PDP-8 Family
Commonly Used
Utility Routines

(Dumps, Verifier, Duplicator,

Conversion and Printing Routines)

For additional copies, order NO. DEC-8I-RZPA-D from the Program Library, Digital Equipment Corporation, Maynard, Mass, 01754.

PDP-8
LIBRARY

First Edition, January 1971

This volume is a collection
of manuals printed 1965-1970.

PREFACE

This document is a collection of proven routines for the PDP-8 family of computers. These routines in themselves are useful to many programmers and are all illustrative of assembly language programming techniques. The user is advised to first investigate the sections on assembly language programming in Programming Languages and Introduction to Programming 1970.

CONTENTS

CHAPTER 1
RIM LOADER

1.1 ABSTRACT

The Read-In Mode (RIM) Loader is a minimum-sized routine for reading and storing into core information contained in Read-In-Mode coded tapes via the 33-ASR perforated tape reader or high speed perforated tape reader.

1.2 REQUIREMENTS

The RIM Loader requires 17_{10} (21_8) core locations and is used with any PDP-8® family computer with a 33-ASR Teletype®. A high speed perforated tape reader is optional.

1.3 USAGE

1.3.1 Loading

To place the RIM Loader into memory via the console switches, proceed as follows:

a. Set 7756 in the switch register (SR).
b. Press LOAD ADDress.
c. Set the first instruction (6032 for 33-ASR).
d. Press DEPosit.
e. Set the next instruction (6031 for 33-ASR).
f. Press DEPosit.
g. Repeat steps e. and f. until all instructions have been deposited.

1.3.2 Start-up/Entry

a. Place the perforated tape which must be in RIM format in the perforated-tape reader.

®PDP, Programmed Data Processor, is a registered trademark of the Digital Equipment Corporation.

®Teletype is a trademark of the Teletype Corporation.

b. Make sure the reader is set to LINE.

c. Place the starting address (7756) in the switch register.

d. Press the LOAD ADDress key.

e. Press the START key.

f. If the 33-ASR version is used, move the reader control to START.

There are no error stops in this routine.

1.4 DESCRIPTION

This is a basic routine that alternately assembles an address from two successive characters on tape, then assembles data contained in the next two characters and stores this data at the associated address.

Because a tape in RIM format is twice as long as a comparable tape in binary format, it is suggested that the RIM Loader only be used to load the Binary Loader. After this, the Binary Loader should be used.

Any tapes to be read by this program must be in Read-In-Mode coded format.

Leader tape for RIM format tapes should be about two feet of leader-trailer codes; i.e., any code with channel 8 punched, preferably code 200. (Depress ALT MODE, CTRL, and @ keys simultaneously to punch 200 leader trailer.)

Characters representing the absolute, machine language program are arranged in an alternating pattern of address, contents, address, contents, etc. Addresses have channel 7 punched, channel 8 not punched. Contents have no punch in channel 7 or 8.

Trailer tape should be the same as leader tape.

1.5 EXAMPLE OF READ-IN-MODE CODED FORMAT

Tape Channel 87 654 S 321	Channels 8 and 7 Indicate
10 000 . 000	Leader code is always found first.
01 A1 . A2 00 A3 . A4	Absolute octal Address of data in next two characters.
00 D1 . D2 00 D3 . D4	Octal Data to be stored at preceding address.
This pattern repeats in similar four character groups until	
10 000 . 000	the concluding Trailer Code is encountered.

Note that a 3-bit group (a single octal character) is designated by the notation A1 above. The 1 in this notation indicates that this particular octal character is used as the most significant three bits in specifying the absolute address into which following data is deposited. Correspondingly, A2, A3, and A4 designate successively less significant octal characters in the absolute address.

The remarks above apply equally to data as specified by the notation D1, D2, D3, and D4.

1.6 PROGRAM LISTING

1.6.1 33-ASR Version

```
Abs      Octal
Addr.    Contents      Instruction          Comments

7756,    6032     BEG,     KCC          /clear AC and flag
7757.    6031              KSF          /skip if flag = 1
7760,    5357              JMP .-1      /looking for char
7761,    6036              KRB          /read buffer
7762,    7106              CLL RTL      /ch8 in ACO
7763,    7006              RTL          /checking for leader
7764,    7510              SPA          /found leader
7765,    5357              JMP BEG+1    /OK,ch7 in link
7766,    7006              RTL
7767,    6031              KSF
7770,    5367              JMP .-1
7771,    6034              KRS          /read, do not clear
7772,    7420              SNL          /checking for address
7773,    3776              DCA I TEMP   /store contents
7774,    3376              DCA TEMP     /store address
7775,    5356              JMP BEG      /next word
7776,    0        TEMP,    0            /temp storage
7777,    JMP start of      0
             BIN loader
```

1.6.2 High-speed Version

```
Abs      Octal
Addr.    Contents      Instruction          Comments

7756     6014     BEG,     RCF          /clear flag and fetch char.
7757     6011              RSF          /skip if flag = 1
7760     5357              JMP .-1      /wait for fetching to be done
7761     6016              RCC          /put char.in AC; fetch another
7762     7106              CLL RTL      /rotate channel 8 to
7763     7006              RTL          /bit Ø of AC
7764     7510              SPA          /was it set (leader-trailer)?
7765     5374              JMP TEMP-2   /yes - leader trailer
7766     7006              RTL          /no - rotate channel 7 to link
7767     6011              RSF          /character fetched yet?
7770     5367              JMP .-1      /no - wait for it
7771     6016              RCC          /yes - add it to AC; fetch
                                        /another
7772     7420              SNL          /is 12-bit word in AC an
                                        /address?
7773     3776              DCA I TEMP   /no - store in last stored
                                        /address
7774     3376              DCA TEMP     /yes - store new address
7775     5357              JMP BEG+1    /get next word
7776       Ø       TEMP,     Ø          /temporary storage
7777                                    /start of binary loader
```

1.7 OTHER POSSIBLE RIM LOADERS

Variations of the RIM Loader may prove useful in special cases where, due to circumstances, RIM must be located in a different section of core.

On the other hand, the equipment involved may make it necessary to use a variation of RIM tailored specifically to a particular situation. As an example of this, consider a special-purpose PDP-8 system used for text editing. In this system, no 33-ASR's are used. Instead, several typewriters which use a different code (including provision for upper and lower case) are time-shared with respect to input and output with a central PDP-8.

Please consult the Applied Programming Department at Digital Equipment Corporation for details of other RIM Loaders currently available or for assistance in special cases.

1.8 USE OF PDP-8 SYSTEM PROGRAMS

Certain system programs, such as the DECtape Library System (DEC-08-SUCO), require that the RIM Loader be used precisely as listed in section 1.6.

1.9 USING THE RIM LOADER WITH EXTENDED MEMORY

The RIM Loader as described in section 1.6 can run in any memory field provided that it is loaded into memory following a slightly different procedure than that described in 1.3.1. The Instruction Field register and the Data Field register must both be set to N (a number from 0 to 7) where N indicates the memory field in which the RIM Loader is to be placed. This is easily done.

a. Set the DATA FIELD extension of the switch register to N.
b. Set the INSTRUCTION FIELD extension of the switch register to N.
c. Follow procedure in steps a through g in section 1.3.1.

CHAPTER 2
BINARY LOADER

2.1 ABSTRACT

The Binary Loader is a short routine for reading and storing information contained in binary-coded tapes, using the 33-ASR reader or the High-Speed Reader.

The Binary Loader accepts tapes prepared with the PAL III, PAL-D, PAL8, or MACRO-8 assemblers. Diagnostic messages may be included on tapes produced when using either PAL or MACRO. The Binary Loader ignores all diagnostic messages.

2.2 REQUIREMENTS

This program occupies 94_{10} (136_8) core locations.

The Binary Loader can be used with a system consisting of the PDP-8 and a 33-ASR Teletype only. On the other hand, the same program operates with systems including the High-Speed Tape Reader and/or the Memory Extension Control. This loader is compatible with the 552 DECtape Library System and the TC01 DECtape Library System.

2.3 LOADING PROCEDURES

The Binary Loader is brought into memory by the RIM or Read-In-Mode Loader. This requires that the Binary Loader tape itself be in RIM format. See Introduction to Programming and Chapter 1 for discussions of the RIM Loader and RIM format.

NOTE: Memory Extension users; refer to Special Requirements section.

Proceed as follows:

a. Load the RIM Loader for the type of reader which is to load the Binary Loader.

b. Place the Binary Loader tape in the reader.

c. Make sure that the reader is on-line.

d. Place the starting address of the RIM Loader (7756) in the SWITCH REGISTER.

e. Press the LOAD ADDRESS key.

f. Press the START key.

g. If the 33-ASR is the chosen reader, move the READER CONTROL switch to the START position.

2.3.1 Switch Setting

NOTE: Memory Extension users see "Special Requirements" section.

2.4 USING THE PROGRAM

a. Place the tape to be loaded (which must be in binary format) in either the 33-ASR Tape Reader or the High-Speed Reader, with leader-trailer under the read head. When using the 33-ASR, make sure the reader is on-line. When using the High-Speed reader, make sure the reader is on.

b. Place the starting address of the Binary Loader (7777) in the SWITCH REGISTER.

c. Press LOAD ADDRESS key.

   When using the High-Speed Reader, change the SWITCH REGISTER to 3777 (bit 0 = 0). Omit this step if using the 33-ASR.

d. Press console START key.

   When using the 33-ASR, move the READER CONTROL switch to START.

2.5 ERRORS

When any of the PDP-8 assemblers is used to produce a binary tape, a checksum is automatically punched at the end of the binary tape. The checksum is the sum of all data on the tape including the origin word.

To be more specific, it is the sum of all data contained on tape that will enter the accumulator (AC) in bit positions 4 through 11 from, for example, the 33-ASR Reader buffer. The sum is accumulated character by character and not word by word. Overflow (a carry out of the most-significant bit position of the AC) is ignored both when calculating a checksum (which is done by the assembler used) and when the Binary Loader accumulates a checksum while loading a tape.

If the checksum accumulated while using the Binary Loader does not agree with the last two characters on the tape (i.e., the checksum on the tape calculated and placed there by the assembler), an error has occurred.

When the computer halts, the display lights will be static, the memory buffer (MB) will contain 7402, and the contents of the AC will be unequal to zero if a checksum error has occurred.

Restart the computer after the tape has been repositioned by pressing the CONTINUE key.

2.6 DETAILS OF OPERATION AND STORAGE

This program furnishes the basic means by which the contents of binary-coded tapes are loaded into core.

The heart of the program is a short subroutine (tagged BEGG) which operates in outline as follows:

The incoming character is tested to see if it is a "rubout" (all eight tape channels punched).

If this is the case, all subsequent information coming from the reader is ignored until another rubout is detected.

This is the mechanism by which assembler diagnostic messages are detected. They are preceded and followed by a single rubout character. Within a diagnostic message, in contrast to the rules concerning the balance of the binary tape, any character is valid except, of course, a single rubout character which would prematurely conclude the diagnostic message. Note that two consecutive rubouts within a diagnostic message would, in effect, be ignored.

Next the character is tested to see if it is leader or field setting.

These tests are listed in the order in which they are performed. If none of the actions indicated have occurred upon exit from the BEGG subroutine, the character is part of the origin address, contains part of a data word, or is a part of the checksum, and the appropriate course is followed by the main routine.

2.7 SPECIAL REQUIREMENTS OR FORMATS

2.7.1 External Format

Tapes to be read by this program must be in binary-coded format and have about 1 foot of leader-trailer code (any code with channel 8 punched; preferably code 200).

The first two characters represent the address (origin) into which the first command on the next portion of the tape will be placed. Successive commands are placed in memory at addresses:

origin+1,origin+2,...,origin+n.

The initial character of the origin has no punch in channel 8, while channel 7 is punched. The second character designating the origin has no punches in either channel 8 or 7.

A concluding 2-character group representing the checksum has no punches present in channels 8 or 7.

Trailer tape is similar to leader.

Reference to Program Listing indicates that after the BEGG subroutine tests to see if the character just read was leader/trailer, a test is made to determine whether the character is a "field setting." This is a reference to the fact that the assemblers produce tapes on which characters of the form

11 XXX 000

indicate the memory field into which the following data is to be loaded. If, for example XXX were 101, all data following the field designator should be loaded into memory field five. Unlike origins and other data, field settings are not included in the checksum.

2.7.2 Example of Binary Loader Format

Tape Channel 87 654 S 321	Channels 8 and 7 Indicate	Program Proper	Notes
10 000 . 000	Leader	No	
01 000 . 010 00 000 . 000	Origin	No	In octal the origin 0200. Loading will start at 0200.
00 111 . 010 00 000 . 000	Contents of 200	Yes	The command 7200 or CLA.
00 011 . 010 00 111 . 110	Contents of 201	Yes	The command 3276 or DCA Z 076.

Example of Binary Loader Format (Cont.)

Tape Channel 87 654 S 321	Channels 8 and 7 Indicate	Program Proper	Notes
00 111 . 100 00 000 . 010	Contents of 202	Yes	The command 7402 or HLT.
00 000 . 100 00 010 . 010	Checksum	No	The program determines that these two characters are the checksum since trailer follows.
10 000 . 000	Trailer	No	

The octal checksum in this example is 0422. Note that this is the following sum:

```
102     Origin
000
072     First word
000
032     Second word
076
074     Third word
002
---
422
```

2.7.3 Memory Extension Usage

It is recommended that the Binary Loader exist in field 0. This ensures a permanent program lining around location 7754 and 7755 which are used for TC01 DECtape. The loader can exist in any field, though caution must be taken not to use location 7754 and 7755 in field 0 (this applies only to DECtape users). Also, when the proper field is chosen it should be noted that the RIM Loader must already be in that field.

Binary Loader Loading Procedure for Extended Memory Users

a. Place the Binary Loader tape in the reader.
b. Place the proper FIELD in the INSTRUCTION FIELD REGISTER when putting the starting address of the RIM Loader (7756) in the SWITCH REGISTER.
c. Press the LOAD ADDRESS key.
d. Press the START key.
e. Start the reader. 33-ASR: press READER CONTROL to start. High-Speed Reader: should already be ready to start.

Operation and Usage for Extended Memory Users

a. Place the tape to be loaded (tape must be in binary format) in the reader. When using the 33-ASR, make sure reader is on-line. When using the High-Speed Reader, make sure reader is on and tape is positioned with leader/trailer over read head.

b. In the DATA FIELD register place the field in which the program is to be loaded. In the INSTRUCTION FIELD register place the field that the Binary Loader is in.

c. Press LOAD ADDRESS key.
When using the High-Speed Reader, change the SWITCH REGISTER TO 3777 (bit 0 = 0). Omit this step if using the 33-ASR.

d. Press console START key.

Starting Program

After program has been successfully loaded, place starting address of program in SWITCH REGISTER. Place the field where program exists in the FIELD INSTRUCTION REGISTER.

Press LOAD ADDRESS key.

Press START key.

2.8 PROGRAM LISTING

```
                    /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                    /MAYNARD, MASSACHUSETTS
                    /BINARY AND DECTAPE LOADERS FOR
                    /555 CONTROL

                    *7612
7612    0000        SWITCH,         0
7613    0000        MEMTEM,         0
7614    0000        CHAR,           0
7615    0000        CHKSUM,         0
7616    0000        ORIGIN,         0

                    *7626
                    /EXTRACT ERRORS, FIELD, L/T
7626    0000        BEGG,           0
7627    3212                        DCA SWITCH      /SET SWITCH
7630    4260                        JMS READ        /GET A CHARACTER
7631    1300                        TAD M376        /TEST FOR 377
7632    7750                        SPA SNA CLA
7633    5237                        JMP .+4         /NO
7634    2212                        ISZ SWITCH      /YES: COMPLEMENT SWITCH
7635    7040                        CMA
7636    5227                        JMP BEGG+1
7637    1212                        TAD SWITCH      /NOT 377
7640    7640                        SZA CLA         /IS SWITCH SET?
7641    5230                        JMP BEGG+2      /YES; IGNORE
7642    1214                        TAD CHAR        /NO; TEST FOR CODE
7643    0274                        AND MASK        /TYPES
7644    1341                        TAD M200
7645    7510                        SPA
7646    2226                        ISZ BEGG        /DATA OR ORIGIN
7647    7750                        SPA SNA CLA
7650    5626                        JMP I BEGG      /DATA, ORIGIN, or L/T

7651    1214                        TAD CHAR        /FIELD SETTING
7652    0256                        AND FMASK
7653    1257                        TAD CHANGE
7654    3213                        DCA MEMTEM
7655    5230                        JMP BEGG+2      /CONTINUE INPUT
7656    0070    FMASK,              70
7657    6201    CHANGE,             CDF
7660    0000    READ                0
7661    0000                        0
7662    6031    LOR,                KSF             /WAIT FOR FLAG
7663    5262                        JMP .-1
7664    6036                        KRB
7665    3214                        DCA CHAR
7666    1214                        TAD CHAR
7667    5660                        JMP I READ
7670    6011    HIR,                RSF
7671    5270                        JMP .-1
7672    6016                        RRB RFC
```

```
7673    5265                    JMP LOR+3
7674    0300    MASK,           300

                /TRAILER CODE SEEN
7675    4343    BEND,           JMP ASSEMB
7676    7041                    CIA
7677    1215                    TAD CHKSUM
7700    7402    M376,           HLT
7701    6032    BEGIN,          KCC
7702    6014                    RFC
7703    6214                    RDF
7704    1257                    TAD CHANGE
7705    3213                    DCA MEMTEM      /SAVE FIELD INSTRUCTION
7706    7604                    CLA OSR
7707    7700                    SMA CLA
7710    1353                    TAD HIRI
7711    1352                    TAD LORI
7712    3261                    DCA READ+1
7713    4226                    JMS BEGG
7714    5313                    JMP .-1         /IGNORE LEADER

7715    3215    GO,             DCA CHKSUM
7716    1213                    TAD MEMTEM
7717    3336                    DCA MEMFLD
7720    1214                    TAD CHAR
7721    3376                    DCA WORD1
7722    4260                    JMS READ
7723    3355                    DCA WORD2
7724    4226                    JMS BEGG        /LOOK AHEAD
7725    5275                    JMP BEND        /TRAILER, END
7726    4343                    JMS ASSEMB
7727    7420                    SNL
7730    5336                    JMP MEMFLD

7731    3216                    DCA ORIGIN
7732    1376    CHEX,           TAD WORD1
7733    1355                    TAD WORD2
7734    1215                    TAD CHKSUM
7735    5315                    JMP GO

7736    0000    MEMFLD,         0
7737    3616                    DCA I ORIGIN
7740    2216                    ISZ ORIGIN
7741    7600    M200,           7600
7742    5332                    JMP CHEX

7743    0000    ASSEMB,         0
7744    1376                    TAD WORD1
7745    7106                    CLL RTL
7746    7006                    RTL
7747    7006                    RTL
7750    1355                    TAD WORD2
7751    5743                    JMP I ASSEMB
7752    5262    LORI,           JMP LOR
```

```
7753    0006    HIRI,               HIR-LOR
7754    0000
                WORD1=7776
7755    0000    WORD2,              0

                *7777
7777    5301                        JMP BEGIN

ASSEMB  7743
BEGG    7626
BEGIN   7701
BEND    7675
CHANGE  7657
CHAR    7614
CHEX    7732
CHKSUM  7615
FMASK   7656
GO      7715
HIR     7670
HIRI    7753
LOR     7662
LORI    7752
MASK    7674
MEMFLD  7736
MEMTEM  7613
M200    7741
M376    7700
ORIGIN  7616
READ    7660
SWITCH  7612
WORD1   7776
WORD2   7755
```

CHAPTER 3

RIM PUNCH

3.1 ABSTRACT

The RIM Punch program provides a means of punching information contained in selected blocks of core memory as RIM-coded tape via the 33-ASR Perforated Tape Punch or 75E High Speed Punch. The punch program may occupy either low or high memory depending on the version used.

3.2 REQUIREMENTS

The RIM Punch program will run on any PDP-8 family computer with a 33-ASR (Teletype) or 75E (high-speed) punch.

This program requires 61_{10} (75_8) memory locations.

Program tapes are as follows:

33-ASR Version		High-speed Punch Version	
Low Memory Binary	DEC-08-PMP1-PB	Low Memory Binary	DEC-08-PMP4-PB
Low Source	DEC-08-PMP1-PA	Low Source	DEC-08-PMP4-PA
High Memory Binary	DEC-08-PMP2-PB	High Memory Binary	DEC-08-PMP3-PB
High Source	DEC-08-PMP2-PA	High Source	DEC-08-PMP3-PA

3.3 LOADING PROCEDURES

This routine is loaded using the Binary Loader. See Introduction to Programming or Programming Languages for a complete description of the Binary Loader. (This routine cannot be called as a subroutine.)

3.4 USING THE PROGRAM

The SWITCH REGISTER is used to enter the initial and final address of each block of core memory to be punched.

a. Make sure 33-ASR or 75E punch is on.

b. Set the starting address 0041 (or 7441 if using the high-memory version) into the SWITCH REGISTER and press the LOAD ADDRESS key. Next press the START key.

c. The computer halts. Set the initial address of the block to be punched into the SWITCH REGISTER and press the CONTINUE key.

d. The computer halts. Set the final address of the block to be punched into SWITCH REGISTER and press the CONTINUE key.

   Note that the final address must be larger than the initial address.

e. A block of leader (code 200) tape is punched followed by the selected block of data in RIM format.

f. The computer halts. Steps (c) and (d) can now be repeated to punch as many blocks of data as desired. To terminate the tape, proceed as described in (g) below.

g. Set the terminating address 0074 (7474) into the SWITCH REGISTER and press the LOAD ADDRESS key. Next press the START key and a block of trailer tape is punched.

3.5 DETAILS OF OPERATION AND STORAGE

Reference to section 1.7, Flow Chart, will illustrate the following discussion.

After entry, a short subroutine is entered to punch a block of leader. Next the initial address is picked up and the six most significant bits are rotated right, masked out, added to 0100 (in order to punch channel 7), and punched. The least-significant six bits of the address are next masked out and punched.

A similar process is followed to punch the data associated with the corresponding address except 0100 is not added before the first character is punched.

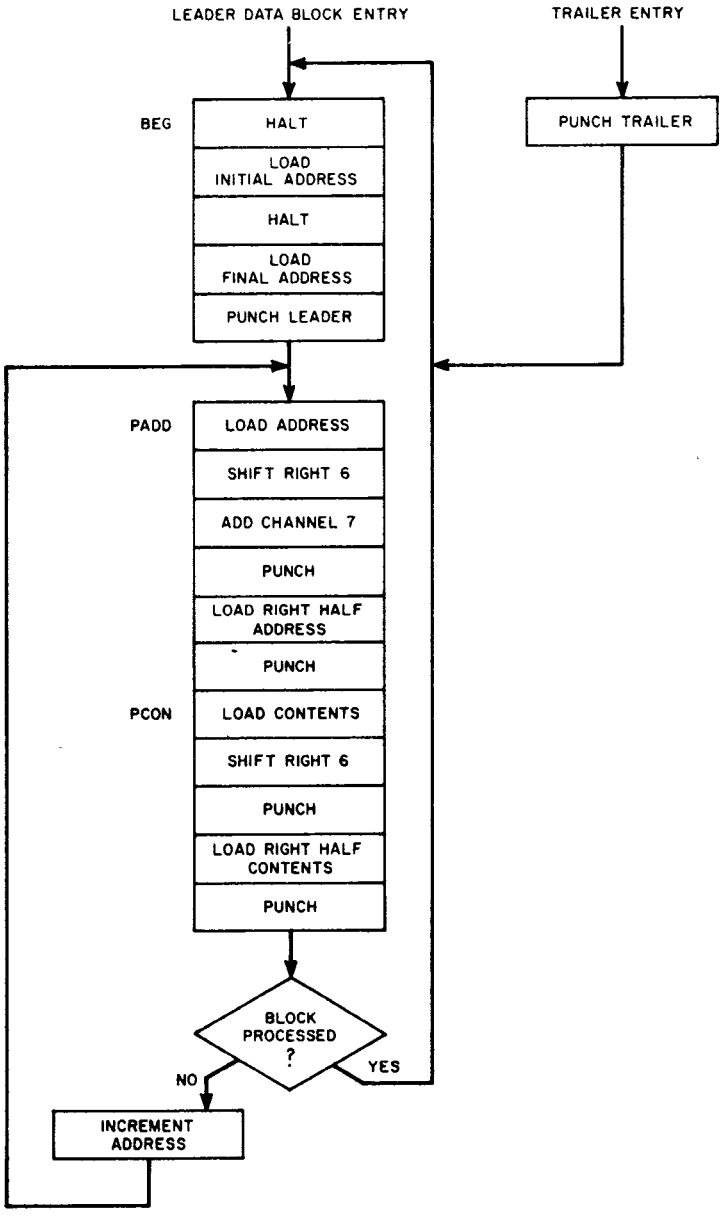
This process is repeated until the final address is reached; then the computer halts at the starting address. If more blocks of data are to be punched, this is done as explained in step (f) above.

The routine is entered at a different address to punch the final trailer.

3.6 EXTERNAL DATA

See Chapter 4 of Introduction to Programming for a description of RIM paper tape format.

