177514          6$:     MOV     @SWR,R0         ;GET SW REG
001426  042700  177400                  BIC     #177400,R0      ;SAVE BITS 0-7
001432  020027  000204                  CMP     R0,#132.        ;TEST # COLUMNS
001436  003003                          BGT     2$              ;TOO BIG, DEFAULT TO 132(10)
001440  020027  000002          1$:     CMP     R0,#2.          ;TEST # COLUMNS
001444  103002                          BHIS    3$              ;BRANCH IF OK
001446  012700  000204          2$:     MOV     #132.,R0        ;DEFAULT TO 132(10)
001452  010037  001150          3$:     MOV     R0,WIDTH        ;SAVE # COLUMNS
001456  105037  001154                  CLRB    TLOOP           ;RESET FLAGS
001462  105037  001152                  CLRB    STRONE
001466  105037  001153                  CLRB    TRONE
001472  005037  011660                  CLR     SVTST
001476  000401                          BR      10$             ;REPLACE THIS INSTRUCTION WITH NOP (204
```

```
                                                                        ;TO SKIP CLOCK TIMINGS
001500  000424                                 BR      20$
001502  012737  000002  000006  10$:    MOV     #RTI,@#6                ;SET TRAP RETURN
001510  012737  000006  000004          MOV     #6,@#4
001516  112737  177777  001155          MOVB    #-1,CKFLAG              ;SET CLOCK FLAG FOR KW11-P
001524  000261                          SEC                             ;SET C-BIT
001526  105777  177402                  TSTB    @PLKS                   ;KW11-P AVAILABLE?
001532  103011                          BCC     30$                     ;YES, CHECK FOR CONSOLE TERMINAL
001534  112737  000001  001155          MOVB    #1,CKFLAG               ;SET CLOCK FLAG FOR KW11-L
001542  000261                          SEC                             ;SET C-BIT
001544  105777  177370                  TSTB    @LKS                    ;KW11-L AVAILABLE?
001550  103002                          BCC     30$                     ;YES, CHECK FOR CONSOLE TERMINAL
001552  105037  001155          20$:    CLRB    CKFLAG                  ;CLEAR CLOCK AVAILABLE FLAG - NONE THERE
001556  004737  001764          30$:    JSR     PC,QUO1                 ;INITILIZE ACTFST AND SELECT DRIVER
001562  112737  177777  001173          MOVB    #-1,$TPFLG              ;SET CONSOLE UNAVAILABLE FLAG
001570  000261                          SEC                             ;SET C-BIT
001572  105777  177302                  TSTB    @TKS                    ;CONSOLE TERMINAL THERE?
001576  103402                          BCS     4$                      ;NO, BRANCH
001600  105037  001173                  CLRB    $TPFLG                  ;YES, CLEAR CONSOLE UNAVAILABLE FLAG
001604  103002                  4$:     BCC     5$                      ;CONTINUE IF CONSOLE
001606  104414  015225                  PRINT   ,NCMSG                  ;PRINT NO CONSOLE MESSAGE
001612  005037  000006          5$:     CLR     @#6                     ;RESET TRAP VECTOR HALTS
001616  005037  000004                  CLR     @#4
001622  004737  002032                  JSR     PC,QUO2                 ;TYPE INTF. MISC. IF CONSOLE
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
001626  005227  177777                  INC     #-1                     ;;FIRST TIME?
001632  001022                          BNE     67$                     ;;BRANCH IF NO
001634  022737  005322  000042          CMP     #$ENDAD,@#42            ;;ACT-11?
001642  001416                          BEQ     67$                     ;;BRANCH IF YES
001644  104401  001652                  TYPE    ,68$                    ;;TYPE ASCIZ STRING
001650  000413                          BR      67$                     ;;GET OVER THE ASCIZ
                                ;;68$:  .ASCIZ  <CRLF>#MAINDEC-11-DZLAE-B#<CRLF>
001700                          67$:
001700  005737  001146                  TST     $TSTNM                  ;WANT CONTROL NOW?
001704  002005                          BGE     11$                     ;NO, CONTINUE
001706  105737  001173          15$:    TSTB    $TPFLG                  ;TERMINAL THERE?
001712  100406                          BMI     13$                     ;NO, DEFAULT TO SW REG CONTROL
001714  000137  007442                  JMP     KYBDST                  ;YES, GO TO KYBD CONTROL
001720  032777  000400  177214  11$:    BIT     #SW8,@SWR               ;WANT TEST SELECTION?
001726  001402                          BEQ     12$                     ;NO, CHECK TEST #
001730  000137  006752          13$:    JMP     SELECT                  ;YES, GO TO TEST SELECTION HALT
001734  013700  001146          12$:    MOV     $TSTNM,R0               ;GET TEST NUMBER
001740  006300                          ASL     R0                      ;SET POINTER
001742  005760  013356                  TST     TAT(R0)                 ;CHECK IF TEST IN TABLE
001746  003004                          BGT     14$                     ;BRANCH IF IN TABLE
001750  002756                          BLT     15$                     ;END OF SEQUENCE, SELECT TEST
001752  005237  001146                  INC     $TSTNM                  ;INCREMENT TEST NUMBER
001756  000760                          BR      11$                     ;CHECK NEXT TEST NUMBER IN TABLE
001760  000170  013356          14$:    JMP     @TAT(R0)                ;GO TO TEST
```

```
001764  005737  001120          QU01:   TST     SERSW           ;IS SERIAL SWITCH SET?
001770  001407                          BEQ     80$             ;ZEROS=NO=PARALLEL=BRANCH
001772  013737  001124  001130        MOV     FSTSAD,ACTFST   ;SET ADDRESS OF 1ST SERIAL INTF.
002000  012737  011206  011042        MOV     #SLOAD,LDPTR+2  ;SWITCH TO SERIAL LOADER
002006  000406                          BR      90$             ;GET AROUND NEXT 2 INSTRS
002010  013737  001126  001130  80$:   MOV     FSTPAD,ACTFST   ;SET ADDRESS OF 1ST PARALLEL INTF.
002016  012737  011044  011042        MOV     #PLOAD,LDPTR+2  ;SWITCH TO PARALLEL LOADER
002024  004737  011316          90$:   JSR     PC,RSTADR       ;ADDRESS 1ST LINE PRINTER
002030  000207                          RTS     PC              ;RETURN


002032  105737  001173          QU02:   TSTB    $TPFLG          ;IS THERE A CONSOLE TERMINAL?
002036  100430                          BMI     78$             ;BRANCH IF NONE
002040  104401  002176                  TYPE    ,74$            ;CRLF CONSOLE
002044  005737  001120                  TST     $ERSW           ;IS SERIAL SWITCH SET?
002050  001476                          BEQ     77$             ;ZEROS=NO=PARALLEL=BRANCH
002052  104401  002122                  TYPE    ,71$            ;SERIAL
002056  104401  002132          70$:    TYPE    ,72$            ;INTERFACED LA180 TEST, CRLF
002062  104401  002161                  TYPE    ,73$            ;# LA180'S =
002066  013746  001122                  MOV     NUMLP,-(SP)     ;PUT # OF LP'S ONTO STACK
002072  104405                          TYPDS                   ;TYPE # OF LP'S BEING TESTED
002074  104401  002176                  TYPE    ,74$            ;CRLF
002100  104401  002200                  TYPE    ,75$            ;FIRST PRINTER'S ADDRESS IS
002104  013746  001130                  MOV     ACTFST,-(SP)    ;PUT VALUE ONTO STACK
002110  104403                          TYPOS                   ;TYPE OCTAL ADDRESS
002112     006                          .BYTE   6
002113     001                          .BYTE   1
002114  104401  002176                  TYPE    ,74$            ;CRLF
002120  000207                  78$:    RTS     PC              ;EXIT
002122  042523  044522  046101  71$:    .ASCIZ  /SERIAL /
002130  000040
002132  047111  042524  043122  72$:    .ASCIZ  /INTERFACED LA180 TEST/<CRLF>
002140  041501  042105  046040
002146  030501  030070  052040
002154  051505  100124     000
002161     043  046040  030501  73$:    .ASCIZ  /# LA180'S = /
002166  030070  051447  036440
002174  000040
002176  000200                  74$:    .ASCIZ  <CRLF>
002200  044506  051522  020124  75$:    .ASCIZ  /FIRST PRINTER'S ADDRESS IS /
002206  051120  047111  042524
002214  023522  020123  042101
002222  051104  051505  020123
002230  051511  000040
002234  040520  040522  046114  76$:    .ASCIZ  /PARALLEL /
002242  046105  000040
                                        .EVEN

002246  104401  002234          77$:    TYPE    ,76$            ;PARALLEL
002252  000701                          BR      70$             ;CONTINUE
```

```
                        ;;*****************************************************************
                        .SBTTL  OPERATOR INTERVENTION TESTS
                        ;/////////////////////////////////////////////////////////////////
                        ;TEST0  -       INTERFACE AND CONTROL TESTS (PARALLEL INTF ONLY)
                        ;
                        ;TEST ERROR AND READY BITS, PRINTER OFF LINE - POWER OFF
                        ;/////////////////////////////////////////////////////////////////

002254  005737  001120      TEST0:  TST     SERSW           ;IS PARALLEL INTF INDICATED?
002260  001402                      BEQ     29$             ;BRANCH IF YES
002262  000137  006644              JMP     EXIT            ;OTHERWISE EXIT TEST
002266  104401  015567      29$:    TYPE,   TOMSG0          ;TYPE INSTRUCTIONS
002272  104421                      HOLD                    ;WAIT FOR OPERATOR
002274  104420              28$:    CHECK                   ;CHECK FOR CONTROL
002276  005777  176612              TST     @LPS            ;CHECK FOR ERROR CONDITION
002302  100402                      BMI     1$              ;OK, ERROR SET
002304  104001                      ERROR   1               ;ERROR CLEAR, POWER OFF
002306  000772                      BR      28$             ;RETEST
002310  105777  176600      1$:     TSTB    @LPS            ;CHECK READY
002314  100002                      BPL     2$              ;OK, READY NOT SET
002316  104002                      ERROR   2               ;READY SET, POWER OFF
002320  000765                      BR      28$             ;RETEST
                        ;/////////////////////////////////////////////////////////////////
                        ;TEST ERROR AND READY BITS, PRINTER OFF LINE - POWER ON
                        ;/////////////////////////////////////////////////////////////////

002322  104401  015626      2$:     TYPE,   TOMSG1          ;TYPE INSTRUCTION - TURN POWER ON
002326  104421                      HOLD                    ;WAIT FOR OPERATOR
002330  104420              3$:     CHECK                   ;CHECK FOR CONTROL
002332  005777  176556              TST     @LPS            ;CHECK ERROR
002336  100402                      BMI     4$              ;OK, ERROR SET
002340  104003                      ERROR   3               ;ERROR CLEAR, PRINTER OFF LINE
002342  000772                      BR      3$              ;RETEST
002344  105777  176544      4$:     TSTB    @LPS            ;CHECK READY
002350  100002                      BPL     5$              ;OK, READY NOT SET
002352  104004                      ERROR   4               ;READY SET, PRINTER OFF LINE
002354  000765                      BR      3$              ;RETEST
                        ;/////////////////////////////////////////////////////////////////
                        ;TEST ERROR AND READY BITS, PRINTER ON LINE
                        ;/////////////////////////////////////////////////////////////////

002356  104401  015651      5$:     TYPE    ,TOMSG2         ;TYPE INSTRUCTION, TURN ON LINE
002362  104421                      HOLD                    ;WAIT FOR OPERATOR
002364  104420              6$:     CHECK                   ;CHECK FOR CONTROL
002366  005777  176522              TST     @LPS            ;CHECK ERROR
002372  100002                      BPL     7$              ;OK, ERROR CLEAR
```

```
002374  104005                          ERROR   5               ;ERROR SET, PRINTER ON LINE
002376  000772                          BR      6$              ;RETEST
002400  105777  176510        7$:       TSTB    @LPS            ;CHECK READY
002404  100402                          BMI     8$              ;OK, READY SET
002406  104006                          ERROR   6               ;READY CLEAR, PRINTER ON LINE
002410  000765                          BR      6$              ;RETEST

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

                                ;TEST PAPER OUT SWITCH

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

002412  104401  015705          8$:       TYPE    ,TOMSG3         ;TYPE INSTRUCTION, PAPER OUT
002416  104421                          HOLD                    ;WAIT FOR OPERATOR
002420  104420                  9$:       CHECK                   ;CHECK CONTROL
002422  012777  000012  176466          MOV     #12,@LPB        ;SEND LF
002430  005777  176460                  TST     @LPS            ;CHECK FOR ERROR CONDITION
002434  100402                          BMI     10$             ;OK, ERROR SET
002436  104007                          ERROR   7               ;ERROR CLEAR, PAPER OUT ERROR
002440  000767                          BR      9$              ;RETEST
002442  105777  176446          10$:      TSTB    @LPS            ;CHECK READY
002446  100002                          BPL     11$             ;OK, READY CLEAR
002450  104010                          ERROR   10              ;READY SET, PAPER OUT, ON LINE
002452  000762                          BR      9$              ;RETEST

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

                                ;TEST ABILITY TO CLEAR ERROR CONDITION

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

002454  104401  015737          11$:      TYPE    .TOMSG4         ;TYPE INSTRUCTION, RESET 3 ON LINE
002460  104421                          HOLD                    ;WAIT FOR OPERATOR
002462  104420                  12$:      CHECK                   ;CHECK FOR CONTROL
002464  005777  176424                  TST     @LPS            ;CHECK ERROR
002470  100002                          BPL     13$             ;OK, ERROR CLEAR
002472  104011                          ERROR   11              ;ERROR DID NOT CLEAR
002474  000772                          BR      12$             ;RETEST
002476  105777  176412          13$:      TSTB    @LPS            ;CHECK READY
002502  100402                          BMI     14$             ;OK, READY SET
002504  104012                          ERROR   12              ;READY NOT SET
002506  000773                          BR      13$             ;RETEST

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

                                ;CHECK ERROR & READY BITS AFTER RESET INSTRUCTION

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

002510  104420                  14$:      CHECK                   ;CHECK CONTROL
002512  000005                          RESET                   ;CLEAR WORLD
002514  005777  176374                  TST     @LPS            ;CHECK ERROR
002520  100002                          BPL     15$             ;OK, ERROR CLEAR
002522  104013                          ERROR   13              ;ERROR BIT SET AFTER RESET INSTR.
002524  000771                          BR      14$             ;RETEST
```

```
002526  104420                  15$:    CHECK                   ;CHECK CONTROL
002530  000005                          RESET                   ;CLEAR WORLD
002532  105777  176356                  TSTB    @LPS            ;CHECK READY
002536  100402                          BMI     16$             ;OK, READY SET
002540  104014                          ERROR   14              ;READY BIT CLEAR AFTER RESET INSTR.
002542  000771                          BR      15$             ;RETEST

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

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

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

002544  104420                  16$:    CHECK                   ;CHECK CONTROL
002546  005005                          CLR     R5              ;CLEAR TIME OUT COUNTER
002550  012777  000015  176340          MOV     #15,@LPB        ;LOAD CARRIAGE RETURN INTO BUFFER
002556  105777  176332                  TSTB    @LPS            ;CHECK READY
002562  100002                          BPL     17$             ;OK, READY CLEAR
002564  104015                          ERROR   15              ;READY BIT NOT CLEAR WHEN CHAR LOADED
002566  000766                          BR      16$             ;RETEST
002570  005777  176320          17$:    TST     @LPS            ;CHECK ERROR
002574  100002                          BPL     18$             ;OK, ERROR CLEAR
002576  104016                          ERROR   16              ;ERROR BIT SET AFTER CHAR LOAD
002600  000761                          BR      16$             ;RETEST
002602  105777  176306          19$:    TSTB    @LPS            ;PRINTER STILL BUSY
002606  100404                          BMI     19$             ;NO, NEXT TEST
002610  005205                          INC     R5              ;YES, INC TIMER
002612  001366                          BNE     17$             ;WAIT FOR FLAG
002614  104017                          ERROR   17              ;NO, TOO LONG
002616  000752                          BR      16$             ;RETEST

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

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

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

002620  012705  000340          19$:    MOV     #PR7,R5         ;SET FIRST LEVEL
002624  104420                  20$:    CHECK                   ;CHECK CONTROL
002626  012737  002730  000200          MOV     #23$,200        ;SET INTERRUPT RETURN
002634  012737  000340  000202          MOV     #PR7,202        ;SET PRIORITY 7 ON INTERRUPT
002642  005777  176246                  TST     @LPS            ;CHECK FOR ERROR
002646  100002                          BPL     21$             ;OK, ERROR CLEAR
002650  104020                          ERROR   20              ;ERROR BIT SET
002652  000764                          BR      20$             ;RETEST
002654  105777  176234          21$:    TSTB    @LPS            ;CHECK READY
002660  100402                          BMI     22$             ;OK, READY SET
002662  104021                          ERROR   21              ;READY BIT NOT SET
002664  000757                          BR      20$             ;RETEST
002666                          22$:
002666  010546                          MOV     R5,-(SP)        ;;PUT NEW PS ON STACK
002670  012746  002676                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
```

```
002674  000002                          RTI                     ;;POP NEW PC AND PS
002676                          64$:
002676  052777  000100  176210          BIS     #BIT6,@LPS      ;SET PRINTER INT. ENABLE
002704  000240                          NOP                     ;DELAY
002706  042777  000100  176200          BIC     #BIT6,@LPS      ;CLEAR PRINTER INT. ENABLE
002714  162705  000040                  SUB     #40,R5          ;SET NEXT LEVEL
002720  020527  000140                  CMP     R5,#PR3         ;LEVEL 3?
002724  001404                          BEQ     24$             ;YES, CONTINUE NEXT TEST
002726  000736                          BR      20$             ;NO, TEST THIS LEVEL

002730  022626                  23$:    CMP     (SP)+,(SP)+     ;RESTORE STACK
002732  104022                          ERROR   22              ;INTERRUPT ABOVE LEVEL 3
002734  000733                          BR      20$             ;RETEST

