

DECUS

PROGRAM LIBRARY

DECUS NO.	8-602A
TITLE	THE PDP-8 COOKBOOK, VOLUME 1
AUTHOR	Floor Anthoni
COMPANY	Medical Biological Laboratory TNO Rijswijk, The Netherlands
DATE	January 1973
SOURCE LANGUAGE	PAL

ATTENTION

This is a USER program. Other than requiring that it conform to submittal and review standards, no quality control has been imposed upon this program by DECUS.

The DECUS Program Library is a clearing house only; it does not generate or test programs. No warranty, express or implied, is made by the contributor, Digital Equipment Computer Users Society or Digital Equipment Corporation as to the accuracy or functioning of the program or related material, and no responsibility is assumed by these parties in connection therewith.

THE PDP8 COOKBOOK

BY

FLOOR ANTHONI

Medical Biological Laboratory TNO, Rijswijk 2100, The Netherlands

SUBJECT: Subroutine standardisation

INTRODUCTION:

By the beginning of 1972, nearly 14 000 computers of the PDP8 family have been produced and field-installed. All of them have to be programmed to fulfill the tasks, dedicated to them.

The small size of most PDP8 configurations has forced most programmers to program the machine in assembly language. Many programs have since then found their way to the DECUS PROGRAM LIBRARY. The typical application-oriented programs, however, were rarely submitted to the LIBRARY, because nobody would ever be likely to apply for them. The experience, accumulated elsewhere, was therefore not available to others.

In programming the PDP8 computer. I have experienced the usefulness of program modularity at the assembly level. The basic modules are, in effect, subroutines that perform a certain function, and that have been programmed in such a way, that they can be used as "recipies" in a cookbook. When these "recipies" are being sent to a central editor, and published regularly, they will accumulate experience into a common module library, THE PDP8 COOKBOOK, available to others.

This paper proposes a norm for modules, submitted to the library.

THE SUBROUTINE AND ITS USE

The subroutine jump certainly is the most powerful instruction of any computer. It enables the programmer to avoid duplication of code, and to build hierarchical structures of software intelligence, increasing the semantic power of each free location in core.

Subroutines in hierarchical structures will in general do the task expected from them, with a minimum of "directions" given from "above". They can, themselves, set lower level subroutines to work for them, also with a minimum of directions. These directions are in general, information, that has to be transferred down to the subroutine. The subroutine can, on the other hand, send information back. Subroutines that can be directed to do many tasks, will, in general need more "instructions" from above. The programmer has to consider this aspect with great care. The following remarks on the ways, information can be sent to and from subroutines may assist him in this respect.

When only one parameter needs to be transferred, use the ACCUMULATOR. The LINK can be used as additional YES or NO information, although it is, in general not frequently used. The use of other registers, like the MULTIPLIER-QUOTIENT register, must be strongly dissuaded, because the module will then not be able to run on many machine configurations.

More information can be transferred as arguments, following the JMS instruction. This is especially useful for parameters that can be set at assembly time, or that need not to change very often. Use the AC for frequently changing information. A common information area in page 0 can also be used. This is especially useful when those parameters need to be accessed by many modules.

(For example program- and buffer-limits, pointers, etc.). The main problem of the sharing of the same storage locations, by

different subroutines, is that extreme care must be exerted when calling subroutines within those subroutines.

All subroutine modules in the COOKBOOK will be provided with the storage locations they need, in order to avoid conflicting use of these locations.

Another way to circumvent such problems is to employ the techniques of reentrant and recursive programming, in which push-down list structures are being used. This aspect will not be within the scope of this paper. The concept of creating an information "vector", that is a limited area in core with all the information, in order that only the pointer to this "vector" needs to be transferred, is, however, very useful for transfers, both in and out of the subroutine.

HOW TO PREVENT UNWANTED INTERFERENCE

When using subroutines, that have been used before, the most likely assembly error is that illegal redefinitions will result from the duplicate use of symbols. Therefore care must be taken to label a location. The following conventions are proposed: use very few tags. Put all storage locations and other items in front of the subroutine entry, that needs to have more than 3 characters. All other tags need to share, at least the first 3 characters of the subroutine entry.

Those programmers that want to "pack" subroutines into the least possible space, will find it easy to modify the subroutines in this respect.

DOCUMENTATION

Simple subroutines need less documentation than the more sophisticated ones. Comments should be inserted, wherever additional

information is needed. Avoid trivial comments like CLA/CLEAR AC, but express the general concept and thoughts, as if it were a flow chart. The documentation must be adequate for the reader to easily understand how the subroutine works. For more sophisticated routines a flow chart is a must. Each subroutine must have a compact functional description of not more than one line (52 characters). Then follows a general description of the subroutine and an example of its use. All program lines and comment lines should not exceed 52 positions, as assembler output and cross-reference numbers must have room to be inserted.

The source tape should be submitted with the tabulations, not being converted to spaces.

The listing should preferably be made with a teletype printer (teletype type of character), printed with tabs converted to spaces. Use a clean typing head and a new black ribbon, as the listing will be offset-copied. Drawings and flow-charts should be drawn with black ink, or taped with special stickers.

For the use of symbols, the reader is referred to Appendix I.

PROGRAM SUBMISSION

Submit your program subroutine to

The Editorial Board of
The PDP8 COOK BOOK
c/o Floor Anthoni,
Medical Biological Laboratory TNO,
139, Lange Kleiweg,
RIJSWIJK (ZH), 2100,
The Netherlands.

NOTE! It is of vital importance that errors are reported back to the authors or the editorial board. Only by doing so one can achieve the highest reliability of the published subroutines.

COOKBOOK VOLUME 1 CATALOG LISTED BY NUMBER

- 001 Type the characters following the JMS instruction X
- 002 Teletype type routine with overlap X
- 003 Type a character chain X
- 004 Binary to decimal conversion, single prec. no sign
- 005 Binary to octal conversion, no sign. fixed field
- 006 High speed reader subroutine X
- 007 Tabulator routine
- 008 Move a block through core X
- 009 Binary punch with field setting, checksum, leader
- 010 PAL message printer X
- 011 General branch routine
- 012 Check AC if octal
- 013 Logical operators, AND, OR, NAND, NOR, EXCL. OR, etc. X
- 014 PS8/OS8 option decoder
- 015 Print 2 digits in decimal
- 016 Print the PS8/OS8 date
- 017 Print the AC as a FOCAL linenummer
- 018 Print 4 decimal digits, using routine 015, no sign
- 019 Read a decimal number in core
- 020 Decimal print, leading blanks, no sign
- 021 Print double length decimal, no sign
- 022 Octal print, no sign, leading spaces
- 023 Double word octal print using 022
- 024 Translate TELEX code to ASCII
- 025 Translate TELEX code to ASCII
- 026 Translate ASCII code to TELEX
- 027 Interrupt ASCII output handler with rotating buffer X
- 028 Device interrupt handler (part of 027) X

- 029 Read or write DECtape in both directions
- 030 Subroutine to pack a fixed buffer in core (300 chars) into a fixed output buffer (200 chars) in TSS8 packed format
- 031 Pack characters into a buffer in TSS8 format, one by one
- 032 As 031, but with a fixed allocated buffer
- 033 Unpack TSS8 format packed buffer into an output buffer
- 034 Unpack TSS8 format packed buffer, one character at a time
- 035 Subroutine to read a 6 character name in core
- 036 Search a file name in DN blocks (Disk monitor)
- 037 Search for an unused block in SAM block, and reserve it for the current file
- 038 Search internal file number in SAM blocks (Disk Monitor)
- 039 Subroutine to read or write on disk (TSS8).

LIST OF CONTRIBUTORS

Contributions

Floor Anthoni
Medisch Biologisch Laboratorium TNO,
Lange Kleiweg 139,
Rijswijk (ZH),
The Netherlands

1, 2, 3, 4, 5, 6, 7, 9,
9, 10, 11, 12, 13, 14,
15, 17, 18

Thierri den Dunnen
Dr.Neher Laboratorium,
St.Paulusstraat 4,
Leidschendam,
The Netherlands

19, 20, 21, 22, 23, 24, 25,
26, 27, 28, 29, 30, 31, 32,
33, 34, 35, 36, 37, 38, 39

Hans Mees,
Prins Maurits Laboratoria, C.D.,
Lange Kleiweg 137,
Rijswijk (ZH),
The Netherlands

8

Paul Lohman,
Medisch Biologisch Laboratorium TNO,
Lange Kleiweg 139,
Rijswijk (ZH),
The Netherlands

16

FLOW-CHART conventions

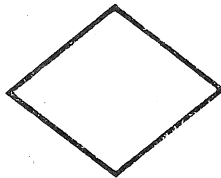
The flow-charts make use of relatively few symbols :



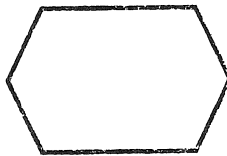
Entry, or exit of a program or sub-program, also used for the inter-connection of flow-charts on different pages.



A rectangle describes one or more program steps.



Decision, branching



Subroutine call.

A subroutine may have more than one return (branching).

/COMMENT

Comments appear behind a slash (/).

START,
LOOP,

Used to name program ties in agreement with the listings.

/001 TYPE THE CHARACTERS FOLLOWING THE JMS INSTR.
/TERMINATOR IS A ZERO.

```
/
/      JMS TYPTEX      /TYPE "ABC"
/      301             /"A"
/      302             /"B"
/      303             /"C"
/      0               /TERMINATOR
/      RETURN          /AC=0
```

```
TYPTEX, 0
      TAD I TYPTEX      /GET CHAR.
      ISZ TYPTEX
      SNA               /ZERO?
      JMP I TYPTEX*     /YES, JMP TO NEXT LOC.
      JMS TYPE          /NO, TYPE
      JMP TYPTEX+1
```

Prog 002

/002 TELETYPE TYPE ROUTINE *used by prog 001, 003*
/INITIALIZES WHEN ENTERED FOR FIRST TIME.
/NOT RESTARTABLE !

/

/ TAD CHARACTER

/ JMS TYPE

/ RETURN /AC=0

TYPE, NOP

0

JMP .+3 /OVERLAI D BY "NOP"

TSF

JMP .-1

TLS

CLA

TAD TYPE-1

DCA TYPE+1

JMP I TYPE

/003 TYPE A CHARACTER CHAIN
 /TYPE THE CHARACTERS IN THE LIST, POINTED TO
 /BY THE FIRST ARGUMENT. LIST TERMINATOR =0

/
 / JMS TYPTX /TYPE "ABC"
 / LIST
 / RETURN /AC=0
 /

/LIST, 301
 / 302
 / 303
 / 0

0 /USED AS POINTER
 TYPTX, 0 /TYPE TEXTSTRING
 TAD I TYPTX /GET ARG
 DCA TYPTX-1 /SAVE TO USE AS POINTER
 ISZ TYPTX /FOR CORRECT RETURN
 TAD I TYPTX-1 /GET CHAR
 SNA /ZERO?
 JMP I TYPTX /YES, RETURN
 JMS TYPE /NO
 ISZ TYPTX-1
 JMP TYPTX+4 /LOOK FOR NEXT

prog 002

/004 BINARY TO DECIMAL CONVERSION AND TYPE; NO SIGN
 /ROUTINE TO CONVERT A BINARY WORD TO DECIMAL AND TYPE IT.
 /VALID FOR NUMBERS 0-4095. NO SIGN.
 /IF USED FOR 3 DIGITS: DELETE 6030; -4=-3 DIGIT COUNT.

