digital

OS/8 HANDOUTS

EDUCATION

OS/8 HANDOUTS

DIGITAL EQUIPMENT CORPORATION

NOTE

This handbook is for information purposes and is subject to change without notice

Associated Documents

Introduction to Programming
OS/8 Handbook
OS/8 Software Support Manual

APRIL 22, 1974

PREFACE

The primary purpose of this booklet is to serve as an aid for those students taking either the OS/8 STANDARD or the OS/8 ACCELERATED COURSE. It is not intended as a substitute for taking notes in class but rather as a supplement to those notes.

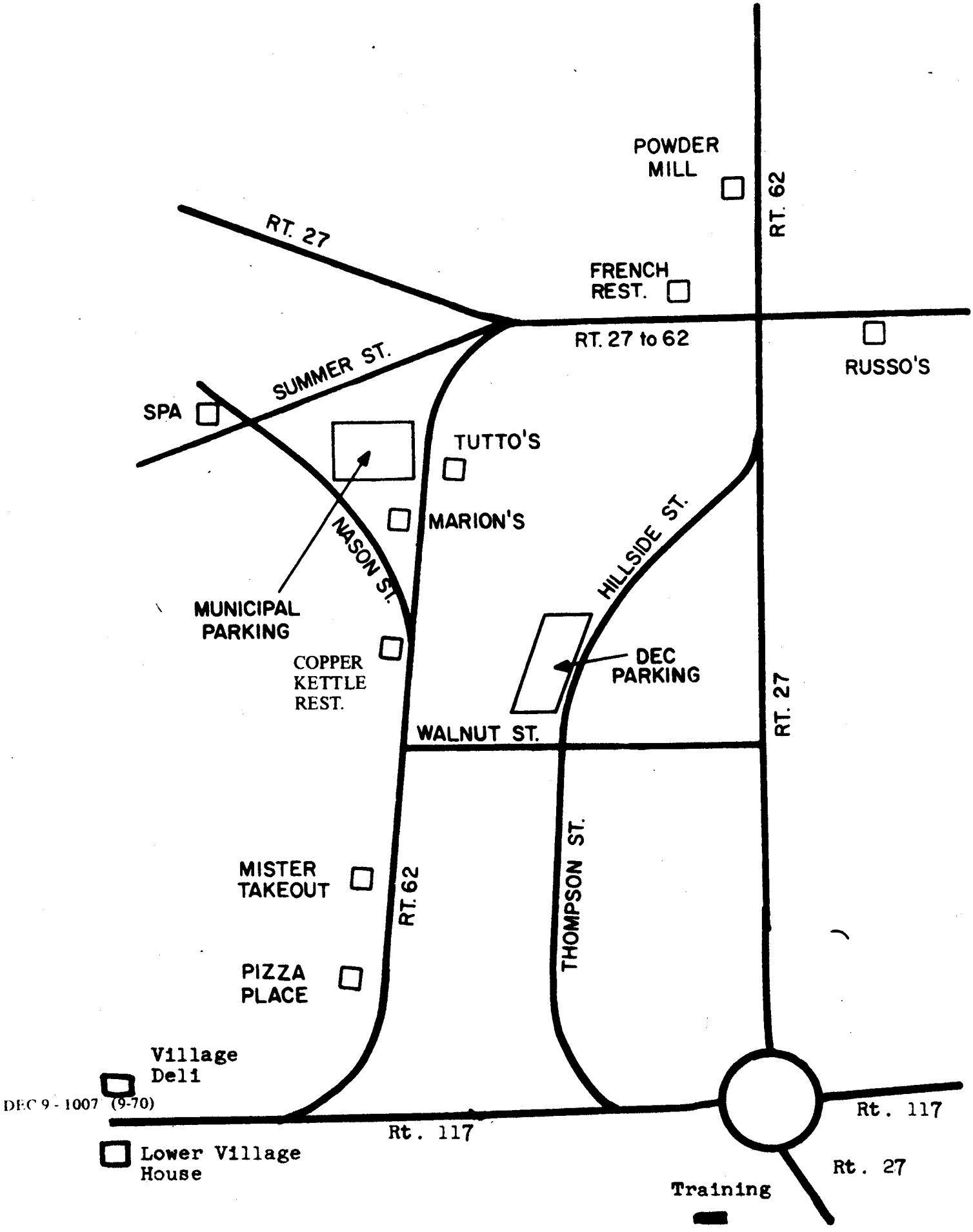
The student is encouraged to separate these pages, to write on them, and in general, to use these handouts in any way that would make his learning of OS/8 easier.

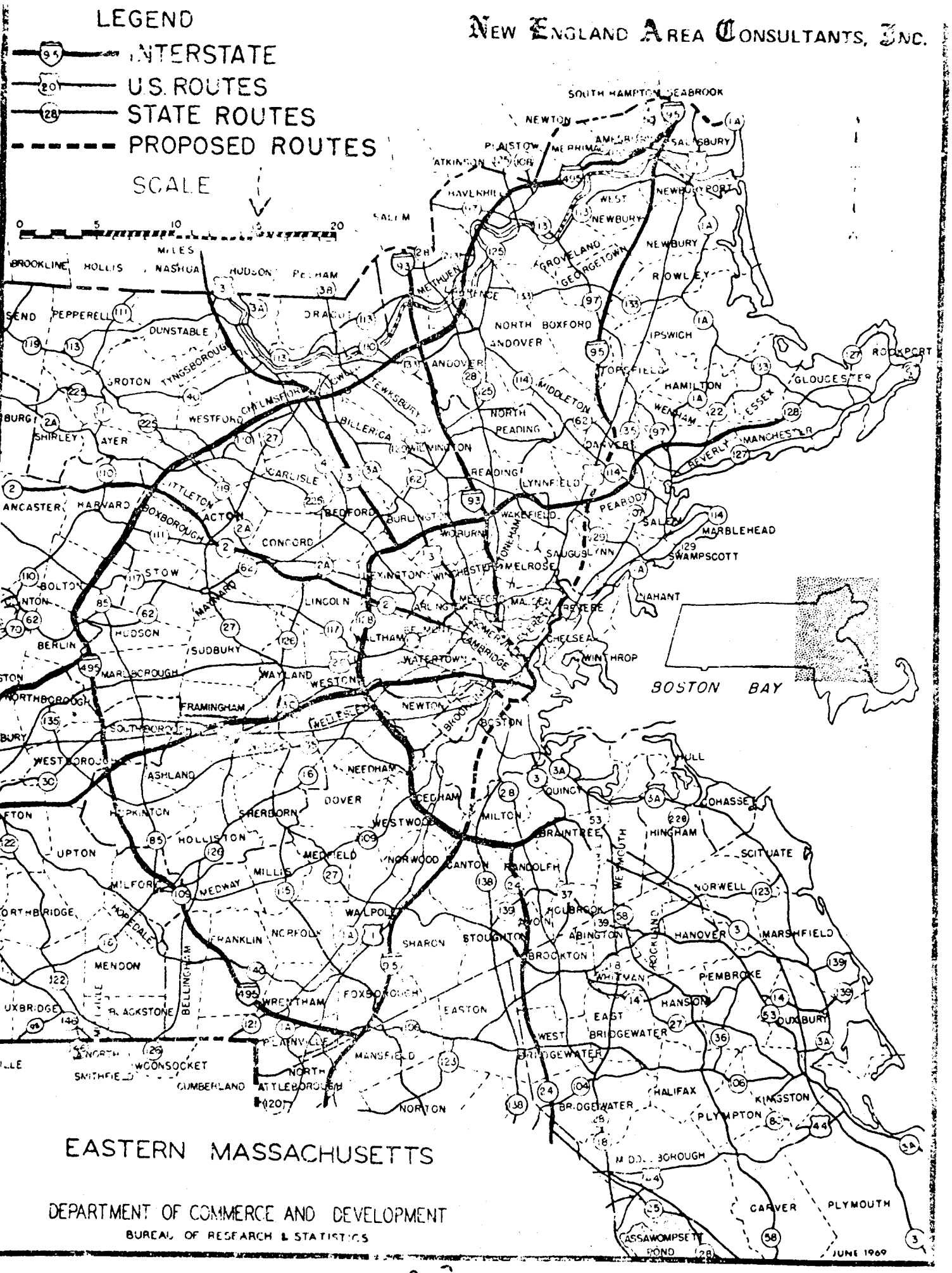
Ronald M. Cardamone
Software Instructor
Digital Equipment Corporation

TABLE OF CONTENTS

PART ONE

- COURSE INFORMATION -


POWDER MILL
RT. 62
RT. 27
FRENCH REST.
RT. 27 to 62
RUSSO'S
SUMMER ST.
SPA
TUTTO'S
MARION'S
NASON ST.
HILLSIDE ST.
MUNICIPAL PARKING
COPPER KETTLE REST.
DEC PARKING
RT. 27
WALNUT ST.
THOMPSON ST.
MISTER TAKEOUT
RT. 62
PIZZA PLACE
Village Deli
DEC 9 - 1007 (9-70)
Rt. 117
Rt. 117
Lower Village House
Rt. 27
Training


LEGEND
INTERSTATE
U.S. ROUTES
STATE ROUTES
PROPOSED ROUTES
SCALE
0 5 10 15 20
MILES
NEW ENGLAND AREA CONSULTANTS, INC.
BOSTON BAY
EASTERN MASSACHUSETTS
DEPARTMENT OF COMMERCE AND DEVELOPMENT
BUREAU OF RESEARCH & STATISTICS
JUNE 1969

OS/8 SOFTWARE (STANDARD)

The course is designed to teach students to use the OS/8 software system and to write, run, and debug assembly language programs.

LENGTH: 10 Days

PREREQUISITES: The student must be familiar with the PDP-8 instruction set and addressing scheme, as well as fundamental programming techniques such as looping, tally operations, address modification, and mathematical operations. Formal training in these areas can be obtained from the INTRODUCTORY PROGRAMMING ON THE PDP-8 course.

CONTENT: This course is designed to familiarize the student with the library of programs available with the system: PAL8 (8K assembler), OS/8 EDIT (Source editor), OS/8 ODT (debugging routine), SABR (relocatable binary assembler), PIP (peripheral interchange program), BUILD (system modification and generation program), CREF (cross reference program), EPIC (edit, punch, and compare), BITMAP (core usage), SRCCOM (source compare), and the floating point packages. The following topics are also covered: Monitor, Command Decoder, User Service Routines, System Tables, and Device Handlers. A portion of the course time is allotted to supervised laboratory sessions.

OS/8 SOFTWARE (ACCELERATED)

The OS/8 Software course (Accelerated) is designed to familiarize the advanced student with the usage and functions on the OS/8 internals.

LENGTH: 5 Days

PREREQUISITES: The student must be thoroughly familiar with fundamental programming techniques, the PDP-8 instruction set (to include the addressing scheme), and the PDP-8 PAL III assembler, editor, debugging routine, and I/O programming format. Those not meeting these prerequisites are advised to attend the OS/8 SOFTWARE (STANDARD) course rather than this course.

CONTENT: The course covers the structure of the OS/8 system including the monitor and files. The student will be shown how to write a program which utilizes the OS/8's User Service Routines and how to modify a device handler. In addition, the course contains a discussion of the system's tables and pertinent locations used by the system. A portion of the course time is allotted to supervised laboratory sessions.