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

002736  104420                  24$:    CHECK                   ;CHECK FOR CONTROL
002740  012737  003024  000200          MOV     #27$,200        ;SET INTERRUPT RETURN
002746  005777  176142                  TST     @LPS            ;CHECK FOR ERROR
002752  100002                          BPL     25$             ;OK, ERROR CLEAR
002754  104020                          ERROR   20              ;ERROR BIT SET
002756  000767                          BR      24$             ;RETEST
002760  105777  176130          25$:    TSTB    @LPS            ;CHECK READY
002764  100402                          BMI     26$             ;OK, READY SET
002766  104021                          ERROR   21              ;READY CLEAR
002770  000762                          BR      24$             ;RETEST
002772                          26$:
002772  010546                          MOV     R5,-(SP)        ;;PUT NEW PS ON STACK
002774  012746  003002                  MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
003000  000002                          RTI                     ;;POP NEW PC AND PS
003002                          65$:
003002  052777  000100  176104          BIS     #BIT6,@LPS      ;SET PRINTER INTR. ENABLE
003010  000240                          NOP                     ;DELAY
003012  042777  000100  176074          BIC     #BIT6,@LPS      ;CLEAR PRINTER INTR. ENABLE
003020  104023                          ERROR   23              ;NO INTERRUPT BELOW LEVEL 4
003022  000745                          BR      24$             ;RETEST

003024  042777  000100  176062  27$:    BIC     #BIT6,@LPS      ;CLEAR PRINTER INTR. ENABLE
003032  022626                          CMP     (SP)+,(SP)+     ;RESET STACK
003034  162705  000040                  SUB     #40,R5          ;SET NEXT LEVEL
003040  002336                          BGE     24$             ;RETEST IF NOT DONE ALL LEVELS
003042  012737  000137  000200          MOV     #137,200        ;RESET INSTRUCTIONS AT 200-202
003050  012737  001174  000202          MOV     #START,202
003056  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
003060  012746  003066                  MOV     #66$,-(SP)      ;;PUT NEW PC ON  STACK
003064  000002                          RTI                     ;;POP NEW PC AND PS
003066                          66$:
003066  000137  006644                  JMP     EXIT            ;EXIT TEST
```

```
                        ;////////////////////////////////////////////////////////////////////////
                        ;TEST1  -       TOP OF FORM SWITCH TEST
                        ;////////////////////////////////////////////////////////////////////////

003072  104413          TEST1:  PRTHDR                  ;PRINT TEST HEADER
003074  012705  003204          MOV     #4$,R5          ;SET TABLE POINTER
003100  012701  000036          MOV     #30.,R1         ;SET CHAR COUNT
003104  012700  000055          MOV     #55,R0          ;SET DASH CHAR
003110  104416                  MLOAD                   ;LOAD DASHED LINE
003112  104414  001164          PRINT   ,$CR            ;PRINT LINE
003116  104401  016035  3$:     TYPE    ,T1MSG3         ;TYPE INSTRUCTIONS
003122  010537  003130          MOV     R5,1$           ;SET SWITCH SETTING FOR MESG
003126  104401                  TYPE
003130  000000          1$:     .WORD   0
003132  104401  016067          TYPE    ,T1MSG4         ;FINISH INSTRUCTIONS
003136  104421                  HOLD                    ;WAIT FOR OPERATOR
003140  104420                  CHECK                   ;CHECK FOR CONTROL
003142  104414  001166          PRINT   ,$FF            ;ISSUE FORM FEED
003146  104414  015777          PRINT   ,T1MSG1         ;PRINT REFERENCE LINE
003152  010537  003160          MOV     R5,2$           ;SET FORM FEED LENGTH FOR MESG
003156  104414                  PRINT
003160  000000          2$:     .WORD   0
003162  104414  016006          PRINT   ,T1MSG2         ;FINISH MESG
003166  022525                  CMP     (R5)+,(R5)+     ;INC TABLE POINTER
003170  005715                  TST     (R5)            ;CHECK TABLE TO SEE IF DONE TEST
003172  001351                  BNE     3$              ;FINISH TEST
003174  104414  001162          PRINT   ,$LF            ;ADVANCE PAPER WHEN DONE
003200  000137  006644          JMP     EXIT            ;EXIT TEST

003204  031440  000040  4$:     .ASCIZ  / 3 /           ;FORM FEED SWITCH SETTINGS
003210  027063  000065          .ASCIZ  /3.5/
003214  032040  000040          .ASCIZ  / 4 /
003220  027065  000065          .ASCIZ  /5.5/
003224  033040  000040          .ASCIZ  / 6 /
003230  033440  000040          .ASCIZ  / 7 /
003234  034040  000040          .ASCIZ  / 8 /
003240  027070  000065          .ASCIZ  /8.5/
003244  030461  000040          .ASCIZ  /11 /
003250  031061  000040          .ASCIZ  /12 /
003254  032061  000040          .ASCIZ  /14 /
003260  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.
                        ;//////////////////////////////////////////////////////////////////////////////

003262  104413                TEST2:  PRTHDR                  ;PRINT TEST HEADER
003264  012737  003500  006610        MOV     #SPD,TORTN      ;SET TIME OUT RETURN
003272  005004                      CLR     R4              ;CLEAR LINE COUNT
003274  105737  001155              TSTB    CKFLAG          ;TEST CLOCK FLAG
003300  001417                      BEQ     4$              ;NONE THERE, MANUAL TIMING
003302  002407                      BLT     1$              ;KW11-P
003304  013737  006612  006606      MOV     MINCNT,CNTR     ;SET KW11-L CLOCK COUNT
003312  012777  000100  175620      MOV     #BIT6,@LKS      ;ENABLE CLOCK INTERRUPT
003320  000427                      BR      T2SP            ;START PRINTING
003322  013777  006612  175606  1$: MOV     MINCNT,@CSBR    ;SET CLOCK COUNT
003330  012777  000105  175576      MOV     #105,@PLKS      ;START CLOCK
003336  000420                      BR      T2SP            ;START PRINTING
003340  023727  001142  177570  4$: CMP     SWR,#177570     ;CLECK SW REG ADR
003346  001406                      BEQ     2$
003350  104414  016134              PRINT   .T2EM           ;CONTINUE IF HARDWARE
003354  104401  016134              TYPE    .T2EM           ;PRINT ERRORS MESG
003360  000137  006644              JMP     EXIT            ;EXIT TEST
003364  104401  015253          2$: TYPE    ,MANMSG         ;PRINT INSTRUCTIONS
003370  032777  010000  175544  3$: BIT     #SW12,@SWR      ;SW12 UP?
003376  001774                      BEQ     3$              ;NO, WAIT FOR START

003400  012702  000041        T2SP: MOV     #41,R2          ;SET START CHAR
003404  010200                1$:   MOV     R2,R0           ;SET CHAR
003406  013701  001150        2$:   MOV     WIDTH,R1        ;SET COLUMN COUNT
003412  104415                3$:   LOAD                    ;LOAD CHAR
003414  005301                      DEC     R1              ;DEC CHAR COUNT
003416  001407                      BEQ     4$              ;BRANCH IF DONE LINE
003420  005200                      INC     R0              ;NEXT CHAR
003422  020027  000177              CMP     R0,#177         ;CHECK CHAR
003426  001371                      BNE     3$              ;OK, CONTINUE
003430  012700  000040              MOV     #40,R0          ;SET CHAR = SPACE
003434  000766                      BR      3$              ;CONTINUE
003436  104414  001162        4$:   PRINT   ,$LF            ;PRINT LINE
003442  005204                      INC     R4              ;INC LINE COUNT
003444  105737  001155              TSTB    CKFLAG          ;USING CLOCK TIMING?
003450  001006                      BNE     5$              ;YES, BYPASS MANUAL TIMING
003452  032777  010000  175462      BIT     #SW12,@SWR      ;SW12 DOWN?
003460  001002                      BNE     5$              ;NO, CONTINUE
003462  000137  003500              JMP     SPD             ;YES, EXIT PRINTING ROUTINE
003466  005202                5$:   INC     R2              ;INC START CHAR
003470  020227  000177              CMP     R2,#177         ;CHECK IT
```

```
003474  001343                  BNE     1$              ;OK, CONTINUE
003476  000740                  BR      T2SP            ;RESET START CHAR

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

                        ;ROUTINE TO PRINT/TYPE MEASURED PRINT SPEED FOR TEST 2.

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

003500  012700  000177  SPD:    MOV     #177,R0         ;SET RUBOUT CHAR
003504  104415                  LOAD                    ;CLEAR CHAR BUFFER
003506  104401  015470          TYPE    ,PRSP1          ;START MESG
003512  104414  015470          PRINT   ,PRSP1
003516  105737  001155          TSTB    CKFLAG          ;USING CLOCK?
003522  001004                  BNE     1$              ;YES, BRANCH
003524  104401  015511          TYPE    ,PRSP2          ;ADD WORD "APPROXIMATELY" TO MESG
003530  104414  015511          PRINT   ,PRSP2
003534                  1$:
003534  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
003536  004737  012650          JSR     PC,$SB2D        ;CONVERT #
003542  104417                  CNLOAD                  ;LOAD #
003544  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
003546  104405                  TYPDS                   ;TYPE #
003550  104414  015522          PRINT   ,PRSP3          ;LOAD MORE OF MESG
003554  104401  015522          TYPE    ,PRSP3
003560  013746  001150          MOV     WIDTH,-(SP)     ;#COLUMNS ON STACK
003564  104405                  TYPDS                   ;TYPE #
003566  104401  015551          TYPE    ,PRSP4          ;TYPE END OF MESG
003572  013746  001150          MOV     WIDTH,-(SP)     ;# COLUMNS ON STACK AGAIN
003576  004737  012650          JSR     PC,$SB2D        ;CONVERT #
003602  104417                  CNLOAD                  ;LOAD #
003604  104414  015551          PRINT   ,PRSP4          ;PRINT MESG ON LA180
003610  000137  006644          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
                        ;//////////////////////////////////////////////////////////////////////

003614  104413          TEST20: PRTHDR                  ;PRINT TEST HEADER
003616  012703  052452          MOV     #"*U,R3         ;SET FIRST CHAR PAIR
003622  012702  000020          MOV     #16.,R2         ;SET LINE COUNT
003626  010300          1$:     MOV     R3,R0           ;SET CHAR PAIR
003630  013701  001150          MOV     WIDTH,R1        ;SET COLUMN COUNT
003634  104415          2$:     LOAD                    ;LOAD CHAR
003636  000300                  SWAB    R0              ;SET NEXT CHAR
003640  005301                  DEC     R1              ;DEC COLUMN COUNT
003642  001374                  BNE     2$              ;FINISH LINE
003644  000303                  SWAB    R3              ;SET NEXT LINE START CHAR
003646  104414  001162          PRINT   $LF             ;PRINT LINE
003652  005302                  DEC     R2              ;DEC LINE COUNT
003654  001364                  BNE     1$              ;FINISH TEST
003656  000137  006644          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.

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

003662  104413          TEST21: PRTHDR                  ;PRINT TEST HEADER
003664  013701  001150          MOV     WIDTH,R1        ;SET COLUMN COUNT
003670  012700  000060  1$:     MOV     #60,R0          ;SET CHAR
003674  104415                  LOAD                    ;LOAD CHAR
003676  005301                  DEC     R1              ;DEC CHAR COUNT
003700  001405                  BEQ     2$              ;PRINT LINE WHEN LOADED
003702  012700  000040          MOV     #40,R0          ;SET SPACE CHAR
003706  104415                  LOAD                    ;LOAD SPACE
003710  005301                  DEC     R1              ;DEC CHAR COUNT
003712  001366                  BNE     1$              ;FINISH LINE
003714  104414  001164  2$:     PRINT   ,$CR            ;PRINT LINE
003720  012702  000002          MOV     #2,R2           ;SET FIRST CHAR COUNT
003724  012700  000040  3$:     MOV     #40,R0          ;SET SPACE CHAR
003730  010201                  MOV     R2,R1           ;GET CHAR COUNT
003732  005301                  DEC     R1              ;SUBTRACT ONE
003734  104416                  MLOAD                   ;LOAD SPACES
003736  012700  000130          MOV     #'X,R0          ;SET X CHAR
003742  104415                  LOAD                    ;LOAD X
003744  104414  001164          PRINT   ,$CR            ;PRINT LINE
003750  062702  000002          ADD     #2,R2           ;ADD 2 TO CHAR COUNT
003754  020237  001150          CMP     R2,WIDTH        ;DONE LINE?
003760  101761                  BLOS    3$              ;NO, FINISH LINE
003762  104414  001162          PRINT   ,$LF            ;YES, ADVANCE PAPER
003766  000137  006644          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.
                        ;//////////////////////////////////////////////////////////////////////////////

003772  104413          TEST22: PRTHDR                  ;PRINT TEST HEADER
003774  012702  000002          MOV     #2,R2           ;SET LINE COUNT
004000  013701  001150  1$:     MOV     WIDTH,R1        ;SET COLUMN COUNT
004004  020127  000177          CMP     R1,#127.        ;CHECK # COLUMNS
004010  003402                  BLE     2$              ;GREATER THAN 127?
004012  012701  000177          MOV     #127.,R1        ;YES, SET TO 127
004016  104414  004054  2$:     PRINT   ,10$            ;LOAD SLASH-BS-BACKSLASH
004022  005301                  DEC     R1              ;DEC CLOUMN COUNT
004024  001405                  BEQ     3$              ;PRINT LINE IF DONE
004026  012700  000055          MOV     #55,R0          ;SET ASCII OF DASH
004032  104415                  LOAD                    ;LOAD DASH CHAR
004034  005301                  DEC     R1              ;DEC COLUMN COUNT
004036  001367                  BNE     2$              ;FINISH LINE
004040  104414  001162  3$:     PRINT   ,$LF            ;PRINT LINE
004044  005302                  DEC     R2              ;DEC LINE COUNT
004046  001354                  BNE     1$              ;FINISH TEST
004050  000137  008544          JMP     EXIT            ;EXIT TEST

004054  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

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

004060  104413          TEST23: PRTHDR                  ;PRINT TEST HEADER
004062  012700  000040          MOV     #40,R0          ;SET FIRST CHAR
004066  012701  000036  1$:     MOV     #30.,R1         ;SET CHAR COUNT
004072  104416                  MLOAD                   ;LOAD LINE
004074  104414  001162          PRINT   $LF             ;PRINT LINE
004100  005200                  INC     R0              ;SET NEXT CHAR
004102  020027  000177          CMP     R0,#177         ;CHAR = RUBOUT?
004106  001367                  BNE     1$              ;NO, FINISH TEST
004110  000137  006644          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.
                        ;//////////////////////////////////////////////////////////////////////////////

004114  104413          TEST24: PRTHDR                  ;PRINT TEST HEADER
004116  012702  000041          MOV     #41,R2          ;SET START CHAR
004122  012705  000036          MOV     #30.,R5         ;SET LINE COUNT
004126  013701  001150          MOV     WIDTH,R1        ;GET COLUMN COUNT
004132  012703  000377          MOV     #255.,R3        ;SET BUFFER SIZE
004136  160103                  SUB     R1,R3           ;SUBTRACT COLUMNS COUNT
004140  005004                  CLR     R4              ;CLEAR CHAR INC COUNT
004142  162703  000035  2$:     SUB     #29.,R3         ;DIVIDE NON-PRINT CHAR COUNT BY 29
004146  002402                  BLT     3$
004150  005204                  INC     R4              ;R4 = NON-PRINT CHAR INC COUNT
004152  000773                  BR      2$
004154  005003          3$:     CLR     R3              ;CLEAR NON-PRINT CHAR COUNT 2ND BLOCK
004156  162701  000377  4$:     SUB     #255.,R1        ;CALCULATE # NON-PRINT CHARS, 1ST BLOCK
004162  005401                  NEG     R1
004164  160301                  SUB     R3,R1
004166  004737  004320          JSR     PC,10$          ;LOAD 1ST BLOCK OF NON-PRINT CHARS
004172  013701  001150          MOV     WIDTH,R1        ;SET # PRINTABLE CHARS (COLUMN COUNT)
004176  010200                  MOV     R2,R0           ;SET FIRST PRINT CHAR
004200  104415          5$:     LOAD                    ;LOAD PRINTABLE CHAR
004202  005301                  DEC     R1              ;DEC CHAR COUNT
004204  001407                  BEQ     6$              ;BRANCH IF DONE PRINTABLE CHARS
004206  005200                  INC     R0              ;NEXT CHAR
004210  020027  000177          CMP     R0,#177         ;CHECK CHAR
004214  001371                  BNE     5$              ;OK - CONTINUE
004216  012700  000040          MOV     #40,R0          ;RESET CHAR = SPACE
004222  000766                  BR      5$              ;CONTINUE
004224  010301          6$:     MOV     R3,R1           ;SET # NON-PRINT CHARS, 2ND BLOCK
004226  004737  004320          JSR     PC,10$          ;LOAD 2ND BLOCK NON-PRINT CHARS
004232  104414  001162          PRINT   ,$LF            ;PRINT LINE
004236  005305                  DEC     R5              ;DEC LINE COUNT
004240  003015                  BGT     8$              ;CONTINUE TEST
004242  105737  001154          TSTB    TLOOP           ;LOOP ON TEST?
004246  001006                  BNE     7$              ;YES, INITIALIZE
004250  032777  040000  174664  BIT     #SW14,@SWR      ;LOOP ON TEST?
004256  001002                  BNE     7$              ;YES, INITIALIZE
004260  000137  006644          JMP     EXIT            ;EXIT TEST
004264  012705  000036  7$:     MOV     #30.,R5         ;RESET LINE COUNT
004270  005003                  CLR     R3              ;CLEAR NON-PRINT CHAR COUNT
004272  000401                  BR      9$              ;CONTINUE SWIRL
004274  060403          8$:     ADD     R4,R3           ;INC NON-PRINT CHAR COUNT, 2ND BLOCK
004276  013701  001150  9$:     MOV     WIDTH,R1        ;RESET COLUMN COUNT
004302  005202                  INC     R2              ;INC START CHAR
004304  020227  000177          CMP     R2,#177         ;CHECK START CHAR
004310  001322                  BNE     4$              ;OK, CONTINUE
004312  012702  000041          MOV     #41,R2          ;RESET START CHAR
```

