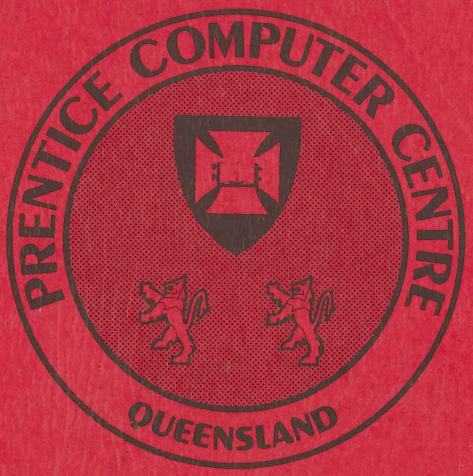
PRENTICE COMPUTER CENTRE



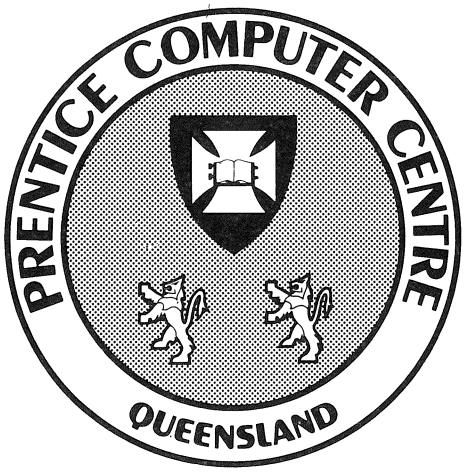
USING THE PDP-10 SYSTEM

Technical Manual Number 2

MNT-2
June, 1981

This manual has been authorized by the Director of the Prentice Computer Centre.

PRENTICE COMPUTER CENTRE



USING THE PDP-10 SYSTEM

Edited: N.J. Meier

MNT-2
June, 1981

INTRODUCTION

The Prentice Computer Centre provides a wide range of computing facilities and associated services for its clients.

Computing services are available from a number of host computers which are interconnected and accessed from user terminals via an extensive communications network. The network is depicted in Chapter 7 of this manual.

The user at his terminal can select and access the service that best meets his specific needs, be it general computing on the DEC KL-10, specialized student computing (SLOTS) on the DEC KA-10, or research work requiring large memory space (LOADS) on the VAX. Users can also access computing services on the CSIRO CYBER Computer or, via the MIDAS satelite computing service, the USA networks.

This manual gives an introduction to the use of the KL-10 and KA-10 host computers located at the Prentice Computer Centre. The manual is intended for those who are using the computer for the first time and wish to work slowly through the fundamental commands necessary to operate the KL-10 or KA-10, and those who have previously used computers and want a quick review of the command language for the PDP-10 machines.

In BOTH these cases it is STRONGLY RECOMMENDED that you

(a) Attend a suitable introductory course at the Prentice Computer Centre

(b) Work through this manual at a terminal, actually entering all the examples given.

CONTENTS

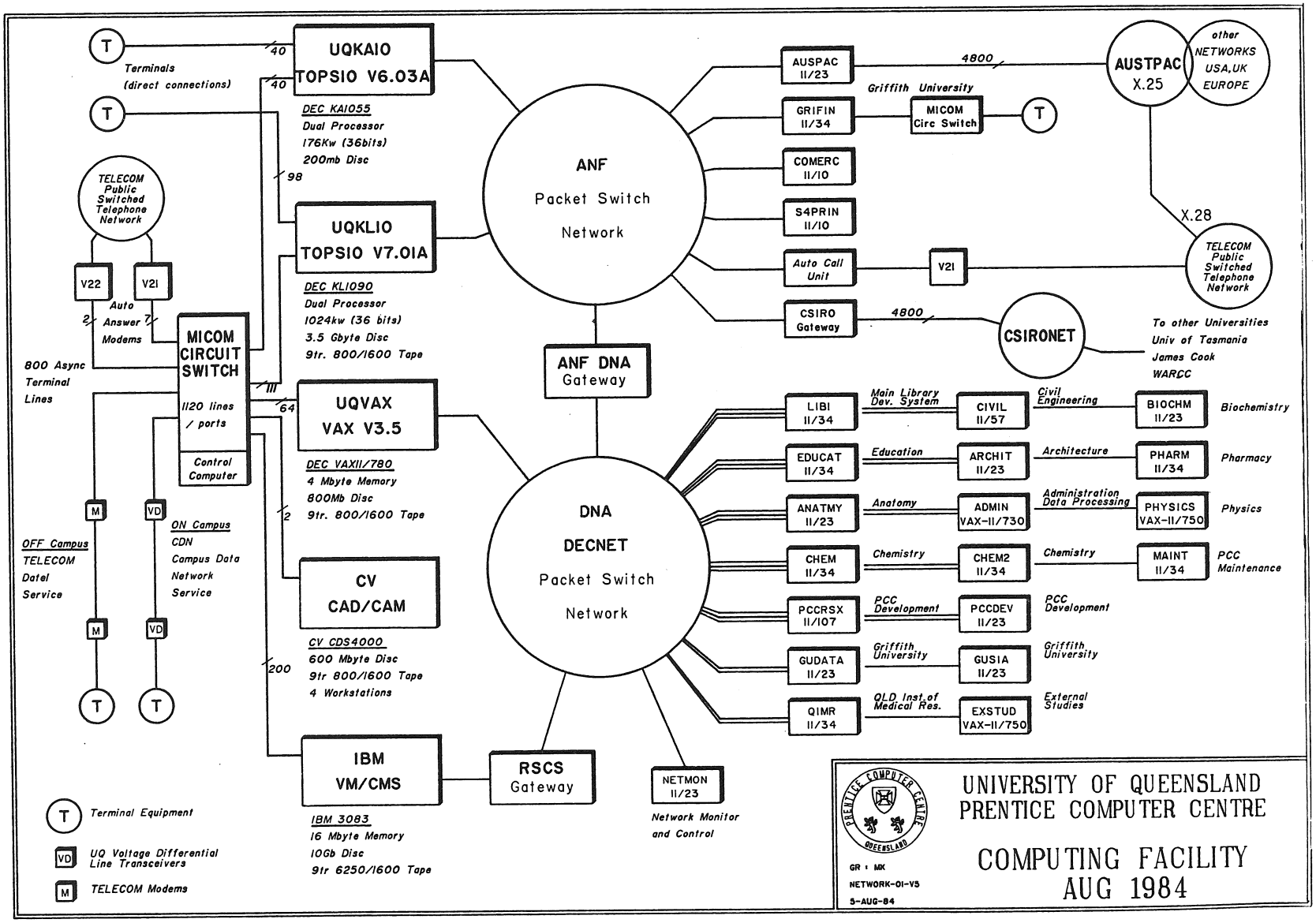
PREFACE

This manual is intended as an introduction to the use of the facilities of the PDP-10 Computer Systems at the University of Queensland. It does not provide the detailed material which may be required by the more experienced user and, as such, should not be considered as an alternative to the programmers' reference handbooks produced by Digital Equipment Corporation or the Centre. To this extent, the manual provides as many cross references as possible to the relevant sections in other publications. It is left to the discretion of the reader as to the depth of knowledge required, and thus the references pursued. Consequently, to gain maximum benefit from this manual, the reader should have access to the various documents referenced throughout.

The manual itself has been subdivided into three sections.

Part I (Chapters 1-3)	provides a general overview of the system and its components;
Part II (Chapters 4-7)	explains the basic operations for general user processing; and
Part III (Chapters 8-10)	provides additional facilities for the more experienced user.

Users not familiar with computers and requiring instructions for its use are referred to Part II - USER PROCESSING.


Terminals
(direct connections)
UQKAI0
TOPSI0 V6.03A
DEC KAI055
Dual Processor
176Kw (36bits)
200mb Disc
UQKLI0
TOPSI0 V7.01A
DEC KLI090
Dual Processor
1024kw (36 bits)
3.5 Gbyte Disc
9tr. 800/1600 Tape
TELECOM
Public
Switched
Telephone
Network
V22
V21
Auto
Answer
Modems
MICOM
CIRCUIT
SWITCH
1120 lines
/ ports
Control
Computer
800 Async
Terminal
Lines
UQVAX
VAX V3.5
DEC VAXII/780
4 Mbyte Memory
800Mb Disc
9tr. 800/1600 Tape
CV
CAD/CAM
CV CDS4000
600 Mbyte Disc
9tr 800/1600 Tape
4 Workstations
IBM
VM/CMS
IBM 3083
16 Mbyte Memory
10Gb Disc
9tr 6250/1600 Tape
RSCS
Gateway
NETMON
11/23
Network Monitor
and Control
ANF
Packet Switch
Network
ANF DNA
Gateway
DNA
DECNET
Packet Switch
Network
OFF Campus
TELECOM
Datel
Service
ON Campus
CDN
Campus Data
Network
Service
Terminal Equipment
UQ Voltage Differential
Line Transceivers
TELECOM Modems
AUSPAC
11/23
GRIFIN
11/34
COMERC
11/10
S4PRIN
11/10
Auto Call
Unit
CSIRO
Gateway
4800
Griffith University
MICOM
Circ Switch
V21
CSIRONET
To other Universities
Univ of Tasmania
James Cook
WARCC
AUSTPAC
X.25
other
NETWORKS
USA,UK
EUROPE
X.28
TELECOM
Public
Switched
Telephone
Network
LIBI
11/34
Main Library
Dev. System
CIVIL
11/57
Civil
Engineering
BIOCHM
11/23
Biochemistry
EDUCAT
11/34
Education
ARCHIT
11/23
Architecture
PHARM
11/34
Pharmacy
ANATMY
11/23
Anatomy
ADMIN
VAX-11/730
Administration
Data Processing
PHYSICS
VAX-11/750
Physics
CHEM
11/34
Chemistry
CHEM2
11/34
Chemistry
MAINT
11/34
PCC
Maintenance
PCCRSX
11/107
PCC
Development
PCCDEV
11/23
PCC
Development
GUDATA
11/23
Griffith
University
GUSIA
11/23
Griffith
University
QIMR
11/34
QLD Inst.of
Medical Res.
EXSTUD
VAX-11/750
External
Studies
UNIVERSITY OF QUEENSLAND
PRENTICE COMPUTER CENTRE
COMPUTING FACILITY
AUG 1984
GR : MK
NETWORK-01-V5
5-AUG-84

CHAPTER 1

THE PDP-10 COMPUTER

1.1 System Hardware

The University of Queensland Prentice Computer Centre has two separate DEC-10 Computer Systems (the PDP-1055 and PDP-1090). The PDP-1055 provides the Student Low Overhead Timesharing Service (SLOTS) while the PDP-1090 handles all other computing needs. The systems are interconnected so that users connected to the PDP-1090 can also access the SLOTS service on the PDP-1055.

The PDP-1055 (KA10) computer system is the older system, and currently processes mainly student jobs. It is a dual processor configuration with 240K of 36 bit words (1Kword=1024words) of core memory, and connecting the following peripherals:

* 2 high speed fixed head disk drives)together providing online disk
* 5 movable head disk drives)storage of 250 million characters
* two 7-track magnetic tape drives
* two 9-track magnetic tape drives
* optical mark card reader
* punched card reader
* line printer (132 characters per line - upper case characters only)
* more than 70 terminals of various types providing access to only the KA system.

The PDP-1090 (KL10) computer system was installed in March 1978. It provides more than twice the capacity of the KA system and comprises one KL10B processor with 768K of core memory connecting the following peripherals:

* disk units providing online disk storage of 2500 million characters
* two 9-track magnetic tape drives
* high speed line printer (132 characters per line - upper and lower case characters)
* access to the 838mm plotter
* more than 170 terminals.
* optical and punched card reader.

The KL10 processor located at the central site can be accessed via the communications network from separate batch stations, one located at Griffith University and the other in the Commerce Building at St. Lucia. The Griffith Uni. batch station comprises a PDP-11 computer (with an optical card reader, 600 line per minute printer, plotter and console terminal) and a MICOM multiplexor both connected to the Prentice Computer Centre via a high

speed communication lines. The Commerce batch station contains a PDP-11 mini computer with a card reader and 180 character per record printer on a communications line.

For both computer systems, the existence of these peripherals does not necessarily imply that they are available to all computer users.

The KL10B processor is largely compatible with the older KA10 processors and has additional hardware instructions for double precision integer and floating point arithmetic plus 19 business instructions to enhance the speed of commercial applications. Each central processor is itself the control unit for that computer system. It governs all input/output equipment, schedules programs, and performs all arithmetical, logical and data handling operations. The processor is connected to core memory by a memory bus, and to peripherals by an input/output (I/O) bus. The secondary storage devices, though controlled by the processor over the I/O bus, have direct access to core memory via a "data channel". Slow devices such as the line printer and the card reader are controlled directly by the processor.

Thoughout the remainder of this manual, all the information given applies equally to both the KL10B and KA10 computers unless there is some specific statement to the contrary.

1.2 Number and Character Representations

A bit is the smallest unit of storage within a computer and is capable of representing one of two possible states, e.g. 0 or 1; ON or OFF. Consequently a binary numbering system is used. Bits are grouped together in WORDS and, in the PDP-10, words comprise 36 contiguous bits. The bits are numbered 0-35 from left to right.

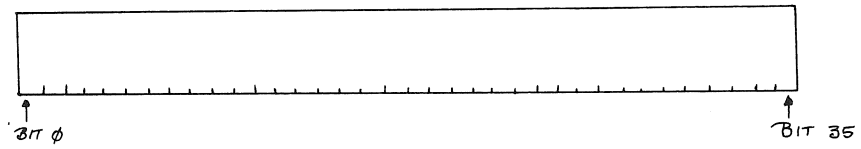


Figure 1.1 Word Format

There are various ways of storing data in a word in memory. Numbers are stored as integers or real numbers, and data and text can be stored as characters, each character having its own binary code.

1.2.1 Integers

Integers are numerical values without any fractional part, but their values must fall in the range -{(2**35)-1} to +{(2**35)-1}. The sign of the number is represented in bit position 0, and the magnitude in bit positions 1 - 35.

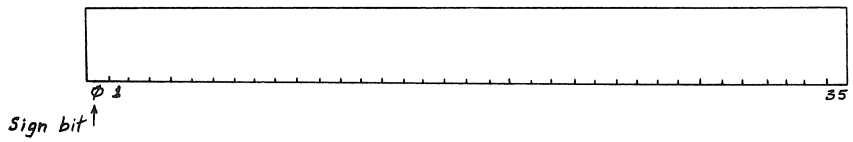


Figure 1.2 Integer Representation

1.2.2 Real Numbers

Real (or floating point) numbers are numerical values containing decimal points. They are represented as some number modified by a decimal scale factor called the exponent that gives the power of ten by which the number should be modified.

Values of floating point numbers must fall in the range 0.14E-38 to 1.7E+38. Only the nine most significant digits of a fraction are stored, the ninth digit being rounded.

In the word, bit 0 represents the sign, bits 1-8 represent value of exponent, and bits 9-35 represent magnitude of the fraction.

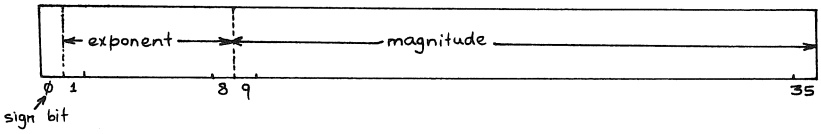


Figure 1.3 Real or Floating Representation

It is possible to increase the accuracy of floating point numbers by using Double Precision numbers. These must be within the same range as Single Precision numbers but, by utilizing two words of memory, the number is stored correct to 16 significant decimal digits.

1.2.3 Characters

Data or text can be stored in ASCII form (American Standard Code for Information Interchange). For ascii character representation, a 36 bit word is divided into five characters, each represented by a seven bit code. Bit 35 is ignored.

Data can also be stored in SIXBIT form. For sixbit character representation, a 36 bit word is divided into 6 characters, each represented by a six bit code.

Not only alphabetic characters and special symbols, but also numeric digits can be represented as either ASCII or SIXBIT characters. (However, meaningful arithmetic cannot be performed on a number which has been stored as a character.)

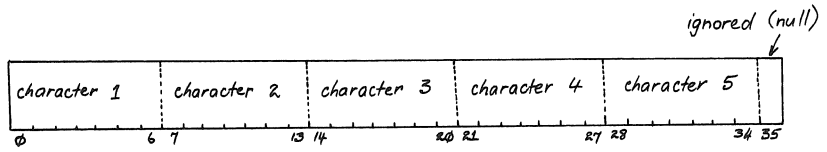


Figure 1.4 ASCII Character Representation

1.3 The PDP-10 Operating System

Users communicate with the PDP-10 through an intermediary, the OPERATING SYSTEM, in order to direct their problems to the actual machine and receive solutions back. The operating system also keeps track of what each user does and the devices and resources that each user employs. Though the operating system cannot be physically seen, it is the most important part of the machine to each user. In most respects, the operating system handles any differences in hardware configuration for the user.

The operating system is always resident in core memory and is composed of three parts:

1) Service Request Handler
2) Sharable Resource Allocator
3) I/O Service Routines

The Service Request Handler receives requests for such things as core memory and processor time, and passes these requests to the Sharable Resource Allocator which is responsible for the actual allocation. The I/O Service Routines handle requests for I/O device use.

1.4 PDP-10 Timesharing and Multiprogramming

The PDP-10 provides concurrent operation of interactive, batch, real time and remote communications work. Interactive users can also initiate batch jobs.