OS/8 COURSE OUTLINE (STANDARD)

WEEK 1

MONDAY

I. INTRODUCTION

II. REVIEW OF PDP/8
   A. COMPUTER ORGANIZATION
   B. INSTRUCTION SET
   C. EXTENDED MEMORY ADDRESSING

III. OS/8 CONFIGURATION
   A. HARDWARE
      1. MINIMUM CONFIGURATION
   B. SOFTWARE
      1. BOOTSTRAPS
      2. DEVICE HANDLERS
      3. KEYBOARD MONITOR
      4. COMMAND DECODER
      5. SYSTEM PROGRAMS

IV. KEYBOARD COMMANDS
   A. DATE, ASSIGN, DEASSIGN
      R, RUN, START

V. HOMEWORK ASSIGNMENT

ASSOCIATED READING

"INTRO TO PROGRAMMING"
CHAP. 1,2,3,4,6

"OS/8 HANDBOOK"
CHAP. 1

TUESDAY

I. HOMEWORK REVIEW

II. CONCISE COMMAND LANGUAGE
   A. THEORY OF OPERATION
   B. CALLING & USING CCL

III. EDITOR
   A. CALLING EDIT
   B. MODES OF OPERATION

IV. PERIPHERAL INTERCHANGE PROG.
   A. CALLING PIP
   B. OPTIONS

V. DIRECT
   A. CALLING DIRECT
   B. OPTIONS

VI. LABORATORY SESSION

"OS/8 HANDBOOK"
CHAP. 1
CHAP. 2

WEDNESDAY

I. PAL8 ASSEMBLER	"OS/8 HANDBOOK"
A. CALLING PAL8	CHAP. 3
B. SYMBOLS	CHAP. 1
C. PSEUDO-OPS	CHAP. 2
D. OPTIONS	
II. CROSS REFERENCE PROG.	
A. CALLING & USING CREF	
III. ABSOLUTE LOADER	
A. CALLING ABSLDR	
B. OPTIONS	
IV. BITMAP	
A. CALLING & USING BITMAP	
V. HOMEWORK ASSIGNMENT	
VI. LABORATORY SESSION	

THURSDAY

I. HOMEWORK REVIEW	"OS/8 HANDBOOK" CHAP. 1
II. KEYBOARD COMMANDS (CONT)	
A. SAVE, GET	"SOFTWARE SUPPORT MANUAL" -APPENDIX A
III. FILE STRUCTURES	
A. FILE FORMATS	
B. FILE DIRECTORIES	
IV. LABORATORY SESSION	

FRIDAY

I. OCTAL DEBUGGING TECHNIQUE	"OS/8 HANDBOOK"
A. CALLING & USING ODT	CHAP. 1
II. FILE ORIENTED TRANSFER PROGRAM	CHAP. 2
A. CALLING & USING FOTP	
III. QUIZ 1	
IV. LABORATORY SESSION	

WEEK 2

MONDAY	ASSOCIATED READING
I. BUILD A. CALLING BUILD B. THEORY OF OPERATION C. COMMANDS	"OS/8 HANDBOOK" CHAP. 1 CHAP. 2
II. SYSTEM LAYOUT A. LAYOUT OF SYSTEM DEVICE B. CORE RESIDENCY C. SYSTEM TABLES	"SOFTWARE SUPPORT MANUAL" -APPENDIX B
III. RESORC A. CALLING & USING RESORC	
IV. LABORATORY SESSION	

TUESDAY	
I. MONITOR SERVICES A. CALLING THE USR B. USR FUNCTIONS	"SOFTWARE SUPPORT MANUAL" CHAP. 2,4
II. USING DEVICE HANDLERS	
III. HOMEWORK ASSIGNMENT	
IV. LABORATORY SESSION	

WEDNESDAY	
I. MONITOR SERVICES (CONT) A. DECODE B. CHAIN	"SOFTWARE SUPPORT MANUAL" CHAP. 2 "OS/8 HANDBOOK" CHAP. 1
II. BUILD DEVICE HANDLER FORMAT	
III. HOMEWORK ASSIGNMENT	
IV. LABORATORY SESSION	

THURSDAY	
I. SABRE ASSEMBLER	"OS/8 HANDBOOK" CHAP. 4
II. LINKING LOADER A. CALLING & USING LOADER B. OPTIONS	
III. LIBRARY SETUP A. USING LIBSET B. SYSTEM LIBRARY (LIB8)	

THURSDAY (CONT)

IV. FORTRAN II
   A. CALLING & USING FORT
   B. OPTIONS
   C. CALLING PAL8 SUBROUTINES

V. FINAL QUIZ

VI. LABORATORY SESSION

FRIDAY

I. HOMEWORK REVIEW

"OS/8 HANDBOOK"
CHAP. 2

II. UTILITY PROGRAMS
   A. BOOT
   B. SRCOM
   C. EPIC
   D. MCPIP
   E. CAMP

III. QUIZ REVIEW

IV. LABORATORY SESSION (OPTIONAL)

OS/8 COURSE OUTLINE (ACCELERATED)

MONDAY	ASSOCIATED READING
I. INTRODUCTION	"OS/8 HANDBOOK" CHAP. 1
II. OS/8 CONFIGURATION	
A. HARDWARE	CHAP. 2
1. MINIMUM CONFIGURATION	
B. SOFTWARE	
1. BOOTSTRAPS	
2. DEVICE HANDLERS	
3. KEYBOARD MONITOR	
4. COMMAND DECODER	
5. SYSTEM PROGRAMS	
III. KEYBOARD COMMANDS	
IV. FILE STRUCTURES	
A. FILE FORMATS	
B. DIRECTORIES	
V. CONCISE COMMAND LANGUAGE	
A. THEORY OF OPERATION	
B. CALLING & USING CCL	
VI. PERIPHERAL INTERCHANGE PROGRAM	
A. CALLING PIP	
B. OPTIONS	

TUESDAY	
I. EDITOR	"OS/8 HANDBOOK"
A. CALLING EDIT	
B. MODES OF OPERATION	CHAP. 1
II. PAL8 ASSEMBLER	
A. CALLING PAL8	
B. SYMBOLS	
C. PSEUDO-OPS	
D. OPTIONS	
III. ABSOLUTE LOADER	
A. CALLING ABSLDR	
B. OPTIONS	
IV. OCTAL DEBUGGING TECHNIQUE	
A. CALLING & USING ODT	
V. HOMEWORK ASSIGNMENT	
VI. LABORATORY SESSION	

WEDNESDAY

I. MONITOR SERVICES
   A. CALLING THE USR
   B. USR FUNCTIONS

"SOFTWARE SUPPORT MANUAL"
CHAP. 2,4

II. USING DEVICE HANDLERS

III. HOMEWORK ASSIGNMENT

IV. LABORATORY SESSION

THURSDAY

I. MONITOR SERVICES (CONT)
   A. DECODE
   B. CHAIN

"OS/8 HANDBOOK"
CHAP. 1

II. BUILD
   A. CALLING BUILD
   B. THEORY OF OPERATION
   C. COMMANDS

"SOFTWARE SUPPORT MANUAL"
CHAP. 2
-APPENDIX B

III. SYSTEM LAYOUT
   A. LAYOUT OF SYSTEM DEVICE
   B. CORE RESIDENCY
   C. SYSTEM TABLES

IV. HOMEWORK ASSIGNMENT

V. FINAL QUIZ

VI. LABORATORY SESSION

FRIDAY

I. HOMEWORK REVIEW

"OS/8 HANDBOOK"
CHAP. 4

II. FORTRAN II
   A. OPTIONS

III. LINKING LOADER
   A. OPTIONS

IV. LIBRARY SETUP
   A. USING LIBSET
   B. SYSTEM LIBRARY (LIB8)

V. QUIZ REVIEW

VI. LABORATORY SESSION (OPTIONAL)

PART TWO

- NOTES -

OPERATING SYSTEM/8

A. DEF. - An integrated collection of routines for supervising the sequencing of programs by a computer; eg. debugging, input/output, compilation and storage assignment.

B. TWO SEGMENTS

1. Program which controls operation of the computer and handles all I/O devices

   ⟹ called the MONITOR

2. Series of often used programs such as the EDITOR, ASSEMBLERS, LOADERS, etc..

   ⟹ called SYSTEM PROGRAMS

C. MINIMUM CONFIGURATION

PDP8/F with 8K
TTY
TD8/E DECTAPE (DUAL RECOMMENDED)
MR8EC ROM (256 WORD READ-ONLY MEMORY)

*Monitor requires 15K of peripheral storage and full complement of sys progs requires 32K.

⟹ SYSTEM DEVICE of at least 64K is required.

OS/8 - TWO PARTS

1. MONITOR

   A. CORE RESIDENT PORTION

      -section of program which deals with all essential operations has to reside in core permanently.

      *in OS/8 core resident portion is 256 words.

   KEYBOARD LISTENER

      -actual routine which listens for and interprets commands from the TTY.

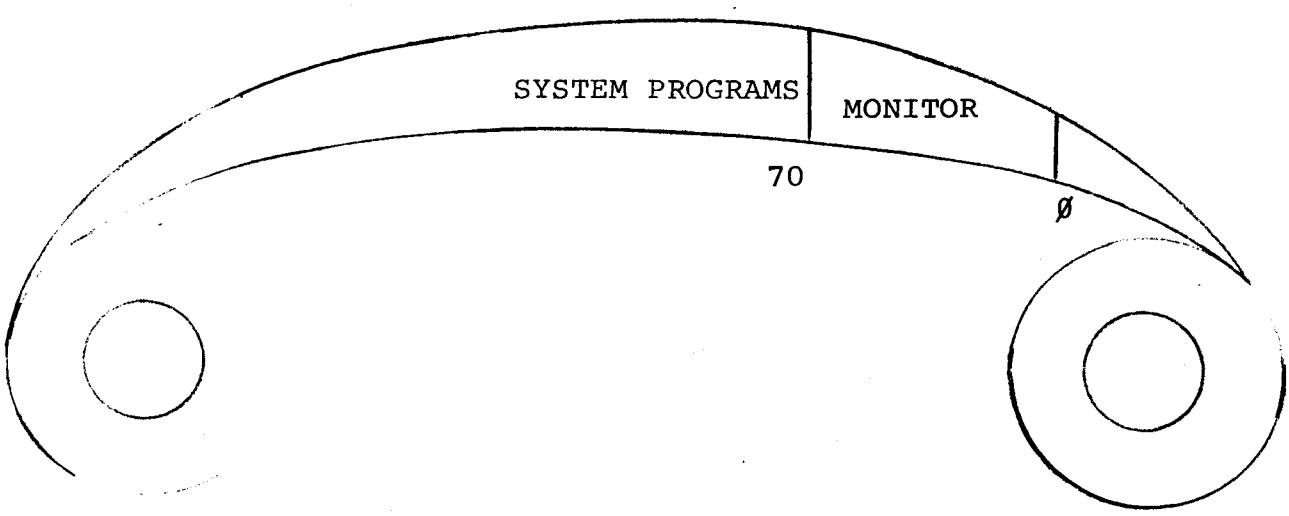
   C. USER SERVICE ROUTINE (USR)

      -performs all device independent input and output coordination.

2. SYSTEM PROGRAMS

   A. ASSEMBLERS
   B. EDITORS
   C. LOADERS
   D. COMPILER
   E. MORE UTILITY PROGRAMS

WHAT WE HAVE!!