```
004316  000717                          BR      4$              ;CONTINUE
                                ;///////////////////////////////////////////////////////////////////////
                                ;ROUTINE TO LOAD NON-PRINTABLE CHARACTERS FOR TEST 24.
                                ;///////////////////////////////////////////////////////////////////////
004320  005701                  10$:    TST     R1              ;TEST CHAR COUNT
004322  001430                          BEQ     14$             ;RETURN IF ZERO
004324  005000                  11$:    CLR     R0              ;SET FIRST NON-PRINT CHAR
004326  104415                  12$:    LOAD                    ;LOAD CHAR
004330  005301                          DEC     R1              ;DEC CHAR COUNT
004332  003424                          BLE     14$             ;RETURN IF DONE
004334  005200                  13$:    INC     R0              ;NEXT CHAR
004336  020027  000007                  CMP     R0,#7           ;CHAR = BELL ?
004342  001774                          BEQ     13$             ;YES, NEXT CHAR
004344  020027  000010                  CMP     R0,#10          ;CHAR = BS?
004350  001771                          BEQ     13$             ;YES, NEXT CHAR
004352  020027  000012                  CMP     R0,#12          ;CHAR = LF?
004356  001766                          BEQ     13$             ;YES, NEXT CHAR
004360  020027  000014                  CMP     R0,#14          ;CHAR = FF?
004364  001763                          BEQ     13$             ;YES, NEXT CHAR
004366  020027  000015                  CMP     R0,#15          ;CHAR = CR?
004372  001760                          BEQ     13$             ;YES, NEXT CHAR
004374  020027  000040                  CMP     R0,#40          ;CHAR = SPACE?
004400  001751                          BEQ     11$             ;YES, RESET CHAR
004402  000751                          BR      12$             ;CONTINUE
004404  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.
                        ;//////////////////////////////////////////////////////////////////////////

004406  104413          TEST25: PRTHDR                  ;PRINT TEST HEADER
004410  012701  000020          MOV     #16.,   R1      ;SET CHAR COUNT
004414  012700  000105          MOV     #'E,R0          ;SET E CHAR
004420  104416                  MLOAD                   ;LOAD BUFFER
004422  012700  000177          MOV     #177,R0         ;SET RUBOUT CHAR
004426  104415                  LOAD                    ;CLEAR BUFFER
004430  005002                  CLR     R2              ;CLEAR CHAR COUNT
004432  013701  001150          MOV     WIDTH,R1        ;SET COLUMN COUNT
004436  012700  000061          MOV     #61,R0          ;SET CHAR
004442  104415          1$:     LOAD                    ;LOAD CHAR
004444  005301                  DEC     R1              ;DEC COLUMN COUNT
004446  001414                  BEQ     2$              ;PRINT LINE WHEN LOADED
004450  005202                  INC     R2              ;INC CHAR COUNT
004452  020227  000144          CMP     R2,#100.        ;DONE ONES?
004456  002771                  BLT     1$              ;NO, CONTINUE
004460  012700  000063          MOV     #63,R0          ;SET NEXT CHAR
004464  020227  000202          CMP     R2,#130.        ;DONE 3'S?
004470  002764                  BLT     1$              ;NO, CONTINUE
004472  012700  000062          MOV     #62,R0          ;YES, SET NEXT CHAR
004476  000761                  BR      1$              ;CONTINUE LOADING CHARACTERS
004500  104414  001162  2$:     PRINT   ,$LF            ;PRINT LINE
004504  012701  000400          MOV     #256.,R1        ;SET CHAR COUNT
004510  012700  000105          MOV     #'E,R0          ;SET E CHAR
004514  104416                  MLOAD                   ;LOAD BUFFER
004516  104414  001162          PRINT   ,$LF            ;PRINT BLANK LINE
004522  012701  000376          MOV     #254.,R1        ;SET CHAR COUNT
004526  012700  000105          MOV     #'E,R0          ;SET E CHAR
004532  104416                  MLOAD                   ;LOAD BUFFER
004534  012700  000177          MOV     #177,R0         ;SET RUBOUT CHAR
004540  104415                  LOAD                    ;CLEAR BUFFER
004542  104414  001162          PRINT   ,$LF            ;BLANK LINE
004546  012701  000400          MOV     #256.,R1        ;SET CHAR COUNT
004552  012700  000105          MOV     #'E,R0          ;SET E CHAR
004556  104416                  MLOAD                   ;LOAD BUFFER
004560  013701  001150          MOV     WIDTH,R1        ;SET COLUMN COUNT
004564  012700  000060          MOV     #60,R0          ;SET CHAR
004570  005002                  CLR     R2              ;CLEAR CHAR COUNT
004572  104415          3$:     LOAD                    ;LOAD CHAR
004574  005301                  DEC     R1              ;DEC COLUMN COUNT
004576  001407                  BEQ     4$              ;PRINT LINE WHEN LOADED
004600  005202                  INC     R2              ;INC CHAR COUNT
004602  020227  000143          CMP     R2,#99.         ;DONE ZEROS?
004606  002771                  BLT     3$              ;NO, CONTINUE
004610  012700  000061          MOV     #61,R0          ;YES, SET NEXT CHAR
004614  000766                  BR      3$              ;CONTINUE
```

```
004616  104414  001162          4$:     PRINT   ,$LF           ;PRINT LINE
004622  012701  000400                  MOV     #256.,R1        ;SET CHAR COUNT
004626  012700  000105                  MOV     #'E,R0          ;SET E CHAR
004632  104416                          MLOAD                   ;LOAD BUFFER
004634  012700  000177                  MOV     #177,R0         ;SET RUBOUT CHAR
004640  104415                          LOAD                    ;CLEAR BUFFER
004642  012700  000060                  MOV     #60.R0          ;SET CHAR
004646  012702  000011                  MOV     #9.,R2          ;SET GROUP COUNT
004652  013701  001150                  MOV     WIDTH,R1        ;SET CHAR COUNT
004656  104415                  5$:     LOAD                    ;LOAD CHAR
004660  005302                          DEC     R2              ;DEC GROUP COUNT
004662  001010                          BNE     7$              ;FINISH GROUP
004664  005200                          INC     R0              ;SET NEXT CHAR
004666  020027  000072                  CMP     R0,#72          ;GOOD CHAR?
004672  103402                          BLO     6$              ;YES, CONTINUE
004674  012700  000060                  MOV     #60.R0          ;NO, RESET CHAR TO ZERO
004700  012702  000012          6$:     MOV     #10.,R2         ;RESET GROUP COUNT
004704  005301                  7$:     DEC     R1              ;DEC CHAR COUNT
004706  001363                          BNE     5$              ;FINISH LINE
004710  104414  001162                  PRINT   ,$LF            ;PRINT LINE
004714  012701  000400                  MOV     #256.,R1        ;SET CHAR COUNT
004720  012700  000105                  MOV     #'E,R0          ;SET E CHAR
004724  104416                          MLOAD                   ;LOAD BUFFER
004726  012700  000061                  MOV     #61,R0          ;SET CHAR
004732  013701  001150                  MOV     WIDTH,R1        ;SET CHAR COUNT
004736  104415                  8$:     LOAD                    ;LOAD CHAR
004740  005200                          INC     R0              ;SET NEXT CHAR
004742  020027  000072                  CMP     R0,#72          ;GOOD CHAR?
004746  103402                          BLO     9$              ;YES, CONTINUE
004750  012700  000060                  MOV     #60.R0          ;RESET CHAR
004754  005301                  9$:     DEC     R1              ;DEC CHAR COUNT
004756  001367                          BNE     8$              ;FINISH LINE
004760  104414  001162                  PRINT   ,$LF            ;PRINT LINE
004764  000137  006644                  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.
                        ;//////////////////////////////////////////////////////////////////////////////

004770  104413          TEST26: PRTHDR                  ;PRINT TEST HEADER
004772  012703  005050          MOV     #6$,R3          ;SET TABLE POINTER
004776  012702  000003  1$:     MOV     #3,R2           ;SET OVERPRINT COUNT
005002  011300          2$:     MOV     (R3),R0         ;GET CHAR PAIR
005004  001417                  BEQ     5$              ;EXIT IF DONE TEST
005006  013701  001150          MOV     WIDTH,R1        ;SET COLUMN COUNT
005012  104415          3$:     LOAD                    ;LOAD CHAR
005014  000300                  SWAB    R0              ;SET NEXT CHAR
005016  005301                  DEC     R1              ;DEC COLUMN COUNT
005020  001374                  BNE     3$              ;FINISH LINE
005022  005302                  DEC     R2              ;DEC OVERPRINT COUNT
005024  001403                  BEQ     4$              ;BRANCH IF DONE OVERPRINT
005026  104414  001164          PRINT   ,$CR            ;PRINT LINE
005032  000763                  BR      2$              ;CONTINUE
005034  104414  001162  4$:     PRINT   ,$LF            ;PRINT LINE
005040  005723                  TST     (R3)+           ;INC TABLE POINTER
005042  000755                  BR      1$              ;PRINT NEXT LINE
005044  000137  006644  5$:     JMP     EXIT            ;EXIT TEST

005050  020105          6$:     .ASCII  /E /
005052  040040                  .ASCII  / @/
005054  020115                  .ASCII  /M /
005056  021440                  .ASCII  / #/
005060  000000                  .WORD   0               ;END OF TABLE
```

```
                        ;//////////////////////////////////////////////////////////////////////////////
                        ;TEST27 -       MULTIPLE LINE FEED TEST
                        ;
                        ;NUMBER PRINTED INDICATES NUMBER OF LINE FEEDS FOLLOWING THAT LINE.
                        ;DASHED REFERENCE LINES ARE PRINTED TO AID IN CHECKING PROPER
                        ;LINE FEEDS.
                        ;//////////////////////////////////////////////////////////////////////////////

005062  104413          TEST27: PRTHDR                  ;PRINT TEST HEADER
005064  012703  005170          MOV     #5$,R3          ;SET TABLE POINTER
005070  111346  1$:      MOVB    (R3),-(SP)      ;NUMBER ONTO STACK
005072  004737  012650          JSR     PC,$$B2D        ;CONVERT #
005076  104417                  CNLOAD                  ;LOAD NUMBER
005100  121327  000001          CMPB    (R3),#1         ;TEST #
005104  101406                  BLOS    2$              ;PRINT FULL DASH LINE IF 0 OR 1
005106  012701  000035          MOV     #29.,R1         ;SET DASH LENGTH
005112  121327  000010          CMPB    (R3),#8.        ;CHECK #
005116  101411                  BLOS    4$              ;2,4 OR 8 - PRINT 29 DASHES
005120  000407                  BR      3$              ;16 OR 32 - PRINT 28 DASHES
005122  013701  001150  2$:      MOV     WIDTH,R1        ;SET CHAR COUNT
005126  020127  000036          CMP     R1,#30.         ;CHECK IT
005132  103002                  BHIS    3$              ;OK, CONTINUE
005134  012701  000036          MOV     #30.,R1         ;SET TO 30
005140  005301  3$:      DEC     R1              ;SUBTRACT ONE
005142  012700  000055  4$:      MOV     #55,R0          ;SET DASH CHAR
005146  104416                  MLOAD                   ;LOAD DASH LINE
005150  111301                  MOVB    (R3),R1         ;SET LF COUNT
005152  113700  001162          MOVB    $LF,R0          ;SET LF CHAR
005156  104416                  MLOAD                   ;LOAD LF'S
005160  105723                  TSTB    (R3)+           ;INC TABLE POINTER
005162  003342                  BGT     1$              ;FINISH TEST
005164  000137  006644          JMP     EXIT            ;EXIT TEST

005170      001     002     004  5$:     .BYTE   1,2,4,8.,16.,32.,0
005173      010     020     040
005176      000

        005200                  .EVEN
```

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

                        ;TEST30 -       RIBBON FEED TEST
                        ;
                        ;THIS TEST PRINTS A SINGLE COLUMN OF 24 LINES OF X'S DOWN THE LEFT
                        ;HAND MARGIN OF THE PAGE.

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

005200  104413          TEST30: PRTHDR                  ;PRINT TEST HEADER
005202  012701  000030          MOV     #24.,R1         ;SET LINE COUNT
005206  104414  005222  1$:     PRINT   2$              ;PRINT X - LF
005212  005301                  DEC     R1              ;DEC LINE COUNT
005214  001374                  BNE     1$              ;FINISH TEST
005216  000137  006644          JMP     EXIT            ;EXIT TEST

005222  005130     000  2$:     .ASCIZ  /X/<12>

        005226                  .EVEN


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

                        ;TEST31 -       BELL TEST
                        ;
                        ;THIS TEST WILL SOUND 5 BELLS WHILE PRINTING "BELL TEST"

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

005226  104413          TEST31: PRTHDR                  ;PRINT TEST HEADER
005230  104414  005240          PRINT   1$              ;DO TEST
005234  000137  005262          JMP     $EOP            ;EXIT TEST

005240  041007  046105  003514  1$:     .ASCIZ  <7>/BELL/<7>/ TEST/<7><15><7><12><7><15>
005246  052040  051505  003524
005254  003415  003412  000015

                                .EVEN
```

```
                                END OF PASS ROUTINE

                                .SBTTL  END OF PASS ROUTINE

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

005262                          $EOP:
005262  000240                          NOP
005264  005237  005336                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
005270  042737  100000  005336          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
005276  005327                          DEC     (PC)+           ;;LOOP?
005300  000001                $EOPCT: .WORD   1
005302  003013                          BGT     $DOAGN          ;;YES
005304  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
005306  000001                $ENDCT: .WORD   1
005310  005300                          $EOPCT
005312  013700  000042        $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
005316  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
005320  000005                          RESET                   ;;CLEAR THE WORLD
005322  004710                $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
005324  000240                          NOP                     ;;SAVE ROOM
005326  000240                          NOP                     ;;FOR
005330  000240                          NOP                     ;;ACT11
005332                          $DOAGN:
005332  000137                          JMP     @(PC)+          ;;RETURN
005334  006644                $RTNAD: .WORD   EXIT
005336  000000                $PASS:  .WORD   0               ;;NUMBER OF PASSES
```

```
                        .SBTTL  OPTION TESTS

                        ;//////////////////////////////////////////////////////////////////////////
                        ;
                        ;TEST 50 - SECONDARY CHARACTER SET
                        ;
                        ;//////////////////////////////////////////////////////////////////////////

005340  104413          TEST50: PRTHDR                  ;PRINT TEST HEADER
005342  012737  000004  005422        MOV     #4,T50PCT       ;SET PAIR COUNT
005350  104414  005424  1$:     PRINT   ,T50M1          ;PRINT LEADER # 1
005354  012737  000040  005440        MOV     #40,CHLD        ;SET CHAR (SET # 1)
005362  004737  005442                JSR     PC,PRTSET       ;PRINT ENTIRE CHARACTER SET
005366  104414  005430                PRINT   ,T50M2          ;PRINT LEADER # 2
005372  012737  000240  005440        MOV     #240,CHLD       ;SET CHAR (SET # 2)
005400  004737  005442                JSR     PC,PRTSET       ;PRINT ENTIRE CHARACTER SET
005404  104414  001162                PRINT   ,$LF            ;BLANK LINE
005410  005337  005422                DEC     T50PCT          ;DECREMENT PAIR COUNTER
005414  001355                        BNE     1$              ;CONTINUE IF NOT ZERO
005416  000137  006644                JMP     EXIT            ;EXIT TEST

005422  000000          T50PCT: 0
005424  030443  000075  T50M1:  .ASCIZ  /#1=/
005430  031043  000075  T50M2:  .ASCIZ  /#2=/
005434  020040  000040  SP3:    .ASCIZ  /   /               ;3 SPACES
005440  000000          CHLD:   0                       ;CHARACTER HOLD

005442  013701  001150  PRTSET: MOV     WIDTH,R1        ;SET COLUMN COUNT
005446  162701  000003                SUB     #3,R1           ;ADJUST COUNT
005452  013700  005440  1$:     MOV     CHLD,R0         ;FETCH CURRENT CHARACTER
005456  104415                      LOAD                    ;LOAD CHAR
005460  005301                      DEC     R1              ;DEC CHAR COUNT
005462  003412                      BLE     3$              ;BRANCH IF # OF COLUMNS IS EXHAUSTED
005464  005237  005440  2$:     INC     CHLD            ;INCREMENT CHAR
005470  013702  005440                MOV     CHLD,R2         ;COPY IT
005474  042702  177600                BIC     #177600,R2      ;STRIP LEFT BITS
005500  022702  000177                CMP     #177,R2         ;IS RESIDUE = 177?
005504  001412                        BEQ     4$              ;YES, BRANCH, LAST CHAR OF SET IS LOADED
005506  000761                        BR      1$              ;GO LOAD NEXT CHAR
005510  104414  001162  3$:     PRINT   ,$LF            ;PRINT PARTIAL SET
005514  013701  001150                MOV     WIDTH,R1        ;SET COLUMN COUNT AGAIN
005520  162701  000003                SUB     #3,R1           ;ADJUST COUNT AGAIN
005524  104414  005434                PRINT   ,SP3            ;LOAD THREE SPACES
005530  000755                        BR      2$              ;CONTINUE LOADING SET
005532  104414  001162  4$:     PRINT   ,$LF            ;PRINT LINE
005536  000207                      RTS     PC              ;RETURN
```

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

                        .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.

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

005540  005037  005652  TEST60: CLR     PASCNT          ;CLEAR PASS COUNT
005544  104413          1$:     PRTHDR                  ;PRINT TEST HEADER, FEED BLANK LINES
005546  012700  000041          MOV     #41,R0          ;SET FIRST CHAR
005552  013701  001150  2$:     MOV     WIDTH,R1        ;SET COLUMN COUNT
005556  104416                  MLOAD                   ;LOAD LINE
005560  104414  001162          PRINT   ,$LF            ;PRINT LINE
005564  012702  000005          MOV     #5,R2           ;SET OVERPRINT COUNT
005570  013701  001150  3$:     MOV     WIDTH,R1        ;SET COLUMN COUNT
005574  104416                  MLOAD                   ;LOAD LINE
005576  104414  001164          PRINT   ,$CR            ;PRINT LINE
005602  005302                  DEC     R2              ;DEC OVERPRINT COUNT
005604  001371                  BNE     3$              ;FINISH OVERPRINT
005606  104414  001162          PRINT   ,$LF            ;ADVANCE PAPER
005612  005200                  INC     R0              ;SET NEXT CHAR
005614  020027  000177          CMP     R0,#177         ;TEST CHAR
005620  001354                  BNE     2$              ;OK, CONTINUE
005622  005237  005652          INC     PASCNT          ;INC PASS COUNT
005626  104414  015102          PRINT   ,PASMSG         ;LOAD PASS COUNT MESG
005632  013746  005652          MOV     PASCNT,-(SP)    ;;PUSH PASCNT ON STACK
005636  004737  012650          JSR     PC,$SB2D        ;CONVERT #
005642  104417                  CNLOAD                  ;LOAD #
005644  104414  001162          PRINT   ,$LF            ;PRINT MESG
005650  000735                  BR      1$              ;START NEXT PASS
005652  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.

