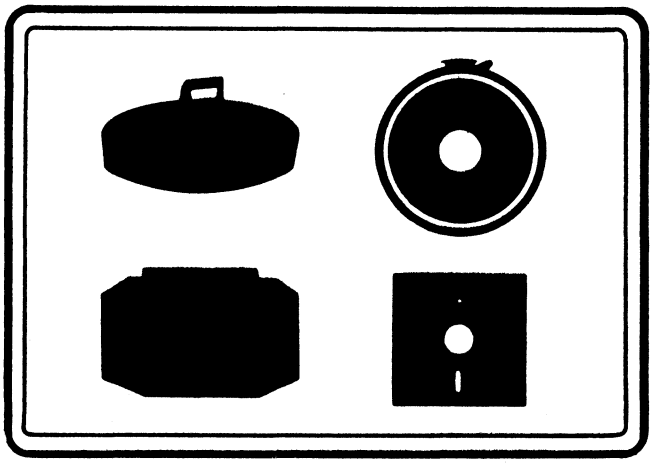
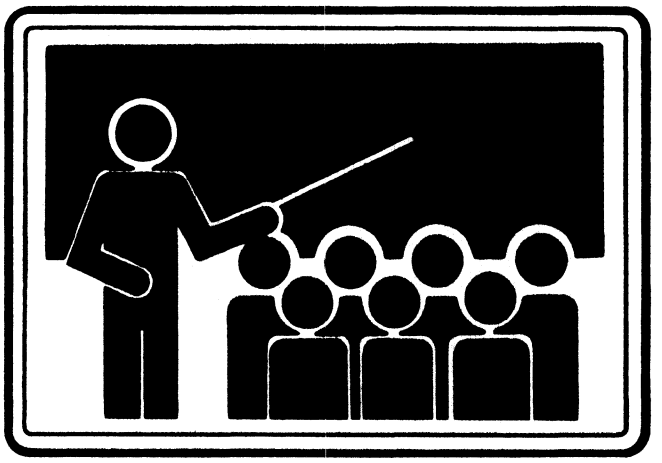
DataGeneral

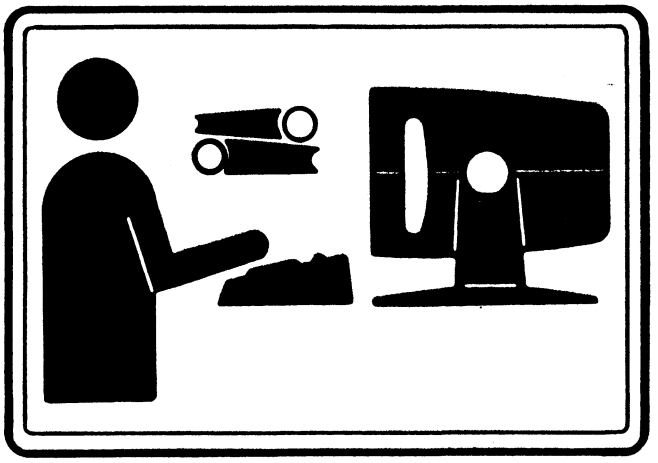
Educational Services

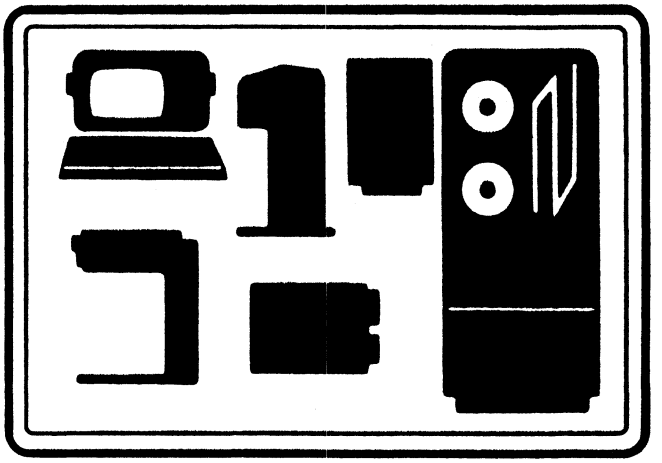
SH109/VS – AOS/VS OPERATOR TRAINING

STUDENT HANDOUT









019-000102-01

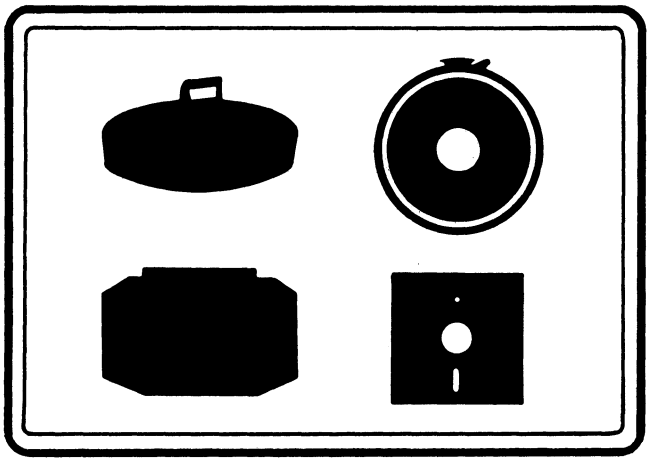
NOTICE

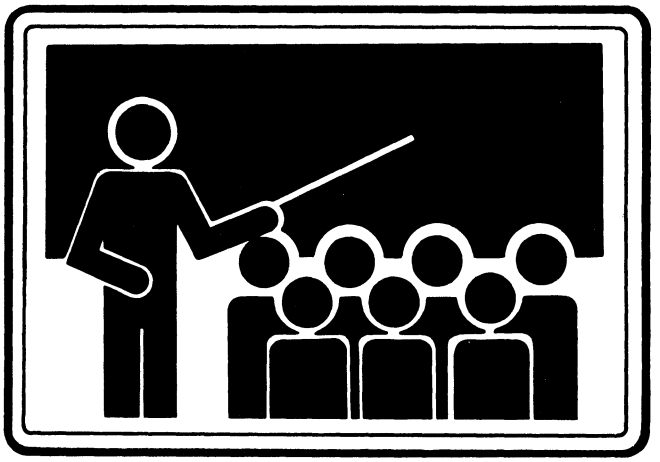
DataGeneral

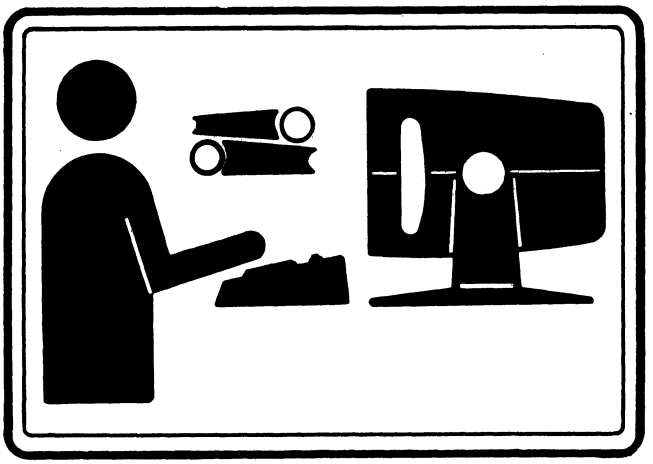
Educational Services

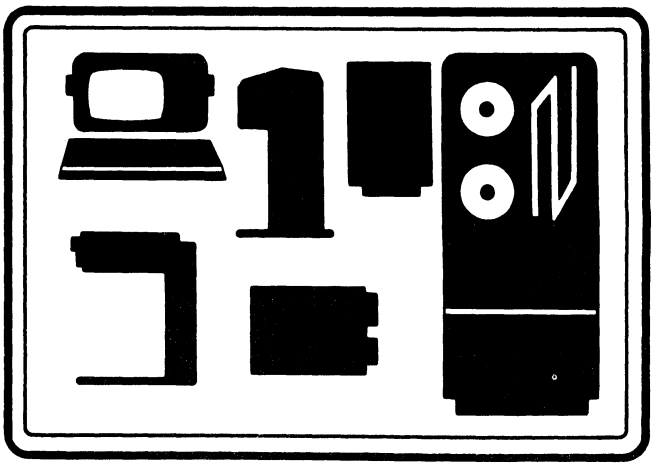
SH109/VS – AOS/VS OPERATOR TRAINING

STUDENT HANDOUT









019-000102-01

TABLE OF CONTENTS

Title	Page
Course Summary	v
Module 1 - Minicomputer Concepts	1-1
Module 2 - System Startup and Shutdown	2-1
Module 3 - Routine Operation and Maintenance of Peripherals	3-1
Module 4 - Introduction to AOS/VS	4-1
Module 5 - Communicating With AOS/VS Through the Command Line Interpreter	5-1
Module 6 - File Organization	6-1
Module 7 - AOS/VS Process Concepts	7-1
Module 8 - The EXEC Process	8-1
Module 9 - Operator Information Utilities	9-1
Module 10 - System Backup	10-1
Module 11 - System Problems	11-1
Appendix	A-1

PREREQUISITES

S100 or equivalent

COURSE OBJECTIVES

Provided prerequisites are met, the student successfully completing this course should be able to:

1. Discuss basic minicomputer concepts.
2. Power up, operate and power down a typical AOS/VS configuration.
3. Perform operator-level routine operations and maintenance on peripherals.
4. Communicate with the AOS/VS system through the Command Line Interpreter.
5. Recognize and distinguish among various logical disc structures.
6. Control the AOS/VS process environment by using process type and priority.
7. Use the log-on, batch and spooling features of EXEC.
8. Invoke specific system utilities and/or user programs.
9. Maintain system files through back-up facilities.
10. Interpret system error messages and take appropriate action.
11. Operate the MV/Family System Control Processor.

MODULE 1

MINICOMPUTER CONCEPTS

MODULE 1
MINICOMPUTER CONCEPTS

OBJECTIVES

Upon successful completion of this module, the student will be able to:

1. List and define the major components of the CPU.
2. List and define the functions and organization of memory.
3. State the purpose of a controller and list the types of controllers.
4. State the reserved device names.
5. Describe the major functions of an operating system.

PARTS OF A COMPUTER

I/O

TELEPHONE

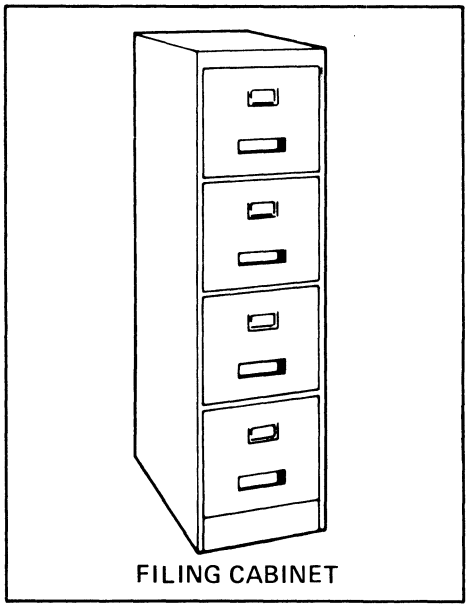
- PASS INFORMATION BACK AND FORTH BETWEEN SECRETARY AND ENVIRONMENT

CPU

SECRETARY

- MANAGES THE OPERATION
- CHANGES THE INFORMATION

MEMORY



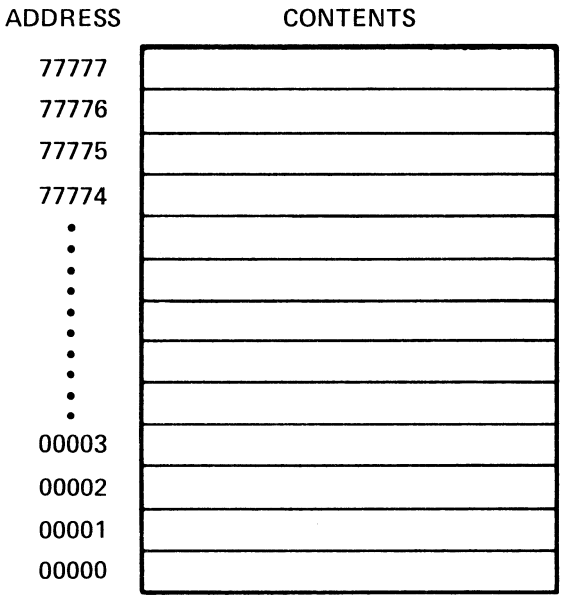
FILING CABINET

- STORES INFORMATION FOR FUTURE USE

CS-01373

Figure 1.1 Parts of a Computer

MEMORY STORES INFORMATION



CS-01374

Figure 1.2 Memory

EACH MEMORY LOCATION HAS TWO PARTS.

1. Address - where the word is
2. Contents - what the word is

THE CONTENTS CAN BE ONE OF TWO TYPES OF WORDS.

1. Instructions

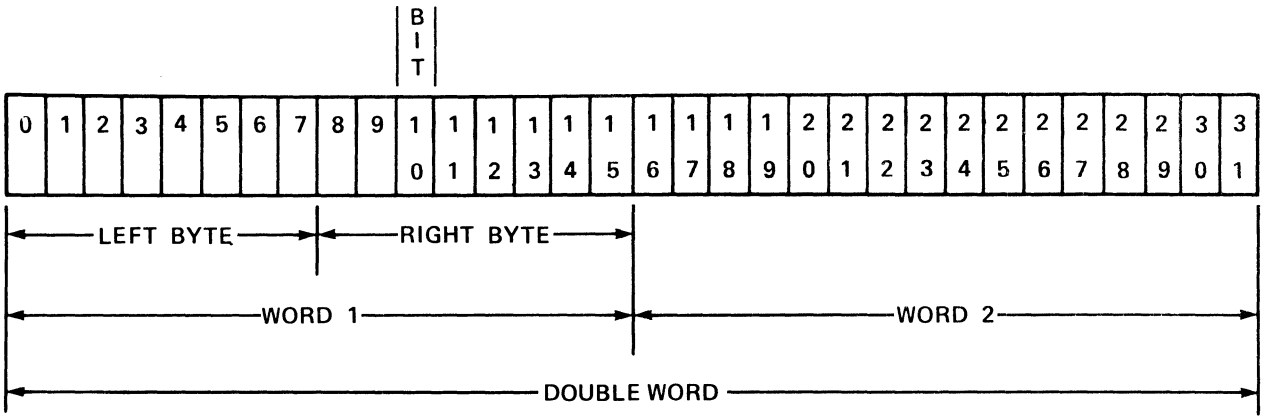
   EXAMPLE:

Instruction	As stored
XWLDA 2,B01	01100110000100100001100111000001

2. Data

   EXAMPLE:

Decimal value	As stored
4872	00000000000000000001001100001000



CS-01516

Figure 1.3 Memory Word Structure

BIT

- Defines one binary position in memory
- Represents a "1" or a "0"

BYTE

- Eight bits
- Accomodates one ASCII character

WORD

- 16 bits (two bytes)
- Accomodates an instruction or one data word

DOUBLE WORD

- 32 bits
- Accomodates an instruction or one data word

KB

- (kilo-byte) 1024 bytes (1KB)

KW

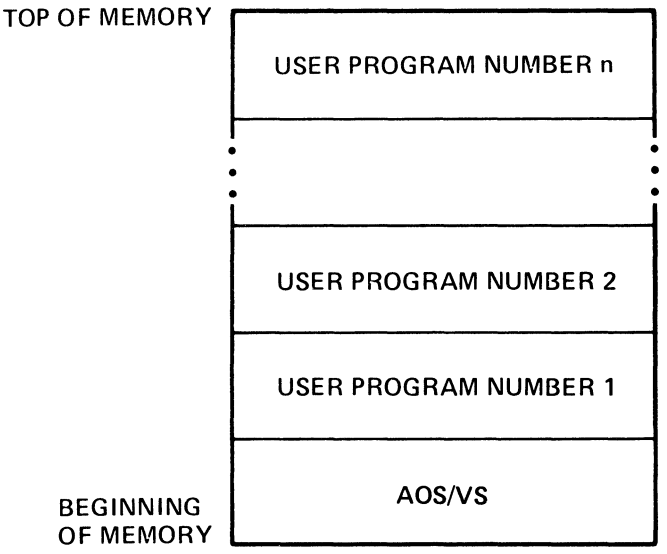
- (kilo-words) 1024 words (1KW)

MB

- (mega-byte) 1,048,576 bytes (1MB)

BLOCK — 512 BYTES (4096 BITS)

MEMORY CAN HOLD MORE THAN ONE PROGRAM



CS-01517

Figure 1.4 Programs in Memory

- AOS/VS allows more than one program to be in memory at one time. This is called Multiprogramming.
- Only one program can actually be executing at any instant.