SYSTEM DEVICE

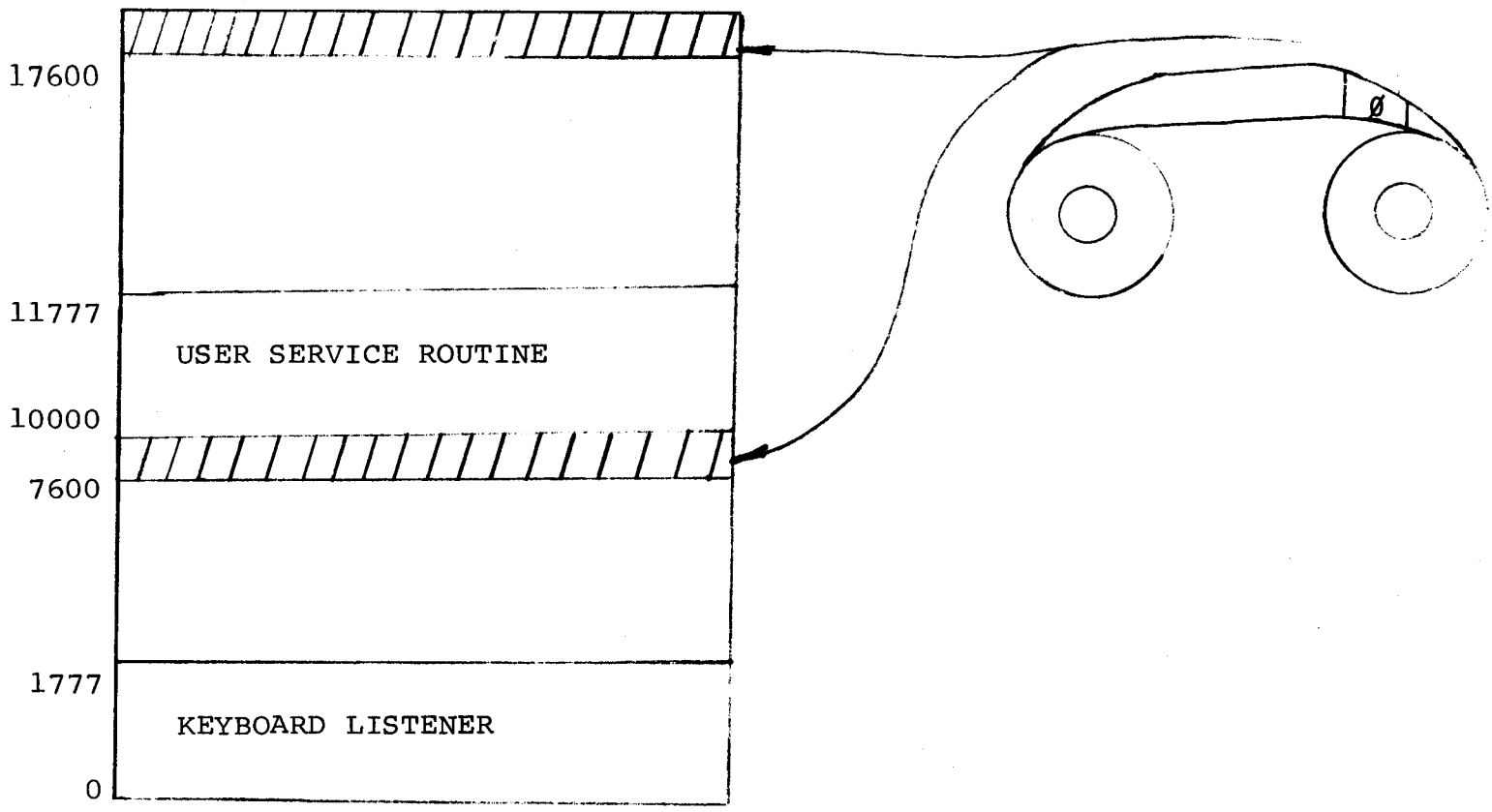
A. BLOCKS 0-67
   MONITOR

B. BLOCKS 70-
   SYSTEM PROGRAMS
   USER PROGRAMS

TAPE MOUNTED

Bootstrapping process is process of moving core resident portion (BLOCK Ø) into core (top page of Field Ø and 1)

*We need a program to do this
-TWO WAYS
1. MANUALLY TOGGLE
2. HARDWARE BOOTSTRAP
   (FLICK SW)



SUMMARY

What we have done is load Block 0. Our Bootstrap transferred control to the resident portion which in turn loaded the Keyboard Listener (it will use another routine - USR - to help out).

⟹ KYBD LISTENER THEN RESPONDS WITH A (.)

*USER PROGRAMS CAN EXIT TO KYBD LISTENER BY:

JMP 7600 OF FIELD 0
(saves 0-1777)

OR

JMP 7605 OF FIELD 0
(doesn't save)

OS/8 KEYBOARD LISTENER COMMANDS

The Keyboard Listener is the section of software that understands nine basic commands.

- RUN - get the core image file with the name specified from the device specified, load it into core, and start execution. For example:

  ```
  RUN     DTA5     PAL8

  RUN     SYS      FORT
  ```

- GET - This is the same as RUN, except that execution is not started. This is especially useful when ODT is being used.

- R - This is the same as RUN, except that the systems device (SYS) is assumed.

- SAVE - This command will save the program currently in core as a core image file on the device specified with the name specified. For example:

  ```
  SAVE    DTA1     DEMO

  SAVE    SYS      TEST
  ```

- START - This command will start execution of the program currently in core.

- DATE - This command will save the date in the monitor. The date is then available to all programs.

- ASSIGN
  DEASSIGN - These commands allow the user to substitute device names in the system, and are extremely useful when programs written for one configuration are run on another.

- ODT - This command pre-sets the system for running ODT by setting up swapping parameters, etc.

These are the keyboard commands that allow the user run, save, and load programs as well as vary the I/O device structure of an OS/8 system.

A. COMMAND DECODER

1. DEF. - a routine which accepts a command line containing the files to be used as input, destination of output, and options.

2. COMMAND STRING FORMAT

*DEVICE:OUTPUT<DEVICE:INPUT/OPTIONS

FROM 0 to 3 FROM 0 to 9

B. PERMANENT DEVICE NAMES

SYS - SYSTEM DEVICE
DSK - DEFAULT DEVICE (does not stand for DISK!!)
DTA(N)- DECTAPE 0≤N≤7
LTA(N)- LINCTAPE 0≤N≤7
TTY - TERMINAL
LPT - LINE PRINTER
CDR - CARD READER
PTP - PAPER TAPE PUNCH
PTR - PAPER TAPE READER
RKA(N)- RK8E DISK

C. SPECIAL CHARACTERS

1. CR (↵) - RETURN KEY
-line typed is processed

2. LF (↓)
-causes echo of input line

3. RUBOUT (\)
-deletes last character typed

4. ALTMODE ($)
-same as CR

5. CNTRL/U (↑U)
-deletes an entire line

6. CNTRL/C (↑C)
-return to KEYBOARD LISTENER

CONCISE COMMAND LANGUAGE

A. DEF. - a routine (CCL overlay) which makes it easier for the operator to enter certain commands.

*TO ENABLE CCL TYPE .R CCL

B. COMMAND STRING FORMAT

.COMMAND OUTPUT < INPUT

C. CCL COMMANDS (only those covered in this course are included)

.CREATE FILE-SPECIFICATION
-chains to EDIT and opens an output file with the name specified

.EDIT FILE-SPECIFICATION
-chains to EDIT

.COMPILE FILE SPECIFICATION
-chains to one of the OS/8 compilers or assemblers

.PAL FILE-SPECIFICATION
-similar to compile except PAL8 is always chained to

.EXECUTE FILE-SPECIFICATION
-similar to compile with the addition that the binary produced is loaded and started

.LOAD FILE-SPECIFICATION
-chains to the appropriate loader depending on the extension of the first input file

.CREF FILE-SPECIFICATION
-chains to PAL8 including the /C option which will cause it to chain to CREF

.MAP FILE-SPECIFICATION
-chains to BITMAP

.PUNCH FILE-SPECIFICATION
-chains to PIP. If no output is specified PTP is assumed

.ZERO DEVICE
-chains to PIP and PIP zeroes the device

.SQUISH FILE-SPECIFICATION
-chains to PIP

.HELP FILE-SPECIFICATION
-chains to PIP

.COPY FILE-SPECIFICATION
  -chains to FOTP

.DELETE FILE-SPECIFICATION
  -chains to FOTP

.LIST FILE-SPECIFICATION
  -chains to FOTP

.RENAME FILE-SPECIFICATION
  -chains to FOTP

.TYPE FILE-SPECIFICATION
  -chains to FOTP

.BOOT/DV
  -chains to BOOT

.DIRECT FILE-SPECIFICATION
  -chains to DIRECT and lists directories

.RES
  -chains to RESORC

.CCL
  -disables CCL

.CORE
  -types out on the TTY how much core is on the computer and how much is available to OS/8

.DATE
  -types current date. If SYS:DATE.SV exists it is chained to

.DEA
  -same as MONITOR'S DEASSIGN

.VER
  -prints the version no. of both the OS/8 MONITOR and of CCL

PAL8 ASSEMBLER

1. CALLING PAL8

```
.R PAL8
*DEV:BINARY,DEV:LISTING < DEV:SOURCE
```

2. CONTROL CHARACTERS

COMMA	(,)
EQUAL SIGN	(=)
ASTERISK	(*)
POINT	(.)
PLUS	(+)
MINUS	(-)
DOLLAR SIGN	($)
CR	(↵)
SEMI-COLON	(;)
SLASH	(/)
LEFT PARENTHESIS	(( )
LEFT BRACKET	([ )

3. PSEUDO-OPS

```
        DECIMAL
        OCTAL
        PAUSE
        FIELD N   0 ≤ N ≤ 7
        EXPUNGE
        FIXTAB
        FIXMRI
        TEXT
        DEVICE
        FILENAME
        XLIST
        EJECT
        ZBLOCK  N
        PAGE
IFDEF   SYM  <  SOURCE CODE >
IFNDEF  SYM  <  SOURCE CODE >
IFZERO  EXP  <  SOURCE CODE >
IFNZRO  EXP  <  SOURCE CODE >
```

SYMBOLIC EDITOR

1. CALLING EDIT

```
.R EDIT
*DEV:OUTPUT < DEV:INPUT
```

2. SPECIAL CHARACTERS

CR	(↵)
CNTRL/U	(↑U)
RUBOUT	(\)
CNTRL/L	
CNTRL/TAB	
POINT	(.)
SLASH	(/)
EQUAL	(=)
LINE FEED	(↓)
LEFT ANGLE BRACKET	(<)
CNTRL/O	(↑O)
CNTRL/C	(↑C)

3. COMMANDS

A. INPUT

#A
#R

B. OUTPUT

#L	#Y
#V	#E
#B	#Q
#P	#G
#N	##

C. EDITING

#D
#C
#I
#M
#K

D. SEARCHES

#S	#F
#$STRING'	#"
#J	

PERIPHERAL INTERCHANGE PROGRAM (PIP)

1. CALLING PIP

```
.R PIP
*DEV:OUTPUT < DEV:INPUT/OPTIONS
```

2. OPTIONS

   A. MERGING & FILE XFERS

      /A
      /B
      /I

   B. DELETING

      /D

   C. LISTING DIRECTORIES

      /E
      /F
      /L

   D. COMPRESS OR ZERO DIRECTORIES

      /Z /O
      /S
      /Y

NOTE: COPY SYSTEM TAPE?