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

005654  104413          TEST61: PRTHDR                  ;PRINT TEST HEADER
005656  000422                  BR      11$             ;CHECK SWITCH REG FIRST
005660  013701  001150  10$:    MOV     WIDTH,R1        ;SET COLUMN COUNT
005664  017700  173252  1$:     MOV     @SWR,R0         ;GET CHARACTER
005670  042700  177600          BIC     #^C177,R0       ;MASK UNWANTED BITS
005674  104415                  LOAD                    ;LOAD CHAR
005676  020027  000015          CMP     R0,#15          ;CHAR = CR?
005702  001410                  BEQ     11$             ;YES, RESET COLUMN COUNT
005704  020027  000012          CMP     R0,#12          ;CHAR = LF?
005710  001405                  BEQ     11$             ;YES, RESET COLUMN COUNT
005712  020027  000040          CMP     R0,#40          ;NON-PRINTABLE CHAR?
005716  103404                  BLO     12$             ;YES, DON'T DEC COLUMN COUNT
005720  005301                  DEC     R1              ;DEC COLUMN COUNT
005722  000402                  BR      12$             ;CONTINUE
005724  013701  001150  11$:    MOV     WIDTH,R1        ;RESET COLUMN COUNT
005730  032777  001000  173204  12$:    BIT     #SW9,@SWR       ;TEST SWITCH 9
005736  001402                  BEQ     2$              ;PRINT FULL LINE
005740  000000                  HALT                    ;ONE CHAR - HALT
005742  000750                  BR      1$              ;GET NEXT CHAR
005744  005701          2$:     TST     R1              ;TEST COLUMN COUNT
005746  003346                  BGT     1$              ;CONTINUE IF NOT DONE LINE
005750  001403                  BEQ     3$              ;LOADED MORE THAN WIDTH?
005752  012700  000177          MOV     #177,R0         ;YES, CLEAR BUFFER
005756  104415                  LOAD
005760  104414  001162  3$:     PRINT   $LF             ;PRINT LINE, ADVANCE PAPER
005764  000735                  BR      10$             ;CONTINUE
```

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

                        ;TEST62 -       LINE PRINT TEST
                        ;
                        ;THIS TEST PRINTS FULL LINES CONTINUOUSLY OF WHATEVER CHARACTER
                        ;IS TYPED ON THE CONSOLE KEYBOARD.  TO CHANGE CHARACTERS, RESELECT
                        ;THIS TEST.  AN ERROR MESG WILL BE PRINTED IF THIS TEST IS SELECTED
                        ;AND A CONSOLE TERMINAL DOES NOT EXIST.

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

005766  104413          TEST62: PRTHDR                          ;PRINT TEST HEADER
005770  105737  001173          TSTB    $TPFLG                 ;CHECK IF TERMINAL EXISTS
005774  100473                  BMI     TERR                   ;EXIT IF NONE
005776  104401  015205          TYPE    ,TCHAR                 ;TYPE INSTR
006002  105777  173072  2$:     TSTB    @TKS                   ;WAIT FOR KYBD FLAG
006006  100375                  BPL     2$
006010  104420                  CHECK                          ;CHECK CHAR FOR CONTROL
006012  017700  173064          MOV     @TKB,R0                ;GET CHAR
006016  010046                  MOV     R0,-(SP)               ;CHAR ONTO STACK
006020  004737  010344          JSR     PC,$TYPEC              ;ECHO CHAR
006024  012746  000200          MOV     #CRLF,-(SP)            ;SEND CR-LF
006030  004737  010344          JSR     PC,$TYPEC
006034  013701  001150  1$:     MOV     WIDTH,R1               ;SET COLUMN COUNT
006040  104416                  MLOAD                          ;LOAD LINE
006042  104414  001162          PRINT   $LF                    ;PRINT LINE
006046  000772                  BR      1$                     ;CONTINUE
```

```
                        ;//////////////////////////////////////////////////////////////////////
                        ;TEST63 -       CHARACTER PRINT TEST
                        ;
                        ;THIS TEST LOADS WHATEVER CHARACTER IS TYPED ON THE CONSOLE KEYBOARD
                        ;TO THE LA180, CHARACTER BY CHARACTER.
                        ;IF THIS TEST IS SELECTED AND A CONSOLE TERMINAL DOES NOT EXIST,
                        ;AN ERROR MESSAGE WILL BE PRINTED.
                        ;//////////////////////////////////////////////////////////////////////

006050  104413          TEST63: PRTHDR                  ;PRINT TEST HEADER
006052  105737  001173          TSTB    $TPFLG          ;CHECK IF TERMINAL EXISTS
006056  100442                  BMI     TERR            ;EXIT IF NONE
006060  104401  015205          TYPE    ,TCHAR          ;TYPE INSTR
006064  104401  001161          TYPE    ,$CRLF          ;SEND CR-LF
006070  105777  173004  1$:     TSTB    @TKS            ;WAIT FOR KYBD FLAG
006074  100375                  BPL     1$              ;WAIT FOR FLAG
006076  104420                  CHECK                   ;CHECK CHAR FOR CONTROL
006100  017700  172776          MOV     @TKB,R0         ;GET CHAR
006104  010046                  MOV     R0,-(SP)        ;CHAR ONTO STACK
006106  004737  010344          JSR     PC,$TYPEC       ;ECHO CHAR
006112  104415                  LOAD                    ;LOAD CHAR
006114  020027  000015          CMP     R0,#CR          ;SEND LF AFTER CR
006120  001003                  BNE     2$
006122  012746  000012          MOV     #LF,-(SP)
006126  000413                  BR      4$
006130  020027  000012  2$:     CMP     R0,#LF          ;SEND CR AFTER LF
006134  001003                  BNE     3$
006136  012746  000015          MOV     #CR,-(SP)
006142  000405                  BR      4$
006144  020027  000014  3$:     CMP     R0,#FF          ;SND CRLF AFTER FF
006150  001347                  BNE     1$
006152  012746  000200          MOV     #CRLF,-(SP)
006156  004737  010344  4$:     JSR     PC,$TYPEC
006162  000742                  BR      1$              ;CONTINUE

006164  104414  015225  TERR:   PRINT   ,NCMSG          ;PRINT NO CONSOLE MESG
006170  000137  006752          JMP     SELECT          ;RETURN TO SELECT TEST (HAL
```

```
                        ;//////////////////////////////////////////////////////////////////////
                        ;
                        ;TEST 64 -     SELECTED PATTERN PRINT TEST
                        ;
                        ;1) TYPE DESIRED CHARACTERS INCLUDING, WHERE APPROPRIATE, LA180
                        ;   PRINT COMMANDS (IE., LF OR CR OR FF). TERMINATE STRING BY
                        ;   TYPING CTL-A. PRINTING WILL BEGIN.
                        ;
                        ;OR
                        ;
                        ;2) TYPE A DESIRED PATTERN (TO BE AUTOMATICALLY REPEATED UNTIL
                        ;   PREVIOUSLY SPECIFIED # OF COLUMNS HAS BEEN FILLED).
                        ;   DO NOT INSERT ANY PRINT COMMANDS. TERMINATE PATTERN BY TYPING
                        ;   CTL-B. AFTER PROPAGATING THE PATTERN TO THE PROPER # OF COLUMNS
                        ;   AN LF PRINT COMMAND WILL BE INSERTED AND PRINTING WILL BEGIN.
                        ;
                        ;TO STOP PRINTING AND ENTER A NEW PATTERN TYPE CTL-SPACE.
                        ;TO SELECT A NEW TEST TYPE DELETE/RUBOUT.
                        ;TO CHANGE # OF COLUMNS TYPE CTL-C.
                        ;
                        ;//////////////////////////////////////////////////////////////////////

006174  104413          TEST64: PRTHDR                  ;PRINT TEST HEADER
006176  105737  001173          TSTB    $TPFLG          ;CHECK IF CONSOLE TERMINAL EXISTS
006202  100770                  BMI     TERR            ;EXIT IF NONE
006204  104401  015415          TYPE    ,T64MSG         ;TYPE BRIEF INSTRS
006210  104401  015464  ACCEPT: TYPE    ,TWOBEL         ;RING CONSOLE BELL TWICE
006214  012701  016174          MOV     #TABL64,R1      ;SET POINTER LEFTMOST IN TABLE
006220  012700  000177          MOV     #177,R0
006224  104415                  LOAD                    ;RESET PRINTER BUFFER POINTER.
006226  105777  172646  2$:     TSTB    @TKS
006232  100375                  BPL     2$              ;WAIT FOR CONSOLE KBD FLAG
006234  104420                  CHECK                   ;CHECK FOR KBD CONTROL CHARACTERS
                                                        ;AND CHECK SWITCHES ALSO.
006236  017700  172640          MOV     @TKB,R0         ;SAME CHAR TO R0
006242  042700  177600          BIC     #177600,R0      ;STRIP UNWANTED BITS
006246  022700  000001          CMP     #1,R0
006252  001435                  BEQ     CTLA            ;BRANCH IF CONTROL A
006254  022700  000002          CMP     #2,R0
006260  001440                  BEQ     CTLB            ;BRANCH IF CONTROL B
006262  010046                  MOV     R0,-(SP)        ;CHAR ONTO STACK
006264  004737  010344          JSR     PC,$TYPEC       ;ECHO CHARACTER TO CONSOLE
006270  005726                  TST     (SP)+           ;ADJUST STACK
006272  110021                  MOVB    R0,(R1)+        ;CHAR TO PRINT TABLE
006274  020027  000015          CMP     R0,#CR          ;SEND LF AFTER CR (TO CONSOLE)
006300  001003                  BNE     4$
006302  012746  000012          MOV     #LF,-(SP)
006306  000413                  BR      8$
006310  020027  000012  4$:     CMP     R0,#LF          ;SEND CR AFTER LF (TO CONSOLE)
006314  001003                  BNE     6$
006316  012746  000015          MOV     #CR,-(SP)
006322  000405                  BR      8$
006324  020027  000014  6$:     CMP     R0,#FF          ;SEND CRLF AFTER FF (TO CONSOLE)
006330  001336                  BNE     2$              ;CONTINUE ACCEPTING PATTERN
006332  012746  000200          MOV     #CRLF,-(SP)
006336  004737  010344  8$:     JSR     PC,$TYPEC
```

```
006342  005726                          TST     (SP)+           ;ADJUST STACK
006344  000730                          BR      2$              ;CONTINUE ACCEPTING PATTERN

006346  022701  016174          CTLA:   CMP     #TABL64,R1      ;IF POINTER IS LEFTMOST
006352  001445                          BEQ     PRPAT           ;RESUME PRINTING PREVIOUS PATTERN
006354  010137  006512                  MOV     R1,ENDADR       ;CURRENT POINTER TO ENDADR
006360  000442                          BR      PRPAT           ;GO PRINT NEW PATTERN

006362  022701  016174          CTLB:   CMP     #TABL64,R1      ;IF POINTER IS LEFTMOST
006366  001437                          BEQ     PRPAT           ;RESUME PRINTING PREVIOUS PATTERN
006370  162701  016174                  SUB     #TABL64,R1      ;GET PATTERN LENGTH INTO R1
006374  013737  001150  006506          MOV     WIDTH,COLCTR    ;GET # OF COLUMNS INTO COUNTER
006402  010137  006510                  MOV     R1,CHACTR       ;GET # OF CHARS INTO COUNTER
006406  062701  016174                  ADD     #TABL64,R1      ;RE-ESTABLISH POINTER
006412  012702  016174                  MOV     #TABL64,R2      ;ESTABLISH SECONDARY POINTER
006416  163737  006510  006506          SUB     CHACTR,COLCTR   ;REMAINING COLUMNS INTO COLCTR
006424  112221          PROPAT: MOVB    (R2)+,(R1)+     ;PROPAGATE PATTERN TO # OF SPECIFIED COLUMNS
006426  005337  006506                  DEC     COLCTR
006432  003001                          BGT     1$
006434  000407                          BR      DUNCOL
006436  005337  006510          1$:     DEC     CHACTR
006442  001401                          BEQ     DUNPAT
006444  000767                          BR      PROPAT
006446  012702  016174          DUNPAT: MOV     #TABL64,R2      ;RE-ADDRESS PATTERN
006452  000764                          BR      PROPAT
006454  112721  000012          DUNCOL: MOVB    #LF,(R1)+       ;INSERT LA180 PRINT COMMAND
006460  010137  006512                  MOV     R1,ENDADR
006464  000400                          BR      PRPAT           ;GO PRINT REPEATED PATTERN

006466  012701  016174          PRPAT:  MOV     #TABL64,R1      ;SET POINTER LEFTMOST
006472  112120          1$:     MOVB    (R1)+,R0
006474  104415                  LOAD                    ;LOAD CHAR (FROM R0) TO LA180
                                                        ;ALSO CHK CONSOLE KBD AND CPU SWS
006476  020137  006512                  CMP     R1,ENDADR       ;END OF TABLE DATA?
006502  001771                          BEQ     PRPAT           ;YES, PRINT IT OVER.
006504  000772                          BR      1$              ;NO, CONTINUE LOADING LA180 BUFFER

006506  000000          COLCTR: 0                       ;COLUMN COUNTER
006510  000000          CHACTR: 0                       ;CHARACTER COUNTER
006512  016240          ENDADR: #TABL64+44              ;HOLDS ENDING ADDR+1 OF PRINT DATA.


006514  005037  001120          TEST76: CLR     SERSW           ;CLEAR SWITCH (TO PARALLEL)
006520  004737  001764                  JSR     PC,QUD1         ;INITILIZE ACTFST AND SELECT DRIVER
006524  004737  002032                  JSR     PC,QUD2         ;TYPE INTF MISC IF CONSOLE
006530  000137  006644                  JMP     EXIT            ;EXIT TEST.


006534  012737  052525  001120  TEST77: MOV     #52525,SERSW    ;SET SWITCH (TO SERIAL)
006542  004737  001764                  JSR     PC,QUD1         ;INITILIZE ACTFST AND SELECT DRIVER
006546  004737  002032                  JSR     PC,QUD2         ;TYPE INTF MISC IF CONSOLE
006552  000137  006644                  JMP     EXIT            ;EXIT TEST
```

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

                                .SBTTL  CLOCK INTERRUPT SERVICE ROUTINES

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

006556  004737  006614        DCI:    JSR     PC,SRDCI        ;DISABLE KW11-P INTERRUPT
006562  000177  000022                JMP     @TORTN          ;TIME OUT RETURN

006566  005337  006606        LKSRV:  DEC     CNTR            ;DEC TIME COUNT
006572  001401                        BEQ     1$              ;TIME OUT?
006574  000002                        RTI                     ;NO, RETURN
006576  004737  006614        1$:     JSR     PC,SRDCI        ;DISABLE KW11-L INTERRUPT
006602  000177  000002                JMP     @TORTN          ;TIME OUT RETURN

006606  000000                CNTR:   .WORD   0               ;KW11-L TIMING COUNT
006610  000000                TORTN:  .WORD   0               ;TIME OUT RETURN ADDRESS