AOS/VS 32 BIT PROCESS ADDRESS SPACE

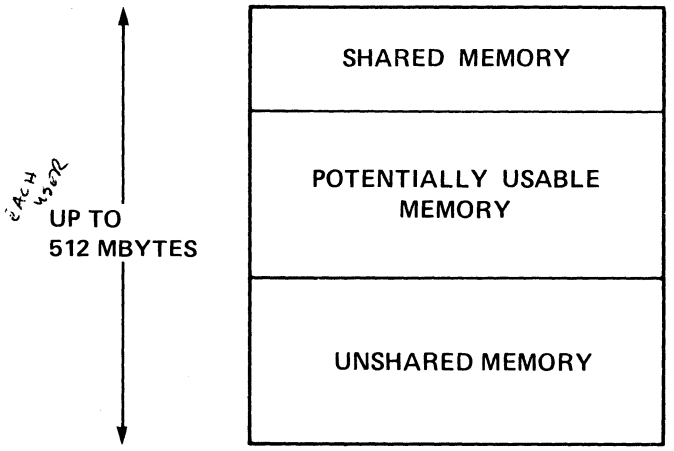
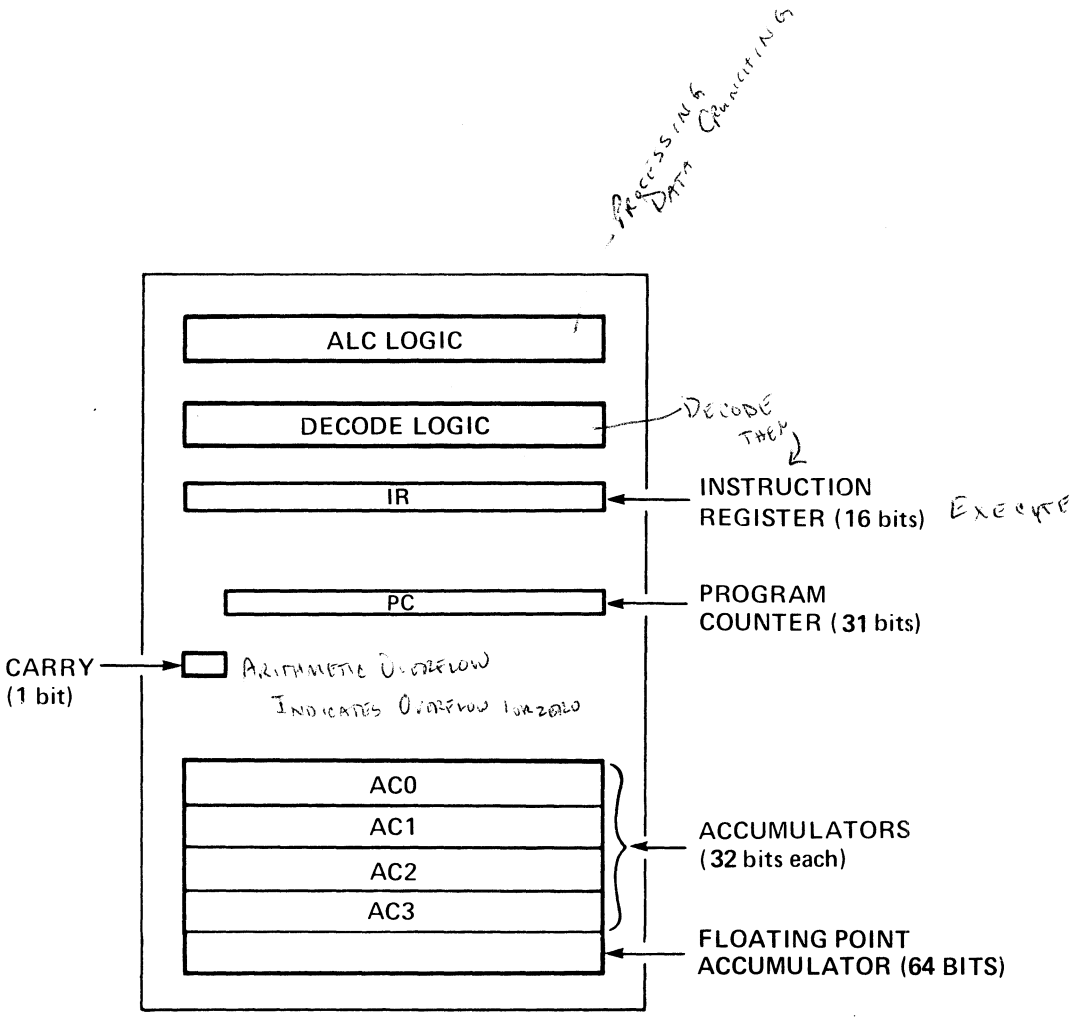


Figure 1.5 Address Space

- Note that this size exceeds the MV/Family physical memory size. AOS/VS with hardware support of the MV/Family computer only brings into memory (from disc) the memory pages that are currently required by an executing program. This memory management technique is called DEMAND PAGING.
- A page is defined as a contiguous block of 2048 bytes. The pages currently in memory for a given process are called the WORKING SET of that process.
- The AOS/VS 16 bit process address space is equivalent to the AOS process address space.



CS-01528

Figure 1.6 Central Processing Unit

- The Central Processing Unit (CPU) manages the computer.

TYPES OF DATA TRANSFERS

BYTE TRANSFERS

- One character transferred at a time

DATA CHANNEL

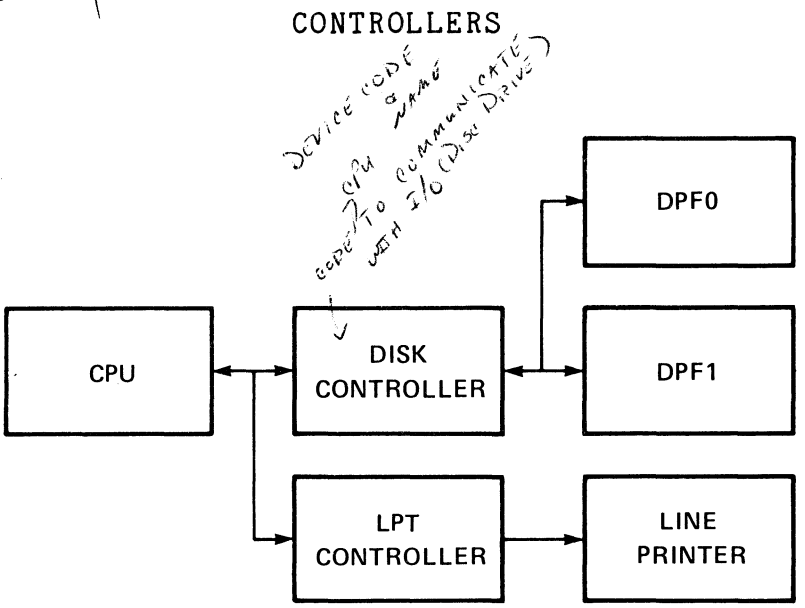
- Group of words (one BLOCK) transferred at a time

BYTE DEVICES

- Consoles
- Some line printers

DATA CHANNEL DEVICES

- Discs
- Magnetic tape
- Some line printers

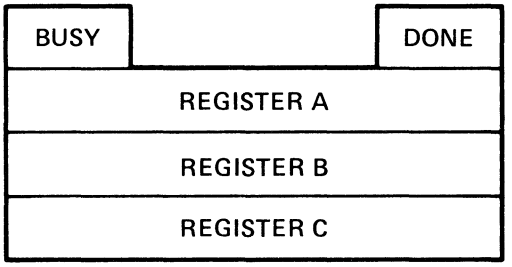


CS-01378

Figure 1.7 Controllers

- A controller is an interface between the CPU and a peripheral.
- A controller may be able to handle more than one device.

TYPICAL CONTROLLER REGISTERS



CS-01379

Figure 1.8 Controller Register

- If BUSY = 1 and DONE = 0, the device is in use.
- If BUSY = 0 and DONE = 1, the device is ~~idle~~.
- Data is transferred using one of the three registers.
- Controllers make all peripherals look alike to the CPU.

Appendix B DEVICE CODES

- A DEVICE CODE is a six-bit binary code that has been assigned to a peripheral to uniquely identify it.
- The device code is what the CPU uses to "talk" to a peripheral.
- During bootstrapping, the device code is used to determine where the program to be loaded "lives".
- Some examples of device codes are:

Table 1.A Device Codes

DEVICE	DEFAULT OCTAL CODE
First console input device	10
First console output device	11
First disc controller	27 or 33
Second disc controller	67 or 73
First line printer controller	17
Second line printer controller	57
First magnetic tape controller	22
Second magnetic tape controller	62

NOTE: The second controller's device code (if one exists) is always the first controller's device code + 40 .
8

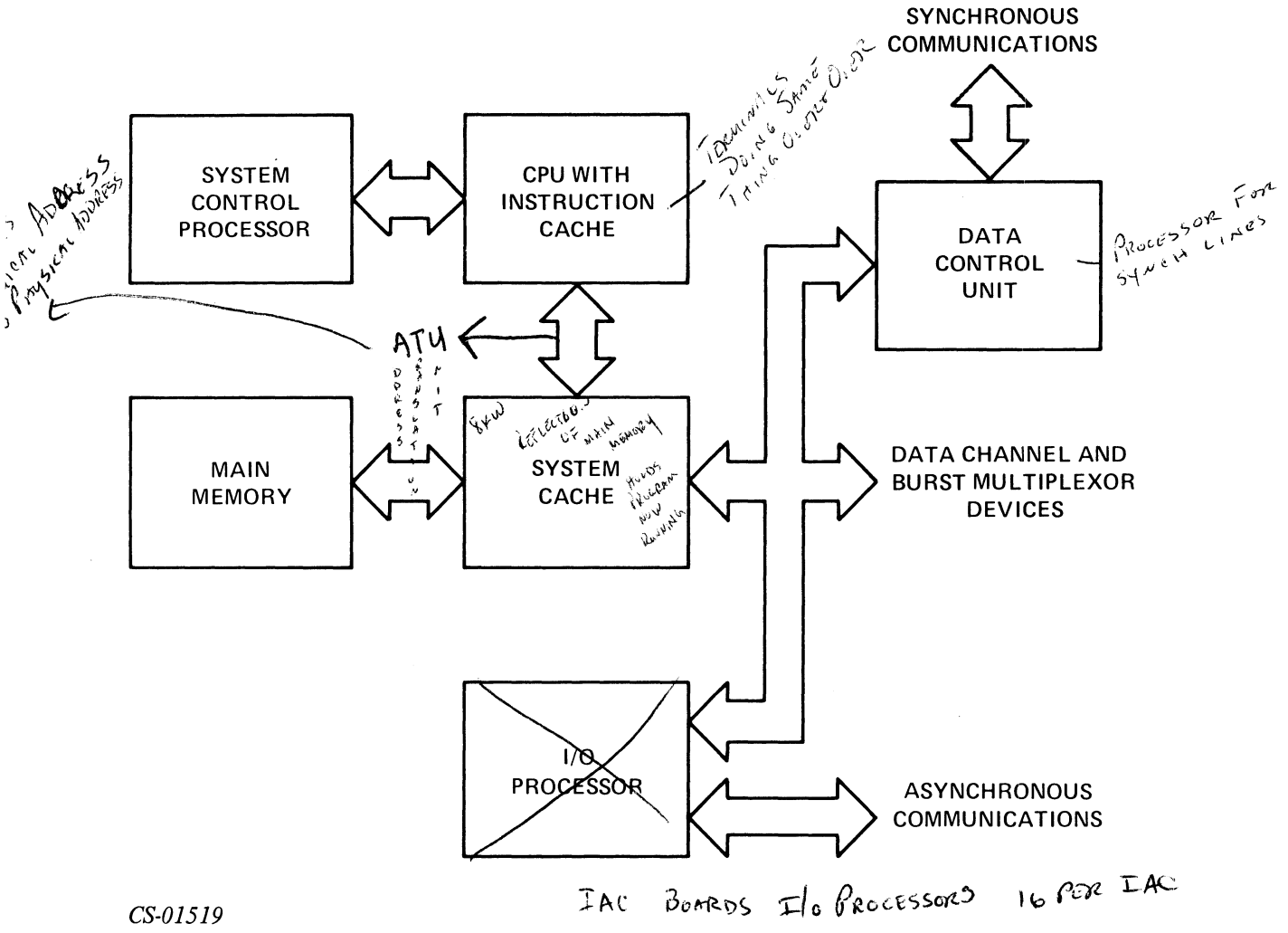
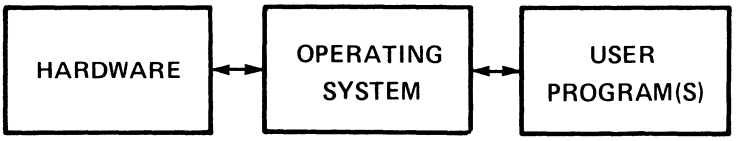


Figure 1.9 MV/Family Basic Architecture



CS-01380

Figure 1.10 Operating System Interface

- A vendor-supplied collection of software routines
- A software interface between the user programs and the hardware
- A sophisticated program

OBJECTIVES OF AN OPERATING SYSTEM

- Supports program development
- Controls access to peripherals
- Manages memory
- Allocates CPU time to multiple users
- Guarantees data and program security
- Insulates application software from hardware changes

WHAT IS AOS/VS?

I. Definition - A multitasking, multiprogramming, demand paged, virtual storage operating system.

II. Characteristics

A. Supports time-sharing, real-time and batch operations.

B. Consists of easy-to-use program development modules.

C. Supports a maximum of 255 processes.

1. A process is a tool used by AOS/VS to allocate system resources "intelligently".

2. A process is composed of system resources, including:

a. Memory

b. CPU time

c. Files

d. I/O

D. Supports dynamic memory management and demand paging.

E. Allows for secure data management.

F. Supports 512 MBYTE address space per user.

MODULE 2

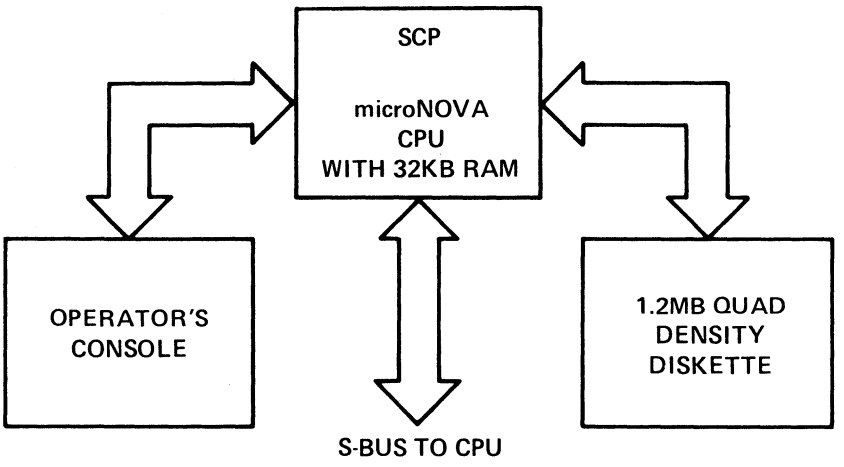
SYSTEM STARTUP AND SHUTDOWN

MODULE 2
SYSTEM STARTUP AND SHUTDOWN

OBJECTIVES

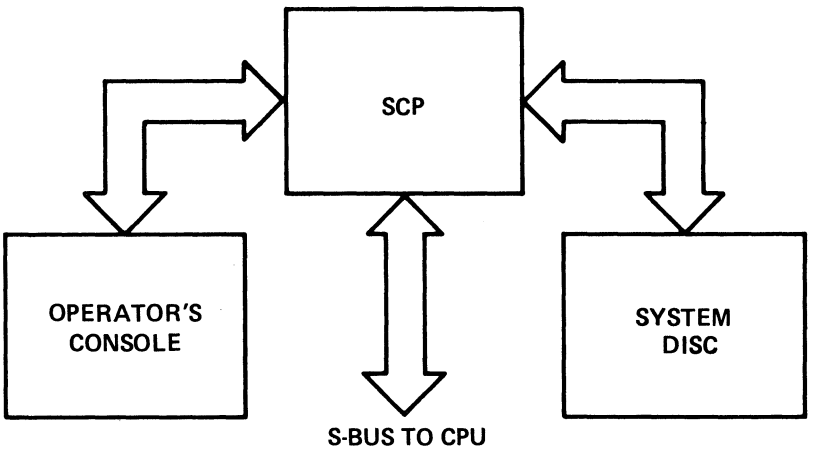
Upon successful completion of this module, the student will be able to:

1. Power on all system devices (demonstration in lab).
2. Bring up the System Control Processor (SCP).
3. Control the SCP through the SCP Command Line Interpreter.
4. Perform the proper steps to bring up the correct AOS/VS system.
5. Perform the proper actions to shut down the system.



CS-01520

a. MV/8000



CS-01756

b. MV/6000

Figure 2.1 System Control Processor (SCP)

SCP FEATURES

1. Primary control interface to CPU

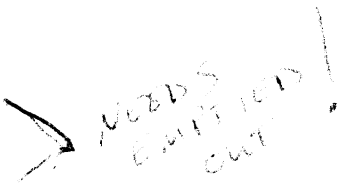
    - RUN
    - HALT
    - EXAMINE
    - MODIFY

2. Micro-code loader

    - LOAD
    - VERIFY

3. Hardware error logger

    - Memory errors
    - Cache errors, etc.



4. System integrity monitor (MV/8000 only)

    - Air flow
    - Temperature
    - Power supply margins

5. Runtime diagnostics

    - Power up
    - Microdiagnostics

6. Operator's console

    - Both soft console and AOS/VS OP console

SCP COMPONENTS

- A microNOVA® board computer (MBC) with 32 Kbytes of RAM and 4 Kbytes of ROM
- A 1.2 Megabyte diskette drive (MV/8000 only)
- A diskette controller board (MV/8000 only)
- A hard-copy or video operator's console
- A console control (CC) board that is the interface between the MBC and the MV/Family computer
- An SCP-OS diskette (MV/8000) or mag tape (MV/6000) containing:

    1. The SCP operating system.
    2. The HELP file.
    3. MV/Family microcode.
    4. Initial MV/Family scratchpad values.
    5. Diagnostic programs.

SCP OPERATING MODES

TTY MODE - When the SCP is in TTY mode, it is a system console. You cannot get into this mode unless the MV computer is running. Enter TTY mode from COMMAND mode with the TTY command.

COMMAND MODE - When the SCP is in COMMAND mode the SCP uses the console.