/
 / TAD WORD
 / JMS PRINTD
 / RETURN /AC=0

CCC 6030 /-1000 CONVERSION CONSTANTS
 7634 /-100
 7766 /-10
 7777 /-1

-6 1204 TAD . /USED FOR CONV. CONSTANTS
 -5 DIGIT 0 /DIGIT BCD TO BE TYPED
 -4 CNTR, 0 /COUNTER
 -3 K260 260 /TO MAKE A CHAR.
 -2 REMAIN, 0 /SAVE AREA
 -1 MA, -4 /DIGITS TO BE TYPED (-4,-3,-2) 7774
 PRINTD, 0 /ENTER WITH WORD IN AC

DCA PRINTD-2
 TAD PRINTD-1 /SET UP COUNT
 LCA PRINTD-4

LOOP1, DCA PRINTD-5 ← /CLEAR BCD
 LOOP2, TAD PRINTD-6 ← /FETCH CURR. CONV. CONST.
 TAD PRINTD-4 /BY ADDING COUNT TO TAL
 DCA .+1
 HLT ← why not! /SA to TAD CCC + (0,1,2,3)

1204
 7774
 1200

CLL
 TAD PRINTD-2 /VALUE - CONSTANT
 SNL /OVERFLOW? L=1 for each addition until SUM goes NEGATIVE
 JMP .+4 ← L=1 for each addition until SUM goes NEGATIVE
 ISZ PRINTD-5 /NO, TYPE IT
 DCA PRINTD-2 /YES, NEXT TRY
 JMP PRINTD+5 /SAVE REMAINDER ~ REMAIN
 CLA
 TAD PRINTD-5 /BCD
 TAD PRINTD-3 /+260
 JMS TYPE
 ISZ PRINTD-4
 JMP PRINTD+4 /NEXT DIGIT
 JMP I PRINTD

Prog 002

CLA before leaving

Codit 23

Prints

Consts 7

Vars 3

33

5 instructions shorter than DEC routine

— /005 BINARY TO OCTAL CONVERSION AND PRINT
/ROUTINE PRINTS THE AC IN OCTAL, NO SIGN.

```

/
/      TAD WORD
/      JMS PRINT8
/      RETURN      /AC=0
/
      260
      7      /MASK
      0      /DIGIT COUNTER
      -4     /# OF DIGITS
      0      /TEMPORARY
PRINT8, 0
      RAL CLL
      LCA PRINT8-1
      TAD PRINT8-2
      LCA PRINT8-3      /SET UP COUNT
LOOP  TAD PRINT8-1
      RAL
      RTL
      LCA PRINT8-1
      TAD PRINT8-1
      AND PRINT8-4      /MASK
      TAD PRINT8-5      /MAKE ASCII
      JMS TYPE
      ISZ PRINT8-3      /4 DONE?
      JMP PRINT8+5      /NOT YET  i.e. to loop
      JMP I PRINT8

```

— Prog 002

Code 16
Vars Counts 5
21

```

/006 HIGH SPEED READER SUBROUTINE
/ENTER WITH AC=0; ROUTINE INITIALIZES HSR. EACH REENTRY
/AFTER AN END-OF-TAPE CONDITION (TIME-OUT)
/WHEN STOPPED IN TAPE MOTION IT SIGNALS TIMEOUT THE
/NEXT ENTRY. THE ROUTINE HAS A BUILT-IN TIMING LOOP
/THAT TIMES OUT IF THE READER IS NOT SWITCHED ON,
/OR IF THE READER LOOSES ITS FLAG BY RUNNING OUT OF TAPE.
/
/      JMS HSREAD
/      OUT OF TAPE RETURN
/      NORMAL RETURN, CHAR. IN AC
/

```

```

      0                      /USED AS TIME-OUT COUNT
HSREAL, 0                    /ENTRY
      LCA HSREAL-1          /SET UP COUNT
HSRFLG, 1                    /FLAG SIGNALS TO INIT READER
      TAD HSRFLG            /THESE INSTR. CONTRIBUTE TO LOOP
      SZA CLA
      JMP .+3               /INIT READER
      RSF                   /SKIP?
      JMP .+5               /NO, COUNT TIME-OUT
      DCA HSRFLG            /CLEAR FLAG
      6016                  /READ
      ISZ HSREAL            /RETURN, CHAR IN AC
      JMP I HSREAL
      ISZ HSREAL-1
      JMP HSRFLG
      ISZ HSRFLG            /SET FLAG TO SIGNAL TIMEOUT
      JMP I HSREAL          /EOT RETURN

```

```

/007 TABULATOR ROUTINE
/THE USER HAS TO TAKE CARE OF:
/INCREMENTING TABCNT WITH EACH INCOMING CHARACTER, CLEARING
/IT WHEN CARRIAGE RETURN. TAB-INTERVAL IS VARIABLE.
/A JMS TO TAB WILL MOVE THE TYPING HEAD TO THE NEXT
/TABULATOR POSITION.
/
/      CLA
/      JMS TAB
/      RETURN          /AC=0

TABCNT, 0
      -10              /TAB INTERVAL
      240              /SPACE
TAB,    0              /ENTER WITH AC =0
      TAL TABCNT       /SUBTRACT N TIMES TO GIVE REMAINDER
      TAD TAB-2
      SMA
      JMP .-2
      LCA TABCNT       /USE AS NEGATIVE COUNTER
      TAL TAB-1
      JMS TYPE
      ISZ TABCNT       /READY?
      JMP .-3
      JMP I TAB        /YES

```

Prag 002

TAD MOVE-1	/INCREMENT OUTPUT POINTER
DCA MOVE-3	
ISZ MOVE-4	/INCREMENT WORD COUNT
JMP MVLOOP	/AGAIN IF NOT ZERO
/	
MOVRET, ISZ MOVE	/SET UP RETURN ADDRESS
JMP I MOVE	/RETURN

```

/009 BINARY PUNCH WITH FIELD SETTING
/THREE SUBROUTINES TO PUNCH AN AREA OF CORE IN BINARY
/LOADER FORMAT. FIELD SETTINGS AND ORIGIN SETTINGS
/ARE BEING PUNCHED AT EACH ENTRY; CHECKSUM IS PUNCHED
/WHEN PUNCHK IS CALLED.
/THE ROUTINE CAN TAKE DATA FROM A DIFFERENT FIELD.
/IT CAN OPERATE IN ALL FIELDS. SEVERAL USES APPLY:
/1) NORMAL USE. THE DATA IS LOCATED IN THE SAME FIELD OR
/STRANGE FIELD. ENTER WITH FIELD IN AC; LINK=0
/2) THE CODE IS IN SAME FIELD AS BINPUN, ONLY A DIFFERENT
/FIELD SETTING NEEDS TO BE PUNCHED. ENTER BINPUN
/WITH FIELD IN AC AND LINK=1.
/3) THE CODE HAS BEEN MOVED IN CORE. THE FIRST LOCATION
/IS NOT NECESSARILY THE ORIGIN. NOW ENTER BINPUN WITH
/AC=FIELD FOR SETTING; LINK=1; SET ORIGIN UNEQUAL TO
/FIRST LOCATION IF THIS IS TRUE.
/EXAMPLE OF NORMAL USE:

```

```

/
/      JMS LEADER      /PUNCH LEADER, CLEAR CHECKSUM
/      CLL
/      TAD (0010      /FIELD 1
/      JMS BINPUN
/      ORIGIN          /IN NORMAL USE=FIRST LOC.
/      FIRST LOC.
/      LAST LOC.
/      JMS PUNCHK      /PUNCH CHKSM AND TRAILER

BINEND, 0              /LAST LOC. TO PUNCH
        6201
        100
        300           /FOR FIELD SETTING
        0              /TEMP. STORAGE

BINPUN, 0
        DCA BINPUN-1
        SZL            /SET UP DF IF LINK=0
        JMP BIN3
        TAD BINPUN-1
        TAD BINPUN-4   /MAKE CDF
BIN3,    DCA BIN2
        TAD BINPUN-1   /MAKE FIELD SETT. AND PUNCH
        TAD BINPUN-2   /NOTE! FIELD SETT. NOT IN CHECKSUM!!
        JMS TYPE
        TAD I BINPUN   /GET ORIGIN
        ISZ BINPUN
        DCA BINPUN-1
        TAD BINPUN-1   /PUNCH ORIGIN
        JMS BINLH      /LEFT HALF
        TAD BINPUN-3   /+100 FOR ORIGIN
        JMS BINCHK
        TAD BINPUN-1   /RIGHT HALF AND PUNCH
        AND BINLH-1
        JMS BINCHK
        TAD I BINPUN   /SET UP POINTER
        DCA BINPUN-1
        ISZ BINPUN
        TAD I BINPUN   /GET END
        DCA BINEND

BIN2,    CDF 0          /OVERLAID BY CDF STRANGE FIELD
        TAD I BINPUN-1 /GET DATA
        JMS BINLH      /PUNCH LEFT HALF

```

JMS BINCHK
 TAD I BINPUN-1 /PUNCH RIGHT HALF
 AND BINLH-1
 JMS BINCHK
 TAD BINEND /END REACHED?
 CIA
 CLL
 TAD BINPUN-1
 ISZ BINPUN-1
 SNL CLA
 JMP BIN2+1 /NO, TAKE NEXT DATA
 RIF /YES, RESTORE LF
 TAD BINPUN-4
 DCA .+1
 CDF 0 /OVERLAID
 ISZ BINPUN
 JMP I BINPUN
 SPA CLA

/GET LEFT HALF OF THE AC

77
 BINLH, 0
 RTR
 RTR
 RTR
 AND BINLH-1
 JMP I BINLH

/UPDATE THE CHECKSUM, AND PUNCH FRAME

0 /CHECKSUM
 PINCHK, 0 /ENTER WITH 6 BIT FRAME IN AC
 DCA BINLH /USE BINLH TEMPORARILY
 TAD BINLH
 TAD PINCHK-1 /UPD. CHKSM
 DCA BINCHK-1
 TAD BINLH
 JMS TYPE
 JMP I BINCHK

/PUNCH 100(8) LEADER OR TRAILER HOLES; CLEAR CHECKSUM

200
 LEADER, 0
 TAD BINPUN-3 /USE CHKSM AS NEG COUNT
 CIA
 DCA BINCHK-1
 TAD LEADER-1
 JMS TYPE
 ISZ BINCHK-1 /READY?
 JMP .-3 /NO
 JMP I LEADER /YES

/PUNCH CHECKSUM; PUNCH TRAILER; CLEAR CHECKSUM

PUNCHK, 0
 TAD BINCHK-1 /GET CHECKSUM
 JMS BINLH
 JMS TYPE
 TAD BINCHK-1
 AND BINLH-1
 JMS TYPE
 JMS LEADER /CLEARS CHECKSUM
 JMP I PUNCHK

Full
 & See *Revision in Vol 2 p 14*

/010 PAL MESSAGE PRINTER
 /PRINTS A MESSAGE CODED WITH THE PAL PSEUDO-OP
 /'TEXT'. PAL3 AND PAL8 COMPATIBLE

/

/ JMS PRMSG
 / MSG
 / RETURN /AC=0
 /

/MSG, TEXT 'ABC82' /CODED AS 0102;0370;6200

77 /MASK
 -40 /TO TEST
 240 /TO MAKE ASCII
 100 /TO MAKE ASCII
 0 /PACKSWITCH 0=LEFT;7777=R
 0 /POINTER

PRMSG, 0

CMA /SAVE POINTER(-1)

better to use CLA CMA

See p13-Vol2.