006612  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

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

006614  105737  001155        SRDCI:  TSTB    CKFLAG          ;CHECK CLOCK OPTION FLAG
006620  001410                        BEQ     2$              ;RETURN IF NONE
006622  002404                        BLT     1$              ;BRANCH IF KW11-P
006624  042777  000100  172306        BIC     #BIT6,@LKS      ;DISABLE KW11-L INTERRUPT
006632  000403                        BR      2$              ;RETURN
006634  042777  000100  172272  1$:   BIC     #BIT6,@PLKS     ;DISABLE KW11-P INTERRUPT
006642  000207                2$:     RTS     PC
```

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

                                        .SBTTL  TEST EXIT ROUTINE

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

006644  004737  007140          EXIT:   JSR     PC,KYBDF        ;CHECK FOR KYBD FLAG
006650  032777  040000  172264          BIT     #SW14,@SWR      ;LOOP ON TEST SWITCH?
006656  001016                          BNE     3$              ;YES, RETURN TO TEST
006660  032777  000400  172254          BIT     #SW8,@SWR       ;WANT SW REG TEST SELECTION?
006666  001031                          BNE     SELECT          ;YES, SELECT TEST (HALT)
006670  105737  001154                  TSTB    TLOOP           ;KYBD CTRL - LOOP ON TEST?
006674  001007                          BNE     3$              ;YES, RETURN TO TEST
006676  105737  001153                  TSTB    TRONE           ;KYBD CTRL - RUN TEST ONCE?
006702  001402                          BEQ     2$              ;NO, CONTINUE
006704  000137  007510                  JMP     TSEL            ;YES, SELECT TEST
006710  005237  001146          2$:     INC     $TSTNM          ;INCREMENT TEST NUMBER
006714  013700  001146          3$:     MOV     $TSTNM,R0       ;GET NEW TEST NUMBER
006720  006300                          ASL     R0              ;SET TABLE POINTER
006722  005760  013356                  TST     TAT(R0)         ;CHECK ADDRESS IN TABLE
006726  003005                          BGT     4$              ;OK, GO TO TEST
006730  001767                          BEQ     2$              ;ZERO, SKIP TEST
006732  012737  000020  001146          MOV     #20,$TSTNM      ;END OF SEQ. - RESET TEST # TO 20
006740  000765                          BR      3$              ;TEST NEW NUMBER
006742  012706  001100          4$:     MOV     #STACK,SP       ;RESET STACK
006746  000170  013356                  JMP     @TAT(R0)        ;GO TO TEST
```

```
                                TEST EXIT ROUTINE

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

                                .SBTTL  ROUTINES TO SELECT DESIRED TEST


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

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

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

006752  004737  006614        SELECT: JSR     PC,SRDCI        ;CLEAR CLOCK INTER
006756  105037  001152                CLRB    STRONE          ;CLEAR PROGRAM CONTROL FLAGS
006762  105037  001153                CLRB    TRONE
006766  105037  001154                CLRB    TLOOP
006772  004737  007140                JSR     PC,KYBDF        ;CHECK IF KYBD FLAG
006776  000000                1$:     HALT                    ;NO KYBD FLAG - HALT
                                                                ;WAIT FOR OPERATOR TO SELECT TEST
                                                                ;PRESS CONTINUE WHEN READY
007000  032777  000400  172134        BIT     #SW8,@SWR       ;CHECK SW8
007006  001403                      BEQ     2$              ;WANT TO RUN TEST ONCE & HALT
007010  112737  177777  001152      MOVB    #-1,STRONE      ;YES, SET FLAG
007016  017700  172120        2$:     MOV     @SWR,R0         ;NO, CHECK SW REG
007022  042700  177700                BIC     #^C77,R0        ;MASK BITS 0-5
007026  010037  001146                MOV     R0,$TSTNM       ;SAVE TEST NUMBER
007032  006300                      ASL     R0              ;SET TABLE POINTER
007034  005760  013356              TST     TAT(R0)         ;CHECK IF TEST IS IN TABLE
007040  003744                      BLE     SELECT          ;NOT THERE, GET NEW SELECTION
007042  012706  001100              MOV     #STACK,SP       ;RESET STACK
007046  000170  013356              JMP     @TAT(R0)        ;GO TO SELECTED TEST

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

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

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

007052  004737  007140        $CHECK: JSR     PC,KYBDF        ;CHECK FOR KYBD FLAG
007056  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
007060  005000                      CLR     R0              ;CLEAR R0
007062  032777  000400  172052      BIT     #SW8,@SWR       ;CHECK SW8
007070  001401                      BEQ     1$              ;BRANCH IF SW DOWN
007072  005300                      DEC     R0              ;SET FLAG IF UP
007074  123700  001152        1$:     CMPB    STRONE,R0       ;CHANGE IN SWITCH?
007100  001402                      BEQ     2$              ;NO, RETURN (CHANGE TO 0402 TO DISABLE SW 8)
007102  000137  006752              JMP     SELECT          ;YES,  SELECT TEST (HALT)
007106                          2$:
007106  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
007110  000002                      RTI                     ;RETURN
```

```
                                ;////////////////////////////////////////////////////////////////////////////////
                                ;ROUTINE TO WAIT FOR OPERATOR ACTION
                                ;////////////////////////////////////////////////////////////////////////////////

007112  105737  001173      $HOLD:  TSTB    $TPFLG          ;TERMINAL THERE?
007116  100002                      BPL     1$              ;BRANCH IF YES
007120  000000                      HALT                    ;HALT IF NO
007122  000405                      BR      2$              ;RETURN ON CONTINUE SWITCH
007124  104401  015144      1$:     TYPE    ,WTMSG          ;TYPE WAIT MESG
007130  105777  171744      3$:     TSTB    @TKS            ;KYBD FLAG?
007134  100375                      BPL     3$              ;NO, WAIT FOR FLAG
007136  000002              2$:     RTI                     ;RETURN

                                ;////////////////////////////////////////////////////////////////////////////////
                                ;ROUTINE TO CHECK FOR KEYBOARD FLAGS
                                ;WHEN LOOKING FOR CONTROL FROM THE CONSOLE DEVICE KEYBOARD
                                ;CALL:  JSR     PC,KYBDF
                                ;////////////////////////////////////////////////////////////////////////////////

007140  105737  001173      KYBDF:  TSTB    $TPFLG          ;TERMINAL THERE?
007144  100527                      BMI     1$              ;NO, EXIT
007146  105777  171726              TSTB    @TKS            ;FLAG SET?
007152  100124                      BPL     1$              ;NO - EXIT
007154  104406              2$:     RDCHR                   ;FLAG SET - READ CHAR
007156  023727  001142  000176      CMP     SWR,#SWREG      ;USING HARDWARE SW REG?
007164  001101                      BNE     4$              ;BRANCH IF YES
007166  022716  000007              CMP     #7,(SP)         ;CHAR = BEL <007>?
007172  001076                      BNE     4$              ;BRANCH IF NOT
007174  005726              5$:     TST     (SP)+           ;RESET STACK
007176  005037  007436              CLR     20$             ;CLEAR NEW SW SETTING
007202  005037  007440              CLR     30$             ;CLEAR INPUT FLAG
007206  104401  015122              TYPE    ,DSMSG1         ;TYPE MESG
007212  013746  000176              MOV     $WREG,-(SP)     ;;PUSH SWREG ON STACK
007216  104402                      TYPOC                   ;TYPE IT
007220  104401  015132              TYPE    ,DSMSG2         ;TYPE MORE OF MESG
007224  104406              9$:     RDCHR                   ;READ CHAR
007226  021627  000025              CMP     (SP),#25        ;CHAR = CONTROL-U ?
007232  001003                      BNE     10$             ;BRANCH IF NOT
007234  104401  012242              TYPE    ,$CNTLU         ;ECHO CONTROL-U
007240  000755                      BR      5$              ;RESTART ROUTINE
007242  021627  000015      10$:    CMP     (SP),#CR        ;CHAR = CR?
007246  001011                      BNE     7$              ;BRANCH IF NOT
007250  104401  001161              TYPE    ,$CRLF          ;ECHO CR-LF
007254  005737  007440              TST     @30$            ;CHECK INPUT FLAG
007260  001460                      BEQ     6$              ;LEAVE SW SETTINGS ALONE IF NO INPUT
007262  013737  007436  000176      MOV     @20$,SWREG      ;SET NEW SW REG
007270  000454                      BR      6$              ;RETURN TO TEST
007272  021627  000012      7$:     CMP     (SP),#LF        ;CHAR = LF?
007276  001011                      BNE     8$              ;BRANCH IF NOT
007300  104401  001161              TYPE    ,$CRLF          ;ECHO CR-LF
007304  005737  007440              TST     @30$            ;CHECK INPUT FLAG
```

```
007310  001477                          BEQ     TSEL            ;LEAVE SW SETTINGS ALONE IF NO INPUT
007312  013737  007436  000176          MOV     20$,SWREG       ;SET NEW SW REG
007320  000473                          BR      TSEL            ;GO TO TEST SELECT (VIA KEYBOARD)
007322  042716  177770          8$:     BIC     #^C7,(SP)       ;MASK DIGIT
007326  062716  000060                  ADD     #60,(SP)        ;MAKE ASCII
007332  004737  010344                  JSR     PC,$TYPEC       ;PRINT DIGIT
007336  042716  177770                  BIC     #^C7,(SP)       ;MASK DIGIT
007342  006337  007436                  ASL     20$             ;SHIFT SWITCH SETTINGS FOR NEW ONE
007346  006337  007436                  ASL     20$
007352  006337  007436                  ASL     20$
007356  062637  007436                  ADD     (SP)+,20$       ;ADD NEW SWITCH
007362  005237  007440                  INC     30$             ;SET INPUT FLAG
007366  000716                          BR      9$              ;CONTINUE
007370  022716  000177          4$:     CMP     #177,(SP)       ;CHAR = RUBOUT?
007374  001001                          BNE     3$              ;NO, CHECK AGAIN
007376  000444                          BR      TSEL            ;YES, GET TEST SELECTION (VIA KEYBOARD)
007400  022716  000003          3$:     CMP     #3,(SP)         ;CHAR = CNTL C ?
007404  001416                          BEQ     KYBDST          ;YES, GET # COLUMNS
007406  005716                          TST     (SP)            ;IS CODE = NUL, IE., CTL-SPACE?
007410  001004                          BNE     6$              ;BRANCH IF NOT
007412  022737  000064  001146          CMP     #64,$TSTNM      ;IS THIS TEST 64 ?
007420  001402                          BEQ     11$             ;BRANCH IF YES
007422  005726                  6$:     TST     (SP)+           ;RESET STACK
007424  000207                  1$:     RTS     PC              ;NO, RETURN

007426  012706  001100          11$:    MOV     #STACK,SP       ;FIX STACK
007432  000137  006210                  JMP     ACCEPT          ;TO WITHIN TEST 64

007436  000000          20$:    .WORD   0               ;SW SETTING INPUT
007440  000000          30$:    .WORD   0               ;INPUT FLAG
```

```
                                ;//////////////////////////////////////////////////////////////////////////////
                                ;ROUTINE TO SET NUMBER OF COLUMNS FROM CONSOLE DEVICE KYBD
                                ;//////////////////////////////////////////////////////////////////////////////
007442  004737  006614          KYBDST: JSR     PC,SRDCI        ;CLEAR CLOCK INTER
007446  104401  014751                  TYPE    ,COLUMN         ;TYPE COLUMNS MESSAGE
007452  104410                  2$:     RDDEC                   ;GET # COLUMNS
007454  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
007456  020527  000204                  CMP     R5,#132.        ;TEST SIZE OF NUMBER
007462  003003                          BGT     1$              ;TOO BIG, GET AGAIN
007464  020527  000002                  CMP     R5,#2           ;TEST SIZE AGAIN
007470  103005                          BHIS    3$              ;BRANCH IF OK
007472  104401  001160          1$:     TYPE    ,$QUES          ;INPUT ERROR - TYPE QUESTION MARK
007476  104401  001161                  TYPE    ,$CRLF
007502  000763                          BR      2$              ;GET INPUT AGAIN
007504  010537  001150          3$:     MOV     R5,WIDTH        ;OK, SAVE # COLUMNS
                                ;//////////////////////////////////////////////////////////////////////////////
                                ;ROUTINE TO FETCH TEST # FROM CONSOLE  KEYBOARD
                                ;AND DETERMINE TEST ACTION .
                                ;TEST NUMBER MUST BE OCTAL, FOLLOWED BY ONE OF THE
                                ;FOLLOWING CONTROL CHARACTERS:
                                ;
                                ;       PERIOD  .       =       RUN TEST ONCE AND SELECT NEXT TEST
                                ;
                                ;       L       =       LOOP ON SELECTED TEST
                                ;
                                ;       S       =       START TEST SEQUENCE WITH SELECTED TEST
                                ;//////////////////////////////////////////////////////////////////////////////
007510  004737  006614          TSEL:   JSR     PC,SRDCI        ;CLEAR CLOCK INTER
007514  105037  001153                  CLRB    TRONE           ;CLEAR PROGRAM CONTROL FLAGS
007520  105037  001154                  CLRB    TLOOP
007524  105037  001152                  CLRB    STRONE
007530  104401  014767                  TYPE    ,SELTST         ;TYPE SELECT TEST MESG
007534  104407                  5$:     RDLIN                   ;GET TEST SELECTION
007536  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
007540  112500                          MOVB    (R5)+,R0        ;SET TEST # IN R0
007542  042700  177600                  BIC     #^C177,R0
007546  022700  000003                  CMP     #3,R0           ;CHECK IF CHAR = CNTL-C
007552  001733                          BEQ     KYBDST          ;GET # COLUMNS IF CNTL-C
007554  020027  000060                  CMP     R0,#60          ;CHECK IF OCTAL NUMBER
007560  103461                          BLO     4$              ;NOT OCTAL - INPUT ERRO
007562  020027  000067                  CMP     R0,#67
007566  101061                          BHI     4$              ;NOT OCTAL - INPUT ERROR
007570  042700  177770                  BIC     #^C7,R0         ;OK - STORE DIGIT
007574  006300                          ASL     R0
007576  006300                          ASL     R0
007600  006300                          ASL     R0
007602  112501                          MOVB    (R5)+,R1        ;GET SECOND DIGIT
007604  042701  177600                  BIC     #^C177,R1       ;CHECK IF AN OCTAL DIGIT
007610  020127  000060                  CMP     R1,#60
```

```
007614  103446                          BLO     4$              ;NOT OCTAL - INPUT ERROR
007616  020127  000067                  CMP     R1,#67
007622  101043                          BHI     4$              ;NOT OCTAL - INPUT ERROR
007624  042701  177770                  BIC     #^C7,R1         ;OK - MASK DIGIT
007630  060100                          ADD     R1,R0           ;MAKE COMPLETE TEST NUMBER
007632  010037  001146                  MOV     R0,$TSTNM       ;SAVE TEST NUMBER
007636  006300                          ASL     R0              ;SET TABLE POINTER
007640  005760  013356                  TST     TAT(R0)         ;CHECK IF TEST IS IN TABLE
007644  003432                          BLE     4$              ;NOT IN TABLE, GET NEW SELECTION
007646  112501                          MOVB    (R5)+,R1        ;GET CONTROL CHAR
007650  042701  177600                  BIC     #^C177,R1       ;MASK BITS
007654  020127  000056                  CMP     R1,#56          ;CHAR = PERIOD?
007660  001004                          BNE     1$              ;NO, CONTINUE CHECK
007662  112737  177777  001153          MOVB    #-1,TRONE       ;YES, SET FLAG
007670  000414                          BR      3$              ;CONTINUE
007672  042701  000040          1$:     BIC     #BIT5,R1                ;MASK BIT 5 (ALLOW UPPER OR LOWER CASE)
007676  022701  000114                  CMP     #'L,R1          ;CHAR = L?
007702  001004                          BNE     2$              ;NO, CONTINUE
007704  112737  177777  001154          MOVB    #-1,TLOOP       ;YES, SET FLAG
007712  000403                          BR      3$              ;CONTINUE
007714  022701  000123          2$:     CMP     #'S,R1          ;CHAR = S?
007720  001004                          BNE     4$              ;NO, INPUT ERROR
007722  012706  001100          3$:     MOV     #STACK,SP       ;RESET STACK
007726  000170  013356                  JMP     @TAT(R0)        ;ALL OK - GO TO SELECTED TEST
007732  104401  001160          4$:     TYPE    ,$QUES          ;INPUT ERROR
007736  104401  001161                  TYPE    ,$CP_F          ;PRINT QUESTION MARK
007742  000674                          BR      5$              ;GET NEW INPUT
```

```
                                        ;;*****************************************************************
                                        .SBTTL  ERROR HANDLER ROUTINE

                                        ;THIS ROUTINE WILL SAVE THE ERROR NUMBER AND THE ADR OF THE ERROR CALL
                                        ;AND GO TO REPORT ON ERROR
                                        ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;       SW15 = 1        HALT ON ERROR
                                        ;       SW13 = 1        INHIBIT ERROR TYPEOUTS
                                        ;       SW10 = 1        BELL ON ERROR
                                        ;
                                        ;CALL
                                        ;       ERROR   N               ;ERROR = EMT & N = ERROR ITEM NUMBER

007744  013777  001146  171172  $ERROR: MOV     $TSTNM,@DISPLAY;DISPLAY TEST NUMBER AND ERROR FLAG
007752  032777  002000  171162          BIT     #BIT10,@SWR     ;BELL ON ERROR?
007760  001402                          BEQ     1$              ;NO - SKIP
007762  104401  001156                  TYPE    ,$BELL          ;RING BELL
007766  011637  01004C          1$:     MOV     (SP),$ERRPC     ;GET ADDRESS OF ERROR INSTRUCTION
007772  162737  000002  010040          SUB     #2,$ERRPC
010000  117737  000034  010042          MOVB    @$ERRPC,$ITEMB  ;STRIP AND SAVE THE ERROR ITEM CODE
010006  032777  020000  171126          BIT     #BIT13,@SWR     ;SKIP TYPEOUTS IF SET
010014  001004                          BNE     2$              ;SKIP TYPEOUTS
010016  004737  010044                  JSR     PC,REPORT       ;GO TO REPORT ROUTINE
010022  104401  001161                  TYPE    ,$CRLF
010026  005777  171110          2$:     TST     @SWR            ;HALT ON ERROR IF SET
010032  100001                          BPL     3$              ;SKIP IS CONTINUE
010034  104421                          HOLD                    ;DYNAMIC HALT
010036  000002          3$:     RTI                             ;RETURN

010040  000000  $ERRPC: .WORD   0       ;LAST ERROR INSTRUCTION EXECUTED
010042  000000  $ITEMB: .WORD   0       ;ITEM CODE
```

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

                        ;ERROR REPORT ROUTINE
                        ;
                        ;ERROR MESSAGE WILL USE THE FOLLOWING FORM:
                        ;TEST #XX,  PC=XXXXXX,  ERROR #XXX,  MESSAGE >>>>>>>>>>>>>

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

010044  105737  001173        REPORT: TSTB    $TPFLG          ;TERMINAL EXIT?
010050  100435                        BMI     2$              ;NO, EXIT
010052  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
010054  104401  015044                TYPE    ,ETSTNO         ;TYPE FIRST PART OF ERROR MESG
010060  013746  001146                MOV     $TSTNM,-(SP)    ;;PUSH $TSTNM ON STACK
010064  104403                        TYPOS                   ;TYPE TEST NUMBER
010066     002                        .BYTE   2               ;TWO DIGITS MAX.
010067     001                        .BYTE   1               ;TYPE LEADING ZEROS
010070  104401  015054                TYPE    ,PCMSG          ;TYPE PART OF ERROR MESG
010074  013746  010040                MOV     $ERRPC,-(SP)    ;;PUSH $ERRPC ON STACK
010100  104402                        TYPOC                   ;TYPE ERROR PC
010102  104401  015063                TYPE    ,ERR            ;TYPE MORE OF ERROR MESG
010106  013746  010042                MOV     $ITEMB,-(SP)    ;;PUSH $ITEMB ON STACK
010112  104403                        TYPOS                   ;TYPE ERROR NUMBER
010114     003                        .BYTE   3               ;3 DIGITS MAX.
010115     001                        .BYTE   1               ;TYPE LEADING ZEROS
010116  104401  015076                TYPE    ,ERRS           ;TYP SPACES
010122  013700  010042                MOV     $ITEMB,R0       ;GET ERROR NUMBER
010126  006300                        ASL     R0              ;SET TABLE POINTER
010130  016037  013554  010140        MOV     EMAT-2(R0),1$   ;SET ERROR MESSAGE ADR
010136  104401                        TYPE                    ;TYPE ERROR MESG
010140  000000              1$:       .WORD   0
010142  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
010144  000207              2$:       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

010146                          $SAVREG:
010146  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
010150  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
010152  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
010154  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
010156  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
010160  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
010162  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
010166  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
010172  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
010176  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
010202  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
010204                          $RESREG:
010204  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
010210  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
010214  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
010220  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
010224  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
010226  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
010230  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
010232  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
010234  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
010236  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
010240  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