VARIOUS WAYS TO GET INTO TTY MODE

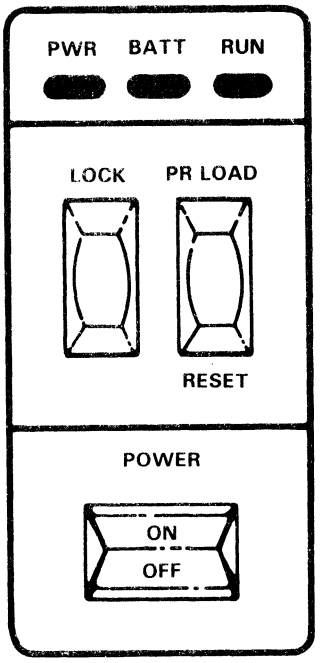
- Typing TTY from the SCP-CLI
- Typing LOCK from the SCP-CLI - ONLY 10S & OLD BOOTS
- Using the CONTINUE command from the SCP-CLI
- Using the BOOT command from the SCP-CLI
- Using the START command from the SCP-CLI

VARIOUS WAYS TO GET INTO COMMAND MODE

- Striking the BREAK key (or CMD + BREAK) on the master console
- Execution of a HALT instruction by the MV/Family computer.
- Resetting the CONSOLE RESET switch on the front panel of the processor. (MV/8000)
- By powering up the processor
- Through a hard hardware fault

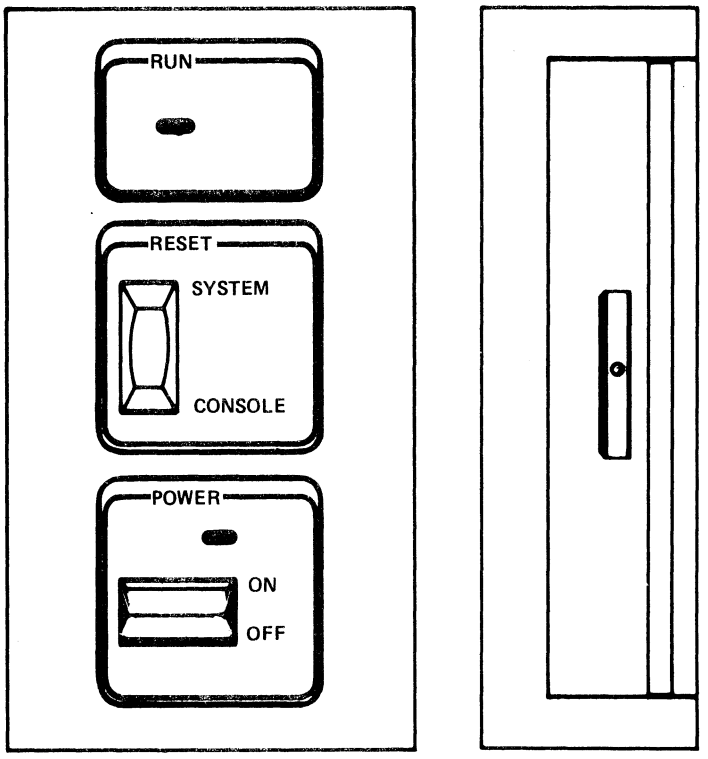
SCP PROMPTS

SCP-CLI>	SCP Command Line Interpreter
SCP-ERR>	SCP ERRORLOG utility
SCP-MCODE>	SCP MCODE utility
SCP-MEM>	SCP MEMORY utility (MV/8000 only)
*	SCP DTOS utility
!	ROM Octal Debugging Tool (ODT)
)	MV/8000 operating system, AOS/VS CLI (TTY MODE)



CS-01757

a. MV/6000 Front Panel



CS-01529

b. MV/8000 Front Panel

Figure 2.2 Front Panels

POWER UP SEQUENCE
MV/8000

1. Insert the SCP-OS diskette.

2. Press the POWER ON switch located on the front panel of the MV/8000. The power-on indicator above the switch will light up.

3. The SCP now performs the POWER UP tests. These programs, stored in ROM, determine whether the SCP is functioning properly. If it is, you receive the following message:

```
**CONSOLE READY**

MV/8000 SYSTEM CONTROL PROGRAM
TYPE H<CR> FOR HELP
COPYRIGHT DATA GENERAL .....
STARTING POWER UP SEQUENCE
```

4. ENTER DATE (MO DAY YR)
   You must enter the date. There is no default.

5. ENTER TIME (HR MIN SEC)
   You must enter the time. There is no default.

6. SCP will then load the MICRO-CODE and the SCRATCH-PAD. You will receive the following messages:

```
LOADING MCODE FROM FILE MV8000 REV n.n
LOADING COMPLETE
VERIFYING MCODE FROM FILE MV8000 REV n.n
VERIFICATION COMPLETE
LOADING SCRATCH-PAD FROM FILE MV8000 REV n.n
LOADING COMPLETE
VERIFYING SCRATCH-PAD FROM FILE MV8000 REV n.n
VERIFICATION COMPLETE
```

7. Next the SCP will go through system initialization.

```
USTORE    REV n.n
BEGINNING SYSTEM INITIALIZATION
# OF 256 KB MEMORY MODULES  - 08
END SYSTEM INITIALIZATION

SCP-CLI>
```

You are now in COMMAND mode and ready to load the MV/8000 operating system (AOS/VS).

POWER UP SEQUENCE
MV/6000

1. Load and power on the System disc.

2. Press the CPU LOCK switch to LOCK.

3. Turn the CPU POWER switch ON. The POWER lamp should light.

4. Now, EPROM code in the SCP does some power up tests. When these succeed, the system console displays:

```
**POWER UP TESTING COMPLETED**
```

5. The SCP then loads the disc bootstrap, which in turn loads SYSBOOT, from disc. SYSBOOT says

SPECIFY EACH DISK IN THE LDU DISK UNIT NAME?	Enter the name of the Master Logical disc. Generally DPF0.
DEVICE CODE?	Enter a new-line, if default device codes were used.
uCode FILE [:MV6000.MCF]?	Enter a new-line, unless you want to enter microcode from another file.

SYSBOOT will load the SCP operating system from the master logical disc. The SCP-OS will then load microcode into the main processor. You will receive the following messages:

```
MV/6000 SYSTEM CONTROL PROGRAM REV n
COPYRIGHT DATA GENERAL....
CHECK-SUM OK

BEGIN SYSTEM INITIALIZATION
# OF 256 KB MEMORY MODULES - n
END SYSTEM INITIALIZATION

LOADING MV/6000 MICROCODE REV n

SYSTEM PATHNAME?
```

Before entering SYSTEM PATHNAME, you can get to the SCP-CLI by pressing CMD and BREAK, or BRK.

```
SCP-CLI>
```

When you turn CPU power on, EPROM code in the SCP performs some power up tests on both CPUs. If they pass these tests, you see the messages:

CONSOLE READY	on MV/8000 systems
POWER UP TESTING COMPLETED	on MV/6000 systems

If the SCP fails a test, the program stops testing and the console displays those letters in the message representing the tests it has passed.

EXAMPLE:

CONSOL_

This would indicate that the SCP failed its temperature control test.

Table 2.A SCP Power Up Test -- MV/8000

Message	Faulty Component(s)
(none)	MBC CPU and/or TTO circuit boards
_	MBC CPU circuit board
C_	MBC TTO circuit board
CO_	CC circuit board
CON_	Microsequencer or CC board
CONS_	CC circuit board
CONSO_	Power supply
CONSOL_	Power supply - Temperature is too high
CONSOLE_	Power supply - Air flow is restricted
CONSOLE R_	MBC RAM circuit board
CONSOLE RE_	CC, ATU, or Microsequencer circuit boards
CONSOLE REA_	Microsequencer circuit board
CONSOLE READ_	Diskette controller board

Table 2.B SCP Power Up Test -- MV/6000

Message	Faulty Component(s)
(none)	SCP circuit board
_	SCP circuit board
P_	SCP circuit board
PO_	SCP circuit board
POW_	SCP, Microsequencer, or Bank Controller
POWE_	SCP circuit board
POWER_	SCP circuit board
POWER UP_	Microsequencer, SCP, or ATU board
POWER UP T_	Microsequencer or SCP board
POWER UP TE_	ALU, Microsequencer or SCP board
POWER UP TES_	System Cache, Instruction Processor or ATU
POWER UP TEST_	System Cache, SCP or I/O Channel (IOC)
POWER UP TESTI_	ATU or ALU-1 board
POWER UP TESTIN_	System Cache, ALU-1 or ATU board
POWER UP TESTING_	System Cache, ALU-1 or ATU board
POWER UP TESTING C_	System Cache, Bank Controller or Memory Module 0 board
POWER UP TESTING CO_	System Cache, SCP or IOC board
POWER UP TESTING COM_	Instruction Processor (IP) or ALU-1 board
POWER UP TESTING COMP_	Displayed with a numeric error code 0 = IP, ALU-1 or ATU boards 1 = IP, ALU-1 or ATU boards 2 = IP, ALU-1 or Microsequencer 3 = System Cache, Bank Controller or Memory Module 0 boards 4 = IOC, SCP or ALU-1 boards 5 = IOC, SCP or ALU-1 boards

PROMPT - SCP-CLI>

Table 2.C SCP CLI Commands

HELP	CONTROL	EXAMINE/MODIFY	SCP-OS
HELP	BOOT CONTINUE HALT ISTEP RESET START TRACE	DISPLAY EXAMINE STATUS or (.)	DEGRADE FLAGS LOCK TIME TTY XEQ

- The HELP command provides information about SCP commands and utilities.
- CONTROL commands start, stop, and load programs stored in the MV/Family memory.
- EXAMINE/MODIFY commands allow you to examine and modify the MV/Family main memory and registers. You can access the MV/Family memory by giving either the logical or physical address.
- SCP-OS commands permit you to run utility programs, to set internal SCP flags, and to affect the state of the SCP and the MV/Family computer.

Some commands require that the MV/Family computer be either running or halted. Others do not care. If you try to execute a command that requires the MV computer to be in a state other than the one it is in, the command will not execute and you will get an error message.

Table 2.D SCP CLI Flags

NAME	VALUE	DESCRIPTION
AUTO	Y,N	Setting this flag causes the SCP to enter TTY mode automatically after it has executed A START, BOOT or CONTINUE command.
ELOG	Y,N	With this flag you can turn the SCP logging function on or off.
LOCK	N,Y	If this flag is set, the SCP automatically sets its software lock when it enters TTY mode.
PHYS	N,Y	If this flag is set, the SCP assumes that main memory references in the DISPLAY and EXAMINE commands have physical addresses.
RADIX	8,16	This flag determines the radix of data entered or displayed by SCP commands.
SCOPE	N,Y	Set this flag if you have a video console.
SING	N,Y	If you set this flag, the SCP displays only the least significant 16 bits of the cell you are examining.

EXAMPLE:

```
SCP-CLI> FLAGS SCOPE YES
```

With this command you can bypass the system cache, the instruction cache, or both. You can also slow down the ATU, requiring it to translate every memory reference.

Table 2.E SCP DEGRADE Command

NAME	VALUE	DESCRIPTION
ATU	N,Y	When Y, the ATU must translate every memory reference. The ATU cannot store previous translations.
SCACHE	N,Y	When Y, the MV/Family computer bypasses the system cache.
ICACHE	N,Y	When Y, the MV/Family computer bypasses the instruction cache.

EXAMPLE:

```
SCP-CLI> DEGRADE ATU YES
```

```
SCP-CLI> RESET

SCP-CLI> BOOT 27
```

- After performing these steps the system console will display the following prompts which you must answer:

  SPECIFY EACH DISK IN LDU

  DISK UNIT NAME? Enter the name of the master logical disc. If you do not know the name of the master logical disc, ask the system manager.

  DEVICE CODE? Enter a new-line if default device codes were used otherwise enter the octal device code that has been assigned to the disc named in the DISK UNIT query.

  Initializing AOS/VS -- All systems

  SYSTEM PATHNAME? Type a new-line unless you want to bring up a system other than the one the system manager has installed.

- The specified AOS/VS system will be loaded and then the following will be displayed:

  AOS/VS REV xx.xx.xx.xx — Where xx.xx is the revision number of the installed system.

  MASTER LDU: name

  DATE (mm/dd/yy)? — Enter the data in mm/dd/yy format.

  TIME (hh:mm:ss)? — Enter the time in 24-hour format.

  OVERRIDE DEFAULT SPECS [N]? — Enter NO and a new-line, unless the system manager tells you otherwise.

  AOS/VS CLI REV nn date time
)

- After the CLI prompt appears, the system is up and system processing can begin.

UP MACRO

The UP marco is a file containing the necessary commands needed to finish bringing up the system.

The macro may vary from system to system.

EXECUTION:

From the operator console

```
)UP
```

COMMUNICATING WITH USER CONSOLES

Every process has a unique Process ID (PID) and name.

EXAMPLE:

```
PID 2 is OP
```

You can send messages to others by using the SEND command.

EXAMPLES:

```
)SEND @CON4 HI THERE

)SEND @CON-  THE SYSTEM WILL BE COMING DOWN IN 10 MINUTES!!
```

NOTE: SEND gives you the PID of the sender. Using the WHO command, you can find out what console the sender is on, and return a message.

SYSTEM SHUTDOWN DIALOG

To shut down AOS/VS, simply type "BYE" at the operator's console.

An example of a shutdown dialogue follows. The operator's input is in lower case and the system prompts are in upper case.

```
) bye
YOU HAVE SONS.  DO YOU WANT TO TERMINATE?  yes
DO YOU REALLY WANT TO SHUT THE SYSTEM DOWN?  yes
STARTING SYSTEM SHUTDOWN        4-MAY-80          17:16:50
SYSTEM SHUTDOWN
```

DOWN MACRO

The DOWN macro is a file containing necessary commands to shut down the system.

The macro may vary from system to system.

EXECUTION:

From the operator console

```
)DOWN
```

MODULE 3

ROUTINE OPERATION AND MAINTENANCE OF PERIPHERALS

MODULE 3
ROUTINE OPERATION AND MAINTENANCE OF PERIPHERALS

OBJECTIVES

Upon successful completion of this module, the student will be able to maintain operational capability, at the operator level, of the following devices:

1. Lineprinter
   a. Data Products
   b. DASHER®
2. Magnetic tape units
3. CPU
4. Disc units
5. Consoles

GENERAL PREVENTIVE MAINTENANCE CONSIDERATIONS

- Do not use the equipment as a counter top.
- Always check the line voltage before plugging equipment into an unknown socket.
- Do not get metal filings into the equipment. Never clean equipment room with steel wool.
- Never allow routing cables to become strained, cramped or crushed underfoot.
- Too much heat, moisture or contaminates can be harmful to computer equipment.

GENERAL CLEANING

- Clean external surfaces of all equipment with a soft, lint-free cloth once a week.
- You may dampen the cloth with soap and water.
- Rinse any soap off with a damp cloth.
- Buff dry with a clean dry cloth.
- NEVER use alcohol or cleaning solvents.

DISCS AND DISC DRIVES

REFER TO:

OPERATOR'S REFERENCE SERIES - DGC Disc Drives #014-000099

FOR INFORMATION ON:

1. Handling disc cartridges and packs
2. Operator controls
3. Operating procedures
    a. Loading a disc
    b. Unloading a disc
    c. Starting the disc drive
    d. Changing the disc
    e. Shutting down the disc drive
4. Maintenance

NOTE: The instructor will demonstrate typical disc operations in the lab.

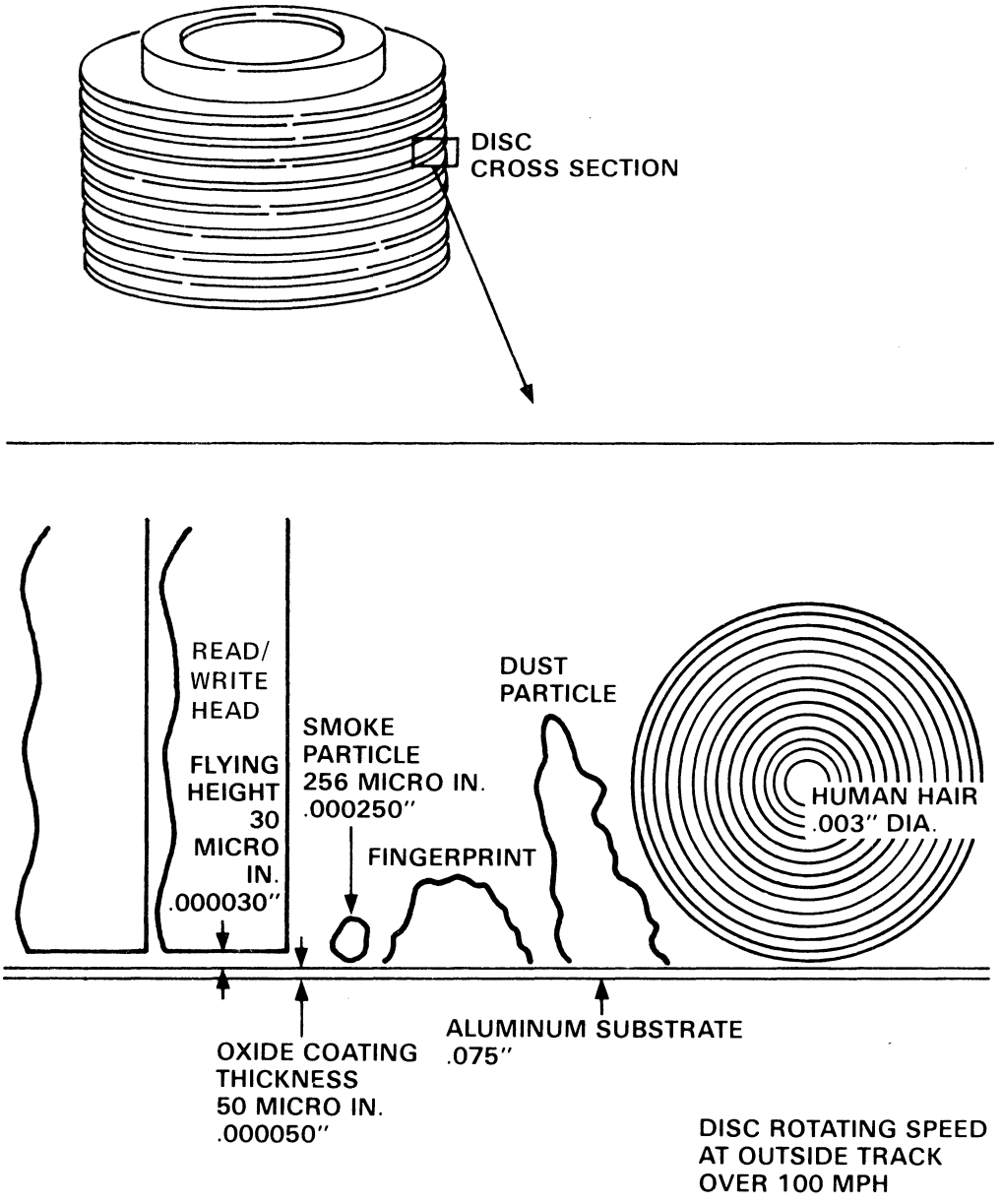


Figure 3.1 Disc Surfaces and Contaminates

MAGNETIC TAPE TRANSPORT

REFER TO:

- Magnetic Tape Transports - Operator's Manual #014-000095

FOR INFORMATION ON:

1. Tape markers
2. Operator controls
   a. Front panel
   b. Back panel
3. Operating procedures
   a. Loading tape
   b. Backward loading
   c. Unloading tape
4. Cleaning

NOTE: The instructor will demonstrate the above procedures in the lab.