1.4.1 Timesharing

The PDP-10 makes maximum use of its resources by sharing the facilities among users. Processor time and system resource usage are optimised, in contrast to a single user system where facilities are not shared. Users are not restricted to a small set of system resources, but generally have access to most of the systems resources available. This access is controlled by the operating system. The user communicates with the operating system via commands which enable him to control the running of his job. He can create, edit and delete files; start, suspend, and terminate a job; compile, execute and debug programs. In addition, users can submit jobs for batch processing from remote terminals.

1.4.2 Batch Processing

Multiprogram Batch (MPB) software on the DEC-1055 system is functionally very similar to the batch software (GALAXY) on the DEC-1090. Several streams of batch jobs run under control of BATCON concurrently with terminal controlled timesharing jobs. Whereas the interactive user supplies input from his terminal, the batch user must give the input as a deck of cards or a disk file.

1.4.2.1 Batch Components -

The batch software consists of a series of programs: the card input spooler, SPRINT; the terminal user's interface to the queues, QUEUE; the batch controller, BATCON; the queue manager, QMANGR(on the KA-10) or QUASAR(on the KL-10); and the output spoolers for the various slow output devices (LPTSPL, PLTSPL, SPROUT). The input spooler is responsible for reading the cards and acting on the control cards to create disk files for batch processing. One of the files created is the control file from which the batch job will get its commands. Data files and program files are created as requested by control cards. In addition, the input spooler creates the job's log file, in which it enters a report of its processing of the job along with any error messages concerning faulty cards. This log file is part of the standard output of each job and is invaluable for tracing causes of trouble. The log file will be described in greater detail in Chapter 6.

Once the input spooler has read the end-of-job card and closed the disk files, it makes an entry in the batch controller's input queue.

The terminal user may submit control files for batch processing, put entries in the output queues and check the progress of jobs in the batch queues. To do this he uses the SUBMIT, PRINT and PLOT commands to invoke the program QUEUE. The batch controller processes batch jobs by reading the entries in the queue. Operating system commands are detected by the batch controller and passed to the operating system for action. Most commands are available to both the interactive and batch user, thus simplifying overall system use.

The queue manager is responsible for scheduling jobs and maintaining both the batch controller's input queue and the output spooling queues. The queue manager examines parameters specified by the user such as processor time to be used, job priority and core requirements.

The output spooler programs improve system throughput by allowing output from jobs to be written to disk for later transfer, rather than being copied directly to the output device. This prevents programs with small volumes of sparse output, monopolsing a slow output device for the duration of its processing cycle.

1.4.3 Program Scheduling

In order to be able to carry out the tasks for which it was written, a program must have available to it a number of the resources of the system. In a timesharing system, where a number of users may be competing for the use of the limited number of resources, some means must be provided whereby each user receives a fair share of the resources required. The process whereby this is achieved is called SCHEDULING.

Each program is assigned a fixed time period, and operation is switched from one program to another until each is completed. This process is called CONTEXT SWITCHING and is driven by a clock which interrupts the processor when the time period allocated to each job has elapsed. The operating system also restricts each user's actions to within the limits of his own area, both on disk space and core memory. The area that a particular user can access is limited and any attempt by a user's program to access outside this area will cause the program to stop.

The most restricting influence on the number of users which the system can handle at any time, is the amount of core memory. Since core memory is rather expensive, a satisfactory supplement to core memory can be provided by secondary storage devices such as magnetic tapes and disk packs. User programs are located on a secondary storage medium until required for execution, at which stage they are taken from secondary storage and loaded into core for execution. Programs entering core take the place of programs that have just been serviced by the processor. This operation is called SWAPPING.

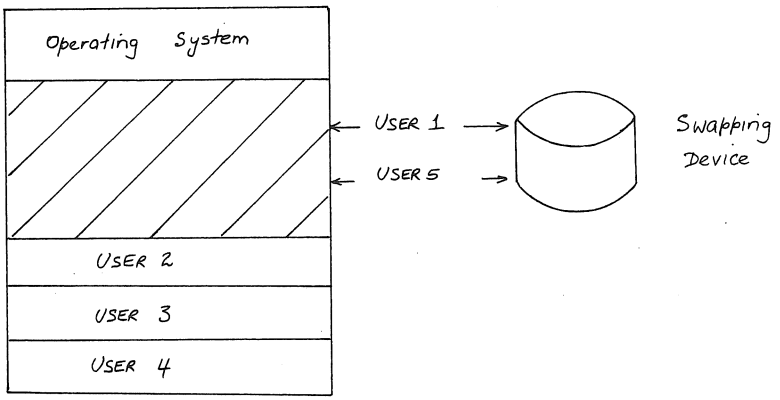


Figure 1.5 Swapping

In operation, core is divided into separate memory areas called PAGES. Secondary storage is connected to these pages via high speed data channels. The data channel allows data to be transferred between core and secondary storage without control of the processor. Thus the central processor can operate a user program at the same time as programs are being swapped into and out of other pages of memory. This overlapping greatly improves the efficiency of the system.

On the KA system, swapping is performed using two high speed fixed head disk drives, while on the KL system one moving-head disk unit is used.

1.4.4 Dynamic Scheduling

The efficiency of program scheduling, where each program operates in sequence and receives a fixed amount of time is dependent upon the mix of job types running on the system. The ideal situation rarely occurs. At any particular time, the timesharing system will be handling a large mix of jobs, ranging from COMPUTE BOUND jobs that require a large amount of processor time, and INPUT/OUTPUT BOUND jobs that must stop frequently for input/output. To serve programs between these extremes, the scheduling algorithm must provide frequent service for I/O bound jobs and must give compute bound jobs longer time periods of processor time to prevent wasteful swapping.

CHAPTER 2

SYSTEM SOFTWARE

The SOFTWARE of a computer system is the collection of programs and procedures required for operation of the system. There are various classes of software on the PDP-10.

2.1 Core Resident System Software

This includes the OPERATING SYSTEM discussed previously. On a timesharing system such as the PDP-10, the operating system can be very large (currently about 89k on the KA and 146K on the KL), and is in core at all times.

The core resident software on the PDP-10 includes the COMMAND DECODER which interprets the user commands and passes them to the relevant section for action, the SCHEDULER which performs the function outlined in the previous chapter, and the SWAPPER which rotates jobs between core and secondary storage, after deciding what should be in core at a particular time and what should not.

2.2 Non-Resident System Software

For a computer to execute any of the operations which it is capable of executing, it must be told what operation to perform and where to find the information on which it is to operate. This requires that a language be defined, by means of which the user can indicate to the computer what it needs to know. This language is the MACHINE LANGUAGE of the computer and differs from computer to computer. Though machine language programming is the most powerful, it is not the easiest to employ and so SYMBOLIC LANGUAGES have been developed to enable the user to manipulate the computer.

With symbolic language programming, programs are written with symbols, which when translated become the machine language of the computer. Symbolic operation codes are translated into the actual operation codes that the computer understands and symbolic memory addresses are converted into actual addresses.

There are three types of translators used in symbolic language programming: assemblers, compilers and interpreters.

An ASSEMBLER is a program that takes another program written in a symbolic language and translates it instruction by instruction into machine language.

A COMPILER also translates a symbolic language into a machine language, but the substitution is not one for one. A program written in a compiler language is freer in format than an assembly language program and the language elements are problem-oriented, resembling algebraic notation or English words.

An INTERPRETER differs from a compiler or an assembler in that a binary version of the program is not produced for storage and the source program is converted to machine language everytime that it is used, allowing for extensive checking of errors during execution.

The symbolic assembly program on the PDP-10 is called MACRO-10. It is a two-pass assembler in that, during the first pass, all the symbols are read and, defined and on the second pass, the actual machine code is generated. Though slower than a one-pass assembler, MACRO-10 is more efficient since less core is required and output to the user is minimised.

The MACRO-10 assembler is device-independent and has powerful macro capabilities which allow for the expansion and adaption of the assembler in order to perform specialized functions for each programming job.

The PDP-10 has an interpreter for the BASIC programming language, and compilers for a number of other programming languages. These include FORTRAN, COBOL, ALGOL, SIMULA, and PASCAL. It also has editors (QEDIT, VIDED, VISED, SOS and TECO) and utility programs (CREF, DDT, PIP).

2.3 Reentrant Software

In a timesharing system, users may have a wide range of programs in execution at the same time. When a program is duplicated for several users, however, core storage can be excessive, and to overcome this, REENTRANT SOFTWARE is used.

A program is generated in two parts, one containing pure code that is not modified during execution and can be used to service simultaneously any number of users, and the other which belongs strictly to each user and consists of the code and data that is developed during the compiling process (impure code).

A comparison between the core usage of reentrant and non-reentrant software can be seen in figure 2.1.

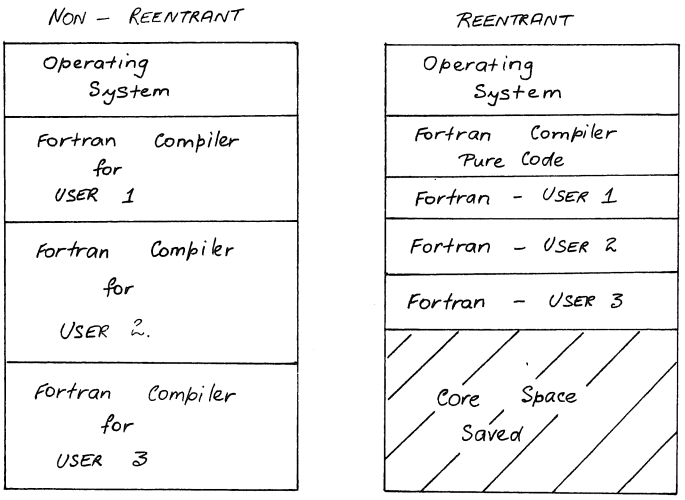


Figure 2.1 Reentrant & Non-reentrant Software

Reentrant software has many advantages, perhaps the most important being the saving of core memory. The operating system thus swaps less often and spends less time swapping the smaller impure sections. Also, since the pure code never changes, it does not have to be swapped out and, as long as a single copy is kept on disk at all times, it can be swapped into core at any time.

To protect the pure code from being modified, dual memory protection and relocation is provided. This hardware feature allows a program to execute as two separate segments, one of which is protected.

Further reading for Chapters 1 and 2 may be found in the DECsystem10 Users Handbook pages 9-23. A Glossary of Terms may be found in the same handbook on pages 31-44.

CHAPTER 3

PDP-10 FILE SYSTEM

3.1 Description of File System

The PDP-10 is a file based system. A file can be defined as a collection of 36 bit words, organised into BLOCKS (physical records). On the PDP-10 there are 128 words per disk block and blocking is done automatically by the system. A file can contain either programs or data, a PROGRAM being defined as a collection of instructions for the computer, written in some programming language, and DATA being defined as a collection of information to be used by a program, or which has been produced by a program. Programs and data are stored as files in exactly the same way and generally are only distinguishable by physical observation or name.

All information in the system is stored as named FILES, in a uniform and consistent fashion, thus allowing information to be referred to by name rather than by physical disk addresses.

Each user is allocated an amount of disk space for his files and he can have any number of files of any length within the limits of his allocated area. The user will be informed of his disk area allocation when he opens an account.

A file can exist on devices other than disk. For example, a deck of cards to be input via the card reader is in fact a file. Similarly, information on magnetic tape can be a series of files.

Since users wish to keep a number of separate files on the disk, some logical method of storage must be employed. The names and locations of the files on each user's area are listed in order in a special file called the User's File Directory (UFD). A Master File Directory (MFD) and a disk map are then required to maintain the locations of the UFDs and also to keep track of the amount of free storage that can be assigned to new files. The resulting hierarchy is shown in figure 3.1.

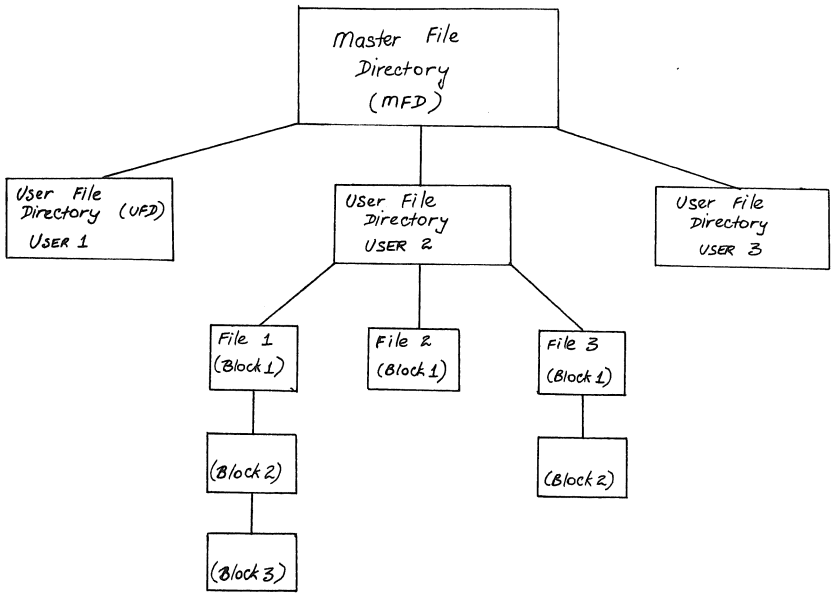


Figure 3.1 File Hierarchy

The DISK STORAGE SYSTEM on the PDP-10 must cater for a large number of different users. To facilitate this, it is composed of separate STRUCTURES, each structure using one or more disk drives. Most user files are contained on the structures designated DSKB:, DSKD:, DSKE: or DSKH: on the KA, and DSKB:, DSKD:, DSKE:, DSKF:, or DSKG: on the KL. Users may store their files ONLINE as above, for immediate access, or OFFLINE on disk or magnetic tape. To obtain files which have migrated from the online storage to offline storage requires a request from the user which may take some short time to service. Files should be migrated by the user to overcome cluttering of the online storage facilities (see Chapter 8 - File Migration System).

3.2 Filenames

Files are identified by filenames. These are composed of a name, decided by the user, and optionally an extension. The general form of a filename is as follows:

```
name.extension
```

The NAME part can be a maximum of 6 alphanumeric characters (excluding special symbols).

Examples:

(i)	FILE42	)
(ii)	SAMPLE	) correct filenames
(iii)	TT4	)
(iv)	123TRS	)
(v)	UPD*$	) incorrect filenames
(vi)	TESTING	)

The EXTENSION name, if present, is part of the filename, but separated from the name part by a period (.). The extension can be a maximum of 3 alphanumeric characters.

Examples:

(i) FILE1.BOB
(ii) TEST.F10
(iii) TEST.1

The extension may be used as a type of identification decided by the user, or it may have some special significance to the system.

Extensions belong to a small group of standard names used to indicate the type of information in a file. Since they can also indicate the type of processor that will be required by a particular file, some extensions permit automatic compilation and execution of program files (see Section 5.4.1 "Compilation and Execution").

The following standard extensions are presently used by the system. A more comprehensive list of standard extensions may be found in Appendix A of the DECsystem10 Operating System Commands Manual.

Standard Extension	Indication
BAK	Backup copy of edited file
TMP	Temporary file
ALG	Source file in ALGOL language
BAS	" " " BASIC "
CBL	" " " COBOL "
FOR	" " " FORTRAN-10 "
F10	" " " FORTRAN-10 "
MAC	" " " MACRO "
PAS	" " " PASCAL "
SIM	" " " SIMULA "
REL	Relocatable binary file
EXE	Core image file (SAVE command)
CDR	Card reader file
CMD	Command file
CTL	Control file for batch processing
DAT	Data file
LPT	Line printer file
LST	Listing file
SPS	SPSS control file

Files with any of the language extensions (ALG, CBL, FOR, F10, MAC, PAS, SIM) will cause the respective compilers to be invoked automatically.

Example:

TEST1.F10	indicates a source program in Fortran-10 language. Automatic compilation will cause the FORTRAN-10 compiler to be used.

A file with a DAT extension, when output to the line printer, will cause FORTRAN carriage control conventions to be employed.

3.3 File Specification

As well as being identified by name and extension, files can also be addressed by device, project,programmer number (ppn), and protection. The general form of the file specification is:

```
device:filename.extension[proj,prog]<protection>
```

A RETURN will be represented in the text as <cr>, and underlining to signify characters entered by the user. For example,

```
.COMPILE DSK:TEST1.FOR[160,105]<155><cr>
```

The device usually defaults to DSK:, and if no ppn specified, the user's own ppn is assumed.

By means of this general file specification, a user may access files from another user's area, depending on the protection of the required files.

3.4 Wildcard Constructions

In many commands, the filename, extension, or the directory name may be replaced totally with an asterisk, or partially with a question mark.

The ASTERISK is used as a wildfield to designate an entire name, extension, or directory.

Examples:

(i)	filename.*	all files with this filename and ANY extension
(ii)	*.ext	all files with this extension
(iii)	*.*	all files
(iv)	[65,*]	all ppn's with a project number 65.

The QUESTION MARK is used as a wild character to designate part of a filename, directory or extension. A question mark is used for each character that is to be matched.

Examples:

(i)	P???	matches on all filenames starting with P of four characters or less in size
(ii)	filename.M??	all files with filename and any extension starting with M
(iii)	TES??.ext	all files with filename 3 to 5 characters starting TES and with this extension
(iv)	BILT.MAC[6?,132]	all files with this filename and extension for any of the project numbers in the range 60 to 67 (ppn's being octal numbers).

CHAPTER 4

PDP-10 REMOTE TERMINALS

The main purpose of this chapter is to describe interactive processing and introduce users to the concept of timesharing and the use of remote terminals as input/output devices to the PDP-10. This section therefore, although discussing in general terms the whole concept of processing work on a computer, concentrates mainly on interactive processing and not on batch processing. Batch processing will be covered in as much detail in Chapter 6.

Underlining of text is used here to signify characters entered by the user.