010242  105737  001173          $TYPE:  TSTB    $TPFLG          ;IS THERE A CONSOLE TERMINAL?
010246  100407                          BMI     3$              ;BRANCH IF NO CONSOLE TERMINAL
010250  010046                  1$:     MOV     R0,-(SP)        ;SAVE R0
010252  017600  000002                  MOV     @2(SP),R0       ;GET ADR OF ASCIZ STRING
010256  112046                  2$:     MOVB    (R0)+,-(SP)     ;PUSH CHARACTER TO BE TYPED ONTO STACK
010260  001005                          BNE     4$              ;BR IF IT ISN'T THE TERMINATOR
010262  005726                          TST     (SP)+           ;IF TERMINATOR, POP IT OFF STACK
010264  012600                          MOV     (SP)+,R0        ;RESTORE R0
010266  062716  000002          3$:     ADD     #2,(SP)         ;ADJUST RETURN PC
010272  000002                          RTI                     ;RETURN
010274  122716  000200          4$:     CMPB    #CRLF,(SP)      ;BRANCH IF NOT <CRLF>
010300  001004                          BNE     5$
010302  005726                          TST     (SP)+           ;POP <CR><LF> EQUIV
010304  104401  001161                  TYPE,   $CRLF           ;TYPE CR-LF
010310  000762                          BR      2$              ;GET NEXT CHAR
010312  004737  010344          5$:     JSR     PC,$TYPEC       ;GO TYPE CHAR
010316  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;IS IT TIME FOR FILLER CHARS?
010322  001355                          BNE     2$              ;IF NO GO GET NEXT CHAR
010324  013746  001170                  MOV     $NULL,-(SP)     ;GET # FILLER CHARS NEEDED
                                                                ;AND THE NULL CHAR
010330  105366  000001          7$:     DECB    1(SP)           ;DOES A NULL NEED TO BE TYPED?
010334  002770                          BLT     6$              ;BR IF NO - GO POP THE NULL OFF OF STACK
010336  004737  010344                  JSR     PC,$TYPEC       ;GO TYPE NULL
010342  000772                          BR      7$              ;LOOP

010344  105777  170534          $TYPEC: TSTB    @$TPS           ;WAIT UNTIL CONSOLE PRINTER IS READY
010350  100375                          BPL     $TYPEC
010352  116677  000002  170526        MOVB    2(SP),@$TPB     ;LOAD CHAR TO BE TYPED INTO DATA REG.
010360  000207                          RTS     PC              ;RETURN
```

```
                                CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                .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

 010362                         $TYPDS:
 010362  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 010364  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 010366  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 010370  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 010372  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 010374  012746  020200                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 010400  016605  000020                 MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 010404  100004                         BPL     1$              ;;BR IF INPUT IS POS.
 010406  005405                         NEG     R5              ;;MAKE THE BINARY NUMBER POS.
 010410  112766  000055  000001         MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
 010416  005000                 1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
 010420  012703  010576                 MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
 010424  112723  000040                 MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
 010430  005002                 2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
 010432  016001  010566                 MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
 010436  160105                 3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
 010440  002402                         BLT     4$              ;;BR IF DONE
 010442  005202                         INC     R2              ;;INCREASE THE BCD DIGIT BY 1
 010444  000774                         BR      3$
 010446  060105                 4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
 010450  005702                         TST     R2              ;;CHECK IF BCD DIGIT=0
 010452  001002                         BNE     5$              ;;FALL THROUGH IF 0
 010454  105716                         TSTB    (SP)            ;;STILL DOING LEADING 0'S?
 010456  100407                         BMI     7$              ;;BR IF YES
 010460  106316                 5$:     ASLB    (SP)            ;;MSD?
 010462  103003                         BCC     6$              ;;BR IF NO
 010464  116663  000001  177777         MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 010472  052702  000060         6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 010476  052702  000040         7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 010502  110223                         MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 010504  005720                         TST     (R0)+           ;;JUST INCREMENTING
 010506  020027  000010                 CMP     R0,#10          ;;CHECK THE TABLE INDEX
 010512  002746                         BLT     2$              ;;GO DO THE NEXT DIGIT
 010514  003002                         BGT     8$              ;;GO TO EXIT
 010516  010502                         MOV     R5,R2           ;;GET THE LSD
 010520  000764                         BR      6$              ;;GO CHANGE TO ASCII
 010522  105726                 8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 010524  100003                         BPL     9$              ;;BR IF NO
 010526  116663  177777  177776         MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 010534  105013                 9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 010536  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
 010540  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
 010542  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
010544  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
010546  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
010550  104401  010576                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
010554  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
010562  012616                          MOV     (SP)+,(SP)
010564  000002                          RTI                     ;;RETURN TO USER
010566  023420                  $DTBL:  10000.
010570  001750                          1000.
010572  000144                          100.
010574  000012                          10.
010576  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.
                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                        ;*      .BYTE   N               ;;N=1 TO 5 FOR NUMBER OF DIGITS TO TYPE
                        ;*      .BYTE   M               ;;M=1 OR 0
                        ;*                                      ;;1=TYPE LEADING ZEROS
                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                        ;*
                        ;*$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

010606  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
010612  116637  000001  011031          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
010620  112537  011033                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
010624  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
010630  000406                          BR      $TYPON
010632  112737  000001  011031  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
010640  112737  000006  011033          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
010646  112737  000005  011030  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
010654  010346                          MOV     R3,-(SP)        ;;SAVE R3
010656  010446                          MOV     R4,-(SP)        ;;SAVE R4
010660  010546                          MOV     R5,-(SP)        ;;SAVE R5
010662  113704  011033                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
010666  005404                          NEG     R4
010670  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
010674  110437  011032                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
010700  113704  011031                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
010704  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
010710  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
010712  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
010714  000404                          BR      3$              ;;GO DO MSB
010716  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
010720  006105                          ROL     R5
010722  006105                          ROL     R5
010724  010503                          MOV     R5,R3
010726  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
010730  105337  011032                  DECB    $OMODE          ;;TYPE THIS DIGIT?
010734  100016                          BPL     7$              ;;BR IF NO
010736  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
010742  001002                          BNE     4$              ;;TEST FOR 0
010744  005704                          TST     R4              ;;SUPPRESS THIS 0?
010746  001403                          BEQ     5$              ;;BR IF YES
```

```
011750  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
011752  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
011756  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
011762  110337  011026                  MOVB    R3,8$           ;;SAVE FOR TYPING
011766  104401  011026                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
011772  105337  011030          7$:     DECB    $OCNT           ;;COUNT BY !
011776  003347                          BGT     2$              ;;BR IF MORE TO DO
011000  002402                          BLT     6$              ;;BR IF DONE
011002  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
011004  000744                          BR      2$              ;;GO DO THE LAST DIGIT
011006  012605          6$:             MOV     (SP)+,R5        ;;RESTORE R5
011010  012604                          MOV     (SP)+,R4        ;;RESTORE R4
011012  012603                          MOV     (SP)+,R3        ;;RESTORE R3
011014  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
011022  012616                          MOV     (SP)+,(SP)
011024  000002                          RTI                     ;;RETURN
011026     000          8$:             .BYTE   0               ;;STORAGE FOR ASCII DIGIT
011027     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
011030     000          $OCNT:          .BYTE   0               ;;OCTAL DIGIT COUNTER
011031     000          $OFILL:         .BYTE   0               ;;ZERO FILL SWITCH
011032  000000          $OMODE:         .WORD   0               ;;NUMBER OF DIGITS TO TYPE


011034  004737  011316  $LOAD:  JSR     PC,RSTADR       ;RESTORE ADDRESSING TO FIRST PRINTER
011040  000137  000000  LDPTR:  JMP     0               ;SWITCH TO EITHER PARALLEL OR
                                                        ;SERIAL SINGLE-CHARACTER LOADER.
                                                        ;THIS SWITCH IS SET UP BY THE PROGRAM
                                                        ;AT STARTUP PER THE CONTENTS OF SERSW WORD.
```

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

                                .SBTTL  ROUTINES TO LOAD CHARACTERS TO LA180

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

                                ;ROUTINE TO LOAD SINGLE CHARACTERS TO LA180 PRINTER
                                ;USING THE STANDARD PARALLEL INTERFACE
                                ;CALL:  MOV     CHAR,R0         ;PUT CHAR IN R0
                                ;       LOAD
                                ;
                                ;IF PRINTER DOES NOT GO READY WITHIN 9 TO 32 SECONDS
                                ;(DEPENDING ON CPU AND MEMORY TYPE) THE DIAGNOSTIC WILL
                                ;INDICATE AN ERROR - ERROR #24 - NOT READY.

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

011044  104420          PLOAD:  CHECK                   ;CHECK FOR CONTROL
011046  012737  000040  011140          MOV     #40,6$  ;SET DELAY LOOP COUNT
011054  005777  170034          1$:     TST     @LPS    ;CHECK FOR ERROR CONDX
011060  100002                  BPL     1$              ;BRANCH IF OK
011062  104024                  ERROR   24              ;PRINTER ERROR BEFORE LOAD
011064  000423                  BR      4$              ;EXIT
011066  005037  011136          2$:     CLR     5$      ;CLEAR DELAY COUNT
011072  005237  011136          3$:     INC     5$      ;INC COUNT
011076  001005                  BNE     3$              ;COUNTINUE
011100  005337  011140          6$:     DEC     6$      ;DEC DELAY LOOP COUNT
011104  001370                  BNE     1$              ;WAIT IF NOT READY
011106  104025                  ERROR   25              ;ERROR, PRINTER NOT READY
011110  000411                  BR      4$              ;EXIT
011112  105777  167776          3$:     TSTB    @LPS    ;CHECK FOR PRINTER READY
011116  100365                  BPL     2$              ;WAIT FOR READY
011120  110077  167772                  MOVB    R0,@LPB ;LOAD CHAR
011124  005777  167764                  TST     @LPS    ;TEST FOR ERROR CONDX AGAIN
011130  100001                  BPL     4$              ;BRANCH IF OK
011132  104026                  ERROR   26              ;PRINTER ERROR AFTER LOAD
011134  000402          4$:     BR      NXTADR          ;DOES ANOTHER PRINTER REQUIRE LOADING ALSO?

011136  000000          5$:     .WORD   0               ;DELAY COUNTER
011140  000040          6$:     .WORD   32.             ;DELAY LOOP COUNTER


011142  005337  001132  NXTADR: DEC     LPCTR           ;DECREMENT LINE PRINTER COUNTER
011146  003416                  BLE     10$             ;BR IF COUNT IS ZERO OR LESS (TO EXIT)
011150  062737  000010  001110          ADD     #10,LPKS ;ADDRESS NEXT CONTIGUOUS LINE PRINTER
011156  062737  000010  001112          ADD     #10,LPKB
011164  062737  000010  001114          ADD     #10,LPS
011172  062737  000010  001116          ADD     #10,LPB
011200  000177  177636                  JMP     @LDPTR+2        ;GO LOAD SAME CHARACTER TO NEXT PRINTER
                                                        ;USING SAME LOADER.
011204  000002          10$:    RTI                     ;RETURN, SINGLE CHARACTER LOAD IS
                                                        ;COMPLETE (TO ALL PRINTERS).
```

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

                                ;ROUTINE TO LOAD SINGLE CHARACTERS TO THE LA180 PRINTER
                                ;USING THE SERIAL INTERFACE.
                                ;CALL:   MOV     CHAR,R0         ;PUT CHAR IN R0
                                ;        LOAD
                                ;

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

011206  105777  167676  SLOAD:  TSTB    @LPKS           ;CHECK FOR PRINT KBD INPUT
011212  100031                  BPL     2$              ;BRANCH IF NONE
011214  017737  167672  011314  MOV     @LPKB,10$       ;READ PRINTER KEYBOARD CHARACTER
011222  042737  177600  011314  BIC     #177600,10$     ;STRIP EXCESS BITS
011230  023727  011314  000023  CMP     10$,#23         ;IS CHARACTER XOFF?
011236  001017                  BNE     2$              ;NO, SO BRANCH
011240  104420          3$:     CHECK                   ;YES, CHECK FOR OPERATOR INTERVENTION
011242  105777  167642          TSTB    @LPKS           ;CKECK FOR MORE PRINTER KBD INPUT
011246  100374                  BPL     3$              ;NONE, WAIT FOR EXPECTED XON
011250  017737  167636  011314  MOV     @LPKB,10$       ;READ PRINTER KEYBOARD CHARACTER
011256  042737  177600  011314  BIC     #177600,10$     ;STRIP EXCESS BITS
011264  023727  011314  000021  CMP     10$,#21         ;IS CHARACTER XON?
011272  001362                  BNE     3$              ;NO, WAIT SOME MORE.
011274  000404                  BR      4$              ;YES, GO DIRECTLY TO LOAD A CHARACTER
011276  104420          2$:     CHECK                   ;CHECK FOR OPERATOR INTERVENTION
011300  105777  167610          TSTB    @LPS            ;IS PRINTER READY?
011304  100340                  BPL     SLOAD           ;NO, START CHECKS OVER
011306  110077  167604  4$:     MOVB    R0,@LPB         ;YES, LOAD CHARACTER
011312  000713                  BR      NXTADR          ;GO LOAD SAME CHARACTER TO NEXT PRINTER
                                                        ;USING SAME LOADER.

011314  000000          10$:    .WORD   0               ;LA180 KEYBOARD CHARACTER STORAGE.


011316  013737  001130  001110  RSTADR: MOV     ACTFST,LPKS     ;SET PRINTER KBD STATUS REG ADDRESS
011324  013737  001110  001112          MOV     LPKS,LPKB
011332  062737  000002  001112          ADD     #2,LPKB         ;SET PRINTER KBD BUFFER ADDRESS
011340  013737  001112  001114          MOV     LPKB,LPS
011346  062737  000002  001114          ADD     #2,LPS          ;SET PRINTER STATUS REG ADDRESS
011354  013737  001114  001116          MOV     LPS,LPB
011362  062737  000002  001116          ADD     #2,LPB          ;SET PRINTER BUFFER ADDRESS
011370  013737  001122  001132          MOV     NUMLP,LPCTR     ;PUT # OF LP'S INTO A COUNTER.
011376  000207                  RTS     PC              ;RETURN
```

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

                        ;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

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

011400  104415          $MLOAD: LOAD                    ;LOAD CHAR
011402  005301                  DEC     R1              ;DEC COUNT
011404  003375                  BGT     $MLOAD          ;FINISH LOAD
011406  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

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

011410                  $CNLD:
011410  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011412  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
011414  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
011416  005002                  CLR     R2
011420  016601  000012          MOV     12(SP),R1       ;GET ADR
011424  112100          1$:     MOVB    (R1)+,R0        ;GET CHAR
011426  001410                  BEQ     3$              ;RETURN WHEN DONE
011430  005702                  TST     R2              ;DONE LEADING ZEROS?
011432  001004                  BNE     2$              ;YES, LOAD CHAR
011434  020027  000060          CMP     R0,#60          ;NO, CHECK THIS CHAR
011440  001771                  BEQ     1$              ;SKIP IF ZERO
011442  005202                  INC     R2              ;SET FLAG IF NON-ZERO
011444  104415          2$:     LOAD                    ;LOAD CHAR
011446  000766                  BR      1$              ;CONTINUE
011450  005702          3$:     TST     R2              ;ALL CHARS ZERO?
011452  001003                  BNE     4$              ;NO, EXIT
011454  012700  000060          MOV     #60,R0          ;YES, LOAD ZERO
011460  104415                  LOAD
011462                  4$:
011462  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
011464  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
011466  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
011470  016666  000002  000004  MOV     2(SP),4(SP)     ;ADJUST STACK
011476  012616                  MOV     (SP)+,(SP)
011500  000002                  RTI                     ;RETURN
```

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

                        ;ROUTINE TO PRINT ASCIZ STRINGS ON LA180 PRINTER
                        ;STRING MUST BE TERMINATED BY NULL BYTE
                        ;CALL:          PRINT
                        ;               MESADR