3.7 FLOW CHART



3.8 PROGRAM LISTING

```
                /COPYRIGHT 1971 DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /DEC-08-PMP2-PA
                /RIM PUNCH 33-ASR HIGH MEMORY
                *7441
7441   7402     BEG,    HLT             /ENTRY FOR LEADER DATA BLOCK
7442   7604             LAS             /SET INITIAL ADDRESS
7443   3322             DCA IA
7444   7402             HLT
7445   7604             LAS             /SET FINAL ADDRESS
7446   3323             DCA FA
7447   4276             JMS LTS         /GO TO L/T SUBROUTINE
7450   1322     PADD,   TAD IA          /PUNCH ADDRESS
7451   4306             JMS SHFT
7452   1326             TAD CH7
7453   4314             JMS PUN
7454   1322             TAD IA
7455   0325             AND SL6
7456   4314             JMS PUN
7457   1722     PCON,   TAD I IA        /PUNCH CONTENTS
7460   4306             JMS SHFT
7461   4314             JMS PUN
7462   1722             TAD I IA
7463   0325             AND SL6

7464   4314             JMS PUN
7465   1322             TAD IA
7466   7041             CIA
7467   1323             TAD FA
7470   7650             SNA CLA         /TEST FOR END
7471   5275             JMP .+4
7472   2322             ISZ IA
7473   5250             JMP PADD
7474   4276             JMS LTS         /ENTRY FOR L/T
7475   5241             JMP BEG
7476   0000     LTS,    0               /L/T SUBROUTINE
7477   1327             TAD M101
7500   3324             DCA CTR
7501   1330     MORE,   TAD C200
7502   4314             JMS PUN
7503   2324             ISZ CTR
7504   5301             JMP MORE        /MORE L-T CODES
7505   5676             JMP I LTS
7506   0000     SHFT,   0               /SHIFT RIGHT
7507   7012             RTR
7510   7012             RTR
7511   7012             RTR
7512   0325             AND SL6
7513   5706             JMP I SHFT
```

```
7514  0000  PUN,    0                 /PUNCH SUBROUTINE
7515  6046          TLS
7516  6041          TSF
7517  5316          JMP .-1
7520  7200          CLA
7521  5714          JMP I PUN
7522  0000  IA,     0
7523  0000  FA,     0
7524  0000  CTR,    0
7525  0077  SL6,    77
7526  0100  CH7,    100
7527  7677  M101,   -101
7530  0200  C200,   200
```

```
              /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
              /MAYNARD, MASSACHUSETTS
              /DIGITAL-8-4-U-RIM
              /RIM PUNCH 33 LOW MEMORY
              *41
0041  7402  BEG,    HLT               /ENTRY FOR LEADER DATA BLOCK
0042  7604          LAS               /SET INITIAL ADDRESS
0043  3122          DCA IA
0044  7402          HLT
0045  7604          LAS               /SET FINAL ADDRESS
0046  3123          DCA FA
0047  4076          JMS LTS           /GO TO L/T SUBROUTINE
0050  1122  PADD,   TAD IA            /PUNCH ADDRESS
0051  4106          JMS SHFT
0052  1126          TAD CH7
0053  4114          JMS PUN
0054  1122          TAD IA
0055  0125          AND SL6
0056  4114          JMS PUN
0057  1522  PCON,   TAD I IA          /PUNCH CONTENTS
0060  4106          JMS SHFT
0061  4114          JMS PUN
0062  1522          TAD I IA
0063  0125          AND SL6
0064  4114          JMS PUN
0065  1122          TAD IA
0066  7041          CIA
0067  1123          TAD FA
0070  7650          SNA CLA           /TEST FOR END
0071  5075          JMP .+4
0072  2122          ISZ IA
0073  5050          JMP PADD
0074  4076          JMS LTS           /ENTRY FOR L/T
0075  5041          JMP BEG
```

```
0076  0000  LTS,    0                /L/T SUBROUTINE
0077  1127          TAD M101
0100  3124          DCA CTR
0101  1130  MORE,   TAD C200
0102  4114          JMS PUN
0103  2124          ISZ CTR
0104  5101          JMP MORE         /MORE L-T CODES
0105  5476          JMP I LTS
0106  0000  SHFT,   0                /SHIFT RIGHT
0107  7012          RTR
0110  7012          RTR
0111  7012          RTR
0112  0125          AND SL6
0113  5506          JMP I SHFT
0114  0000  PUN,    0                /PUNCH SUBROUTINE
0115  6046          TLS
0116  6041          TSF
0117  5116          JMP .-1
0120  7200          CLA
0121  5514          JMP I PUN
0122  0000  IA,     0
0123  0000  FA,     0
0124  0000  CTR,    0
0125  0077  SL6,    77
0126  0100  CH7,    100
0127  7677  M101,   -101
0130  0200  C200,   200
```

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /DIGITAL-8-4-U-RIM
                /RIM PUNCH 75 HIGH MEMORY
                *7441
7441  7402  BEG,      HLT          /ENTRY FOR LEADER DATA BLOCK
7442  7604             LAS          /SET INITIAL ADDRESS
7443  3322             DCA IA
7444  7402             HLT
7445  7604             LAS          /SET FINAL ADDRESS
7446  3323             DCA FA
7447  4276             JMS LTS      /GO TO L/T SUBROUTINE
7450  1322  PADD,     TAD IA       /PUNCH ADDRESS
7451  4306             JMS SHFT
7452  1326             TAD CH7
7453  4314             JMS PUN
7454  1322             TAD IA
7455  0325             AND SL6
7456  4314             JMS PUN
7457  1722  PCON,     TAD I IA     /PUNCH CONTENTS
7460  4306             JMS SHFT
7461  4314             JMS PUN
7462  1722             TAD I IA
7463  0325             AND SL6
7464  4314             JMS PUN
```

```
7465  1322            TAD IA
7466  7041            CIA
7467  1323            TAD FA
7470  7650            SNA CLA     /TEST FOR END
7471  5275            JMP .+4
7472  2322            ISZ IA
7473  5250            JMP PADD
7474  4276            JMS LTS     /ENTRY FOR L/T
7475  5241            JMP BEG
7476  0000  LTS,       0          /L/T SUBROUTINE
7477  1327            TAD M101
7500  3324            DCA CTR
7501  1330  MORE,      TAD C200
7502  4314            JMS PUN
7503  2324            ISZ CTR
7504  5301            JMP MORE    /MORE L-T CODES
7505  5676            JMP I LTS
7506  0000  SHFT,      0          /SHIFT RIGHT
7507  7012            RTR
7510  7012            RTR
7511  7012            RTR
7512  0325            AND SL6
7513  5706            JMP I SHFT
7514  0000  PUN,       0          /PUNCH SUBROUTINE
7515  6026            PLS
7516  6021            PSF
7517  5316            JMP .-1
7520  7200            CLA
7521  5714            JMP I PUN
7522  0000  IA,        0
7523  0000  FA,        0
7524  0000  CTR,       0
7525  0077  SL6,       77
7526  0100  CH7,       100
7527  7677  M101,      -101
7530  0200  C200,      200
```

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /DIGITAL-8-4-U-RIM
                /RIM PUNCH 75 LOW MEMORY
                *41
0041  7402  BEG,      HLT             /ENTRY FOR LEADER DATA BLOCK
0042  7604            LAS             /SET INITIAL ADDRESS
0043  3122            DCA IA
0044  7402            HLT
0045  7604            LAS             /SET FINAL ADDRESS
0046  3123            DCA FA
0047  4076            JMS LTS         /GO TO L/T SUBROUTINE
0050  1122  PADD,     TAD IA          /PUNCH ADDRESS
0051  4106            JMS SHFT
0052  1126            TAD CH7
```

```
0053  4114          JMS PUN
0054  1122          TAD IA
0055  0125          AND SL6
0056  4114          JMS PUN
0057  1522  PCON,   TAD I IA        /PUNCH CONTENTS
0060  4106          JMS SHFT
0061  4114          JMS PUN
0062  1522          TAD I IA
0063  0125          AND SL6
0064  4114          JMS PUN
0065  1122          TAD IA
0066  7041          CIA
0067  1123          TAD FA
0070  7650          SNA CLA         /TEST FOR END
0071  5075          JMP .+4
0072  2122          ISZ IA
0073  5050          JMP PADD
0074  4076          JMS LTS         /ENTRY FOR L/T
0075  5041          JMP BEG
0076  0000  LTS,     0               /L/T SUBROUTINE
0077  1127          TAD M101
0100  3124          DCA CTR
0101  1130  MORE,    TAD C200
0102  4114          JMS PUN
0103  2124          ISZ CTR
0104  5101          JMP MORE        /MORE L/T CODES
0105  5476          JMP I LTS
0106  0000  SHFT,    0               /SHIFT RIGHT
0107  7012          RTR
0110  7012          RTR
0111  7012          RTR
0112  0125          AND SL6
0113  5506          JMP I SHFT
0114  0000  PUN,     0               /PUNCH SUBROUTINE
0115  6026          PLS
0116  6021          PSF
0117  5116          JMP .-1
0120  7200          CLA
0121  5514          JMP I PUN
0122  0000  IA,      0
0123  0000  FA,      0
0124  0000  CTR,     0
0125  0077  SL6,     77
0126  0100  CH7,     100
0127  7677  M101,    -101
0130  0200  C200,    200
```

CHAPTER 4

BINARY PUNCH

(Binary Core Dump to High-speed or Teletype Punch)

4.1 ABSTRACT

This program provides a means of punching information contained in selected blocks of core memory as binary-coded paper tape using the high-speed or Teletype punch.

4.2 REQUIREMENTS

This program occupies 75_{10} (113_8) core memory locations.

The Binary Punch program runs on the basic PDP-8, 8/S, 8/I, 8/L, or 8/E with standard 33-ASR Teletype or standard high-speed punch.

Program tapes are as follows:

33-ASR Binary	DEC-08-YX1A-PB
33-ASR Source	DEC-08-YX1A-PA
High-speed Binary	DEC-08-YX2A-PB
High-speed Source	DEC-08-YX2A-PA

4.3 LOADING PROCEDURES

This program is loaded by means of the Binary Loader. See Introduction to Programming or Programming Languages for a complete discussson of the Binary Loader and its use. (This program cannot be called as a subroutine.)

The SWITCH REGISTER is used to enter initial and final addresses of blocks to be punched as well as the number of blocks to be punched.

This program is used in the following manner:

a. Assuming the program is in memory as listed in 4.7, place the starting address 7465 in the SWITCH REGISTER and press the LOAD ADDRESS key.

b. Press the START key. Leader tape is punched and the computer halts. Set the number of blocks to be punched into the SWITCH REGISTER and press the CONTINUE key.

c. The computer halts. Set the initial address of the block to be punched into the SWITCH REGISTER and press the CONTINUE key.

d. The computer halts. Set the final address of the block to be punched into the SWITCH REGISTER and press the CONTINUE key.

e. Note that the final address must be greater than the initial address.

f. The indicated block of data is punched. If only one block has been called for, the trailer tape is punched and the computer halts. If more than one block has been called for, the computer halts at step (a) waiting for a new initial address. The second block is punched following completion of steps (c) and (d), etc.

4.4 METHOD

This is a basic program used to produce tapes acceptable by the Binary Loader.

With each punched block of data, an initial address (into which that data is to be loaded) is punched as the first two characters. Following the initial address, each 2-character group represents the binary contents of a computer word. At the end of each block, a 2-character checksum is punched.

Reference to Section 4.6, Flow Chart, will illustrate the computational approach. Basically data is picked up from memory, the most significant half shifted right and punched, and the least significant half masked out and punched.

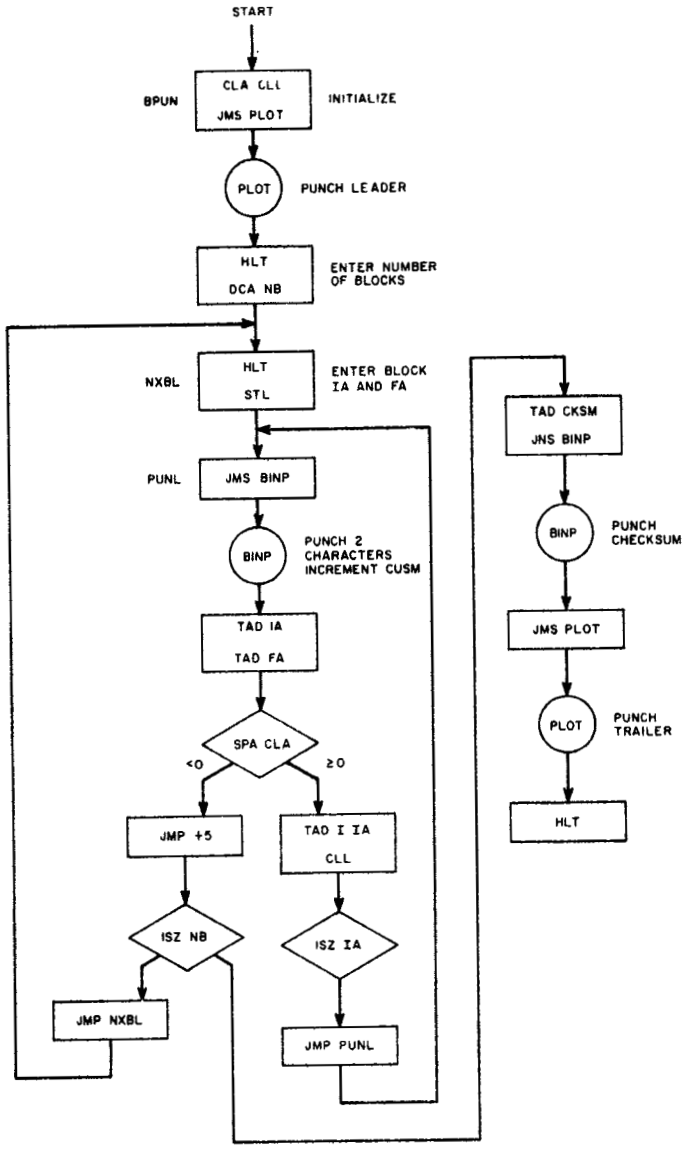
A similar process is followed with respect to the initial address and the checksum, which is accumulated character by character as a block, is punched.

4.5 EXTERNAL DATA

See Chapter 4 of Introduction to Programming for a complete discussion of tape format.

4.6 FLOW CHART

Note that in this diagram circles represent subroutine utilization, not connectives.



4.7 PROGRAM LISTING

A listing of this program with BPUN located at 7465 is as follows:

```
                    /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                    /MAYNARD, MASSACHUSETTS
                    /BIN PUNCH HIGH SPEED PUNCH (PC03, PP8I, PP8L)
                    *7465
7465    7300    BPUN,   CLA CLL
7466    6026            PLS             /INITIAL PUNCH
7467    3366            DCA CKSM        /CLEAR CHECK-SUM
7470    4330            JMS PLOT        /GO PUNCH LEADER CODES
7471    7402            HLT             /SET SWITCHES=NUMBER OF
                                         BLOCKS
7472    7604            LAS
7473    7041            CIA
7474    3367            DCA NB          /STOKE MINUS NUMBER OF
                                         BLOCKS
7475    7402    NXBL,   HLT             /SET SWITCHES=INITIAL ADDRESS
                                         OF BLOCK
7476    7604            LAS
7477    3370            DCA IA
7500    7402            HLT             /SET SWITCHES=FINAL ADDRESS
                                         OF BLOCK
7501    7604            LAS
7502    7001            IAC
7503    3371            DCA FA
7504    1370            TAD IA
7505    7120            STL             /TO PUNCH IA AS ORIGIN
7506    4341    PUNL,   JMS BINP        /GO PUNCH WORD AS TWO LINES
                                         OF TAPE
7507    1370            TAD IA
7510    7041            CIA
7511    1371            TAD FA          /AC=FA-IA
7512    7650            SNA CLA         /WAS IT LAST WORD OF BLOCK?
7513    5320            JMP .+5         /IT WAS THE LAST WORD
7514    1770            TAD IIA         /GET WORD TO PUNCH
7515    7100            CLL             /NOT AN ORIGIN
7516    2370            ISZ IA          /JUST INDEX IA
7517    5306            JMP PUNL
7520    2367            ISZ NB          /IS THERE ANOTHER BLOCK?
7521    5275            JMP NXBL        /HANDLE NEXT BLOCK
7522    1366            TAD CKSM
7523    7100            CLL
7524    4341            JMS BINP        /GO PUNCH CHECK SUM
7525    4330            JMS PLOT        /GO PUNCH TRAILER CODES
7526    7402            HLT             /DONE
7527    5265            JMP BPUN
7530    0000     PLOT,  0
7531    7300            CLA CLL
7532    1372            TAD M212        /TO PUNCH 212 OCTAL LEADER
                                        TRAILER CODES
```

```
7533    3373            DCA CTR1
7534    1374            TAD C200        /LEADER TRAILER CODE
7535    4361            JMS PUN         /PUNCH C (AC)
7536    2373            ISZ CTR1        /ANOTHER L-T CODE OR NOT?
7537    5335            JMP .-2         /GO PUNCH ANOTHER
7540    5730            JMP I PLOT      /EXIT
7541    0000    BINP,   0
7542    3375            DCA TEM1
7543    1375            TAD TEM1
7544    7012            RTR
7545    7012            RTR
7546    7012            RTR
7547    0376            AND SL 7        /FIRST TWO OCTAL DIGITS IN
                                        AC 5-11
7550    4361            JMS PUN         /PUNCH C (AC)
7551    1366            TAD CKSM
7552    3366            DCA CKSM
7553    1375            TAD TEM1
7554    0377            AND SL6         /LAST TWO OCTAL DIGITS IN
                                        AC 6-11
7555    4361            JMS PUN         /PUNCH C (AC)
7556    1366            TAD CKSM
7557    3366            DCA CKSM
7560    5741            JMP I BINP      /EXIT
7561    0000    PUN,    0               /ROUTINE TO PUNCH C (AC)
7562    6021            PSF             /AND EXIT WITH C (AC)
7563    5362            JMP .-1         /UNALTERED
7564    6026            PLS             /PUNCH IT
7565    5761            JMP I PUN       /EXIT
7566    0000    CKSM,   0
7567    0000    NB,     0
7570    0000    IA,     0
7571    0000    FA,     0
7572    7566    M212,   -212
7573    0000    CTR1,   0
7574    0200    C200,   200
7575    0000    TEM1,   0
7576    0177    SL7,    177
7577    0077    SL6,    77
```

4.8 33-ASR TELETYPE PUNCH PROGRAM

To use this program with the 33-ASR Teletype, make the following changes:

```
7466    6046            TLS     /INITIAL PUNCH
7562    6041            TSF     /AND EXIT WITH C (AC)
7564    6046            TLS     /PUNCH IT
```

CHAPTER 5
OCTAL MEMORY DUMP
(Octal Core Dump to Paper Tape)

5.1 ABSTRACT

This program enables the user to dump, in octal, any or all data in any memory field to either the Teletype or high-speed paper tape punch. During dumping the absolute address of each location being dumped is held in the accumulator. When dumping is completed output devices and memory fields can be changed to dump another section of memory.

5.2 REQUIREMENTS

This program requires one core page; initially 7400-7577.

The Octal Memory Dump program runs on any PDP-8 family computer with at least 4K words of core, a 33-ASR Teletype and/or high-speed paper tape punch.

No additional software is required. The program leaves the BIN and RIM Loaders untouched. The program tapes are as follows:

Binary	DEC-08-YPPA-PB
Source	DEC-08-YPPA-PA

5.3 USAGE

The program is supplied in ASCII format on punched paper tape, and can be assembled by any 4K PDP-8 assembler (i.e., PAL III, MACRO-8, or PAL-D). The origin of this program (7400) can be changed with the PDP-8 Symbolic Editor in order to dump locations 7400-7577. (See the appropriate assembler section of Programming Languages for assembly instructions.)

5.3.1 Loading

The program is loaded into core with the Binary Loader (see Introduction to Programming or Programming Languages for loading procedures) and can be loaded into any available memory field.

5.3.2 Operating Procedures

The SWITCH REGISTER on the PDP-8 console is used to control the program; all options are determined by the position of bit 0. The program can be interrupted by depressing the STOP switch.

With Octal Memory Dump program in core:

a. Set the SWITCH REGISTER to the starting address (7400) and the INSTRUCTION FIELD to the field containing the Octal Dump. Set DATA FIELD to the field containing the code to be dumped. Press the LOAD ADDRESS key.

b. Set SWITCH REGISTER bit Ø to 1 for a core dump to the Teletype punch, or to Ø when dumping via the high-speed paper tape punch.

c. Press the START switch. The computer halts.

d. Set the SWITCH REGISTER to the starting address of the section of core to be dumped.

e. Press the CONTINUE switch. The computer halts.

f. Set the SWITCH REGISTER to the final core address of the section of core to be dumped.

g. Press the CONTINUE switch; dumping commences and stops after dumping the contents of the final core address specified in step (f) above.

Another dump can be performed at this time by continuing at step (a) when the output device or data field is to change. Otherwise, continue at step (d).

The program halts after each dump.

The preceding operations are illustrated in Figure 5-1.

5.4 INPUT/OUTPUT

The program contains its own Teletype and high-speed punch output, and there are no external I/O handlers used. SWITCH REGISTER bit 0 determines the output device.

5.5 FUNCTIONAL DESCRIPTION

The program is written in the PAL III language. Four routines are used in the program:

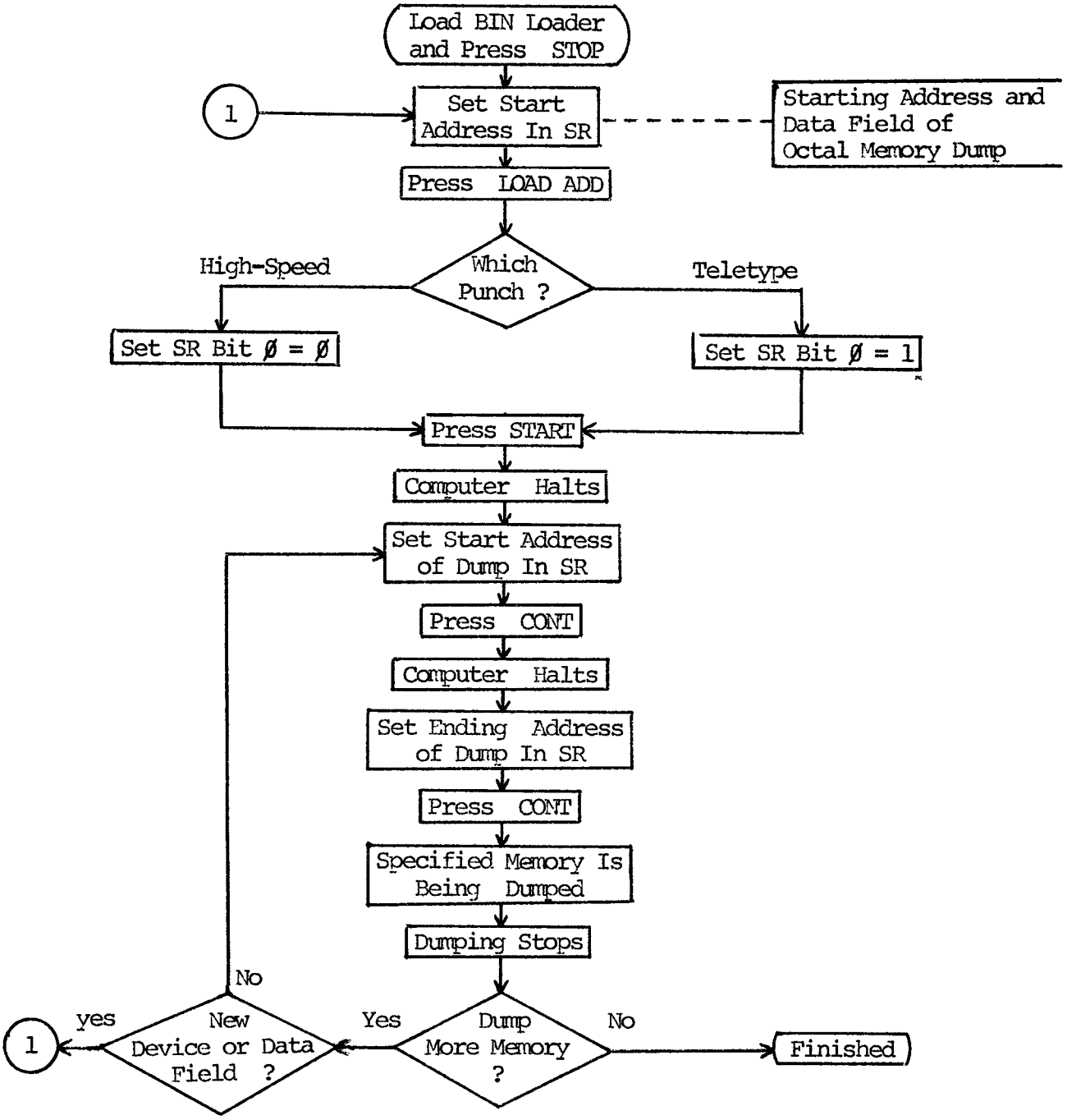


Figure 5-1 Operating Procedures

a. The TOCT routine causes a number to be formatted for a typeout or punchout.

b. The TCR routine outputs a carriage return-line feed.

c. The TSP routine outputs a space.

d. The TCHAR routine is the output routine for both the Teletype and the high-speed punch.

The main routine begins with the initialization of variables, and the two address arguments are picked up from the switch register. Two carriage return-line feeds are performed, followed by the starting address and several spaces. A loop is then entered to type the contents of eight memory locations (if eight remain). If more data remains to be output, a JMP to LPØ2 repeats the process. If during this loop the routine finds that it has processed the last memory location, the loop exits, a carriage return line feed is performed, a JMP to LPØØ is executed, and the program halts.

See the program listing that follows for more precise information.