REFER TO:

OPERATOR REFERENCE SERIES

DASHER DISPLAY Terminals 6052/6053 #014-000090

DASHER DISPLAY Terminals D100/D200 #014-000641

FOR INFORMATION ON:

1. Operator Controls
   a. Front and rear panel
   b. Keyboard
2. Maintenance

PRINTERS

REFER TO EITHER:

DASHER TP1 Terminal/Printers #14-000088

DASHER LP2-TP2 Printers #14-000093

or

DGC Line Printers, # 14-000089

FOR INFORMATION ON:

1. Operator controls - switches and indicators
2. Operating procedures
   a. Loading paper
   b. Changing the ribbon
3. Routine upkeep
   a. Cleaning
   b. Maintenance
   c. Operating notes

NOTE: The instructor will demonstrate typical operations in the lab.

MODULE 4

INTRODUCTION TO AOS/VS

MODULE 4
INTRODUCTION TO AOS/VS

OBJECTIVES

Upon successful completion of this module, the student will be able to state the purpose and function of AOS/VS.

WHAT IS AOS/VS?

- A general-purpose, disc-based operating system that controls and monitors user-program processing on a Data General MV/Family computer.
- A manager of many program control, input/output, and file access functions.
- You manage AOS/VS, and AOS/VS manages the computer.

AOS/VS RESPONSIBILITIES

BOOKKEEPING:

- What is coming in?
- Where is new information to be stored?
- What old jobs do I have?
- What jobs do I have to do?

SUPERVISING:

- Is a job allowed to get the information it wants?
- Which job runs next?
- Can a job write to the space it wants?

TYPICAL AOS/VS APPLICATIONS

- REAL-TIME
- BATCH
- TIME-SHARING

THE RESPONSIBILITIES OF THE OPERATOR

The OPERATOR is a part of AOS/VS who performs the following:

- Wakes up AOS/VS and hardware.
- Tells AOS/VS what jobs have to be run.
- Tells AOS/VS which jobs are more important.
- Tells AOS/VS what I/O devices a job may use.

The OPERATOR backs up important information for recovery in case of failure.

INTRODUCTION TO A PROCESS

1. A user of SYSTEM resources
2. Each PROCESS consists of many components, including:
   a. PID - Process I.D. #
   b. Simple process name
   c. Username
   d. Memory
   e. Program

MODULE 5

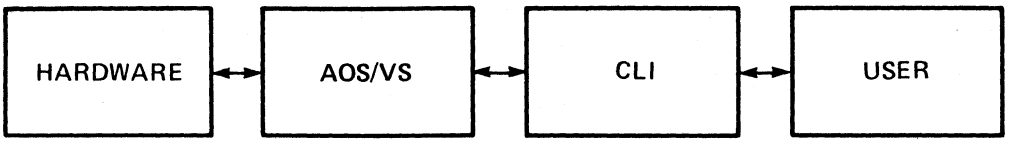
COMMUNICATING WITH AOS/VS THROUGH THE COMMAND LINE INTERPRETER

MODULE 5
COMMUNICATING WITH AOS/VS THROUGH THE COMMAND LINE INTERPRETER

OBJECTIVES

Upon successful completion of this module the student will be able to:

1. State the function of the Command Line Interpreter (CLI).
2. Generate syntactically correct CLI commands.
3. Use selected CLI commands.
4. Use the control character sequences.
5. Respond properly to errors.
6. Use the features provided by SCREENEDIT.
7. Employ expansion capabilities.
8. Set the terminal characteristics.



CS-01521

Figure 5.1 The Command Line Interpreter (CLI) Interface

- The Command Line Interpreter (CLI) is an interface to the operating system.
- The user talks to AOS/VS by entering CLI commands from the console.
- The CLI commands can invoke user programs and system utilities, control files and peripheral devices.
- The CLI commands can set, display or control the CLI environment.
- The CLI can be used in both batch or interactive mode.

1. When commanded to monitor a console, EXEC waits for the NEW LINE key to be pressed on that console.

2. After NEW LINE is pressed, EXEC prompts

   ```
   USERNAME:

   PASSWORD:
   ```

   User inputs name and password.

3. EXEC

   - Acknowledges logon
   - Reports last logon
   - Types the contents of a log-on message file

4. AOS/VS CLI is then executed and displays:

   ```
   AOS/VS CLI REV xx.xx 2-FEB-81 8:57:25
   )
   ```

COMMAND LINE SYNTAX

GENERAL FORM:

COMMAND[/SWITCH.../SWITCH],[ARGUMENT[/SWITCH]...ARGUMENT]

Where:

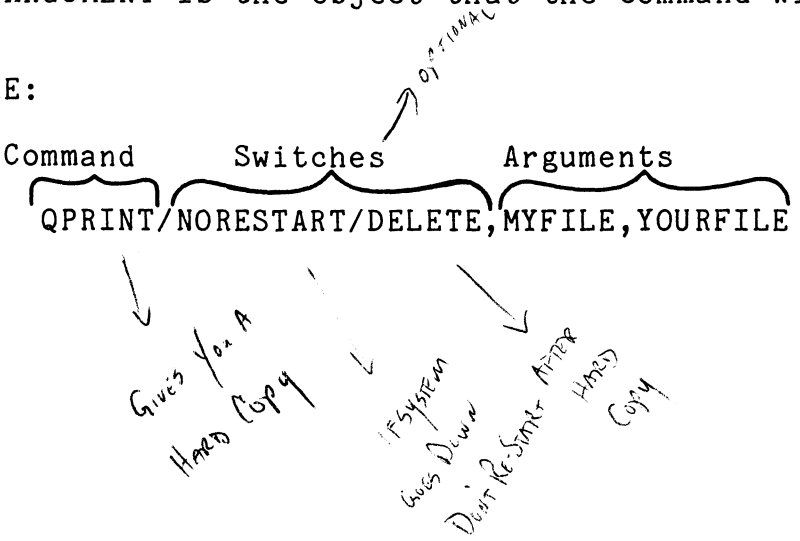
[ ] indicates an option.

COMMAND is the operation to be performed.

SWITCH is a slash (/) followed by a letter, number or word which changes the default action of the CLI command or argument.

ARGUMENT is the object that the command will affect.

EXAMPLE:



)TYPE, FILENAME) - ON CRT - NOT PRINTER

A command may consist of:

- A command
- A command and one argument
- A command and multiple arguments

EXAMPLES:

) BYE	logs the user off
) DELETE,FILE1	deletes file FILE1
) DELETE,FILE1,FILE2	deletes files FILE1 and FILE2

Commands and arguments may be truncated to the number of characters necessary to uniquely identify it to CLI.

EXAMPLES:

X or XE or XEQ are legal for the XEQ command which executes a program.

DE is not a legal abbreviation since DELETE, DEBUG and DEASSIGN all begin with the same character combination.

To be unique they would be:

DEL for DELETE

DEB for DEBUG

DEA for DEASSIGN

SWITCHES

- A switch is a slash (/) followed by either a letter or number, or a keyword and a value.
- Switches modify the default action taken by the command.
- Arguments may also have switches.

EXAMPLES:

1. A single letter or number is called a SIMPLE switch.

```
DELETE/V,MYFILE          Delete MYFILE and verify
                         the deletion on the console.
```

2. A keyword and value switch use a standard KEYWORD and a value for that keyword during the execution of the command.

```
QPRINT/COPIES=3,MYFILE   Print three copies of MYFILE
                         on the line printer.
```

DELIMITERS

1. Commands must have a separator between the command and the first argument, and between each argument.

   Legal separators are:

   - Blanks (one or more)
   - A single comma
   - Tabs (TAB key) (one or more)

2. There must be a terminator at the end of the command line.

   Legal terminators are:

   - NEW LINE
   - CARRIAGE RETURN
   - FORM FEEL (CTRL L or ERASE PAGE)
   - END OF FILE (CTRL D)

3. A command may continue to the next line by preceding the the end of the line with an & (ampersand).

ERASE-PAGE KEY WILL EXECUTE COMMAND

CLI PROMPTS

PROMPT	SUPERPROCESS	SUPERUSER
)	OFF	OFF
*)	OFF	ON
+)	ON	OFF
#)	ON	ON

PROMPT COMMAND

Up to 8 CLI commands with no arguments or switches

```
)PROMPT [cli command]...
```

EXAMPLE:

```
)PROMPT,TIME;DATE

20-NOV-80
11:11:20
)
```

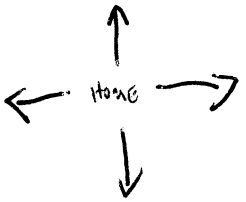
Entering a DEL or a RUB OUT will delete the character to the left of the cursor.

On softcopy terminals only:

- If SCREENEDIT is ON, it will allow you to correct or modify the last command line entered.
- Control characters are used to redisplay and correct, or modify the command.

Table 5.A SCREENEDIT Control Characters

CONTROL CHARACTER	EFFECT
CTRL A	Redisplay the last command and move to the end of the character string.
CTRL B	Move to the end of the previous word.
CTRL E	Enter/exit insert character mode. OVER WHERE YOU WANT TO INSERT — NO LIMIT
CTRL F	Move to the beginning of the next word.
CTRL H	Move to the beginning of the character string.
CTRL I	Insert a tab.
CTRL K	Erase everything to the right of the cursor.
CTRL X	Move the cursor one character to the right (the -> key has the same effect).
CTRL Y	Move the cursor one character to the left (the <- key has the same effect.)



EOL - ERASE END OF LINE - ERASE EVERYTHING TO RIGHT

CONTROL CHARACTERS

- The following control characters have a special effect on console activity.

Table 5.B Control Characters and Console Activity

CONTROL CHARACTER	EFFECT
CTRL D	End of file from keyboard.
CTRL O CTRL O	Do not display data on screen. Resume display after a CTRL O.
CTRL Q	Continue console display.
CTRL S	Freeze console display.
CTRL U	Cancel current input line.

CONTROL CHARACTER SEQUENCES

- The following control character sequences have a special effect on the AOS/VS activity.

Table 5.C Control Character Sequences and AOS Activity

CONTROL CHARACTER SEQUENCE	EFFECT
CTRL C CTRL A	Interrupt the process.
CTRL C CTRL B	Terminate the process.
CTRL C CTRL C	Echo ^C^C on screen (use to see if the system is still running.)
CTRL C CTRL E	Terminate process and create a breakfile.

- You can enter more than one command on a line by separating the commands with a ; (semi-colon).

EXAMPLE:

```
) TIME;DATE is equivalent to:

    ) TIME
    ) DATE
```

- One command can expand to many command lines by using parentheses.

EXAMPLES:

```
) DELETE,(FILE1,FILE2,FILE3,FILE4) is equivalent to:

    ) DELETE,FILE1
    ) DELETE,FILE2
    ) DELETE,FILE3
    ) DELETE,FILE4

) (TYPE,DELETE) FILE is equivalent to:

    ) TYPE,FILE
    ) DELETE,FILE
```

- One argument can expand to more than one argument by enclosing the "core" of the arguments in angle brackets.

EXAMPLES:

```
) DELETE,FILE<1,2,3,4> is equivalent to:

    ) DELETE,FILE1,FILE2,FILE3,FILE4
```

- The HELP command will give the user information about a CLI command or topic. The examples below are actual printout from the HELP command.

```
) HELP
TOPICS ARE:

*1_SWITCH       *2_SWITCH       *AFTER_SWITCH    *CLI_SWITCH     *COMMANDS
*CONDITIONALS   *CONTROL_CHARS  *CURSOR_CONTROL  *ENVIRONMENT    *EXCEPTIONS
*EXEC           *FILENAMES      *FORTRAN5        *GENERIC_FILES  *I_SWITCH
*LINKER         *LINKS          *L_SWITCH        *MACROS         *M_SWITCH
*NEWLINE        *ODM            *PATCH           *PATHNAMES      *PED
*PSEUDO_MACROS  *P_SWITCH       *QUEUES          *Q_SWITCH       *REPORT
*SWITCHES       *TEMPLATES      *TOPICS

FOR MORE HELP ABOUT ANY ITEM ABOVE, TYPE 'HELP' *ITEM


)HELP *CONTROL_CHARS
CONTROL
CHARACTER          EFFECT

CTRL-C             BEGIN A CONTROL CHARACTER SEQUENCE
CTRL-D             END-OF-FILE
CTRL-O             DISCARD DATA WRITTEN TO CONSOLE
CTRL-P             NO SPECIAL INTERPRETATION FOR NEXT CHARACTER
CTRL-Q             RESUME DISPLAY OF DATA ON CONSOLE
CTRL-S             FREEZE CONSOLE DISPLAY
CTRL-T             (RESERVED AND IGNORED)
CTRL-U             CANCEL CURRENT INPUT LINE
CTRL-V             (RESERVED AND IGNORED)
CTRL-C CTRL-A      INTERRUPT THE PROCESS IF IT HAS DEFINED
                   A CONSOLE-INTERRUPT SERVICE TASK
CTRL-C CTRL-B      TERMINATE THE PROCESS
CTRL-C CTRL-C      ECHO ^C^C ON THE CONSOLE
CTRL-C CTRL-E      TERMINATE THE PROCESS AND
                   CREATE A BREAK FILE
```

```
) HELP *COMMANDS
CLI COMMANDS ARE:

ACL             ASSIGN            BIAS          BLOCK        BYE
CHAIN           CHARACTERISTICS   CHECKTERMS    CLASS1       CLASS2
CONTROL         COPY              CPUID         CREATE       CURRENT
DATAFILE        DATE              DEASSIGN      DEBUG        DEFACL
DELETE          DIRECTORY         DISMOUNT      DUMP         ENQUEUE
EXECUTE         FILESTATUS        HELP          INITIALIZE   LEVEL
LISTFILE        LOAD              MESSAGE       MOUNT        MOVE
PATHNAME        PAUSE             PERFORMANCE   PERMANENCE   POP
PREVIOUS        PRIORITY          PROCESS       PROMPT       PRTYPE
PUSH            QBATCH            QCANCEL       QDISPLAY     QHOLD
QPLOT           QPRINT            QSUBMIT       QUNHOLD      RELEASE
RENAME          REVISION          REWIND        RUNTIME      SCREENEDIT
SEARCHLIST      SEND              SPACE         SQUEEZE      STRING
SUPERPROCESS    SUPERUSER         SYSID         SYSLOG       TERMINATE
TIME            TRACE             TREE          TYPE         UNBLOCK
WHO             WRITE             XEQ

FOR MORE HELP ABOUT ANY ITEM ABOVE, TYPE 'HELP ITEM'
```

```
) HELP CREATE
CREATE          - REQUIRES ARGUMENT(S)
  SWITCHES: /1= /2= /L(=) /Q    /LINK       /TYPE=          /I
/DYNAMIC        /DATASENSITIVE  /FIXED=     /VARIABLE       /M
/INDEXLEVELS=   /ELEMENTSIZE=   /DIRECTORY  /HASHFRAMESIZE= /MAXSIZE=

FOR MORE HELP TYPE
HELP/V CREATE



) HELP/V CREATE
CREATE  CREATE A FILE

FORMAT: CREATE PATHNAME <RESOLUTION-PATHNAME>
        WHERE RESOLUTION-PATHNAME IS THE PATHNAME
              TO WHICH THE LINK WILL RESOLVE
              (IF /LINK SWITCH IS USED)

COMMAND SWITCHES:  /1= /2= /L(=) /Q
        /DATASENSITIVE RECORD TYPE OF CREATED FILE
        /DIRECTORY     CREATE A DIRECTORY OR CONTROL POINT DIRECTORY
        /DYNAMIC       RECORD TYPE OF THE CREATED FILE
        /ELEMENTSIZE=N NUMBER OF DISK BLOCKS FILE WILL GROW BY
        /FIXED=N       RECORD TYPE OF THE CREATED FILE
        /HASHFRAMESIZE=N  SIZE OF HASH FRAME (DEFAULT 7)
                          (DIRECTORIES ONLY)
        /I             CONTENTS OF FILE TAKEN FROM @INPUT
        /INDEXLEVELS=N MAXIMUM NUMBER OF INDEX LEVELS (DEFAULT=3)
        /LINK          CREATE A LINK
        /M             FILE TAKEN FROM REMAINDER OF MACRO FILE
        /MAXSIZE=N     CREATE A CONTROL POINT DIRECTORY OF THE
                           SPECIFIED MAXIMUM SIZE (IN DISK BLOCKS) --
                           VALID ONLY WITH THE /DIRECTORY SWITCH
        /TYPE=N        CREATE A FILE OF THE SPECIFIED TYPE
                           (63 < N < 256)
        /VARIABLE      RECORD TYPE OF FILE CREATED
```

DEVICE CHARACTERISTICS

The CHARACTERISTICS command allows you to modify the way AOS/VS interprets input or modifies output to a device. This command is especially useful when using non-Data General devices.

The default characteristics are specified when the AOS/VS Operating System is created. These characteristics can later be overwritten with the CHARACTERISTICS command.

```
)CHARACTERISTICS/SWITCHES...  /OFF/SWITCHES../ON/SWITCHES.. DEVICES
```

FREQUENTLY USED SWITCHES:

/ON/PM

/ON/NRM

/LPP=N

/CPL=n

/HARDCOPY

MODULE 6

FILE ORGANIZATION

MODULE 6
FILE ORGANIZATION

OBJECTIVES

Upon successful completion of this module the student will be able to:

1. Recognize the components of a physical disc.
2. Define and create a file.
3. Define and distinguish between a directory and a control point directory.
4. Discuss the directory structure and trees.
5. Access a file through its pathname or by using the template facility.
6. Employ filename templates.
7. Create and use links.
8. Apply ACL capability to various file types.
9. Recognize CLI macro files and their uses.