TAD I PRMSG

DCA PRMSG-1

ISZ PRMSG /FOR RETURN

PRM1, CMA

DCA PRMSG-2 /PACKSW=RIGHT

ISZ PRMSG-1 /NEXT WORD

TAD I PRMSG-1 /FETCH AND ROTATE 6

RTR

RTR

RTR

PRM2, AND PRMSG-6 /MASK 6 BITS

SNA

JMP I PRMSG /ZERO ENDS THE LIST

TAD PRMSG-5 /<40?

SPA

TAD PRMSG-3 /YES ASCII 301-337

TAD PRMSG-4 /NO, ASCII 240-277

JMS PRINT

ISZ PRMSG-2 /LEFT OR RIGHT?

JMP PRM1 /LEFT

TAD I PRMSG-1 /RIGHT

JMP PRM2

21

See Revision in Vol 2 p 15

/M12 CHECK IF OCTAL
/ROUTINE CHECKS WHETHER THE AC IS AN OCTAL DIGIT.
/
/ TAD CHARACTER — *ASCII coded.*
/ JMS OCTCHK
/ NOT OCTAL RETURN /AC=0
/ OCTAL RETURN /AC=0

10
-270
OCTCHK, 0
TAD OCTCHK-1
SMA
JMP OCT2
TAL OCTCHK-2
SPA CLA
JMP I OCTCHK
ISZ OCTCHK
OCT2, CLA
JMP I OCTCHK

/013 LOGICAL OPERATORS ON TWO NUMBERS
/THE RESULT OF LOGICAL OPERATIONS IS IN THE AC.

/AND (MASKING) A 1010
/ B 1100
/ = 1000
/ CLA
/ TAD A
/ AND B

/INCLUSIVE OR A 1010
/SETS BITS B IN A B 1100
/ = 1110
/ CLA
/ TAD A
/ CMA
/ AND B
/ TAD A

/CLEAR BITS B IN A A 1010
/ B 1100
/ = 0010
/ CLA
/ TAD B
/ CMA
/ AND A

/NOT A 1010
/ B 1100
/ = 0001
/ TAD A
/ CMA
/ DCA TEM
/ TAD B
/ CMA
/ AND TEM

/NAND A 1010
/ B 1100
/ = 0111
/ TAD A
/ AND B
/ CMA

/EXCLUSIVE OR A 1010
/ B 1100
/ = 0110
/ TAD A
/ AND B
/ CMA
/ DCA TEM
/ TAD A
/ AND TEM

TAD B
AND TEM

*It clears bits A in B to prevent carry's
then adds A*

*See Revision in
Vol 2 p13*

*See Revision in
Vol 2 p13*

A B 1000
CMA 0111 → TEM
A-TEM 0000
AND B 1100
A-TEM 0100

```

/014 PS8-0S/8 OPTION DECODER
/CHECKS THE OPTION, SPECIFIED IN THE AC AND CAUSES
/A RETURN, DEPENDING ON WHETHER THE OPTION HAS BEEN
/SET
/OPTIONS IN OS8 RESIDE IN FIELD 1 LOC 7643-7645 :
/
/7643 A B C D E F G H I J K L ASCII 301-314
/7644 M N O P Q R S T U V W X ASCII 315-330
/7645 Y Z 0 1 2 3 4 5 6 7 8 9 ASCII 331,332,260-271
/
/      TAD (16          /CHECK OPTION 16 (N)
/      JMS OPTION
/      OPTION NOT SET RETURN/AC=0
/      OPTION SET RETURN   /AC=0
/

```

```

OPTM1, 7777
-14          /-12(10)
7642        /POINTER
0           /TEMP. STORAGE
7642        /COUNTER, ALSO POINTER
OPTION, 0    /ENTER WITH POSITION IN AC
DCA OPTION-2
TAD OPTION-3 /RESTORE COUNTER
DCA OPTION-1
TAD OPTION-2 /SUBTRACT 12 TO FIND WORD
TAD OPTION-4
ISZ OPTION-1
SMA SZA
JMP --3
TAD OPTM1    /FOR L AND X
DCA OPTION-2 /SAVE REMAINDER MODULO 12
CLL CML      /AND ROTATE ONE BIT INTO POSITION
RAL          /ROTATE FURTHER
ISZ OPTION-2
JMP --2
CDF 10       /AND WITH OPT WORD FIELD 1
AND I OPTION-1
CDF 0
SZA CLA
ISZ OPTION   /IN CASE IT HAD BEEN SET
JMP I OPTION

```

/015 PRINT TWO DIGITS IN DECIMAL
 /THE VALUE OF THE AC IS PRINTED IN TWO DIGITS
 /CORRECTLY IF < 99(DECIMAL).

(21)

```

/
/      TAD (VALUE
/      JMS PRNT2
/      RETUEN          /AC=0

260          /TO MAKE ASCII
-12          /10 DECIMAL
0           /TEMP STORAGE
0           /COUNTER
PRNT2, 0
DCA PRNT2-2
TAD PRNT2-2    /TRY SUBTRACT 10 UNTIL OVFL0
TAD PRNT2-3
SPA
JMP .+3
ISZ PRNT2-1    /SUBTRACT FURTHER
JMP PRNT2+1
CLA
TAD PRNT2-1    /PRINT HIGH ORDER DIGIT
TAD PRNT2-4
JMS PRINT
TAD PRNT2-2
TAD PRNT2-4
JMS PRINT
DCA PRNT2-1    /RESET COUNTER
JMP I PRNT2
  
```



Code 17
 Vars 2
 Consts 2

 21

/016 PRINT THE PS8-OS8 DATE
 /THE DATE IS PRINTED AS: 07/17/72
 /THE ROUTINE MAKES USE OF PRNT2, TO TYPE TWO
 /DECIMALS. REQUIRES ROUTINES PRNT2 AND PRINT.
 /DATE IN OS8 IS STORED IN LOC 7666, FIELD 1:

/
 /7666 MMMMDDDDYY /M=MONTH,D=DAY,Y=YEAR

/
 / JMS DATE
 / RETURN

/AC=0

DATM, 7 /MASKS

17

37

257

/SLASH

106

/70 YEARS

0

/STORAGE

7666

/DATE LOC. IN OS8

DATE, 0

CDF 10

/PICK TH E DATE

TAD I DATE-1

CDF 0

DCA DATE-2

TAD DATE-2

CLL RTL

/SHIFT MONTH OUT

RTL

RAL

AND DATM+1

/AND (17

JMS PRNT2

TAD DATE-4

/PRINT SLASH

JMS PRINT

TAD DATE-2

RTB

/SHIFT MONTH OUT AND PRINT

RAF

AND DATM+2

JMS PRNT2

TAD DATE-4

/SLASH

JMS PRINT

TAD DATE-2

/NOW THE YEAR

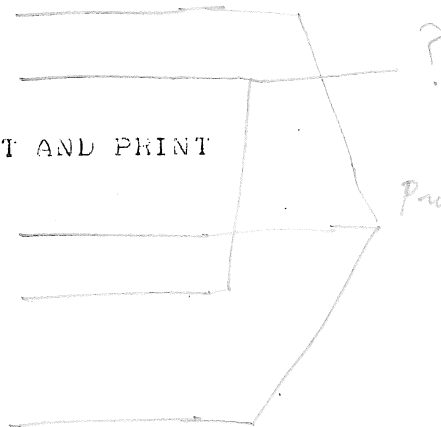
AND DATM

TAD DATE-3

/+70

JMS PRNT2

JMP I DATE



/017 PRINT THE AC AS A FOCAL LINENUMBER
 /THE VALUE OF THE AC IS PRINTED AS IN FOCAL:11.35
 /XX.YY STORED AS FOLLOWS: XXXXXYYYYYYY IN 1 WORD.
 /IF YYYYYYY>99 STRANGE DIGITS OCCUR AS IN FOCAL.
 /REQUIRES ROUTINES PRNT2 AND PRINT.

/
 / TAD VALUE
 / JMS PRNTF
 / RETURN /AC=0

PRNTFM, 37 /MASKS
 177
 256 /PERIOD.
 0 /STORAGE

PRNTF, 0
 DCA PRNTF-1
 TAD PRNTF-1 /ISOLATE AND PRINT HIGH ORDER
 CLL RTL
 RTL
 RTL
 AND PRNTFM /AND (37
 JMS PRNT2
 TAD PRNTF-2
 JMS PRINT
 TAD PRNTF-1 /NOW LOW ORDER
 AND PRNTFM+1
 JMS PRNT2
 JMP I PRNTF



/018 PRINT 4 DECIMAL DIGITS USING ROUTINE PRNT2
 /THE CONTENT OF THE AC IS DIVIDED BY 100(10)
 /GIVING TWO LOW ORDER DIGITS AND 2 HIGH ORDER.
 /THESE ARE PRINTED BY PRNT2.