5.6 PROGRAM LISTING

```
                / OCTAL MEMORY DUMP PROGRAM

                / COPYRIGHT 1969
                / DIGITAL EQUIPMENT CORP.
                / MAYNARD, MASS.
                /TO OPERATE:
                /       LOAD ADDRESS 7400 IN SR
                /       TO CHOSE OUTPUT DEVICE:
                /       SET BIT 0=0 FOR H. S. PUNCH OUTPUT OR
                /       SET BIT 0=1 FOR TTY OUTPUT THEN PRESS START
                /       SET STARTING ADDRESS AND DATA FIELD IN SR -PRESS C
ONTINUE
                /       SET ENDING ADDRESS AND DATA FIELD IN SR -PRESS CON
TINUE

                        *7400
7400   7604   DUMP,    CLA OSR         /EXAMINE SR FOR OUTPUT DEVICE
7401   7700            SMA CLA
7402   1265            TAD C10
7403   1270            TAD C7400
7404   3325            DCA SKPZ        /STORE A "SKP" IN SKPZ IF H. S. PU
NCH OUTPUT
7405   7402   LP00,    HLT             /STOP. ENTER DUMP STARTING ADDRESS
```

```
7406  7604          LAS
7407  3261          DCA ADDR
7410  7402          HLT               /STOP. ENTER DUMP ENDING ADDRESS
7411  7604          LAS
7412  7040          CMA
7413  1261          TAD ADDR
7414  3262          DCA INDEX         /COUNTER FOR NUM OF LOCS TO BE DUM
PED
7415  4312          JMS TCR           /TYPE CR-LFS
7416  4312  LP01,   JMS TCR
7417  1261          TAD ADDR
7420  4272          JMS TOCT          /OUTPUT STARTING ADDRESS IN OCTAL
7421  4320          JMS TSP           /OUTPUT 3 SPACES
7422  4320          JMS TSP
7423  4320  LP02,   JMS TSP
7424  1661          TAD I ADDR        /GET CONTENTS OF LOC
7425  4272          JMS TOCT          /TYPE OUT CONTENTS
7426  2262          ISZ INDEX         /DONE DUMPING?
7427  7410          SKP
7430  5247          JMP OUT           /YES. EXIT
7431  2261          ISZ ADDR          /NO. KICK ADDRESS UP
7432  1261          TAD ADDR          /HAVE WE OUTPUT 8 LOCS ON A LINE?
7433  0263          AND C3
7434  7640          SZA CLA
7435  5223          JMP LP02          /NO. SPACE OVER ONE AND GET NEXT
7436  1261          TAD ADDR
7437  0264          AND C7
7440  7640          SZA CLA
7441  5222          JMP LP02-1
7442  1261          TAD ADDR
7443  0266          AND C177
7444  7640          SZA CLA
7445  5216          JMP LP01          /OUTPUT CR/LF THEN NEW ADDRESS
7446  5215          JMP LP01-1
7447  4312  OUT,    JMS TCR           /OUTPUT CR/LF
7450  1267          TAD C214
7451  4324          JMS TCHAR         /OUTPUT A FORM FEED
7452  1271          TAD M20           /THEN OUTPUT 20 BLANKS OF TRAILER
7453  3262          DCA INDEX
7454  4324          JMS TCHAR
7455  2262          ISZ INDEX
7456  5254          JMP .-2
7457  1261          TAD ADDR          /LEAVE WITH FINAL ADDRESS IN AC
7460  5205          JMP LP00          /GO TO HALT FOR POSSIBLE RESTART

              / VARIABLES AND CONSTANTS

7461  0000  ADDR,   0                 /LOC OF STARTING ADDRESS TO BE DUM
PED
7462  0000  INDEX,  0                 /COUNTER FOR NUMBER OF LOCS TO BE
DUMPED
```

```
7463  0003  C3,      3                 /MASK VALUES
7464  0007  C7,      7
7465  0010  C10,     10
7466  0177  C177,    177
7467  0214  C214,    214               /FORM FEED
7470  7400  C7400,   7400              /USED TO FORM SKP COMMAND
7471  7760  M20,     -20               /COUNTER FOR NUM OF BLANKS TO OUTP
UT

              / OCTAL TYPEOUT ROUTINE

7472  0000  TOCT,    0
7473  7104            CLL RAL /ROTATE ADDRESS 1 LEFT
7474  3344            DCA WORD
7475  1352            TAD M4            /SET NUMBER OF DIGITS PER WORD
7476  3345            DCA NDX
7477  1344  LP03,     TAD WORD          /ROTATE WORD 3 LEFT
7500  7006            RTL
7501  7004            RAL
7502  3344            DCA WORD
7503  1344            TAD WORD
7504  0264            AND C7            /MASK BITS 9-11
7505  1351            TAD C260          /ADD 260 FOR OUTPUT
7506  4324            JMS TCHAR         /OUTPUT DIGIT
7507  2345            ISZ NDX           /DONE FOUR?
7510  5277            JMP LP03          /NO. PICK UP ANOTHER DIGIT
7511  5672            JMP I TOCT        /YES. RETURN

              / ROUTINE TO OUTPUT A CARRIAGE RETURN/LINE FEED

7512  0000  TCR,     0
7513  1347            TAD C215          /OUTPUT A C. R.
7514  4324            JMS TCHAR
7515  1346            TAD C212          /OUTPUT A L. F.
7516  4324            JMS TCHAR
7517  5712            JMP I TCR

              / ROUTINE TO OUTPUT A SPACE

7520  0000  TSP,     0
7521  1350            TAD C240          /OUTPUT A SPACE
7522  4324            JMS TCHAR
7523  5720            JMP I TSP
```

```
                / ROUTINE TO OUTPUT A CHARACTER ON TTY OR H. S. PUNCH

7524  0000  TCHAR,  0
7525  7000  SKPZ,   NOP             /CHANGED TO A "SKP" IF H. S. OUTPU
T
7526  5335          JMP TCH1        /OTHERWISE GO TO TTY  OUTPUT
7527  6026          PLS
7530  7200          CLA
7531  1261          TAD ADDR        /KEEP ADDRESS IN AC WHILE PUNCHING

7532  6021          PSF
7533  5332          JMP .-1
7534  5342          JMP TCH2
7535  6046  TCH1,   TLS             /TTY OUTPUT ROUTINE
7536  7200          CLA
7537  1261          TAD ADDR
7540  6041          TSF
7541  5340          JMP .-1
7542  7200  TCH2,   CLA
7543  5724          JMP I TCHAR

                / VARIABLES AND CONSTANTS

7544  0000  WORD,   0               /STORAGE FOR DIGIT TO BE FORMATTED

7545  0000  NDX,    0               /COUNTER FOR NUM OF DIGITS OUTPUT
7546  0212  C212,   212             /CODE FOR LINE FEED
7547  0215  C215,   215             /  "  "  CARRIAGE RETURN
7550  0240  C240,   240             /  "  "  SPACE
7551  0260  C260,   260             /  "  "  FORMATTING DIGITS
7552  7774  M4,     -4              /NUMBER OF DIGITS PER WORD
```

CHAPTER 6
TELETYPE I/O SUBROUTINES

6.1 ABSTRACT

The routines described in this chapter are illustrative of the procedures to be followed in creating I/O routines to be used with the ASR-33 Teletype. The user is advised to peruse these routines prior to writing I/O routines tailored for his particular needs. Subroutines are provided which perform input and output of character strings and single alphanumeric characters. These routines are illustrative and by no means exhaustive of routines to handle the ASR-33 Teletype.

6.2 REQUIREMENTS

The routines as supplied require 124_{10} (174_8) core locations.

The Teletype I/O Subroutines run on any PDP-8 family computer with an ASR-33 Teletype console. The program is distributed as an ASCII tape as follows:

DEC-08-FIKA-PA

6.3 USAGE

6.3.1 Assembly

The routines as supplied will be automatically assembled onto the first available core page. There is no $ character at the end of the tape; although a PAUSE statement is present to allow for the later loading of additional programs, if any.

This collection of subroutines can be assembled with PAL III, PAL-D, PAL8, or MACRO-8. (If using the tape with TSS/8, remove the PAUSE statement from the end of the tape.)

If the routines are to be assembled separately (without a user program), they will assemble at location 200. In this case, the

user should append a separate tape with a $ character to the end of the Teletype I/O Subroutines tape before assembly.

If the routines are to be assembled with a user program (such as the example program in section 6.6), the Teletype I/O Subroutines tape should be loaded after the user program and the whole followed by a tape with a $ character. In this case, the I/O subroutines will fit on the first available free core page and will not overlay the user program. (Tapes can be loaded after the I/O routines if allowance is made for the length of the routines or by assigning a specific address to the beginning of the I/O routines.)

6.3.2 Calling Sequence

The calling sequence for the Subroutines is designed so that the user can easily incorporate messages into his program. The user inserts a series of JMS instructions to the I/O routines followed by the address of the message to be transmitted.

The subroutines the user is likely to call are as follows:

Subroutine	Purpose	Calling Sequence
TYPX	Prints a message. Message is coded by programmer if using PAL III, or set in a labeled TEXT pseudo-op statement if using PAL-D, PAL8, or MACRO-8.	ENTRY, JMS TYPX POINTER /ADDR OF /MESSAGE /EXIT IS TO /ENTRY+2 WITH /AC CLEAR
TLSX	Prints a single character, used by TYPX to print message, character by character. Subroutine is entered with character to be printed in AC bits 6 to 11.	JMS TLSX /CHAR IN AC /BITS 6 TO 11
KREAD	Inputs a message. KREAD accepts 8-bit ASCII and does not convert to 6-bit ASCII, therefore TYPX cannot directly cause a message read by KREAD to be printed, although a simple routine to do so can be written by the user. Remember, the size of the input buffer must include the carriage return character.	ENTRY, JMS KREAD POINTER /ADDR OF /INPUT BUFFER -LENGTH /-SIZE OF /INPUT BUFFER /EXIT IS TO ENTRY+3 /WITH AC CLEAR /ZERO PLACED AT /BUFFER END

Subroutine	Purpose	Calling Sequence
KRBX	Inputs single character from Teletype, exits with character read in the AC.	ENTRY, JMS KRBX /EXIT IS TO /ENTRY+1 WITH /CHAR READ IN AC

Other routines provided include KRUB, which deletes the last character in the input buffer when the RUBOUT key is typed. This routine is not necessary to Teletype I/O but allows for erasing of typing mistakes and also performs echoing. If the user rubs out characters past the beginning of the input buffer, a carriage return/line feed is performed as a warning.

6.4 RESTRICTIONS

The user program must initialize the teleprinter flag before calling these subroutines. Initialization is performed as follows:

```
CLA     /SET AC TO ZERO, GOOD PRACTICE
TLS     /INITIALIZE TELETYPE
```

The routines can only be called from the memory field in which they reside.

The following characters have special meaning to the output subroutines:

6-bit value	Character	Purpose
00	@	Marks the end of a message; supplied via the TEXT pseudo-op when using PAL-D, PAL8, or MACRO-8. Must be supplied by user if using PAL III.
37	←	Causes a carriage return-line feed to be output to the Teletype.

The following characters have special meaning to the input subroutine:

8-bit value	Character	Purpose
212	LINE FEED	Ignored on input.
000	null	Ignored on input.
377	RUBOUT	Deletes the previous character typed, echoes a backslash after the first RUBOUT typed and a closing back-slash after the first non-RUBOUT character typed.
215	RUBOUT	Echoes a carriage return/line feed and exits from the input subroutine to the user program (calling program).

On input buffer overflow while reading characters into the Teletype buffer, characters echo as "bell" (the bell within the Teletype rings). All characters other than RETURN and RUBOUT are lost if typed while "bell" is being rung as a warning.

6.5 DESCRIPTION

Table 6.1 shows the ASCII values of the characters which can be used with the supplied Teletype I/O Subroutines. When using PAL III, the user must code these ASCII values at the location referenced by the output routine. The PAL III assembler reads the ASCII codes directly as octal numbers.

PAL-D, PAL8, and MACRO-8 can use the TEXT pseudo-op to directly format alphanumeric characters into ASCII code. The TEXT pseudo-op puts the desired message into 6-bit ASCII format; the routines convert the 6-bit ASCII to the printable 8-bit ASCII and then output the message.

If the user were outputting the message HELLO, it would be done as follows, where MESG is the location given for the message:

PAL III

```
MESG,   1ØØ5    /HE
        1414    /LL
        17ØØ    /O END OF
                /MESSAGE CODE
```

PAL-D, MACRO-8, or PAL8

```
MESG, TEXT/HELLO/
```

TABLE 6.1

6-BIT ASCII CHARACTER SET FOR INPUT WHEN USING THE PAL III ASSEMBLER, 8-BIT ASCII FORMAT IS ALSO SHOWN FOR COMPLETENESS.

Character	6-BIT Value	8-BIT Value	Character	6-BIT Value	8-BIT Value
A	01	301	!	41	241
B	02	302	"	42	242
C	03	303	#	43	243
D	04	304	$	44	244
E	05	305	%	45	245
F	06	306	&	46	246
G	07	307	'	47	247
H	10	310	(	50	250
I	11	311	)	51	251
J	12	312	*	52	252
K	13	313	+	53	253
L	14	314	,	54	254
M	15	315	-	55	255
N	16	316	.	56	256
O	17	317	/	57	257
P	20	320	:	72	272
Q	21	321	;	73	273
R	22	322	<	74	274
S	23	323	=	75	275
T	24	324	>	76	276
U	25	325	?	77	277
V	25	326	@	00	300
W	27	327	[	33	333
X	30	330		34	334
Y	31	331	]	35	335
Z	32	332	↑	36	336
0	60	260	←	37	337
1	61	261	leader tape		200
2	62	262	LINE FEED		212
3	63	263	RETURN		215
4	64	264	SPACE	40	240
5	65	265	RUBOUT		377
6	66	266	blank		000
7	67	267	BELL		207
8	70	270	TAB		211
9	71	271	FORM		214

The same results are achieved in both cases. The TEXT pseudo-op performs all of the necessary operations in the case of PAL-D, PAL8, and MACRO-8 which the programmer must perform if using PAL III.

The input routines echo characters typed at the keyboard, accept 8-bit ASCII characters (as they come from the Teletype keyboard) for internal storage, and allow character editing via the RUBOUT key. Once the RETURN key is typed, no further input is accepted from the keyboard until the next time such input is requested by the calling program.

Additional information on these routines and their usage can be found by reading through the listing (section 6.7) and by reading Chapter 5 in Introduction to Programming, 1970.

6.6 EXAMPLE PROGRAM

The following example program was assembled with the Teletype I/O Subroutines as shown in section 6.7 (PROGRAM LISTING). The demonstration program was loaded prior to the I/O routines. If no program had preceded the I/O routines they would, of course, have started at location 200 (instead of location 400). The output of the demonstration program is shown below:

```
PLEASE TYPE YOUR NAME
HERMAN

IT IS A PLEASURE TO MEET YOU, HERMAN
```

The computer causes

```
PLEASE TYPE YOUR NAME
```

and a carriage return/line feed to be output. The user types his name on the keyboard, enters it with the RETURN key, and the computer then prints a carriage return/line feed followed by

```
IT IS A PLEASURE TO MEET YOU, HERMAN
```

The program as input to the Assembler (PAL-D in this case), looks as follows:

```
/DEMONSTRATION OF TTY I/O SUBROUTINES
/
*200                    /ADDRESS OF START OF PROGRAM
START,  TLS             /INITIALIZE TELEPRINTER FLAG
        JMS TYPX        /PRINT A MESSAGE
        MESG1           /"PLEASE TYPE YOUR NAME"
        JMS KREAD       /READ IN REPLY
        INAREA          /UP TO 16 CHAR, 1 PER WORD
        -20             /BEGINNING IN INAREA
        JMS TYPX        /PRINT SECOND MESSAGE
        MESG2           /"IT IS A PLEASURE TO MEET YOU, "
        TAD (INAREA-1   /SET AUTOINDEX REGISTER 10 TO
        DCA 10          /ADDRESS WHERE NAME IS STORED(-1)
LOOP,   CLA CLL         /BEGIN PRINTING THE NAME
        TAD I 10        /GET A CHARACTER
        SNA             /TEST FOR ZERO
        JMP DONE        /IF ZERO, ALL DONE--QUIT
        JMS TLSX        /ELSE PRINT A CHARACTER
        JMP LOOP
DONE,   HLT
        JMP START       /IT IS A GOOD PRACTICE TO PUT
                        /A JMP TO SOMEWHERE AFTER THE
                        /HLT AT THE END OF A PROGRAM
                        /IN CASE SOMEONE INADVERTANTLY
                        /HITS "CONTINUE".
MESG1,  TEXT /←PLEASE TYPE YOUR NAME←/
MESG2,  TEXT /←IT IS A PLEASURE TO MEET YOU, /
INAREA, 0               /INPUT AREA FOR NAME
```

The demonstration program assembled with PAL-D looks as follows:

```
                /DEMONSTRATION OF TTY I/O SUBROUTINES
                /
                *200                    /ADDRESS OF START OF PROGRAM
0200  6046    START,  TLS               /INITIALIZE TELEPRINTER FLAG
0201  4777            JMS TYPX          /PRINT A MESSAGE
0202  0222            MESG1             /"PLEASE TYPE YOUR NAME"
0203  4776            JMS KREAD         /READ IN REPLY
0204  0256            INAREA            /UP TO 16 CHAR, 1 PER WORD
0205  7760            -20               /BEGINNING IN INAREA
0206  4777            JMS TYPX          /PRINT SECOND MESSAGE
0207  0236            MESG2             /"IT IS A PLEASURE TO MEET YOU, "
0210  1375            TAD (INAREA-1     /SET AUTOINDEX REGISTER 10 TO
```

```
0211  3010          DCA 10          /ADDRESS WHERE NAME IS STORED(-1)
0212  7300  LOOP,   CLA CLL         /BEGIN PRINTING THE NAME
0213  1410          TAD I 10        /GET A CHARACTER
0214  7450          SNA             /TEST FOR ZERO
0215  5220          JMP DONE        /IF ZERO, ALL DONE--QUIT
0216  4774          JMS TLSX        /ELSE PRINT A CHARACTER
0217  5212          JMP LOOP
0220  7402  DONE,   HLT
0221  5200          JMP START       /IT IS A GOOD PRACTICE TO PUT
                                    /A JMP TO SOMEWHERE AFTER THE
                                    /HLT AT THE END OF A PROGRAM
                                    /IN CASE SOMEONE INADVERTANTLY
                                    /HITS "CONTINUE".
0222  3720  MESG1,  TEXT /←P
0223  1405  LE
0224  0123  AS
0225  0540  E
0226  2431  TY
0227  2005  PE
0230  4031   Y
0231  1725  OU
0232  2240  R
0233  1601  NA
0234  1505  ME
0235  3700  ←/

0236  3711  MESG2,  TEXT /←I
0237  2440  T
0240  1123  IS
0241  4001   A
0242  4020   P
0243  1405  LE
0244  0123  AS
0245  2522  UR
0246  0540  E
0247  2417  TO
0250  4015   M
0251  0505  EE
0252  2440  T
0253  3117  YO
0254  2554  U,
0255  4000   /
0256  0000  INAREA, 0              /INPUT AREA FOR NAME
```

6.7 PROGRAM LISTING

As explained in section 6.6, this listing of the I/O routines was made following the example program shown in that section. These routines can start at any address if the user changes the starting address or will start at the beginning of the first free page after any programs loaded previously.

```
/BASIC TELETYPE I/O SUBROUTINES
/REVISION: 18-JAN-71                /GWB

/COPYRIGHT 1971          DIGITAL EQUIPMENT CORPORATION
/                        MAYNARD, MASSACHUSETTS 01754

/THESE SUBROUTINES ILLUSTRATE TYPICAL METHODS OF USING THE

/STANDARD TELETYPE TO INPUT AND OUTPUT ALPHANUMERIC DATA
/ON A PDP-8 FAMILY COMPUTER.


/               --NOTES--
/       (1) THE USER PROGRAM MUST INITIALIZE THE TELE-
/       PRINTER FLAG BEFORE CALLING THESE SUBROUTINES.

/       (2) THESE ROUTINES MAY ONLY BE CALLED FROM
/       THE FIELD IN WHICH THEY RESIDE.

/ORIGIN TO NEW PAGE



*.-1 177+1



/SUBROUTINE TO TYPE MESSAGES.
/THIS SUBROUTINE PRINTS A MESSAGE TO BE STORED IN STRIPPED

/SIX-BIT ASCII, TWO CHARACTERS PER WORD.  THIS FORMAT

/CAN BE EASILY GENERATED BY USING THE "TEXT" PSEUDO-OP IN
/PROGRAMS ASSEMBLED BY PALD, PAL8, OR MACRO-8.

/ENTRY, JMS     TYPX
/       POINTER                 (ADDRESS OF MESSAGE)
/EXIT IS TO ENTRY+2 WITH THE AC CLEAR.
```

```
                /                --NOTE--
                /THE FOLLOWING CHARACTERS HAVE SPECIAL MEANINGS:
                /VALUE  CHARACTER        COMMENTS
                / 00             @       MARKS THE END OF MESSAGE, THE TEXT

                /                        PSEUDO-OP INSERTS THIS CHARACTER.
                / 37             -       CAUSES A CARRIAGE RETURN/LINE FEED

                /                        TO BE OUTPUT.

0400  0000  TYPX,   0
0401  7300          CLA CLL
0402  1600          TAD I   TYPX            /GET POINTER
0403  3216          DCA     TYPNT           /AND SAVE IT LOCALLY
0404  2200          ISZ     TYPX

0405  1616  TYPX1,  TAD I   TYPNT           /GET LEFT HAND CHARACTER
0406  7012          RTR                     /THIS CAN BE CHANGED TO A
0407  7012          RTR                     /BYTE SWAP ON THE PDP-8E!
0410  7012          RTR
0411  4217          JMS     TYPY            /CONVERT AND TYPE
0412  1616          TAD I   TYPNT           /GET RIGHT HAND CHARACTER
0413  2216          ISZ     TYPNT           /MOVE POINTER TO NEXT WORD

0414  4217          JMS     TYPY            /CONVERT AND TYPE
0415  5205          JMP     TYPX1           /CONTINUE UNTIL DONE.

0416  0000  TYPNT,  0                       /POINTER TO STRING


                /THIS SUBROUTINE IS CALLED BY "TYPX" (AND "KRBX"!)--
                /ENTER WITH THE CHARACTER TO TYPE IN AC6-11:
                /       (1) TESTS FOR TERMINATOR (00)
                /       (2) TESTS FOR CR-LF (37) CHARACTER
                /       (3) CONVERTS CHARACTER TO 8-BIT ASCII AND TYPES IT


0417  0000  TYPY,   0
0420  0236          AND     TK77            /MASK OFF CHARACTER
0421  7450          SNA                     /TEST FOR TERMINATOR
0422  5600          JMP I   TYPX            /EXIT IF TERMINATOR
0423  1237          TAD     TKM37
0424  7440          SZA                     /TEST FOR CR-LF
0425  5231          JMP     TYPY1           /NOT A 37
0426  1240          TAD     TK215           /TYPE A -CR-
0427  4244          JMS     TLSX
0430  1241          TAD     TKM125          /CONVERTS TO A -LF- (212)

0431  7510  TYPY1,  SPA                     /TEST RANGE
0432  1242          TAD     TK100           /RANGE IS 301-336
0433  1243          TAD     TK237           /RANGE IS 240-277
0434  4244          JMS     TLSX            /TYPE CHARACTER
0435  5617          JMP I   TYPY
```

```
0436  0077  TK77,   77                        /MASK FOR AC6-11
0437  7741  TKM37,  -37                       /TEST FOR CR-LF CHARACTER
0440  0215  TK215,  215                       /ASCII VALUE OF -CR-
0441  7653  TKM125, -125                      /THIS PLUS 337 = 212 -LF-
0442  0100  TK100,  100                       /CONVERT TO RANGE 301-336
0443  0237  TK2o7,  237                       /CONVERT TO RANGE 240-277

            /TELETYPE OUTPUT SUBROUTINE.
            /ENTER WITH CHARACTER IN THE AC.
            /       --NOTE--
            /THE TELETYPE FLAG MUST BE INITIALIZED
            /BEFORE CALLING THIS SUBROUTINE!

0444  0000  TLSX,   0
0445  6041        TSF
0446  5245        JMP     .-1               /WAIT FOR TELETYPE READY
0447  6046        TLS
0450  7200        CLA
0451  5644        JMP I   TLSX


            /TELETYPE INPUT SUBROUTINE.
            /EXIT WITH CHARACTER READ IN THE AC.

0452  0000  KRBX,   0
0453  6031        KSF
0454  5253        JMP     .-1               /WAIT FOR CHARACTER
0455  6036        KRB
0456  5652        JMP I   KRBX



            /TELETYPE INPUT SUBROUTINE.
            /THIS SUBROUTINE DEMONSTRATES HOW TO INPUT
            /ALPHANUMERIC CHARACTERS FROM THE TELETYPE.     IT READS
            /THE CHARACTERS INTO A BUFFER, ECHOES CHARACTERS TYPED,
            /AND PERFORMS MINOR EDITING:
            /       (1) CHARACTERS NULL(200 AND 000) AND LINE FEED
            /       ARE DELETED ON INPUT.
            /       (2) CHARACTER RUBOUT (377) DELETES THE PREVIOUS
            /       CHARACTER TYPED.  IT ECHOES AS A BACKSLASH FOLLOWED

            /       BY THE CHARACTER DELETED (MULTIPLE RUBOUTS DO NOT
            /       ECHO BACKSLASH AFTER THE FIRST: A NON-RUBOUT CHAR-

            /       ACTER CAUSES A "CLOSING" BACKSLASH TO BE PRINTED).

            /THE CHARACTER CARRIAGE RETURN TERMINATES INPUT.        IT

            /ECHOES AS A CARRIAGE RETURN FOLLOWED BY A LINE FEED.
```

```
                /ENTRY, JMS     KREAD
                /       POINTER         (ADDRESS OF INPUT BUFFER)
                /       -LENGTH         (MINUS SIZE OF INPUT BUFFER)
                /EXIT IS TO ENTRY+2; ON EXIT:
                /       (1) AC IS CLEAR.
                /       (2) A TERMINATING WORD OF ZERO IS PLACED IN THE
                /       INPUT BUFFER (THE CARRIAGE RETURN IS NOT ENTERRED
                /       IN THE BUFFER).

                /               --NOTE--
                /ON BUFFER OVERFLOW, CHARACTERS WILL BE ECHOED AS "BELL"
                /TO INFORM USER THAT BUFFER IS FULL.     ALL CHARACTERS,
                /OTHER THAN CARRIAGE RETURN AND RUBOUT, ARE LOST.

                /THIS SUBROUTINE IS WRITTEN TO CO-RESIDE WITH THE TELETYPE

                /OUTPUT SUBROUTINES "TYPY" AND "TLSX".

0457  0000  KREAD, 0
0460  7300        CLA CLL
0461  1657        TAD I   KREAD           /GET ADDRESS OF BUFFER
0462  2257        ISZ     KREAD
0463  3216        DCA     KRPNT           /SET UP POINTER
0464  1657        TAD I   KREAD           /GET SIZE OF BUFFER
0465  3200        DCA     KRCNT           /SET UP COUNTER

0466  4252  KRB1, JMS     KRBX            /GET CHARACTER
0467  3616        DCA I   KRPNT           /SAVE CHARACTER
0470  1360        TAD     KRTAB           /LOAD POINTER TO TABLE
0471  3345        DCA     KRBKS           /INTO TEMP. STORAGE
                /ENTER SCANNING LOOP.

0472  1616  KRB3, TAD I   KRPNT           /ADD IN CURRENT CHARACTER
0473  2345        ISZ     KRBKS           /ADVANCE INDEX TO JMP WORD

0474  7650        SNA CLA
0475  5745        JMP I   KRBKS           /CALL SPECIAL ROUTINE
0476  2345        ISZ     KRBKS           /GO ON TO NEXT ENTRY
0477  1745        TAD I   KRBKS           /GET TABLE ENTRY
0500  7440        SZA                     /ZERO MARKS END OF TABLE
0501  5272        JMP     KRB3            /CONTINUE SCAN


                /NORMAL CHARACTER ROUTINE--
                /CHECK FOR BUFFER OVERFLOW, ECHO CHARACTER, AND
                /RETURN TO FETCH NEXT CHARACTER.
0502  4345        JMS     KRBKS           /ECHO BACKSLASH IF NEEDED
0503  2200        ISZ     KRCNT           /TEST FOR BUFFER OVERFLOW
0504  5312        JMP     KRB6            /NORMAL ROUTE
                /BUFFER OVERFLOW!
0505  1356        TAD     TK207           /ECHO "BELL"
0506  4244  KRB5, JMS     TLSX
0507  7240        CLA CMA                 /DECREMENT CHAR COUNTER
0510  1200        TAD     KRCNT
0511  5265        JMP     KRB1-1          /RETURN TO GET NEXT CHAR
```

```
0512  1616  KRB6,   TAD I   KRPNT          /GET CHARACTER
0513  2216          ISZ     KRPNT          /ADVANCE BUFFER POINTER
0514  4244          JMS     TLSX           /ECHO CHARACTER
0515  5266          JMP     KRB1           /RETURN TO GET NEXT CHAR


              /RUBOUT ROUTINE--
              /THIS ROUTINE IS CALLED WHEN A RUBOUT IS TYPED; IT DELETES

              /THE LAST CHARACTER IN THE BUFFER, RESET THE POINTERS,
              /AND DOES SOME FANCY ECHOING.
0516  7240  KRUB,   CLA CMA                 /REVERSE RUBOUT FLAG TEST
0517  4345          JMS     KRBKS          /EHCO BACKSLASH IF NEEDED
0520  2355          ISZ     KRFLAG         /SET RUBOUT FLAG
0521  1657          TAD I   KREAD          /CHECK FOR "EMPTY" BUFFER
0522  7041          CIA
0523  1200          TAD     KRCNT          /BY COMPARING COUNTERS
0524  7650          SNA CLA
0525  5333          JMP     KRUB1          /NO RUBOUT PAST BEGINNING
0526  7240          CLA CMA
0527  1216          TAD     KRPNT          /DECREMENT BUFFER POINTER
0530  3216          DCA     KRPNT
0531  1616          TAD I   KRPNT          /GET CHARACTER TO ECHO
0532  5306          JMP     KRB5

              /ECHO CR-LF-BACKSLASH ON ATTEMPT TO RUBOUT PAST THE START
              /OF THE INPUT BUFFER.
0533  1243  KRUB1,  TAD     TK237          /LOOKS LIKE A "←"!
0534  4217          JMS     TYPY           /FORCES A CR-LF
0535  4345          JMS     KRBKS          /ALWAYS FORCES A BACKSLASH

0536  5266          JMP     KRB1

              /CARRIAGE RETURN ROUTINE--
              /THIS ROUTINE IS CALLED WHEN A CARRIAGE RETURN IS TYPED;
              /ECHOES CR-LF, DEPOSITS 0 IN INPUT BUFFER, AND EXITS.
0537  4345  KRCR,   JMS     KRBKS          /ECHO BACKSLASH IF NEEDED.

0540  1243          TAD     TK237          /LOOKS LIKE A "←"!
0541  4217          JMS     TYPY           /FORCES A CR-LF
0542  3616          DCA I   KRPNT          /STORE ZERO IN BUFFER
0543  2257          ISZ     KREAD          /SET TO EXIT ADDRESS
0544  5657          JMP I   KREAD          /AND EXIT.
```

```
                /THIS SUBROUTINE ECHOES BACKSLASH WHEN KRFLAG IS NON-ZERO
                /NOTE-- LOCATION "KRBKS" IS ALSO USED AS A TEMPORARY

0545  0000  KRBKS,  0
0546  1355          TAD     KRFLAG          /GET FLAG
0547  7640          SZA CLA
0550  1357          TAD     TK334           /BACKSLASH CHARACTER
0551  7440          SZA
0552  4244          JMS     TLSX            /TYPE A BACKSLASH
0553  3355          DCA     KRFLAG          /CLEAR FLAG
0554  5745          JMP I   KRBKS           /EXIT

0555  0000  KRFLAG, 0                       /SET TO +1 IF A RUBOUT WAS

                                            /LAST CHAR, OTHERWISE 0.

                /SAVE A COUPLE LOCATIONS BY USING TYPX AND TYPNT.
                KRPNT=TYPNT                 /POINTER TO INPUT BUFFER
                KRCNT=TYPX                  /MINUS NO. OF WORDS LEFT
0556  0207  TK207,  207                     /BELL IS ASCII 207
0557  0334  TK334,  334                     /BACKSLASH IS ASCII 334


                /SPECIAL CHARACTER TABLE--
                /ENTRIES IN THIS TABLE ARE TWO WORDS LONG:
                /       (WORD 1)        -VALUE OF CHARACTER
                /       (WORD 2)        JMP TO PROPER ROUTINE
                /THE TABLE IS TERMINATED BY AN ENTRY OF 0.

0560  0560  KRTAB,  .

0561  5266          JMP KRB1                /NULL -- IGNORE
0562  7600          -200;
0563  5266          JMP KRB1        /NULL -- IGNORE
0564  7566          -212;
0565  5266          JMP KRB1        /LINE FEED -- IGNORE
0566  7563          -215;
0567  5337          JMP KRCR        /CARRIAGE RETURN -- EXIT
0570  7401          -377;
0571  5316          JMP KRUB        /RUBOUT -- DELETE CHARACTER
0572  0000          0                       /DENOTES END OF TABLE.


                PAUSE
```

CHAPTER 7

MASTER TAPE DUPLICATOR/VERIFIER

7.1 ABSTRACT

This program duplicates and verifies 8-channel paper tapes using a PDP-8 family computer with high-speed reader and high-speed punch. The program uses the program interrupt and allows both the reader and the punch to operate at maximum speed.