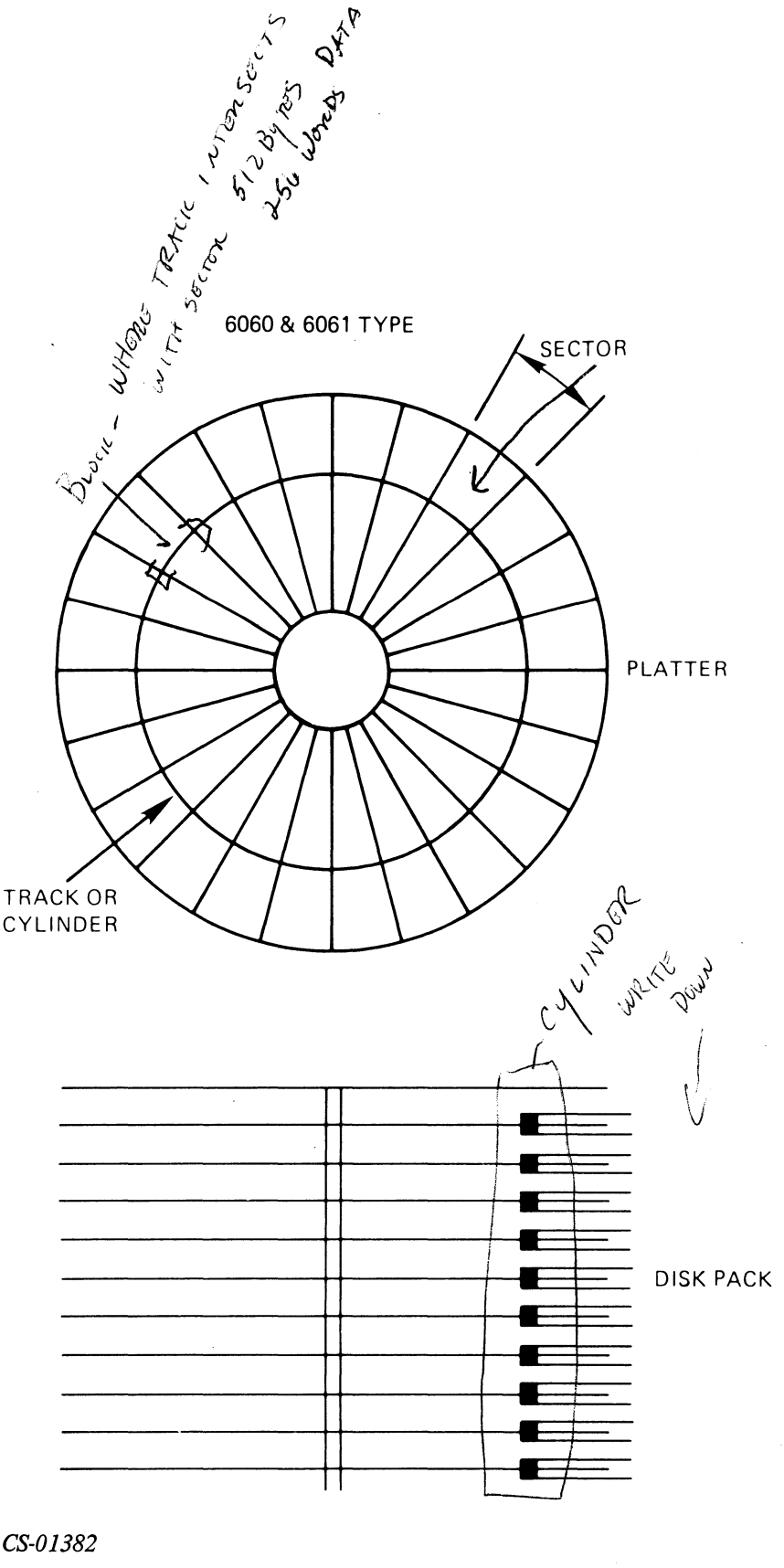
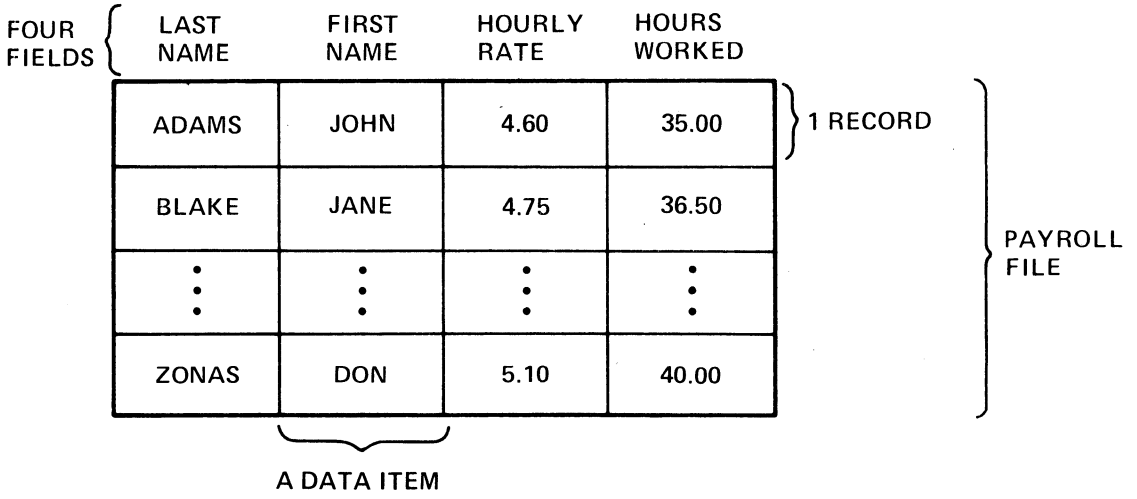


Figure 6.1 Physical Disc Structures

Generally, a file is an organized collection of records having a common feature or purpose.



CS-01383

Figure 6.2 Data Files

More examples of data files:

- Personnel Records (date started, Social Security number, address, etc.)
- Inventory Records (stock #, quantity, cost, etc.)
- Student Records (grades, attendance, parents' names, etc.)

PROGRAMS

Files also can contain instructions rather than data.

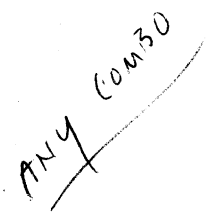
1. Source files - collection of instructions that have not been translated into machine language.
2. Executable files - programs that are ready to run. They must have the extension .PR.

FILE NAMING CONVENTIONS

Each filename can be from 1 to 31 characters long.

Legal characters:

1. A thru Z (upper or lower case)
2. 0 thru 9
3. ? (hook)
4. . (dot)
5. _ (underscore)
6. $



CREATING FILES

```
)CREATE,SCORES
```

This creates an empty file called SCORES to be filled in later by an application program.

To override normal CLI action, use switches.

```
)CREATE/I MYFILE
))THIS IS LINE 1
))THIS IS LINE 2
))THIS IS LINE 3
))THIS IS THE LAST LINE
)))
)
```

WHAT IS A DIRECTORY?

A directory is a catalog of bookkeeping information and pointers to files and subordinate directories within the directory tree.

TYPES OF DIRECTORIES

1. Control Point Directory (CPD) - a directory with a fixed maximum size.

2. Directory - a directory which can expand dynamically.

DIRECTORIES

AOS/VS simplifies file handling by grouping files together in "DIRECTORIES" (e.g., telephone directories are similar).

```
)DIRECTORY
INVENTORY
)DIRECTORY,GAMES
)FILESTATUS
STARTREK
ADVENTURE
)
```

Why do you need directories?

For grouping purposes

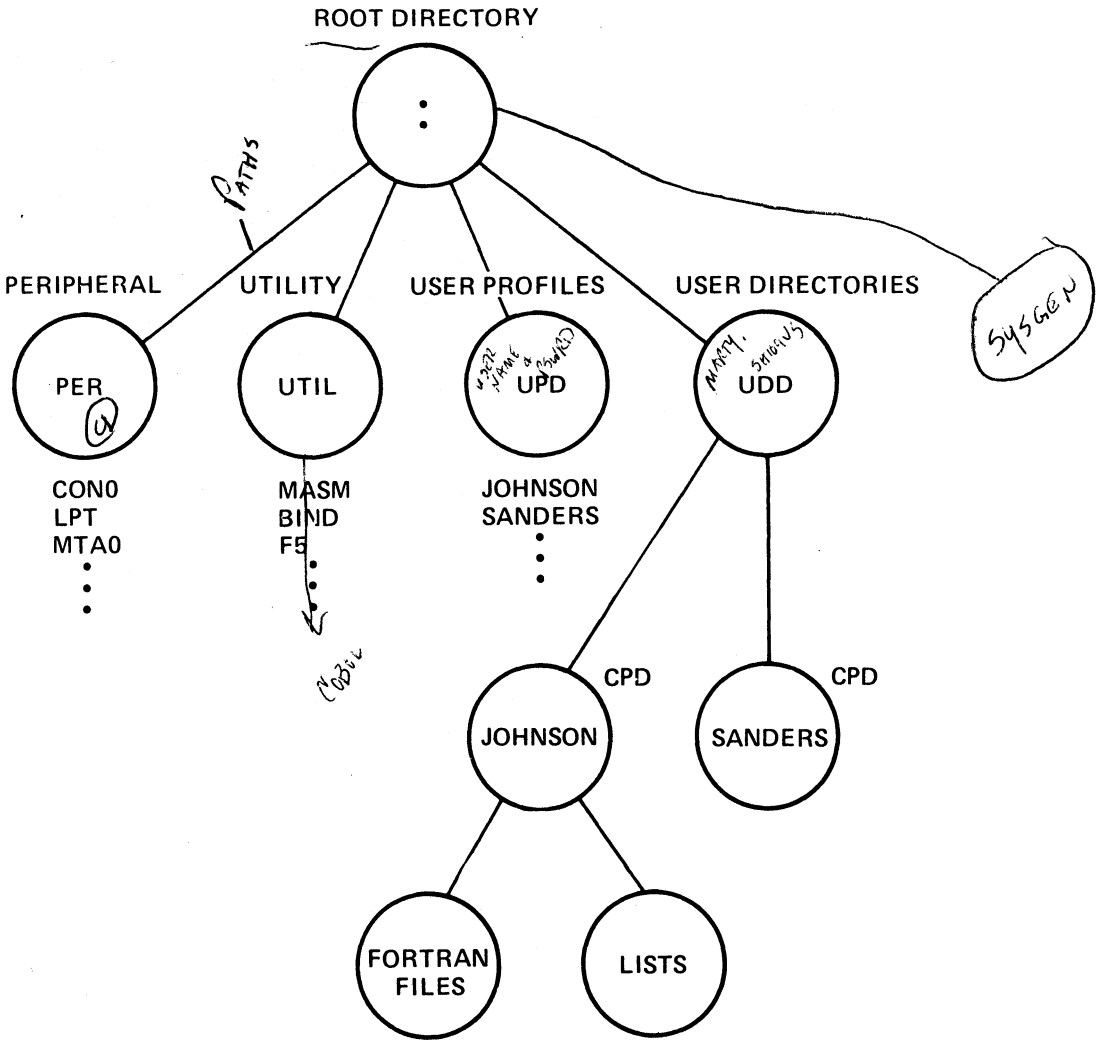
- All files written by one person
- All files written by one client
- All files written by one programmer

ADVANTAGES:

Reporting	The FILESTATUS command works with only the current working directory.
Backup	Directly move directory to tape.
Restore	Load directory from tape.
Clean up	Delete entire directory.
Control	Limit number of blocks for all files in a directory.

DIRECTORY TREES

- Each disc has a Master Directory called the ROOT.
- The Master Disc's root directory is symbolized by a colon (:).
- Each directory can contain information about any type of data files as well as about subordinate directories.
- Directories are organized into a network resembling an inverted tree.
- The directory used by AOS/VS as a reference point in the tree is known as the WORKING DIRECTORY.
- Disc space may be implicitly controlled by setting limits in a directory designated as a CONTROL POINT DIRECTORY.



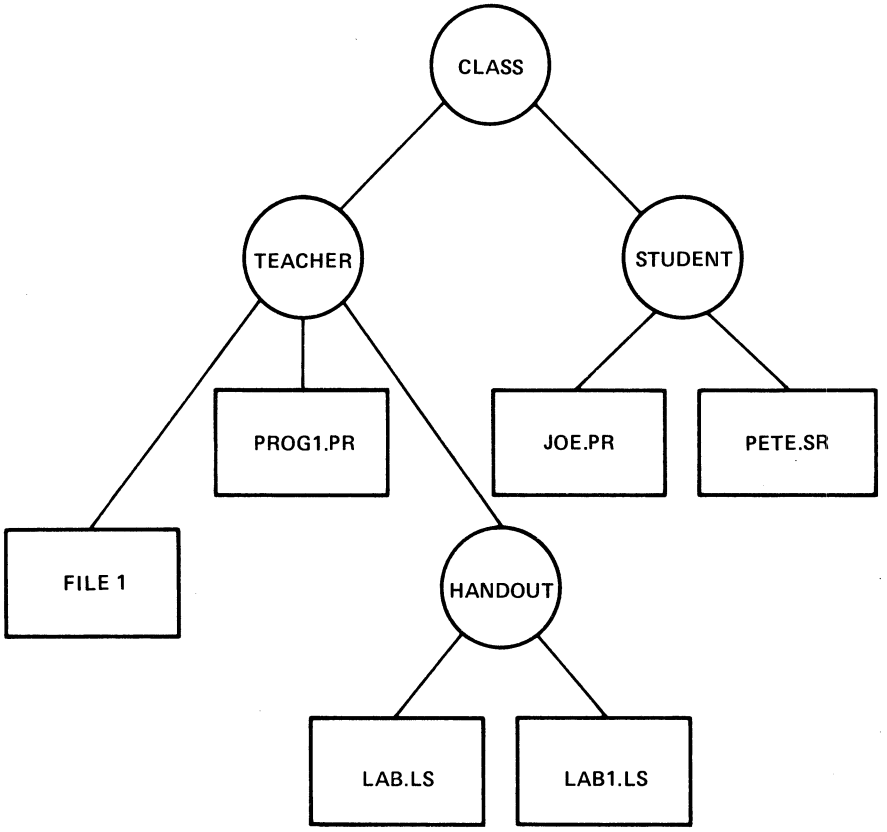
CS-01384

Figure 6.3 Directory Structure

```
)CREATE/DIR/MAX=2000,TEACHER
)CREATE/DIR/MAX=500,STUDENT
```

Directories can contain other files or other directories.

```
)DIR TEACHER
)CREATE/DIR HANDOUT

)SPACE/V
=MAX 2000, CURR 499, REM 501
```



CS-01385

Figure 6.4 Sample Directory

This picture is called an INVERTED TREE because it branches down.

It can branch down for up to eight levels.

WORKING DIRECTORY	Directory that the user is currently located within.
INITIAL WORKING DIRECTORY	Directory that user is in following logon.
PATHNAME	Describes the route through the directory tree structure that the system must take to find a file.
SEARCHLIST	List of directories that are automatically searched when a file cannot be found in the working directory. Order of search is: 1. Look in working directory. 2. Look in all directories in searchlist. 3. FILE NOT FOUND.
LINKS	A file that contains a pathname segment.

PATHNAMES

A PATHNAME describes the route that AOS/VS takes to find a file in the tree structure.

It has the following format:

```
[PREFIX]...FILENAME[:FILENAME...]
```

EXAMPLE:

```
:UDD:JOHNSON:LISTS
```

PATHNAME PREFIXES ARE:

: (COLON)	Start at the ROOT directory.
= (EQUAL SIGN)	Search the WORKING directory only.
^ (UP ARROW)	Move up to the PARENT directory.
@ (AT SIGN)	Search the PERIPHERAL directory only.

PATHNAME COMMAND

```
)PATHNAME TEST
:UDD:SUE:BETA:TEST
```

EXAMPLE:

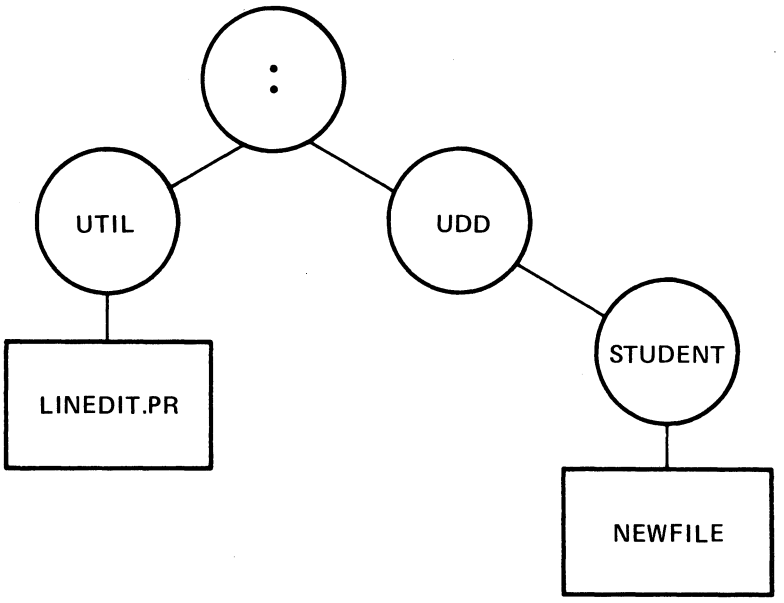
A student wants to run an editor program (LINEDIT.PR).

Where does it live?

In a directory called :UTIL.

LINEDIT will create a file in a student's directory.

```
)DIRECTORY,:UTIL
)X LINEDIT :UDD:STUDENT:NEWFILE
          or
)DIRECTORY :UDD:STUDENT
)X :UTIL:LINEDIT NEWFILE
```



CS-01386

Figure 6.5 Sample Directory Structure

The SEARCHLIST is a list of directories the system automatically searches when it can't find a file in your working directory. You can display or change your searchlist with the SEARCHLIST CLI command.

```
)SEARCHLIST
):,:PER
)SEARCHLIST :UTIL,:,:PER
)X LINEDIT,NEWFILE


)DELETE,LINEDIT.PR

FILE NOT FOUND
```

There is a limit of eight names on a searchlist.