/
 / TAD VALUE
 / JMS PRNT4
 / RETURN /AC=0
 7634 /-100(10)
 0 /STORAGE AND LOW ORDER
 0 /HIGH ORDER COUNTER
 PRNT4, 0
 LCA PRNT4-2
 CLL
 TAD PRNT4-2 /TRY TO SUBTRACT 100 UNTIL OVERFLOW
 TAD PRNT4-3
 SNL
 JMP .+3
 ISZ PRNT4-1
 JMP PRNT4+1
 CLA
 TAD PRNT4-1 /PRINT HIGH ORDER DIGITS
 JMS PRNT2
 TAD PRNT4-2 /PRINT LOW ORDER DIGITS
 JMS PRNT2
 LCA PRNT4-1 /RESET COUNTER
 JMP I PRNT4

} Prog 015

code 16
 octal 3
 19

/019 SUBROUTINE READS A DECIMAL NUMBER FROM KEYBD
/RUBOUT REMOVES NUMBER COMPLETELY

/CALL :JMS DECINP
/ RETURN WITH NUMBER BINARY IN AC

DECINP,0

CLA	
DCA DECNUM	/CLEAR REGISTER
JMS READ	/READ CHAR FROM KEYBOARD
TAD CHAR	
JMS PRINT	/PRINT THAT CHAR
TAD CHAR	/GET CHARACTER
TAD M377	/IS IT RUBOUT?
SNA CLA	
JMP DECINP+1	/YES READ ALL OVER AGAIN
TAD CHAR	/NO
TAD M260	
SPA	/CHAR>=260?
JMP DECOU	/NO, CHARACTER IS DELIMETER
TAD M12	/YES
SMA CLA	/CHAR<272?
JMP DECOU	/NO, CHAR IS DELIMETER
TAD DECNUM	/YES, CHAR IS FIGURE
CLL RAL	
DCA DECTMP	/NUMB.*2
TAD DECTMP	
RTL	/NUMB*8
TAD DECTMP	/NUMB*8+NUMB*2=NUMB*10
TAD CHAR	/ADD LAST FIGURE
TAD M260	
DCA DECNUM	/DECIMAL NUMBER
JMP DECINP+3	

DECOU, CLA

TAD DECNUM
JMP I DECINP /EXIT

/VARIABLES

DECNUM,0
DECTMP,0

/GENERAL CONSTANTS

M12, -12
M260, -260
M377, -377

```

/020 DECIMAL PRINT ROUTINE,
/PRINTS AC DECIMAL IN 4 DIGITS
/MAX NUMBER = 4095 DECIMAL
/SKIPS LEADING ZERO'S
/

```

```

DPRT, 0
    DCA DPRREG      /SAVE AC IN PRINTREG.
    TAD LPRINS      /GET INSTRUCTION
    DCA DPRPTP      /PUT INSTR. ON POINTER
    TAD M4
    DCA DPRFAC      /4 DIGITS
    DCA DPRFL       /CLEAR PRINT 0 FLAG
    DCA DPRFIG      /CLEAR DIGIT

DPRSUB, CLL
    TAD DPRREG      /PICK UP SAVED AC
    DPRPTP, TAD DPRTEN /SUBTRACT POWER OF TEN
    SNL             /REMAINDER POSITIVE?
    JMP .+4         /NO, PRINT DIGIT
    DCA DPRREG      /YES, SAVE REMAINDER
    ISZ DPRFIG      /DIGIT:=DIGIT+1
    JMP DPRSUB      /REPEAT SUBTRACTION
    CLA CLL
    TAD DPRFIG      /GET DIGIT
    SNA             /A ZERO?
    JMP DPRZRO      /YES

DPRIN, TAD C260     /NO, CONVERT TO ASCII
    JMS PRINT
    ISZ DPRFL       /MAKE NOT EQUAL 0
    DPRIN1, ISZ DPRPTP /MODIFY INSTR ON DPRPTP
    ISZ DPRFAC      /PRINTED 4 DIGITS?
    JMP DPRSUB-1    /NO, PRINT NEXT DIGIT
    JMP I DPRT      /YES, RETURN
/

```

```

LPRZRO, TAD DPRFL
    SZA CLA
    JMP DPRIN
    JMP DPRIN1
/

```

```

DPRREG, 0
DPRFL, 0
LPRINS, TAD DPRTEN
DPRFAC, 0
DPRFIG, 0
DPRTEN, 6030      /-1000
          7634      /-100
          7766      /-10
          7777      /-1
/

```

```

/GENERAL CONSTANTS
M4, -4
C260, 260

```

(42) Worse than DEC's.

/021 SUBROUTINE TO PRINT DOUBLE LENGTH DECIMAL

/
 /CALL: JMS DDECPR
 / MOST SIGNIFICANT PART
 / LEAST SIGNIFICANT PART
 / NUMBER OF DIGITS TO BE PRINTED (<=8)
 / RETURN
 /
 /

DDECPR,0

TAD I DDECPR /FETCH MOST SIGNIFICANT PART
 DCA DLX /SAVE
 DCA DLPD /CLR NUMB. OF PRINTED DIGITS
 ISZ DDECPR
 TAD I DDECPR /FETCH LEAST SIGNIFICANT PART
 DCA DLX+1 /SAVE
 ISZ DDECPR
 TAD I DDECPR /FETCH FORMAT
 DCA DDNDIG
 ISZ DDECPR /CORRECT RETURN
 TAD DDATFL /ADDRESS 10-POWER LOW
 DCA DDPTPL /POINTER 10-POWER LOW
 TAD DDATPH /ADDRESS 10-POWER HIGH
 DCA DDPTPH /POINTER 10-POWER HIGH
 TAD #10
 DCA DDNFAC /FACTORISE 8 DIGITS
 DCA DDIGIT /CLEAR DIGIT

DDSUB, CLL

TAD DLX+1 /L SIGNIFIC PART OF NUMB.
 TAD I DDPTPL /LOW PART FACTOR
 DCA DLX+1 /STORE
 HAL /OVERFLOW IN AC
 TAD DLX /M SIGNIFIC PART OF NUMB.
 TAD I DDPTPH /HIGH FACTOR
 SNL /RESULT NEGATIVE?
 JMP .+4 /YES
 DCA DLX /STORE RESULT OF SUBTRACTION
 ISZ DDIGIT /NO, STEP UP DIGIT
 JMP DDSub /SUBTRACT 2-LENGTH AGAIN
 CLA /CLEAR BEFORE CORRECTION
 TAD I DDPTPL /10-POWER LOW
 CIA /MINUS
 TAD DLX+1 /CORRECT LAST SUBTRACTION
 DCA DLX+1 /STORE
 TAD DDIGIT /GET DIGIT
 SZA /=0?
 JMP DDPDIN /NO
 TAD DLPD
 SZA CLA /ALREADY PRINTED?
 JMP DDPDIN /YES
 IAC
 TAD DDNFAC
 SXA CLA /ALL DI#D0= 0?
 JMP DDPDIN /YES
 TAD DDNFAC /NEGATIVE VALUE
 TAD DDNDIG /POSITIVE VALUE
 SPA CLA /SPACE?
 JMP DDPTIN
 TAD C240 /YES
 JMP DDFFI

```

DDPEIN, ISZ DDPE
      TAD C260
DDFPR, JMS PRINT
DDPTIN, ISZ DDPTPL
      ISZ DDPTPH
      ISZ DDNFAC
      JMP DLSUB-1
      TAD DDPE
      CIA
      TAD DDNDIG
      SPA SNA CLA
      JMP .+3
      TAD DDNDIG
      DCA DDPE
      CLL
      JMP I DDECPR
/
/
/CONSTANT'S PUNCH OUT ROUTINE
DDATPL, DDTPPL
LDATPH, DDTPH
DDPTPL, 0
DDPTPH, 0
DLX, 0
DLX, 0
DDNFAC, 0
DDNDIG, 0
DDPL, 4600
      6700
      4540
      4360
      6030
      7634
      7766
      7777
DDTPH, 3166
      7413
      7747
      7775
      7777
      7777
      7777
      7777
DDPE, 0
DDNDIG, 0
/
/GENERAL CONSTANTS
M10, -10
C240, 240
C260, 260

```

/CONVERT TO ASCII

/PRINT DIGIT

/STEP UP POINTER LOW

/STEP UP POINTER HIGH

/READY FACTORIZE?

/NO, NEXT DIGIT

/EXIT, END PUNCHOUT ROUTINE

```

/022 OCTAL PRINT ROUTINE
/NONSIGNIFICANT ZERO'S BECOME SPACES
/
/      CLA
/      DCA OCTFIG      /CLEAR FLAG FIGURE PRINTED
/      DCA OCTSPC      /CLEAR SPACE-COUNTER
/      TAD NUMBER
/CALL   :JMS OCTPRT      / WITH NUMBER IN AC
/      RETURN AC=0      /IF NUMBER=0,
/OCTSPC=4, = # OF SPACES TO PRINT
/      IF NUMBER IS ZERO, OCTSPC=4 IS #SPACES TO PRINT
/
OCTPRT, 0
      RAL              /ROTATE IN LINK
      DCA OCTTMP      /TEMP. STORAGE
      TAD M4          /4 OCTADES
      DCA OCTCNT
OCTPR0, TAD OCTTMP
      RAL
      RTL
      DCA OCTTMP      /STORE RESULT
      TAD OCTTMP      /GET IT BACK
      AND C7          /MASK OCTADE
      SNA CLA         /ZERO ?
      JMP OCTZER      /YES
      TAD OCTSPC      /NO, SPACES TO PRINT?
      SNA
      JMP OCTNUM      /NO, GO PRINT FIGURE
      CIA             /YES, SET COUNTER
      DCA OCTSPC
      TAD C240
      JMS PRINT       /PRINT THE SPACES
      ISZ OCTSPC
      JMP .-3
OCTNUM, CLA IAC       /SET FLAG FIG. PRINTED
      DCA OCTFIG
      TAD OCTTMP
      AND C7
OCT0PR, TAD C260      /MAKE THE FIGURE
      JMS PRINT
OCTPR1, ISZ OCTCNT    /READY?
      JMP OCTPR0      /NO
      JMP I OCTPRT    /YES, EXIT
/
OCTZER, TAD OCTFIG
      SZA CLA         /FIGURES PRINTED ?
      JMP OCT0PR      /YES, PRINT THIS ZERO TOO
      ISZ OCTSPC      /NO COUNT AS SPACE
      JMP OCTPR1
/
OCTTMP, 0
OCTCNT, 0
OCTSPC, 0
OCTFIG, 0
M4, -4
C7, 7
C240, 240
C260, 260

```

/023 DOUBLE WORD OCTAL PRINT ROUTINE

/USES ROUTINE OCTPR

/CALLING: JMS DOCTPR

/HIGH ORDER NUMBER

/LOW ORDER NUMBER

/RETURN AC=0

/

DOCTPR, 0

CLA

DCA OCTFIG

/CLEAR FLAG FIGURE PRINTED

DCA OCTSPC

/CLEAR SPACE-COUNTER

TAD I DOCTPR

/HIGH ORDER PART

ISZ DOCTPR

JMS OCTPRT

/PRINT OCTAL

TAD I DOCTPR

/LOW ORDER PART

ISZ DOCTPR

JMS OCTPRT

/PRINT OCTAL

TAD OCTSPC

CIA

SNA

/SPACES TO PRINT?