The program accumulates two types of checksums while reading and punching: 1) the number of nonzero characters on the tape, and 2) the sum of characters on the tape (both are taken modulo 4096).

When duplicating, the program compares the checksums at the end of the tape with the checksums accumulated by the read routine. If these differ, a reader error has occurred and a message is printed. Tapes are verified by reading them and comparing accumulated checksums with those at the end of the tape. Only master tapes produced by the program can be duplicated. The master tape has the two checksums punched at the end.

7.2 REQUIREMENTS

The program uses all of memory, except for the last page, as a buffer.

The Master Tape Duplicator program runs on any PDP-8 family computer with high-speed reader and high-speed punch. The program tapes are as follows:

Binary	DIGITAL-8-16-S-BIN
Source	DIGITAL-8-16-S-ASC

7.3 USAGE

7.3.1 Loading

The program is loaded with the Binary Loader (see Introduction

to Programming or Programming Languages for details).

7.3.2 To Produce a Master Tape

A tape is read and duplicated by the punch. When the tape has run out of the reader, the accumulated checksums are punched. The tape that has been punched is the master tape used for duplication. It should be compared against the original to ensure that the tape was read correctly.

7.3.3 To Duplicate the Master Tape

The master tape that has been produced (see 7.3.2) is reproduced by the punch. Checksums are accumulated by the read routine and are compared with the checksums at the end of the tape. Checksums are punched and are used for verification (see 7.3.4). If the master tape is short enough to fit into the buffer, the program will notify the operator that more copies can be made without rereading the master. Blank tape is punched between copies.

7.3.4 Verify Duplication

Similar to duplication, but no punching takes place. Tapes are read and the accumulated checksums are compared against the checksums punched at the end.

7.4 OPERATION PROCEDURES

a. Set the SWITCH REGISTER to 200.

b. Press the LOAD ADDRESS key; press the START key; the program halts.

c. Set SWITCH REGISTER for the mode of operation as follows:

Bit 0 = 1	Make master tape
Bit 1 = 1	Duplicate master tape
Bit 2 = 1	Verify duplication

d. Place tape in reader starting on blank tape (all modes of operation must be started with blank leader tape in the reader).

e. Turn reader on. Turn punch on.

f. Press the CONTINUE key.

g. The program prints a message when the operation has been completed and then halts.

h. Proceed from step (c) unless multiple copies are being made.

7.5 DESCRIPTION

This program uses the program interrupt to keep the reader and the punch running at full speed. The reader fills a buffer and the punch punches from it. Checksums are accumulated by both the reader and the punch routines.

7.6 NOTE ON EXTRA BLANK FRAMES IN DUPLICATED TAPES

The Master Tape Duplicator does not check for extra blank frames in the duplicate tape. A future version of this program will perform such a check. Until this version is released, users with tape Digital-8-16-S having difficulty with binary tapes which load and verify properly but do not run properly should order tape and document number 5-10 from the DECUS Program Library. The document is called Paper Tape Reader Tester. It is a program for the PDP-5, but will run on the PDP-8 and should be used as a second verifying operation. Programs which are too long for the space left in core should be broken into two or more shorter tapes for this operation.

7.7 PROGRAM LISTING

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /TAPE DUPLICATOR FOR PDP-5/8
                /       -DEC-1/15/65
                /SINGLE BUFFERING-READ AND PUNCH UTILIZING
                /PROGRAM INTERRUPT
                /COMPUTE A CHARACTER COUNT AND CHECKSUM
                /FOR EACH TAPE-COMPARE WITH CHECKS AT
                /END OF TAPE
                /CHECKS ARE ALSO COMPUTED DURING PUNCHING
                /AND COMPARED
                /THREE MODES OF OPERATION:
                /A. SWITCH 0 ON-MAKE MASTER TAPE
                /B. SWITCH 1 ON-DUPLICATE MASTER TAPE
                /C. SWITCH 2 ON-VERIFY DUPLICATION
                /DURING DUPLICATION, THE PROGRAM WILL NOTIFY
                /THE OPERATOR WHETHER OR NOT MORE COPIES
                /CAN BE MADE WITHOUT RE-READING THE
                /MASTER
                /DEFINITIONS OF INTERRUPT LOCATIONS:
                /FOR THE PDP-8; INTER=0
                /FOR THE PDP-5; INTER=1
                /PAGE 1
                INTER=0

                *INTER+1
0001  5020      HNDL,   JMP HNDL                /HANDLE INTERRUPT
0002  0632      NPNT,   DPRT
0003  0600      E1,     TES1
0004  0615      E2,     TES2
                *16
0016  0000      NDXR,   0                       /AUTO-INDEX REGISTER
0017  0000      NDXP,   0                       /AUTO-INDEX REGISTER
0020  6011      HNDL,   RSF
0021  7410              SKP
0022  5431              JMP I READ              /750 CAUSED INTERRUPT
0023  6021              PSF
0024  7410              SKP
0025  5434              JMP I PNC               /75A PUNCH CAUSED INTERRUPT
0026  4567              JMS I CRLF              /EXTRANEOUS-CLEAR FLAGS
0027  0000      DSMS,   ION                     /ENABLE INTERRUPT
0030  5400              JMP I INTER             /RETURN
0031  0000      READ,   0                       /CALLED AS A
0032  6014              RFC                     /SUBROUTINE TO PROVIDE
0033  5027              JMP DSMS                /RETURN ADDRESS
0034  0000      PNC,    0                       /CALLED AS SUBROUTINE
0035  6026              PLS                     /PROVIDES RETURN
0036  7200              CLA
0037  5027              JMP DSMS
```

```
0040  1417  PNCH,   TAD I NDXP          /GET NEXT CHARACTER
0041  7510          SPA                 /IF IT IS 7777, IT
0042  5570          JMP I PDUN          /IS END OF TEXT
0043  3143          DCA HLD2
0044  1143          TAD HLD2
0045  7440          SZA
0046  2145          ISZ ZROP            /COUNT NON-ZERO CHARACTERS
0047  7000          NOP                 /MODULO 4096
0050  1147          TAD CHKP            /ACCUMULATE SUM MODULO 4096
0051  3147          DCA CHKP
0052  1143          TAD HLD2
0053  4034          JMS PNC
0054  5040          JMP PNCH            /GO GET NEXT
0055  6012  RD1,     RRB                 /READ 750 BUFFER
0056  3142          DCA HLD1            /SAVE IT
0057  5105  RST1,    JMP FRST            /OF "SCND" OR "THRD"
0060  1142          TAD HLD1
0061  7440          SZA
0062  2144          ISZ ZROR            /COUNT MODULO 4096
0063  7000          NOP
0064  1146          TAD CHKR            /ACCUMULATE SUM
0065  3146          DCA CHKR
0066  1160          TAD TIME            /RESET END-OR-TAPE TIMER
0067  3157          DCA TIMR
0070  7410  RST2,    SKP
0071  5155          JMP VY
0072  1142          TAD HLD1            /GET CHARACTER
0073  3416          DCA I NDXR          /PUT IN BUFFER
0074  2163          ISZ RCNT            /IS BUFFER FULL?
0075  7410          SKP                 /NO
0076  5571          JMP I FUL           /YES
0077  4031          JMS READ            /FETCH NEXT CHARACTER
0100  2164          ISZ STRT            /DELAY START OF PUNCHING
0101  5055          JMP RD1
0102  7240          CLA CMA
0103  3175          DCA SCON
0104  5040          JMP PNCH            /START PUNCHING
0105  1142  FRST,    TAD HLD1            /TEST TO SEE IF
0106  1161          TAD TST1            /CHARACTER IS FIRST
0107  7640          SZA CLA             /IN CHECK-SUM IDENTIFIER
0110  5060          JMP RST1+1          /IF IT IS-SET SWITCH
0111  1153          TAD TRY2            /TO TEST FOR SECOND
0112  3057          DCA RST1            /CHARACTER NEXT
0113  5060          JMP RST1+1
0114  1142  SCND,    TAD HLD1            /CHECK FOR SECOND CHARACTER
0115  1162          TAD TST2            /IDENTIFIER - IF FOUND
0116  7640          SZA CLA             /TEST FOR THIRD NEXT
0117  5123          JMP .+4             /IF NOT, RESET FOR FIRST
0120  1154          TAD TRY3
0121  3057          DCA RST1
0122  5060          JMP RST1+1
```

```
0123  1152          TAD TRY1
0124  5121          JMP .-3
0125  1142  THRD,   TAD HLD1          /TEST FOR THIRD CHARACTER
0126  1161          TAD TST1          /IN IDENTIFIER - IF FOUND
0127  7640          SZA CLA           /READ CHECKS FROM TAPE
0130  5123          JMP .-5           /IF NOT - RESET FOR FIRST
0131  7240          CLA CMA
0132  3416          DCA I NDXR        /SET END-OF PUNCH FLAG
0133  4541          JMS I GET1
0134  3151          DCA MCHK          /MEASURED CHECK-SUM
0135  4541          JMS I GET1
0136  3150          DCA MZRO          /MEASURED ZERO-COUNT
0137  5540          JMP I .+1
0140  0303          SWT1
0141  0565  GET1,   GET               /GET 12-BIT WORD
0142  0000  HLD1,   0
0143  0000  HLD2,   0
0144  0000  ZROR,   0                 /#OF NON-ZERO READ
0145  0000  ZROP,   0                 /#OF NON-ZERO PUNCHED
0146  0000  CHKR,   0                 /CHECK SUM - READ
0147  0000  CHKP,   0                 /CHECK SUM - PUNCH
0150  0000  MZRO,   0                 /#OF NON-ZERO MEASURED
0151  0000  MCHK,   0                 /CHECK SUM - MEASURED
0152  5105  TRY1,   JMP FRST
0153  5114  TRY2,   JMP SCND
0154  5125  TRY3,   JMP THRD          /WHEN VERIFYING-DON' START
0155  4031  VY,     JMS READ          /PUNCHING
0156  5055          JMP RD1
0157  0000  TIMR,   0
0160  0000  TIME,   0
0161  7526  TST1,   7526              /2'S COMPLEMENT 1ST AND
                                      /THIRD IDENTIFIER
0162  7653  TST2,   7653              /TWO'S COMPLEMENT OF 2ND
                                      /IDENTIFIER
0163  0000  RCNT,   0
0164  0000  STRT,   0
0165  7000  NOPT,   NOP
0166  7410  SKIP,   SKP
0167  0345  CRLF,   CFLG
0170  0000  PDUN,   0
0171  0320  FUL,    BFUL
0172  0077  BIT6,   0077
0173  0000  DCON,   0
0174  0000  VCNT,   0
0175  0000  SCON,   0                 /START OF PUNCHING FLAG
0176  0377  C377,   0377
0177  0002  TWO,    0002
```

```
                *200
0200   7200             CLA
0201   3174             DCA Z VCNT              /RESET VERIFY COUNT
0202   7200  LOOP,      CLA
0203   3144             DCA ZROR                /RESET PUNCH,READER
0204   3145             DCA ZROP                /COMPUTED CHECKS
0205   3146             DCA Z CHKR
0206   3147             DCA Z CHKP
0207   3173             DCA     DCON            /RESET  BUFFER OVER FLAG
0210   6012             RRB                     /CLEAR HARDWARE FLAGS
0211   6022             PCF
0212   7200             CLA
0213   3175             DCA SCON                /RESET START PUNCH FLAG
0214   1152             TAD Z TRY1              /SET PROGRAM SWITCHES
0215   3057             DCA Z RST1
0216   1166             TAD Z SKIP
0217   3070             DCA Z RST2
0220   1166             TAD Z SKIP
0221   3303             DCA SWT1
0222   1341             TAD DLAY                /SET START OF PUNCHING DELAY
0223   3164             DCA Z STRT
0224   7402  LOP1,      HLT
0225   7604             CLA OSR
0226   7004             RAL
0227   7430             SZL
0230   5250             JMP CRTE                /BIT 0=1, CREATE MASTER
0231   7004             RAL
0232   7430             SZL
0233   7402             JMP DUP                 /BIT 1=1, DUPLICATE
0234   7004             RAL
0235   7630             SZL CLA
0236   5241             JMP .+3                 /BIT 2=1, VERIFY
0237   7040             CMA                     /ERROR SET C(AC)=7777
0240   5224             JMP LOP1
0241   1165             TAD NOPT                /NO PUNCHING - DON'T
0242   3303             DCA SWT1                /WAIT FOR PUNCH
0243   1165             TAD Z NOPT
0244   3070             DCA RST2                /DON'T START PUNCHING
0245   1342             TAD VRPT
0246   3336             DCA CONR                /SET-UP RETURN FOR END
0247   5261             JMP GO
0250   7200  CRTE,      CLA
0251   1343             TAD CRPT
0252   3336             DCA CONR                /SET-UP RETURN FOR END
0253   3174             DCA Z VCNT
0254   5261             JMP GO
0255   7200  DUP,       CLA
0256   1344             TAD DUPT
0257   3336             DCA CONR                /SET-UP RETURN FOR END
0260   3174             DCA Z VCNT
```

```
0261  4345  GO,     JMS CFLG              /CLEAR FLAGS
0262  1337          TAD BUF
0263  3016          DCA NDXR              /SET-UP BUFFER
0264  1337          TAD BUF               /POINTERS FOR
0265  3017          DCA NDXP              /READ AND PUNCH
0266  1340          TAD OVR
0267  3163          DCA RCNT              /SET BUFFER-FULL COUNT
0270  1333          TAD SA
0271  3000          DCA INTER             /INITIALIZE INTERRUPT
0272  1160          TAD Z TIME
0273  3157          DCA TIMR              /SET END-OF-TAPE TIMER
0274  4031          JMS READ              /START READING
0275  5055          JMP Z RD1
0276  2157  SAT,     ISZ Z TIMR
0277  5276          JMP .-1
0300  0000          IOF                   /END-OF-TAPE
0301  7240          CLA CMA
0302  3416          DCA I Z NDXR          /SET END-OF-PUNCHING FLAG
0303  7410  SWT1,    SKP                   /OR NOP FOR VERIFY
0304  5736          JMP I CONR
0305  1336          TAD CONR
0306  3170          DCA PDUN              /SET RETURN FOR PUNCH DONE
0307  1175          TAD Z SCON
0310  7700          SMA CLA               /DID WE START PUNCHING?
0311  5315          JMP .+4               /NO -
0312  0000          ION                   /YES - WAIT FOR PUNCHING
0313  7200          CLA
0314  5313          JMP .-1
0315  1334          TAD SA1               /START PUNCHING
0316  3000          DCA INTER
0317  5102          JMP Z FRST-3
0320  7240  BFUL,    CLA CMA               /BUFFER-FULL
0321  3416          DCA I Z NDXR          /SET-UP END-OF-PUNCHING FLAG
0322  2173          ISZ Z DCON
0323  1335          TAD RET               /'FILL'
0324  3170          DCA Z PDUN
0325  5307          JMP BFUL-11
0326  1341  FILL,    TAD DLAY              /ENTER WHEN BUFFER
0327  3164          DCA STRT              /OVERFLOWED AND HAS
0330  3175          DCA Z SCON
0331  6022          PCF
0332  5262          JMP GO+1              /BEEN PUNCHED
0333  0276  SA,      SAT
0334  0312  SA1,     BFUL-6
0335  0326  RET,     FILL
0336  0000  CONR,    0
0337  1063  BUF,     X-1                   /START OF BUFFER
0340  1274  OVR,     X+210                 /BUFFER-FULL COUNT
0341  7760  DLAY,    7760                  /DELAY START OF PUNCHING
0342  0400  VRPT,    VRFY                  /DONE POINTERS
```

```
0343  0404  CRPT,   CRET
0344  0445  DUPT,   DUPL
0345  0000  CFLG,   0                    /CLEAR FLAGS
0346  6042          TCF                  /TELEPRINTER
0347  6072          6072                 /LIGHT PEN
0350  6772          6772                 /MICRO TAPE
0351  6502          6502                 /PLOTTER
0352  6732          6732                 /DISABLE ERF FLAG (57A)
0353  7320          CLA CLL CML
0354  7012          RTR                  /SET C(AC)=2000
0355  6722          6722                 /DISABLE WCO FLAG (57A)
0356  7004          RAL                  /SET C(AC)=4000
0357  6702          6702                 /DISABLE TCR FLAG (57A)
0360  6652          6652                 /LINE-PRINTER FLAG
0361  6534          6534                 /138 ADC FLAG
0362  6032          KCC                  /KEY-BOARD (AND AC)
0363  5745          JMP I CFLG

              PAUSE
              *400
0400  2174  VRFY,   ISZ Z VCNT
0401  4403          JMS I E1             /COMPUTED VS MEASURED CHECKS
0402  5361          JMP VER              /VERIFY ERROR
0403  5351          JMP VOK              /VERIFY OK
0404  4404  CRET,   JMS I E2             /COMPUTED VS PUNCHED
0405  5345          JMP MER              /MEMORY ERROR
0406  4314          JMS BLNK
0407  1327          TAD HERE
0410  3000          DCA INTER            /SET INTERRUPT POINTER
0411  1325          TAD I1               /PUNCH CHECKSUM
0412  4034          JMS Z PNC            /IDENTIFIER CODES
0413  1326          TAD T2               /BIT PATTERN IS:
0414  4034          JMS PNC              /10101010
0415  1325          TAD T1               /01010101
0416  4034          JMS PNC              /10101010
0417  1146          TAD CHKR             /PUNCH CHECKS
0420  1176          TAD C377             /ALTER CHECKS
0421  4231          JMS SIX
0422  1144          TAD Z ZNOR
0423  1177          TAD Z TWO
0424  4231          JMS SIX
0425  4314          JMS BLNK             /PUNCH BLANK TAPE
0426  1335          TAD AD1
0427  4734          JMS I PRNT
0430  5733          JMP I LOP
0431  0000  SIX,    0                    /PUNCH NUMBER IN AC
0432  3143          DCA Z HLD2
0433  1143          TAD HLD2             /IN BINARY FORMAT
0434  7012          RTR
0435  7012          RTR
0436  7012          RTR
```

```
0437  0172          AND Z BIT6
0440  4034          JMS Z PNC
0441  1143          TAD Z HLD2
0442  0172          AND Z BIT6
0443  4034          JMS Z PNC
0444  5631          JMP I SIX
0445  4403  DUPL,    JMS I Z E1          /COMPARE COMPUTED VS
                                        /MEASURED CHECKS
0446  5347          JMP RER             /READER ERROR
0447  4404  GO1,     JMS I Z E2          /COMPARE COMPUTED VS
                                        /PUNCHED CHECKS
0450  5345          JMP MER             /MEMORY ERROR
0451  1327          TAD HERE
0452  3000          DCA INTER
0453  1325          TAD T1              /PUNCH THIRD IDENTIFIER
0454  4034          JMS Z PNC           /READER STOPPED WHEN THIRD
                                        /IDENTIFIER
0455  1146          TAD Z CHKR          /HAS BEEN FOUND, IE IT IS
                                        /NOT IN THE BUFFER
0456  4231          JMS SIX
0457  1144          TAD Z ZROR
0460  4231          JMS SIX
0461  4314          JMS BLNK            /PUNCH BLANK TAPE
0462  1173          TAD Z DCON
0463  7640          SZA CLA
0464  5356          JMP DOK             /BUFFER OVERLAP-NO MORE DUP.
0465  1324          TAD DCNT
0466  7640          SZA CLA             /STARTED MULTIPLE
                                        /DUPLICATION??
0467  5305          JMP TST4            /MULTIPLE DUPLICATION HAS
                                        /STARTED
0470  1336          TAD AD2
0471  4734          JMS I PRNT
0472  7402          HLT                 /READ NUMBER FROM SR
0473  7604          CLA OSR
0474  7041          CMA IAC
0475  3324          DCA DCNT
0476  1332          TAD HER1
0477  3170          DCA Z PDUN          /SET RETURN FOR PUNCHING
                                        /DONE
0500  1360          TAD BF              /RESET BUFFER POINTER
0501  3017          DCA Z NDXP          /RESET PUNCH-COMPUTED CHECKS
0502  3145          DCA Z ZROP
0503  3147          DCA Z CHKP
0504  5040          JMP Z PNCH          /START PUNCHING
0505  2324  TST4,    ISZ     DCNT        /ARE WE DONE YET?
0506  7410          SKP
0507  5356          JMP DOK             /YES
0510  1337          TAD AD3             /NO
0511  4734          JMS I PRNT
0512  7602          HLT CLA             /HALT
0513  5300          JMP TST4-5          /MAKE NEXT COPY
```

```
0514  0000  BLNK,   0                        /SUBROUTINE TO PUNCH
0515  1323          TAD MCNT                 /BLANK TAPE
0516  3157          DCA Z TIMR
0517  4034          JMS Z PNC
0520  2157          ISZ Z TIMR
0521  5317          JMP .-2
0522  5714          JMP I BLNK
0523  7600  MCNT,   7600
0524  0000  DCNT,   0
0525  0252  T1,     0252                     /CODES FOR CHECK-SUM
0526  0125  T2,     0125                     /IDENTIFIERS
0527  0530  HERE,   HERE+1
0530  7200          CLA
0531  5330          JMP .-1
0532  0447  HER1,   GO1
0533  0202  LOP,    LOOP
0534  0673  PRNT,   PRIN
0535  0741  AD1,    TAB1                     /MASTER CREATED
0536  0751  AD2,    TAB2                     /PRINT TABLE
0537  1007  AD3,    TAB3                     /PRINT TABLE
0540  1017  AD4,    TAB4                     /VERIFY OK
0541  1026  AD5,    TAB5                     /DUP OK
0542  1036  AD6,    TAB6                     /MEMORY ERROR
0543  1045  AD7,    TAB7                     /READER ERROR
0544  1054  AD8,    TAB8                     /VERIFY ERROR
0545  1342  MER,    TAD AD6
0546  5354          JMP VOK+3
0547  1343  RER,    TAD AD7
0550  5354          JMP VOK+3
0551  1174  VOK,    TAD Z VCNT               /VERIFY OK
0552  4402          JMS I NPNT
0553  1340          TAD AD4
0554  4734          JMS I PRNT
0555  5733          JMP I LOP
0556  1341  DOK,    TAD AD5                  /DUPLICATION OK
0557  5354          JMP VOK+3
0560  1063  BF,     X-1
0561  1174  VER,    TAD Z VCNT               /VERIFY OK
0562  4402          JMS I Z NPNT
0563  1344          TAD AD8
0564  5354          JMP VOK+3
0565  0000  GET,    0                        /ROUTINE TO READ 2-6 BIT
                                             /CHARACTERS
0566  4031          JMS Z READ
0567  6012          RRB
0570  7106          CLL RTL
0571  7006          RTL
0572  7006          RTL
0573  3142          DCA Z HLD1
0574  4031          JMS Z READ
0575  6012          RRB
0576  1142          TAD Z HLD1
0577  5765          JMP I GET
```