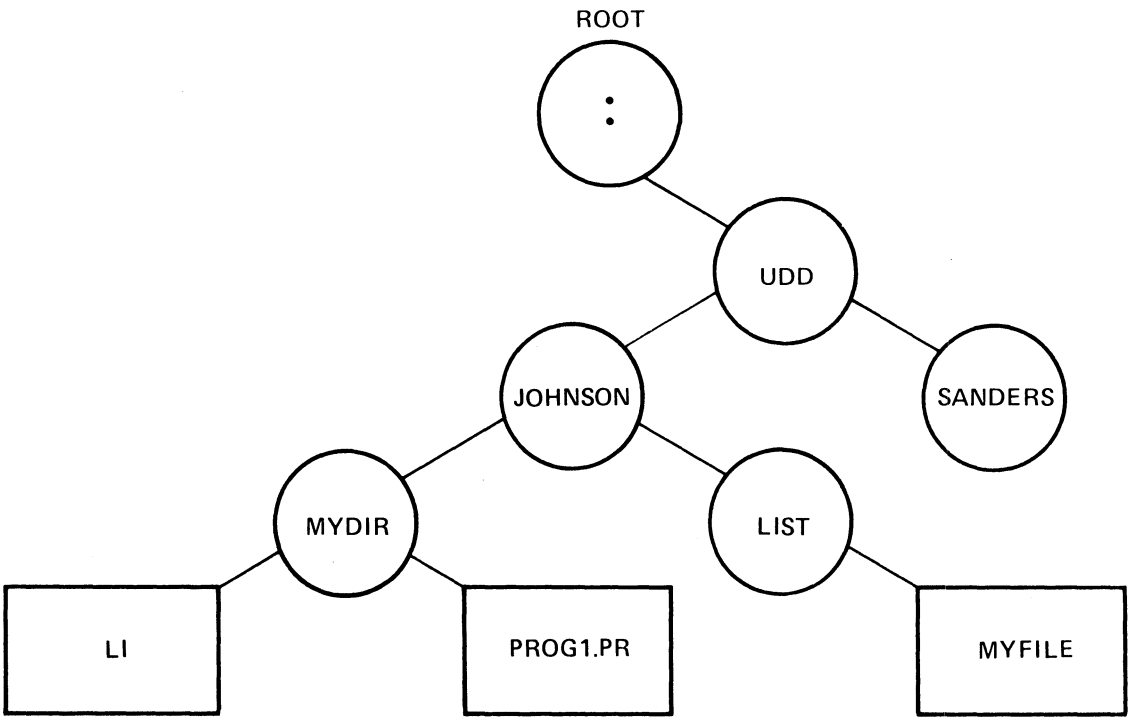
COMMANDS THAT USE THE SEARCHLIST

CHAIN	PATHNAME	QPUNCH
COPY	PROC	QSUBMIT
DATAFILE	QBATCH	TYPE
EXECUTE	QPLOT	X
LISTFILE	QPRINT	

LINKS

A LINK is a file that contains a pathname or pathname segment. It is essentially a shorthand technique used to simplify file referencing. The CLI command CREATE/LINK is used to create it.

```
)CREATE/LINK LI :UDD:JOHNSON:LIST

)TYPE LI:MYFILE
```



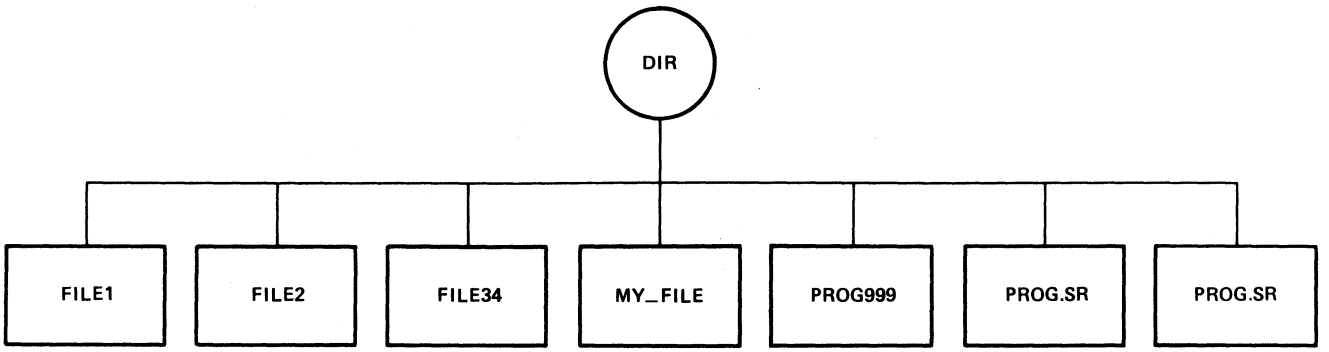
CS-01387

Figure 6.6 Simple Directory Structure with a Link

FILENAME TEMPLATES

The CLI will access any file that matches a template supplied in a command line argument list. Special purpose template characters are:

ASTERISK (*)	Matches any SINGLE character except a PERIOD.
HYPHEN (-)	Matches any character STRING that does not contain a PERIOD, including the NULL STRING.
PLUS SIGN (+)	Matches EVERY character STRING, including the NULL STRING.
NUMBER SIGN (#)	Matches the DIRECTORY filename template which appears immediately before it in the pathname, and all of its subordinate directories.
BACK SLASH (\)	All files matching the template after the back slash are excluded from the set of files meeting the overall template definition.



CS-01388

Figure 6.7 Sample Directory

DIR is the working directory.

Table 6.A Template Examples

TEMPLATE	FILENAMES MATCHED
FILE1	FILE1
FILE*	FILE1 AND FILE2
FILE-	FILE1,FILE2,AND FILE34
PROG.-	PROG.SR AND PROG.TEST
PROG+	PROG.SR, PROG999 AND PROG.TEST
+	All filenames

ACCESS CONTROL LIST

AOS/VS allows you to restrict access to any file by stipulating conditions for the file's use. The Access Control List (ACL) contains a list of users who can access the file, and the types of access allowed. The effect of each type is different for directory and non-directory and files.

The format of the ACL is a sequence of username-templates, each one of which is followed by the access-types permitted it.

The entry-pairs should be ordered such that the earlier username templates match the fewest number of users!

CLI FORMAT:

```
)ACL FILEB,BENTON OWARE,JENSEN WAR,+ RE
```

Table 6.B Access Types

ACCESS	ABBREVIATION	NON-DIRECTORY FILE	DIRECTORY
EXECUTE	E	User can execute file.	User can use directory's name in a pathname.
READ	R	User can read (examine) data in the file.	User can examine list of files in the directory.
APPEND	A	n/a	User can insert names of new files into the directory.
WRITE	W	User can modify contents of files.	User can insert and delete files from the directory; also change the ACLs.
OWNER	O	User can change file's ACL and delete file.	User can change directory's ACL, or delete directory.

DEFAULT ACL

- Associated with any file/directory created

```
)DEFACL                              (To display current setting)
)DEFACL [username,access].....       (To change current setting)
```

SUPERUSER

- A privilege
- Allows user to bypass ACL

```
)SUPERUSER [{ON }]
           [{OFF}]
```

PERMANENCE

- To protect key files from accidental deletion

```
)PERMANENCE pathname [{ON }]
                     [{OFF}]
```

FILE MAINTENANCE COMMANDS

DELETE pathname [pathname].....	Delete a file.
FILESTATUS [pathname]....	Display the status of one or more files.
RENAME pathname newname	Rename a file.
TYPE pathname [pathname]...	Type the contents of a file.
COPY dest-file sourcefile [sourcefile]....	Copy a file to a destination file.
MOVE dest-dir sourcefile [sourcefile]....	Move copies of one or more files.

DEFINITION:

An executable user-defined file containing a sequence of CLI commands. The file normally has a .CLI extension.

BENEFITS:

- Conciseness
- Saves time and effort by combining repetitive operations
- Looks like regular CLI command
- Execution-time flexibility through dummy arguments and switches
- Recursion

EXAMPLE:

To create the CLI macro called STATS.CLI:

```
)CREATE/I STATS.CLI
))DATE
))TIME
))WHO
))PERFORMANCE
))SPACE;TREE
))RUNTIME
)))
)
```

The System Manager can define and write special-purpose MACROS, complete with switches and arguments, which can be used later by operators and which will appear to them exactly like existing CLI commands.

MACRO DUMMY ARGUMENTS

- CLI allows the user to write generalized macros for which arguments can be substituted at execution time.
- Dummy arguments are enclosed between percent signs (%n%).

EXAMPLE:

To create a macro called TRANSFER.CLI which will accept arguments at execution time:

```
) CREATE/I TRANSFER.CLI
))WRITE TRANSFER MACRO EXECUTING
))MOVE/NACL,%1%,%2%
))DIR,%1%
))RENAME,%2%,%3%
)))
)
```

To execute the TRANSFER.CLI macro:

```
) TRANSFER,:UDD:GREG,FILE1,MYFILE
```

```
:UDD:GREG      is substituted for %1%
FILE1          is substituted for %2%
MYFILE         is substituted for %3%
```

- Notice that the TRANSFER.CLI macro is more useful since it will accept the arguments at execution time and therefore is not limited to transferring only a file called FILE1.

MODULE 7

AOS/VS PROCESS CONCEPTS

MODULE 7
AOS/VS PROCESS CONCEPTS

OBJECTIVES

Upon successful completion of this module, the student will be able to:

1. Define the types of AOS/VS processes.
2. List the possible states a process may be in.
3. Define what effect priority has on process scheduling.
4. State the hierarchy of processes in a typical AOS/VS system.
5. List and define the components of a process.

PROGRAMS AND PROCESSES

A PROGRAM is a series of instructions, produced by linking together a set of compiled or assembled source files.

A PROGRAM is a static entity which contains potentially executable code paths.

A PROCESS is a dynamic entity, existing only when it is executing program instructions.

A PROCESS, therefore, is seen by AOS/VS as the primary user of system resources.

For this reason, a PROCESS competes with other processes for resources such as memory, I/O devices, disc file space, and CPU time.

PROCESS COMPONENTS

1. Program

2. PID - Unique

3. Username

4. Process Name - CON 12 (EX)

5. Priority

6. State

7. Type

8. Working Directory

9. Searchlist

10. Generic Files - (LISTFILE ON) @LIST @DATA @INPUT (KEYBOARD) @OUTPUT - (SCREEN) @CONSOLE - (WHOLE SYS) AT LOG-ON

11. Privileges

12. Memory

13. Sons - 256 MAX - CAN LIMIT AMOUNT OF SONS FOR EACH USER

14. Working Set - AMOUNT OF PHYSICAL MEMORY CURRENTLY USING

PROCESS ID

AOS/VS assigns a unique number to each process created. The number is in the range of 1 through 255, and is the smallest number not currently being used.

When a process terminates, the PID associated with the process may be re-assigned by AOS/VS to a new process.

PROCESS ID 1 = The Peripheral Manager (PMGR)

PROCESS ID 2 = The Operator Process (OP)

PROCESS ID 3 is usually assigned to a process called EXEC.

PROCESS NAMES

Each process has a unique FULL PROCESS NAME.

EXAMPLE:

USERNAME:SIMPLE_PROCESS_NAME

where the SIMPLE_PROCESS_NAME may be optionally specified by the parent process during creation.

The default SIMPLE_PROCESS_NAME is a three-character PID.

EXAMPLE:

STUDENT1:006

At logon, EXEC assigns a SIMPLE_PROCESS_NAME that is equivalent to the device name assigned to the user.

EXAMPLE:

STUDENT2:CON3

Either the PID or the FULL PROCESS NAME can be used to access a process.

PROCESS TYPES

RESIDENT	Always remains in memory.
PREEMPTIBLE	Usually resides in memory, but is swapped to disc if: 1. It becomes blocked and any other process needs memory. 2. A resident or higher priority preemptible process needs memory.
SWAPPABLE	Receives main memory only if any remains available after all higher type processes have received what they need.

PROCESS STATES

ELIGIBLE

- Ready to run
- In main memory

INELIGIBLE

- Ready to run, BUT
- Main memory not allocated
    1. Swapped to disc
    2. Initial creation

BLOCKED

- Waiting for some external event
- In main memory, or swapped to disc

PROCESS PRIORITY CONCEPTS

Shows relative importance

- User assigned number
- Low number = high priority (e.g., 1 = highest)

Used to award memory to preemptible and swappable processes

Used for scheduling of all types

Resources passed around on an equal basis if many processes at same priority

PRIORITY DECIDES WHO WILL RUN FIRST

PROCESS PRIORITY

RESIDENT PREEMPTIBLE — Numerical priorities ranging from 1 (highest) to 255 (lowest) are given. Both process types are treated as a single group and are allocated CPU time according to relative priority numbers.

RUN LAST

SWAPPABLE — User-defined relative priorities are assigned from 1 (highest) to 3 (lowest). The derived priority is a function of relative priority and BEHAVIOR RATING.

1-2-OR-3

PRINTERS (IF SLOW)

PRE-EMPTIBLE 255

PROCESS CHARACTERISTICS

INTERACTIVE vs. COMPUTE-BOUND

INTERACTIVE PROCESSES

- Conversational
- Frequent I/O
- Fast terminal response necessary
- Use small amount of CPU time

COMPUTE-BOUND PROCESSES

- Internal processing only
- No console I/O
- Turn-around time not critical
- Need large amounts of CPU time

BEHAVIOR RATING

AOS/VS dynamically allocates a priority to swappable processes in an effort to minimize response time for interactive processes.

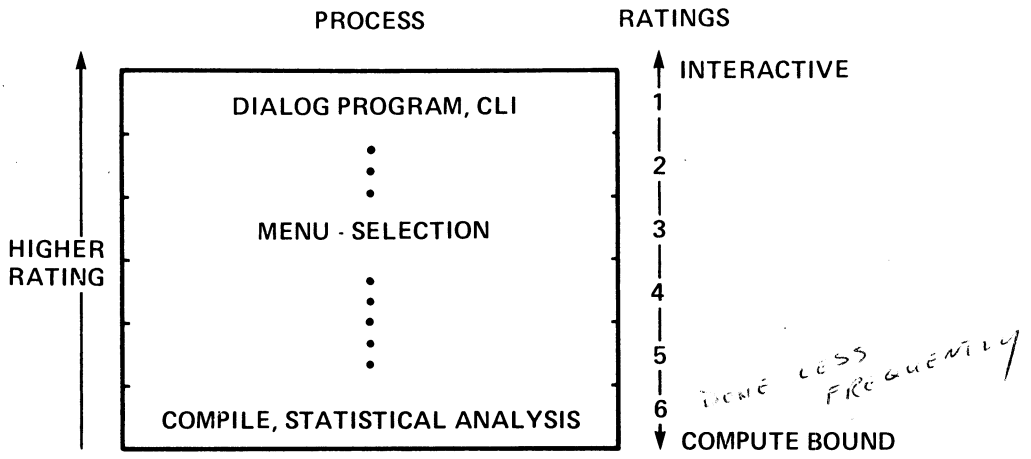
To support this, AOS/VS assigns a BEHAVIOR RATING (B.R.) to each swappable process and then updates the B.R each time the process gets control of the CPU.

The BEHAVIOR RATING is a number in the range of 1 through 6, where 1 is assigned to a highly interactive process and 6 is assigned to a compute-bound process.

Each time AOS/VS decides to run a swappable process it assigns a Time Slice to the process.

If the process becomes blocked before using the entire Time Slice, AOS/VS adjusts the B.R. to reflect a more INTERACTIVE process, e.g., NEW B.R. = OLD B.R. -1.

If the process uses the entire Time Slice, AOS/VS adjusts the B.R. to reflect a more COMPUTE BOUND process, e.g., NEW B.R. = OLD B.R. +1.



CS-01389

Figure 7.1 Behavior Ratings - Swappable

PROCESS SUMMARY

The operating environment determines your selection of process types and priorities.

PROCESS TYPES

A. Resident/Preemptible

   1. Real-Time

      a. Obtain resources when input demands

      b. Lock up a minimum of memory

      c. Usually deals with non-disc input

B. Swappable - Behavior rating supports changing process nature: CPU time is allocated according to behavior rating and priority.

   1. Time-Share

      a. Highly interactive

      b. Short CPU slices

      c. Good response to terminals

   2. Batch

      a. Compute-bound

      b. Lots of CPU time

      c. Minimal scheduling overhead

PROCESS HIERARCHY

The relationship of processes permits an additional dimension of control over process execution.

At startup, AOS/VS creates two subordinate processes:

PMGR	The Peripheral Manager, which handles all byte I/O devices for AOS/VS.
OP	The Operator Process, which executes the initial CLI and permits the operator to create other subordinate processes of any type, at any priority.