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

011502                  $PRINT:
011502  010046                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011504  010146                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
011506  017601  000004         MOV     @4(SP),R1       ;GET ADDRESS OF ASCIZ STRING
011512  112100          1$:    MOVB    (R1)+,R0        ;GET CHAR
011514  001407                 BEQ     2$              ;BRANCH IF TERMINATOR (NULL)
011516  120027  000200         CMPB    R0,#CRLF        ;CHECK IF CRLF CODE
011522  001002                 BNE     3$              ;BRANCH IF NOT
011524  012700  000012         MOV     #LF,R0          ;YES, DO LF ONLY
011530  104415          3$:    LOAD                    ;LOAD CHAR
011532  000767                 BR      1$              ;GET NEXT CHAR
011534                  2$:
011534  012601                 MOV     (SP)+,R1        ;;POP STACK INTO R1
011536  012600                 MOV     (SP)+,R0        ;;POP STACK INTO R0
011540  062716  000002         ADD     #2,(SP)         ;ADJUST RETURN PC
011544  000002                 RTI                     ;RETURN
```

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

                        .SBTTL  PRINT TEST HEADER

                        ;THE NUMBER OF COLUMNS WILL ALSO BE PRINTED FOR TEST 25 ONLY
                        ;CALL:  PRTHDR

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

011546                  $PRHDR:
011546  005045                 CLR     -(SP)           ;;PUT NEW PS ON STACK
011550  012746  011556         MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
011554  000002                 RTI                     ;;POP NEW PC AND PS
011556                  64$:
011556  012700  000177         MOV     #177,R0         ;SET RUBOUT CHAR
011562  104415                 LOAD                    ;CLEAR LA180 CHAR BUFFER
011564  023737  001146  011660 CMP     $TSTNM,SVTST    ;CHECK IF PRINTED THIS # LAST
011572  001427                 BEQ     3$              ;YES,  PRINT BLANK LINE & EXIT
011574  013737  001146  011660 MOV     $TSTNM,SVTST    ;NO, STORE NEW #
011602  104414  015011         PRINT   .TSTNO          ;LOAD TEST # MESG
011606  013746  001146         MOV     $TSTNM,-(SP)    ;;PUSH $TSTNM ON STACK
011612  004737  013024         JSR     PC,$SB2O        ;CONVERT #
011616  104417                 CNLOAD                  ;LOAD NUMBER
011620  104414  001162         PRINT   .$LF            ;PRINT LINE
011624  023727  001146  000025 CMP     $TSTNM,#25      ;IS THIS TEST 25?
011632  001007                 BNE     3$              ;NO, SKIP COLUMN MESG
011634  013746  001150         MOV     WIDTH,-(SP)     ;PUT # COLUMNS ON STACK
011640  004737  012650         JSR     PC,$SB2D        ;CONVERT #
011644  104417                 CNLOAD                  ;LOAD NUMBER
011646  104414  015031         PRINT   .COLMN          ;LOAD CLOUMN MESG
011652  104414  001162 3$:     PRINT   .$LF            ;BLANK LINE
011656  000002         4$:     RTI                     ;RETURN

011660  000000         SVTST:  .WORD   0               ;SAVE TEST NUMBER
```

```
                                .SBTTL  TTY INPUT ROUTINE

                                ;;*****************************************************************
                                .ENABL  LSB

                                .DSABL  LSB


                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;

011662  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
011664  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
011672  105777  167202          1$:     TSTB    @$TKS           ;;WAIT FOR
011676  100375                          BPL     1$              ;;A CHARACTER
011700  117766  167176  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
011706  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
011714  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
011722  001013                          BNE     3$              ;;BRANCH IF NO
011724  105777  167150          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
011730  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
011732  117746  167144                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
011736  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
011742  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
011746  001366                          BNE     2$              ;;IF NOT DISCARD IT
011750  000750                          BR      1$              ;;YES, RESUME
011752  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
011760  002407                          BLT     4$              ;;BRANCH IF YES
011762  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
011770  003003                          BGT     4$              ;;BRANCH IF YES
011772  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
012000  000002                  4$:     RTI                     ;;GO BACK TO USER
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

012002  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
012004  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
012006  012703  012236          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
012012  022703  012242          2$:     CMP     #$TTYIN+4,R3    ;;BUFFER FULL?
012016  101456                          BLOS    4$              ;;BR IF YES
012020  104406                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
012022  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
012024  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
012030  001022                          BNE     5$              ;;BR IF NO
012032  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
012034  001007                          BNE     6$              ;;BR IF NO
012036  112737  000134  012234          MOVB    #'\,.9$         ;;TYPE A BACK SLASH
```

```
012044  104401  012234                  TYPE    9$
012050  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
012054  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
012056  020327  012236                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
012062  103434                  BLO     4$              ;;BR IF YES
012064  111337  012234                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
012070  104401  012234                  TYPE    9$              ;;GO TYPE
012074  000746                  BR      2$              ;;GO READ ANOTHER CHAR.
012076  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
012100  001406                  BEQ     7$              ;;BR IF NO
012102  112737  000134  012234          MOVB    #'\,9$          ;;TYPE A BACK SLASH
012110  104401  012234                  TYPE    9$
012114  005016                  CLR     (SP)            ;;CLEAR THE RUBOUT KEY
012116  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
012122  001003                  BNE     8$              ;;BR IF NO
012124  104401  012242                  TYPE    $CNTLU          ;;TYPE A CONTROL "U"
012130  000726                  BR      1$              ;;GO START OVER
012132  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
012136  001011                  BNE     3$              ;;BRANCH IF NO
012140  105013                  CLRB    (R3)            ;;CLEAR THE CHARACTER
012142  104401  001161                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
012146  104401  012236                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
012152  000717                  BR      2$              ;;GO PICKUP ANOTHER CHACTER
012154  104401  001160          4$:     TYPE    $QUES           ;;TYPE A '?'
012160  000712                  BR      1$              ;;CLEAR THE BUFFER AND LOOP
012162  111337  012234          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
012166  104401  012234                  TYPE    ,9$
012172  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
012176  001305                  BNE     2$              ;;LOOP IF NOT RETURN
012200  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
012204  104401  001162                  TYPE    ,$LF            ;;TYPE A LINE FEED
012210  005726                  TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
012212  012603                  MOV     (SP)+,R3        ;;RESTORE R3
012214  011646                  MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
012216  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
012224  012766  012236  000004          MOV     #$TTYIN,4(SP)
012232  000002                  RTI                             ;;RETURN
012234     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
012235     000                  .BYTE   0                       ;;TERMINATOR
012236  000004          $TTYIN: .BLKB   4                       ;;RESERVE 4 BYTES FOR TTY INPUT
012242  052536  005015     000  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
012247     136  006507  000012  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
012254  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
012262  020075     000
012265     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
012272  036440  000040
```

```
                                .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
                                ;

012276  011646                  $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
012300  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
012306  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
012310  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
012312  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
012314  104407                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
012316  012600                          MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
012320  010037  012444                  MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
012324  005046                          CLR     -(SP)           ;;CLEAR DATA WORD
012326  005002                          CLR     R2              ;;SIGN SET POSITIVE
012330  122710  000055                  CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
012334  001001                          BNE     2$              ;;BR IF NO MINUS SIGN
012336  112002                          MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
012340  112001                  2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
012342  001424                          BEQ     3$              ;;GET OUT IF ZERO
012344  122701  000060                  CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
012350  003032                          BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
012352  122701  000071                  CMPB    #'9,R1
012356  002427                          BLT     5$
012360  032716  170000                  BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
012364  001024                          BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
012366  006316                          ASL     (SP)            ;;*2
012370  011646                          MOV     (SP),-(SP)      ;;SAVE FOR LATER
012372  006316                          ASL     (SP)            ;;*4
012374  006316                          ASL     (SP)            ;;*8.
012376  062616                          ADD     (SP)+,(SP)      ;;*10.
012400  102416                          BVS     5$              ;;OVERFLOW ISN'T ALLOWED
012402  162701  000060                  SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
012406  060116                          ADD     R1,(SP)         ;;ADD IN THIS DIGIT
012410  102412                          BVS     5$              ;;OVERFLOW ISN'T ALLOWED
012412  000752                          BR      2$              ;;LOOP
012414  005702                  3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
012416  001401                          BEQ     4$              ;;BR IF NO
012420  005416                          NEG     (SP)            ;;YES--NEGATE THE NUMBER
012422  012666  000012          4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
012426  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
012430  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
012432  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
012434  000002                          RTI                     ;;RETURN

012436  005726                  5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
012440  105010                          CLRB    (R0)            ;;SET A TERMINATOR
```

```
012442  104401                  TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
012444  000000          6$:     .WORD   0               ;;POINTER GOES HERE
012446  104401  001160          TYPE    $QUES           ;;"?" "CR" &"LF"
012452  000720                  BR      1$              ;;TRY AGAIN
```

```
                                DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                                .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


012454  104411                  $DB2D:  SAVREG                  ;;SAVE REGISTERS
012456  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
012462  012700  012634                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
012466  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
012472  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
012474  012202                          MOV     (R2)+,R2
012476  012737  000012  012552          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
012504  012704  012564                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
012510  012705  012566                  MOV     #$TNPWR+2,R5
012514  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
012516  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
012520  005602                          SBC     R2
012522  161502                          SUB     (R5),R2
012524  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
012526  005203                          INC     R3              ;;ADD 1 TO PARTIAL
012530  000772                          BR      2$              ;;LOOP
012532  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
012534  005502                          ADC     R2
012536  062402                          ADD     (R4)+,R2
012540  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
012542  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
012546  110320                          MOVB    R3,(R0)+        ;;SAVE IT
012550  005327                          DEC     (PC)+           ;;DONE?
012552  000000                  4$:     .WORD   0
012554  001357                          BNE     1$              ;;BR IF NO
012556  105020                          CLRB    (R0)+           ;;TERMINATOR
012560  104412                          RESREG                  ;;RESTORE REGISTERS
012562  000207                          RTS     PC              ;;RETURN
012564  145000                  $TNPWR: 145000                  ;;1.0E09
012566  035632                          35632
012570  160400                          160400                  ;;1.0E08
012572  002765                          2765
012574  113200                          113200                  ;;1.0E07
012576  000230                          230
012600  041100                          041100                  ;;1.0E06
012602  000017                          17
012604  103240                          103240                  ;;1.0E05
012606  000001                          1
012610  023420                          23420                   ;;1.0E04
012612  000000                          0
012614  001750                          1750                    ;;1.0E03
012616  000000                          0
012620  000144                          144                     ;;1.0E02
```

```
012622  000000                          0
012624  000012                          12                      ;;1.0E01
012626  000000                          0
012630  000001                          1                       ;;1.0E00
012632  000000                          0
012634  000014          $DECVL: .BLKB   12.                     ;;RESERVE STORAGE FOR ASCIZ STRING

                        .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

                        ;;**************************************************************
                        ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                        ;*UNSIGNED DECIMAL ASCIZ NUMBER.
                        ;*CALL
                        ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
                        ;*      JSR     PC,@#$SB2D      ;;CALL
                        ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK


012650  016637  000002  012700  $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
012656  012746  012700                  MOV     #1$,-(SP)       ;;SET POINTER
012662  004737  012454                  JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
012666  062716  000005                  ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
012672  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
012676  000207                          RTS     PC              ;;RETURN
012700  000000  000000          1$:     .WORD   0,0
```

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

                        ;;**********************************************************************
                        ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
                        ;*UNSIGNED OCTAL ASCIZ NUMBER.
                        ;*CALL
                        ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                        ;*      JSR     PC,@#$DB20      ;;CALL THE ROUTINE
                        ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK

012704  104411          $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
012706  016601  000002          MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
012712  012705  013023          MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
012716  012704  000014          MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
012722  012703  177770          MOV     #^C7,R3         ;;MASK
012726  012100                  MOV     (R1)+,R0        ;;LOWER WORD
012730  012101                  MOV     (R1)+,R1        ;;HIGH WORD
012732  005002                  CLR     R2              ;;TERMINATOR
012734  110245          1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
012736  010002                  MOV     R0,R2           ;;GET THIS DIGIT
012740  005304                  DEC     R4              ;;COUNT THIS CHARACTER
012742  003007                  BGT     3$              ;;BR IF NOT THE LAST DIGIT
012744  001405                  BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
012746  005205                  INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
012750  010566  000002          MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
012754  104412                  RESREG                  ;;RESTORE ALL REGISTERS
012756  000207                  RTS     PC              ;;RETURN TO USER
012760  006203          2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
012762  006001          3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
012764  006000                  ROR     R0              ;;      THE NEXT OCTAL DIGIT
012766  006001                  ROR     R1
012770  006000                  ROR     R0
012772  006001                  ROR     R1
012774  006000                  ROR     R0
012776  040302                  BIC     R3,R2           ;;MASK OUT ALL JUNK
013000  062702  000060          ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
013004  000753                  BR      1$              ;;GO PUT IT IN THE DATA TABLE
013006  000016          $OCTVL: .BLKB   14.             ;;RESERVE DATA TABLE
```

```
MAINDEC-11-DZLAE-B      MACY11 27(1006)  01-MAR-77  14:45  PAGE 84
DZLAEB.P11    28-FEB-77 15:38            SINGLE LENGTH BINARY TO OCTAL ASCIZ ROUTINE

                                         .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,@#$SB20      ;;CALL
                                         ;*      RETURN                  ;;ADDRESS OF 1ST ASCIZ CHAR. IS ON THE STACK

013024  016637  000002  013054  $SB20:  MOV     2(SP),1$        ;;SAVE THE BINARY NUMBER
013032  012746  013054                  MOV     #1$,-(SP)       ;;SET POINTER
013036  004737  012704                  JSR     PC,@#$DB20      ;;CALL DOUBLE LENGTH CONVERT ROUTINE
013042  062716  000005                  ADD     #5,(SP)         ;;ONLY ALLOW SIX CHARACTERS
013046  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
013052  000207                          RTS     PC              ;;RETURN
013054  000000  000000          1$:     .WORD   0,0
```

```
                                .SBTTL  TRAP DECODER

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

013050  016646  000002    $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
013054  042716  000020            BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
013060  012746  013066            MOV     #1$,-(SP)       ;; T-BIT TRAPS
013064  000002                    RTI                     ;;SET THE NEW STATUS
013066  010046            1$:     MOV     R0,-(SP)        ;;SAVE R0
013070  016600  000002            MOV     2(SP),R0        ;;GET TRAP ADDRESS
013074  005740                    TST     -(R0)           ;;BACKUP BY 2
013076  111000                    MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
013100  022700  000044            CMP     #$TERM,R0       ;;CHECK FOR OUT OF BOUNDS
013104  003002                    BGT     .+6             ;;BR IF OK
013106  000000                    HALT                    ;;OUT OF BOUNDS
013110  000776                    BR      .-2             ;;HANGUP
013112  006300                    ASL     R0              ;;POSITION FOR INDEXING
013114  016000  013134            MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
013120  000200                    RTS     R0              ;;GO TO ROUTINE

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

013122  011646    $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
013124  016666  000004  000002    MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
013132  000002                    RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                :       ROUTINE
                                :       -------
013134  013122    $TRPAD: .WORD   $TRAP2
013136  010242            $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
013140  010632            $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
013142  010606            $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
013144  010646            $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
013146  010362            $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

013150  011662            $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
013152  012002            $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
013154  012276            $RDDEC  ;;CALL=RDDEC    TRAP+10(104410) READ A DECIMAL NUMBER FROM TTY
013156  010146            $SAVREG ;;CALL=SAVREG   TRAP+11(104411) SAVE R0-R5 ROUTINE
013160  010204            $RESREG ;;CALL=RESREG   TRAP+12(104412) RESTORE R0-R5 ROUTINE
013162  011546            $PRHDR  ;;CALL=PRTHDR   TRAP+13(104413) PRINT TEST HEADER
013164  011502            $PRINT  ;;CALL=PRINT    TRAP+14(104414) PRINT ROUTINE
013166  011034            $LOAD   ;;CALL=LOAD     TRAP+15(104415) LOAD CHAR ROUTINE
013170  011400            $MLOAD  ;;CALL=MLOAD    TRAP+16(104416) MULTIPLE CHAR LOAD
013172  011410            $CNLD   ;;CALL=CNLOAD   TRAP+17(104417) LOAD CONVERTED NUMBER
```

```
        013204  007052                          $CHECK  ::CALL=CHECK    TRAP+20(104420) CHECK FOR KYBD OR SW REG CONTROL
        013206  007112                          $HOLD   ::CALL=HOLD     TRAP+21(104421) WAIT FOR OPERATOR ACTION
                000044                  $TERM=.-$TRPAD
```

```
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;**********************************************************************
                                ;POWER DOWN ROUTINE
013210  012737  013350  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
013216  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
013224  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
013226  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
013230  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
013232  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
013234  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
013236  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
013240  017746  165676                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
013244  010637  013354                MOV     SP,$SAVR6       ;;SAVE SP
013250  012737  013262  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
013256  000000                        HALT
013260  000776                        BR      .-2             ;;HANG UP

                                ;;**********************************************************************
                                ;POWER UP ROUTINE
013262  012737  013350  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
013270  013706  013354                MOV     $SAVR6,SP       ;;GET SP
013274  005037  013354                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
013300  005237  013354         1$:    INC     $SAVR6          ;;WAIT FOR THE INC
013304  001375                        BNE     1$              ;;OF  WORD
013306  012677  165630                MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
013312  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
013314  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
013316  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
013320  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
013322  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
013324  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
013326  012737  013210  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
013334  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
013342  104401                        TYPE                    ;;REPORT THE POWER FAILURE
013344  015215          $PWRMG: .WORD   PWRMSG                  ;;POWER FAIL MESSAGE POINTER
013346  000002                        RTI
013350  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
013352  000776                        BR      .-2             ;;  BEFORE THE POWER DOWN WAS COMPLETE
013354  000000          $SAVR6: 0                               ;;PUT THE SP HERE
```

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

                        .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

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

013356  002254          TAT:    TEST0
013360  003072                  TEST1
013362  003262                  TEST2
013364  000000                  0               ;TEST3
013366  000000                  0               ;TEST4
013370  000000                  0               ;TEST5
013372  000000                  0               ;TEST6
013374  000000                  0               ;TEST7
013376  000000                  0               ;TEST10
013400  000000                  0               ;TEST11
013402  000000                  0               ;TEST12
013404  000000                  0               ;TEST13
013406  000000                  0               ;TEST14
013410  000000                  0               ;TEST15
013412  000000                  0               ;TEST16
013414  000000                  0               ;TEST17
013416  003614                  TEST20
013420  003662                  TEST21
013422  003772                  TEST22
013424  004060                  TEST23
013426  004114                  TEST24
013430  004406                  TEST25
013432  004770                  TEST26
013434  005062                  TEST27
013436  005200                  TEST30
013440  005226                  TEST31
013442  177777                  -1              ;TEST32
013444  000000                  0               ;TEST33
013446  000000                  0               ;TEST34
013450  000000                  0               ;TEST35
013452  000000                  0               ;TEST36
013454  000000                  0               ;TEST37
013456  000000                  0               ;TEST40
013460  000000                  0               ;TEST41
013462  000000                  0               ;TEST42
013464  000000                  0               ;TEST43
013466  000000                  0               ;TEST44
013470  000000                  0               ;TEST45
013472  000000                  0               ;TEST46
013474  000000                  0               ;TEST47
013476  005340                  TEST50
013500  000000                  0               ;TEST51
013502  000000                  0               ;TEST52
013504  000000                  0               ;TEST53
013506  000000                  0               ;TEST54
```

```
013510  000000                  0               ;TEST55
013512  000000                  0               ;TEST56
013514  000000                  0               ;TEST57
013516  005540                  TEST60
013520  005654                  TEST61
013522  005766                  TEST62
013524  006050                  TEST63
013526  006174                  TEST64
013530  000000                  0               ;TEST65
013532  000000                  0               ;TEST66
013534  000000                  0               ;TEST67
013536  000000                  0               ;TEST70
013540  000000                  0               ;TEST71
013542  000000                  0               ;TEST72
013544  000000                  0               ;TEST73
013546  000000                  0               ;TEST74
013550  000000                  0               ;TEST75
013552  006514                  TEST76
013554  006534                  TEST77

                        ;////////////////////////////////////////////////////////////////////////
                        .SBTTL  ERROR MESSAGE ADDRESS TABLE
                        ;////////////////////////////////////////////////////////////////////////

013556  013632          EMAT:   ERR1
013560  013661                  ERR2
013562  013706                  ERR3
013564  013744                  ERR4
013566  014000                  ERR5
013570  014033                  ERR6
013572  014070                  ERR7
013574  014117                  ERR10
013576  014144                  ERR11
013600  014170                  ERR12
013602  014232                  ERR13
013604  014266                  ERR14
013606  014324                  ERR15
013610  014360                  ERR16
013612  014414                  ERR17
013614  014451                  ERR20
013616  014503                  ERR21
013620  014537                  ERR22
013622  014573                  ERR23
013624  014632                  ERR24
013626  014671                  ERR25
013630  014713                  ERR26
```