4.1 Remote Terminal Communication

The remote terminal is a very useful and flexible communication medium between the user and the PDP-10. It enables the user to interact directly with the computer and, in so doing, carry out a large amount of work in a short space of time.

Programs can be typed directly into the computer by means of the terminal under the control of programs already resident in the system. The most important of these resident programs is the MONITOR, and just as the terminal is your link with the computer so the monitor is your link with the programs within the computer.

As communication between the terminal and the computer is carried over the TELECOM telephone network using modems (Modulator - demodulator equipment), terminals can be installed virtually anywhere. Each terminal is connected via a private line which is used solely for data transmission, with the exception of ACOUSTIC-COUPLED terminals, which enable communication with the computer from any location where a normal telephone is available. Acoustic-coupled terminals may only be installed temporarily in any location.

4.2 Types of Terminals

Although there are a variety of terminals currently connected to the PDP-10 Systems, the basic principles of operation are similar for all. Computer terminals are similar to electric typewriters, having a keyboard for sending information to the computer and a printer for receiving information from the computer. Alternatively, they may have some visual display capability on a cathode ray tube (CRT). Some terminals may also have a paper tape reader and punch or a cassette magnetic tape facility attached.

4.3 Operation of the Terminal

4.3.1 Switching On

Naturally, the terminal must be correctly plugged in before any useful work can commence - i.e. the terminal signal plug should be connected and the power plug connected and switched on.

Somewhere on the terminal should be a switch which has two positions: LINE and OFF; or LINE and LOCAL. (Some terminals may also have an extra power switch). If this switch is in the OFF or LOCAL positions, the terminal acts just as a typewriter and is not communicating with the computer. Terminals which have paper tape or magnetic tape facilities may be used in local mode for producing files.

For interactive use, this switch must be in the LINE or ON position to establish communication with the computer. Thus information typed on the keyboard is sent to the computer, examined, and then characters are echoed back for the user to see the results of processing. Typing the return establishes contact with the network; the terminal should respond with 'HOST = ' (see chapter 7 for hosts available): you respond with the name or number of the host computer you wish to use, e.g. HOST = 1. If the machine responds by typing 'GO' it is ready to start receiving commands. You then type <cr> and the system will respond by telling you which host you have connected to; you should then follow the login procedure (section 5.2).

If the system fails to respond, then the computer is unavailable for timesharing and the user should consult the Computer Centre Recorded Message Service (ext 3101) to determine the current system status.

The terminal should always be switched off when not in operation, as excessive running only wears the mechanical components and adds to maintenance expenses.

4.3.2 The Keyboard and Special Characters

The majority of keys on the terminal are self-explanatory and should be easily mastered by the new user. Some terminals provide more characters than others and some provide for upper and lower case. The manufacturer's handbook provides the range of facilities offered by a particular terminal. Also the mnemonics used by some manufacturers may vary. The basic operation, however, is independent of the mnemonic (e.g. RUBOUT/ERASE/DELETE).

The following paragraphs describe a number of keys of particular importance:

(a) RETURN (Depress the RETURN key)

A RETURN is echoed by the system as two operations, a carriage return and a line feed. This means that the typing head returns to the first character position of the line and then advances to the next line. All input to the terminal must be terminated by a RETURN to signify that the command entered is complete and the required processing can be started.

NOTE: In LOCAL operation, the combined carriage return/line feed action of the RETURN key does not occur and depressing the RETURN key only causes the head to return to the first character position of the same line. To advance one line, a separate key, the LINE FEED key should be depressed.

A RETURN will be represented in this text as <cr>, e.g.

```
.DELETE *.BAK<cr>
```

(b) RUBOUT (DELETE or ERASE)

Typing errors can be corrected using the RUBOUT key. Depressing this key once, results in the last character typed being deleted. Depressing the RUBOUT key n times results in the last n characters being deleted. The operating system echoes the deleted characters, delimited by backslashes. (The end backslash is echoed by the system when the user depresses the key for the next new character.) On the KL10 if the correct terminal characteristics are set (refer section 4.4), a video terminal will do a back-space rub out, and the characters rubbed out will disappear.

Example:

If the command CREATE was to be given, but was inadvertantly typed CRAET, then three RUBOUTs could be given to delete the A, E, and T, and enable the user to correct the mistake. When the RUBOUT character is entered, it is sent to the computer and causes the last uncorrected character entered by the user to be echoed back.

```
.CRAET\TEA\EATE(paper terminal only)
```

Note: Deleted characters appear in the order in which they are deleted. The end backslash is echoed when 'E' is typed.

(c) ALTMODE (ESC or PREFIX)

The ALTMODE key is used as a command terminator for various progams and monitor commands. Since the altmode is a non-printing character, the terminal prints it as a dollar sign ($).

(d) CONTROL (Depress the CONTROL key)

This key is used to type control characters. The symbol '^' (up-arrow) when appearing before a character, indicates that the particular character is a control character. The CONTROL key must be depressed while typing the particular letter. A number of control characters have special meaning and provide certain features on the terminal. These are described below.

(e) CONTROL-C (^C)

^C normally interrupts a program if one is currently running and returns control to the monitor. The monitor responds by typing a period (.) if it is ready to accept a new command. If however a program is in the middle of execution rather than waiting for input, it is necessary to type control-C twice to get control back to the monitor.

(f) CONTROL-I (^I)

^I is a horizontal tab character. On the terminal there are a number of tab positions which may be set (see the manufacturer's manual). When the monitor recognises a tab character, all intervening positions are spaced until the next tab stop. Within the computer, the tab is considered as a single character and requires only one RUBOUT to delete it.

(g) CONTROL-O (^O)

^O suppresses output on the terminal. For example, if output from a long program is being typed and the user has already discovered the information required, then ^O can be typed to suppress the remainder of the listing, without terminating the normal execution of the program. A second ^O restores output.

(h) CONTROL-R (^R)

^R retypes the current line and is useful after rubout processing.

Example:

Using the example given in (b) on use of the RUBOUT key, if the user wished to see the corrected line then he simply types ^R

```
.CRAET\TEA\EATE ^R
CREATE
```

Please note that underlining signifies characters entered by the user, and normal face signifies typeout by the computer.

(i) CONTROL-S (^S)

^S suspends typing of output to the terminal. This is particularly useful with a video screen display where output would otherwise disappear from the screen before it could be read.

Unlike ^O, output is not lost but is halted until a ^Q is typed.

(j) CONTROL-Q (^Q)

^Q restores the typeout mode for continuation of output after a ^S.

NOTE:
If the ^S and ^Q commmands do not work, type
.TTY PAGE<cr>
to enable their special functions. This command is described in Section 4.4.

(k) CONTROL-T (^T)

^T provides job status information without interrupting the execution of the current program, and can be issued either at user level or at monitor level. The information returned consists of

1. incremental day time (time since user last issued a ^T or, if this is the first time, since LOGIN time)
2. incremental run time (CPU time used since last ^T)
3. incremental disk reads (number of disk blocks read since last ^T)
4. incremental disk writes (number of disk blocks written since last ^T)
5. incremental cost (money used since last ^T)
6. program name
7. core size (low and high segment in K words)
8. job state
9. program counter.

Example:

After starting his program the user may wish to check its progress, i.e.

.RUN MYPROG then a little later

^T
DAY: :23:08 RUN: 3.34 RD:21 WR:6 $0.31 MYPROG 10+7K RN PC:405277

This indicates that since the last ^T the job had used 3.34 seconds of CPU time, 21 disk blocks had been read and 6 written and that the program was currently in the queue of jobs ready to recommence execution (Run state).

The USESTAT command can also be used to obtain the same information. It is useful in Batch control files but can only be issued at the monitor level.

(l) CONTROL-U (^U)

^U is typed to completely erase the current line entered by the user. If a RETURN has already been given, then this will not have the desired effect, since ^U only deletes the characters which have been typed in since the last carriage return/line feed. ^U results in a carriage return/line feed being typed back.

Note: After a ^U, the command symbol '.' should not be retyped, if this was on the line deleted. The '.' had been sent previously by the monitor and it is only user entered characters that are cancelled by ^U.

Example:

.CRAETE ^U
CREATE

(m) CONTROL-W (^W)

^W (Control-W) deletes the last word typed. A word is defined as all spaces, tabs, and alphanumeric characters until a nonalphanumeric character is typed. On video terminals, the deleted word is erased from the screen, while on hardcopy terminals it is printed backwards between backslashes. (^W is available on the KL10 only)

Eample:

CRAETE^W/ETEARC

(n) CONTROL-Z (^Z)

^Z acts as an end-of-file mark for terminal input. This allows the operating system to realise that the end-of-file has been encountered and to process the command using that file. It is not however the end-of-file mark under all cricumstances.

4.4 SETTING TERMINAL CHARACTERISTICS

Although the system sets default conditions for a terminal, various properties of the terminal may be altered by the user through the SET TTY monitor command (or TTY command). If however ALL terminal parameters are to be reset to standard values, type

```
I<cr>
```

at monitor level. This causes initialization of the terminal by running progam INITIA.

The most common TTY commands are outlined below with a comprehensive list given on pages 2-221 to 2-226 of the Operating System Command Manual.

(a) TTY NO BLANKS

Suppresses blank lines, with form feeds and vertical tabs output as a single blank line.

TTY BLANKS restores spaced output.

(b) TTY CRLF

A carriage return is output at the end of a line exceeding the carriage width (default condition).

TTY NO CRLF suppresses this carriage return.

(c) TTY FILL n (where n=0, 1, 2, or 3)

supplies filler characters at the end of a line. Filler characters slow down the terminal and enables output of characters missing or overprinted at the beginning of each line.

TTY FILL 0 supplies no extra filler characters.

(d) TTY NO FORM

outputs eight line feeds for a FORM FEED and four line feeds for a Vertical Tab (VT) to compensate software-wise for hardware non-options.

TTY FORM indicates that the terminal has hardware FORM (page) and vertical tab characters.

(e) TTY GAG

Prevents any messages transmitted by the SEND command from being received at this terminal unless the terminal is in monitor mode. This command does not affect messages sent by the system operator.

TTY NO GAG cancels the GAG option.

(f) TTY LC

sets the terminal so that the user can enter text in both upper and lower case.

TTY UC causes the monitor to translate lower case characters to upper case.

(g) TTY PAGE

allows the user to suspend current printing at the terminal without losing any output (through use of ^S and ^Q commands outlined above.)

TTY NO PAGE disables the special ^S and ^Q functions.

(h) TTY NO TAB

indicates that the monitor is to simulate TAB output by sending the necessary number of space characters.

TTY TAB indicates that the terminal has hardware TAB stops.

(i) TTY TAPE

allows the user to suspend the reading of paper tape at the terminal. ^S causes the terminal to stop reading paper tape and ^Q causes reading to continue.

TTY NO TAPE disables the special ^S and ^Q paper tape functions.

(j) TTY WIDTH nn

sets the default carriage width in characters on the terminal, where nn is usually set to 80.
By typing 'I TTY <CR>', your current terminal setting will be displayed.

On the KL-10 by typing 'HELP TTYTYP<CR>' you obtain a list of terminal settings which will best suit the type of terminal you are using.

CHAPTER 5

PDP-10 INTERACTIVE PROCESSING

5.1 Introduction

Interactive processing requires in part a dialogue between the user and the system Monitor in the form of commands issued by the user and responses and requests from the Monitor. The following section introduces the user to some of the simplest commands.

The complete set of commands is described in the following publications which may be purchased, or referred to, in the service areas of the remote job entry stations.

DECsystem-10 Operating System Command Manual
Operating System Commands - Reference Card

5.2 Getting onto the System (Logging in)

The user must establish communication with the system before doing any processing. This is apart from the normal SWITCHING ON procedures described in the previous chapter. Similarly, the logout procedure must be completed on termination of processing (Refer to section 5.6 - "Getting off the System").

This process by which the user identifies himself to the PDP-10 is called LOGGING IN. Logging in performs a number of vital functions and the system will refuse to accept commands until the login procedure has been successfully completed.

The work done in the period between entering and leaving the system is called a JOB. When a user logs in, he is first allocated a job number (if one is available). The computer then requests the user's project,programmer number, password, and cost limit. The Project,Programmer number is the standard identification code and is assigned to each new user by the Computer Centre. It consists of two integers, separated by a comma, the first being the project number and the second, the programmer number.

The COST LIMIT is the maximum expenditure which the user wishes to incur on the job which he is about to initiate. (If the allocated cost limit is exceeded during the job, an error message will be given and no further processing may occur until the cost limit has been increased. This cost limit may be increased at any stage, even if the current cost limit has not been exceeded - but it cannot exceed the current account balance.)

PASSWORDS (up to 6 characters) are established by the user at the Computer Centre when an account is opened. A password may be changed by the user from a remote terminal at login time (see section 5.7 for details). This facility ensures that no unauthorised use of an account may take place.

The password given is compared with the one known to the system and the cost limit is checked against the balance of that account. If all checks are satisfactory, the user is logged in and allowed to continue.

Various details of system status, planned operating schedule, special messages and so on, also appear in the LOGIN dialogue given below.

Note: In all examples of terminal output, actual typein by the user will be underlined to distinguish it from typeout of the computer. <cr> indicates that the user must press the RETURN key which causes a carriage return and line feed to be echoed, and signals the computer to act on the line just entered on the terminal.

Example:

.LOGIN 175,171<cr>	this causes the LOGIN program to be read into core, to read your ppn and to take control of your terminal.
JOB 25 Prentice 10603A-SLOTS #16 TTY11 Node 6 Line 11 or JOB 25 Prentice KL 701 #3 TTY33 Node 10 Line 106	the system (either KA or KL) has assigned job 25 to the user. It also gives the monitor name and version, and the terminal, node, and line numbers.
Password:	The LOGIN program then requests the password which is not echoed (for security reasons) and is terminated with a carriage return.
A/c balance is $166.21 [Updated 18:57:19 09-OCT-79]	If the password is recognized then the account balance and the date of last update are printed.
Charge no. 20 Cost Limit: $2.00<cr>	The cost limit for the job is requested and should be entered and terminated by a carriage return.
Seq. no. 102622	If it is acceptable, some more general information, including
15:58:49 10-Oct-79 Wed	the time and date are typed on the terminal, and the user will be logged in.

```
UNTIL FRIDAY 8TH NOV BOTH
THE KA AND KL SYSTEMS WILL BE
AVAILABLE UNTIL MIDNIGHT
(message of the day from Computer Centre)

.
```

Note the period to indicate the monitor is ready.

The CHARGE NUMBER output is the particular account to which the cost is to be charged.

The SEQUENCE NUMBER is a unique integer number associated with this particular job.

If the password entered does not match the password stored in the system, the following response will occur:

```
%LGNIET Invalid Entry - Try Again
.KJOB
```

and the user must restart the login procedure.

There are various abbreviated forms of logging in which the user will learn with experience. These are useful for convenience only.

Once logged in, the user has the capability to perform processing as required. Such processing may include storage of new information, alteration of old information, execution of programs, and so on. These functions are outlined in the remainder of this chapter.

To maintain security a user should change his password regularly. This is done during the LOGIN process as shown in section 5.7.

5.2.1 Mail

Anyone with access to a terminal connected to the computer network can send messages to anyone or a number of other persons with similar access. Messages could take the form of memos, letter, notices, etc., or any other information you would type and dispatch via the internal mail.

Mail is a store and forward system; users enter messages at their terminals and MAIL stores the messages until the intended recipients login their terminals, at which point they are presented with a list of waiting messages.

In its simplest form MAIL provides a method of posting a letter to another person, but with instantaneous delivery. In addition, MAIL provides facilities for group mailing, timed messages, acknowledgements of receipt, automated replies, filing, etc..

This is only a brief introduction to MAIL, for full details of the available

features you are referred to the document MAIL, which is available by TYPE'ing or PRINT'ing the file DOC:MAIL.DOC

5.3 File Manipulation

File manipulation involves creation of a new file, changes to contents of an existing file, deletion of a file, printing the contents of a file, and performing various housekeeping functions associated with the files.

5.3.1 File Creating, Updating, and Superseding

A file is CREATED if no file of the same name exists in the user's directory at the time of creation. Clearly, the fact that files are referred to by name precludes having multiple files with the same name. Once created, the file can then be UPDATED if changes are required.

The University of Queensland currently supports a number of editors, each of which can either create or update a file.

The QEDIT is a line editing program which enables the user to create and make changes to most types of ASCII character files. It has a simple command structure and is more widely recommended for simple usage.

Reference:

MNT-6 "QEDIT-A LINE EDITOR FOR THE PDP-10"

SOS editor is available on both the KA and KL. It supports line numbers or current line editing in either input, edit or alter mode, and at either the expert or novice level. For a complete list of commands type 'HELP SOS<cr>'.
(For reference:- SOS reference manual)
[NOTE when edit is used in the 1022
Data Managment System, SOS is invoked]

TECO on the other hand is a more powerful character editor which is adequate for all editing. It can perform programmed editing, and after learning TECO the user can edit quickly with a minimum of keystrokes on the keyboard. It should be realised however that through TECO it is much easier for the user to completely destroy the contents of a file.

References:
MNT-16 "TECO EDITING - A TUTORIAL COURSE FOR PDP-10 AND

PDP-11 USERS"
DECsystem-10 TECO
DECsystem-10 INTRODUCTION TO TECO

If a file with the same name exists at the time of writing, the latest file is said to SUPERSEDE the previous version. If the new file has been created with the QEDIT or SOS (or TECO using the Enable Backup command), then the previous version is kept for backup or safety, while the new file is stored on another part of the user's disk area. Any reference to the filename will refer to the latest copy. For example, if SORTRN.FOR already existed on the disk at the same time as another file SORTRN.FOR was created then the old version would become SORTRN.BAK and the new version would remain SORTRN.FOR.