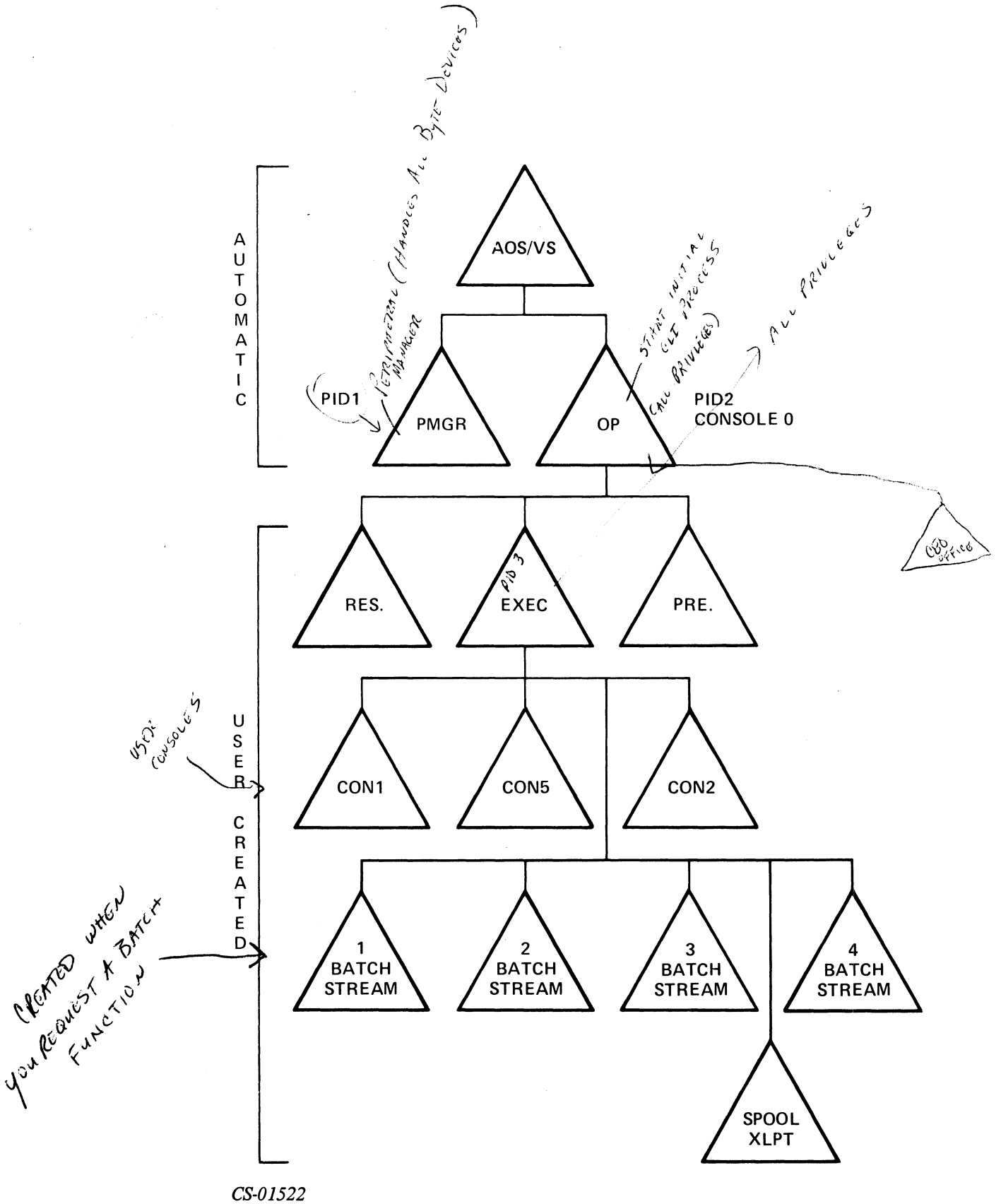
The creator (parent) of a process can assign any privileges it possesses to its offspring.

When one process creates another, the new one is subordinate to the creator.

The parent (creating) process can terminate or block any of its descendents.

When a process terminates, any process subordinate to it is also terminated.

Subordinate processes cannot terminate or block superior processes.



CS-01522

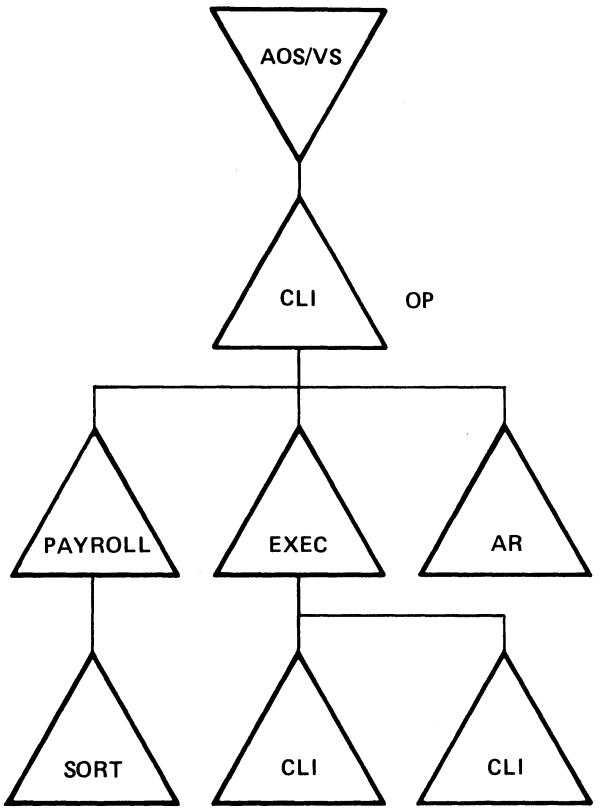
Figure 7.2 AOS/VS Process Tree (Hierarchy)

SYSTEM PROCESS CONCERNS

- AOS/VS supports 255 processes.
- PMGR, OP, EXEC use up one process each.
- Each general-purpose batch stream uses two processes.
- Each device being spooled uses one process.
- Each general time-sharing terminal uses two processes.
- Each additional process stretches resources thinner while increasing AOS/VS bookkeeping demands.

1. EXECUTE / or / XEQ } program-name [arguments]

2. PROCESS program-name [arguments]



CS-01523

Figure 7.3 Process Tree

- XEQ = PROC/BLOCK/IOC/DEFAULT
- Programmers normally setup the process command in a CLI macro for the operator.

SUPERPROCESS

- A privilege
- Allows the user to bypass process hierarchy

```
) SUPERPROCESS [{ON }]
               [{OFF}]
```

PROCESS CONTROL COMMANDS

BLOCK {username:procname} {process_id}	Block a processs by processname or process_id.
PRIORITY {username:procname} {process_id}	Set or display a process's priority.
TYPE {username:procname} [{PREEMPTIBLE}] {process_id} [{RESIDENT }] [{SWAPPABLE }]	Set or display a process's type.
TERMINATE {username:procname} {process_id}	Terminate a process.
UNBLOCK {username:procname} {process_id}	Unblock a process.

MORE PROCESS COMMANDS

BYE	Terminate CLI at your console.
CHECKTERMS	Check for the termination of a son process.
PAUSE {seconds.tenths}	Delay the CLI for n seconds.
RUNTIME {username:procname} {process-id}	Check a process's runtime information.
TREE {username:procname} [argument] {process-id}	Display a process's family tree.
WHO {username:procname} [argument] {process-id}	Display process information.

MODULE 8

THE EXEC PROCESS

MODULE 8
THE EXEC PROCESS

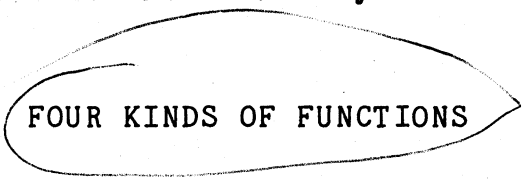
OBJECTIVES

Upon successful completion of this module, the student will be able to use the CONTROL @EXEC commands to control:

1. Logging on to the system.
2. Spooling.
3. Batch streams.
4. Mount and dismount of tapes.

THE EXEC PROCESS

- Handles AOS/VS user interaction at all consoles.
- Creates swappable process for time-shared terminals, batch streams and spooling (virtual devices).
- Generally created at start-up time by OP, as part of the UP macro.
- Controls user access to the AOS/VS system.



1. LOG-ON FUNCTION - Monitors consoles to allow users to gain access to the system.
2. BATCH FUNCTION - Can supervise and run user programs in batch streams.
3. SPOOLING FUNCTION - Ensures that users sending output to certain character-oriented devices can use them in an orderly way.
4. MOUNT/DISMOUNT FUNCTION - Allows users in batch or interactive mode to request the operator to mount and dismount tapes.

EXEC

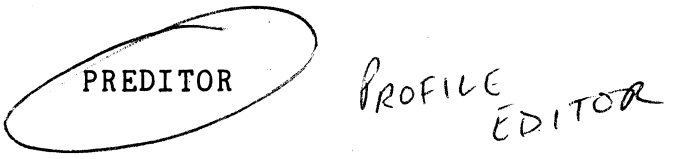
- Creates only swappable processes
- Provides user-level accounting information

LOG-ON FUNCTION

When commanded to monitor a console, EXEC waits for the NEW LINE key to be pressed on that console. After NEW LINE, EXEC prompts for Username and Password and compares it to the User Profile.

USER PROFILES

- A list of privileges and attributes of system users
- Access control vehicle - username/password pairs
- Resource allocation aid
    1. Privileges
    2. Limits
- Each user profile is a one-block file in the profile directory (:UPD)

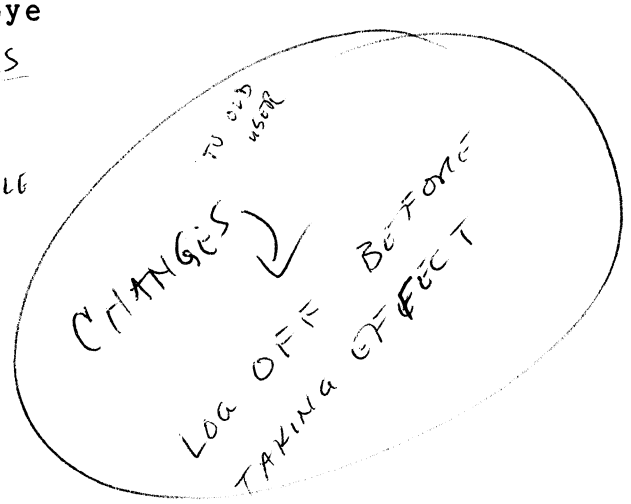


- Creates, edits, lists and deletes user profiles

) X PREDITOR

SAMPLE PREDITOR DIALOGUE

```
) X PREDITOR
AOS/VS USER PROFILE EDITOR REV XX.XX 23-DEC-80 9:33:21
COMMAND:create
USERNAME:white
PASSWORD CHANGE?  (Y OR NL) y
NEW (3-15 CHARS):jack
INITIAL IPC FILE [] CHANGE?  (Y OR NL) y
NEW (1-63 CHARS):  :udd:white:start_up.cli
PROGRAM [:CLI.PR] CHANGE?  (Y OR NL)
CREATE WITHOUT BLOCK [NO]?  (Y,N OR NL)
USE IPC [NO]:(Y,N OR NL) y
USE CONSOLE [YES]?  (Y,N OR NL)
USE BATCH [YES]?  (Y,N OR NL)
USE VIRTUAL CONSOLE [YES]? (Y,N, OR NL)
ACCESS LOCAL RESOURCES FROM REMOTE MACHINES [YES]?  (Y,N OR NL)
UNLIMITED SONS [NO]?  (Y,N OR NL) y
CHANGE PRIORITY [NO]:(Y,N OR NL) y
CHANGE TYPE [NO]?  (Y,N,OR NL) y
CHANGE USERNAME [NO]?  (Y,N OR NL)
ACCESS DEVICES [NO]?  (Y,N OR NL)
SUPERUSER [NO]?  (Y,N OR NL)
SUPERPROCESS [NO]?  (Y,N OR NL)
MODEM [NO]?  (Y,N OR NL) y
CHANGE ADDRESS SPACE TYPE [NO]?  (Y,N OR NL) y
CHANGE WORKING SET LIMIT [NO]?  (Y,N OR NL) y
PRIORITY [2] CHANGE?  (Y OR NL) y
LET SYSTEM ASSIGN?  (Y,N OR NL) n
NEW (1-3): 1
MAX QPRIORITY [0] CHANGE?  (Y OR NL) y
NEW (0-255): 127
DISK QUOTA [500] CHANGE (Y OR NL) y
NEW (0-2147483647): 100000
LOGICAL ADDRESS SPACE -BATCH [-1 SYSTEM DEFAULT] CHANGE:(Y OR NL)
LOGICAL ADDRESS SPACE -NON-BATCH [-1 SYSTEM DEFAULT] CHANGE:(Y OR NL)
MINIMUM WORKING SET SIZE -BATCH [-1 SYSTEM DEFAULT] CHANGE:(Y OR NL)
MAXIMUM WORKING SET SIZE -BATCH [-1 SYSTEM DEFAULT] CHANGE:(Y OR NL)
MINIMUM WORKING SET SIZE -NON-BATCH [-1 SYSTEM DEFAULT] CHANGE:(Y OR NL)
MAXIMUM WORKING SET SIZE -NON-BATCH [-1 SYSTEM DEFAULT] CHANGE:(Y OR NL)
COMMENT
COMMAND:bye
```



USER NAME MUST BE OP

EXEC LOG-ON COMMANDS

CONTROL @EXEC

DISABLE @CONn - DISABLE/ALL WON'T GO INTO EFFECT UNTIL THEY LOG OFF

ENABLE @CONn

TERMINATE @CONn - LOGGED OF AUTOMATICALLY

CONSOLESTATUS [@CONn]

PROMPTS {ON }
{OFF}

LOGGING/START[/MAX=n] pathname
LOGGING/STOP
LOGGING

MESSAGE message

LIST OF ALL EXEC COMMANDS

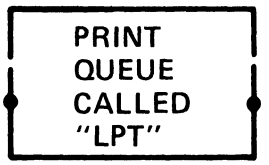
)XHELP

- Slow output devices need not slow down processing.
- A spool queue is itself a "virtual" output device. "Actual" output is deferred.
- Multiple processes may appear to use the same devices simultaneously.
- When you open a spool queue, any process accessing the queue has its output temporarily diverted to and queued by a disc file. This permits output speeds to be at fast disc rates.
- CLI commands QPRINT, QFTA and QPLOT assume queue names LPT, FTA and PLT.
- Operator can control:
    1. Characters per line
    2. Lines per page
    3. Pages per request
    4. Header and trailer sheets

SPOOLING EXAMPLE

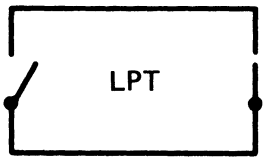
)CONTROL @EXEC CREATE PRINT LPT

Creates:



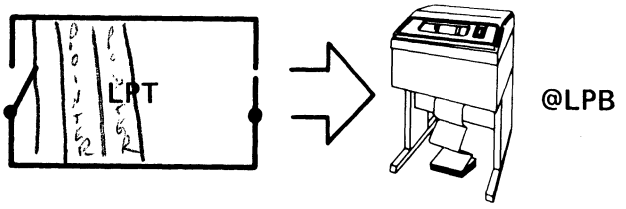
)CONTROL @EXEC OPEN LPT

Opens an "input door" to the queue, thereby allowing output data from processes to be collected.



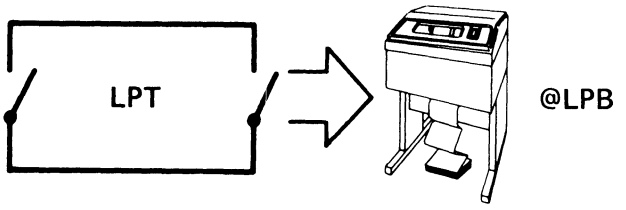
)CONTROL @EXEC START LPT @LPB UPPER

Creates a logical "output door" connecting the print queue LPT to a printer called @LPB.



)CONTROL @EXEC CONTINUE @LPB

Opens the "output door" and permits printing to begin on @LPB of the spooled contents of print queue LPT.



CS-01392

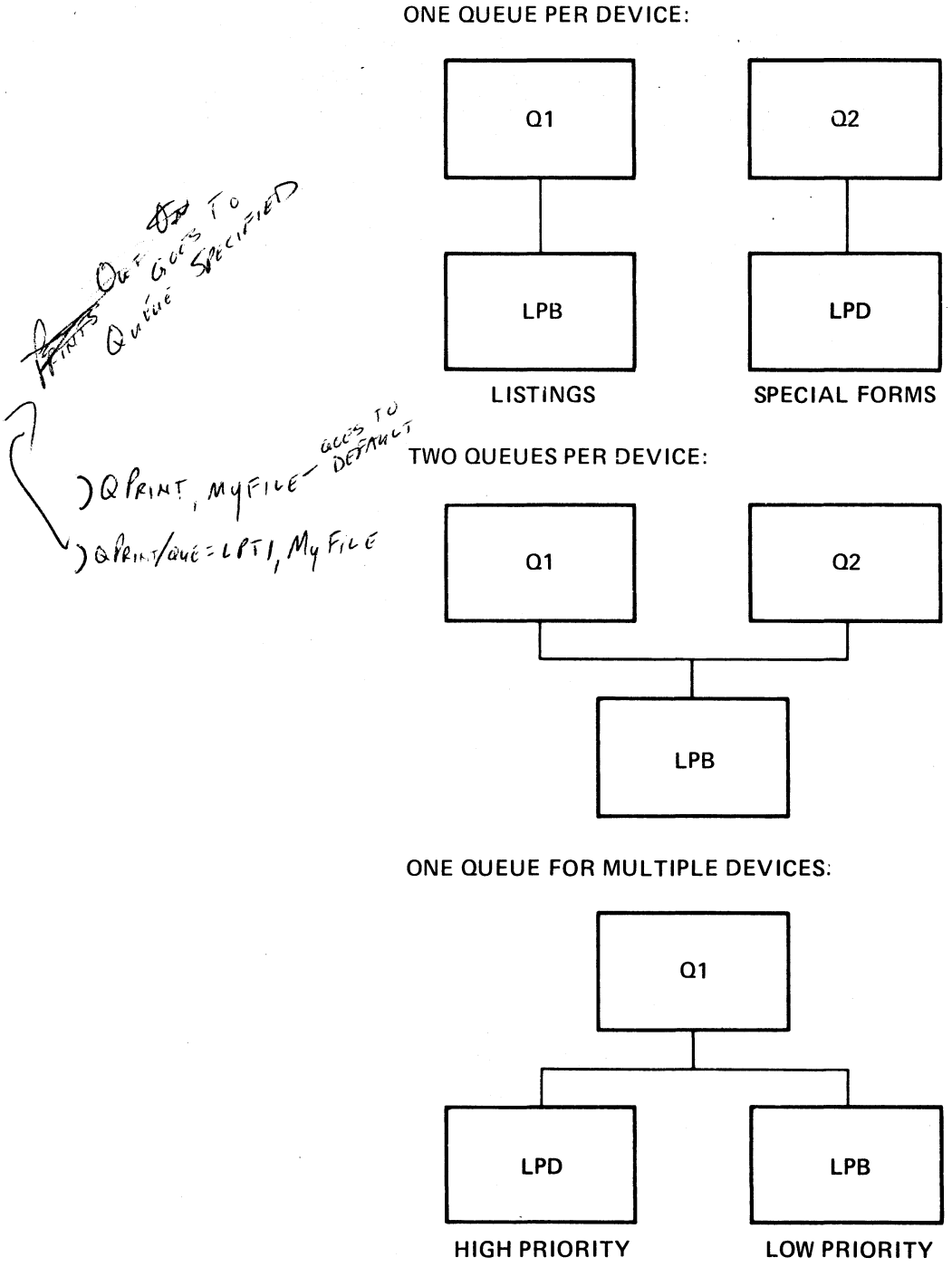


Figure 8.1 Spooling Variations

EXEC COMMANDS SPOOLING VARIATIONS

One queue per device:

```
CONTROL @EXEC START Q1 @LPB
CONTROL @EXEC START Q2 @LPD
```

Two queues per device:

```
CONTROL @EXEC START Q1 @LPB
CONTROL @EXEC START Q2 @LPB
```

One queue for multiple devices:

```
CONTROL @EXEC START Q1 @LPD
CONTROL @EXEC START Q1 @LPB
CONTROL @EXEC QPRIORITY @LPD 0 126
CONTROL @EXEC QPRIORITY @LPB 127 255
```

How do you get a filename into the spool queue?