⟹ (YZ)
  /S

OCTAL DEBUGGING TECHNIQUE

1. CALLING ODT

   .ODT - Not considered a system program.

2. COMMANDS

SLASH	(/)
CR	(↵)
LF	(↓)
SHIFT/N	(↑)
SHIFT/O	(←)
B	
G	
A	
L	
C	
F	
D	
M	
W	

BUILD

1. CALLING BUILD

```
.R BUILD
$ <------------ BUILD RESPONDS AND AWAITS YOUR COMMANDS
```

2. COMMANDS

```
$PRINT
$QLIST
$LOAD DEV:FILE.BN
$INSERT GROUP NAME:PERMANENT NAME
$DELETE ACTIVE NAME
$REPLACE ACTIVE NAME=GROUP NAME: PERMANENT NAME
$UNLOAD GROUP NAME
$DSK=ACTIVE NAME
$ALTER GROUP NAME, LOC=NEW VALUE
$EXAMINE GROUP NAME, LOC
$DCB ACTIVE NAME
$CTL ACTIVE NAME
$CORE N
$NAME ACTIVE NAME=NEW NAME
$VERSION
$SYSTEM
$BOOT
```

THE USER SERVICE ROUTINE (USR)

The User Service Routine is the second section of software that is swapped into core as required, and does all the device independent input and output coordination.

There are five basic operations the USR will perform, plus several utility type functions.

- FETCH HANDLER - Take the four character name I give you, figure out if that handler is in core already. If it is, give me the entry point; if it is not, load it into core and give me the entry point.

- LOOKUP FILE (A file that already exists) - Take the six character name I give you and find out where on the specified device it is. Returns two numbers; where it starts and how long it is so I may process that line.

- ENTER OUTPUT - Take the six character name I give you and open an output file on the device specified. Return two numbers to me: where it will start, and how much room is available.

- CLOSE OUTPUT - Take that file I opened earlier, and close it to all additional output using the length I give you.

- CHAIN - Take the name I give you, and find out where on SYS it is, get it, load it into core, and start execution.

There are several utility type functions in the USR, such as signal user error, reset tables, lock USR in core, dismiss USR, command decode, etc.. that are also useful.

These are the functions that the USR may perform for any program and these are functions used by the systems programs.

USR FUNCTIONS

USRIN:LOCK USR IN CORE

WHY? SAVES EXCESS SWAPPING

```
CDF N
CIF 10
JMS I (7700
10                    /USRIN=10
```

*NOTE - If current job status word BIT 11 is not set ⟹ 10000 - 11777 is saved first.

RESET:CLEAR ALL ENTRIES IN DEVICE HANDLER RESIDENCY TABLE.

WHY? IF YOU WANT A HANDLER BUT IT IS ALREADY IN CORE, IT MIGHT NOT BE WHERE YOU WANT IT.

```
CDF N
CIF 10
JMS I (200
13                    /RESET=13
0                     /PRESERVES  TENTATIVE FILES
```

FETCH: LOADS A HANDLER INTO CORE WHERE YOU SPECIFY.

WHY? HANDLER MUST BE IN CORE IN ORDER TO DO I/O OPERATIONS.

```
                    CLA
                    CDF N                    DEV NO., RETURNED HERE
                    CIF 10
PSEUDO-OP           JMS I (200
GENERATES           1                        /FETCH=1
TWO WORDS ------>   DEVICE DTA3

ENTRY,              6001                     /LOAD IN PAGE 30
                    JMP ERR                  /2 PAGES
                    NORMAL RETURN
ENTRY POINT <-------
RET. HERE                .
                         .
                         .
```

*NOTE: HANDLERS ARE ALWAYS LOADED IN FIELD 0!

LOOKUP:FINDS THE STARTING BLOCK NO. AND LENGTH OF A FILE.

```
                TAD ENTRY-1          /GET DEV. NO. IN AC
                CDF N
                CIF 10               FROM FETCH
                JMS I (200
                2                    / LOOKUP=2
        START,  NAME1
        LNGTH,  0                    STARTING BLOCK OF FILE RET.
                JMP ERR
                NORMAL RETURN
NEGATIVE                .
FILE LENGHT             .
RETURNED                .

        NAME1,  FILENAME PROG.EX
```

*NOTE: VALID EVEN FOR NON FILE STRUCTURED DEVICE. ALTHOUGH IT MUST BE READABLE.

NOW ENTER THE HANDLER

```
        CDF N
        CIF 0
        JMS I ENTRY        <-- TAG FROM FETCH
        ARG 1              /FUNCTION WORD
        ARG 2              /BUFFER ADDRESS
BLOCK,  ARG 3              /STARTING BLOCK
        JMP ERR
        NORMAL RETURN      <-- GOTTEN FROM LOOKUP
```

FUNCTION CONTROL WORD

0	1	2	3	4	5	6	7	8	9	10	11

READ=0
WRITE=1

NO. OF PAGES
TO TRANSFER

FIELD

ENTER:ENTERS A TENTATIVE FILE IN THE DIRECTORY.

```
                TAD TAG          /DEV. NO. IN AC8-11
                CDF N
STARTING        CIF 10
BLOCK OF        JMS I (200
HOLE RET.       3                 /ENTER=3
        STBLK,  NAME             /TAG OF FILE NAME
                0
                JMP ERR                  NEGATIVE LENGTH OF
                NORMAL RETURN            HOLE RETURNED
```

*NOTE: YOU CAN ENTER A NON-FILE STRUCTURED DEVICE. ALTHOUGH IT MUST NOT BE READ ONLY.

⟹ IF NO ROOM OR A TENTATIVE FILE ALREADY EXISTS ERROR RETURN.

AFTER OUTPUT IS COMPLETED

CLOSE: MAKES A TENTATIVE FILE PERMANENT AND DELETES ANY FILE WITH SAME NAME.

```
        TAD X                  /DEV. NO. IN AC8-11
        CDF N
        CIF 10
        JMS I (200
        4                      /CLOSE=4
        NAME                   /TAG OF FILE NAME
LENTH,  ARG1                   /FILE LENGTH
        JMP ERR
        NORMAL RETURN
```

YOU MUST SET THIS

*NOTE: YOU CAN CLOSE A NON-FILE STRUCTURED DEVICE. ALTHOUGH IT MUST NOT BE READ ONLY.

OTHER USR FUNCTIONS

ERROR: INDICATES AN ERROR RETURN WAS TAKEN. CONTROL BACK TO MONITOR.

```
ERR,    CDF N
        CIF 10
        JMS I (200
        7                    /ERROR=7
ADDR,   N                    /0 ≤ n ≤ 11
```

'USER ERROR N AT ADDR'

USROUT: RESTORES 10000-11777

```
        CDF N
        CIF 10
        JMS I (200
        11                   /USROUT=11
```

DECODE: CALLS THE COMMAND DECODER.

THE COMMAND DECODER:

-VALIDATES THE COMMAND LINE FOR ACCURACY
-PERFORMS A LOOKUP ON ALL INPUT FILES
-SETS UP VARIOUS TABLES FOR CALLING PROG.

```
CDF N
CIF 10
JMS I (200
5                          /DECODE=5
2001                       /ASSUME.PA
0                          /PRESERVE TENTATIVE FILES
NORMAL RETURN
```

*NOTE: DECODE PERFORMS AN AUTOMATIC RESET.

CHAIN: PERMITS A PROGRAM TO LOAD AND START ANOTHER PROGRAM.

```
CDF N
CIF 10
JMS I (200
6                          /CHAIN=6
BLOCK                      /STARTING BLOCK
```

GOTTEN BY LOOKUP

*NOTE: -PROGRAM IS STARTED AT ITS ST.ADDR.+1
-PROGRAM CHAINED TO MUST BE CORE IMAGE ON SYSTEM DEVICE
-CHAIN PERFORMS AUTOMATIC

1. USROUT
2. RESET - DOESN'T DELETE TENTATIVE FILES!

PART THREE

- LABORATORY AIDS -

INITIAL STARTING PROCEDURE

1. Turn Computer ON and TTY to LINE

2. Set DECtape drive panel switches to:

   REMOTE

   WRITE LOCK

3. Mount the system DECtape on Unit Ø (appears as 8 on some units)

   OR

   Set unit select switch to Ø (or 8)

4. Wind tape about 10 feet

TC01/TC08 Users

5. Toggle in Bootstrap

   7613/6774
   7614/1222
   7615/6766
   7616/6771
   7617/5216
   7620/1223
   7621/5215
   7622/0600
   7623/0220

   7754/7577
   7755/7577

6. Load S. A. 7613 (Field Ø)
   Hit CLEAR CONT tape starts rocking and the TTY responds with a .

7. Set DTAØ to: WRITE ENABLE

8. GO!!!!!

TD8E Users

5A. Toggle in RIM loader

5B. Read in TD8E Bootstrap through the Paper Tape reader.

6A. Load S. A. 7300 (Field Ø)
   Hit CLEAR CONT tape starts rocking and the TTY responds with a .

7A. Set DTAØ to: WRITE ENABLE

8A. GO!!!!!

PRINT DIRECTORIES

1. .R PIP
   *TTY:< SYS:/E ↲

   -After the directory of the system device is printed on the terminal, the COMMAND DECODER is recalled and prints an asterisk.

   *TTY:< DTA1:/E ↲

   -The directory of DTA1 will be printed.

   NOTE: If DTA1 doesn't have an OS/8 directory you will have to command PIP to put an OS/8 directory on the tape.

   *DTA1:</Z ↲

   -BE SURE DTA1 is output device and NOT SYS!

2. After directory of DTA1 is listed return to Monitor by typing CNTRL/C

   *CNTRL/C

SOME COMMAND STRINGS

USING PIP

.R PIP

-PIP THEN ASKS FOR INPUT AND OUTPUT SPECIFICATIONS

1. FILE TRANSFERS

*DEV:OUTPUT.EX< DEV:INPUT.EX/A,B,OR I ↲

2. DIRECTORY LISTINGS

*TTY:< DEV:/E ↲

NOTE: OUTPUT FOR DIRECTORY LISTINGS DEFAULTS TO TTY AND INPUT DEFAULTS TO DSK. (THE SYSTEM DEFAULT DEVICE)

*TTY:< DSK:/E ↲

*< DSK:/E ↲

*/E ↲

-ALL OF ABOVE WILL GIVE A DIRECTORY LISTING OF DSK.

USING EDITOR

.R EDIT

-EDIT THAN ASKS FOR INPUT & OUTPUT SPECIFICATIONS

1. FILE TO BE CREATED

*DEV:OUTPUT.EX < ↲

#A ↲

-NO INPUT FILE IS SPECIFIED WHEN FILE IS TO BE CREATED AT KEYBOARD

2. FILE TO BE EDITED

*DEV:OUTPUT.EX< DEV:INPUT.EX ↲

#R ↲

USING PAL8

.R PAL8

-PAL8 THEN ASKS FOR INPUT AND OUTPUT SPECIFICATIONS

SOME COMMAND STRINGS (CONT.)

USING PAL8 (continued)

1. BINARY FILE AND LISTING DESIRED

   *DEV:BINARY.BN,TTY:< DEV:INPUT ↲

   OR

   *DEV:BINARY.BN,DEV:LISNG.LS< DEV:INPUT ↲

   NOTE: IF NO EXTENSIONS ARE GIVEN PAL8 APPENDS .BN TO BINARY FILE NAME AND .LS TO LISTING FILE NAME.