```
                *600
0600  0000  TES1,  0                 /COMPARE READER COMPUTED
                                     /SUMS
0601  7200        CLA
0602  1144        TAD Z ZROR         /TO MEASURED SUMS
0603  7041        CMA IAC            /IF EQUAL, RETURN TO CALL+2
0604  1150        TAD Z MZRO         /OTHERWISE RETURN TO CALL+1
0605  7640        SZA CLA
0606  5600        JMP I TES1
0607  1146        TAD Z CHKR
0610  7041        CMA IAC
0611  1151        TAD Z MCHK
0612  7650        SNA CLA
0613  2200        ISZ TES1
0614  5600        JMP I TES1
0615  0000  TES2,  0                 /COMPARE PUNCH COMPUTED
                                     /SUMS
0616  7200        CLA
0617  1145        TAD Z ZROP         /TO READER COMPUTED SUMS
0620  7041        CMA IAC
0621  1144        TAD Z ZROR         /IF EQUAL, RETURN TO CALL+2
0622  7640        SZA CLA            /OTHERWISE RETURN TO CALL+1
0623  5615        JMP I TES2
0624  1146        TAD Z CHKR
0625  7041        CMA IAC
0626  1147        TAD Z CHKP
0627  7650        SNA CLA
0630  2215        ISZ TES2
0631  5615        JMP I TES2
0632  0000  DPRT,  0                 /CONVERT BINARY WORD IN AC
0633  3143        DCA Z HLD2         /TO 4 DIGIT UNSIGNED DECIMAL
0634  3142        DCA Z HLD1         /NUMBER AND TYPE IT
0635  1264        TAD CNTR
0636  3272        DCA CNT            /IDENTICAL TO ROUTINE IN
                                     /LIBRARY
0637  1263        TAD ADDR
0640  3245        DCA XYZ+3
0641  7410        SKP
0642  3143  XYZ,   DCA Z HLD2
0643  7100        CLL
0644  1143        TAD Z HLD2
0645  7422        TAD CON
0646  7430        SZL
0647  2142        ISZ Z HLD1
0650  7430        SZL
0651  5242        JMP XYZ
0652  7200        CLA
0653  1142        TAD Z HLD1
0654  1271        TAD C260
0655  4325        JMS TYPE
0656  3142        DCA Z HLD1
0657  2245        ISZ XYZ+3
0660  2272        ISZ CNT
0661  5244        JMP XYZ+2
```

```
0662  5632          JMP I DPRT
0663  7422  ADDR,   TAD CON
0664  7774  CNTR,   7774
0665  6030  CON,    6030
0666  7634          7634
0667  7766          7766
0670  7777          7777
0671  0260  C260,   0260
0672  0000  CNT,    0
0673  0000  PRIN,   0                       /ROUTINE TO PRINT A STRING
0674  3143          DCA Z HLD2              /OF PACKED ASCII CHARACTERS
0675  1543          TAD I HLD2              /ENTER WITH S.A. OF STRING
0676  7450          SNA
0677  5320          JMP CR                  /IN AC: EXIT ON 0 ELEMENT
STRING                                      /IN STRING
0700  7012          RTR                     /AFTER TYPING CR-LF
0701  7012          RTR
0702  7012          RTR
0703  4310          JMS GPRT
0704  1543          TAD I Z HLD2
0705  4310          JMS GPRT
0706  2143          ISZ Z HLD2
0707  5275          JMP PRIN+2
0710  0000  GPRT,   0                       /THIS ROUTINE CONVERTS
0711  0172          AND Z BIT6              /6BIT TO ASCII
0712  1334          TAD M40
0713  7510          SPA
0714  1335          TAD C100
0715  1336          TAD C200
0716  4325          JMS TYPE
0717  5710          JMP I GPRT
0720  1337  CR,     TAD CAR                 /TYPE CR-LF
0721  4325          JMS TYPE
0722  1340          TAD LF
0723  4325          JMS TYPE
0724  5673          JMP I PRIN
0725  0000  TYPE,   0                 /TYPE CHARACTER IN AC
0726  6046          TLS
0727  6041          TSF
0730  5327          JMP .-1
0731  6042          TCF
0732  7200          CLA
0733  5725          JMP I TYPE
0734  7740  M40,    7740
0735  0100  C100,   0100
0736  0240  C200,   0240
0737  0215  CAR,    0215
0740  0212  LF,     0212
0741  1501  TAB1,   1501                  /PRINT TABLES
0742  2324          2324
0743  0522          0522
```

```
0744   4003            4003
0745   2205            2205
0746   0124            0124
0747   0504            0504
0750   0000            0000
0751   2305   TAB2,    2305
0752   2440            2440
0753   2327            2327
0754   1124            1124
0755   0310            0310
0756   0523            0523
0757   4024            4024
0760   1740            1740
0761   1625            1625
0762   1502            1502
0763   0522            0522
0764   4017            4017
0765   0640            0640
0766   0317            0317
0767   2011            2011
0770   0523            0523
0771   4024            4024
0772   1740            1740
0773   0205            0205
0774   4015            4015
0775   0104            0104
0776   0540            0540
0777   2022            2022
1000   0523            0523
1001   2340            2340
1002   0317            0317
1003   1624            1624
1004   1116            1116
1005   2505            2505
1006   0000            0000
1007   2022   TAB3,    2022
1010   0523            0523
1011   2340            2340
1012   0317            0317
1013   1624            1624
1014   1116            1116
1015   2505            2505
1016   0000            0000
1017   4040   TAB4,    4040
1020   2605            2605
1021   2211            2211
1022   0631            0631
1023   4017            4017
1024   1340            1340
1025   0000            0000
```

```
1026  0425  TAB5,   0425
1027  2014          2014
1030  1103          1103
1031  0124          0124
1032  1117          1117
1033  1640          1640
1034  1713          1713
1035  0000          0000
1036  1505  TAB6,   1505
1037  1517          1517
1040  2231          2231
1041  4005          4005
1042  2222          2222
1043  1722          1722
1044  0000          0000
1045  2205  TAB7,   2205
1046  0104          0104
1047  0522          0522
1050  4005          4005
1051  2222          2222
1052  1722          1722
1053  0000          0000
1054  4040  TAB8,   4040
1055  2605          2605
1056  2211          2211
1057  0631          0631
1060  4005          4005
1061  2222          2222
1062  1722          1722
1063  0000          0000
1064  0000  X,       0                    /START OF BUFFER
```

CHAPTER 8

INCREMENTAL PLOTTER SUBROUTINE

8.1 ABSTRACT

The Incremental Plotter Subroutine moves the pen of a type 350 plotter to a new position along the best straight line. The pen can be raised or lowered during the motion.

8.2 REQUIREMENTS

The subroutine requires one memory page of storage (128_{10} or 200_8 words). The routine works on any PDP-8 family computer equipped with a type 350 Plotter Control and Plotter.

The routine is distributed as an ASCII source tape as follows:

Digital-8-12-U-ASCII

8.3 USAGE

8.3.1 Loading and Assembly

The source tape as supplied has no origin setting and ends with a PAUSE statement. This tape can be assembled with a user program (which supplies an origin setting) or assembled by itself (if a dollar sign [$] is supplied at the end of the tape).

The tape can be assembled with any of the PDP-8 family assemblers.

8.3.2 Calling Sequence

The plotter routine is called by executing a JMS PLOTX. The contents of the accumulator specify the operation of the subroutine as follows:

C(AC) = -1	The location registers internal to the subroutine are reset to zero and the pen is raised. Control returns to the instruction following the calling JMS instruction.
C(AC) = 0	The pen is lowered (if it was up) and is moved to the new located as described below.
C(AC) = 1	The pen is raised (if it was down) and is moved to the new location as described below.

The two locations following the calling JMS instruction contain, respectively, the new X coordinate and the new Y coordinate in steps (these values must be less than 4096). The pen is moved from the previous location to the new location along the best straight line with the pen up or down depending upon the contents of the accumulator when the subroutine is called. Control returns to the instruction following the Y coordinate.

8.3.3 Examples

Initialization of Plotter:

```
        CLA CMA          /AC = -1, INITIALIZE ROUTINE AND PEN UP
        JMS I PLOT       /JUMP TO PLOTX ROUTINE, PAGE INDEPENDENT
        return           /CONTROL RETURNS TO THIS ADDRESS
        .
        .
        .
PLOT,PLOTX               /CONTAINS ADDRESS OF PLOTX ROUTINE
```

Plot with Pen Down:

```
      CLA              /AC = 0, PEN DOWN
      JMS I PLOT
      X coordinate     /MUST BE IN RANGE -4096<X<4096
      Y coordinate     /-4096<Y<4096
        return
        .
        .
        .
PLOT, PLOTX
```

Plot with Pen Up:

```
      CLA IAC          /AC = 1, PEN UP
      JMS I PLOT
      X coordinate     /-4096<X<4096
      Y coordinate     /-4096<Y<4096
      return
      .
      .
      .
PLOT, PLOTX
```

8.4 DESCRIPTION

The routine has two registers which contain the location of the last position plotted. When the subroutine is entered, the accumulator is tested to determine if initialization is being performed; if so the location registers are set to zero, the pen raised, and the subroutine exits. If the routine is not being initialized, the subroutine compares the current pen position (up or down) with the requested one and raises or lowers the pen if appropriate. The new X and Y coordinates are retrieved from the two locations following the calling JMS and placed in the location registers. The X and Y difference between the current location and the desired location are computed and compared. The subroutine selects motion commands depending upon the quadrant of the new location compared to the old. The possible motions are now parallel to either the X-axis (drum motion) or the Y-axis (pen motion) or a combined motion. The subroutine determines which of these motions to use, and when the new location is reached, it exits.

The X and Y coordinates are specified in numbers of steps. Increasing X corresponds to lowering the drum. Increasing Y corresponds to moving the pen left.

The subroutine is limited by the speed of the plotter. The minor subroutine, PLOTWT can be replaced, if necessary, by a routine making use of the program interrupt.

8.5 PROGRAM LISTING

```
/COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
/MAYNARD, MASSACHUSETTS
/DIGITAL 8-12-U
/PLOT SUBROUTINE
/CALLING SEQUENCE

/   C(AC)=-1;  INITIALIZE
/   C(AC)= 0;  PLOT WITH PEN DOWN
/   C(AC)= 1; PLOT WITH PEN UP
/     JMS PLOTX
/     X CO-ORDINATE (IN STEPS) (RETURN IF AC=-1)
/     Y CO-ORDINATE (IN STEPS)

0200  0000  PLOTX,      0
0201  7510    SPA                 /MOVE THE PEN?
0202  5220    JMP PLOTA           /NO: CONTINUE
0203  1361    TAD PLOTPN          /ADD PEN STATUS
0204  7112    CLL RTR
0205  7710    SPA CLA             /ANY CHANGE?
0206  5227    JMP PLOT1           /NO: CONTINUE
0207  7620    SNL CLA
0210  5214    JMP .+4             /LOWER THE PEN
0211  3361    DCA PLOTPN          /RAISE THE PEN
0212  6504    PLPU
0213  5216    JMP .+3
0214  2361    ISZ PLOTPN          /LOWER THE PEN
0215  6524    PLPD
0216  4370    JMS PLOTWT          /WAIT FOR FLAG
0217  5227    JMP PLOT1           /CONTINUE
0220  7200  PLOTA,      CLA
0221  6504    PLPU                /RAISE THE PEN
0222  3361    DCA PLOTPN
0223  3362    DCA PLOTNX          /0 TO X CO-ORDINATE
0224  3363    DCA PLOTNY          /0 TO Y CO-ORDINATE
0225  4370    JMS PLOTWT
0226  5600    JMP I PLOTX
```

```
            /DIGITAL 8-12-U
            /PAGE 2
            /PICK UP ARGUMENTS

0227  1362  PLOT1,      TAD PLOTNX          /FETCH PREVIOUS X CO-ORDINATE
0230  7141    CIA CLL
0231  1600    TAD I PLOTX           /FORM NX-NPX
0232  7420    SNL                   /L=0: NX<NPX
0233  7041    CIA
0234  3364    DCA PLOTDX            /ABSOLUTE VALUE OF DIFFERENCE
0235  7004    RAL
0236  3367    DCA PLOTMV            /SAVE SIGN BIT
0237  1600    TAD I PLOTX           /SET NEW
0240  3362    DCA PLOTNX            /PREVIOUS X
0241  2200    ISZ PLOTX             /INCREMENT POINTER
0242  1363    TAD PLOTNY            /FETCH PREVIOUS Y CO-ORDINATE
0243  7141    CIA CLL
0244  1600    TAD I PLOTX           /FORM NY-NPY
0245  7420    SNL                   /<=0: NPY<NY
0246  7041    CIA
0247  3365    DCA PLOTDY            /ABSOLUTE VALUE OF DIFFERENCE
0250  1367    TAD PLOTMV            /SAVE SIGN BIT
0251  7004    RAL                   /BIT 10(1)= DRUM-DOWN(POSITIVE)
0252  3367    DCA PLOTMV            /BIT 11(1)=PEN-LEFT (POSITIVE)
0253  1600    TAD I PLOTX           /SET NEW
0254  3363    DCA PLOTNY            /PREVIOUS Y
0255  2200    ISZ PLOTX             /INCREMENT POINTER
0256  1364    TAD PLOTDX
0257  7141    CIA CLL
0260  1365    TAD PLOTDY
0261  7620    SNL CLA               /L=0: DELTA Y < DELTA X
0262  5275    JMP PLOT2
0263  1364    TAD PLOTDX            /REVERSE NUMBERS
0264  3366    DCA PLOTNA
0265  1365    TAD PLOTDY
0266  3364    DCA PLOTDX
0267  1366    TAD PLOTNA
0270  3365    DCA PLOTDY
0271  7001    IAC                   /SET MAJOR MOTION
0272  0367    AND PLOTMV            /INSTRUCTION
0273  1342    TAD PLOTT1
0274  5300    JMP .+4

            /DIGITAL 8-12-U
            /PAGE 3

0275  1367  PLOT2,      TAD PLOTMV
0276  7110    CLL RAR
0277  1345    TAD PLOTT2
0300  3366    DCA PLOTNA
0301  1766    TAD I PLOTNA
0302  3340    DCA PLOT4
0303  1367    TAD PLOTMV            /SET COMBINED MOTION
0304  1350    TAD PLOTT3
0305  3367    DCA PLOTMV
0306  1767    TAD I PLOTMV
```

```
0307  3331     DCA PLOTDB
0310  1364     TAD PLOTDX
0311  7110     CLL RAR
0312  3366     DCA PLOTNA
0313  1364     TAD PLOTDX
0314  7040     CMA
0315  3367     DCA PLOTMV
0316  2367   PLOT3,      ISZ PLOTMV
0317  7410     SKP
0320  5600     JMP I PLOTX          /ALL DONE
0321  1366     TAD PLOTNA
0322  1365     TAD PLOTDY
0323  3366     DCA PLOTNA
0324  1366     TAD PLOTNA
0325  7140     CMA CLL
0326  1364     TAD PLOTDX
0327  7630     SZL CLA
0330  5340     JMP PLOT4            /SINGLE MOTION
0331  0000   PLOTDB,     0                    /COMBINED MOTION
0332  1364     TAD PLOTDX
0333  7041     CIA
0334  1366     TAD PLOTNA
0335  3366     DCA PLOTNA
0336  4370     JMS PLOTWT
0337  5316     JMP PLOT3

0340  0000   PLOT4,      0
0341  5336     JMP .-3

0342  0343   PLOTT1,     .+1
0343  6511     PLPR                 /PEN-RIGHT
0344  6521     PLPL                 /PEN-LEFT
0345  0346   PLOTT2,     .+1
0346  6512     PLDU                 /DRUM-UP
0347  6514     PLDD                 /DRUM-DOWN
0350  0351   PLOTT3,     .+1
0351  6513     PLDU PLPR            /UP-RIGHT
0352  6523     PLUD PLPL            /UP-LEFT
0353  6515     PLDU PLPR            /DOWN-RIGHT
0354  4355     JMS .+1              /DOWN-LEFT
0355  0000     0
0356  6514     PLDD
0357  6521     PLPL
0360  5755     JMP I .-3

               /DIGITAL 8-12-U
               /PAGE 4

0361  0000   PLOTPN,     0
0362  0000   PLOTNX,     0
0363  0000   PLOTNY,     0
0364  0000   PLOTDX,     0
0365  0000   PLOTDY,     0
0366  0000   PLOTNA,     0
0367  0000   PLOTMV,     0
```

```
0370  0000  PLOTWT,     0
0371  6501    PLSF                  /WAIT FOR DONE FLAG
0372  5371    JMP .-1               /NOT YET
0373  6502    PLCF                  /CLEAR FLAG
0374  5770    JMP I PLOTWT          /EXIT

              PAUSE
```

```
 PLOTA   0220
PLOTDB  0331
PLOTDX  0364
PLOTDY  0365
PLOTMV  0367
PLOTNA  0366
PLOTNX  0362
PLOTNY  0363
PLOTPN  0361
PLOTT1  0342
PLOTT2  0345
PLOTT3  0350
PLOTWT  0370
PLOTX   0200
PLOT1   0227
PLOT2   0275
PLOT3   0316
PLOT4   0340
```

CHAPTER 9

DECIMAL TO BINARY CONVERSION AND INPUT
(Single Precision, Signed or Unsigned, 33-ASR)

9.1 ABSTRACT

This routine accepts a string of up to four decimal digits (single precision for the PDP-8) from the Teletype keyboard and converts it to the corresponding 2's complement binary number.

The string can contain as legal characters a sign (+,-, or space) and the digits from 0 - 9. If the first legal character is not a sign, the conversion is unsigned. A back arrow (←) at any point in the string erases the current string and allows the operator to reenter the correct value. Any character after the first, other than another digit or back arrow, causes the conversion to terminate and is found in location SISAVE within the subroutine.

9.2 REQUIREMENTS

This subroutine requires 74_{10} (112_8) core locations and runs on any standard PDP-8 family computer with a 33-ASR Teletype console. Program tape is labelled Single Precision Decimal Input,

Digital-8-28-U-ASCII

9.3 USAGE

9.3.1 Loading

The symbolic tape provided can be assembled with the user's main program by PAL III, MACRO-8, or PAL-D. The symbolic tape has neither an origin setting nor a terminating "$", but does have a PAUSE pseudo-instruction at the end.

9.3.2 Calling Sequence

The subroutine is called by an effective JMS to location SICONV. Return is to the location immediately following the calling JMS with the binary number in the accumulator.

9.4 ERRORS IN USAGE

If a sign (+, -, or space) precedes the string of decimal digits, the maximum decimal number correctly accepted is 2047 ($2^{11} - 1$). The sign, if any, must appear first. If a sign does not precede the string of decimal digits, the maximum decimal number correctly accepted is 4095 ($2^{12} - 1$). If either of these maxima is exceeded, the results are unspecified.

9.5 RESTRICTIONS

The status of the AC and link is not preserved.

This subroutine should not be used when the interrupt is on.

The magnitude restrictions on numbers are described in Section 9.4.

9.6 DESCRIPTION

This subroutine converts to the binary equivalent a signed or unsigned string of decimal numbers read from the console keyboard of the PDP-8. If a minus sign is specified, the results are in 2's complement negative form. The first character is examined and, if it is a sign (+, -, or space), a switch is set to provide the correct sign for the conversion. Regardless, a switch is set after the first character to terminate conversion if a character other than a decimal digit or rubout appears. If a back arrow appears at any time, the conversion is reinitialized and the subroutine waits for the correct entry.

The last four bits of the ASCII code for each of the decimal digits are identical to the standard 8-4-2-1 BCD code. Thus, the BCD digit is extracted from the 8-bit code by the AND instruction with a "mask" of 17_8. When the first BCD digit comes in, it is added to a cleared location (SJHOLD) in memory and stored back in that location. When the next legal character comes in, location SJHOLD is multiplied by 10, <u>then</u> added to the BCD code of the character and returned to location STORE. This sequence holds

true for a decimal number of any arbitrary length.

9.7 EXAMPLE

Since the PDP-8 can add and shift easily, the multiplication by 10 can be accomplished in three instructions. Since a shift left is equivalent to a multiplication by 2, a double shift left is equivalent to a multiplication by 4. Assume that the number currently in STORE is 5, and the new code just coming in is the number 1 stored in HOLD. The program sequence to perform the multiplication and storage is as follows:

Instruction Sequence	Comments	Contents of AC
CLA		
TAD STORE	/Load C(STORE) into AC	000 000 000 101
CLL RTL	/Multiply C(STORE) by 4	000 000 010 100
TAD STORE	/Add STORE giving C(STORE) by 5	000 000 011 001
CLL RAL	/Multiply by 2 giving C(STORE) by 10	000 000 110 010
TAD HOLD	/Add in the next number	000 000 110 011
DCA STORE	/Store back into STORE and return to wait for next character	000 000 000 000

The number residing in location STORE is 0063_8 or 0051_{10}.

If the next number to come in were "9", using the same sequence and conditions, the result would be 001 000 000 111, the binary equivalent of 519.

9.8 SCALING

This subroutine assumes an integral decimal number (signed or unsigned) and yields an integral binary equivalent (signed or unsigned respectively).

9.9 FORMAT

9.9.1 Input

The input string may or may not contain a sign (+, -, or space).

Any character other than a sign, 0 - 9, or back arrow causes the subroutine to terminate, as does a sign in any but the first position.

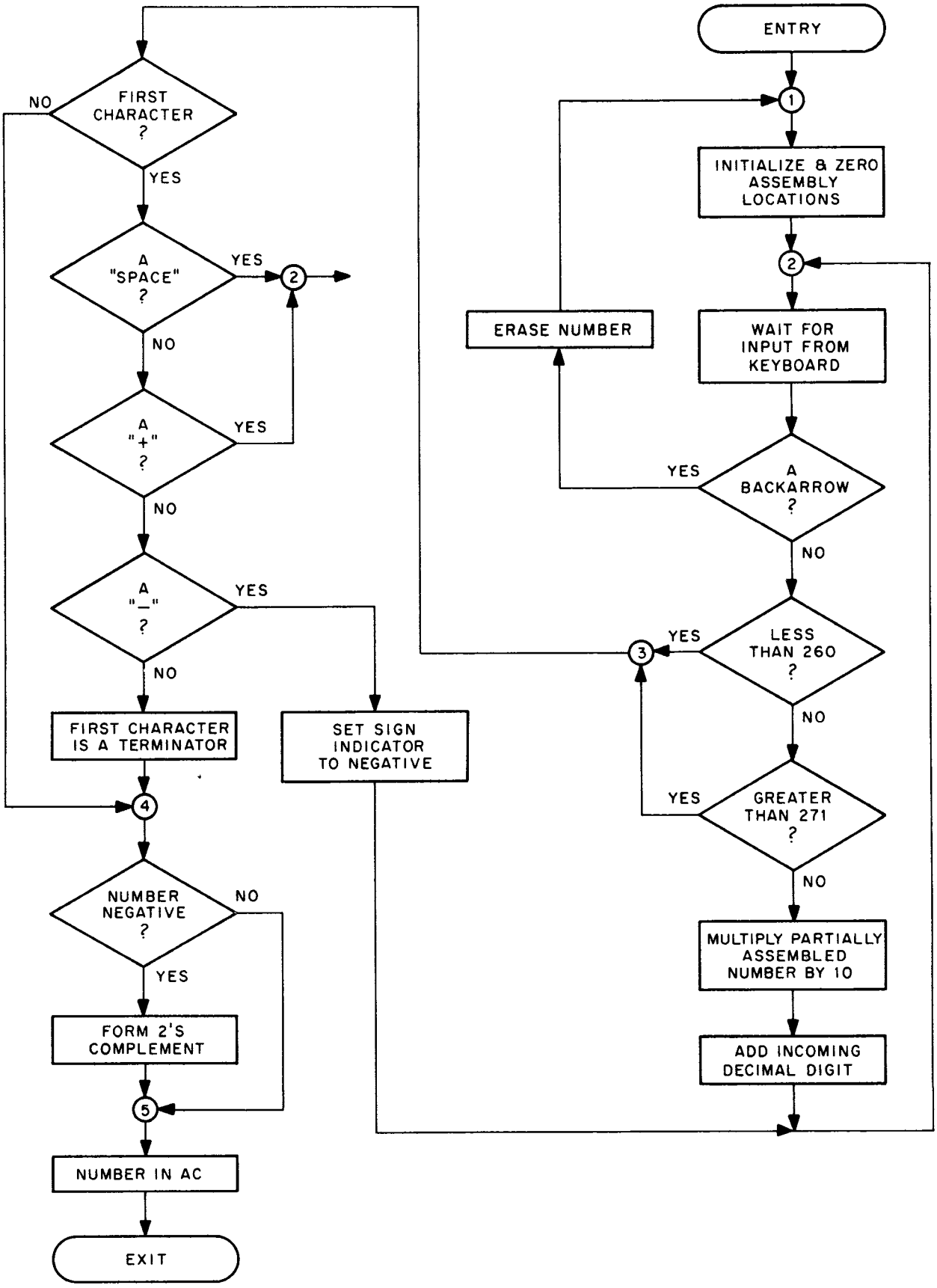
9.9.2 Core Data

The terminating character is found in location SISAVE.

9.9.3 Output

Spacing, tabulation, carriage return, etc., are not provided for in this subroutine. See Chapter 6 which contains short subroutines for those purposes.

9.10 FLOW CHART



9.11 PROGRAM LISTING

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /SINGLE PRECISION DECIMAL INPUT FROM KEYBOARD
                /CALLING SEQUENCE: JMS SICONV
                /ACC IGNORED, RETURN WITH BINARY WORD IN ACC

0200  0000  SICONV,   0
0201  7300            CLA CLL
0202  1273            TAD SISET1 +1         /INITIALIZE PROGRAM SWITCHES

0203  3232            DCA SICTRL
0204  1273            TAD SISET1 +1
0205  3224            DCA SIXSW1
0206  3310            DCA SIHOLD
0207  3311            DCA SINEG1            /CLEAR NEGATIVE SWITCH
0210  5257            JMP SINPUT
0211  3307  SIPROC,   DCA SISAVE
0212  1307            TAD SISAVE            /STORE AND THE PROCESS
                                            /CHARACTER
0213  1301            TAD SIRBUT
0214  7450            SNA                   /IS IT A "BACK-ARROW"
                                            /(IE. ERASE) KEY
0215  5201            JMP SICONV +1         /YES, REINITIALIZE
0216  1302            TAD SIM260
0217  7510            SPA                   /IS IT LESS THAN 260
                                            /(IE. "0")
0220  5232            JMP SICTRL            /YES. TRANSFER TO SEE WHAT
                                            /CHAR. IT IS
0221  1303            TAD SIM271
0222  7740            SMA SZA CLA           /IS IT GREATER THAN 271
                                            /(IE. "9")?
0223  5232            JMP SICTRL            /YES, TRANSFER TO SEE WHAT
                                            /CHAR. IT IS
0224  7300  SIXSW1,   CLA CLL               /NO, FIRST CHARACTER WAS A
                                            /DECIMAL DIGIT
0225  1231            TAD .+4               /CLOSE SWITCH TO GO TO
0226  3224            DCA .-2               /"SINMBR" NEXT
0227  1777            TAD SINMBR -1         /SET SWITCH TO SENSE
                                            /TERMINATING CHAR.
0230  3232            DCA SICTRL
0231  5246            JMP SINMBR
0232  7300  SICTRL,   CLA CLL               /CONTINUE CHECKING
0233  1307            TAD SISAVE
0234  1304            TAD SIMSPC
0235  7450            SNA                   /IS IT A SPACE?
0236  5273            JMP SISET1 +1         /YES, SET SWITCH TO SENSE
                                            /TERMINATING CHAR.
0237  1305            TAD SIMPLS
0240  7450            SNA                   /IS IT A "PLUS"?
0241  5273            JMP SISET1 +1         /YES, SET SW TO SENSE
                                            /TERMINATING CHAR.
0242  1306            TAD SIMMNS
0243  7650            SNA CLA               /IS IT A MINUS?
0244  5273            JMP SISET1            /YES, SET NEGATIVE XSWITCH
                                            /AND TERM. SWITCH
```

```
0245  5264            JMP SIEND              /NO, IT WAS A TERMINATING
                                             /CHARACTER
0246  1310  SINMBR,   TAD SIHOLD             /MULTIPLY CURRENT ASSEM-
                                             /BLED NUMBER BY 10
0247  7106            CLL RTL
0250  1310            TAD SIHOLD
0251  7004            RAL
0252  3310            DCA SIHOLD
0253  1307            TAD SISAVE             /PICK UP CURRENT DIGIT
0254  0300            AND SIMASK             /MASK OFF THE HIGH ORDER
                                             /BIT
0255  1310            TAD SIHOLD             /ADD TO ASSEMBLED NUMBER
0256  3310            DCA SIHOLD             /STORE BACK IN  SIHOLD
0257  6031  SINPUT,   KSF          /INPUT ROUTINE
0260  5257            JMP .-1
0261  6036            KRB
0262  6046            TLS
0263  5211            JMP SIPROC

                /TERMINATING ROUTINE
0264  7300  SIEND,    CLA CLL
0265  1311            TAD SINEG1
0266  7010            RAR                    /PUT NEGATIVE SWITCH INTO
                                             /LINK
0267  1310            TAD SIHOLD
0270  7430            SZL                    /IS THE LINK "1"?
0271  7041            CMA IAC                /YES, NUMBER NEGATIVE.
                                             /COMPLEMENT
0272  5600            JMP I SICONV           /RETURN.
0273  2311  SISET1,   ISZ SINEG1             /SET NEGATIVE SWITCH
0274  7300            CLA CLL
0275  1777            TAD SINMBR -1          /CLOSE SW TO TRANSFER TO
                                             /TERM.
0276  3232            DCA SICTRL
0277  5257            JMP SINPUT

                /CONSTANTS AND VARIABLES
0300  0017  SIMASK,   17
0301  7441  SIRBUT,   -337                   /CODE FOR ERASE
0302  0057  SIM260,   57                     /NUMBER USED TO GENERATE
                                             /CODE "260"
0303  7767  SIM271,   -11                    /NUMBER USED TO GENERATE
                                             /CODE "271"
0304  7540  SIMSPC,   -240                   /CODE FOR SPACE
0305  7765  SIMPLS,   -13                    /NUMBER USED TO GENERATE
                                             /CODE "253" (+)
0306  7776  SIMMNS,   -2                     /NUMBER USED TO GENERATE
                                             /CODE "255" (-)
0307  0000  SISAVE,   0                      /STORAGE LOCATIONS
0310  0000  SIHOLD,   0
0311  0000  SINEG1,   0
            PAUSE
```

Formerly
Digital-8-29-U-Sym

CHAPTER 10

DECIMAL TO BINARY CONVERSION AND INPUT

(Double Precision, Signed or Unsigned, 33-ASR)

10.1 ABSTRACT

This routine accepts and echoes a string of up to eight decimal digits (double-precision for the PDP-8) from the Teletype keyboard and converts it to the corresponding two's complement binary number.