JMP I DOCTPR

/NO, EXIT

IAC

/YES, NUMBER IS ZERO

DCA OCTSPC

/PRINT SPACES

TAD C240

JMS PRINT

ISZ OCTSPC

JMP .-3

TAD C260

/AND A "0"

JMS PRINT

JMP I DOCTPR

/EXIT

Prog 022

?

```

/024 SUBROUTINE TRANSLATES TELEX TO ASCII
/
/CALL :JMS TLXAS WITH TELEX CHARACTER IN AC
/      RETURN CHARACTER IS SHIFT
/      RETURN WITH ASCII CHARACTER IN AC
/
/WHO IS TRANSLATED AS $
/? IS TRANSLATED AS *
/BELL IS TRANSLATED AS ;
/
TLXAS,  0
        AND C37          /MASK 5 BITS
        DCA TLXTMP       /TEMP. STORAGE
        TAD TLXTMP
        SNA
        JMP TLXOUT       /BLANK
        TAD M2
        SNA
        JMP TLXCR        /CARRIAGE RETURN
        TAD M6
        SNA
        JMP TLXNL        /NEW LINE
        TAD M23
        SNA
        JMP TLXSW1       /FIGURESHIFT
        TAD M4
        SNA CLA
        JMP TLXSW0       /LETTERS SHIFT
        TAD TLXTMP       /GET CHARACTER AGAIN
        TAD TLXLA        /ADD LIST ADDRESS
        DCA TLXTMP       /TEMP STORAGE
        TAD TLXSW        /WHICH SIDE?
        SZA CLA
        JMP TLXRGT       /RIGHT SIDE
        TAD I TLXTMP     /GET ASCII 6 BIT
        RTR
        RTR
        RTR
TLXMS,  AND C77          /MASK 6 BIT
        TAD M40
        SPA
        TAD C100         /CHAR<40:300<=CHAR<=337
        TAD C240         /CHAR>40:240<=CHAR<=277
TLXOUT, ISZ TLXAS        /NORMAL RETURN
        JMP I TLXAS
/
TLXRGT, TAD I TLXTMP
        JMP TLXMS
/
TLXSW1, IAC
TLXSW0, DCA TLXSW       /REMEMBER WICH SHIFT
        JMP I TLXAS      /RETURN SHIFT
/
TLXCR,  TAD C215
        JMP TLXOUT
TLXNL,  TAD C212
        JMP TLXOUT
/
TLXLA,  TLXLST
TLXLST, 0

```

	2465	/T	5
C37,	37		
	1771	/O	9
	4040	/SPACE	
	1036	/H	↑
	1654	/N	,
	1556	/e	.
M40,	-40		
	1451	/L	)
	2264	/R	4
	0735	/G	]
	1170	/I	8
	2060	/P	0
	0372	/C	:
	2675	/V	=
	0563	/E	3
	3253	/Z	+
	0477	/D	WHO=\$
	0252	/B	?=*
	2347	/S	'
	3166	/Y	6
	0633	/F	[
	3057	/X	/
	0155	/A	-
	2762	/W	2
	1273	/J	BELL=;
C100,	100		
	2567	/U	7
	2161	/Q	1
	1350	/K	(

/

/VARIABLES

/

TLXTMP,0

TLXSW, 0

/

/GENERAL CONSTANTS

M2,	-2
M4,	-4
M6,	-6
M23,	-23
C77,	77
C212,	212
C215,	215
C240,	240

/025 SUBROUTINE TO TRANSLATE TELEX CHAR TO ASCII

/

/CALL: JMS TLXAS1
/ RETURN IF SHIFT CHARACTER
/ RETURN
/

TLXAS1, 0
AND TLX37
DCA TLXTMP /STORE
TAD TLXTMP
TAD TLXM37
SNA /LETTERSHIFT?
JMP TLXLSH /YES, SET SHIFT
TAD C4
SNA CLA /FIGURESHIFT?
JMP TLXF5H /YES, CLEAR SHIFT
TAD TLXTMP
TAD TLXSH
TAD TLXLST
DCA TLXTMP
TAD I TLXTMP
ISZ TLXAS1
JMP I TLXAS1

/

TLXLSH, TAD TLX40
TLXF5H, DCA TLXSH
JMP I TLXAS1

/

/

TLXLST, .+1
0000 /BLANK
"5
0215 /CR
"9
0240 /SPACE
0000
",
".
0212 /LF
")
"4
0000
"8
"0
":
"=
"3
"+
0205 /TAB
"?
"'
"6
0000
"/
"-
"2
0207 /PELL
TLXSH, 0
"7
"1

```

      "C
TLX40,  40
      0000
      "T
      0215  /CH
      "O
      "  /SPACE
      "H
      "N
      "A
      0212  /LF
      "L
      "R
      "G
      "I
      "P
      "C
      "V
      "E
      "Z
      "D
      "F
      "S
      "Y
      "F
      "X
      "A
      "J
TLX37,  37
      "U
      "O
      "X
TLX037, -37
/VARIABLES
/
TLXTMP,0
/
/GENERAL CONSTANTS
/
C4,      4

```

/026 ROUTINE TO TRANSLATE ASCII TO TELEX

/CALL :JMS ASTLX

/ RETURN

/BEFORE FIRST CALL INITIALIZE ASTSFT:=4 AND

/PRINT A LETTERSHIFT

/NOT EXISTING CHARACTERS ARE PRINTED AS BLANK

/ALTMODE IS TRANSLATED AS FIGURESHIFT

/EUBOUT IS TRANSLATED AS LETTERSHIFT

ASTLX, 0

DCA ASTTMP /TEMP. STORAGE

TAD ASTTMP

AND C77 /MAKE 6 BIT

SNA

JMP ASTOUT+2 /BLANK=BLANK

TAD ASTLA /LISTADDRESS

DCA ASTHLP /LISTADDRESS + 6-BIT CHAR

TAD ASTTMP

TAD M300

SMA CLA

JMP ASTBIG /CHAR>=300;RIGHT HALF OF LIST

TAD I ASTHLP /CHAR<300;LEFT HALF OF LIST

RTI

RTB

RTB

SKP

ASTBIG, TAD I ASTHLP

DCA ASTTMP /TEMP. STORAGE

TAD ASTTMP

AND C77

SNA

JMP ASTOUT+2 /NOT EXISTING IN TELEX:BLANK

AND C40 /GET SHIFT BIT

SZA CLA /WHICH SHIFT

JMP ASTSHF /MUST BE FIGURES

TAD ASTSFT /MUST BE LETTERS

SZA CLA /IS IT LETTERS?

JMP ASTOUT /YES,PRINT CHAR

CLA CLL IAC RTL /+4;NO, MAKE AND PRINT

ASTPSH, DCA ASTSFT

TAD ASTSFT

TAD C33 /MAKE SHIFT

JMS PRINT /PRINT

ASTOUT, TAD ASTTMP

AND C37 /MASK 5 BITS

JMS PRINT /PRINT

JMP I ASTLX /EXIT

ASTSHF, TAD ASTSFT /MUST BE FIGURES

SNA CLA /IS IT FIGURES?

JMP ASTOUT /YES,PRINT CHAR

JMP ASTPSH /NO,MAKE AND PRINT

ASTLA, ASTLST

ASTLST, 0000 /0

0030 /A

0023 /E

0016	/C
0022	/D
6220	/WHO, E
0026	/F
7213	/BELL, G
0005	/H
0014	/I
1032	/NL, J
0036	/K
0011	/L
0207	/CR, M
0006	/N
0003	/O
0015	/P
0035	/Q
0012	/R
0024	/S
0001	/T
0034	/U
0017	/V
0031	/W
0027	/X
0025	/Y
0021	/Z
0000	/[
0000	/ \
0000	/]
0000	/↑
0000	/←
0400	/SPACE
0000	/
0000	/"
0000	/#
0000	/\$
0000	/%
0000	/&
6400	/'
7600	/(
5100	/)
0000	/*
6100	/+
4600	/,
7000	/-
4700	/.
6700	/ /
5500	/0
7500	/1
7100	/2
6000	/3
5200	/4
4100	/5
6500	/6
7400	/7
5400	/8
4300	/9
5600	/:
0000	/;
0000	/<

```
5773      /=, ALTMOD=FIGSHIFT
0000      />
6337      /?, NOBOUT=LETTERS SHIFT
```

```
/
/VARIABLES
```

```
/
ASTSFT,0
ASTTMP,0
ASTHLP,0
```

```
/
/GENERAL CONSTANTS
```

```
/
C33,      33
C37,      37
C40,      40
C77,      77
M300,     -300
```

```

/027 INTERRUPT OUTPUT HANDLER
/WITH HEAD-TAIL COUPLED BUFFER
/
/INITIALIZE ONCE BUFIPT:=BUFOPT:=BUFFER
/      BUFIBO:=0
/
/
/CHARACTER HANDLER
/
/CALL      :JMS BUFINP WITH CHAR IN AC
/      RETURN AC=0
/
/
BUFINP, 0
      DCA BUFTMP      /TEMP. STORAGE
      TAD BUFIBO      /INPTR BEHIND OUTPTR?
      SNA CLA
      JMP BUFPUT      /NO, STORE CHARACTER
      TAD BUFIPT      /YES
      CIA
      TAD BUFOPT      /INPTR = OUTPTR ?
      SNA CLA
      JMP BUFINP+2    /YES, WAIT FOR PLACE TO STORE
BUFPUT, TAD BUFTMP     /NO, GET CHAR
      DCA I BUFIPT
      ISZ BUFIPT
      TAD BUFBUS      /PRINTER BUSY?
      SNA CLA
      6046           /NO, INIT WITH AC=0
      IAC            /YES, SET PRINTER BUSY
      DCA BUFBUS
      TAD BUFIPT
      TAD BUFBND      /END OF BUFFER?
      SZA CLA
      JMP I BUFINP    /NO, EXIT
      TAD BUFADR      /YES, POINTER TO HEAD
      DCA BUFIPT
      IAC            /AND SET INPTR BEHIND OUTPTR
      LCA BUFIBO
      JMP I BUFINP    /EXIT

```

/028 DEVICE INTERRUPT HANDLER

```

/
/CALL      :JMP BUFOUT      /DEVICE INTERRUPT DETECTED!
/          ROUTINE RETURNS TO INTERRUPT RESTORE "EXIT"
/

```

```

BUFOUT, CLA
        6042                /CLEAR DEVICE FLAG
        TAD BUFIPT
        CIA
        TAD BUFOPT          /INPTR = OUTPTR ?
        SZA CLA
        JMP BUFGT           /NO, GET CHAR AND PRINT
        TAD BUFIBO          /YES, INPTR BEHIND OUTPTR?
        SZA CLA
        JMP BUFGT           /YES, GET AND PRINT
        DCA BUFBUS          /NO, PRINTER READY
        JMP EXIT

```

```

/
BUFGT,  TAD I BUFOPT        /GET CHAR
        ISZ BUFOPT
        6044                /PRINT CHAR
        CLA
        TAD BUFOPT
        TAD BUFBND          /END OF BUFFER?
        SZA CLA
        JMP EXIT            /NO, END OF ROUTINE
        TAD BUFBDR          /YES, POINTER TO HEAD
        DCA BUFOPT
        DCA BUFIBO          /RESET INPTR BEHIND OUTPTR
        JMP EXIT            /END OF HANDLING

```

/GENERAL INTERRUPT RETURN ROUTINE

```

/
EXIT,   CLA CLL
/       TAD LINK
/       BAL                /RESTORE LINK
/       TAD ACCU           /RESTORE ACCU
/       ION                /INTERRUPT ON
/       JMP I 0