2. ONLY BINARY

   *DEV:BINARY.BN< DEV:INPUT ↲

3. ONLY LISTING

   *,TTY:< DEV:INPUT ↲

USING ABSLDR

.R ABSLDR

-ABSLDR THEN ASKS FOR INPUT SPECIFICATIONS

*DEV:INPUT $

NOTE: ABSLDR ASSUMES .BN EXTENSION. ALSO IF LINE IS TERMINATED WITH A CR, MORE INPUT IS EXPECTED. ALTMODE INDICATES END OF INPUT.

USR LABSHEET

STARTING PROCEDURE

1. BOOTSTRAP THE MONITOR

2. GIVE THE DATE COMMAND

3. EDIT AND ASSEMBLE THE PROG (XFER) ACCOMPANING THIS LABSHEET

   NOTE: REMEMBER TO PUT THE NAME OF ONE OF YOUR ASCII FILES IN LOC NAME1.

4. LOAD THE BINARY AND CALL ODT

   .R ABSLDR

   *DEV:PROG/9$

   .ODT

USING ODT

1. WHAT IS IN LOC 200?

2. WHAT IS IN LOC 200 OF FIELD 1? FIELD 2? (IF YOU HAVE 12K)

3. OPEN THE NEXT SEQUENTIAL LOCATION

4. OPEN LOC 210; CLOSE IT AND TREAT ITS CONTENTS AS A MEMORY REFERENCE AND OPEN IT (^ OR /)

   -CLOSE THAT LOC AND TREAT ITS CONTENTS AS ANOTHER LOC TO BE OPENED. (← OR _)

5. USING THE WORD SEARCH MECHANISM GET AN OCTAL DUMP OF ALL LOC WITHIN THE PROG WHICH CONTAIN A JMS INSTRUCTION.

   -GET AN OCTAL DUMP OF ALL LOCATIONS WHICH CONTAIN A CIF INSTRUCTION.

   -GET AN OCTAL DUMP OF XFER

6. INSERT A BREAKPOINT IN LOC 224. THEN START XFER (VIA 200G)

   -WHAT IS IN AC & LINK WHEN BP IS REACHED?

   -WHAT IS IN LOC 221? WHAT DOES THIS NO REPRESENT?

-WHAT IS IN LOC 222? WHAT DOES THIS NO REPRESENT?

7. INSERT A BREAKPOINT IN LOC 235 AND PROCEED FROM LAST BREAKPOINT.

   -WHAT IS IN AC & LINK WHEN BP IS REACHED?

   -FIND THE DEVICE NO OF TTY WHICH WAS RETURNED BY USR.

   -FIND THE TTY HANDLER ENTRY POINT WHICH WAS RETURNED BY USR.

8. INSERT A BREAKPOINT IN LOC 253 AND PROCEED FROM LAST BREAKPOINT.

   -WHAT IS IN AC & LINK WHEN BP IS REACHED?

   -WHAT DOES THE NO IN AC REPRESENT?

9. INSERT A BP IN LOC 310 AND PROCEED FROM LAST BREAKPOINT.

   -WHAT DID THE PROGRAM DO UP UNTIL THE BP IS REACHED.

   -WHAT IS IN AC WHEN BP IS REACHED? WHAT IS THIS NO USED FOR?

10. REMOVE THE BP AND PROCEED TO THE END OF XFER.

11. GIVE THE DEASSIGN COMMAND - .DE

<u>REASSIGN DEVICES</u>

1. USING THE ASSIGN COMMAND HAVE OUTPUT GO TO DTA1 INSTEAD OF TTY (OR LPT)

2. USING THE ASSIGN COMMAND HAVE INPUT COME FROM TTY AND OUTPUT GO TO THE LPT (IF A LPT IS AVAILABLE)

   NOTE: DON'T USE ODT AFTER GIVING THE ASSIGN COMMAND SINCE ODT USES THE USER DEVICE NAME TABLE FOR SCRATCH.

```
                /ASCII DUMP ON TTY
                /'XFER' TRANSFERS A FILE FROM DTA0 TO THE TERMINAL
                /FILE SHOULD BE ASCII
                /INPUT FILE NAME IS IN LOCATION NAME1
                /OUTPUT FILE NAME (TO INSURE DEVICE INDEPENDENCE)IS IN LOC NAME2
        0200            *200
        0001            FETCH=1
        0002            LOOKUP=2
        0003            ENTER=3
        0004            CLOSE=4
        0007            ERROR=7
        0010            USRIN=10
        0013            RESET=13

                /FIRST LOCK USR IN CORE

00200   7300            CLA CLL
00201   6201            CDF 0               /SET DF TO CURRENT FIELD
00202   6212            CIF 10              /SET IF TO FIELD OF USR
00203   4777            JMS I (7700
00204   0010            USRIN


                /RESET SYSTEM TABLES

00205   7300            CLA CLL
00206   6201            CDF 0
00207   6212            CIF 10
00210   4776            JMS I (200          /USR IS IN
30211   0013            RESET
00212   0000            0                   /PRESERVE TENTATIVE FILES

                /FETCH DTA0 HANDLER
00213   7300            CLA CLL
00214   6201            CDF 0
00215   6212            CIF 10
00216   4776            JMS I (200
00217   0001            FETCH
00220   0424            DEVICE DTA0         /USR REPLACES SECOND WORD WITH
00221   0160
                                            /DEVICE NO OF DTA0
00222   1001    ENTRY,  1001                /PUT HANDLER AT PAGE 4;ROOM FOR 2 PAGE HAN
                                            /REPLACED BY ENTRY POINT OF HANDLER!
00223   5335            JMP ERR             /ERROR RETURN--BYE,BYE

                /FETCH TTY HANDLER
00224   7300            CLA CLL
00225   6201            CDF 0
00226   6212            CIF 10
00227   4776            JMS I (200
00230   0001            FETCH
00231   2424            DEVICE TTY
00232   3100
00233   1401    TTYIN,  1401                /LOAD IN PAGE 6;ROOM FOR 2 PAGE HANDLER!
90234   5335            JMP ERR             /ERROR RETURN --BYE,BYE!
```

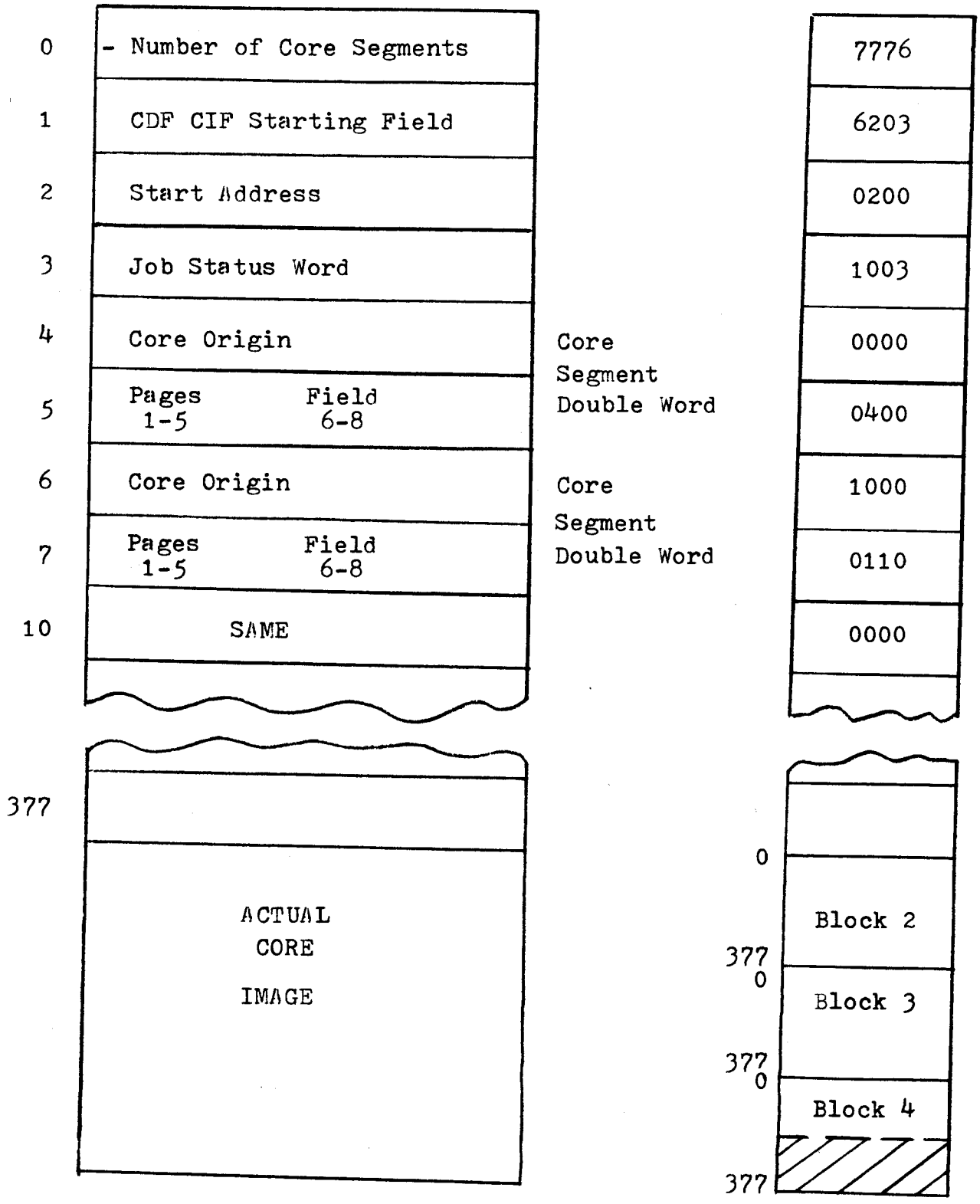
```
                /NOW LOOKUP INPUT FILE
0235  7300              CLA CLL
0236  1221              TAD ENTRY-1       /GET DEVICE NO IN THE AC
0237  6201              CDF 0
0240  6212              CIF 10
0241  4776              JMS I (200
0242  0002              LOOKUP
0243  0342      START,  NAME1             /POINTER TO INPUT FILE.USR REPLACES WITH
                                          /STARTING BLOCK OF FILE
0244  0000      LENGTH, 0                 /USR INSERTS NEGATIVE FILE LENGTH
0245  5335              JMP ERR           /OH!OH!

                /CALCULATE NO OF PAGES FOR READ FUNCTION CONTROL WORD
                /NOT A GOOD WAY TO DETERMINE BUFFER  SIZE FOR OBVIOUS REASONS!
0246  1244              TAD LENGTH
0247  7041              CIA
0250  3352              DCA TEMP
0251  1352              TAD TEMP
0252  1352              TAD TEMP
0253  7100              CLL
0254  7006              RTL;RTL;RTL
0255  7006
0256  7006
0257  3265              DCA BLKNO-2
0260  1243              TAD START
0261  3267              DCA BLKNO         /STARTING BLOCK OF FILE INTO HANDLER CALL

                /WE NOW ARE READY TO READ THE FILE!
0262  6201              CDF 0
0263  6202              CIF 0             /HANDLERS ALWAYS GO INTO FIELD 0
0264  4622              JMS I ENTRY
0265  0000              0                 /GETS FUNCTION CONTROL WORD
0266  2000              2000              /BUFFER ADDRESS
0267  0000      BLKNO,  0                 /GETS STARTING BLOCK OF FILE
0270  7700              SMA CLA           /ERROR RETURN
0271  7410              SKP
0272  5335              JMP ERR           /FATAL RETURN

                /OPENING OF OUTPUT FILE FOLLOWS.ONLY INCLUDED TO
                /INSURE DEVICE INDEPENDENCE
0273  7300              CLA CLL
0274  1232              TAD TTYIN-1       /DEV NO OF TTY IN AC
0275  6201              CDF 0
0276  6212              CIF 10
0277  4776              JMS I (200
0300  0003              ENTER
0301  0346      STBLK,  NAME2             /POINTER TO OUTPUT FILE NAME
                                          /REPLACED WITH STARTING BLOCK OF HOLE
0302  0000              0                 /REPLACED WITH SIZE OF HOLE
0303  5335              JMP ERR

                /SET UP FOR WRITE OPERATION
0304  1301              TAD STBLK
0305  3316              DCA BLK
```

```
00306  1265            TAD BLKNU-2
00307  1375            TAD (4000
00310  3314            DCA BLK-2)

                /WRITING OF NEW FILE FOLLOWS
00311  6201            CDF 0
00312  6202            CIF 0
00313  4633            JMS I TTYIN
00314  0000            0                  /FUNCTION CONTROL WORD
00315  2000            2000               /BUFFER ADDRESS
00316  0000  BLK,      0                  /STARTING BLOCK
00317  7700            SMA CLA            /ERROR RETURN
00320  7410            SKP
00321  5335            JMP ERR            /FATAL!

                /CLOSE OUTPUT FILE
00322  1352            TAD TEMP
00323  3332            DCA SIZE
00324  1232            TAD TTYIN-1        /DEV NO IN AC
00325  6201            CDF 0
00326  6212            CIF 10
00327  4776            JMS I (200
00330  0004            CLOSE
00331  0346            NAME2              /NEW NAME
00332  0000  SIZE,     0
00333  5335            JMP ERR            /BYE!BYE!
00334  5774            JMP 7605           /BACK TO MONITOR

                /ERROR SUBROUTINE--DON'T WANT TO BE HERE!!!!!!!
00335  6201  ERR,      CDF 0
00336  6212            CIF 10
00337  4776            JMS I (200
00340  0007            ERROR
00341  0002            2                  /PRINT 'USER ERROR 2.....'
00342  2201  NAME1,    FILENAME RAM
00343  1500
00344  0000
00345  0000
00346  0201  NAME2,    FILENAME BAM
00347  1500
00350  0000
00351  0000
00352  0000  TEMP,     0
00374  7605
00375  4000
00376  0200
00377  7700
                $$$$$$$$$$
```