An example of the creation of a file with the editor is given below. (The file is typed in just as it would appear on punched cards).

```
.CREATE TEST1.FOR<cr>
INPUT:
C<cr>
C     THIS IS A PROGRAM TO READ IN TWO INTEGERS<cr>
C     AND FIND THEIR PRODUCT<cr>
C<cr>
      READ(5,10)NUM1,NUM2<cr>
 10   FORMAT(2I3)<cr>
      IPROD=NUM1*NUM2<cr>
      WRITE(5,20)IPROD<cr>
 20   FORMAT(1H ,'PRODUCT = ',I7)<cr>
      STOP<cr>
      END<cr>
<cr>
*FILE<cr>
[EDIFIL Filed : TEST1.FOR]
.
```

Note that the double carriage return causes an asterisk to be printed by the editor. The user then types FILE and this causes the file to be written on his disk area.

5.3.2 File Directory

The user may, at any time, get a directory of his disk area. This directory contains information as to the names, size, creation date, and location of all his files. A directory can be obtained for the whole area or just for files specified, and can be obtained in a full or abbreviated form.

Below are some examples of obtaining a directory using the DIRECT command.

Example 1 - Full Directory

```
.DIRECT<cr>

INFO                1   <155>   13-Sep-79   DSKD:[60,105]
NAME       FOR      1   <055>   14-Sep-79
LAB1       DAT      1   <055>   14-Sep-79
LAB2       DAT      1   <055>   18-Sep-79
RLGND      FOR      1   <055>   18-Sep-79
RNORMD     FOR      1   <055>   19-Sep-79
NTEST      FOR      1   <055>   19-Sep-79
TEST1      FOR      1   <055>   19-Sep-79

Total of 21 blocks in 8 files on DSKD: [60,105]
```

where the detail line contains filename, extension, size of the file in blocks, protection (section 5.3.3), creation date, disk structure file is stored on, and project,programmer number. [Note that 1 block contains 128 PDP-10 words or 640 characters]

Example 2 - Fast Directory

```
.DIRECT/FAST<cr>

INFO                DSKE:   [60,105]
NAME       FOR
LAB1       DAT
LAB2       DAT
RLGND      FOR
RNORMD     FOR
TEST1      FOR
```

Example 3 - Directory of a particular file

```
.DIRECT TEST1.FOR<cr>

TEST1   FOR      1  <055>   19-Sep-79  DSKE:  [60,105]
```

Example 4 - Directory switches available

```
.DIRECT/HELP<cr>
```

will output a complete list of the switches available.

5.3.3 Protection

Three classes of user can access a file and so the owner of a file may give his file a level of protection pertaining to each class. The three classes are:

1. Owner of the file
2. Other programmers with the same project number
3. All other users.

The access protection of a file is indicated by three digits, the first digit pertaining to class 1, the second to class 2, and the third to class 3. The digits may be one of the following:

7	No access privileges for classes 2 and 3. The owner, however, may change the protection of any file.
6	Execute the file.
5	Read and execute the file.
4	Append, read, and execute the file.
3	Update, append, read, and execute the file. This protection code in the owner's field is identical to code 7, except that File Daemon(see section 5.3.3.1)is not called when a protection failure occurs.
2	Write, update, append, read, and execute the file.
1	Rename, write, update, append, read, and execute the file.
0	Change protection, rename, write, update, append, read, and execute the file.

Additional details on protection codes and File Daemon can be found in the DECsystem-10 Monitor Calls Manual.

The standard protection is <055> which means the owner has all privileges (0), and users in the owner's project and all other users can read and execute the file (5). Note that the owner of a file can alter the file's protection regardless of the existing protection code.

The user may change the protection on his files at any time by use of the PROTECT command.

```
.PROTECT TEST1.FOR<155><cr>
Files renamed:
TEST1.FOR
```

The protections for a number of files may be changed simultaneously by separating each with a comma.

5.3.3.1 FILE DAEMON -

File Daemon allows any user to specify who can and cannot access their files. Each user may create a file called ACCESS.USR. This file, at the discretion of the user, lists the names of some or all of that user's files and specifies, on an individual file basis, the users who can and cannot access those files. File Daemon examines the user's ACCESS.USR file, and if desired will record in a file called ACCESS.LOG the level at which users accessed files on your area.

If File Daemon finds ACCESS.USR but cannot find the accessed file name in ACCESS.USR, File Daemon denies access to that user. Also if File Daemon finds a file in ACCESS.USR, without the PPN of the user trying to gain access, access is again denied.

Please take note, if [1,2] jobs are denied access files will not be failsafed, and the users must make their own arrangement for failsafing their files.

For complete details on how this system works either print or type HLP:FILDAE.HLP.

5.3.4 Renaming and Copying Files

RENAME

The user may change the name of any of his files by using the RENAME command. The general form of the RENAME command is:

.RENAME newfile1=oldfile1,...,newfilen=oldfilen

Example:

```
.RENAME BEST.FOR=TEST1.FOR<cr>
Files renamed:
TEST1.FOR
```

In this example, file TEST1.FOR has been renamed BEST.FOR.

Files can be renamed from one PPN to another on the same structure. If the file needs to be transfered from one structure to another then COPY must be used (discussed below)

Example:

```
RENAME BEST.FOR[123,444]=BEST.FOR[123,555]<cr>
  FILES RENAMED
  BEST.FOR
```

In this example the name stays the same but if you wish to change the name, just specify desired name on the

output side (OUTPUT=INPUT).

COPY

A user may copy files from one location to another by use of the COPY command, and the existing file remains unaltered. This enables the user to duplicate a disk file from another device (e.g. magnetic tape), from another user area, or from his own area.

The general form of the COPY command is:

```
.COPY dev1:destination/switches=dev2:source,dev3:source,...
        file                    file 1       file 2
```

where devn signifies the device, for example:

DSK:	system disk
DSKB:	particular disk named DSKB
LPT:	line printer
MTA:	magnetic tape
TTY:	user's terminal

Examples:

(i) .COPY MINE.FOR=YOURS.FOR[160,105]<cr>
Copies file YOURS.FOR from disk area [160,105] onto user's area and names it MINE.FOR.

(ii) .COPY =MINE.FOR[175,101]<cr>
Copies file from area [175,101] onto user's area and gives it the same name.

(iii) .COPY DONE.FOR=MINE.FOR,YOURS.FOR[120,103]<cr>
Copies two files (one from the user's area, and one from area [120,103]), onto the user's area as one file and gives this new file the name DONE.FOR.

(iv) .COPY DSKE:MAIN.FOR=PRIV:CALC.FOR<cr>
Copies file CALC.FOR from the private pack (PRIV:) to disk (DSKE:) and names it MAIN.FOR.

Note: If the filename to be copied to, already exists as a file, then the original contents of the file will be overwritten during copying.

COPY Switches

Switches can be used in the COPY command string to alter information for the output file. Some of the more useful switches are:

/C delete trailing spaces and convert multiple spaces to tabs
/E ignore card sequence numbers (i.e. characters 73 to 80 on each line are replaced by spaces)
/N delete sequence numbers
/S insert sequence numbers (incremented by 10)
/O insert sequence numbers (incremented by 1)
/P prepare FORTRAN output for line printer listing
/T delete trailing blanks
/W convert tabs to spaces
/X keep copied files separate (default for COPY command).

Examples:

(i) .COPY NEW.FOR/T=MINE.FOR[175,101]<cr>
Copies file from area [175,101] onto user's area. The copied version has all trailing spaces removed and is named NEW.FOR.

(ii) .COPY =[175,101]F1.FOR,F2.FOR,F3.FOR<cr>
Copies several files from area [175,101] onto the user's area. The files are kept separate and have the corresponding names F1.FOR, F2.FOR, and F3.FOR.

Since COPY currently runs the system program PIP, the COPY and PIP switches are identical. A complete list of the available switches is given under the description of PIP on pages 145 to 167 of the DECsystem-10 Utilities Manual, or by typing 'HELP PIP <CR>'.

5.3.5 Deleting Files

A user may wish to clear his area of files which are no longer required. For example, it is generally not useful to keep backup files over long periods and so the system provides a means of getting rid of unwanted files.

The DELETE command provides this facility.

Example:

.DELETE TEST.FOR, *.BAK<cr>

will delete TEST.FOR and all files with a BAK extension from the user's area. If no file is found, a warning message is output

%No file named DSK:TEST.FOR

5.3.6 Obtaining Listings of Files

The user may print out the contents of his files on either the terminal or the line printer. To obtain listings on the terminal, the user should use the TYPE command.

Example:

.TYPE TEST1.FOR<cr>
will type the file TEST1.FOR on the terminal.

To type a file that uses FORTRAN carriage control characters on the terminal, the following command is used

.TYPE filename/FORTRAN<cr>

To have files listed on the line printer, the user should use a PRINT command. The following examples imply that the output is to be directed to the high-speed printer on the computer being used, unless the user terminal is connected to the Griffith or Commerce nodes. For details of how to direct output elsewhere refer to section 7.3.2 - "Location of Output".

Example:

.PRINT TEST1.FOR<cr>

With a timesharing system, a number of users may try to use the line printer at the same time. To regulate use, the PRINT command simply places a request for the file to be printed on a queue. The system then services these requests one at a time.

The user has a number of options which may be used when printing a file. These options or SWITCHES enable the user to specify such things as output format, number of copies and special types of paper.

Example:

.PRINT TEST1.FOR/COPIES:6/FORMS:2PART<cr>

will cause 6 copies of file TEST1.FOR to be printed on 2-part paper.

Reference:
"Operating System Command Manual" page 2-175 to 2-184 for the complete list and description of all switches available to the PRINT command.

Note: (i) Only ASCII files should be typed or printed. Binary relocatable and core image files should not be printed.

(ii) Files with DAT extensions are assumed to be output from FORTRAN programs, so the FORTRAN convention of interpreting and not printing the first character of each line as the carriage control for the line printer is followed. (This is not assumed when typing DAT files on the terminal).

Therefore, when printing files with extension .DAT which do not contain

carriage control characters, the switch /FILE:ASCII is used. This switch will cause the first character of each line to be printed out and NOT interpreted as a carriage control.

5.4 RUNNING A PROGRAM

In summary, the procedure for writing and running a program is as follows:

1. Create a source program (using QEDIT,SOS or TECO)
2. Compile the program using the appropriate compiler (COMPILE command)
3. Link and load the necessary routines into memory as a core image (LOAD command)
4. If required, save the core image to a disk file (SAVE command)
5. Execute the core image program (RUN, or GET and START commands).

5.4.1 Compilation and Execution

Once the source program has been placed on file, it can be compiled to produce a relocatable binary file and/or compilation listings for the specified source program. The COMPILE command is normally used for this processing.

If no switches appear in the command string, the following translators are used:

Source File Extension	Translator Used
.FOR	FORTRAN-10 compiler
.F10	FORTRAN-10 compiler
.ALG	ALGOL compiler
.CBL	COBOL compiler
.F4	FORTRAN-IV compiler
.MAC	MACRO assembler
other than above or null	Standard processor, which is FORTRAN-10

The source file is translated if there is no corresponding binary (.REL) file, or if the source file's date and time of creation is later than that of the binary file.

The general form of the COMPILE command is:

.COMPILE/switches string

where string is a single file specification or a string of file specifications separated by commas.

Some of the switches which may be used with the COMPILE command are:

/LIST Generates a compilation listing file on disk. On the KA system, this file has an .LPT extension and a default name generated by the system. The file will be printed and deleted when the user logs off the system, or if a PRINT command is given, eg. .PRINT *.LPT.
On the KL system, the listing file is queued directly to the line printer.

/COMPILE This switch will force a compilation of a source program, even if a binary relocatable file exists with a later or equal date/time to the source file.

/CREF Produces a cross-reference listing file on the disk for each file for later processing by the CREF program.

Reference:
DECsystem-10 Operating System Command Manual (pages 2-17 to 2-22)

Example:

Using the Fortran-10 program file created previously (now called BEST.FOR)

```
C
C     THIS IS A PROGRAM TO READ IN TWO INTEGERS
C     AND FIND THEIR PRODUCT
C
      READ(5,10)NUM1,NUM2
 10   FORMAD(2I3)
      IPROD=NUM1*NUM2
      WRITE(5,20)IPROD
 20   FORMAT(1H ,'PRODUCT = ',I7)
      STOP
      END
```

Note that FORMAT has been spelt incorrectly. This will be detected as a compilation error when the source file is compiled, i.e.

```
.COMPILE/LIST BEST.FOR<cr>
FORTRAN: BEST
00002 10     FORMAD(2I3)
?FTNMSP LINE:00002  STATEMENT NAME MISSPELLED

UNDEFINED LABELS
    10

?FTNFTL  MAIN.      2 FATAL ERRORS AND NO WARNINGS
.
```

Notice that only the error detected has been printed on the terminal. The /LIST switch has caused a complete compilation listing to be written to a disk file with an extension LPT (on KA system) or LST (on KL system). This may be printed on the line printer as follows:

```
.PRINT *.LPT<cr>
Total of 1 block in 1 file in LPT request
```

However, on the KL system the listing file would be automatically spooled to the line printer.

The user can then correct the error using the Editor and re-compile:

```
.COMPILE/LIST BEST.FOR<cr>
FORTRAN:  BEST.FOR

MAIN.
.
```

The program has successfully been compiled, producing a binary relocatable file called BEST.REL, which may be executed.

The EXECUTE command causes the specified .REL file (or files) to be loaded into core and started.

```
.EXECUTE BEST.REL<cr>
LINK:   Loading
[LNKXCT BEST Execution]
123 40
PRODUCT =   4920

STOP
END OF EXECUTION
CPU TIME: 0.08  ELAPSED TIME: 20.46
Exit
```

Once [LNKXCT BEST Execution] has been typed back, the computer is ready to accept input data. The user then types the input data and gives a carriage return.

NOTE: If a .REL file of an equal or later date or time does not exist for the source file, then the EXECUTE command will cause the source file to be automatically compiled before execution is attempted.

5.4.2 Saving a Core Image

The EXECUTE command causes the binary relocatable file to be loaded into core and the execution started.

These two functions can, however, be separated. The LOAD command runs the loader and loads the prescribed .REL file(s) into core.

```
.LOAD  BEST.REL<cr>
LINK:    Loading

Exit

.
```

The program is now in core and needs to be started. This can be achieved by using a START command:

```
.START<cr>
```

It is possible to keep a core image of a file which has been loaded into core. This is achieved by use of a SAVE command.

The SAVE command writes out a core image of the file currently in core.

Example:

After loading BEST.REL into core, instead of starting it, a core image could have been obtained by typing:

```
.SAVE  BEST<cr>
BEST saved
```

This causes the core image BEST.EXE to be written onto the user's area. (Core image files are given an extension EXE.) The advantage of a core image file is that it does not need to go through the same loading process as binary relocatable files before execution and so is ideal for storing frequently used programs.

5.4.3 Running a Core Image File

Core image files may be executed by means of a RUN command. The RUN command obtains a core image file from a retrievable storage device and sets it into execution.

Example:

```
.RUN BEST<cr>
```

If a filename is not specified, the default is taken to be the job's current name as set by the last RUN or SAVE command.

NOTE: The RUN command should not be confused with the R command, which obtains a core image from the System Area and starts it into execution.

Example:

```
.R PIP<cr>
*
```

5.4.4 Interrupting Execution

The execution of most programs may be interrupted at any time by typing two control-C's. This causes control to be returned to the monitor.

CONTINUE

Provided the user's core is not disturbed, the program can be continued from the point at which it was interrupted by using the CONTINUE command.

This facility enables the user to check his cost limit and reset it if necessary, without destroying the integrity of his program.

Example:

```
.RUN BEST<cr>
^C
^C                              stop the program
.COST<cr>                  check the job cost
Lim $1.00
Use $0.95
Bal $24.05
.SET COST $2.00<cr>  increase the cost limit to allow job to finish
.CONT<cr>                  continue the program
```

Similarly, if the program has been halted by exceeding the cost limit, it can be CONTINUEd after the cost limit has been increased.

Example:

```
.RUN BEST<cr>
?Cost Limit Exceeded        Insufficient money allocated by user
.SET COST +$2<cr>           Increase the cost limit
.CONTINUE<cr>               Continue execution of the program
```

CLOSE

After execution has been interrupted, the user may wish to save the files processed rather than continuing. For example, if execution was halted by a fatal error (excluding "?Cost Limit Exceeded"), then closing the files may help to determine the amount of processing performed and so isolate the cause of error. The CLOSE command terminates any input or output currently in progress on the specified device and closes the files. Logical names and device assignments are preserved.

The general command format is:

```
.CLOSE dev<cr>
```

where the argument 'dev' is optional and if it is omitted I/O is terminated on ALL devices and all files are closed.

5.5 Miscellaneous Useful Commands

The following commands can be used to determine or reset various parameters relevant to the user's job.

5.5.1 COST and SET COST Commands

The COST command is used to check the job cost. It does not destroy the user's core image (thus the job can be continued 'on the KL only').

For SLOTS users on the KA there is a maximum amount of $3-00 allowed in any one login, which cannot be altered.

The SET COST command is used to reset the job cost limit. This command can be used in two ways:

(i) .SET COST $15.00
sets the cost limit to $15.00. This does not make a new $15.00 available, but merely adjusts the old cost limit to be $15.00.

(ii) .SET COST +$10.00
adds $10.00 to the amount used to obtain the new cost limit. Consequently, if an old cost limit of $10.00 had expired, then the new cost limit would be $20.00. If only $4.00 of the original cost limit of $10.00 had been used, the new cost limit would be $14.00.