ERROR MESSAGE ADDRESS TABLE

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

013632  051105  047522  020122  ERR1:   .ASCIZ  /ERROR CLEAR, POWER OFF/
013640  046103  040505  026122
013646  050040  053517  051105
013654  047440  043106     000
013661     122  040505  054504  ERR2:   .ASCIZ  /READY SET, POWER OFF/
013666  051440  052105  020054
013674  047520  042527  020122
013702  043117  000106
013706  051105  047522  020122  ERR3:   .ASCIZ  /ERROR CLEAR, PRINTER OFF LINE/
013714  046103  040505  026122
013722  050040  044522  052116
013730  051105  047440  043106
013736  046040  047111  000105
013744  042522  042101  020131  ERR4:   .ASCIZ  /READY SET, PRINTER OFF LINE/
013752  042523  026124  050040
013760  044522  052116  051105
013766  047440  043106  046040
013774  047111  000105
014000  051105  047522  020122  ERR5:   .ASCIZ  /ERROR SET, PRINTER ON LINE/
014006  042523  026124  050040
014014  044522  052116  051105
014022  047440  020116  044514
014030  042516     000
014033     122  040505  054504  ERR6:   .ASCIZ  /READY CLEAR, PRINTER ON LINE/
014040  041440  042514  051101
014046  020054  051120  047111
014054  042524  020122  047117
014062  046040  047111  000105
014070  051105  047522  020122  ERR7:   .ASCIZ  /ERROR CLEAR, PAPER OUT/
014076  046103  040505  026122
014104  050040  050101  051105
014112  047440  052125     000
014117     122  040505  054504  ERR10:  .ASCIZ  /READY SET, PAPER OUT/
014124  051440  052105  020054
014132  040520  042520  020122
014140  052517  000124
014144  051105  047522  020122  ERR11:  .ASCIZ  /ERROR DID NOT CLEAR/
014152  044504  020104  047516
014160  020124  046103  040505
014166  000122
014170  042522  042101  020131  ERR12:  .ASCIZ  /READY NOT SET AFTER ERROR CLEARED/
014176  047516  020124  042523
014204  020124  043101  042524
014212  020122  051105  047522
014220  020122  046103  040505
014226  042522  000104
014232  051105  047522  020122  ERR13:  .ASCIZ  /ERROR SET AFTER RESET INSTR/
014240  042523  020124  043101
014246  042524  020122  042522
014254  042523  020124  047111
```

```
014262  052123  000122
014266  042522  042101  020131  ERR14:  .ASCIZ  /READY CLEAR AFTER RESET INSTR/
014274  046103  040505  020122
014302  043101  042524  020122
014310  042522  042523  020124
014316  047111  052123  000122
014324  042522  042101  020131  ERR15:  .ASCIZ  /READY SET AFTER CHAR LOADED/
014332  042523  020124  043101
014340  042524  020122  044103
014346  051101  046040  040517
014354  042504  000104
014360  051105  047522  020122  ERR16:  .ASCIZ  /ERROR SET AFTER CHAR LOADED/
014366  042523  020124  043101
014374  042524  020122  044103
014402  051101  046040  040517
014410  042504  000104
014414  042522  042101  020131  ERR17:  .ASCIZ  /READY NEVER SET, CHAR LOADED/
014422  042516  042526  020122
014430  042523  026124  041440
014436  040510  020122  047514
014444  042  1  042105     000
014451     105  051122  051117  ERR20:  .ASCIZ  /ERROR SET, INTERRUPT TEST/
014456  051440  052105  020054
014464  047111  042524  051122
014472  050125  020124  042524
014500  052123     000
014503     122  040505  054504  ERR21:  .ASCIZ  /READY CLEAR, INTERRUPT TEST/
014510  041440  042514  051101
014516  020054  047111  042524
014524  051122  050125  020124
014532  042524  052123     000
014537     120  044522  052116  ERR22:  .ASCIZ  /PRINTER INTER ABOVE LEVEL 3/
014544  051105  044440  052116
014552  051105  040440  047502
014560  042526  046040  053105
014566  046105  031440     000
014573     116  020117  051120  ERR23:  .ASCIZ  /NO PRINTER INTER BELOW LEVEL 4/
014600  047111  042524  020122
014606  047111  042524  020122
014614  042502  047514  020127
014622  042514  042526  020114
014630  000064
014632  051120  047111  042524  ERR24:  .ASCIZ  /PRINTER ERROR BEFORE CHAR LOAD/
014640  020122  051105  047522
014646  020122  042502  047506
014654  042522  041440  040510
014662  020122  047514  042101
014670     000
014671     120  044522  052116  ERR25:  .ASCIZ  /PRINTER NOT READY/
014676  051105  047040  052117
014704  051040  040505  054504
014712     000
014713     120  044522  052116  ERR26:  .ASCIZ  /PRINTER ERROR AFTER CHAR LOAD/
014720  051105  042440  051122
014726  051117  040440  052106
014734  051105  041440  040510
```

```
014742  020122  047514  042101
014750     000


                                ;////////////////////////////////////////////////////////////////////////////////
                                .SBTTL  PROGRAM MESSAGES
                                ;////////////////////////////////////////////////////////////////////////////////

014751     200  020043  047503  COLUMN: .ASCIZ  <CRLF>/# COLUMNS = /
014756  052514  047115  020123
014764  020075     000
014767     200  042523  042514  SELTST: .ASCIZ  <CRLF>/SELECT TEST #   /
014774  052103  052040  051505
015002  020124  020043  020040
015010     000
015011     012  052012  051505  TSTNO:  .ASCIZ  <LF><LF>/TEST NUMBER  /
015016  020124  052516  041115
015024  051105  020040     000
015031     040  041440  046117  COLMN:  .ASCIZ  /  COLUMNS/<LF>
015036  046525  051516  000012
015044  052200  051505  020124  ETSTNO: .ASCIZ  <CRLF>/TEST #/
015052  000043
015054  020054  050040  036503  PCMSG:  .ASCIZ  /,  PC=/
015062     000
015063     054  020040  051105  ERR:    .ASCIZ  /,  ERROR #/
015070  047522  020122  000043
015076  020054  000040          ERRS:   .ASCIZ  /,  /
```

```
015102  042412  042116  047440  PASMSG: .ASCIZ  <LF>/END OF PASS  #/
015110  020106  040520  051523
015116  020040  000043
015122  051600  051127  036440  DSMSG1: .ASCIZ  <CRLF>/SWR = /
015130  000040
015132  020040  047040  053505  DSMSG2: .ASCIZ  /   NEW = /
015140  036440  000040
015144  040527  052111  047111  WTMSG:  .ASCIZ  /WAITING.  TYP SPACE TO CONTINUE/<CRLF>
015152  026107  020040  054524
015160  020120  050123  041501
015166  020105  047524  041440
015174  047117  044524  052516
015202  100105     000
015205     103  040510  020122  TCHAR:  .ASCIZ  /CHAR = /
015212  020075     000
015215     200  047520  042527  PWRMSG: .ASCIZ  <CRLF>/POWER/<CRLF>
015222  100122     000
015225     012  047516  041440  NCMSG:  .ASCIZ  <LF>/NO CONSOLE TERMINAL/<LF>
015232  047117  047523  042514
015240  052040  051105  044515
015246  040516  005114     000
015253     120  044522  052116  MANMSG: .ASCII  /PRINT SPEED MANUAL TIMING/<CRLF>
015260  051440  042520  042105
015266  046440  047101  040525
015274  020114  044524  044515
015302  043516     200
015305     120  052125  051440          .ASCII  /PUT SWITCH 12 UP TO START TIMING/<CRLF>
015312  044527  041524  020110
015320  031061  052440  020120
015326  047524  051440  040524
015334  052122  052040  046511
015342  047111  100107
015346  052520  020124  053523          .ASCIZ  /PUT SWITCH 12 DOWN AT END OF 1 MINUTE/<CRLF>
015354  052111  044103  030440
015362  020062  047504  047127
015370  040440  020124  047105
015376  020104  043117  030440
015404  046440  047111  052125
015412  100105     000
015415     105  052116  051105  T64MSG: .ASCIZ  /ENTER PATTERN PER LISTING OR DOCUMENT/<CRLF>
015422  050040  052101  042524
015430  047122  050040  051105
015436  046040  051511  044524
015444  043516  047440  020122
015452  047504  052503  042515
015460  052116  000200
015464  003407  000200          TWOBEL: .ASCIZ  <7><7><CRLF>
015470  050200  044522  052116  PRSP1:  .ASCIZ  <CRLF>/PRINT SPEED IS /
015476  051440  042520  042105
015504  044440  020123     000
015511     101  050120  047522  PRSP2:  .ASCIZ  /APPROX. /
015516  027130  000040
015522  020040  044514  042516  PRSP3:  .ASCIZ  /  LINES/<57>/MINUTE . WITH
015530  027523  044515  052516
015536  042524  026040  053440
015544  052111  020110     000
```

```
015551     040  041440  040510  PRSP4:  .ASCIZ  /  CHARS/<57>/LINE/<CRLF>
015556  051522  046057  047111
015564  100105     000
015567     124  051125  020116  TOMSG0: .ASCIZ  /TURN POWER OFF & SET OFF LINE/<CRLF>
015574  047520  042527  020122
015602  043117  020106  020046
015610  042523  020124  043117
015616  020106  044514  042516
015624  000200
015626  045517  020054  052524  TOMSG1: .ASCIZ  /OK, TURN POWER ON/<CRLF>
015634  047122  050040  053517
015642  051105  047440  100116
015650     000
015651     117  026113  051440  TOMSG2: .ASCIZ  /OK, SET PRINTER TO ON-LINE/<CRLF>
015656  052105  050040  044522
015664  052116  051105  052040
015672  020117  047117  046055
015700  047111  100105     000
015705     117  026113  052040  TOMSG3: .ASCIZ  /OK, TRY PAPER OUT SWITCH/<CRLF>
015712  054522  050040  050101
015720  051105  047440  052125
015726  051440  044527  041524
015734  100110     000
015737     117  026113  051040  TOMSG4: .ASCIZ  /OK, RESTORE PRINTER TO ON-LINE/<CRLF>
015744  051505  047524  042522
015752  050040  044522  052116
015760  051105  052040  020117
015766  047117  046055  047111
015774  100105     000
015777     055  026455  026455  T1MSG1: .ASCIZ  /----- /
016004  000040
016006  044440  041516  020110  T1MSG2: .ASCIZ  / INCH FORM FEED -----/<CR>
016014  047506  046522  043040
016022  042505  020104  026455
016030  026455  006455     000
016035     123  052105  043040  T1MSG3: .ASCIZ  /SET FORM FEED SWITCH TO  /
016042  051117  020115  042506
016050  042105  051440  044527
016056  041524  020110  047524
016064  020040     000
016067     040  044440  041516  T1MSG4: .ASCIZ  /  INCHES & DEPRESS TOF RESET SWITCH/<CRLF>
016074  042510  020123  020046
016102  042504  051120  051505
016110  020123  047524  020106
016116  042522  042523  020124
016124  043523  052111  044103
016132  000200
016134  047516  046440  052105  T2EM:   .ASCIZ   NO METHOD OF TIMING AVAILABLE/<CRLF>
016142  047510  020104  043117
016150  052040  046511  047111
016156  020107  053101  044501
016164  040514  046102  100105
016172     000
        016174                        .EVEN
```

```
016174  047531  020125  047506  TABL64: .ASCIZ  /YOU FORGOT TO ENTER DATA; CTL-SPACE/<LF>
016202  043522  052117  052040
016210  020117  047105  042524
016216  020122  040504  040524
016224  020073  052103  026514
016232  050123  041501  005105
016240     000

                                                ;OPEN-ENDED TABLE FOR TEST 64 DATA.


                00000:                  .END
```

```
ACCEPT  006210
ACTEST  001130
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
CHACTR  006510
CHECK = 104420
CHLD    005440
CKFLAG  001155
CNLOAD= 104417
CNTR    006606
COLCTR  006506
COLMN   015031
COLUMN  014751
CONTRL  001220
CR    = 000015
CRLF  = 000200
CSBR    001136
CTLA    006346
CTLB    006362
DCT     006556
DDISP = 177570
DISPLA  001144
DISPRE  000174
DSMSG1  015122
DSMSG2  015132
DSWR  = 177570
DUNCOL  006454
DUMPA   006446
EMAT    013556
EMTVEC= 000030
ENDADR  006512
ERR     015063
ERRS    015076
ERRVEC= 000004
ERR1    013632
ERR10   014117
ERR11   014144
ERR12   014170
ERR13   014232
ERR14   014266
ERR15   014324
ERR16   014360
ERR17   014414
ERR2    013661
ERR20   014451
ERR21   014503
ERR22   014537
ERR23   014573
ERR24   014632
ERR25   014671
ERR26   014713
ERR3    013706
ERR4    013744
ERR5    014000
ERR6    014033
ERR7    014070
ETSTNO  015044
EXIT    006644
FF    = 000014
FSTPAD  001126
FSTSAD  001124
HOLD  = 104431
HT    = 000011
IOTVEC= 000020
KYBDF   007140
KYBDST  007442
LDPTR   011040
LF    = 000012
LKS     001140
LKSRV   006566
LOAD  = 104415
LPB     001116
LPCTR   001132
LPKB    001112
LPKS    001110
LPS     001114
MANMSG  015253
MINCNT  006612
MLOAD = 104415
NCMSG   015225
NUMLP   001122
NXTADR  011142
PASCNT  005652
PASMSG  015102
PCMSG   015054
PIRQ  = 177772
PIRQVE= 000240
PLKS    001134
PLOAD   011044
PRINT = 104414
PROPAT  006424
PRPAT   006466
PRSP1   015470
PRSP2   015511
PRSP3   015522
PRSP4   015551
PRTHDR= 104413
PRTSET  005442
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PWRMSG  015215
PWRVEC= 000024
QUO1    001764
QUO2    002032
RDCHR = 104406
RDDEC = 104410
RDLIN = 104407
REPORT  010044
RESREG= 104412
RESTRT  001210
RESVEC= 000010
RSTADR  011316
R6    =%000006
R7    =%000007
SAVREG= 104411
SELECT  006752
SELTST  014767
SERSW   001120
SLOAD   011206
SP0     003500
SP3     005434
SRDCI   006614
STACK = 001100
START   001174
STARTX  001226
STKLMT= 177774
STRONE  001152
SVTST   011660
SWR     001142
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
TABL64  016174
TAT     013356
TBITVE= 000014
TCHAR   015205
TERR    006164
TEST0   002254
TEST1   003072
TEST2   003262
TEST20  003614
TEST21  003662
TEST22  003772
TEST23  004060
TEST24  004114
TEST25  004406
TEST26  004770
TEST27  005062
TEST30  005200
TEST31  005226
TEST50  005340
TEST60  005540
TEST61  005654
TEST62  005766
TEST63  006050
TEST64  006174
TEST76  006514
TEST77  006534
TKB     001102
TKS     001100
TKVEC = 000060
TLOOP   001154
TORTN   006610
TPB     001106
TPS     001104
TPVEC = 000064
TRAPVE= 000034
TRONE   001153
TRTVEC= 000014
TSEL    007510
TSTNO   015011
TWOBEL  015464
TYPDS = 104405
TYPE  = 104401
TYPOC = 104402
TYPON = 104404
TYPOS = 104403
TOMSG0  015567
TOMSG1  015626
TOMSG2  015651
TOMSG3  015705
TOMSG4  015737
T1MSG1  015777
T1MSG2  016006
T1MSG3  016035
T1MSG4  016067
T2EM    016134
T2SP    003400
T50M1   005424
T50M2   005430
T50PCT  005422
T64MSG  015415
WIDTH   001150
WTMSG   015144
$BELL   001156
$CHECK  007052
$CNLD   011410
$CNTLG  012247
$CNTLU  012242
$CR     001164
$CRLF   001161
$DBLK   010576
$DB2D   012454
$DB2O   012704
$DECVL  012634
$DOAGN  005332
$DTBL   010566
$ENDAD  005322
```

```
$ENDCT  005306          $LOAD   011034          $QUES   001160          $SWR  = 162000          $T?IN  012236
$EOP    005262          $MLOAD  011400          $RDCHR  011662          $TEPM = 000044          $TYPDS  010362
$EOPCT  005300          $MNEW   012265          $RDDEC  012276          $TKB    001102          $TYPE   010242
$ERROR  007744          $MSWR   012254          $RDLIN  012002          $TKS    001100          $TYPEC  010344
$ERRPC  010040          $NULL   001170          $RDSZ = 000004          $TN   = 000000          $TYPOC  010632
$FF     001166          $OCNT   011030          $RESRE  010204          $TNPWR  012564          $TYPON  010646
$FILLC  001172          $OCTVL  013006          $RTNAD  005334          $TPB    001106          $TYPOS  010606
$FILLS  001171          $OMODE  011032          $SAVRE  010146          $TPFLG  001173          $$GET4= 000000
$GET42  005312          $PASS   005336          $SAVR6  013354          $TPS    001104          $OFILL  011031
$HC   = 000000          $PRHDR  011546          $SB2D   012650          $TRAP   013060          .     = 01624!
$HOLD   007112          $PRINT  011502          $SB20   013024          $TRAP2  013132
$!__JP  013350          $PWRDN  013210          $SETUP= 000034          $TRP  = 000022
$!TEMB  013042          $PWRMG  013344          $STUP = 177777          $TRPAD  013144
$LF     001162          $PWRJP  013262          $SVPC = 000200          $TSTNM  001146

. ABS.  01624!     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DZLAEB,DZLAEB.SEQ/SOL:NL:SEQ=DZLAEB.P11
 RUN-TIME: 16 9 .3 SECONDS
 RUN-TIME RATIO: 642/26=24.3
 CORE USED:  18K  (36 PAGES)
```