PART FOUR

- ILLUSTRATIONS -

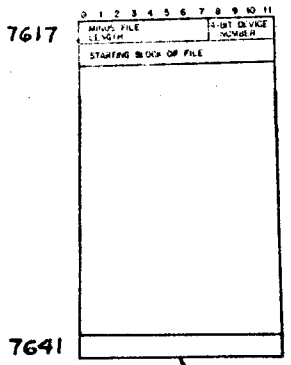
.SAVE SYS FILE 200-750,11100;200=1003



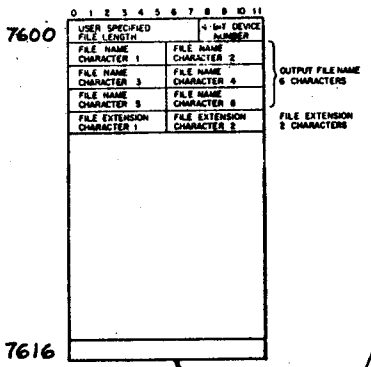
FIELD I TABLES

ALWAYS CORE RESIDENT

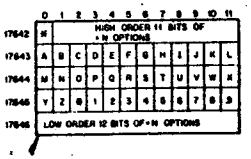
INPUT ENTRIES



OUTPUT ENTRIES



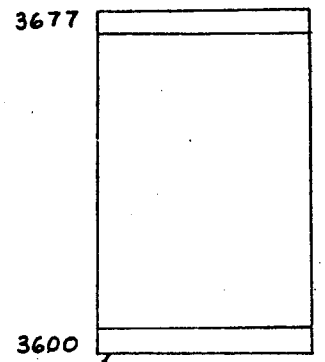
OPTION TABLE



SET BY COMMAND DECODER
(INPUT IS SET AFTER "LOOKUP")

ONLY RESIDENT WHEN PIP OR USR IS IN CORE

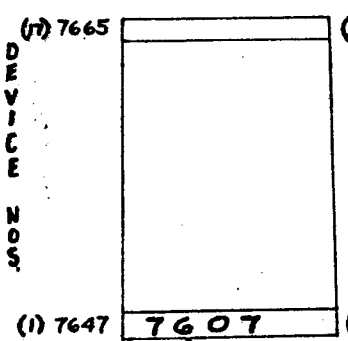
DEVICE LENGTH TABLE



RESIDES IN PIP
64 LOC FOR DEV TYPES 0-77
(CURRENTLY ONLY 0-20 ASSIGN.)
ONLY USED FOR /S OR /Z
FOR NEW DEV-PATCH WITH ODT

HANDLER RES. TABLE

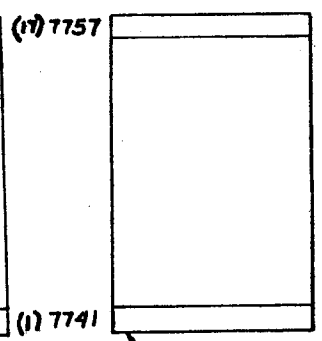
DEVICE NOS.



SYS ENTRY
(NOT ZEROED BY RESET)

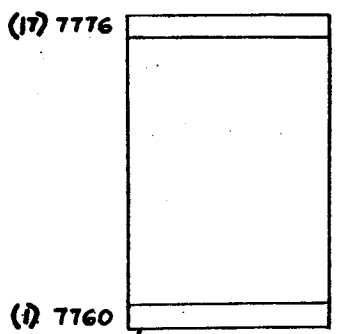
SET BY USR;
CLEARED BY GET, R, RUN, ST,
SAVE & RESET

USER DEV NAME TABLE



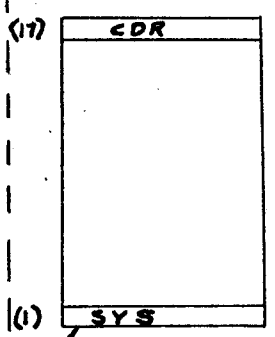
SET BY ASSIGN
& DEASSIGN;
CLEARED BY RESET

DEV CONTROL WD TABLE



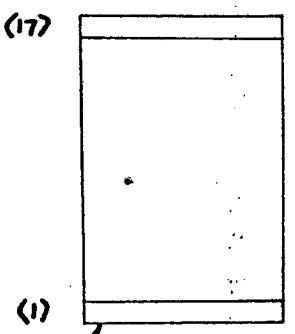
INDICATES DEV TYPE, FILE STRUCT,
READ ONLY, WRITE ONLY
BITS 9-11 CONT. THE DIRECTORY
BLOCK NO. OF CURRENT ACTIVE
TENATIVEFILE
BITS 9-11 ARE ZEROED BY RUN, R,
SAVE, GET, ST, DECODE & RESET

PERM DEVICE NAME



RESIDES IN USR
WHEN USR IS IN CORE
FIRST ADDR. IS IN 0036

HANDLER INFO TABLE



RESIDES IN USR
WHEN USR IS IN CORE
FIRST ADDR IS IN 0037
CLEARED BY RESET
INDICATES 2 PG HANDLER,
REL. BLOCK LOC ON SYS DEV,
REL. ADDR. OF HANDLER ENTRY

```
.R BUILD

$??

TCA8:   SYS
TC  :   DTA0   DTA1   DTA2   DTA3
TD8E:   SYS    DTA0   DTA1
TD8A:   DTA0   DTA1
KOM :   SYS    DTA0   DTA1
RK8E:   SYS    RKB0
RK05:   RKA0   RKB0   RKA1   RKB1
RK8 :   SYS    RKA1
RK01:   RKA0   RKA1
LINC:   SYS
LNC :   LTA0   LTA1   LTA2   LTA3
RF08:   SYS
KL8E:   TTY
KS32:   PTP    PTR
PT8E:   PTP    PTR
LPSV:   LPT
TA8A:   CSA0   CSA1
VR12:   TV
CR8E:   CDR
BAT :   BAT
$


$IN RK8E:SYS

$IN KL8E:TTY

$IN RK8E:RKB0

$IN PT8E:PTR,PTP

$IN TD8A:DTA0,DTA1

$IN LPSV:LPT

$DSK=SYS

$BO
WRITE ZERO DIRECT?YES
SYS BUILT

.SAVE SYS BUILD 0-7577,10000-17577,200=0
```

USING BUILD (CONT)

```
.R BUILD

$PR

TC08:  SYS
TC  :  DTA0  DTA1  DTA2  DTA3
TD8E:  SYS   DTA0  DTA1
TD8A: *DTA0 *DTA1
ROM :  SYS   DTA0  DTA1
RK8E: *SYS  *RKB0
RK05:  RKA0  RKB0  RKA1  RKB1
RK8 :  SYS   RKA1
RK01:  RKA0  RKA1
LINC:  SYS
LNC :  LTA0  LTA1  LTA2  LTA3
RF08:  SYS
KL8E: *TTY
KS33:  PTP   PTR
PT8E: *PTP  *PTR
LPSV: *LPT
TA8A:  CSA0  CSA1
VR12:  TV
CR8E:  CDR
BAT :  BAT
$
```

SYSTEM TABLES BEFORE USING BUILD

17647	/7607
17650	/7607
17651	/0000
17652	/0000
17653	/0000
17654	/0000
17655	/0000
17656	/0000
17657	/0000
17660	/0000
17661	/0000
17662	/0000
17663	/0000
17664	/0000
17665	/0000

Dev. Handler Residency

17760	/4160
17761	/4160
17762	/4160
17763	/4160
17764	/4160
17765	/4160
17766	/4050
17767	/0000
17770	/1020
17771	/2010
17772	/1040
17773	/0270
17774	/0270
17775	/0000
17776	/0000

Dev. Control Wd.