5.5.2 TIME Command

The TIME command causes the total runtime since the last TIME command to be typed out, followed by the total runtime since the job was logged in, followed by the integrated product of runtime and core size (KILO-CORE-SEC).

Example:

```
.TIME<cr>
4.20
4.20
kilo-core-sec = 58
.
```

5.5.3 DAYTIME Command

This command causes the day, followed by the time, to be typed out.

Example:

```
.DAYTIME<cr>
14-SEP-79  12:36:34
```

5.5.4 PLEASE Command

The PLEASE command enables the user to communicate with the operator. The user should type PLEASE followed by the text, and the operator will reply at a later stage.

When the message is terminated by pressing either the CARRIAGE RETURN key or the ALTMODE key, the line is sent to the operator. The terminal is left in monitor mode and the user may continue processing. The operator will respond later.

Example:

```
.PLEASE WHEN CAN I MOUNT RP03 PACK PRIV:<esc>
[ Operator has been notified]
.EDIT TEST.FOR<cr>
.
.
.
*FILE<cr>
.;;OPR - IN ABOUT HALF AN HOUR(the operators reply)
```

5.5.5 SEND Command

The SEND command enables communication with a user at another terminal connected to the same computer system. It therefore allows a line of information to be sent from one terminal to another. Identification of the terminal transmitting the information is also sent.

Command formats:

```
.SEND (destination terminal) (message)<cr>
.SEND JOB (destination job) (message)<cr>
```

Examples:

.SEND TTY12 test<cr>
.SEND JOB 25 test<cr> sends message to job number 25.

NOTE that the SEND command is not available to the BATCH user.

5.5.6 PJOB Command

The PJOB command causes output of information about the job to which the user's terminal is attached. It is therefore used to determine if there is a job running on the particular terminal without destroying the job's core image.

Examples:

.PJOB<cr>
JOB 60 User Smith [170,105] TTY20

.PJOB<cr>
?Not a job Indicates that the terminal is not being used.

5.5.7 SYSTAT Command

The SYSTAT command provides output of the present status of the relevant PDP-10 system. Such output provides the TTY number associated with each job (useful for the SEND command). Various switches are associated with the command, a comprehensive list being given in the DECsystem-10 Operating System Command Manual on pages 2-235 to 2-236, or by typing 'SYS/H<CR>'

Examples:

.SYSTAT<cr> list entire systat output - includes all jobs running and all file structures.
.SYSTAT/G<cr> general system status - used to determine the current job load on the system.
.SYSTAT/S<cr> output short job status - includes jobs running.

The SYSTAT command can also be used to obtain the status of individual jobs, e.g. the jobs running under a particular PPN and the devices under control of that PPN.

Examples:

.SYS [170,105]<cr> lists all jobs logged in under [170,105] and includes any devices (eg. magnetic tape) that the user may currently have mounted.

.SYSTAT 25<cr>	causes information to be listed only for job number 25.
.SYSTAT .<cr>	causes information to be listed only for the user's job.

5.6 Getting off the System (Logging out)

Once the user has completed whatever work was to be performed, he must indicate to the system that he wishes to cease operation. This action is called LOGGING OFF and is achieved by use of the KJOB command.

On the KL system, the KJOB command format is

.KJOB/switch<cr>

where switch can be one of the following:

/FAST	Log out and save all files. Unpreserved temporary files (nnn???.TMP) are deleted. This is the default action on the KL.
/BATCH	Delete no files except if over logged out quota. The files are deleted in the order given in section 6.7 under the "List files" subheading.
/HELP	List the KJOB options
/NOMESSAGE	Suppress all output to the terminal when the job is killed.

Example:

```
.KJOB<cr>
Job 20  User SMITH [107,102]
Logged-off TTY21 at 14:01:02 on 28-Sep-79
Cost $0.10 [Excluding spooled I/O & MOUNT charges ]
Runtime: 0:00:00, KCS:4, Connect time: 0:01:15
Disk Reads: 58, Writes: 1, Blocks Saved: 965
```

On the KA system, the KJOB command formats are:

.KJOB/switch<cr>

where the available switches are /FAST, /BATCH, and /HELP (as above);
or

.KJOB<cr>
Confirm: the user must then type one of the letters described below or else type ^C to abort the logout.

F logout immediately saving all files
K delete all unprotected files, i.e. all files having a <Onn>

protection
P protect and save all files except temporaries (.TMP, .CRF, .LST)
I individually determine the fate of each file.
After each file name, respond:
S to save the file
P to protect the file
K to kill the file

Example:

```
.KJOB<cr>
Confirm: I
(K=DELETE !!)
DSKD:
TEST4    .TST    <055>    200.    BLKS    : K<cr>
```

Each user has a limited amount of space on the public disk structures called a LOGGED-OUT QUOTA. The amount of disk space used for file storage must be within this quota before a session can end. If a user exceeds the logged-out quota during the session, he is prevented from logging out until some of the files have been deleted. Consequently, if the message

```
?Logged out quota exceeded
Confirm:
```

appears while trying to log out, the user should type ^C and delete relevant files until under quota. Then again type

```
.K/F<cr>
```

to log out. If the user does not wish to delete the over-quota files, they can be ARCHIVED (on the KL system), but then they must be RETRIEVED before use (Refer to Chapter 8 - File Migration System). This retrieval of files does however take on average 2-3 hours to process.

5.6.1 QUOLST Program

The QUOLST program enables the user to determine whether or not he has exceeded the logged-out quota. It informs the user of both the amount of disk space he has used and the amount that remains. This program also returns the amount of free space that the system has left for all users of the structure. Although the DIR/ALLOC/SUMM command can also be used to obtain this information, use of this program is the cheaper alternative.

The output given for each file structure consists of:

1. structure name

2. number of blocks allocated, and

3. number of blocks left in the logged-in quota, logged-out quota, and on the structure.

Example:

```
.R QUOLST<cr>
USER  111,222
STR  USED     LEFT:(IN)     (OUT)     (SYS)
DSKC: 110        900          -10        4703
PRVA: 0        10000         5000         396
```

This indicates that the user is over quota on DSKC: (-10 blocks) and must therefore delete files before he can logout,or ARCHIVE, if logged in on the KL. The total allocated space for the file deleted must be at least 10 blocks in this example so that the number of blocks left in the logged-out quota will be equal to or greater than zero.

5.6.2 Nurse

Nurse takes care of the system resources, job use and disk space.

Disk space is checked every five minutes. If free space is below a predetermined level users of the disk who are logged in will get the message 'NURSIS Space is short on DSKx:. Please delete unwanted files' every ten minutes. All jobs that have used more than half their logged in quota are told 'NURMJC You are a major consumer of DSKx:, Delete unwanted files' every five minutes and the centre is told who has been notified.

Nurse can also watch job usage. Any job that is inactive for five minutes is warned 'NURWRN Please use your terminal or KJOB'. After ten minutes inactive jobs may be logged off forcibly. Please note that you may lose files if you are over your logout quota so it pays to heed the warnings.

5.7 Changing a Password

Users are able to change their passwords directly by use of the /PASSWORD switch on the login command. The system requests the old password and the new password. These are typed in by the user, but are not echoed back.

Example:

```
.LOGIN [173,106]/PASSWORD<cr>
JOB 25 Prentice KL 701 #3 TTY14 Node10 Line 106
PASSWORD:          enter old password terminated by CARRIAGE RETURN
NEW PASSWORD:      enter new password (up to 6 characters) terminated
                   by carriage return
```

. new password has been accepted

The new password has been accepted and will be in effect for the next login. For security reasons, passwords should be changed from time to time.

CHAPTER 6

BATCH PROCESSING

Multi-program batch consists of a group of programs which allow a user to submit a job for processing at some later stage. Normally, batch jobs are submitted from a remote terminal (Section 6.3), though jobs may be input to batch via decks of punched cards (Section 6.2). The time lapse between entering a job for processing and receiving the output depends, firstly, on any time requirements specified by the user and, secondly, on the number of other jobs to be processed at the time.

Batch processing is ideal for jobs which do not require a fast response, jobs which do not require human interaction, jobs that are frequently run, jobs that are large and long running, and jobs that require large amounts of input data.

Because processing charge rates vary during the day, the batch system has the added advantage of allowing a user to leave his job to be processed at a time when maximum benefit can be obtained from lower charge rates.

Output is never returned to a remote terminal, even if the job was submitted from one. Normally hard-copy output is produced on the line printer at the Computer Centre, or at the node remote station.

For a more detailed description of BATCH PROCESSING, the user is referred to the DECsystem-10 publication Beginner's Guide to Multiprogram Batch.

6.1 Batch Components

In order to understand the operation of the batch system on the PDP-10 systems, it is useful to know something of its actual components.

6.1.1 The Stacker

The card stacker program (also called the input spooler) is the first stage in the processing of a batch job. As cards are read through the card reader they are scanned to identify control cards. It has several functions and creates at this stage:

(i) the user's data files and program files
(ii) the batch control file
(iii) the user's log file
(iv) a queued request to the batch controller

The data files are created according to the control cards and are written to the user's disk area. Programs and data are copied into these files and are later passed to the user's job while it is running under the batch controller program.

A batch control file is created for each valid job and is subsequently processed by the batch controller. This file contains all the monitor, batch and user level commands encountered in the input. The Stacker also inserts any commands generated from control cards into the control file on the user's disk area.

The log file contains details of the whole processing of the job, including any operator intervention. The log file exists in the user's disk area until it is printed at logout. This gives the user a trace of all functions which occurred during his job and is very useful in debugging.

Finally, if a control file has been created and the job needs to be logged in, the Stacker writes a request in the batch input queue.

It is important for the user to distinguish correctly between BATCH control cards ($ sign in column 1) and MONITOR commands (. in column 1), since all control cards are processed by the Stacker before BATCON initiates the job.

6.1.2 The Batch Controller

The batch controller, BATCON, controls the running of all jobs entered into the batch system, by reading the control file created by the Stacker or by the user and initiating and controlling the running of the job by passing data and system program commands directly to it. Monitor commands (those prefixed by a '.') are passed to the monitor for action. The controller determines the destination of commands by interpreting the first character on each line of the control file. If column 1 contains an alphabetic or a numeric character, the line is either a monitor or user command level. If the first column contains a special character, the batch controller interprets the line as follows:

(a) $ (dollar sign)

Since the $ control cards have been processed by the stacker, they are copied to the log file as comments. To pass a line beginning with $ to the job, precede it with an asterisk (*) in column 1.

(b) . (period)

The interpretation depends on the character in the second column.

(i) If the character in column 2 is an alphabetic, the line is treated as a monitor or batch command and the period is suppressed. If the user job is waiting for input a single ^C is sent to put it into monitor mode.

(ii) If the character in column 2 is non-alphabetic, the line is treated as data, with the period as part of the data.

(c) * (asterisk)

The line is treated as a user level command and the asterisk is suppressed. If the job is at monitor level the line is ignored to facilitate error recovery.

(d) ; (semicolon)

The line is treated as a comment and is entered into the log file.

6.2 Entering a Batch Job via Cards

6.2.1 Preparing Punched Cards

The card reader on the PDP-10 accepts normal 80 column punched cards. These must be prepared using IBM 029 card punches.

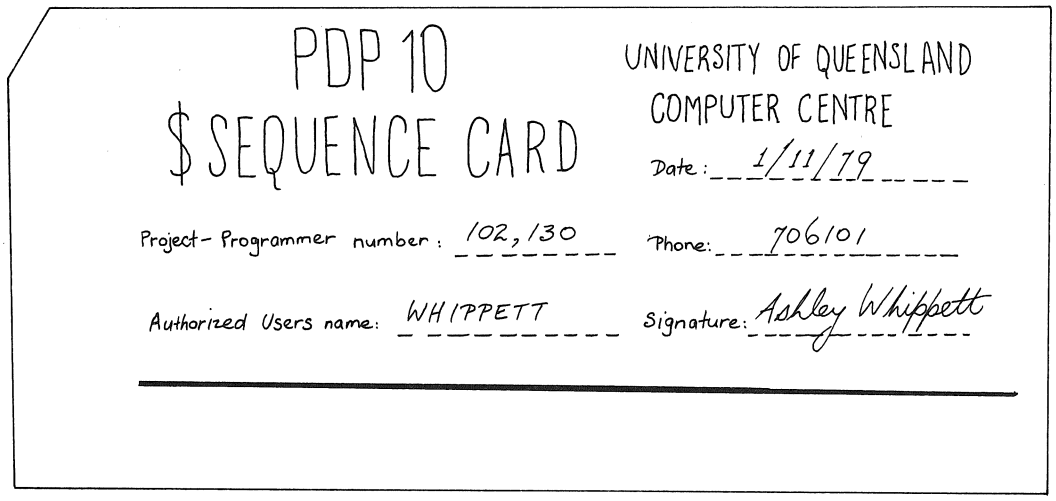
6.2.2 Essential Control Cards

With Batch, the job comprises a deck of cards bounded by control cards that mark its beginning and end. Other control cards can be interspersed among the cards in the job deck to control processing.

Normal monitor commands (see DECsystem-10 Operating System Commands Manual) can be used in a batch job, except for SET TIME, SET TTY, and SEND.

The following cards are essential to any batch job:

(i) $SEQUENCE - This card is used to identify the user, to prevent unauthorized use of the account, and also to identify the particular job to the system. These cards are preprinted and are available at the batch submission point at the Hawken Building remote job entry station.


PDP 10
$ SEQUENCE CARD

UNIVERSITY OF QUEENSLAND
COMPUTER CENTRE

Date: 1/11/79

Project-Programmer number: 102,130

Phone: 706101

Authorized Users name: WHIPPETT

Signature: Ashley Whippett

(ii) $JOB - This is the command which causes the stacker to create a control file and a log file on disk into which commands are placed for the batch controller.

The general format of the card is:

```
$JOB name [proj,prog]/S /S .../S
```

where

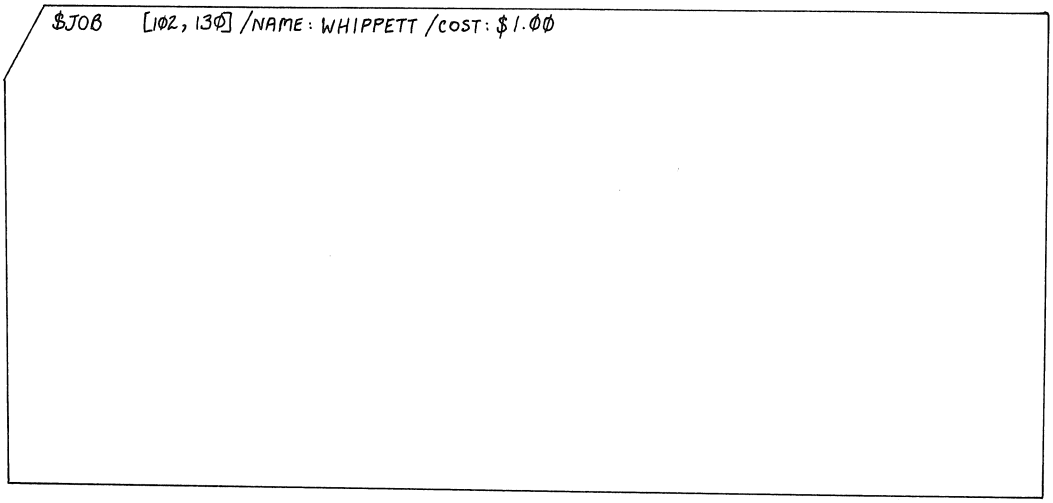
name	- user assigned name for the job. If omitted, the stacker creates its own unique name for the job. This is the exercise name for student jobs.
[proj,prog]	- is the project,programmer number of the user (enclosed in square or round brackets). This is mandatory.
/S /S .../S	- are switches some of which are defined below. The only mandatory switch is /NAME.

A complete list of switches available to the $JOB card is given on pages 2-27 to 2-29 of the DECsystem-10 Beginner's Guide to Multiprogram Batch.

Switches

(a) /NAME:aa - the user name up to 12 characters long.
(b) /COST:cost - cost limit for this job. If not specified, default is $1.00
(c) /PRIORITY:n - the normal priority is 10, however lower priorities attract discount rates - Refer to MNT-1 for a discussion of the charge rates.
(d) /TIME:hh:mm:ss - the limit on the amount of CPU time for the job.

An example of a typical $JOB card is:

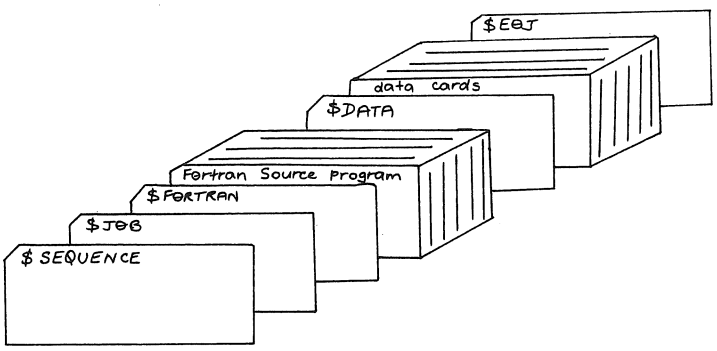


(iii) $EOJ CARD - This card should always be the last card of any batch card deck. It terminates the entire user job and if it is not the last card in the deck, will cause termination of processing at the point it is encountered.

These cards are available at the Hawken Batch station.