```

/VARIABLES

```

BUFTMP, 0
BUFIBO, 0
BUFIPT, 0
BUFOPT, 0
BUFBUS, 0
BUFBND, -BUFEND
BUFBDR, BUFFER
BUFFER, 0
/
*BUFFER+200
BUFEND, 0

```

```

/029 SUBROUTINE READS OR WRITES DECTAPE
/IN BOTH DIRECTIONS
/
/CALL    :JMS DCTAPE
/        DEFINING BITS
/        BLOCKNUMBER
/        -# WORDS (12 BITS)
/        BUFFERADDRESS-1
/        ERROR RETURN OR RETURNADDRESS
/        NORMAL RETURN OR RETURNADDRESS
/
/DEFINING BITS:BIT 0,1,2      UNIT NUMBER
/      3      0=FORWARD;1=REVERSE
/      4,5     0 (NOT USED)
/      6,7,8   MEMORY FIELD
/      9       0 (NOT USED)
/     10       0=DIRECT RETURN;1=INDIRECT
/     11       0=READ;1=WRITE
/
DTCA=    6762
DTXA=    6764
DTLR=    6774
DTRA=    6761
DTSF=    6771
LTPR=    6772
/
DCTAPE, 0
CLA
TAD I DCTAPE    /DEFINING BITS
DCA DCTCOD      /SAVE
ISZ DCTAPE
TAD DCTCOD
AND C7400       /UNIT# & DIRECTION BIT
TAD C10         /SEARCH MOLE
DTCA DTXA       /I/O
DTLR           /CLEAR FIELD REGISTER
TAD DCTWC       /WORD COUNT ADDRESS
DCA I DCTCA     /WORD COUNT:=BLKNE ADDRESS
TAD C200        /GO BIT
DCTCNT, JMS DCTTRN /TURN DECT AND WAIT FOR FLAG
TAD I DCTWC     /READ NUMBER
CIA            /NEG.
TAD I DCTAPE    /NUMBER TO FIND
SNA
JMP DCTMAY     /FOUND, CHECK DIRECTION
DCTSET, CLL RAL /SAVE SIGN DIFFERENCE
CLA
DTRA
AND C400       /DIRECTION BIT
SNA CLA
CML           /IS FORWARD
SNL           /IS REVERSE
TAD C400      /CHANGE DIRECTION
JMP DCTCNT    /DIRECTION OK, NEXT NUMBER
/
DCTMAY, TAD DCTCOD /UNIT# & DIRECTION
AND C400      /MASK DIRECTION
SNA CLA
JMP DCTRFW    /MUST BE FORWARD
DTRA         /MUST BE REVERSE

```

```

      AND C400
      SZA CLA
      JMP DCTEDR      /IS REVERSE, GO READ OR WRITE
      JMP DCTCNT      /IS FORWARD, CONT SEARCHING
      /              /MUST BE FORWARD
DCTRFW, DTRA
      AND C400
      SNA CLA
      JMP DCTEDR      /IS FORWARD, GO READ OR WRITE
      JMP DCTCNT      /IS REVERSE, CONT SEARCHING
/
/
DCTEDR, ISZ DCTAPE
      TAD I DCTAPE    /-# WORDS
      DCA I DCTWC      /SET WORD COUNT
      ISZ DCTAPE
      TAD I DCTAPE    /CORE ADDRESS-1
      DCA I DCTCA      /SET CURRENT ADDRESS
      TAD DCTCOD
      DTLE            /LOAD FIELD BITS
      TAD DCTCOL
      RAR
      SZL CLA          /READ OR WRITE?
      TAD C20          /WRITE
      TAD C130         /WRITE
      DTXA
      DTSE DTRB
      JMP .-1
      ISZ DCTAPE      /ADVANCE TO ERROR RETURN
      SNA CLA          /SKIP IF ERROR
      ISZ DCTAPE      /NORMAL RETURN
      TAD DCTCOL      /DIRECT OR INDIRECT?
      RTR
      SNL CLA
      JMF .+3          /DIRECT
      TAD I DCTAPE    /INDIRECT, PREPARE
      DCA DCTAPE
      DTRA
      AND C200        /GO BIT
      TAD C2           /PRESERVE ERROR FLAG
      DTXA            /STOP TAPE
      JMP I DCTAPE     /READY, EXIT
/
/
DCTTEN, 0
      DTXA
      DTSE DTRB
      JMP .-1
      SPA
      JMP DCTERR
      CLA
      JMP I DCTTEN
/
DCTERR, RTL
      RAL
      CLA CML
      SNL
      TAD C400
      JMP DCTCNT-1

```

```
/
/
/VARIABLES
/
DCTCOD, 0
DCTWC, 7754
DCTCA, 7755
/
/GENERAL CONSTANTS
/
C2,      2
C10,     10
C20,     20
C130,    130
C200,    200
C400,    400
C7400,   7400
```

```

/030 SUBROUTINE TO PACK CHARACTERS (TSS8)
/THREE CHARACTERS IN TWO WORDS (TSS8 FORMAT)
/PACKED:111111112222
/      222233333333
/
/
/CALL   :JMS PACK
/      ADDRESS INPUTBUFFER
/      ADDRESS OUTPUTBUFFER
/      RETURN
/
/ROUTINE USES AUTO INDEX 10 AND 11
/
/FORMAT INPUTBUFFER= 1 CHAR/WRD
/LENGTH OUTPUTBUFFER= 200
/LENGTH INPUTBUFFER= 300
/
PACK,   0
        TAD PCKBFL          /-BUFFERLENGTH OUTPUTBUFFER
        STL FAF             /DEVIDE BY 2
        DCA PCKCNT
        CLA CMA              /-1
        TAD I PACK          /ADDRESS INPUTBUFFER
        DCA 10
        ISZ PACK
        CMA                  /-1
        TAD I PACK          /ADDRESS OUPUTBUFFER
        DCA 11
        ISZ PACK
PCKLOP, TAD I 10             /GET CHAR
        CLL RTL
        RTL
        DCA PCKTMP          /TEMP. STORAGE
        TAD I 10            /NEXT CHAR
        RTR
        DCA PCKTP1
        TAD PCKTP1
        AND C17
        TAD PCKTMP
        DCA I 11            /FIRST WORD
        TAD PCKTP1          /PICK UP AGAIN
        RAR
        AND C7400
        TAD I 10            /NEXT CHAR
        DCA I 11            /SECOND WORD
        ISZ PCKCNT          /BUFFER FULL ?
        JMP PCKLOP          /NO,PACK NEXT
        JMP I PACK          /YES, EXIT
/
/VARIABLES
/
PCKCNT,0
PCKTMP,0
PCKTP1,0
PCKBFL,-200
/
/GENERAL CONSTANTS
C17,    17
C7400, 7400

```

```

/031 SUBROUTINE PACKS CHARACTERS ONE BY ONE (TSS8)
/THREE CHARACTERS IN TWO WORDS (TSS8 FORMAT)
/PACKED:111111112222
/      222233333333
/
/
/

```

```

/CALL      :JMS PCKSGL WITH CHAR IN AC
/      ADDRESS OF OUTPUTBUFFER
/      RETURN BUFFER FULL
/      RETURN NARMAL      AC=0
/

```

```

/INITIALIZE CE PCKSWT:=0
/
/
/

```

```

PCKSGL, 0
      ISZ PCKSWT      /INITIALIZE?
      JMS PCKINI      /YES
      DCA I PCKRP      /NO PUT CHAR IN TEMP BUF
      ISZ PCKRP      /INCREMENT POINTER
      ISZ PCKRCT      /3 CHAR'S IN TEMP BUF?
      JMP PCKNRM      /NO, NORMAL EXIT
      JMS PCKRES      /YES, RESET POINTER TEMP. BUF
      TAD I PCKRP      /GET FIRST CHAR
      ISZ PCKRP
      CLL RTL
      RTL
      DCA I PCKPTR      /TEMP STORAGE
      TAD I PCKRP      /GET SECOND CHAR
      ISZ PCKRP
      RTH
      RTR
      DCA PCKSWT
      TAD PCKSWT      /TEMP. STORAGE
      AND C17      /MOST SIGN. 4 BITS
      TAD I PCKPTE
      DCA I PCKPTR      /FIRST WORD
      ISZ PCKPTR
      TAD PCKSWT
      RAR
      AND C7400      /LEAST SIGNIFICANT 4 BITS
      TAD I PCKRP      /GET THIRD CHAR
      DCA I PCKPTR      /SECOND WORD
      ISZ PCKPTE
      JMS PCKRES      /RESET POINTER TEMP BUF
      ISZ PCKCNT      /BUFFER FULL?
      JMP PCKNRM      /NO
      DCA PCKSWT      /YES SET SWITCH
      JMP PCKEND
/
/

```

```

PCKNRM, CMA
      DCA PCKSWT      /SET SWITCH
      ISZ PCKSGL
PCKEND, ISZ PCKSGL
      JMP I PCKSGL
/
/

```

```

PCKINI, 0
      DCA PCKSWT      /TEMP STORAGE

```

```

JMS PCKRES          /SET POINTER TEMP BUF
TAD I PCKSGL        /GET BUFFERADDRESS
DCA PCKPTR
TAD PCKBFL
STL KAR             /BUFFERSIZE DEVIDED BY 2
DCA PCKCNT
TAD PCKSWT
JMP I PCKINI

/
/
PCKRES, 0
TAD M3
DCA PCKRCT          /TEMP BUF IS 3 WORDS
TAD PCKRBA          /TEMP BUF ADDRESS
DCA PCKRP
JMP I PCKRES

/
/
/VARIABLES
/
PCKSWT, 0
PCKPTR, 0
PCKRP, 0
PCKRCT, 0
PCKCNT, 0
PCKRBA, PCKRB
PCKBFL, -400
PCKRB, 0
0
0

/
/GENERAL CONSTANTS
M3, -3
C17, 17
C7400, 7400

```

```

/032 SUBROUTINE TO PACK CHARACTERS ONE BY ONE (TSS8)
/THREE CHARACTERS IN TWO WORDS (TSS8 FORMAT)
/PACKED:111111112222
/      222233333333
/
/
/CALL   :JMS DSOUT WITH CHAR IN AC
/      RETURN BUFFER FULL
/      RETURN NORMAL
/
/INITIALIZE ONCE DSPTR TO BUFFERADDRESS
/AND DSCNT:=DSBFL DEVIDED BY 2
/
BSW=7002
DSBUF=400
/
/
DSOUT,  0
        DCA DSTMP      /TEMP. STORAGE
        FAR
        DCA DSLNK      /SAVE LINK
        TAD DSCNTW     /FIRST, SECOND OR THIRD CHAR
        CLL RTE
        SNL SMA CLA
        JMP DSFRST     /FIRST CHAR OF THREE
        SNL
        JMP DSSEC      /SECOND CHAR OF THREE
        TAD DSTMP      /THIRD CHAR
        TAD I DSPTR
        DCA I DSPTR    /PUT IN BUFFER
        DCA DSCNTW     /RESET CHAR COUNT
        ISZ DSPTR
        ISZ DSCNT      /BUFFER FULL ?
        JMP DSEX3      /NO, EXIT
        TAD DSBFA      /YES, RESET POINTER
        DCA DSPTR
        TAD DSBFL      /-BUFFERLENGTH
        STL RAR        /DEVIDE BY 2
        DCA DSCNT
        TAD DSLNK      /RESTORE LINK
        CLL RAL
        JMP I DSOUT    /EXIT BUFFER FULL
DSSEC,  TAD DSTMP
        CLL RTL
        BSW            /BYTE SWAP
        AND C77
        TAD I DSPTR
        DCA I DSPTR
        ISZ DSPTR
        TAD DSTMP
        AND C17
        BSW
        CLL RTL
        DCA I DSPTR
        JMP DSEX2
DSFRST, TAD DSTMP
        CLL RTL
        RTL
        DCA I DSPTR
DSEX2,  ISZ DSCNTW
DSEX3,  TAD DSLNK      /RESTORE LINK