17741	/0000
17742	/0000
17743	/0000
17744	/0000
17745	/0000
17746	/0000
17747	/0000
17750	/0000
17751	/0000
17752	/0000
17753	/0000
17754	/0000
17755	/0000
17756	/0000
17757	/0000

User Dev. Name

13600	/0000
13601	/0000
13602	/0000
13603	/0000
13604	/0000
13605	/1520
13606	/6000
13607	/4000
13610	/2000
13611	/0001
13612	/7601
13613	/7401
13614	/7201
13615	/7001
13616	/6437
13617	/6437
13620	/0000
13621	/6437
13622	/0000
13623	/1520
13624	/0000

Device Length
(Resides in PIP)

10772	/0000
10773	/0000
10774	/0210
10775	/0211
10776	/0212
10777	/0213
11000	/0420
11001	/4776
11002	/1000
11003	/1112
11004	/1203
11005	/5407
11006	/5401
11007	/0000
11010	/0000

Dev. Handler INFO
(Resides in USR)

10564	/4631	SYS
10565	/5723	DSK
10566	/4604	DTAØ
10567	/4605	DAT1
10570	/4606	DTA2
10571	/4607	DTA3
10572	/6373	RKAØ
10573	/5524	TTY
10574	/4024	PTP
10575	/4224	PTR
10576	/4020	LPT
10577	/4503	CSAØ
10600	/4504	CSA1
10601	/0000	
10602	/0000	

Perm. Dev. Name
(Resides in USR)

SYSTEM TABLES AFTER USING BUILD

```
17647/7607
17650 /7607
17651 /0000
17652 /0000
17653 /7621
17654 /0000
17655 /0000
17656 /0000
17657 /0000
17660 /0000
17661 /0000
17662 /0000
17663 /0000
17664 /0000
17665 /0000
17666 /0000
```

Dev. Handler Residency

```
17741/0000
17742 /0000
17743 /0000
17744 /0000
17745 /0000
17746 /0000
17747 /0000
17750 /0000
17751 /0000
17752 /0000
17753 /0000
17754 /0000
17755 /0000
17756 /0000
17757 /0000
```

User Dev. Name

```
17760/4230
17761 /4230
17762 /4210
17763 /4210
17764 /4230
17765 /0000
17766 /1020
17767 /2010
17770 /1040
17771 /0000
17772 /0000
17773 /0000
17774 /0000
17775 /0000
17776 /0000
```

Dev. Control Word

```
10772/0000
10773 /0000
10774 /4210
10775 /4214
10776 /0000
10777 /4576
11000 /0600
11001 /0712
11002 /1003
11003 /0000
11004 /0000
11005 /0000
11006 /0000
11007 /0000
11010 /0000
```

Dev. Handler INFO
(Resides in USR)

```
10564/4631   SYS
10565 /5723  DSK
10566 /4604  DTA0
10567 /4605  DTA1
10570 /6473  RKBØ
10571 /5524  TTY
10572 /4024  PTP
10573 /4224  PTR
10574 /4020  LPT
10575 /0000
10576 /0000
10577 /0000
10600 /0000
10601 /0000
10602 /0000
```

Perm. Dev. Name
(Resides in USR)

```
                /HANDLER FORMAT EXAMPLE
                /THIS ROUTINE JUST PRINTS A '&'.  IT IS IN SUITABLE
                /FORMAT TO BE INSERTED INTO BUILD.

                /*****************************************************
                /BUILD DEVICE HANDLER FORMAT

                /                *0
                /        MINUS NO OF SEPARATE HANDLERS IN GROUP
                /        GROUP NAME
                /        PERMANENT NAME
                /        DEVICE CONTROL BLOCK
                /        ENTRY POINT OFFSET
                /        0
                /        0 UNLESS A SYSTEM HANDLER
                /                *200
                /        BODY OF HANDLER

                /*****************************************************

        0000            *0
00000   7777            7777                /1 HANDLER
00001   0405            DEVICE DEMO         /GROUP NAME
00002   1517
00003   0601            DEVICE FAKE         /PERMANENT NAME
00004   1305
00005   1430            1430                /WRITE ONLY,DEV #43
00006   0000            0000                /ENTRY POINT OFFSET-1 PAGE,NON SYS
00007   0000            ZBLOCK 2

        0200            *200
00200   0000  ENTRY,    0                   /INITIALLY POINTS TO FCW
00201   7200            CLA                 /IN CASE USER CALLED WITH NON-ZERO AC
00202   6214            RDF                 /CONSTRUCT A CIF CDF N
00203   1221            TAD CIFX
00204   3216            DCA EXIT
00205   1220            TAD CHAR
00206   6046            TLS
00207   6041            TSF
00210   5207            JMP .-1
00211   7200            CLA
00212   2200            ISZ ENTRY           /POINT TO BUFFER ADDRESS
00213   2200            ISZ ENTRY           /POINT TO STARTING BLOCK NO
00214   2200            ISZ ENTRY           /POINT TO ERROR RETURN
00215   2200            ISZ ENTRY           /POINT TO NORMAL RETURN
00216   0000  EXIT,     0                   /GETS CIF CDF N
00217   5600            JMP I ENTRY
00220   0246  CHAR,     "&
00221   6203  CIFX,     6203
              $$$$$$$$$$$$
```

PART FIVE

- ASSIGNMENTS -

HOMEWORK ASSIGNMENTS

1. HOW MANY LOCATIONS FROM 100-117 CONTAIN A NEGATIVE NO.? HALT WITH THE ANSWER IN THE AC.

2. MOVE A BLOCK OF DATA FROM A LOC IN CORE TO ANOTHER AS A SUBROUTINE.

```
MAIN PROG
    .
    .
    .
JMS MOVE
1000                /OLD ADDR
5000                /NEW ADDR
100                 /NO. OF WORDS TO BE MOVED
```

3. WRITE A PROGRAM USING CONDITIONAL ASSEMBLY. IF A IS ZERO THEN A MESSAGE IS PRINTED INDICATING IT TO BE ZERO; IF A IS NON-ZERO THEN VICE-VERSA.

4. WRITE A SUBROUTINE WHICH WILL PRINT ANY MESSAGE THAT FOLLOWS THE CALL TO THE SUB. THE MESSAGE IS ASSEMBLED USING THE TEXT PSEUDO-OP.

   NOTE: THIS SUB SHOULD BE CAPABLE OF BEING CALLED FROM ANY FIELD AND RETURNING TO CALLING PROGRAM.

5. WRITE A PROGRAM WHICH XFERS A FILE FROM DTAØ TO DTA1.

   NOTE: SET XFER BUFFER TO A FIXED LENGTH.

6. WRITE A PROGRAM WHICH XFERS A FILE FROM ONE DEVICE TO ANOTHER. THE DEVICES AND FILES WILL BE DETERMINED AT RUN TIME THRU A CALL TO THE COMMAND DECODER. IF /X OPTION IS USED HAVE YOUR PROGRAM CHAIN TO PROBLEM #3.

7. ASSUME A FIXED INTEREST RATE OF 5%. SHOW THE VALUE OF $100 AT THE END OF 1 YEAR ASSUMING THAT INTEREST IS ACCRUED 'X' TIMES PER YEAR.

8. DETERMINE THE NUMBER OF YEARS REQUIRED TO DOUBLE AN INITIAL INVESTMENT OF $100, AT AN INTEREST RATE OF 'X'%, WITH INTEREST BEING ACCRUED ONCE A YEAR.

OS/8 QUIZ 1

1. The subroutine below is intended to simulate an Inclusive Or between the two words following the call to the sub. The result is left in the AC. Supply the missing instructions to make it work properly.

```
.                          IOR,Ø
.                               TAD I IOR
.                               CMA
JMS IOR                         DCA TEMP
 A                              ISZ IOR
B                                  AA
NORMAL RETURN                   ________
                                   CMA
                                    BB
                                ________
                                    CC
                                ________
                                  ISZ IOR
                                  JMP I IOR

                           TEMP, O
```

a. AA = TAD TEMP, BB = ISZ IOR, CC = AND IIOR

b. AA = TAD I IOR, BB = AND TEMP, CC=CMA

c. AA = TAD I IOR, BB = TAD TEMP, CC=CMA

d. AA = TAD I IOR, BB = AND TEMP, CC=NOP

2. The following program is run:

```
CLA CLL
TAD ALPHA
CIF 10
DCA BETA
CDF 20
TAD ALPHA
DCA I GAMMA
HLT
```

Assuming the program was loaded and run in Field 0, what observation can be made?

a. There has to be an ALPHA in Field 0 and Field 1
b. BETA is in Field 1
c. Field 1 is never utilized
d. GAMMA is in Field 2.

3. If the following commands were typed to ODT which answer best describes the result?

215 / 1301 230B

a. ODT will type a "?" because B is a non-octal digit and location 215 will be closed

b. A Breakpoint will be inserted in loc. 230

c. 230 will be inserted in loc. 215 and B is ignored

d. None of the above

4. Giving the three EDITOR COMMANDS P ,_____, and R is equivalent to the single EDITOR COMMAND_____.

5. The user typed the following command string to the Editor and then typed the letter L. In your own words how will the Editor respond?

$ AGAIN ' "S (carriage RETURN)

6. What location will PAL8 assign to the symbol THING in the coding below?

```
*377
TAGI, 2500
DECIMAL
*TAGI-12
THING, 1436
```

a. The symbol THING is given the value of 363_8

b. The symbol THING is given the value of 365_{10}

c. The symbol THING is given the value 2488

d. The symbol THING is given the value 3936

7. The Editor command .-2,.+2L will cause what action?

a. The Editor will type four lines and the value of the "dot" is unchanged.
b. The Editor will respond with a question mark (?).
c. Five lines will be printed and the value of the "dot" will be updated.
d. Four lines are typed and the value of the "dot" is two greater than before.

8. The assembly statement A=B has what effect?

   a. Makes ASC11 code 301 and 302 equivalent.
   b. Places B into location A.
   c. Everytime B is referenced it is given the value assigned to A.
   d. Creates a symbol A and gives it the same numeric value as assigned to B.

9. How is the following coding interpreted (assembled) by PAL8?

```
BUFF1, BUFF1
```

   a. The value of the current location counter is assigned to the symbol BUFF1 and that value is also assembled into location BUFF1.

   b. The programmer should have used the parameter assignment statement, BUFF1=BUFF1, in this case.

   c. A location can't be filled with its own address since that value is not defined until the carriage return is given.

   d. The diagnostic message "DT" is given and assembly pass 1 is terminated.

10. There are two ways of giving a numeric value to a symbol. The symbol PLUS3 can be given a value of 315 by the coding PLUS3=315. Select the answer which correctly describes the second way.

   a. PLUS3, 315

   b. *314
      PLUS3, 0

   c. *315
      PLUS3, 0

   d. TAD 315
      DCA PLUS3

11. What is wrong (if anything) with the following:

```
.R ABSLDR
*SAM.BN <DTA1:PROG1.BN,PROG2.BN$
```

   a. No output device was specified and the ABSLDR program needs a specified output device.
   b. The alt mode key is not needed since all binary programs end with a dollar sign anyway.
   c. The RUN command should have been used rather than the R.
   d. No output should be expressed when using the ABSLDR.

12. If the following command string were given to the PAL8 Assembler, what would happen?

*SAM, LPT < DTA1:SAM.PA

a. Output would be to the Default Storage Device in binary and the lineprinter in listing.

b. Since no extension was given to the program SAM on output, an error message would be generated if SAM.BN could not be found.

c. SAM.BN would be stored on the Default Storage Device, and LPT would be stored as well on the same device.

d. If there is no file named SAM.PA on device DTA1, then the Assembler will look for a file named SAM with no extension.

13. Which of the following devices are non-directory devices?

a. DECtape

b. Card Reader

c. Fixed Head Disk

d. High Speed Paper Tape Reader

14. After PAL8 assembles each of the following instructions, what will they do when executed?

a) TAD (3000

b) JMS I (3000

c) JMS (3000

d) TAD I (3000

15. If the following command string were given to the CREF program, where would the output go?

*SAM < DTA1: PROG.LS

a. A file named SAM will be written on DSK
b. A file named SAM will be written on DTAO
c. SAM is ignored as a filename and LPT will be output device
d. Cannot be determined

16. Why is .ODT a system command rather than a program filename gotten by .R ODT?

    a. Because there are too many programs in the system library already.
    b. So that effective swapping can occur between ODT and the user program.
    c. ODT is not in core image format.
    d. The monitor won't allow two programs to be core resident at the same time.

17. What error could occur if the following were input to PIP?

    *DTA0:SAM < PTR:

    a. No error should occur because this is proper
    b. Can't transfer from a non-file structured to a file-structured device
    c. No option was specified so the paper tape had better be in ASC11 format else the transfer will not be proper.
    d. Since no file name was given on input the OS-8 monitor won't know which file to get.