6.2.3 Compiling and Executing a Program

A simple Compile and Execute batch job would have the following deck setup:




```
$SEQUENCE
$JOB [102,130] /NAME:WHIPPETT/COST:$3.00
$FORTRAN
.
.    (Fortran source program)
.
$DATA
.
.       (data)
.
$EOJ
```

Two new cards appear here:

(i) $FORTRAN - this card indicates to the stacker that the following deck of cards comprises a FORTRAN-10 source program. The stacker then reads the cards into a file (until it encounters a control card), and assigns its own default name with an extension FOR. It then generates the appropriate monitor commands to cause a Fortran compilation (producing a .REL file with the same name) and writes these commands to the user's command and log files. (A ".COMPILE/LIST filename.FOR" is in fact generated by the $FORTRAN card).

If a file already exists on the disk, it can be compiled by specifying the filename instead of following the $FORTRAN card with the source cards, for example

```
$FORTRAN filename
```

This also applies to other languages when submitted as cards for batch processing,

i.e. $ALGOL
$MACRO
$COBOL

Switches which may be used with the $language card are /SUPPRESS, /NOLIST, and /CREF (as given in Section 2.4 of the Beginner's Guide to Multiprogram Batch).

(ii) $DATA - this card causes the stacker to copy data into a file on the user's disk area, and to generate a .EXECUTE command which it writes to the control file. The effect to the user is that the $DATA card causes the relocatable file of the preceding compilation to be loaded and executed. This card applies to the running of programs written in any language. For details of switches which may be used with the $DATA card, see pages 2-14 to 2-16 of the DECsystem-10 Beginner's Guide to Multiprogram Batch.

The user has the option of specifying a file name. If this is not done, the stacker uses a default filename of three characters with a .CDR extension, and inserts a .SET CDR command to ensure that input from CDR: will read this file.

WARNING:
It is important to note that the $EOJ card generates commands to delete any files which may have been created by the Stacker (unless the protection has been changed by the user). The user must also ensure that his logged-out quota will not be exceeded at logout, because if it is, he is not given the option of deleting files of his own choice. The system will delete files until it gets the user's disk space below the logged-out quota. The order in which files are selected for deletion under MPB (KA system) and under GALAXY (KL system) is outlined in Section 6.7 under "List files".

6.2.4 Copying a Deck of Cards to a Disk File

A user may wish to copy a deck of cards to a disk file, perhaps so that it may be manipulated from a remote terminal.

A simple deck setup to perform this task would be:

```
$SEQUENCE
$JOB [102,130] /NAME:WHIPPETT
$DECK  TEST1.FOR
.
.     (card file)
.
$EOJ
```

The $DECK card causes the cards up to the next control card to be read onto a disk file with the specified name. Because no Batch job is run, specification of the cost limit is not required.

Switches applying to the $DECK card are given on pages 2-20 to 2-21 of the Beginner's Guide to Multiprogram Batch.

6.2.5 Running a Program on Disk with Data on Cards

The user may wish to keep a frequently used program on disk and then run it with data on cards. This is achieved using either of the following deck structures:

```
$SEQUENCE
$JOB [102,130] /NAME:WHIPPET/COST:$2.00
$DATA
.
.
.
$TOPS10
.RUN TEST1          file TEST1.EXE is stored on the user area
$EOJ
```

The $TOPS10 card directs Batch to copy all the following cards up to the next control card ($ in column 1), into the Batch Control file as commands for processing. Therefore, monitor and/or batch commands MUST be preceded by a $TOPS10 card.

If however the file TEST1.REL and not TEST.EXE is stored on the user area, then use

```
$SEQUENCE
$JOB [102,130] /NAME:WHIPPETT/COST:$2.50
$INCLUDE TEST1.REL  with file TEST1.REL on user area
$DATA
   .
   .
   .
$EOJ
```

The $INCLUDE card directs Batch to include the specified file in the loading process. Then in this example, the $DATA card causes an EXECUTE TEST1 monitor command to be generated.

In both cases, the $DATA card causes the Stacker to use a random nnn.CDR filename, generate a .SET CDR nnn and ensure deletion of the nnn.CDR file before log-out. The SET CDR command sets the filename for the next card reader spooling intercept and causes the data on cards to be used as input to the program.

6.2.6 Further Examples

Several examples illustrating use of the control cards discussed previously are outlined below.

Example 1 - using IMSL statistical routines

```
$SEQUENCE
$JOB [102,130] /NAME:WHIPPETT/COST:$3.00
$FORTRAN            default file name is provided by the stacker
  .
  .
  .
$INCLUDE STA:IMSL/SEA
$DATA                generates a  .EXECUTE name,STA:IMSL/SEA command
  .
  .
  .
$EOJ
```

Example 2 - using Fortran plotting routines

```
$SEQUENCE
$JOB [102,130] /NAME:WHIPPETT/COST:$2.50
$INCLUDE TEST1
$INCLUDE PLO:CALF10/SEA
$DATA               generates a .EXECUTE TEST1,PLO:CALF10/SEA command
  .
  .
  .
$EOJ
```

Example 3 - running a program requiring the data file TEST.DAT

```
$SEQUENCE
$JOB [102,130] /NAME:WHIPPETT/COST:$2.00
$DECK TEST.DAT      copy cards up to next control card into the file
  .                  TEST.DAT
  .
  .
$TOPS10
.RUN TEST3          TEST.DAT is used by program TEST3
.DELETE TEST.DAT    file is no longer required
$EOJ
```

6.3 Entering a Batch Job from a Remote Terminal

When entering a batch job from a remote terminal, the user must create a control file that the batch control program can use to run the job. In effect, the user must usurp the role of the stacker in creating a control file for the batch control program, as is done when entering the job via cards. The control file contains all of the commands which the user would employ if running the job from a terminal, and any batch commands and labels.

Example:

If you wish to compile and execute a FORTRAN program, the control file used might appear as:

```
.COMPILE TEST1.FOR
.EXECUTE TEST1.REL
```

When the job is run, the commands are passed to the monitor to be executed. The commands and the replies from the monitor are written in the log file.

The following procedure should be observed when creating a control file and submitting it to batch from a terminal.

1. LOGIN to the system as an interactive user.
2. Create a control file using the editor. It is usual to use .CTL as the extension.
3. Submit the job to batch using the monitor command SUBMIT.

6.3.1 Creating the Control File

The control file can contain monitor commands, system program commands, data that would normally be entered from a terminal, and special batch commands, labels and comments.

What the user actually writes in the control file depends on what the job is to accomplish. Monitor commands and input data are entered into the control file in a line-by-line correspondence with the sequence that would be typed at the terminal. The Batch controller then recognizes what to do with each line in the control file by looking at the first character, ie.

'.'	as the first character indicates a monitor command;
'*'	indicates data to be passed to a program being run from the control file; and
'!'	indicates a comment line which can be positioned anywhere in the control file.

An example of a job that you can enter to Batch from a terminal is as follows:

1. Compile a program that is on disk.
2. Load and execute the program. Data from the disk file DATA.DAT is required.
3. Print the output on the line printer.

4. Write the output into a disk file also.

5. Compile a second program.

6. Load and execute the second program with the data output from the first program.

7. Print the output from the second program.

The control file that you would create for the above job could contain the following commands:

```
.EXECUTE MYPROG.FOR/COMPILE
!data is obtained from file DATA.DAT (comment line)
*DATA.DAT
.EXECUTE PROG2.FOR/COMPILE
```

Statements to write the output to the printer and the disk have been included in the programs. The output to the line printer is printed with the log file as part of the total output of your job.

To avoid having a job terminated because an error occurs, the user can specify error recovery in the control file using the special Batch commands. Error recover is described below in Section 6.6, and also in Section 3.5 of the Beginner's Guide to Multiprogram Batch.

Similarly, in normal processing the lines of the control file are read sequentially, but occasions do arise when it is necessary to jump to lines out of sequence. The '.GOTO label' command refers only to the forward direction while the '.BACKTO label' command only allows the user to go back up the control file.

Any monitor command that you can use in a timesharing job can be used in a Batch job with the following exceptions. The ATTACH, DETACH, CCONT, CSTART, SEND, and SET TTY commands have no meaning in a Batch job. If you include one of these commands in your job, Batch will write the command and an error label BAERR into your log file, and will not process the command. The user should not include a LOGIN or a KJOB command in the control file because batch does this for you.

6.3.2 Submitting the Job to Batch

After the control file has been created, it must be entered onto the batch queue of jobs to be run. This is achieved using the SUBMIT command. All programs and data that are to be processed when the job is run must be made up in advance or be generated during the running of the job. These files may be on any medium, but if they are on any device other than public disks, the user must make provision in the control file, to have the required devices mounted (see MOUNT Command section 9.1).

Unless otherwise specified all printed output from the job will be sent to the printer at the station where the job originated. If the user desires that, instead, it be printed at one of the other stations, he should issue the LOCATE command prior to submitting the control file. Refer to Section 7.3.2 for details of this command.

The SUBMIT command has the general form:

.SUBMIT jobname=controlfile.ext,logfile.ext/switches

where:

jobname

is the name that you give to your job. If this name is omitted, Batch uses the name of the control file. This is the exercise name for student jobs.

controlfile.ext

is the name that you have given to the control file that you created. You can add an extension, but if you don't, Batch will assume an extension of .CTL.

logfile.ext

is the name that Batch will give the log file when it is created. You can add an extension, but if you don't, Batch will assume an extension of .LOG.

The user must specify the name of the control file. If the name of the log file is omitted, its name will be taken from the name of the control file.

/switches

are switches to Batch to tell it how to process your job and what the output will look like. Most switches can appear anywhere in the command string; however, a few must be placed after the files to which they pertain. The various kinds of switches are described below.

The most basic form of the command is

```
.SUBMIT control-file
```

Three groups of switches are available to the user in the SUBMIT command. The switches are: queue operation, general, and file control, and are designed to give the user more control over processing of the control file.

Switches

The switches detailed under the QUEUE command in the Operating System Commands Manual may be used with the SUBMIT command. In addition the following can be used to specify how much output a job should produce; how long it will run, whether or not it should be restarted after a system crash; and so on. These switches have the default values indicated.

/COST:cost where cost=job cost limit (default $1.00).

/PAGE:n Use n (decimal) as the maximum number of pages of output that the job can print (Default 200).

/TIME:hh:mm:ss — Specify the central processor time limit for the job. A colon is used to separate the hours, minutes and seconds. If no switch is specified, the limit is 5 minutes.

/UNIQUE:0 or 1 — Run any number of batch jobs under this project,programmer number at the same time, if value is 0. Run only one Batch job at any one time, if 1 (default).

/PRIORITY:n — Indicates the external priority (default is 10). Refer to MNT-1 for details of the effect of priority on job cost.

/CORE:n — Use n (in decimal K) as the maximum amount of core memory that the job can use (default 60K).

/RESTART:1 — the job is to be restarted automatically if the machine crashes during its execution (default is 0 indicating no restart).

Examples:

1. .SUBMIT USRJOB=CONTRL,LOGFIL<cr>
   Submit the file CONTRL.CTL for processing. The exercise name is USRJOB and the log file produced is LOGFIL.LOG.

2. .SUBMIT<cr>
   List all entries in the Batch input queue. This command can be used to determine when a job has been completed.

3. .SUBMIT [170,103]<cr>
   List all entries for ppn [170,103] in the batch input queue.

4. .SUBMIT TEST=/KILL<cr>
   Remove the job TEST from the queue if the Batch system has not started to process the control file.

FILE CONTROL switches can be used in the SUBMIT command. They mainly control the disposal of the file, ie.

/DISPOSE:DELETE	remove file from user area after it has been printed.
/DISPOSE:PRESERVE	retain file on user area.
/DISPOSE:RENAME	remove file immediately from user area when the job is finished.

or the starting location in the control file:

/TAG:n	begin processing at label n.

Full details for entering a job to batch from a remote terminal may be found in chapter 3 of the DECsystem-10 Beginner's Guide to Multiprogram Batch.

6.4 Interpreting the Printed Output

All jobs run through the batch system produce printed output of some sort. This output can be:

1. Output that the user requests.
2. Output produced by the batch system, e.g. the log file.
3. Output that is a consequence of actions by the user's job, batch, the monitor, or system programs, e.g. compilation listings.

Program output may be directed to any allowable device (e.g. disk). However, a log file will always be produced and if the program output is directed to the line printer, then it will be printed separately to the log file.

The printed output from the user's program will be preceded by two pages. The first page contains information such as the user's name and ppn, and the second contains information relating to the file to be printed. If more files are to be printed, then they will appear in sequence preceded by a header page. The last page of any printed output will contain the user's name and ppn and also details of the printing, e.g. number of pages.

Other output that you can get as a result of your job includes compiler and cross-reference listings, loader maps for programs that were successfully loaded, and dumps.

The compiler and cross-reference listings are those listings generated by the compiler if you request them. When you enter your job from cards, batch requests compilation listings for you unless you specify otherwise. When you enter your job from a terminal, you must request the listings in the COMPILE command.

If you enter your job from a terminal, you may request a loader map in the EXECUTE, by use of a /MAP switch. If you wish to know the locations into which your program was loaded the loader map can be of use to you.

If a fatal error occurs within a program in the job and you have not included an error recovery command to Batch, Batch will not try to recover from the error for you. Instead, it will write the error into the log file and terminate the job.

6.4.1 Interpreting a Log File

The following log file is an example of what is produced by a simple FORTRAN Compile and Execute batch job. The log file is very useful since it contains a trace of all the dialogue between the user job and the monitor, system programs and the commands created by batch. It is a record of the complete

job and contains run time, job cost, all TTY communication and error messages.

Each line of the log file is preceded by the time of day when the entry was made and a word which gives the origin of the information on the line.

```
12:47:27 STDAT   27-NOV-79 Prentice KL 701 #3  SPRINT Version 102A(132043)-
                 2 Running on CDR010
12:47:27 STCRD   $SEQUENCE   197132
12:47:27 STCRD   $JOB TEST [60,106]/NAME:TEST/COST:$5.
12:47:28 STCRD   $FORTRAN
12:47:29 STMSG   File  DSK:LN3QXV.FOR  Created - 16 Cards Read - 3 Blocks
                 Written
12:47:30 STCRD   $DATA
12:47:31 STMSG   File  DSK:QXW.CDR  Created - 12 Cards Read - 2 Blocks Written
12:47:31 STCRD   $EOJ
12:47:31 STSUM   End of Job Encountered
12:47:31 STSUM   33 Cards Read
12:47:31 STSUM   Batch Input Request Created


12:47:33 BAJOB   BATCON version 102(134072) running TEST sequence 19713 in
                 stream 3
12:47:33 BAFIL   Input from DSKD:JB3QXU.CTL[60,106]
12:47:33 BAFIL   Output to DSKD:JB3QXU.LOG[60,106]
12:47:33 BASUM   Job Parameters
                 Time:00:05:00   Core:120P   Unique:YES   Restart:YES
                 Output:LOG

12:47:33 MONTR
12:47:33 MONTR   .LOGIN 60/106 /DEFER/SPOOL:ALL/TIME:300/CORE:120P/LOCATE:1
                  /COS:500/EX:TEST/SEQ:19713/PRIO:10
12:47:34 USER    JOB 32 Prentice KL 603A  15 TTY222
12:47:36 USER    A/c balance is $29.19
12:47:36 USER    [Updated 18:16:22 26-Nov-79]
12:47:36 MONTR   Charge no. 30
12:47:36 MONTR   Cost limit $5.00
12:47:36 MONTR   Seq. no. 19713
12:47:36 MONTR   12:47:36    27-Nov-79       Tue
12:47:38 MONTR
                 !start of LN3QXV.CTL generated by input spooler
12:47:38 MONTR   ..COMPIL /COMP/F10 DSK:LN3QXV.FOR/LIST
12:47:40 USER    FORTRAN: LN3QXV
12:47:41 USER    MAIN.
12:47:41 MONTR
12:47:41 MONTR   ..SET CDR QXW     !file name for input from spooled
                                   !card reader
12:47:42 MONTR
12:47:42 MONTR   ..EXECUTE/REL DSK:LN3QXV.REL
12:47:42 USER    LINK:   Loading
12:47:43 USER    [LNKXCT LN3QXV Execution]
12:47:43 USER
12:47:43 USER    STOP
12:47:43 USER
12:47:43 USER    END OF EXECUTION
```

```
12:47:43 USER   CPU TIME: 0.54  ELAPSED TIME: 1.64
12:47:43 MONTR  Exit
12:47:43 MONTR
12:47:43 MONTR  .
                %ERR::
12:47:43 BLABL  %FIN::
                !the above label and the following commands
                !have been inserted by the input spooler for
                !housekeeping purposes
12:47:43 MONTR  .DELETE DSK:LN3QXV.FOR,DSK:LN3QXV.REL,DSK:QXW.CDR
12:47:44 USER   Files deleted:
12:47:44 USER   LN3QXV.FOR
12:47:44 USER   03 Blocks freed
12:47:45 USER   LN3QXV.REL
12:47:45 USER   01 Blocks freed
12:47:45 USER   QXW.CDR
12:47:45 USER   02 Blocks freed
12:47:45 MONTR
                !This is the last line of the .CTL file,
                !subsequent lines generated by LOGOUT &
                !BATCON
12:47:46 USER   .KJOB/BATCH
12:47:46 USER
12:47:46 MONTR  Job 32, User TEST
12:47:46 MONTR  Logged-off TTY222  at 12:47:46  on 27-Nov-79
12:47:46 MONTR  Cost $0.22 [Excluding spooled I/O & MOUNT charges]
12:47:46 MONTR  Runtime: 0:00:04, KCS:94, Connect time:0:00:12
12:47:46 MONTR  Disk Reads:164, Writes 20
```

A Control cards entered by the user. the stacker generates a control file of commands according to the control cards encountered.

B Message from the stacker telling of the filename assigned to the card deck being read in, the number of cards read and the number of disk blocks occupied by the file.

C Summary message from the stacker describing the card reading just done and the batch job still to be done.