The string may contain as legal characters a sign (+, -, or space) and the digits 0 - 9. If the first legal character is not a sign, the conversion is unsigned. A "back-arrow" (←) at any point in the string erases the current string and allows the operator to re-enter the value. Termination of input is accomplished by typing one illegal character which will then be found in location DIDSAV within the subroutine.

10.2 REQUIREMENTS

This subroutine requires 110_{10} (156_8) core locations and runs on any standard PDP-8 family computer with a 33-ASR Teletype console. The paper tape is labelled Double Precision Decimal to Binary Conversion,

Digital-8-29-U-ASCII

10.3 USAGE

10.3.1 Loading

The symbolic tape provided can be assembled with the user's main program with PAL III, MACRO-8, or PAL-D. There is neither origin setting nor terminating "$" on the symbolic tape, but a PAUSE pseudo-instruction is the last line on the tape.

10.3.2 Calling Sequence

The subroutine is called by an effective JMS to location DICONV. The location immediately following the JMS instruction contains the address of the location where the high-order portion of the number is to be stored. (It is assumed that the low-order portion of the number is in the location immediately following the high-order portion.) Return is to the second location following the calling JMS with the AC clear. For example:

```
        JMS DICONV
        ADDR
        HLT
ADDR,   Ø
        Ø
```

10.4 ERRORS IN USAGE

If the string of decimal digits is preceded by a sign (+, -, or space), the maximum decimal number that is correctly accepted is 16777215 (2^{24} -1).

If neither of these maxima is exceeded, the results are unspecified.

10.5 RESTRICTIONS

The status of AC and link is not preserved.

This subroutine should not be used with the interrupt on.

The magnitude restrictions on numbers is described in section 10.4

10.6 DESCRIPTION

The discussion, example, and scaling information about the conversion are given in Chapter 9. The only difference is that the multiplications by "4" and "2" are performed by the arithmetic shifts as described in the section on Arithmetic Shift Subroutines in the PDP-8 Math Routines writeup, DEC-08-FFAD-D.

Information on techniques used in this program can also be found in Chapter 9 of this document.

10.7 FORMAT

10.7.1 Input Data

The input string may or may not contain a sign (+, -, or space). Any character other than a sign, 0 - 9, or rubout causes the subroutine to terminate as does a sign in any but the first position.

10.7.2 Core Data

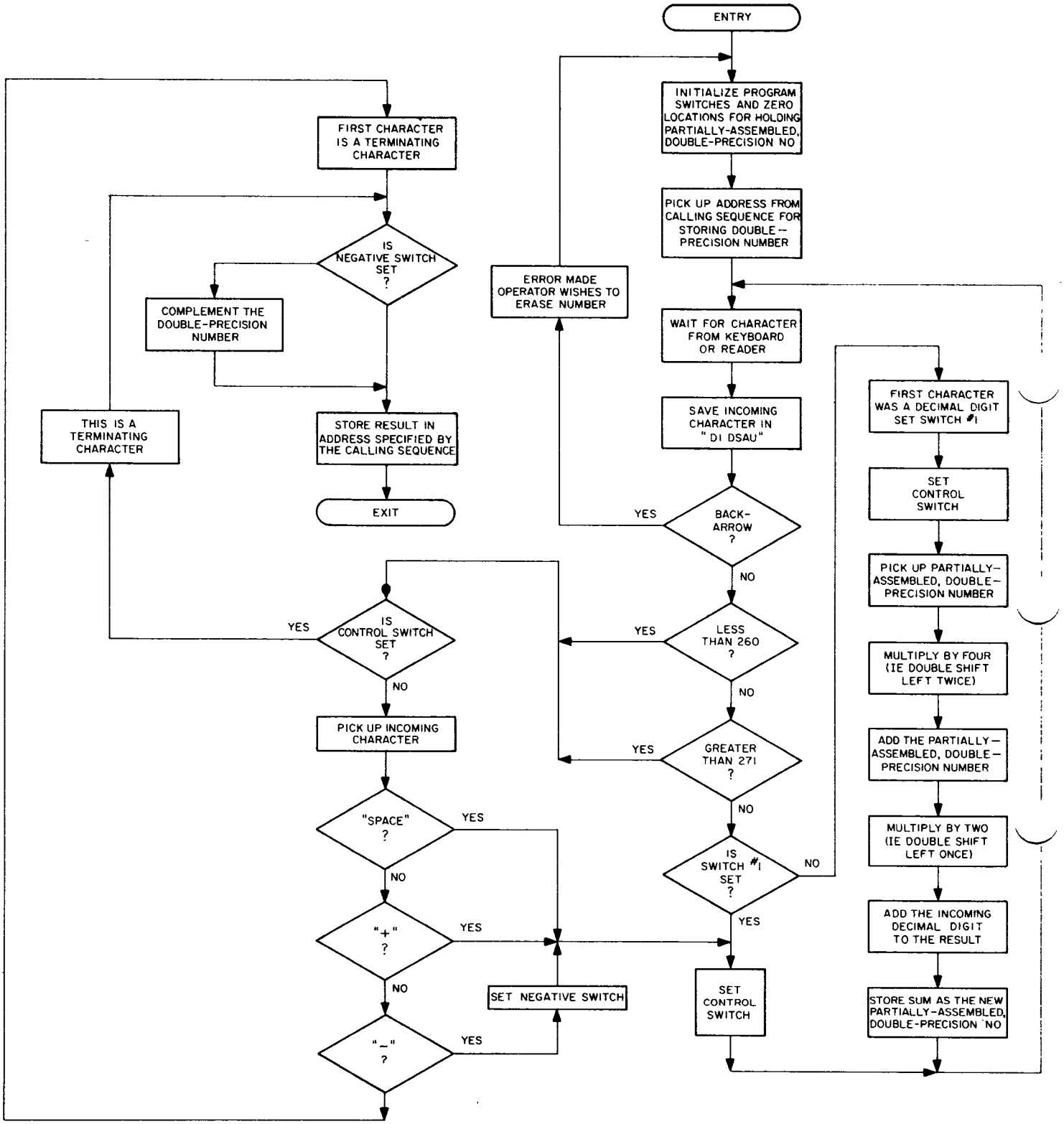
The high-order portion of the binary equivalent of the number is found in the location specified by the address following the JMS. The low-order portion is found in the next successive location. This is the format compatible with the double-precision, fixed point arithmetic subroutines. The terminating character is found in location DIDSAV.

10.7.3 Output Data

Spacing, tabulation, carriage return, etc., are not provided for in this subroutine. See Chapter 6 which contains short subroutines for such purposes.

This subroutine is input limited at a maximum of 10 cps.

10.8 FLOW CHART



10.9 PROGRAM LISTING

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /DOUBLE PRECISION DECIMAL-TO-BINARY CONVERSION AND INPUT
                /CALLING SEQUENCE:      /AC IGNORED
                /       JMS DICONV /SUBROUTINE CALLED
                /       ADDRES  /ADRESS TO STORE HIGH-ORDER WORD
                /       /LOW -ORDER WORD IN ADDRESS+1
0200  0000  DICONV, 0
0201  7300          CLA CLL /INITIALIZE PROGRAM SWITCHES
0202  1324          TAD DISET1+1
0203  3235          DCA DICTRL
0204  1324          TAD DISET1+1
0205  3227          DCA DIXSW1
0206  1600          TAD I DICONV    /PICK UP ADDRESS TO STORE
                                    /HIGH-ORDER WORD
0207  3351          DCA DIGET
0210  3352          DCA DIHIHD      /CLEAR LOCATIONS USED TO HOLD
                                    /INCOMING
0211  3353          DCA DILOHD      /NUMBER
0212  3347          DCA DINEG1      /CLEAR NEGATIVE SWITCH
0213  5275          JMP DIIN
0214  3350  DIPROC, DCA DIDSAV      /STORE CHARACTER
0215  1350          TAD DIDSAV
0216  1341          TAD DIRBUT
0217  7450          SNA     /IS IT A "BACK-ARROW"(IE. ERASE) KEY?
0220  5201          JMP DICONV+1    /YES, REINITIALIZE
0221  1342          TAD DIM260
0222  7510          SPA     /IS IT LESS THAN 260 (IE. "0")?
0223  5235          JMP DICTRL      /YES, TRANSFER TO SEE WHAT
                                    /CHARACTER IT IS
0224  1343          TAD DIM271
0225  7740          SMA SZA CLA     /IS IT GREATER THAN 271 (IE. "9")?

0226  5235          JMP DICTRL      /YES, TRANSFER TO SEE WHAT
                                    /CHARACTER IT IS
0227  7300  DIXSW1, CLA CLL /NO, FIRST CHARACTER WAS A DECIMAL DIGIT
0230  1234          TAD .+4 /CLOSE SWITCH TO GO TO "DINMBR" NEXT
0231  3227          DCA .-2
0232  1250          TAD DINMBR-1    /SET SWITCH TO SENSE TERMINATING
                                    /CHARACTER
0233  3235          DCA DICTRL
0234  5251          JMP DINMBR
0235  7200  DICTRL, CLA     /CONTINUE CHECKING TO DETERMINE CHAR.
0236  1350          TAD DIDSAV
0237  1344          TAD DIMSPC
0240  7450          SNA     /IS IT A "SPACE"?
0241  5324          JMP DISET1+1    /YES, SET SWITCH TO SENSE TERM.
                                    /CHARACTER
0242  1345          TAD DIMPLS
0243  7450          SNA     /IS IT A "PLUS"?
0244  5324          JMP DISET1+1    /YES, SET SWITCH TO SENSE TERM.
                                    /CHARACTER
```

```
0245  1346           TAD DIMMNS
0246  7650           SNA CLA /IS IT A "MINUS"?
0247  5323           JMP DISET1       /YES,SET NEGATIVE SWITCH AND TERM.
                                      /SWITCH
0250  5302           JMP DIEND        /NO, IT WAS A TERMINATING CHARACTER
0251  1353  DINMBR,  TAD DILOHD       /STORE ASSEMBLED NUMBER
                                      /TEMPORARILY
0252  3354           DCA DIXTM1
0253  1352           TAD DIHIHD
0254  3355           DCA DIXTM2
0255  4330           JMS DIDSPL       /MULTIPLY CURRENT BY "10"
0256  4330           JMS DIDSPL
0257  1353           TAD DILOHD
0260  1354           TAD DIXTM1
0261  3353           DCA DILOHD
0262  7004           RAL
0263  1352           TAD DIHIHD
0264  1355           TAD DIXTM2
0265  3352           DCA DIHIHD
0266  4330           JMS DIDSPL
0267  1350           TAD DIDSAV       /PICK UP CURRENT DIGIT
0270  0340           AND DIXMSK       /MASK OFF HIGH-ORDER BITS
0271  1353           TAD DILOHD       /ADD REMAINDER TO CURRENT NUMBER
0272  3353           DCA DILOHD
0273  7430           SZL      /DID IT OVERFLOW?
0274  2352           ISZ DIHIHD       /YES, CORRECT HIGH-ORDER WORD
            /INPUT ROUTINE
0275  6031  DIIN,    KSF
0276  5275           JMP .-1
0277  6036           KRB
0300  6046           TLS
0301  5214           JMP DIPROC
            /TERMINATING ROUTINE
0302  7200  DIEND,   CLA
0303  1347           TAD DINEG1       /PICK UP NEGATIVE NUMBER
0304  7110           CLL RAR /PUT IT INTO LINK. ("1" IF NEGATIVE)
0305  1352           TAD DIHIHD       /PICK UP HIGH ORDER PORTION
0306  7430           SZL      /IS LINK "1"?
0307  7040           CMA      /YES, NUMBER NEGATIVE. COMPLEMENT IT
0310  3751           DCA I DIGET      /STORE IT
0311  1353           TAD DILOHD       /PICK UP LOW-ORDER PORTION
0312  7430           SZL      /IS LINK "1"?
0313  7141           CLL CMA IAC      /YES, TWO'S COMP.IT. IF OVERFLOW,
                                      /LINK=1
0314  7430           SZL      /IS LINK "1"?
0315  2751           ISZ I DIGET      /INDEX HIGH-ORDER PRTION
0316  7000           NOP      /TAKES CARE WHEN HIGH-ORDER PORTION =0
0317  2351           ISZ DIGET        /INDEX POINTER FOR LOW-ORDER
                                      /PORTION
0320  3751           DCA I DIGET      /STORE LOW-ORDER POTION OF NUMBER
0321  2200           ISZ DICONV       /INDEX FOR CORRECT RETURN
0322  5600           JMP I DICONV     /RETURN
0323  2347  DISET1,  ISZ DINEG1       /SET NEGATIVE SWITCH
```

```
0324  7300          CLA CLL /CLOSE SWITCH TO TRANSFER TO TERMINATION
0325  1250          TAD DINMBR-1
0326  3235          DCA DICTRL
0327  5275          JMP DIIN        /JUMP TO WAIT FOR NEXT CHARACTER
              /DOUBLE PRECISION LEFT SHIFT (X2)
0330  0000  DIDSPL, 0
0331  1353          TAD DILOHD
0332  7104          CLL RAL
0333  3353          DCA DILOHD
0334  1352          TAD DIHIHD
0335  7004          RAL
0336  3352          DCA DIHIHD
0337  5730          JMP I DIDSPL
              /CONSTANTS AND VARIABLES
0340  0017  DIXMSK, 17      /MASK FOR LAST FOUR BITS
0341  7441  DIRBUT, -337    /CODE FOR ERASE
0342  0057  DIM260, 57      /NUMBER USED TO GENERATE CODE "260"
0343  7767  DIM271, -11     /NUMBER USED TO GENERATE CODE "271"
0344  7540  DIMSPC, -240    /CODE FOR SPACE
0345  7765  DIMPLS, -13     /NUMBER USED TO GENERATE CODE "253" (+)
0346  7776  DIMMNS, -2      /NUMBER USED TO GENERATE CODE "255" (-)
0347  0000  DINEG1, 0       /STORAGE LOCATIONS
0350  0000  DIDSAV, 0
0351  0000  DIGET,  0
0352  0000  DIHIHD, 0
0353  0000  DILOHD, 0
0354  0000  DIXTM1, 0
0355  0000  DIXTM2, 0
```

Formerly
Digital-8-10-U-Sym

CHAPTER 11
BCD TO BINARY CONVERSION, SINGLE PRECISION
(Binary Coded Decimal to Binary Conversion Subroutine)

11.1 ABSTRACT

This chapter presents a basic subroutine for converting binary-coded-decimal numbers to their equivalent binary value. Conversion is accomplished by "radix deflation".

11.2 REQUIREMENTS

This subroutine requires 23_{10} (27_8) memory locations and runs on any standard PDP-8 family computer with a 33-ASR Teletype console. The source paper tape is labeled BCD to Binary Conversion,

Digital-8-10-U-ASCII.

11.3 USAGE

11.3.1 Loading

Load the subroutine with the Binary or RIM Loader, as described in either Introduction to Programming or Programming Languages.

11.3.2 Calling Sequence

Call with the number to be converted in the AC. Return will be to the location following the calling JMS with the result in the AC.

11.4 DESCRIPTION

The method used is that of "radix deflation". Upon entry, the BCD number may be considered to be in the following form:

(a) $D_2\ 16^2 + D_1\ 16 + D_0$

What is desired is the number in the form:

(b) $D_2\ 10^2 + D_1\ 10 + D_0$

The PDP-8 can shift (rotate) and add. A right shift is equivalent to a division by a power of two. An appropriate series of shifts, additions, and subtractions is used to convert the number from the form of (a) to that of (b).

11.5 EXAMPLE

Consider the BCD number

0101 0001 1001

representing the decimal number 519.

First the whole number is stored and then brought back into the AC. Next, the four most significant bits are masked out. At this point, the accumulator contains 16x16xA or

0101 0000 0000

A shift to the right of one bit yields

0010 1000 0000

This number is stored and then brought back to the AC, shifted right two bits, and the stored value added as follows:

```
0000    1010    0000
0010    1000    0000
0011    0010    0000
```

Now the original number is added to this result

```
0011    0010    0000
0101    0001    1001
1000    0011    1001
```

and the most significant eight bits masked out as

1000 0011 0000

This is stored, brought back and shifted right once, and the stored value added.

0100	0001	1000
1000	0011	0000
1100	0100	1000

Next the result of this addition is shifted right two places dividing the number by four as follows:

0011	0001	0010

negated and the original number added

1100	1110	1110
0101	0001	1001
0010	0000	0111

This result represents in binary 512 plus 4 plus 2 plus 1 or 519, the original number.

11.6 SCALING

This subroutine assumes an integral BCD number and yields an integral binary equivalent.

11.7 PROGRAM LISTING

A listing of the subroutine with BCDBIN located at 0200 is given below. To simplify mnemonics D_2, D_1, and D_0 have been replaced respectively with A, B, and C.

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS

                /BINARY-CODED-DECIMAL TO BINARY CONVERSION SUBROUTINE
                *263

0263  0000  BCDBIN, 0
0264  3314          DCA TEMPH               /STORE INPUT
0265  1314          TAD TEMPH
0266  0311          AND LDIGIT
0267  7112          CLL RTR
0270  3313          DCA COUNT
0271  1313          TAD COUNT
0272  7010          RAR
0273  1313          TAD COUNT
0274  7041          CMA IAC
0275  1314          TAD TEMPH               /160 H + 16 M + L
0276  3314          DCA TEMPH
0277  1314          TAD TEMPH
0300  0312          AND MDIGIT
0301  7112          CLL RTR
0302  3313          DCA COUNT
0303  1313          TAD COUNT
0304  7010          RAR
0305  1313          TAD COUNT               /60 H + 6 M
0306  7041          CMA IAC
0307  1314          TAD TEMPH
0310  5663          JMP I BCDBIN    /EXIT
0311  7400  LDIGIT, 7400
0312  7760  MDIGIT, 7760
0313  0000  COUNT,  0
0314  0000  TEMPH,  0
```

11.8 REFERENCES

11.8.1 DECUS Programs

See DECUSOPE January 1965, article entitled "Accelerated Radix Deflation on the PDP-7 and PDP-8".

11.8.2 ACKNOWLEDGMENTS

Mr. Donald V. Weaver, Consultant, of New York City, who first described the algorithm used by this subroutine in reference 11.8.1 has granted his kind permission to include this subroutine in the PDP-8 library so that a detailed description may be available.

CHAPTER 12

BCD TO BINARY CONVERSION, DOUBLE PRECISION
(Binary Coded Decimal to Binary Conversion Subroutine)

12.1 ABSTRACT

This subroutine converts a 6-digit BCD number to its equivalent binary value in two computer words.

12.2 REQUIREMENTS

This subroutine requires 89_{10} (131_8) memory locations and runs on any standard PDP-8 with a 33-ASR Teletype console. The source tape is labeled Double Precision BCD to Binary Conversion,

Digital-8-11-U-ASCII

12.3 USAGE

12.3.1 Loading

The subroutine is loaded with the Binary Loader. The symbolic code is either assembled with the user program or separately with the proper origin setting.

12.3.2 Calling Sequence

This subroutine is called with an effective JMS DOUBLE followed by the address of the high-order word of the double-precision BCD number. Control is returned to the following location with the high-order part of the result in C(AC) and with the low-order part of the result in C(LOW).

12.4 DESCRIPTION

Upon entry, the BCD number is in the form:

$$(16^2D_1 + 16D_2 + D_3);(16^2D_4 + 16D_5 + D_6)$$

(each digit is 4 bits, $2^4 = 16$)

Using the single precision BCD to binary subroutine, this is reduced to:

$$(10^2D_1 + 10D_2 + D_3);(10^2D_4 + 10D_5 + D_6)$$

The high order part of the BCD word is effectively multiplied by 1000 (=8(128 - 3)) and the low-order part is added, giving

$$10^5D_1 + 10^4D_2 + 10^3D_3 + 10^2D_4 + 10D_5 + D_6.$$

See Chapter 11.

12.5 EXAMPLES

```
GO,     JMS I X
        HIGH
        HLT
X,      DOUBLE
HIGH,   1001    1001    1001    1    999,999
LOW,    1001    1001    1001
```

If this program were started at GO, the C(AC) at the halt would be 0364_8 and C(LOW) would be 1077_8, i.e., $03641077_8 = 999{,}999_{10}$.

12.6 PROGRAM LISTING

```
/COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
/MAYNARD, MASSACHUSETTS
/DIGITAL  8-11-U-SYM

/DOUBLE PRECISION BCD TO BINARY CONVERSION
/CALLING SEQUENCE:
/       JMS DOUBLE
/       ADDRESS OF HIGH ORDER ARGUMENT
/       RETURN:      C(AC)=HIGH ORDER PART
/                    C(LOW) = LOW ORDER PART

/ALSO CONTAINS SINGLE PRECISION BCD TO BINARY
/CALLING SEQUENCE:
/       C(AC) = 3 BCD CHARACTERS
/       JMS BCDBIN
/       RETURN:      ANSWER IN C(AC)
```

```
0200  0000  DOUBLE, 0
0201  7300         CLA CLL
0202  1600         TAD I DOUBLE     /FETCH ADDRESS
0203  3271         DCA LOW1                 /STORE
0204  2200         ISZ DOUBLE               /INCREMENT RETURN
0205  1671         TAD I LOW1       /FETCH HIGH ORDER
0206  4275         JMS BCDBIN       /CONVERT IT
0207  3272         DCA HIGH1                /STORE
0210  2271         ISZ LOW1                 /INCREMENT POINTER
0211  1671         TAD I LOW1       /FETCH LOW ORDER
0212  4275         JMS BCDBIN       /CONVERT IT
0213  3271         DCA LOW1                 /STORE IT
0214  1272         TAD HIGH1
0215  7112         CLL RTR
0216  7012         RTR
0217  7010         RAR              /MULTIPLY HIGH ORDER
0220  3275         DCA BCDBIN       /PART BY 128
0221  1275         TAD BCDBIN
0222  0327         AND K177
0223  3274         DCA HIGH
0224  1275         TAD BCDBIN
0225  7010         RAR
0226  0325         AND K7600
0227  3273         DCA LOW
0230  1272         TAD HIGH1                /MULTIPLY HIGH ORDER
0231  7104         CLL RAL          /BY THREE
0232  1272         TAD HIGH1                /FORM 128*HIGH-3*HIGH
0233  7141         CIA CLL
0234  1273         TAD LOW
0235  3273         DCA LOW
0236  7420         SNL
0237  7040         CMA
0240  1274         TAD HIGH
0241  3274         DCA HIGH                 /125*HIGH
0242  1274         TAD HIGH                 /NOW MULTIPLY BY 8
0243  7106         CLL RTL
0244  7004         RAL
0245  0326         AND K7770                /MASK 9 BITS
0246  3274         DCA HIGH
0247  1273         TAD LOW
0250  7106         CLL RTL
0251  7004         RAL
0252  3273         DCA LOW
0253  1273         TAD LOW
0254  7004         RAL
0255  0324         AND K7           /3 BITS
0256  1274         TAD HIGH
0257  3274         DCA HIGH
0260  1273         TAD LOW
0261  0326         AND K7770                /9 BITS
0262  7100         CLL
0263  1271         TAD LOW1                 /ADD LOW ORDER PART
0264  3273         DCA LOW          /STORE LOW ORDER PART
0265  1274         TAD HIGH
0266  7430         SZL
0267  7001         IAC              /CARRY
0270  5600         JMP I DOUBLE
```

```
0271  0000  LOW1,    0
0272  0000  HIGH1,   0
0273  0000  LOW,     0
0274  0000  HIGH,    0

            /SINGLE PRECISION CONVERSION
0275  0000  BCDBIN,  0
0276  3274          DCA HIGH
0277  1274          TAD HIGH
0300  0330          AND K7400               /LEFT DIGIT
0301  7112          CLL RTR
0302  3273          DCA LOW
0303  1273          TAD LOW
0304  7010          RAR
0305  1273          TAD LOW
0306  7041          CIA
0307  1274          TAD HIGH
0310  3274          DCA HIGH
0311  1274          TAD HIGH
0312  0323          AND K7760
0313  7112          CLL RTR
0314  3273          DCA LOW
0315  1273          TAD LOW
0316  7010          RAR
0317  1273          TAD LOW
0320  7041          CIA
0321  1274          TAD HIGH
0322  5675          JMP I BCDBIN
0323  7760  K7760,  7760
0324  0007  K7,      7
0325  7600  K7600,  7600
0326  7770  K7770,  7770
0327  0177  K177,   177
0330  7400  K7400,  7400
```

CHAPTER 13

UNSIGNED DECIMAL INTEGER PRINT SUBROUTINE, SINGLE PRECISION

13.1 ABSTRACT

This subroutine permits the printing of the contents of a computer word as a 4-digit, positive, decimal integer.