18. What are the maximum number of input files allowed to the command decoder?

    a. 3
    b. no limit
    c. 9
    d. depends on the input device

19. If we wish to debug a program using ODT, which of the following would be the correct procedure under OS-8?

    a.
    ```
    .R ABSLDR/G $ (ALT MODE)
    *PTR:
    .OD
    ```

    b.
    ```
    .OD          (C.R.)

    .GET  SYS  SAM.BN
    ```

    c.
    ```
    .GET  SYS  SAM.BN   (C.R.)
    .OD                 (C.R.)
    ```

    d.
    ```
    .R ABSLDR           (C.R.)
    *SAM.BN$            (ALT MODE)
    .OD                 (C.R.)
    ```

20. While using ODT the user typed -

1615 / 3725 ↑ (SHIFT N)
X / 5066 ← ~~(SHIFT O)~~
Y / 7777 ↓ (LINE FEED)
Z / 1036 ↙ (C.R.)

What is the value of X, Y, & Z?_____

OS/8 FINAL

1. How would PAL8assemble the following--

```
A=0
B=1

IFNZRO A < IFNDEF B < XLIST
  TAD A
  TAD B
  XLIST >>
```

   a. The instructions TAD A & TAD B would not be assembled

   b. The instructions TAD A & TAD B would be assembled but would not appear on the listing

   c. PAL8 would give a PH error

   d. The instruction TAD A & TAD B would be assembled and also would appear in the listing.

2. After giving the command 15, 20$9M to the Editor

   a. lines 15 thru 20 will now be lines 8 thru 13 and "." will be equal to 9

   b. lines 15 thru 20 will now be lines 8 thru 13 and "." will be equal to 20

   c. lines 15 thru 20 will now be lines 8 thru 13 and "." will be equal to 13

   d. none of the above

3. Which of the following can change the job status word?

   a. Loading a program with the absolute loader
   b. Loading a core image file
   c. Changing location 7746 under program control
   d. Using either the .R or the .RU commands
   e. All of the above

4. Which of the following devices is (are) non directory?

   a. DTA7
   b. CDR
   c. DSK
   d. LPT

5. The following command does what? .R PIP

   a. Loads file PIP.SV from DTA0: and starts the program.
   b. Loads file PIP.SV from SYS: and starts the program.
   c. Loads file PIP from SYS: and starts the program.
   d. Illegal because there are to few arguments.

6. Write the command string required by the command decoder to do the following operations. (Give response to right of asterisk)

.R EDIT (CR)	/Run the editor program
*	/Create file XYZ.PA on Dectape unit #1
*	/Edit file XYZ.PA on Dectape #1 putting /edited version back on Dectape #1.
.R PAL8	/Run PAL8
*	/Assemble XYZ.PA with listing on line /printer and binary on system device
*	/Assemble XYZ.PA with listing on teletype /and no binary output.
*	/Assemble XYZ.PA with binary output as a /paper tape and no listing output
*	/Assemble XYZ.PA excluding symbol /table from listing and binary on Dectape /#1. Also tell PAL8 to chain to loader /for load and go option
.R ABSLDR	/Run the absolute loader
*	/Load file XYZ.BN with starting address /3055
*	/Load file XYZ.BN and after loading exit /back to monitor.
*	/Load the file XYZ.BN and set Bit 10 of JSW
.ODT	/Start ODT
-------	/Examine location 7600 of field 1
-------	/Examine location 7744 thru 7746 of field


```
--------                    /Set breakpoint at 3100

--------                    /Start program at 3055

03100 (010000               /At breakpoint go back to monitor


                            /Save program with following arguments
                            /core limits are 0 thru 600
                            /3000 thru 3600 with starting address
                            /of 200 and JSW of 3401
```

7. If the user has assigned the names "out" to the PTP and "in" to the PTR and wished to change the PTR's symbol "in" to "from", what must he do?

   a. Use the deassign command by .DE
   b. Assign "from" to the PTR by .AS PTR from
   c. Assign the PTR to "from" by .AS from PTR
   d. Use the change command by .CHA in from

8. The core control block contains what information?

   a. The job status word
   b. Starting address of .SV and .BN files
   c. Areas of core used by the program
   d. The date of the file creation

9. Which locations in core should <u>never</u> be used by an OS/8 user?

   a. All locations are available
   b. 0-1777 of Field 0 and Field 1
   c. 7600-7777 of Field 1 and Field 2
   d. Last page of core in Field 0 and Field 1

10. A .GE command returns control where?

    a. Command decoder
    b. Your program that was just loaded
    c. Keyboard monitor
    d. Depends on what the .GE command receives

11. If the command .GE DSK Prog were to be given to the monitor, what previous commands would cause the monitor to load from DTA7?

12. If the following was given to the command decoder
    * DTA5: SAM, PROG, PTR<ABC, DEF, PTR:,,

    a. INPUT - SYS: ABC, SYS:DEF, PTR:, PTR:
       OUTPUT - DTA5: SAM, DTA5: PROG, PTR

    b. INPUT - DSK: ABC, DSK: DEF, PTR:, PTR:
       OUTPUT - DTA5: SAM, DSK: PROG, PTR:

    c. INPUT - DSK: ABC, SYS: DEF, PTR:, PTR:, PTR:
       OUTPUT - DTA5: SAM, DSK: PROG, PTR:

    d. INPUT - DSK: ABC, DSK: DEF, PTR:, PTR:, PTR:
       OUTPUT - DTA5: SAM, DSK: PROG, DSK: PTR

13. To create a new program on DTA1 with the Editor, what input string should be given to the command decoder?

    a. DTA1: NAME.PA < TTY
    b. DTA1: NAME.PA < TTY:
    c. DTA1: NAME.SV < TTY:
    d. DTA1: NAME.PA <

14. The single quote (') indicates what to the buffer search?

    a. A terminating character to begin the search
    b. Nothing. The Editor types a question mark
    c. That the search is to begin at the current location counter's line number
    d. Search begins at .+1

15. How many OS/8 blocks are reserved by the system?

    a. 2
    b. 70
    c. Depends on how many programs are on the SYS device
    d. None

16. What are the three types of files created by OS/8

    a. Null, Permanent, Temporary
    b. Permanent, Temporary, Empty
    c. Permanent, Temporary, Tentative
    d. Tentative, Empty, Permanent

17. Choose the incorrect statement(s):

    a. A core image file always has a core control block
    b. A core image file is not packed
    c. The core segment double words are part of the core control block
    d. The segment header is a four word heading at the beginning of all file directories.

18. Which of the following statements is (are) true?

    a. the command decoder is one of the user service routines
    b. a reset always insures that the next handler fetched will be loaded where you want it.
    c. Arguments specified to the user service routine must be in the same field as the call.
    d. A device handler can be loaded anyplace in core except the last page of field 0 or 1.

19. In your own words explain how OS/8 protects you from having two tentative files on a device?

20. Which of the following statements is false?
    When entering a device handler:
    a. We must have determined previously whether we are to perform input or output
    b. The AC must contain the device code for the particular device handler
    c. The field of the call must be set in the DF register
    d. The device handler has to be resident in core

21. -Name two instances when arguments would have to be given with the Save command.

    -Name two instances when arguments wouldn't have to be given with the Save command.

22. After performing a transfer of data and the device handler exits to your program what should follow?

    a. Halt or JMP error if error return; next instruction of program is normal return
    b. Check to see if AC is positive or negative. If positive, means good transfer, if negative, means fatal error
    c. A JMP to a routine that will close the file
    d. None of the above--If the handler exits to your program the transfer was successful

23. Which location is used when referencing the USR to perform a USRIN?

    a. 7700
    b. 0200
    c. 10200
    d. 17700

24. When performing a decode the subroutine will bring which device handler into core if not already incore?

    a. The default storage device
    b. The TTY
    c. Both if needed
    d. None

25. To assure device independence what should the user do?

    a. Always leave Bits 1-5 of the function control word at 0
    b. Always enter a Fetch with the AC containing the device number
    c. Always specify the device by name when doing a Fetch
    d. Nothing can be done. A device is always dependent.

26. If the User wished to check for a device handler in core what should he do?

    a. Perform a lookup
    b. Examine core locations by getting an octal dump
    c. Perform an inquire
    d. Examine argument 2 of all Fetch routines

27. In your own words explain why the command sequence in example A will save a good copy of Build while the command sequence in example B may not.

```
Ex.A  .RUN SYS BUILD
       $
          .
          .
          .
       BOOT
      .SAVE SYS BUILD
```

```
Ex.B  .R BUILD
       $
          .
          .
          .
       BOOT
      .SAVE SYS BUILD
```

digital

DIGITAL EQUIPMENT CORPORATION, Maynard, Massachusetts, Telephone: (617) 897-5111 • ARIZONA, Phoenix • CALIFORNIA, Sunnyvale, Santa Ana, Los Angeles, San Diego and San Francisco (Mountain View) • COLORADO, Engelwood • CONNECTICUT, Meriden • DISTRICT OF COLUMBIA, Washington (Riverdale, Md.) • FLORIDA, Orlando • GEORGIA, Atlanta • ILLINOIS, Northbrook • INDIANA, Indianapolis • LOUISIANA, Metairie • MARYLAND, Riverdale • MASSACHUSETTS, Cambridge and Waltham • MICHIGAN, Ann Arbor and Detroit (Southfield) • MINNESOTA, Minneapolis • MISSOURI, Kansas City and Maryland Heights • NEW JERSEY, Fairfield, Metuchen and Princeton • NEW MEXICO, Albuquerque • NEW YORK, Huntington Station, Manhattan, New York, Syracuse and Rochester • NORTH CAROLINA, Durham/Chapel Hill • OHIO, Cleveland, Dayton and Euclid • OKLAHOMA, Tulsa • OREGON, Portland • PENNSYLVANIA, Bluebell, Paoli and Pittsburgh • TENNESSEE, Knoxville • TEXAS, Dallas and Houston • UTAH, Salt Lake City • WASHINGTON, Bellevue • WISCONSIN, Milwaukee • ARGENTINA, Buenos Aires • AUSTRALIA, Adelaide, Brisbane, Crows Nest, Melbourne, Norwood, Perth and Sydney • AUSTRIA, Vienna • BELGIUM, Brussels • BRAZIL, Rio de Janeiro, Sao Paulo and Porto Alegre • CANADA, Alberta, Vancouver, British Columbia; Hamilton, Mississauga and Ottawa, Ontario; and Quebec • CHILE, Santiago • DENMARK, Copenhagen and Hellerup • FINLAND, Helsinki • FRANCE, Grenoble and Rungis • GERMANY, Cologne, Hannover, Frankfurt, Munich and Stuttgart • INDIA, Bombay • ISRAEL, Tel Aviv • ITALY, Milano • JAPAN, Osaka and Tokyo • MEXICO, Mexico City • NETHERLANDS, The Hague • NEW ZEALAND, Auckland • NORWAY, Oslo • PHILIPPINES, Manila • PUERTO RICO, Miramar and Santurce • REPUBLIC OF CHINA, Taiwan • SCOTLAND, West Lothian • SPAIN, Barcelona and Madrid • SWEDEN, Solna and Stockholm • SWITZERLAND, Geneva and Zurich • UNITED KINGDOM, Birmingham, Bristol, Edinburgh, London, Manchester, Reading and Warwickshire • VENEZUELA, Caracas

PRINTED IN USA 0302 00174 3534/E 14 1.2