D First message from BATCON, describing input and output files and essential job parameters.

E Login sequence initiated by BATCON.

F Commands in the control file originally generated by the stacker. The $FORTRAN card caused it to insert a .COMPIL command in the control file, and the $DATA card resulted in the .SET CDR and .EXECUTE commands.

G Pre-execution messages from the linking loader LINK, and post-execution summary from the Fortran operating system FOROTS.

H Files created from the card decks by the stacker are deleted before logout.

I Logout sequence initiated by BATCON.

6.5 Error Messages in Batch

Any error conditions which may arise during the processing of a batch job will be reported in the user's log file. These errors may be detected at a number of levels:

(i) by the stacker
(ii) by the batch controller
(iii) by the system.

The following errors may be reported by the stacker:

	FATAL		NON-FATAL
(i)	Error in $JOB card	(i)	Hollerith error (improper punch)
(ii)	Unrecognizable command on a batch control card	(ii)	Missing $EOJ card
(iii)	Error in a parameter on a batch control card	(iii)	Unrecognized switches
(iv)	Improper code (binary rather than Hollerith)		

Errors reported by the batch controller (BATCON) may be found in the DECsystem-10 Operating System Commands Manual.

Normal system errors will be reported as for timesharing and will usually be fatal, unless provision is made to handle them with .ERROR or .NOERROR commands to the batch controller.

6.6 ERROR RECOVERY

Refer and Sections 3.3 and 3.4 of the DECsystem-10 Beginner's Guide to Multiprogram Batch for more details.

When jobs are running under the Batch system, various errors can occur during processing of the job. These errors may arise from a fatal error while executing a program, trying to manipulate non-existent files, and so on. If an error occurs in the job, the batch controller is notified and decides what action is to be taken. When the line of the log file begins with a '?', then the Batch system assumes that there was a fatal error in processing the previous command and normally the job is terminated with any remaining commands being ignored.

This abrupt termination of the job is not always desirable and so the user has the option of recovery from the error. Special Batch commands exist to determine whether or not the execution of a particular command was successful. The user may then transfer control to an appropriate set of

commands in the control file to handle the error. These Batch commands are

.IF (ERROR) command
If an error occurs the COMMAND is executed, else it is ignored and the next statement is processed.

.IF (NOERROR) command
If an error does not occur the COMMAND is executed, else it is ignored and the next command is processed.

Example - Use of Error Recovery commands

```
.R SORT
*TEMP.SRT/RECORD:50/KEY:1:5=TEMP.DAT
.IF (NOERROR) .GOTO LAB1
                    If there is a fatal error (e.g. file TEMP.DAT was
                    not found) execute the next statement, else go to
                    label LAB1.
!TEMP.DAT file does not exist
.CREATE TEMP.SRT
*@@@@@
*
*FILE                !TEMP.SRT file created
LAB1::               !label LAB1
.RUN TEST
.IF (ERROR)          !ignore any error from execution of program TEST
.
.
.
%FIN::               !commands after %FIN are always executed.
```

Warnings or non-fatal errors are prefixed by the character '%' and are normally ignored by the Batch system unless previously designated as a fatal error indicator through use of the command

.ERROR %

6.6.1 Restart Option

If the computer 'crashes' while the Batch job is running, the user may again wish to incorporate recovery procedures within the control file. Normally, the Batch system will not automatically restart the job when the computer resumes operations, but simply abort the job. Therefore, if an automatic restart is required, specify the /RESTART:1 switch with the SUBMIT command or the $JOB card, and indicate where in the control file the job should be restarted.

The user can set up any number of CHECKPOINTS throughout the control file for restart processing. If no checkpoints are specified, then the restart will be from the beginning of the control file, else restart will be from the label found in the last checkpoint processed.

Example - Control file submitted using the /RESTART:1 option

```
START::
.MOUNT PRIV:
.IF (ERROR) .REQUEUE START:
.
.
.
.RUN UPDATE          to alter master files on PRIV:
.GOTO UPDOK:
!Restart processing
CONT1::
.MOUNT PRIV:
.IF (ERROR) .REQUEUE CONT1:
UPDOK::
.CHKPNT CONT1:
.RUN TEST1            using files on PRIV:
```

6.7 MPB - GALAXY Differences

Some of the differences which have been observed between GALAXY (KL system) and MPB (KA system) are discussed below.

Devices LPT: and PLT:

When output is directed to device LPT: by, for example, specifying output to device unit number 3 in FORTRAN, the LPT file under GALAXY is automatically queued for printing when the file is closed within the program and then NO LONGER EXISTS ON THE USER'S AREA. To specify special stationery or multiple copies of this file, the user should therefore direct the output to disk, and then issue the command

```
.PRINT file/switches
```

Plot files should also be queued explicitly using the command

```
.PLOT *.PLT
```

Under MPB on the KA however, these files with extensions .LPT and .PLT, exist until the job is logged out. Logout then causes them to be queued and deleted from the user's area. So please make sure if you don't want the LPT listings generated by your compilations, to delete them before you logout. (PAPER COSTS MONEY)....

LIST FILES

The command K/H provides a comprehensive list of actions on logout.

Under MPB on the KA:

KJOB under MPB invokes the default switch /Z:4 and queues or deletes files in the following order if the user is over the logged-out quota.

1. queued files with owner protection 0 will be spooled;
2. files exceeding the logged-out quota, if protected queued files they will be unprotected and spooled, and if not queued, then they will be deleted;
3. unprotected files will be deleted as follows:
   temporary files, then REL, BAK, EXE, any others;
4. protected files will be deleted as follows:
   temporaty files, then REL, BAK, EXE, any others until quota is enforced.

Under GALAXY on the KL:

KJOB under GALAXY invokes the switch /BATCH and does not queue any files at LOGOUT but will delete files in the following order if the user is over the logged-out quota.

1. unprotected temporary files BAK, LOG, MAP, LST;
2. unprotected REL, EXE, other files not recognised as source or data files
3. other temporary files (including BAK, LOG, MAP, LST);
4. unprotected source and data files;
5. other REL, EXE, and files not recognised as source or data files.

Therefore the user should print all .LST files before KJOB in the Batch control files.

/OUTPUT

This switch is included on either the $JOB card or in the SUBMIT command to alter the degree with which files are automatically queued for output.

Under MPB on the KA:

Line printer output will automatically be sent to the printer and disappear from the user's area, unless specifically held using this switch.

```
/OUT:0    no spooling;
      1   print the log file;
      2   print log file and other queued output;
      3   as for 2, plus .LST files
      4   as for 2, plus deferred output.
```

Under GALAXY on the KL:

/OUT:LOG	print the log file (default);
NOLOG	do not print the log file;
ERROR	print the log file only if errors are encountered.

CHAPTER 7

COMMUNICATIONS NETWORK

7.1 COMPUTER SYSTEM NETWORK

7.1.1 What is a Computer Network?

The Computer Centre has three central computers for processing work. These are the two DECsystem-10s and a VAX-11. Each can operate independently of the others. Each is intended to provide a different type of service.

They aren't isolated from each other. PDP-11 minicomputers are used to control communications between machines. This allows data to be sent from one computer to another computer. Many departments in the University can connect their PDP-11s to the Computer Centre's, so that data can be transferred to or from the central computers.

This linkup of computers, within and outside the Centre, is called a Network.

Computers in a network are referred to as Nodes in the network. Not all nodes perform the same functions. Some are central computers for processing user jobs. Some are used only as a means of connecting two computers. Another type of node has many terminals connected to it - this is a Terminal Front-End. Other nodes have a card reader, or a line printer attached; then it's usually called a Remote Batch Station. These Batch Stations allow a terminal user to print files at a place convenient to him. The Hawken Building, the Commerce Building, and the Science II Building at Griffith University have Batch Stations.

All nodes in the network have a unique Node Name and Node Number. Here is a list of the more important nodes in the network.

Node Name	Number	Description
UQKL10	1	PDP-10/90 system. A Central Computer.
UQVAX	3	VAX-11/780. A Central Computer.
UQKA10	6	PDP-10/55 system. A Central Computer.
DN87SA	10	Terminal Front-End.
DN87SB	11	Terminal Front-End.
COMERC	30	Commerce Building Batch Station.
GUAES	50	Griffith University Batch Station.
CSIRO	70	CSIRO's system.

7.2 TERMINAL USAGE

7.2.1 Getting to the computer.

About 350 terminals are connected to the Centre's computers. They are connected to different computers, and are subject to the limitations of that computer, as far as accessing other computers is concerned.

Terminals can be connected to the KA-10 (i.e. the PDP-10/55), or a terminal concentrator, or to the MICOM Circuit Switcher.

Those terminals on the KA-10 can't get to any other computers. Mainly the Computer Science Department has these terminals.

Changing computers

Terminals connected to a Batch Station or Terminal Front-End are more fortunate. These will usually start up "talking" to the KL-10. You can then select which of the PDP-10s or CSIRO to use - i.e. what their Host Computer is to be. The SET HOST command is used to do this.

The form of the command is

.SET HOST xxxx

where "xxxx" is to be replaced by the new host name. Node numbers can also be used. Giving the command

.SET HOST UQKL10

will set the KL-10 to be your host computer. All of your input will be sent to it.

There are three computers you can SET HOST to. They are -

UQKL10 - node 1, PDP-10 general purpose system.
UQKA10 - node 6, PDP-10 SLOTS system.
CSIRO - node 70, the CSIRO system.

The MICOM

Some terminals are connected to a different style of device. It's not a terminal concentrator, but a circuit switcher - called the MICOM. It attaches you to a computer of your choice, and you appear to be connected directly to that computer. Think of the MICOM as something that allows you to unplug your terminal from a computer, and plug it into another. After doing this, the MICOM is totally transparent.

The MICOM will allow you to connect to the three central computers. These are -

UQKL10 - node 1
UQKA10 - node 6
UQVAX - node 3

Here's how the MICOM is used. Push the RETURN key, and the MICOM prompts you with "HOST =" . Then type in the desired node number.

HOST = 1
GO

and your terminal is now connected to one of the KL-10's terminal concentrators. Remember that you now must type RETURN to establish contact with the KL-10.

You can re-establish contact with the MICOM (to select another computer) by pushing the BREAK key on your terminal. Push the RETURN key, and the MICOM will prompt you again with "HOST =" .

Here are a few hints about the MICOM. It will automatically find the correct speed (or BAUD RATE) of your terminal (up to 2400 baud). The terminal concentrators will also auto-baud - but only to 1200 baud. Use your terminal at speeds of 300 or 1200 baud to minimise difficulties.

Telling the difference

How can you tell what a terminal is connected to? Turn it on, type RETURN, and give the command WHERE. A different message is typed by each machine.

(i) The message

Connected to MICOM line 200.

means that you are connected to the MICOM circuit switcher.

(ii) A response like

Node UQKA10(6) 603A-SLOTS #20 07-07-81

tells you that the terminal is directly connected to the KA-10. The message has printed out the node name - here, UQKA10.

(iii) To a person connected to a terminal front-end or batch station in the network, a similar message is printed. Instead of UQKA10, the terminal front-end or batch station prints its own name. If you recieve any of the node names

UQKL10 DN87SA DN87SB COMERC GUAES RSXANF

you can issue the SET HOST command.

Error Messages

Sometimes you can't get to the computer you asked for. A few of the more

common error messages are listed below.

?DECsystem-10 TTY's can not be switched via HOST
This message comes from the KA-10. It means that your terminal is connected to the KA-10, and that you cannot change your host.

?Undefined Network Node
This message means that the node name (or number) you gave is not known by the system. You might have spelt it incorrectly, or else that node might not be running.

?Node does not support remote terminals
The node specified in your SET HOST command can't be used as a host computer. Only the KA-10, KL-10 and CSIRO can be given in a SET HOST.

No host is available.
None of the nodes valid for a SET HOST command are running, or cannot be accessed for some reason. Ring the Centre's recorded message service (ext. 3101) for more information. Try again later.

The remaining messages come from the MICOM circuit switcher.

Type "HELP" for Help.
The MICOM hasn't been able to understand the node name. Do what the message says, or else check the spelling of the node name.

UNASSIGNED DISCONNECTED
The MICOM has been given a node number which doesn't really exist. Check the node number. Type RETURN to get another prompt.

UNAVAILABLE DISCONNECTED
The node you've asked for isn't working. Ring the recorded message service to find out when that node may be available.

BUSY, WAIT? 001
There aren't any more lines left to the specified node. You are to be the first person in the queue waiting for a free line. Type in a "Y" if you want to wait for a free line (and be notified), or a "N" otherwise. Typing "C" will get you another prompt.

WRONG SPEED, DISCONNECTED
The baud rate of your terminal isn't suitable to connect to the node. Some nodes aren't auto-baud, (e.g. the KA-10) and the MICOM will try to find a line to that node at the same speed as your terminal. If this can't be found, the "WRONG SPEED" message is given.

7.2.1.1 A quick review -

To gain access to the computers, turn on your terminal and type RETURN.

If you are prompted by "HOST =", type in the number of the computer you want to access. Then type RETURN to get acknowleged by that computer.

If you get the message "Connecting to . . . " you are told which computer you are connected to. Issue a SET HOST command if neccessary.

Merely getting a "dot" from the machine means you could be on the KA-10. Type "WHERE" to check.

7.2.2 Location of Output

Terminal users can indicate that all subsequent output (e.g. files explicitly printed, spooled output, and log files from batch jobs submitted from the terminal) be sent to the designated logical node by the use of the command

.LOCATE n<cr>

Where n is the required logical node number.

Example:

.LOCATE 50<cr>	establishes Griffith as logical node.
.PRINT MYPROG.FOR<cr>	prints output at Griffith batch station.
.SUBMIT SURVEY.CTL<cr>	queues all spooled LPT: output together with the logfile to Griffith batch station.

Alternatively, files can be selectively queued to any station by explicit specification on the PRINT command. The device names used to specify output destination in commands such as QUEUE or PRINT should take the recommended form

DEVnn

which refers to any device of type DEV on the nodes specified by the two character argument 'nn'.

Examples:

(i) .PRINT LPT30:=*.LPT,*.LST<cr>

will queue all files with .LPT and .LST extensions to be printed at node 30 (the Commerce remote batch station).

(ii) .PRINT FIXOVF.FOR<cr>

queues given file to the central printer at the Prentice Computer Centre, Batch Station(if the user is not connected to Griffith or Commerce nodes), unless a previous LOCATE command had established a different logical station.

(iii) .PRINT LPT01:=file<cr>

will cause the file to be printed at node 1 which is the Central site in the Hawken Building.

(iv) .PRINT LPT30:=file<cr>

will cause the file to be printed at node 30, which is at the Commerce Batch station.

Plot Files

On either system, plotter output should be queued by using the PLOT command without specifying a destination device, eg.

.PLOT file<cr>
or
.QUEUE PLT:=file<cr>

Although PLT010: exists on the KL10, '565' plot files are transferred by the system operator to the correct queues on the KA10 system.

7.2.3 Self Service Printing

The self-service printing facility is located in the clients' area on the ground floor of the Hawken Building and is available for printing files from either the KL10 or the KA10. Only jobs with a limit of 30 pages or less can be scheduled for printing on the self-service printer.

On either system, files may be queued for printing on the self-service printer by either

(i) specifying the output device as LPT20:
.PRINT LPT20:=file-list/LIMIT:30<cr>

(ii) or by issuing the command
.LOCATE 20<cr>
followed by the print commands.

To obtain these files queued for printing, the user must subsequently log onto the terminal at the self-service printer and these files will be printed.

Help may be obtained by listing the SSLPT help file.

Examples:

.HELP SSLPT<cr> types the help file at the user terminal.

.PRINT HLP:SSLPT.HLP<cr> prints the help file on the line printer.

7.2.4 High Quality Printing

Available at the Prentice Computer Centre are a Sanders Printer and a Diablo Terminal; either can be booked at the accounts enquiry section for a maximum of two hours. These machines produce high quality copy, with the option of using different type face. Users may type files produced by special text producing software from either the KA or KL computers.

Users of the KL should also be aware of the availability of a Photo-typesetting package producing high quality bromides; that is camera-ready copy for reproduction by a printing process. For complete detail please refer to MNT-7 Introduction to Computer Typesetting.

CHAPTER 8

FILE MIGRATION SYSTEM

8.1 General

The File Migration System, available only on the KL10, enables the user to request that online files be moved to offline storage (ARCHIVE) or that files currently held offline be returned to his online area (RETRIEVE). The user can also request that files on offline storage be deleted (ODELETE); request that the names or protection of any offline files be changed (ORENAME); and, at any time, obtain a directory of his offline storage (ODIRECT).

The same filenaming conventions apply to offline as to online files. The full wildcard filename construction is available with all commands. For example, *.DAT refers to all files with a DAT extension.

The requests for retrievals are queued and the request queue is periodically processed by the system. The response time is of the order of a couple of hours. The requests for archivals, deletions, and renames are processed immediately.

The service is available to both Batch and Terminal users, and is strongly recommended as a means of keeping the online public disk area free of unwanted or inactive files. The cost of offline storage is one eighth of the online cost, and is intended to be an added incentive for use of the File Migration System.

8.2 User System Commands

(a) .ARCHIVE filename(s)<cr>

This command causes the named file(s) to be archived (placed into offline storage). The system checks:

(i) that the file named does exist online, and

(ii) that the user has correct access privileges to the file, and

(iii) that the user has correct access privileges to write the file on the offline area

and reports errors to the user.

The file(s) is transferred to offline storage immediately, the online copy of the file being deleted.

Example:

.ARCHIVE FOOBAZ.MAC, *.DAT<cr>

will archive the file FOOBAZ.MAC and all files with the extension DAT.