13.2 REQUIREMENTS

This subroutine requires 38_{10} (46_8) core locations and runs on any standard PDP-8 family computer with a 33-ASR Teletype console. The paper tape provided is labeled Unsigned Decimal Print Subroutine,

Digital-8-22-ASCII

13.3 USAGE

13.3.1 Loading

The subroutine can be placed in core by use of the Binary Loader. See Introduction to Programming or Programming Languages for full details. The symbolic tape provided is either assembled with the user program or separately with the proper origin setting.

13.3.2 Calling Sequence

The subroutine is called by the usual JMS instruction with the number to be printed in the AC. Return to the location following that of the calling JMS.

13.4 DESCRIPTION

This is a basic subroutine used to obtain decimal output corresponding to binary words in memory. The program operates in a straightforward manner. First the binary equivalent of 1000 is subtracted from the original number until a negative result is obtained. A count is kept of the number of subtractions necessary to accomplish this, thus yielding the most significant decimal digit. This process is

repeated, using the proper power of ten, to give the three remaining decimal digits.

13.5 METHOD

This method of binary to binary-coded-decimal conversion is compact and easily understood, if not sophisticated. The latter consideration is of little consequence, since the subroutine is output limited.

13.6 OUTPUT DATA FORMAT

Output is in the form of four consecutive decimal digits. No sign is printed. Spacing, tabulation, carriage return, etc. are not provided in this subroutine.

13.7 PROGRAM LISTING

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /DIGITAL 8-22-U
                /UNSIGNED DECIMAL PRINT
                /CALL WITH NUMBER TO BE TYPED IN C(AC)
                /RETURN TO LOCATION FOLLOWING THE JMS
0200  0000  DECPRT,   0
0201  3243            DCA VALUE           /SAVE INPUT
0202  3244            DCA DIGIT           /CLEAR
0203  1235            TAD CNTRZA
0204  3245            DCA CNTRZB          /SET COUNTER TO FOUR
0205  1234            TAD ADDRZA
0206  3213            DCA ARROW           /SET TABLE POINTER
0207  7410            SKP
0210  3243            DCA VALUE           /SAVE
0211  7100            CLL
0212  1243            TAD VALUE
0213  1236  ARROW,     TAD TENPWR          /SUBTRACT POWER OF TEN
0214  7430            SZL
0215  2244            ISZ DIGIT           /DEVELOP BCD DIGIT
0216  7430            SZL
0217  5210            JMP ARROW-3         /LOOP
0220  7200            CLA                 /HAVE BCD DIGIT
0221  1244            TAD DIGIT           /GET DIGIT
0222  1242            TAD K260            /MAKE IT ASCII
0223  6041            TSF                 /OR TAD DIGIT
0224  5223            JMP .-1             /   JMS TDIGIT(SEE 8-19-U)
0225  6046            TLS                 /TYPE DIGIT
0226  7200            CLA
```

```
0227  3244             DCA DIGIT          /CLEAR
0230  2213             ISZ ARROW          /UPDATE POINTER
0231  2245             ISZ CNTRZB         /DONE ALL FOUR?
0232  5212             JMP ARROW-1        /NO: CONTINUE
0233  5600             JMP I DECPRT       /YES: EXIT
0234  1236  ADDRZA,     TAD TENPWR
0235  7774  CNTRZA,     -4
0236  6030  TENPWR,     -1750              /ONE THOUSAND
0237  7634               -0144            /ONE HUNDRED
0240  7766               -0012            /TEN
0241  7777               -0001            /ONE
0242  0260  K260,       260
0243  0000  VALUE,      0
0244  0000  DIGIT,      0
0245  0000  CNTRZB,     0
```

CHAPTER 14

SIGNED DECIMAL INTEGER PRINT SUBROUTINE, SINGLE PRECISION

14.1 ABSTRACT

This subroutine permits printing the contents of a computer word as a signed two's complement number. If bit 0 of the computer word is a "1", the remaining bits represent a negative integer in two's complement form; if bit 0 equals "0", the remaining bits represent a positive integer. If the number is negative, a minus sign is printed; if positive, a space.

14.2 REQUIREMENTS

This subroutine requires 51_{10} (63_8) core locations and runs on any standard PDP-8 family computer with a 33-ASR Teletype console. The program is provided on a source tape labeled Signed Decimal Print, Single Precision,

Digital-8-23-U-ASCII

14.3 USAGE

14.3.1 Loading

The symbolic tape provided is compatible with the PAL III, MACRO-8, or PAL-D assemblers. It can be assembled with the user's program or separately with the proper origin setting. Neither origin setting nor "$" terminating character exists on the symbolic tape provided.

14.2.1 Calling Sequence

The subroutine is called by an effective "JMS SSPRNT" with the number to be printed in the AC. The return is to the location following that of the calling JMS. The contents of neither the AC nor the link are preserved, and return is with both active registers clear.

14.4 DESCRIPTION

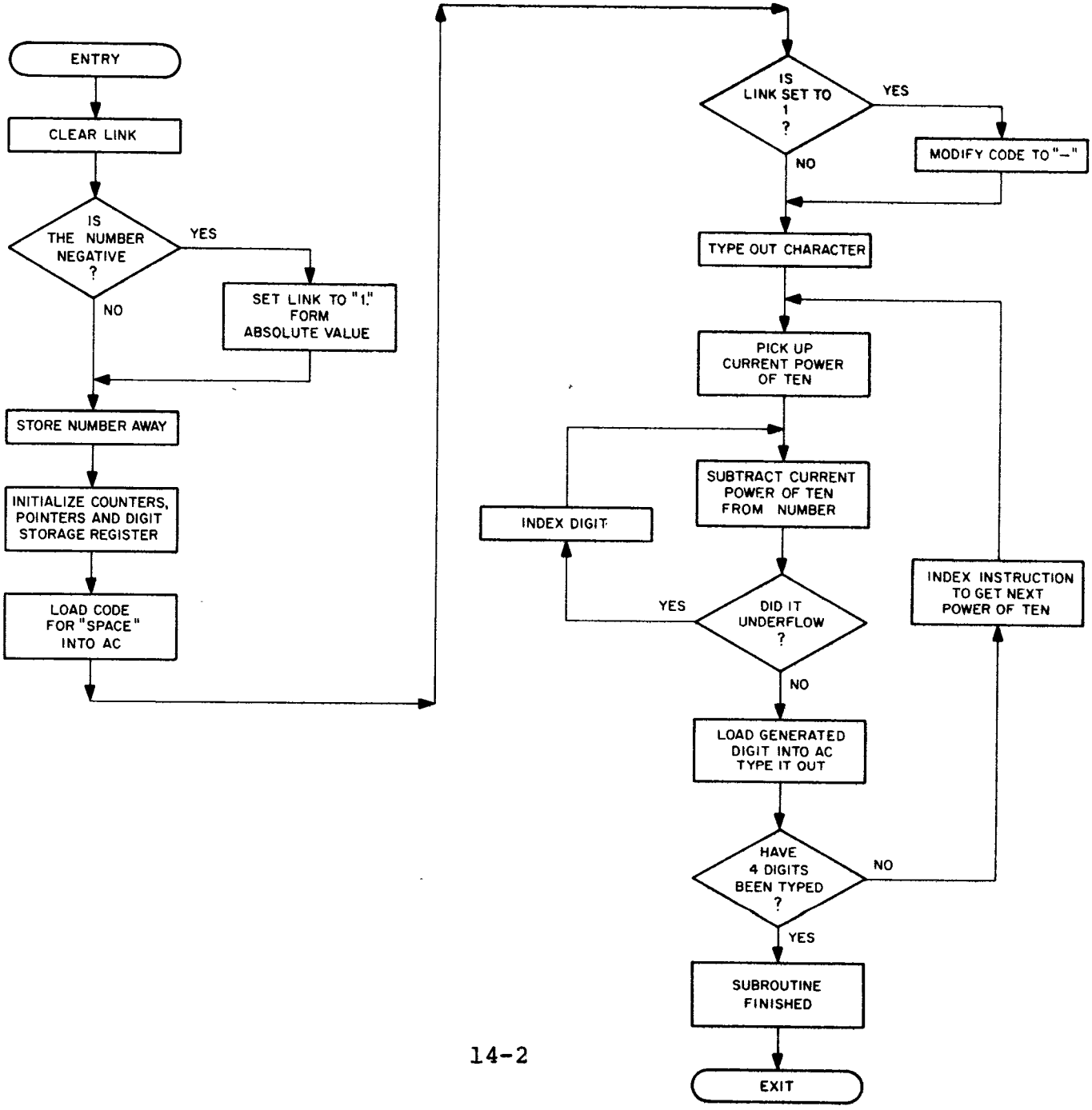
This is a basic subroutine to obtain signed decimal output (integer format) corresponding to binary words in memory stored in two's complement form. First, the number is sensed to determine if it is positive or negative. If positive, a space is printed. If

negative, a minus sign is printed and the number complemented to form the absolute value in two's complement. Then, the same algorithm is followed as in the unsigned printout described in Chapter 13.

14.5 OUTPUT DATA

Output is in the form of four consecutive decimal digits preceded by either a space or minus sign. Spacing, tabulation, carriage return, etc. are not provided in this subroutine. See Chapter 6, which contains details on subroutines for such purposes. This subroutine is output limited at 10 cps by the 33-ASR.

14.6 FLOW CHART



14.7 PROGRAM LISTING

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /BINARY TO DECIMAL CONVERSION AND TYPEOUT
                /SINGLE PRECISION
                /CALLING SEQUENCE:      /AC CONTAINS BINARY WORD
                /       JMS SSPRNT /SUBROUTINE CALL
                        /RETURN. AC AND L CLEAR
0200  0000  SSPRNT, 0
0201  7100          CLL
0202  7510          SPA     /IS IS POSITIVE?
0203  7061          CML CMA IAC     /NO, SET LINK,FORM ABSOLUTE VALUE
0204  3253          DCA SSVAL       /STORE NUMBER AWAY
0205  3251          DCA SSBOX       /SET DIGIT LOCATION TO ZERO
0206  1250          TAD SSCNTR      /INITIALIZE OUTPUT COUNTER TO "4"
0207  3252          DCA SSCNT
0210  1244          TAD SSADDR      /INITALIZE INSTRUCTION TO GET
                                    /FIRST 10
0211  3217          DCA SSXYZ+1
0212  1246          TAD SSPLUS      /GET CODE TO TYPE A "PLUS"
0213  7430          SZL     /IS THE NUMBER NEGATIVE?
0214  1247          TAD SSMNS       /YES, CHANGE CODE TF TYPE A
                                    /"MINUS"
0215  4235          JMS SSOUT       /TYPE IT OUT
0216  1253  SSXYZ,  TAD SSVAL       /PICK UP NUMBER
0217  1254          TAD SSCON       /SUBTRACT CURRENT POWER OF 10
0220  7510          SPA     /IS THE RESULT NEGATIVE?
0221  5225          JMP .+4 /YES, INDEXING IS FINISHED
0222  2251          ISZ SSBOX       /NO, INDEX THE DIGIT LOCATION
0223  3253          DCA SSVAL       /STORE REMAINDER IN SSVAL
0224  5216          JMP SSXYZ       /CONTINUE SUBTRACTING
0225  7200          CLA
0226  1251          TAD SSBOX       /PICK UP THE DIGIT NUMBER
0227  4235          JMS SSOUT       /TYPE IT OUT
0230  3251          DCA SSBOX       /SET DIGIT COUNTER TO "0"
0231  2217          ISZ SSXYZ+1     /INDEX INSTRUCTION TO GET
                                    /POWER OF 10
0232  2252          ISZ SSCNT       /HAVE WE TYPED "4" DIGITS
0233  5216          JMP SSXYZ       /NO, CONTINUE
0234  5600          JMP I SSPRNT    /YES, RETURN
                /TYPEOUT ROUTINE
0235  0000  SSOUT,  0
0236  1245          TAD SSTWO
0237  6046          TLS
0240  6041          TSF
0241  5240          JMP .-1
0242  7300          CLA CLL
0243  5635          JMP I SSOUT
0244  1254  SSADDR, TAD SSCON       /INSTRUCTION TO PICK UP FIRST
                                    /POWER OF 10
```

```
0245  0260  SSTWO,  260      /BASIC CODE FOR DIGITAL OUTPUT
0246  7760  SSPLUS, -20      /NUMBER USED TO GENERATE "SPACE"
0247  0015  SSMNS,  15       /NUMBER USED TO GENERATE "MINUS"
0250  7774  SSCNTR, -4       /COUNT OF "4" DIGITS
0251  0000  SSBOX,  0        /STORAGE REGISTERS
0252  0000  SSCNT,  0
0253  0000  SSVAL,  0
            /TABLE OF POWERS OF 10
0254  6030  SSCON,  6030     /-1000
0255  7634           7634    /-100
0256  7766           7766    /-10
0257  7777           7777    /-1
```

Formerly

Digital-8-24-U-Sym

CHAPTER 15

UNSIGNED DECIMAL INTEGER PRINT SUBROUTINE, DOUBLE PRECISION

15.1 ABSTRACT

This subroutine permits printing a double-precision integer stored in the usual convention for double-precision numbers*. The one exception is that all 24 bits are interpreted as magnitude bits (i.e., the bit "0" of the high-order word is not a sign bit). The printout is in the form of an eight-digit, positive, decimal integer.

15.2 REQUIREMENTS

This subroutine requires 73_{10} (111_8) locations and runs on any standard PDP-8 family computer with a 33-ASR Teletype console.

The source tape is labeled Unsigned Decimal Print, Double Precision,

Digital-8-24-U-ASCII

15.3 USAGE

15.3.1 Loading

The symbolic tape provided can be assembled with PAL III, MACRO-8, or PAL-D. It can be assembled with the user program or separately with the proper origin setting. Neither origin setting nor "$" terminating character exists on the tape; the tape does have a PAUSE statement on the end.

15.3.2 Calling Sequence

This subroutine is called by an effective JMS UDPRNT. The location immediately following the calling JMS contains the address

*For details on storage of double-precision numbers, see the Math Routines writeup available from the PDP-8 Program Library, section on Double Precision Signed Multiply Routine.

of the high-order portion of the double-precision integer stored in the usual double-precision format.

15.4 DESCRIPTION

This is basic double-precision subroutine used to obtain decimal output corresponding to double-precision binary words. First, the binary equivalent of 10,000,000 is subtracted from the original number until under-flow occurs. A count is kept of the number of subtractions necessary to accomplish this, thus yielding the most significant decimal digit. Then this digit is added to 2608 and printed on the 33-ASR through the AC. This process is repeated using the proper power of ten to give the seven remaining digits.

The numbers are interpreted and printed as integers.

See Chapter 13 for a discussion of the techniques used.

15.5 FORMAT

15.5.1 Core Data

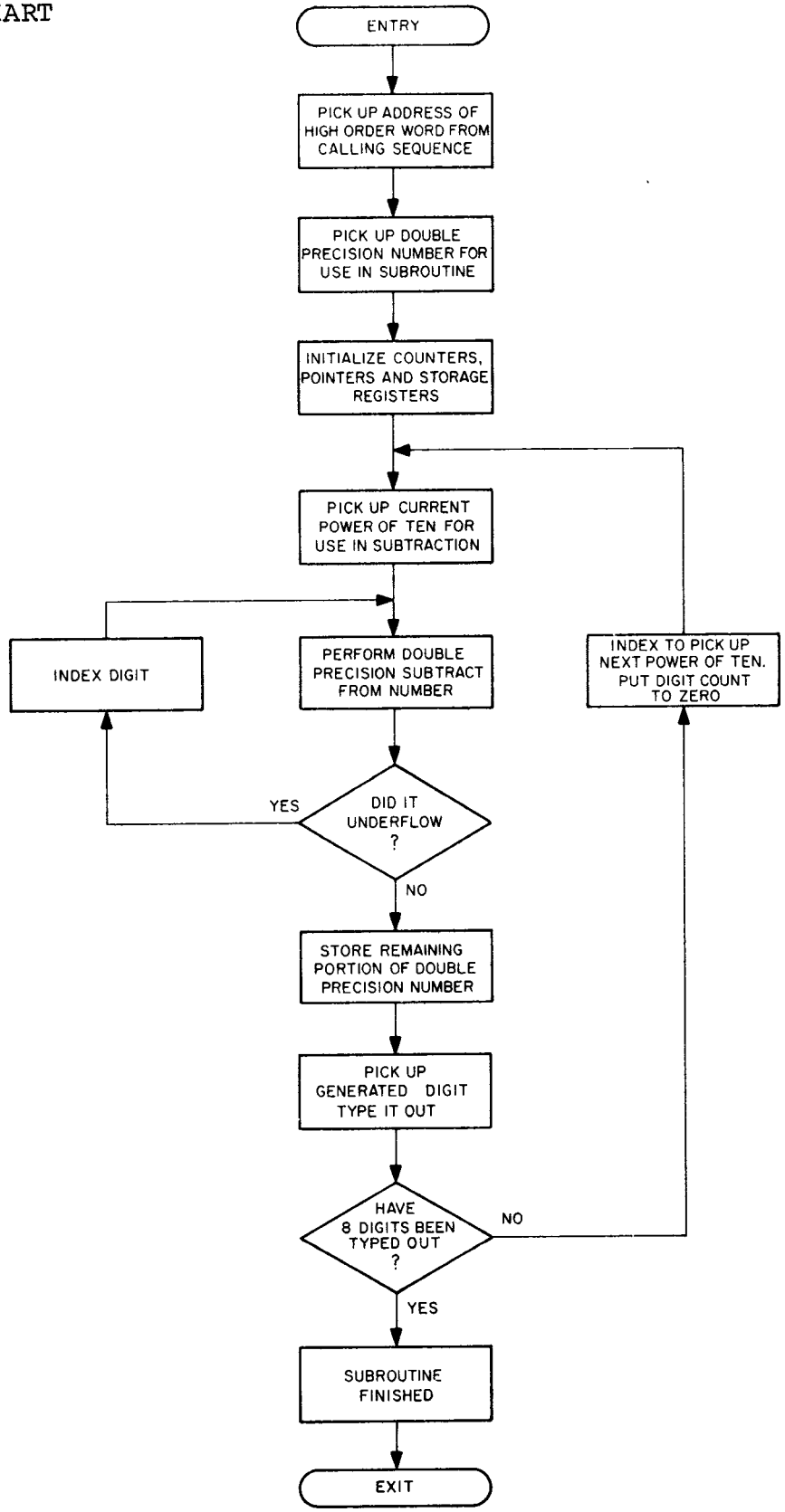
The double-precision integers are stored in the usual double-precision format, with the exception that bit "0" of the high-order word is interpreted as part of the number not a sign bit.

15.5.2 Output Data

Output is in the form of eight consecutive decimal digits. No sign is printed. Spacing, tabulation, carriage return, etc., are not provided for in this subroutine. See Chapter 6 which contains details on short subroutines for such purposes.

This subroutine is output limited at 10 cps by the 33-ASR.

15.6 FLOW CHART



15.7 PROGRAM LISTING

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /UNSIGNED DECIMAL PRINT, DOUBLE PRECISION
                /CALLING SEQUENCE:    JMS UDPRNT /SUBROUTINE CALLED
                /                     HI ADDR    /ADDRESS OF HIGH ORDER
                                                 /WORD
                /                     RETURN     /RETURN WITH AC AND L
                                                 /CLEAR

0200  0000  UDPRNT,   0
0201  7300            CLA CLL
0202  1600            TAD I UDPRNT          /PICK UP ADDRESS OF
                                            /HIGH-ORDER WORD
0203  3267            DCA UDGET
0204  1667            TAD I UDGET           /PICK UP BOTH WORDS FOR
                                            /USE IN SUBROUTINE
0205  3261            DCA UDHIGH
0206  2267            ISZ UDGET
0207  1667            TAD I UDGET
0210  3262            DCA UDLOW
0211  1255            TAD UDLOOP            /INITIALIZE DIGIT COUNTER
0212  3260            DCA UDCNT             /FOR "8"
0213  1256            TAD UDADDR            /INITIALIZE TO TABLE OF
0214  3270            DCA UDPTR             /POWERS OF TEN
0215  2200            ISZ UDPRNT            /INDEX LINKAGE FOR CORRECT
                                            /RETURN
0216  1670  UDARND,   TAD I UDPTR           /PICK UP CURRENT
                                            /POWER OF TEN FOR
0217  2270            ISZ UDPTR             /USE IN SUBTRACTION
0220  3263            DCA UDHSUB
0221  1670            TAD I UDPTR
0222  2270            ISZ UDPTR
0223  3264            DCA UDLSUB
0224  7100  UDDO,     CLL                   /DOUBLE PRECISION
                                            /SUBTRACTION
0225  1264            TAD UDLSUB
0226  1262            TAD UDLOW
0227  3266            DCA UDTEML
0230  7004            RAL
0231  1263            TAD UDHSUB
0232  1261            TAD UDHIGH
0233  7420            SNL                   /DID IT UNDERFLOW?
0234  5242            JMP UDOUT             /NO, COUNT IS DONE
0235  2265            ISZ UDBOX             /YES, COUNT NOT DONE YET.
                                            /INDEX DIGIT
0236  3261            DCA UDHIGH            /DEPOSIT REMAINING PORTIONS
                                            /OF WORD
```

```
0237  1266            TAD UDTEML
0240  3262            DCA UDLOW
0241  5224            JMP UDDO              /GO BACK AND SUBTRACT
                                            /AGAIN
0242  7200  UDOUT,     CLA
0243  1265            TAD UDBOX             /PICK UP RESULTING DIGIT
0244  1257            TAD UDTWO             /ADD "260" TO IT
0245  6046            TLS                   /TYPE IT OUT
0246  6041            TSF
0247  5246            JMP .-1
0250  7300            CLA CLL
0251  3265            DCA UDBOX             /INITIALIZE DIGIT TO "0"
0252  2260            ISZ UDCNT             /HAVE WE TYPED "8" DIGITS.
0253  5216            JMP UDARND            /NO, DETERMINE NEXT DIGIT
0254  5600            JMP I UDPRNT          /YES, SUBROUTINE DONE.
                                            /RETURN
0255  7770  UDLOOP,    -10                   /COUNT OF "8" DIGITS
0256  0271  UDADDR,    UDCON1                /INITIAL ADDRESS OF
                                            /POWERS OF TEN
0257  0260  UDTWO,     260                   /ICODE  FOR DIGITS
0260  0000  UDCNT,     0                     /STORAGE LOCATIONS
0261  0000  UDHIGH,    0
0262  0000  UDLOW,     0
0263  0000  UDHSUB,    0
0264  0000  UDLSUB,    0
0265  0000  UDBOX,     0
0266  0000  UDTEML,    0
0267  0000  UDGET,     0
0270  0000  UDPTR,     0
0271  3166  UDCON1,    3166                  /POWERS OF TEN
0272  4600            4600                  /-10,000,000
0273  7413            7413                  /-1,000,000
0274  6700            6700
0275  7747            7747                  /-100,000
0276  4540            4540
0277  7775            7775                  /-10,000
0300  4360            4360
0301  7777            7777                  /-1,000
0302  6030            6030
0303  7777            7777                  /-100
0304  7634            7634
0305  7777            7777                  /-10
0306  7766            7766
0307  7777            7777                  /-1
0310  7777            7777
      PAUSE
```

CHAPTER 16

SIGNED DECIMAL INTEGER PRINT SUBROUTINE, DOUBLE PRECISION

16.1 ABSTRACT

This subroutine permits printing the contents of two consecutive computer words as one signed double-precision two's complement number. If bit 0 of the high order word is a "1", the remaining 23 bits represent a negative integer in two's complement form; if bit 0 equals "0", the remaining bits represent a positive integer. If the number is negative, a minus sign is printed; if positive, a space.

16.2 REQUIREMENTS

This subroutine requires 86_{10} (126_8) core locations and runs on any standard PDP-8 family computer with a 33-ASR Teletype console.

The source tape supplied is labeled Signed Decimal Print Double Precision,

Digital-8-25-U-ASCII

16.3 USAGE

16.3.1 Loading

The symbolic tape provided is compatible with PAL III, MACRO-8, and PAL-D assemblers. It can be assembled with the user's program or separately with the proper origin setting. Neither origin setting nor "$" terminating character exists on the symbolic tape provided, but a PAUSE pseudo-instruction is the last line on tape.

16.3.2 Calling Sequence

The subroutine is called by an effective JMS SDPRNT. The location immediately following the calling JMS contains the address of the high-order portion of the signed, double-precision integer which is stored in the usual double-precision format. For example:

```
        JMS   SDPRNT
        ADDR
        HLT
ADDR,   Ø123
        4567
```

16.4 DESCRIPTION

This is a basic subroutine to obtain signed decimal output corresponding to a double-precision binary word storage in two consecutive locations in memory. First, the binary number is sensed to determine if it is positive or negative. If positive, a space is printed. If negative, a minus sign is printed, and the number complemented to form the absolute value. Then the same algorithm is followed as in the unsigned double-precision printout described in Chapter 15.

The numbers are interpreted and printed as integers.

16.5 FORMAT

16.5.1 Core Data

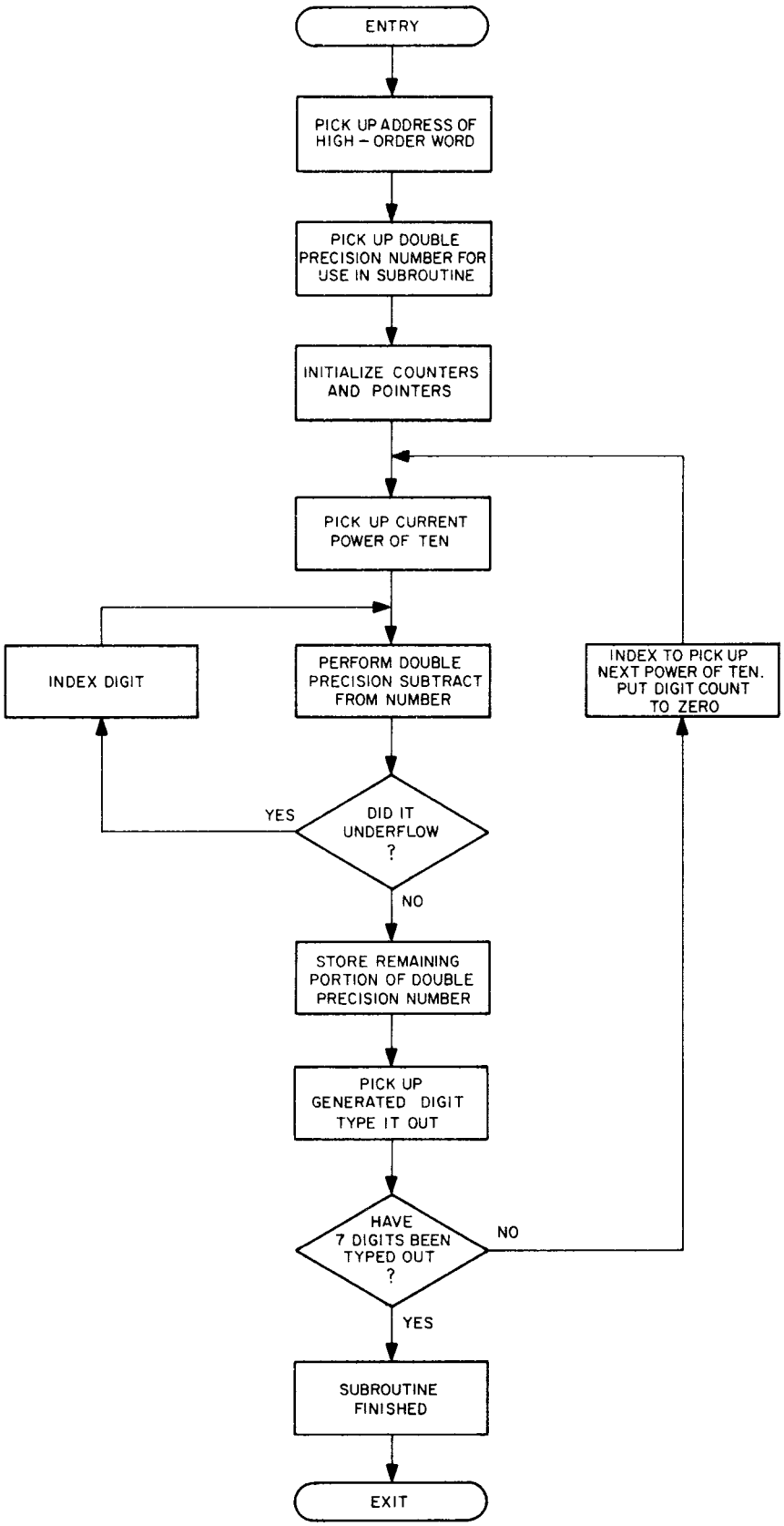
The double precision integers are stored in the usual signed, double-precision format (see the Double Precision Signed Multiply section of the Math Routines writeup, available from the PDP-8 Program Library.

16.5.2 Output Data

Output is in the form of seven consecutive decimal digits preceded by either a space or a minus sign. Spacing, tabulation, carriage return, etc., are not provided in this subroutine. See Chapter 6 which contains details on subroutines for such purposes. If the user wishes to print a "+" sign instead of a space, he can change the contents of location SDPLUS from "-15" to "-2".

This subroutine is output limited at 10 cps by the 33-ASR.

16.6 FLOW CHART



16.7 PROGRAM LISTING

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /SIGNED DECIMAL PRINT, DOUBLE PRECISION
                /CALLING SEQUENCE:      JMS SDPRNT /SUBROUTINE CALLED
                /       HIADDR  /ADDRESS OF HIGH ORDER WORD
                /       RETURN  /RETURN WITH AC AND L CLEAR
0200  0000  SDPRNT, 0
0201  7300          CLA CLL
0202  1600          TAD I SDPRNT    /PICK UP ADDRESS OF
                                    /HIGH-ORDER WORD
0203  3307          DCA SDGET
0204  1707          TAD I SDGET     /PICK UP HIGH-ORER WORD
0205  7700          SMA CLA /IS IT NEGATIVE?
0206  1276          TAD SDPLUS      /NO, GENERATE CODE FOR SPACE
0207  1277          TAD SDMNS       /YES, GENERATE CODE FOR "MINUS"
0210  4264          JMS SDTYPE      /TYPE IT OUT
0211  1707          TAD I SDGET     /PICK UP HIGH-ORDER WORD AGAIN
0212  7510          SPA     /IS IT POSITIVE?
0213  7060          CMA CML /NO, COMPLEMENT IT. SET LINK
0214  3301          DCA SDHIGH      /STORE POSITIVE WORD FOR USE IN
                                    /SUBROUTINE
0215  2307          ISZ SDGET
0216  1707          TAD I SDGET     /PICK UP LOW-ORDER WORD
0217  7430          SZL     /IS LINK SET?
0220  7141          CMA CLL IAC     /YES, FORM TWO'S COMPLEMENT
0221  7430          SZL     /DID AC OVERFLOW FROM "IAC"?
0222  2301          ISZ SDHIGH      /YES, CORRECT HIGH-ORDER WORD
0223  3302          DCA SDLOW       /STORE POSITIVE LOW-ORDER WORD
0224  1273          TAD SDLOOP      /INITIALIZE DIGIT COUNTER TO "7"
0225  3300          DCA SDCNT
0226  1274          TAD SDADDR      /INITIALIZE POINTER TO TABLE OF
                                    /POWERS OF TEN
0227  3310          DCA SDPTR
0230  2200          ISZ SDPRNT      /INDEX LINKAGE FOR CORRECT RETURN
0231  1710  SDARND, TAD I SDPTR     /PICK UP POWER OF TEN FOR USE IN
                                    /SUBTRACT
0232  2310          ISZ SDPTR
0233  3303          DCA SDHSUB
0234  1710          TAD I SDPTR
0235  2310          ISZ SDPTR
0236  3304          DCA SDLSUB
0237  7100  SDDO,    CLL     /DOUBLE PRECISION SUBTRACTION
0240  1304          TAD SDLSUB
0241  1302          TAD SDLOW
0242  3306          DCA SDTEML
0243  7004          RAL
0244  1303          TAD SDHSUB
0245  1301          TAD SDHIGH
0246  7510          SPA     /DID IT UNDERFLOW?
0247  5255          JMP SDOUT       /NO, COUNT IS DONE
0250  2305          ISZ SDBOX       /YES, COUNT NOT DONE. INDEX DIGIT
```

```
0251  3301           DCA SDHIGH      /DEPOSIT REMAINING HIGH-ORDER
                                     /PORTION
0252  1306           TAD SDTEML      /RESTORE REMAINING LOW-ORDER
                                     /PORTION
0253  3302           DCA SDLOW
0254  5237           JMP SDDO        /GO BACK AND SUBTRACT AGAIN
0255  7200  SDOUT,    CLA
0256  1305           TAD SDBOX       /PICK UP RESULTING DIGIT
0257  4264           JMS SDTYPE      /TYPE IT OUT
0260  3305           DCA SDBOX       /INITIALIZE DIGIT TO "0"
0261  2300           ISZ SDCNT       /HAVE WE TYPED "7" DIGITS
0262  5231           JMP SDARND      /NO, DETERMINE NEXT DIGIT
0263  5600           JMP I SDPRNT    /YES, SUBROUTINE DONE. RETURN
0264  0000  SDTYPE,   0       /TYPEOUT ROUTINE
0265  1275           TAD SDTWO
0266  6046           TLS
0267  6041           TSF
0270  5267           JMP .-1
0271  7300           CLA CLL
0272  5664           JMP I SDTYPE
0273  7771  SDLOOP,   -7      /COUNT OF SEVEN DIGITS
0274  0311  SDADDR,   SDCONL  /INITAL ADDRESS OF POWERS OF TEN
0275  0260  SDTWO,    260     /BASIC CODE FOR DIGITS
0276  7763  SDPLUS,   -15     /"SPACE". TO TYPE "+",REPLACE BY "-2"
0277  7775  SDMNS,    -3      /"MINUS"
0300  0000  SDCNT,    0       /STORAGE LOCATIONS
0301  0000  SDHIGH,   0
0302  0000  SDLOW,    0
0303  0000  SDHSUB,   0
0304  0000  SDLSUB,   0
0305  0000  SDBOX,    0
0306  0000  SDTEML,   0
0307  0000  SDGET,    0
0310  0000  SDPTR,    0
0311  7413  SDCONL,   7413    /TABLE OF POWERS OF TEN
0312  6700           6700    /-1,000,000
0313  7747           7747    /-100,000
0314  4540           4540
0315  7775           7775    /-10,000
0316  4360           4360
0317  7777           7777    /-1,000
0320  6030           6030
0321  7777           7777    /-100
0322  7634           7634
0323  7777           7777    /-10
0324  7766           7766
0325  7777           7777    /-1
0326  7777           7777
```

Formerly
Digital-8-14-U-Sym

CHAPTER 17

BINARY TO BCD CONVERSION
(Binary to Binary Coded Decimal Conversion)

17.1 ABSTRACT

This subroutine provides the basic means of converting binary data to binary-coded-decimal (BCD) data for printing, magnetic tape recording, etc.

17.2 REQUIREMENTS

This subroutine uses 33_{10} (41_8) storage locations and runs on any standard PDP-8 family computer with a 33-ASR Teletype console. The source tape provided is labeled Binary to Binary Coded Decimal Conversion,

Digital-8-14-U-ASCII

17.3 CALLING SEQUENCE

The subroutine is called by the JMS instruction. When called, the binary number to be converted must be in the accumulator (AC).

The subroutine returns to the instruction immediately following the calling JMS with the BCD result in the AC.

17.4 DESCRIPTION

Reference to the Flow Chart (Figure 17.1) illustrates this discussion.

On input the binary number is stored, a pointer is initialized, the link is cleared, and a counter to control the number of passes through the computation loop proper is properly set.

The loop is now entered, and is repeated eight times.

The binary equivalents of 800, 400, 200, 100, 80, 40, 20, and 10 are subtracted successively from the original binary number.

After each subtraction, a test on the link is made. If the result of the test shows the link to be 0, the next lower equivalent is subtracted from the same quantity after the contents of the links (0) are shifted into the developing BCD number (Location NUMBER).

If the subtraction leaves a negative link, the contents of the accumulator replace the binary representation currently being processed after the contents of the link (1) have been shifted into the growing BCD number.

After eight passes through the basic loop, the developed BCD representation is shifted left four bits and the "residual" least significant digit is added before exit.

17.5 EXAMPLE

As an example consider the conversion of the binary equivalent of 512 decimal:

Link	Addition	
	001 000 000 000	
	110 011 100 000	-800
0	111 011 100 000	
	001 000 000 000	
	111 001 110 000	-400
1	000 001 110 000	
	000 001 110 000	
	111 110 001 000	-200
0	111 111 111 000	
	000 001 110 000	
	111 110 011 100	-100
1	000 000 001 100	

Notice that the remainder is the binary representation of 12 decimal. Writing the link bits in the order they are developed gives 0101 the BCD character denoting 5.

17.6 SCALING

The original binary number must be no larger than 999 (decimal) which is equivalent to 1747 (octal). The binary point is assumed to be at the extreme right end of the word (to the right of bit position 11) and the decimal point is also so positioned.

In other words, this subroutine converts binary integers to BCD integers.

Note that the subroutine is designed for positive input only!

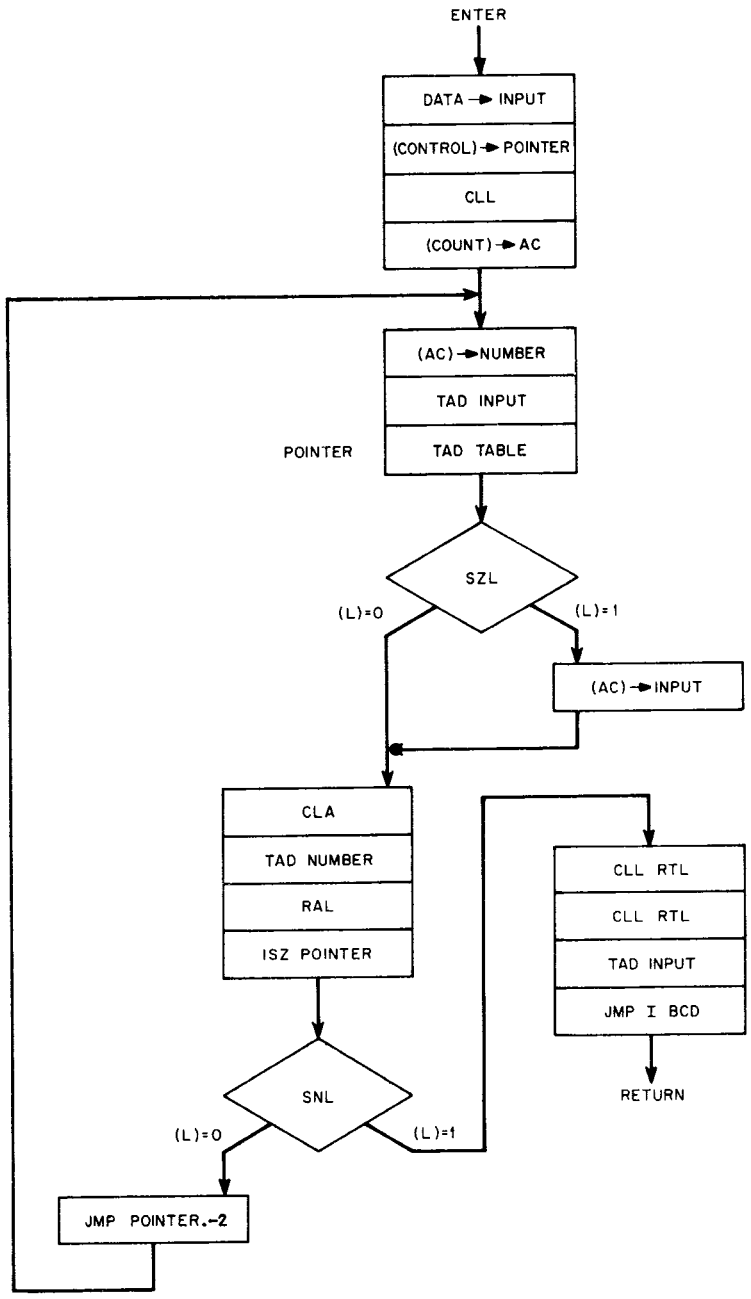


Figure 17.1 FLOW CHART

17.7 PROGRAM LISTING

A listing of the program with BCD located in 0200 appears as follows:

```
                /COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
                /MAYNARD, MASSACHUSETTS
                /BINARY TO BCD CONVERSION 3/6/65-HB-DEC
                /ENTER WITH BINARY NUMBER (<999(10))
                /IN ACCUMULATOR; EXIT WITH THREE CHARACTER
                /BCD NUMBER IN ACCUMULATOR
                /AC 0-3; AC 4-7; AC 8-11 WILL CONTAIN
                /THE BCD CHARACTER ON EXIT
                /WEIGHTING:           AC 0-3  100
                /                     AC 4-7   10
                /                     AC 8-11   1
                /STORAGE    33(10) REGISTERS
                /TIME=216.0-235.2 MICRO-SECONDS PDP-8
                /IF INPUT >999 (10) RESULT IS UNSPECIFIED
0200  0000  BCD,      0
0201  3226            DCA INPUT             /STORE BINARY
0202  1225            TAD CONTRL            /SET UP TABLE
0203  3210            DCA POINTR            /POINTERS
0204  7100            CLL
0205  1230            TAD COUNT             /SET BIT 7=1; 8RAL'S
0206  3227            DCA NUMBER            /WILL PUT IT IN LINK
0207  1226            TAD INPUT
0210  1231  POINTR,   TAD TABLE             /OR TABLE+1, TABLE+2, ETC.

0211  7430            SZL                   /IF C(L)=1, INPUT>-TABLE
0212  3226            DCA INPUT             /IF SO: INPUT=INPUT+TABLE
0213  7200            CLA
0214  1227            TAD NUMBER
0215  7004            RAL                   /PUT THIS BIT IN ANSWER
0216  2210            ISZ POINTR            /UPDATE TABLE POINTER
0217  7420            SNL                   /IF LINK=1, ALL DONE
0220  5206            JMP POINTR-2
0221  7106            CLL RTL               /CONVERTED 2 BCD
0222  7006            RTL                   /CHARACTERS
0223  1226            TAD INPUT             /SHIFT LEFT AND ADD
0224  5600            JMP I BCD             /THE THIRD
0225  1231  CONTRL,   TAD TABLE
0226  0000  INPUT,    0
0227  0000  NUMBER,   0
0230  0020  COUNT,    0020
0231  6340  TABLE,    -1440                 /-800(10)
0232  7160            -0620                 /-400
0233  7470            -0310                 /-200
0234  7634            -0144                 /-100
0235  7660            -0120                 /-80
0236  7730            -0050                 /-40
0237  7754            -0024                 /-20
0240  7766            -0012                 /-10
                /EXAMPLE: INPUT  0726  (8)
                /         OUTPUT   0100/0111/0000   = 470 (10)
```

CHAPTER 18

BINARY TO BCD CONVERSION (4-DIGIT)
(Binary to Binary Coded Decimal Conversion, 4-Digit)

18.1 ABSTRACT

This subroutine extends the method used in Chapter 17 so that binary integers from 0 to 4095 contained in a single computer word may be converted to four binary-coded-decimal characters packed in two computer words.

18.2 REQUIREMENTS

This subroutine uses 53_{10} (65_8) storage locations and runs on any standard PDP-8 family computer with 33-ASR Teletype Console. The source tape provided is labeled Binary to Binary-Coded-Decimal Conversion (Four Digit),

Digital-8-15-U-ASCII

18.3 CALLING SEQUENCE

This subroutine is called by the JMS instruction with the binary number to be converted in the accumulator (AC).

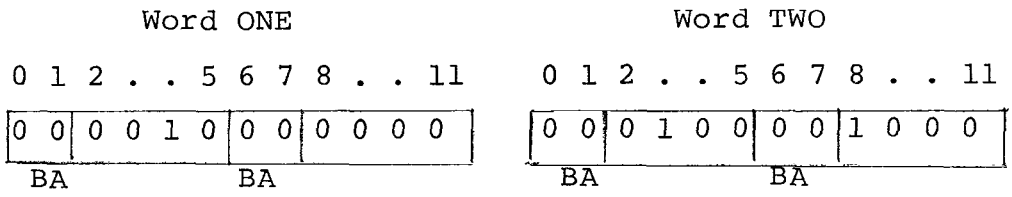
This subroutine returns to the location immediately following that containing the calling JMS. The format of the result is discussed below.

18.4 DESCRIPTION

This program is essentially Digital-8-14-U-ASCII (described in Chapter 17) extended to allow for integers in the range of 1000 to 4095.

18.5 CORE DATA

Results appear in core as:



The decimal coding for 2048 is illustrated.

18.6 IBM COMPATIBILITY

Note that bits 0, 1 and 6, 7 are set so that they can be regarded as zone B and zone A bits required for IBM BCD mode compatible 6-bit numerical characters.

In this mode of recording, the character 1010 is used for zero instead of code 0000 which this subroutine produces. Therefore, to use this routine in conjunction with IBM-compatible mag tape recording, it is necessary to write a short auxiliary routine to make this substitution.

It may also be necessary to generate the even parity required by such recording if this is not accomplished in the tape control hardware.

18.7 PROGRAM LISTING

```
/COPYRIGHT 1971  DIGITAL EQUIPMENT CORPORATION
/MAYNARD, MASSACHUSETTS
/BINARY TO BCD CONVERSION 3/7/65-HB-DEC
/ENTER WITH BINARY NUMBER IN ACCUMULATOR
/EXIT WITH 4 SIX-BIT BCD CHARACTERS
/PACKED TWO TO A WORD IN REGISTERS
/ONE AND TWO OR IN A BUFFER.
/USED FOR WRITING MAG-TAPE IN BCD FORMAT
/IN ADDITION TO BCD PARITY
/OUTPUT FORMAT:
/          ONE  0-1   A,B BITS
/          ONE  2-5   1000 DECADE
/          ONE  6-7   A,B BITS
/          ONE 8-11    100 DECADE
/          TWO  0-1   A,B BITS
/          TWO  2-5     10 DECADE
/          TWO  6-7   A,B BITS
/          TWO 8-11       1 DECADE
/STORAGE   -53 (10) REGISTERS
/TIME    324.8-350.4 MICRO-SECONDS PDP-8
```

```
0200  0000  BCD,     0
0201  3223            DCA INPUT           /STORE BINARY
0202  1225            TAD CONTRL          /SET UP TABLE
0203  3240            DCA POINTR          /POINTER
0204  1226            TAD COUNT1          /SET COUNT
0205  4234            JMS STEP            /CONVERT
0206  7106            CLL RTL
0207  4234            JMS STEP            /CONVERT NEXT
0210  1232            TAD BITS1           /A,B BIT PATTERNS
0211  3230            DCA ONE             /OR DCA I AUTO (10-17)
0212  1227            TAD COUNT2          /SET-UP COUNT
0213  4234            JMS STEP
0214  7106            CLL RTL
0215  7006            RTL
0216  7006            RTL
0217  1223            TAD INPUT           /LEAST SIGNIFICANT BITS
0220  1233            TAD BITS2           /A,B BIT PATTERMS
0221  3231            DCA TWO             /OR DCA I AUTO (10-17)
0222  5600            JMP I BCD           /EXIT
0223  0000  INPUT,    0
0224  0000  NUMBER,   0
0225  1252  CONTRL,   TAD TABLE
0226  1010  COUNT1,   1010
0227  0400  COUNT2,   0400
0230  0000  ONE,      0
0231  0000  TWO,      0
0232  0000  BITS1,    0                   /OR ANY BIT PATTERN
0233  0000  BITS2,    0                   /OR ANY BIT PATTERN
0234  0000  STEP,     0                   /ACTUAL CONVERSION
                                          /SUBROUTINE
0235  7100            CLL
0236  3224            DCA NUMBER
0237  1223            TAD INPUT
0240  1252  POINTR,   TAD TABLE           /OR TABLE+1, TABLE+2, ETC.

0241  7430            SZL                 / IF C(L)=1; INPUT>-TABLE
0242  3223            DCA INPUT           /IF SO: INPUT=INPUT+TABLE
0243  7200            CLA
0244  1224            TAD NUMBER          /ROTATES WILL BRING
0245  7004            RAL                 /COUNT BIT INTO LINK
0246  2240            ISZ POINTR
0247  7430            SZL
0250  5634            JMP I STEP          /STEP DONE
0251  5236            JMP POINTR-2
```

```
0252  0140  TABLE,    -7640                    /-4000 (10)
0253  4060            -3720
0254  6030            -1750
0255  6340            -1440
0256  7160            -0620
0257  7470            -0310
0260  7634            -0144
0261  7660            -0120
0262  7730            -0050
0263  7754            -0024
0264  7766            -0012
                /EXAMPLE: INPUT      7777  (8)
                /         OUTPUT:  ONE    00 0100/ 00 0000   =4095 (10)
                /                  TWO    00 1001/ 00 0101
```

HOW TO OBTAIN SOFTWARE INFORMATION

SOFTWARE NEWSLETTERS, MAILING LIST

The Software Communications Group, located at corporate headquarters in Maynard, publishes software newsletters for the various DIGITAL products. Newsletters are published monthly, and keep the user informed about customer software problems and solutions, new software products, documentation corrections, as well as programming notes and techniques.

There are two similar levels of service:

- The Software Dispatch
- The Digital Software News

The Software Dispatch is part of the Software Maintenance Service. This service applies to the following software products:

PDP-9/15
RSX-11D
DOS/BATCH
RSTS-E
DECsystem-10

A Digital Software News for the PDP-11 and a Digital Software News for the PDP-8/12 are available to any customer who has purchased PDP-11 or PDP-8/12 software.

A collection of existing problems and solutions for a given software system is published periodically. A customer receives this publication with his initial software kit with the delivery of his system. This collection would be either a Software Dispatch Review or Software Performance Summary depending on the system ordered.

A mailing list of users who receive software newsletters is also maintained by Software Communications. Users must sign-up for the newsletter they desire. This can be done by either completing the form supplied with the Review or Summary or by writing to:

Software Communications
P.O. Box F
Maynard, Massachusetts 01754

SOFTWARE PROBLEMS

Questions or problems relating to DIGITAL's software should be reported as follows:

North and South American Submitters:

Upon completion of Software Performance Report (SPR) form remove last copy and send remainder to:

Software Communications
P.O. Box F
Maynard, Massachusetts 01754

The acknowledgement copy will be returned along with a blank SPR form upon receipt. The acknowledgement will contain a DIGITAL assigned SPR number. The SPR number or the preprinted number should be referenced in any future correspondence. Additional SPR forms may be obtained from the above address.

All International Submitters:

Upon completion of the SPR form, reserve the last copy and send the remainder to the SPR Center in the nearest DIGITAL office. SPR forms are also available from our SPR Centers.

PROGRAMS AND MANUALS

Software and manuals should be ordered by title and order number. In the United States, send orders to the nearest distribution center.

Digital Equipment Corporation
Software Distribution Center
146 Main Street
Maynard, Massachusetts 01754

Digital Equipment Corporation
Software Distribution Center
1400 Terra Bella
Mountain View, California 94043

Outside of the United States, orders should be directed to the nearest Digital Field Sales Office or representative.

USERS SOCIETY

DECUS, Digital Equipment Computers Users Society, maintains a user exchange center for user-written programs and technical application information. The Library contains approximately 1,900 programs for all DIGITAL computer lines. Executive routines, editors, debuggers, special functions, games, maintenance and various other classes of programs are available.

DECUS Program Library Catalogs are routinely updated and contain lists and abstracts of all programs according to computer line:

- PDP-8, FOCAL-8, BASIC-8, PDP-12
- PDP-7/9, 9, 15
- PDP-11, RSTS-11
- PDP-6/10, 10

Forms and information on acquiring and submitting programs to the DECUS Library may be obtained from the DECUS office.

In addition to the catalogs, DECUS also publishes the following:

DECUSCOPE	-The Society's technical newsletter, published bi-monthly, aimed at facilitating the interchange of technical information among users of DIGITAL computers and at disseminating news items concerning the Society. Circulation reached 19,000 in May, 1974.
PROCEEDINGS OF THE DIGITAL EQUIPMENT USERS SOCIETY	-Contains technical papers presented at DECUS Symposia held twice a year in the United States, once a year in Europe, Australia, and Canada.
MINUTES OF THE DECsystem-10 SESSIONS	-A report of the DECsystem-10 sessions held at the two United States DECUS Symposia.
COPY-N-Mail	-A monthly mailed communique among DECsystem-10 users.
LUG/SIG	-Mailing of Local User Group (LUG) and Special Interest Group (SIG) communique, aimed at providing closer communication among users of a specific product or application.

Further information on the DECUS Library, publications, and other DECUS activities is available from the DECUS offices listed below:

DECUS
Digital Equipment Corporation
146 Main Street
Maynard, Massachusetts 01754

DECUS EUROPE
Digital Equipment Corp. International
(Europe)
P.O. Box 340
1211 Geneva 26
Switzerland

PDP-8 Family
Commonly Used Utility Routines
DEC-8I-RZPA-D

READER'S COMMENTS

NOTE: This form is for document comments only. Problems with software should be reported on a Software Problem Report (SPR) form (see the HOW TO OBTAIN SOFTWARE INFORMATION page).

Did you find errors in this manual? If so, specify by page.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Is there sufficient documentation on associated system programs required for use of the software described in this manual? If not, what material is missing and where should it be placed?

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Non-programmer interested in computer concepts and capabilities

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

If you do not require a written reply, please check here. ☐

Please cut along this line.

---------- Fold Here ----------

---------- Do Not Tear - Fold Here and Staple ----------

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Software Communications
P. O. Box F
Maynard, Massachusetts 01754