```

CLL RAL
ISZ DSOUT
JMP I DSOUT /NORMAL EXIT

/

/VARIABLES

/

DSBFL, -400
DSEFA, DSEUF /OUTPUT BUFFER ADDRESS
DSLNE, 0
DSTMP, 0
DSCNTW, 0
DSCNT, 0
DSPTR, 0

/

/GENERAL CONSTANTS

C17, 17
C77, 77

```

/033 SUBROUTINE TO UNPACK CHARACTERS (TSS8)
/PACKED THREE CHARACTERS IN TWO WORDS (TSS8 FORMAT)
/
/PACKED:111111112222
/      222233333333
/CALL  :JMS UNPACK
/      ADDRESS OF INPUTBUFFER
/      ADDRESS OF OUTPUTBUFFER
/      RETURN
/
/ROUTINE USES AUTO-INDEX 10
/
UNPACK, 0
    TAD UNPBFL      /-BUFFERLENGTH INPUTBUFFER
    STL RAR         /DEVIDE BY 2
    DCA UNPCNT
    TAD I UNPACK    /ADDRESS INPUTBUFFER
    DCA UNPPTR
    ISZ UNPACK
    CLA CMA         /-1
    TAD I UNPACK    /ADDRESS OUTPUTBUFFER
    DCA 10
    ISZ UNPACK
UNPLOP, TAD I UNPPTR
    RTR
    RTR
    AND C377
    DCA I 10        /FIRST CHAR
    TAD I UNPPTR    /PICK UP CHAR AGAIN
    CLL RTL
    RTL
    AND C360
    DCA UNPTMP      /TEMP. STORAGE
    ISZ UNPPTR
    TAD I UNPPTR
    CLL RAL
    RTL
    RTR
    AND C17
    TAD UNPTMP
    DCA I 10        /SECOND CHAR
    TAD I UNPPTR
    AND C377
    DCA I 10        /THIRD CHAR
    ISZ UNPPTR
    ISZ UNPCNT      /READY ?
    JMP UNPLOP      /NO, CONTINUE
    JMP I UNPACK    /YES, EXIT
/
/VARIABLES
/
UNPPTR, 0
UNPTMP, 0
UNPCNT, 0
UNPBFL, -400
/
/GENERAL CONSTANTS
C17,      17
C360,     360
C377,     377

```

```

/034 SUBROUTINE UNPACKS CHARACTERS ONE BY ONE (TSS8)
/PACKED THREE CHARACTERS IN TWO WORDS (TSS8 FORMAT)
/PACKED:111111112222
/      222233333333
/
/
/CALL   :JMS UNPSGL
/      ADDRESS INPUTBUFFER
/      RETURN BUFFER EMPTY      AC=0
/      NORMAL RETURN AC=CHAR.
/
/INITIALIZE ONCE UNPRBF:=UNPBEF:=UNPCNT:=0
/
/
UNPSGL, 0
    CLA CLL
    TAD UNPRBF      /ARE THERE CHAR'S IN
    SZA CLA         /TEMP. BUFFER ?
    JMP UNPGET      /YES, GET ONE
    TAD UNPBEF      /NO, INPUTBUFFER EMPTY ?
    SZA CLA
    JMP UNPEMP      /YES, RETURN END OF BUFFER
    TAD UNPCNT      /NO OR YES, MUST I
    SNA CLA         /START UP POINTERS ?
    JMS UNPINI      /YES, PLEASE DO
    TAD UNPRBA      /NO, JUST UNPACK NEXT WORDS
    DCA UNPRP
    TAD I UNPPTR    /NEXT WORD FROM INPUTBUF
    RTE
    RTE
    AND C377
    DCA I UNPRP     /FIRST CHAR IN TEMP. BUF
    ISZ UNPRP
    TAD I UNPPTR    /GET WORD AGAIN
    CLL RTL
    RTL
    AND C360
    DCA I UNPRP     /TEMP. STORAGE
    ISZ UNPPTR
    TAD I UNPPTR    /NEXT WORD
    CLL RAL
    RTL
    RTL
    AND C17
    TAD I UNPRP
    DCA I UNPRP     /SECOND CHAR
    ISZ UNPRP
    TAD I UNPPTR    /THAT WORD AGAIN
    ISZ UNPPTR
    AND C377
    DCA I UNPRP     /THIRD CHAR
    TAD UNPRBA      /RESET POINTER TEMP. BUF
    DCA UNPRP
    CLA CLL CMA RTL /-3
    DCA UNPRCT      /3 CHAR'S IN TEMP. BUF
    ISZ UNPCNT      /INPUTBUFFER EMPTY ?
    JMP UNPGET      /NO, GET CHAR NOW
    IAC             /YES, SET FLAG BUFFER EMPTY
    DCA UNPBEF      /AND THEN GET CHAR
UNPGET, ISZ UNPRCT  /LAST FROM TEMP. BUF ?
    IAC             /NO, SET FLAG

```

```

        DCA UNPRBF      /YES RESET FLAG
        TAD I UNPPF     /GET CHAR
        ISZ UNPRP
        ISZ UNPSGL      /NORMAL EXIT
UNPEMT, ISZ UNPSGL
        JMP I UNPSGL

/
UNPEMP, DCA UNPBEF      /RESET FLAG
        JMP UNPEMT      /AND EMPTY BUFFER RETURN

/
UNPINI, 0
        DCA UNPRBF      /RESET FLAG
        TAD I UNPSGL    /ADDRESS INPUTBUFFER
        DCA UNPPTH
        TAD UNPBFL      /-LENGTH OF BUFFER
        STL RAR         /DIVIDE BY 2
        DCA UNPCNT
        JMP I UNPINI

/
/VARIABLES
UNPBFL, -400
UNPCNT, 0
UNPRCT, 0
UNPRP, 0
UNPPTH, 0
UNPRPF, 0
UNPBEF, 0
UNPREA, UNPRB
UNPRB, 0
        0
        0

/
/GENERAL CONSTANTS
C17, 17
C360, 360
C377, 377

```

/035 SUBROUTINE TO READ A NAME FROM KEYBOARD

/

/CALL :JMS RDNAME

/ WORD 1,2 CHAR'S FROM NAME IN EXCESS-40 CODE

/ WORD 2,2 CHAR'S FROM NAME

/ WORD 3,2 CHAR'S FROM NAME

/ERROR RETURN

/NORMAL RETURN

/

/ROUTINE USES AUTO INDEX 10,ROUTINES READ,PRINT

/AND CRLF

/

BSI=7002

EUFA DR=400

/

/

RDNAME, 0

TAD RDNER /ADDRESS ASCII BUFFER

DCA RDPTR

DCA RDCNT /CHAR. COUNTER

RDIN, JMS READ /READ CHAR FROM KEYB.

DCA RDCHAR

TAL RDCHAR

TAD RDMRO /RUBOUT ?

SNA

JMP RDROS /YES, TO SERVICE

TAL RDMCRN /NO, CARRIAGE RETURN ?

SNA

JMP RDTWNR /YES, TO SERVICE

TAD RDMLFD /NO, LINE FEED

SNA

JMP RDTWNR /YES, SAME SERVICE AS CR

TAD RDMSPE /NO, CHAR>240 ?

SPA SNA CLA

JMP RDFTNM /NO, ERROR RETURN

TAL RDCHAR /YES, IN BUFFER

DCA I RDPTR

ISZ RDCNT /+ # CHAR'S

ISZ RDPTR

JMP RDIN /NEXT CHAR

JMP RDFTNM /4K BUFFER FULL, ERROR

/

RDROS, TAD RDCNT /ALREADY SOMETHING IN BUFFER?

SNA CLA

JMP RDIN /NO, STUPID RO-TYPER!

CMA /YES COUNTER BACK 1

TAD RDCNT

DCA RDCNT

CMA

/AND POINTER BACK 1

TAD RDPTR

DCA RDPTR

TAD I RDPTR /PRINT REMOVED CHAR

JMS PRINT

JMP RDIN /END RO-SERVICE

/

RDTWNR, JMS CRLF /PRINT CR LF

TAD RDCNT

SNA

JMP RDFTNM /NAME WITHOUT CHAR'S IS RUBBISH

TAD M6

```

SMA SZA          /SIX OR LESS CHAR'S
CLA              /MORE, THAN MAKE IT SIX
TAD C6
CIA
DCA RDCNT        /-# CHAR'S
TAD RDNMBF       /BUFFER ADDRESS
DCA RDPTR
TAD RDNAME       /PACKED NAME ADDRESS
DCA RDTMP        /PLACED UNDER CALLING
TAD RDTMP
DCA 10
DCA I 10
DCA I 10         /CLEAR BUFFER
RDNXT, TAD I RDPTR /MAKE EXCESS-40 CODE
TAD C240
AND C77
BSW
DCA I RDTMP
ISZ RDPTR
ISZ RDCNT
SKP
JMP RDNMOK       /READY READING NAME
TAD I RDPTR      /NOT READY NEXT CHAR
TAD C240
AND C77
TAD I RDTMP
DCA I RDTMP
ISZ RDTMP
ISZ RDPTR
ISZ RDCNT
JMP RDNXT        /NEXT CHAR'S
RDNMOK, ISZ RDNAME /NORMAL RETURN
RDTNM, ISZ RDNAME
ISZ RDNAME
ISZ RDNAME
JMP I RDNAME     /EXIT

/
/
/VARIABLES
/
RDNMBF, BUFADR   /ADDRESS BUFFER
RDPTR, 0
RDCNT, 0
RECHAR, 2
RDTMP, 0
RDMRO, -377
RDMCRN, 377-215
RDMLEF, 215-212
RDMSPF, 212-240
/
/GENERAL CONSTANTS
M6, -6
C6, 6
C77, 77
C240, 240

```

/036 SUBROUTINE SEARCHES NAME IN DN-BLOCKS (DISMON.)
 /(DISK MONITOR SYSTEM)