The /P switch can be used to preserve the online copy of the file. It may appear directly after the ARCHIVE command, in which case it applies to all files in the list, or after the individual file name to which it applies.

Example:

.ARCHIVE XYZ.FOR/P, TEST.REL<cr>

will result, after queue processing, in XYZ.FOR being both online and offline but TEST.REL offline only.

(b) .RETRIEVE filename(s)<cr>

This command enters a request onto the queue for the named file(s) to be retrieved from offline to online storage. As with the ARCHIVE command, checks for correct access privileges are made and the file will be deleted from offline storage when it has been transferred online. Unlike .ARCHIVE, the request may take up to two hours to process. Again the /P switch is available to preserve the file offline if required.

(c) .ODELETE filename(s)<cr>

This command allows files to be deleted while still on the offline area. The use of the ODELETE command is similar to the ARCHIVE command. It causes the files specified to be deleted immediately. Note that it is NOT necessary to RETRIEVE files to the user's online area before deleting them.

(d) .ODIRECT filename(s)<cr>

This command lists the user's offline directory in a format resembling that produced by the DIRECT command. If a filename is not specified, all the user's files will be listed.

(e) .ORENAME filename1=filename2/protect:nnn,etc.<cr>

This command allows the user to change the name of his offline files without first retrieving them to online. Also the file's protection code may be changed.

8.3 More Advanced Use of the File Migration System

This document outlines only the most basic techniques for each of the File Migration System commands. For more advanced uses it is recommended that the user obtain a listing of the file FMSCOM.DOC using the following command:

.PRINT DOC:FMSCOM.DOC<cr>

CHAPTER 9

USE OF PERIPHERAL STORAGE DEVICES

As well as being able to store files on his online disk area, a user may wish to ultilize other mass storage devices such as private disk packs, or magnetic tapes. The PDP-10 makes provision for this and enables a user to request that a particular storage device be Mounted and assigned to him. Naturally, the number of these devices is limited and so the user may only have access to them when they are available. Once the unit has been allocated to a user, it is available for his use only until it is released back into the system (DISMOUNT command).

9.1 MOUNT command

The MOUNT command allows the user to request assignment of a device via the operator. By requiring that device requests must be channelled through the operator and thus preventing the direct assignment of a device by a user, the system ensures that the operator has greater control and may select and assign a particular device, or may cancel the request completely. Currently the MOUNT command is only available on the KL10 computer.

Command Format:

MOUNT dev:logical-dev/switches

where dev:=one of the following:

1. a physical device name (e.g. MTA:),
2. a logical name previously associated with a physical device by a MOUNT command, or
3. a file structure name (eg. DSKB:).

This argument is mandatory.

logical-dev=any name 1 to 6 characters to be assigned to the device.

It is important that the user should identify the particular unit to be mounted. All tapes stored at the Computer Centre have an identification number and this should be specified in the REEL: switch when requesting that

the device be mounted.

Switches available for the Mount command include the following:

Switches

/CHECK Check and list pending requests.

/MULTI Multi-access, disk only, default condition.

/PAUSE Notify the user before sending the message to the operator for a request. The user can then abort the command if desired.

/REEL:nn The number is the Computer Centre identification number assigned when the tape is registered with the Centre.

/SINGLE Only this job can access files on the structure (single access), disk only.

/VID:name A visual identification passed to the operator as a comment to assist in identifying a particular unit to mount.

/WAIT Wait before continuing with your job when operator intervention is required.

/WENABL Write-enable for this job, complement of /WLOCK (default condition)

/WLOCK Write locked for this job (prevents device being written to)

Example - The private disk pack SPQA: is required to be mounted

```
.MOUNT SPQA:<cr>          Mount the file structure SPQA.
Request queued            The request is queued to the operator.
WAITING...  2 ^C's to exit
                          UMOUNT is waiting for the request
^C                        to be completed. The user does not wait for
                          confirmation.
^C
.MOUNT/WAIT<cr>           Leave terminal in wait mode so that monitor response
                          is given when the pack is mounted.
SPQA mounted
.MOUNT/CHECK<cr>          The user wants to know if his request has been
                          processed.
NONE PENDING              The request has been processed.
0 COMMANDS IN QUEUE.
.MOUNT MTA:OUTPUT/REELID:2000/WENABLE<cr>
                          A 9 track magnetic tape with reel number 2000 is to
                          be mounted with the logical name "OUTPUT", to be
                          written on.
Request queued
Waiting 2 ^C's to exit
                          The request is queued to the operator.
```

9.2 DISMOUNT Command

The DISMOUNT command enables a user to return devices to the monitor pool of available resources when it is no longer required. (This is automatically done when the user logs out). It is in the interests of the user to do this immediately the device is no longer needed, as a charge is incurred for the time a device is assigned to the user.

Command Format

.DISMOUNT dev:switches

dev: = any previously ASSIGNed or MOUNTed device or file structure name. This argument is required.

Switches

/CHECK Check and list pending requests.

/PAUSE Notify the user before requesting operator action. The user can then abort the command if desired.

/REMOVE Notify operator to remove disk packs, or tapes. A file structure is removed from the system only if no other users are using it. A request to remove the pack is queued to the operator. This switch must be specified to notify the operator to remove the pack, even if no other jobs are using it.

/NOWAIT Do not wait for the operator to acknowledge the DISMOUNT requests before allowing the user to continue the job. This is the default condition for timesharing jobs BUT NOT FOR BATCH JOBS.

Example:

```
.DISMOUNT PRIV:/W<cr>          The user asks the operator to deassign PRIV:
OPERATOR NOTIFIED              and remove the disk pack.

WAITING... 2^C's to exit       The command is waiting for completion of the
                               operator action.

^C                             The user does not wish to wait for
^C                             information of removal.

.DISMOUNT/CHECK<cr>            The user checks for completion and determines
NONE PENDING                   that his request is finished.
0 COMMANDS IN QUEUE .
```

9.3 Miscellaneous Useful Commands

9.3.1 ASSIGN and DEASSIGN Commands

ASSIGN

The ASSIGN command allocates an I/O device to a logical device. This will change the assignment until the user logs out or another ASSIGN changes the number again.

Command Format

ASSIGN phy-dev log-dev

where phy-dev is a physical device or file structure name. This argument is required.
log-dev is the logical device name.

Examples:

.ASSIGN TTY: LPT:<cr>
The user assigns his TTY and gives it the logical name LPT so that whenever LPT: is referenced, the unit TTY will actually be used.

.ASSIGN DSK:5<cr>
If Fortran unit 5 is required to be a disk structure rather than the terminal (default), this command should be used.

DEASSIGN

The DEASSIGN command is used to release any or all of these assignments without having to log out.

Examples:

.DEASSIGN<cr>
deassigns all assignments

.DEASSIGN LPT:<cr>
returns the line printer to the list of resources available for use by this job. Thus when LPT is referenced, the line printer is used.

9.3.2 RESOURCES Command

A device may only be mounted if the relevant drive is available. Thus the RESources command is used to check which units are available.

Example:

```
.RES<cr>
ACCB,DSKO,DSKN,DSKB,DSKD,DSKF,DSKG,RPB2,DPA0,DPA1,DPA4,DPA5,CDR010,
MTA010,011
```

The appearance of "MTA010,011" indicates that both the magnetic tape drives are available for use. If neither of these appear in the list, the tape drives are already being used and the user's tape cannot be mounted yet.

The appearance of the drive names indicates the availability of structures in the following manner:

NAME	STRUCTURE
MTA010,011	9-track magnetic tape drives
DPA0,DPA1	RP02 disk drives
DPA2,DPA3,DPA4,DPA5	RP03 disk drives
RPA1,RPA2,RPB1,RPB2	RP06 disk drives

Consequently, the user should know which type of structure he requires and check the list of available drives accordingly.

The user may however book a drive to ensure that it is available at the requested time. Refer to MNT-1 for the appropriate booking procedure.

9.3.3 REWIND Command

The REWIND command rewinds a magnetic tape.

Example:

```
.REWIND MTA011:<cr>
```

9.3.4 SET DENSITY Command

This command sets a default density in bits per inch for the specified magnetic tape (otherwise density of 1600 bpi is assumed on the KL system).

Command Format

```
SET DENSITY dev:nnn
```

where dev: is the magnetic tape drive for which the density is to be set, or the logical name associated with a physical tape;
nnn is one of the following densities in bpi - 200,556,800,1600

Example:

```
.SET DENSITY MTA010:800<cr>
```

9.3.5 SETSRC Program

The SETSRC program is used to manipulate the job's search list. A search list is defined to be the order of file structures that are to be searched whenever the generic device DSK: is explicitly or implicitly specified by the user.
Reference: DECsystem-10 Operating System Command Manual

Example:

```
.R SETSRC<cr>
*T<cr>
DSKN:/NOCREATE,DSKD:,PRIV:,FENCE
^Z
.DISMOUNT PRIV:/REMOVE<cr>
%PRIV has other users - /REMOVE ignored
PRIV dismounted
.R SETSRC<cr>
*T<cr>
DSKN:/NOCREATE,DSKD:,FENCE
*^Z
```

In this example, the user has requested that disk PRIV: be removed from the search list so that the user is no longer charged for its use, and also to be removed from the disk drive. PRIV: is removed from the job's search list, but since there are other jobs using PRIV:, a request to physically remove the pack cannot be completed.

9.4 GENERAL MAGNETIC TAPE USAGE

The KL10 system connects two TU45 9-track magnetic tape units. These units handle odd or even parity written at 800 bpi (NRZI) or 1600 bpi (PE). The system default is 1600 bits per inch odd parity.

In general, binary files created on one type of computer are useless on other types so that files being transferred between computers must be character files.

If a tape is being created on another computer system and is to be read by the Prentice Computer Centre, the following information about the tape should be obtained:

1. Density in bits per inch (preferably 800 or 1600)
2. Number of tracks (9-track)
3. Parity (odd)
4. Character code (8 bit ASCII, EBCDIC)
5. Type of label (none)
6. Blocking factor (fixed size blocking factor)
7. Record length (fixed record length)

If these characteristics cannot be adhered to, the user should initially contact the Computer Centre to discuss alternative formats.

Reference: DOC:MAGTAP.HLP
for information to users who may have to examine or convert foreign tapes that are brought to the Centre.

9.5 Program BACKUP Summary

Program BACKUP is used to save disk files onto magnetic tape and subsequently restore any or all of these files back to disk when DECSYSTEM-10 computers are being used. It enables the user to position the tape at the beginning of the file required for processing, and makes format and character specification unnecessary.

When the required magnetic tape has been mounted, program BACKUP can be used. ie.

.R BACKUP<cr>
/ response from BACKUP indicating that a command can now be accepted.

Some of the commands that can be used are summarized below in the order in which they are issued.

Reference: DOC:BACKUP.MEM
for additional details on backup usage, particularly with respect to the commands and relevant tape formats.

TAPE Command

If the magnetic tape was not given the logical name "BACKUP", use the TAPE Command so that the program will know which tape is being used.

Examples:
/TAPE 1021<cr> indicates logical name 1021 was used for the tape

/TAPE MTA010:<cr>
logical name was omitted in the MOUNT command.

REWIND Command

REW
ensures that positioning is at the beginning of the tape.

FILES Command (optional)

FILES
signifies that the name of each file is to be output as it is processed, and enables the user to check which files have actually been saved or restored.

INTERCHANGE Command (optional)

INTER
directs BACKUP to ignore installation dependent data so that only the information which is critical to the file itself is read or written. Project-programmer numbers are ignored.

EOT Command (optional)

EOT
is used to position the tape at the end of the last information previously written.

SKIP Command

SKIP n
positions the tape after n sets of files if n is positive, or before n sets of files if n is negative.

SSNAME Command (optional)

SSNAME "Name of Save Set"
Program BACKUP stores information consisting of one or more files as a "SET" on the tape. Therefore, when a particular file or group of files is to be accessed, the set or group of sets to be searched must be indicated.

Examples:

```
SSNAME ALL      to search all sets on a tape
SSNAME "SET1"   to search only the save set called
                "SET1".
```

This command is useful in documenting what is stored on the tape.

SAVE Command

SAVE file1,file2,...,filen
writes onto magnetic tape whatever files have been specified in the SAVE command.

Example:

SAVE *.* saves all disk files on the user area onto magnetic tape.

RESTORE Command

RESTORE file1,file2,...filen

Example:

RESTORE *.* restores all files from tape onto disk.

CHECK Command (optional)

CHECK spec-list
verifies that the tape and disk files agree.

Example:

```
.MOUNT MTA:BACKUP/REELID:SCRATCH/WE<cr>
Waiting 2 ^C's to exit
SCRATC mounted
.R BACKUP<cr>
/REWIND<cr>
/FILES<cr>
/INTER<cr>
/SSNAME "SET1 - UPDATE"<cr>
                name the set of files to be saved
/SAVE *.*<cr>   save all files from disk onto magnetic
                tape
!UPD1.FOR
UPD2.FOR
UPDATE.EXE

"Done           indicates completion of saving of files

/REW<cr>
/CHECK<cr>      verify saved tape files are identical to
                the disk files
/^C
Exit

.DIR BACKUP:<cr>
                for directory of files on this tape
```

```
.R BACKUP<cr>  To restore some files
/REW<cr>
/FILES<cr>
/INTER<cr>
/SSNAME "SET1 - UPDATE"<cr>
               only this save set is relevant
               for the files required
/RESTORE UPDATE.EXE<cr>
               copy only this one file to disk

"Done
/REW<cr>
/CHECK UPDATE.EXE<cr>
               verify that file was correctly
               restored
/^C
Exit
```

CHAPTER 10

FILE AREAS

The University of Queensland PDP-10 systems support various file areas for the storage of different types of software. Each of these file areas is identified by a mnemonic which can be used as an ersatz device, the most useful of these being outlined below.

An ERSATZ name is a method used to access a particular set of programs. The use of the Mnemonic type name saves having to remember the appropriate ppn.

Example - To execute a program from one of these libraries

.R UTI:CHANGE<cr>
will cause the program CHANGE on the utilities library to begin execution.

Example - to link-load a subroutine from a library

.LOAD/REL MAIN,PLO:CALF10/SEA <cr>
where MAIN is your main program and it will use subroutines for plotting found in the file CALF10 which is stored in the PLO: library.

The following contains a brief description of most of the ERSATZ libraries on the PDP-10. For details about the various programs and files the user should examine HLP: and DOC: for relevant information. Failing that contact the Consulting Programmer or the Program Librarian.

BLI: [5,5]
BLISS programming library. Bliss is a high level language that has powerful assembly langauage features, supports structured programming, co-routines. This library contains various support systems for BLISS. It is of interest to note that the FORTRAN compiler is written in BLISS.

DOC: [5,14]
DOC: Contains printable files of various documents on some of the software on the system. To obtain a full listing of what is available type
.DIR DOC: <cr>
to obtain a listing, type
.PRINT DOC:nnn.DOC <cr>

FIN: [5,114]
The financial library. Contains packages for various financial models.

GAM: [5,30]
A library of games for entertainment. Games (e.g. CHESS) are stored here.

HLP: [2,5]
The HELP library.

Examples:

.HELP HELP <cr> types out a complete list of all the help files available.
.HELP nnn<cr> types out the help file for nnn (if nnn.HLP exists!).

LAN: [5,102]
LAN: contains various specialized languages.

MAC: [5,7]
This library contains various files for assistance to MACRO programmers.

MAT: [5,101]
The Mathematics library.

MIC: [5,25]
A library of MIC macros. MIC is a system to allow a user to simulate BATCH like operations on your own terminal. MIC: contains some predesigned files for MIC usage. Most of the files on MIC: are of no use to users and are primarily used by the Centre for housekeeping tasks. For more information on the usage of MIC, obtain a listing of the MIC.DOC file as follows:
.PRINT DOC:MIC.DOC<cr>

NEW: [1,5]
NEW: is where software that is under evaluation and testing is placed. User's should be aware that software on NEW: is in this category and should obtain the latest information on the particular sofware from the Centre before use. After a new system has been accepted as satisfactory it is then transferred to a more permanent area.

OLD: [1,3]
When incompatibility arises with the updating of software then some users may wish to continue to use old versions until they have had sufficient time to change their programs. Software in this category is stored on OLD:.

ORS: [5,112]
Operations research software. Linear programming and critical path analysis software is stored here.

PLO: [5,106]
All plotting software is stored on PLO: To use any of the plotting routines for either of the plotters on the PDP-10 users need to load the required routine from this library. Most of the routines need to be linked with a user's main program in Fortran-10. There are some stand alone simple plotting programs for general cases.

PUB: [1,6]
PUB: is a general purpose test and store area for specific purpose systems and some incompatible variations of programming languages.

REL: [5,11]
The general area for all REL files that are available to programmers in the form of linkable subroutines and functions. An example is the MACUTL file which contains routines to allow Fortran programmers to perform various macro functions and monitor calls.

SIM: [5,104]
The simulation library.

STA: [5,105]
Statistics library. The IMSL library of Fortran subroutines is stored here, among many others.

TPS: [5,26]
Type-setting library.

TUT: [5,110]
Tutorial and teaching programs

UNV: [5,17]
Macro universal library.

UTI: [5,113]
Utilities library, contains programs for tape conversions and so on.

NOTE:- in addition to the above mentioned software packages, the Prentice Computer Centre also supplies a large range of specific software for particular applications. QDATA is a package which allows the user to enter data into a file from a terminal in a similar but much easier manner than punching cards. VG and 1022 are Data Base packages which have very wide areas of application - particularly in administration and research work.

There is a large amount of software for text processing ranging from basic formatting of simple text to photo-typesetting of complete books.

Before you write a program always remember to check with the program librarian; it is highly likely that your program (or one that will do the job for you) already exists!

INDEX