/

/CALL	:JMS DNSRC	
/	NA	FIRST TWO CHAR'S IN EXCESS-40 6 BIT
/	ME	LAST " " " "
/	RETURN NAME NOT FOUND	AC=0
/	RETURN NAME FOUND	AC=INT. FILE NR

/

/SUBROUTINE USES AUTO INDEX 11 AND MONITOR DISK HANDLER

/

BUFFER=400

/

DNSRC, 0

	TAD C177	/# FIRST DN-BLOCK	
	JMS DNSRBK	/READ BLOCK	<i>See below</i>
	TAD I DNSRC		
	CIA		
	DCA DNSMNA	/- TWO CHAR'S OF NAME	
	ISZ DNSRC		
	TAD I DNSRC		
	CIA		
	DCA DNSMME	/- LAST CHAR'S	
	ISZ DNSRC		
DNSBLK,	CLA CLL IAC RAL	/+2	
	TAD DNSBFA	/BUFFER ADDRESS	
	DCA 11		
	TAD M31	/# ENTRIES IN ONE BLOCK	
	DCA DNSCNT		
DNSNXT,	TAD I 11	/FIRST HALF OF NAME	
	TAD DNSMNA	/COMPARE WITH NAME TO LOOK FOR	
	SZA CLA	/EQUAL?	
	JMP DNSNOT	/NO TRY NEXT NAME	
	TAD I 11	/YES, TEST 2ND. HALF TOO	
	TAD DNSMME		
	SZA CLA	/EQUAL?	
	JMP DNSNT1	/NO NEXT NAME	
	ISZ 11		
	ISZ 11		
	TAD I 11		
	AND C7	/MASK OF INT FILE #	
	ISZ DNSRC		
DNSERR,	JMP I DNSRC		
/			
DNSNOT,	CLA IAC		
DNSNT1,	TAD C3		
	TAD 11		
	DCA 11		
	ISZ DNSCNT	/END OF THIS BLOCK?	
	JMP DNSNXT	/NO, COMPARE NEXT NAME	
	TAD DNSBLK	/YES NEXT BLOCK?	
	SNA		
	JMP DNSERR	/NO, NAME NOT FOUND	
	JMS DNSRBK	/READ THAT BLOCK	<i>See below</i>
	JMP DNSBLK		
/			
DNSRBK,	0		
	DCA FSTBLK		
	TAD C3		

```

        DCA FUNCTI
        TAD DNSBFA
        DCA BUFADR
        DCA DNSLNK
        JMS I SYSIO      /MONITOR DISK HANDLER

FUNCTI,0
FSTBLK,0
BUFADR,0
DNSLNK,0
        HLT              /ERROR RETURN
        JMP I DNSRBK

/
/VARIABLES
/
SYSIO, 7642
DNSMNA,0
DNSMME,0
DNSCNT,0
DNSBFA,BUFFER
/
/GENERAL CONSTANTS
C3,      3
C7,      7
C177,    177
M31,     -31

```

```

/037 SUBROUTINE SEARCHES UNUSED BLOCK ON DISK (DISKMON)
/AND RESERVES IT FOR FILE      (DISK MONITOR SYSTEM)
/

```

```

/CALLING:JMS SAMFIL      WITH INT. FILE NR IN AC
/      RETURN DISK FULL
/      RETURN NORMAL WITH BLOCKNR IN AC
/
/

```

```

SAMFIL, 0
      DCA SAMSAV      /SAVE INT FILE #
      JMS SAMSRC      /SEARCH FOR EMPTY BLOCK
      JMP I SAMFIL    /NOT FOUND SO DISK FULL
      CLA CMA         /BLOCKNR STILL IN SAMBKN
      TAD 10          /AUTO INDEX STILL ON SPOT
      DCA 10
      TAD SAMMSK      /WHICH HALF IS MASK
      TAD M77
      SNA CLA         /LEFT OR RIGHT?
      JMP SAMRGT      /MASK IS ON RIGHT HALF
      TAD SAMSAV
      CLL RTL         /PUT INT FILE # ON LEFT HALF
      ETL
      ETL
      DCA SAMSAV
      JMP .+3
SAMRGT, TAD I 10
      TAD SAMSAV      /ADD INT FILE #
      DCA SAMSAV      /TEMP. STORAGE
      CMA
      TAD 10
      DCA 10
      TAD SAMSAV      /PUT IN BUFFER
      DCA I 10
      TAD C5
      DCA FUNCTI
      TAD SAMBFA
      DCA BUFADR
      JMS SAMRDB      /RESTORE SAM ON DISK
      ISZ SAMFIL
      TAD SAMBKN      /GET BLOCKNR
      JMP I SAMFIL    /RETURN

```

```

/
/VARIABLES
/

```

```

SAMSAV, 0
/

```

```

/GENERAL CONSTANTS
/

```

```

M77,      -77
C5,        5

```

Page 038

See p 61 top

```

/038 SUBROUTINE SEARCHES INT. FILE # (DISKMON)
/IN SAMBLOCKS (DISK MONITOR SYSTEM)
/
/CALL :JMS SAMSRC WITH INT. FILE # IN AC
/ RETURN NUMBER NOT FOUND; AC=0
/ RETURN NR FOUND, AC=# FIRST BLOCK FROM FILE
/
/SUBROUTINE USES AUTO INDEX 10 AND MONITOR DISK HANDLER
/
/
BUFFER=400
/
/
SAMSRC, 0
      DCA SAMIFN      /INT FILE # TO SEARCH FOR
      TAD SAMIFN      /MAKE IT TWO IN ONE WORD
      CLL RTL
      RTL
      RTL
      TAD SAMIFN
      DCA SAMIFN
      DCA SAMEKN      /COUNTER FOR BLOCKNR
      TAD C200        /# FIRST SAMBLOCK
SAMBK, DCA BLKNR
      TAD C3          /READ FUNCTION
      DCA FUNCTI
      TAD SAMPFA      /BUFFER ADDRESS
      DCA BUFADR
      JMS SAMRDB      /READ BLOCK
SAMS, TAD C77
      DCA SAMMSK      /SEARCH RIGHT HALF
      TAD M200
      DCA SAMCNT      /200 WORDS
      CMA
      TAD SAMBFA
      DCA 10
      SKP
SAMNXT, ISZ SAMBKN      /COUNT BLOCKNR
      TAD I 10        /GET WORD
      AND SAMMSK      /MASK
      CIA            /NEGATIVE
      DCA SAMTMP      /TEMP. STORAGE
      TAD SAMIFN      /INT FILE # TO SEARCH FOR
      AND SAMMSK      /MASK CORRECT HALF
      TAD SAMTMP      /SAME #?
      SNA CLA
      JMP SAMFND      /YES, FOUND IT
      ISZ SAMCNT      /NO, MORE IN THIS HALF?
      JMP SAMNXT      /YES, SEARCH
      ISZ SAMBKN      /NO, UPDATE BLOCKNR
      TAD SAMMSK      /WHERE WERE WE SEARCHING?
      AND C7700
      SZA CLA        /LEFT OR RIGHT HALF
      JMP .+3        /LEFT HALF, BOTH SIDES DONE
      TAD C7700      /RIGHT HALF, SO DO LEFT NOW
      JMP SAMSR+1
      TAD SAMLNK      /LAST SAMBLOCK?
      SNA
      JMP SAMNOT      /YES, SO NOT FOUND
      JMP SAMRBK      /NO, READ NEXT BLOCK

```

— See next page

```

/
/
SAMPND, ISZ SAMSRC
        TAD SAMBKN
SAMNOT, JMP I SAMSRC
/
/
SAMPDB, 0
        JMS I SYSIO      /MONITOR DISK HANDLER
FUNCTI, 0          /READ=3, WRITE=5
ELKNR, 0          /BLOCKNR
BUFADR, 0          /BUFFERADDRESS
SAMLNK, 0          /NR NEXT BLOCK, 0=LAST BLOCK
        HLT          /ERROR RETURN, SYSTEM ERROR
        JMP I SAMRDB
/
/VARIABLES
/
SAMTMP, 0
SAMIEN, 0
SAMPKN, 0
SAMMSK, 0
SAMCNT, 0
SAMEFA, BUFFER
SYSIO, 7642
/
/GENERAL CONSTANTS
/
C3,      3
C77,     77
C200,    200
C7700,   7700
M200,    -200

```

```

/039 SUBROUTINE READS OR WRITES ON DISK (TSS-8)
/
/BEFORE CALLING CALCULATE DISKADDRESS AND
/PUT IN HIOR AND LOWOR
/
/CALL      :JMS DFILE
/          FUNCTION (RFILE OR WFILE)
/          INTERNAL FILE NUMBER
/          -# WORDS
/          CORE ADDRESS
/          ERROR RETURN
/          NORMAL RETURN
/
/FILE MUST BE OPEN !!!!!!!!!!!
/
/
DFILE, 0
      TAD I DFILE      /GET FUNCTION
      DCA DFINST
      ISZ DFILE
      TAD I DFILE      /GET INT. FILE NR
      DCA W6BUF+1
      ISZ DFILE
DFTRY, TAD I DFILE      /-#WORDS
      DCA W6BUF+2
      ISZ DFILE
      CLA CMA
      TAD I DFILE      /CORE ADDRESS
      DCA W6BUF+3
      ISZ DFILE
      TAD W6AD          /ADDRESS 6 WORD BUFFER
DFINST, 0              /DO FUNCTION
      TAD W6BUF+5        /ERROR WORD
      SNA
      JMP DFOKE          /NO ERROR
      CLL RTR           /ERROR
      SZL SNA CLA
      SKP CLA           /ERROR=2
      JMP DFERR          /ERROR IS NOT 2
      IAC
      DCA W2BUF+2        /ERROR IS FILE FULL
      TAD W2AD          /SO MUST EXTEND FILE
      EXT               /EXTENDING WITH ONE SEGMENT
      SZA CLA
      JMP DFERR          /ERROR: DISK FULL
      TAD DFSEGA        /ADDRESS LIST SEGMENTCOUNTERS
      TAD W2BUF         /INT FILE NR
      DCA W6BUF+2        /TEMP USE
      ISZ I W6BUF+2      /INCREMENT COUNTER
      CLL CLA CMA RAL   /-2
      TAD DFILE
      DCA DFILE
      JMP DFTRY          /GO TRY AGAIN NOW
DFOKE, ISZ DFILE
DFERR, JMP I DFILE
/
/VARIABLES
/
DFSEGA, DFSEG0
DFSEG0, 0              /# SEGMENTS FILE 0

```

DFSEG1,0		/# SEGMENTS FILE 1
DFSEG2,0		/# SEGMENTS FILE 2
DFSEG3,0		/# SEGMENTS FILE 3
W6AD,	W6BUF	
W2AD,	W2BUF	
W6BUF,		
HIOR,	0	/HIGH ORDER DISK ADDRESS
W2BUF,	0	/INT FILE NR
	0	/-# WORDS;# SEG'S TO EXT
	0	/CORE ADDRESS-1
LOWOR,	0	/LOW ORDER DISK ADDRESS
	0	/ERROR WORD