1. LISTFILE
2. QPRINT

LISTFILE

)LISTFILE [pathname]

This sets an optional output file for the /L switch on any CLI command.

It may be overwritten by an assigned switch: /L=PATHNAME

If the pathname is a queuename, output is diverted to a scratch disc file until closed, and then it is queued to the device.

The following CLI commands would cause three files to be printed to the line printer.

```
     )LIST @LPT
1.   )F/AS/L +.PR
2.   )TYPE/L PROG.SR
3.   )F/AS/L=@LPT
4.   )X MASM/L TEST.SR
5.   )QPRINT TEST.SR
6.   )LIST/K
```

The first file printed would contain the output from Command 3.

The output from Command 5 would be next.

Lastly, the output from Commands 1,2 and 4 would be printed.

SWITCHES

QPRINT pathname.....

/BINARY

/AFTER=date:time

/COPIES=n

/BEGIN=n

/END=n

/DELETE

/FOLDLONGLINES

/FORMS=type

/NORESTART

/NOTIFY

/PAGES=n

/QPRIORITY=n

/DESTINATION=s

/HOLD

/QUEUE=queuename

/TITLES

/OPERATOR

/S

/V

LABELS OR SPECIAL FORMS

QPRINT / FORMS = NAME

If you have special forms:

```
)QPRINT/FORMS=3_PART
```

When you do a QDISPLAY you will see 3_PART requests backed up in the queue.

"PAUSE" that device

```
)CONTROL @EXEC PAUSE @LPB
```

then

```
)CONTROL @EXEC FORMS @LPB 3_PART
)CONTROL @EXEC CONTINUE @LPB
```

NOTE: There must be a file on the disc called 3_PART in the directory -- :UTIL:FORMS.

Now only 3_PART requests to that device will be processed until

```
)CONTROL @EXEC PAUSE @LPB
)CONTROL @EXEC FORMS @LPB
)CONTROL @EXEC CONTINUE @LPB
```

is typed in.

)QDISPLAY

/QUEUE=name
/SUMMARY
/TYPE=type
/V

QDISPLAY EXAMPLE

```
*****   )QDISPLAY/QUE=LPT

        LPT                PRINT    OPEN
                96 FJM     USER1    :UDD:USER1:TEST1
                90         USER2    :UTIL:TEXT2
                95         USER1    :UTIL:TEXT2
                94 A       USER3    :UDD:USER3:TEXT3
                97         USER4    :UDD:USER4:TEST2

        FLAGS EXPLANATION:
        A = SEQUENCE NUMBER HELD BY USER
        F = /DELETE
        J = CANCELED BY USER
        M = /NOTIFY

*****   )CONTROL @EXEC HOLD 94
*****   )CONTROL @EXEC CANCEL 95

*****   )QDISPLAY/V/QUEUE=LPT

       LPT                  PRINT   OPEN
       SEQ# PRI  TIME       LIMIT FLGS USERNAME FORMS   PATHNAME
                 6-JAN-81
       96    25  8:42:24        4  FJM  USER1    3_PART :UDD:USER1:TEST1
       90    50  8:34:14       11       USER2           :UTIL:TEXT2
       95    50  8:41:10       11   I   USER1    3_PART :UTIL:TEXT2
       94   127  8:40:46        6  AC   USER3           :UDD:USER3:TEXT3
       97   127  8:42:50        4       USER4           :UDD:USER4:TEST2

      FLAGS EXPLANATION:
      A = SEQUENCE NUMBER HELD BY USER
      C = SEQUENCE NUMBER HELD BY OPERATOR
      F = /DELETE
      I = CANCELED BY OPERATOR
      J = CANCELED BY USER
      M = /NOTIFY

*****   )CONTROL @EXEC SPOOLSTATUS LPT

        FROM PID  3 : (EXEC) LPT BEING PROCESSED BY :  @LPB

*****   )CONTROL @EXEC SPOOLSTATUS @LPB

  FROM PID 3 :  (EXEC) @LPB PROCESSING:  BATCH_OUTPUT BATCH_LIST LPT

     FROM PID  3 : CPL=80, LPP=66, HEADERS=1, TRAILERS=0
```

OTHER QUEUE (USER) COMMANDS

```
QCANCEL {seq-no } [{seq-no }].....
        {jobname} [{jobname}]

QHOLD {seq-no } [{seq-no }]....
      {jobname} [{jobname}]

QUNHOLD {seq-no } [{seq-no }]...
        {jobname} [{jobname}]
```

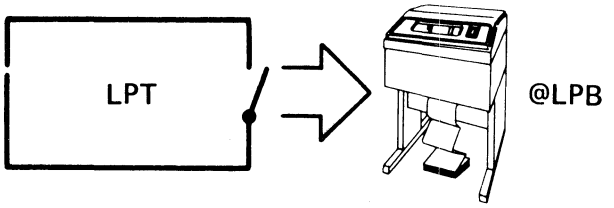
EXEC FORMS CONTROL COMMANDS

```
CONTROL @EXEC

     CPL @printername number

     ELONGATE @devicename

     HEADERS @devicename {0}
                         {1}
                         {2}

     LIMIT @devicename [pages]

     LPP @devicename number

     TRAILERS @devicename {0}
                          {1}
                          {2}

     UNLIMIT @devicename

     DEFAULTFORMS @devicename [form-name]
```

DELETING A QUEUE

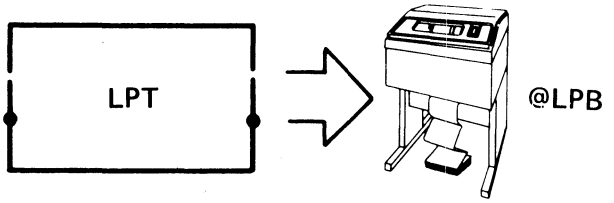
)CONTROL @EXEC CLOSE LPT

Closes the "input" door.
Stops input to the queue.



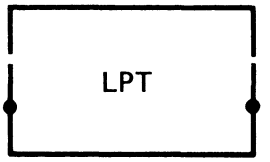
)CONTROL @EXEC PAUSE @LPB

Closes the "output" door
to the printer.



)CONTROL @EXEC STOP LPT

Disconnects the queue
from the device.

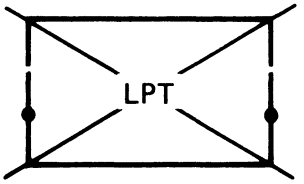


)CONTROL @EXEC PURGE LPT

Removes all information
that is in the queue.

)CONTROL @EXEC DELETE LPT

Deletes the queue.



CS-01394

```
CONTROL @EXEC

     SPOOLSTATUS {[queuename]}
                 {@devicename}


     STATUS @devicename


     CANCEL sequence number


     HOLD {sequence number}
          {username}


     UNHOLD {sequence number}
            {username}


     BRIEF @devicename


     VERBOSE @devicename


     SILENCE @devicename
```

```
UNSILENCE @devicename

FLUSH @devicename

RESTART @devicename [start-page] [end-page]

EVEN @devicename {ON}
                 {OFF}

BINARY @devicename {filename}
                   {OFF}

ALIGN @devicename
ALIGN/CONTINUE @devicename [[-]n]

PRIORITY [ /SWAPPABLE    ] @devicename priority
         [ /PREEMPTIBLE  ]
         [ /RESIDENT     ]

XBIAS @devicename bias_factor
```

THE EXEC BATCH FUNCTION

The BATCH function of EXEC is the non-interactive processing of program requests, submitted by users through:

1. CLI - permitting other on-line processing while the batch job is serviced.

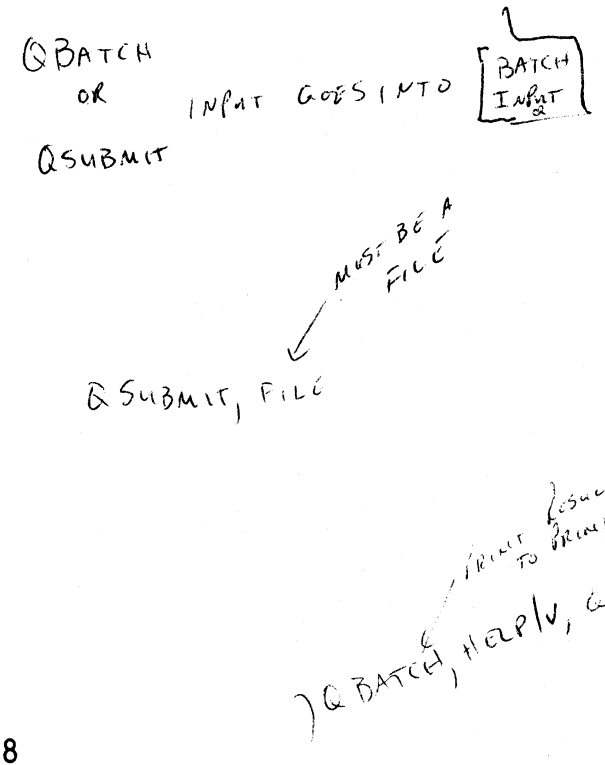
   - QBATCH - one time submission.
   - QSUBMIT - user file containing several commands used for repeated submissions.

2. Card Readers - in STACKED format, including a username/password pair.

EXEC can process up to four batch jobs simultaneously, each one executing in a separate BATCH STREAM.

EXEC also creates three permanent queues to support batch processing:

1. BATCH_INPUT
2. BATCH_OUTPUT
3. BATCH_LIST



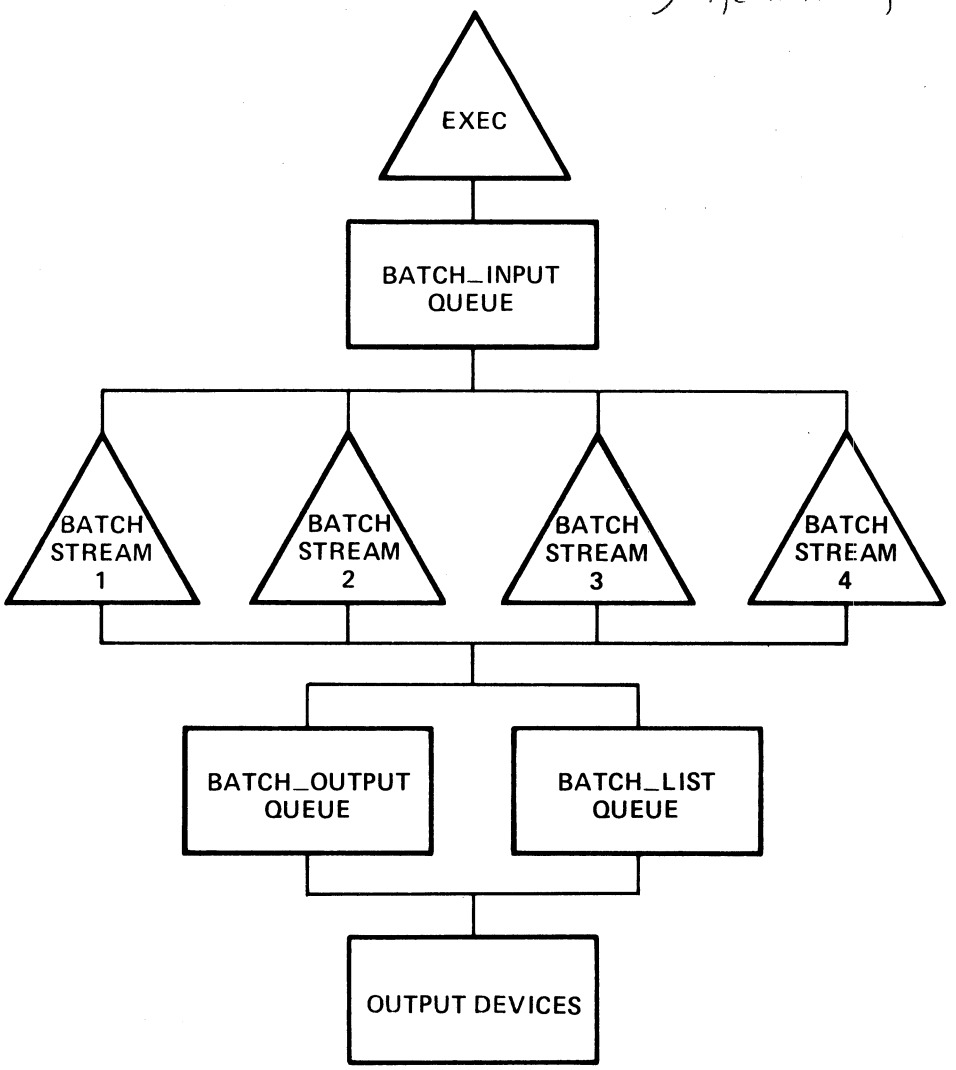
EXEC needs a user profile to create a swappable process and to run the proper initial program.

QBATCH and QSUBMIT interactive requests use the same profile as the requesting program's process.

STATUS OF STREAMS

)CX, STATUS
PAUSED OR IDLE - NOT USABLE

)CX, CONTINUE, 1 → CONTINUE STREAM 1



CS-01395

Figure 8.2 Batch Flow

To create the batch queues you need to open BATCH_INPUT, BATCH_OUTPUT and BATCH_LIST. This needs to be done only once.

```
)CONTROL @EXEC OPEN BATCH_INPUT
)CONTROL @EXEC OPEN BATCH_OUTPUT
)CONTROL @EXEC OPEN BATCH_LIST
```

Once you open the queues, you then process BATCH_INPUT with:

```
)CONTROL @EXEC CONTINUE 1
```

You are allowed to have four batch processes running concurrently. Just specify other stream numbers.

Now you may need to direct your output to a device, such as the line printer.

```
)CONTROL @EXEC START BATCH_OUTPUT @LPB
)CONTROL @EXEC START BATCH_LIST @LPB
```

And finally you need to start a process to the output device.

```
)CONTROL @EXEC CONTINUE @LPB
```

CONTROL @EXEC

CONTINUE [stream number]

STATUS [stream number]

OPERATOR {ON}
{OFF}

UNSILENCE [stream number]

PAUSE [stream number]

BRIEF [stream number]

QPRIORITY [stream number] [high-value low-value]

STACK {pathname}
{@devicename}

FLUSH stream number

VERBOSE [stream number]

SILENCE [stream number]

CONTROL @EXEC

CANCEL sequence number

HOLD {sequence number}
     {username}

UNHOLD {sequence number}
       {username}

LIMIT [stream number [hh:mm:ss]]

UNLIMIT [stream number]

PRIORITY [/SWAPPABLE
         /PREEMPTIBLE
         /RESIDENT] stream number priority

XBIAS stream number bias_factor

EXAMPLE 1:

```
)CREATE/I BATCHJOB
))F/AS/S/L
))X,FORT,MYFILE
))X,BIND,MYFILE
))F/AS,MYFILE
))DIR
))SEA
)))
)
)QSUBMIT BATCHJOB
```

EXAMPLE 2:

```
)QBATCH/I
))F/AS/S/L
))X,FORT,MYFILE
)))
```

OR

```
)QBATCH/V XEQ MASM FILES
```

EXAMPLE 3:

```
)CONTROL @EXEC STACK @CRA
```

IN CARD READER:

```
$$JOB,username
$$PASSWORD,WHATEVER
   "
   "
   "
$$END
:EOF (ALL HOLES PUNCHED)
```

If a user needs mag tape but doesn't know what physical units are available, she can request that the operator mount storage media on any free unit via the MOUNT command.

EXAMPLE:

)MOUNT logicalname message

EXEC passes "message" to operator and holds user until ready.

POSSIBLE REPLIES:

- MOUNTED - Media ready to use.
- REFUSED - Can't find media, no unit free.
- NO OPERATOR - By setting operator switch in EXEC off, all operator action requests can be automatically refused.

)CONTROL @EXEC OPER ON

MOUNT,MYNAME,MESSAGE

DISMOUNT,MYNAME

)CONTROL @EXEC MOUNTED [/MID=mount id] [@devicename] [volid]

)CONTROL @EXEC REFUSED [mount id]

)CONTROL @EXEC MOUNTSTATUS [mount id]

)CONTROL @EXEC UNITSTATUS

)CONTROL @EXEC DISMOUNTED [mount id]

)CONTROL @EXEC PREMOUNT [/IBM] @unit volid username

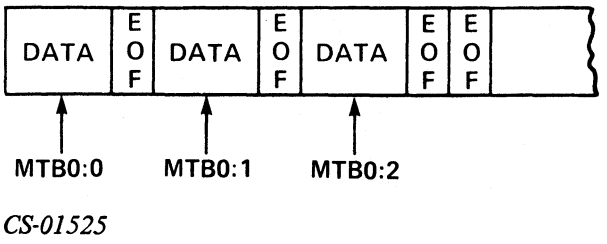


Figure 8.3 Unlabeled Magnetic Tape

- Mag tape files referenced by number.
- Each mag tape file can consist of multiple disc files.
- Files cannot span tape reels (volumes).
- Device referenced by physical name - e.g., @MTB0.

LABELED TAPES

AOS/VS SUPPORTED LABELED TAPES

1. AOS/VS Format	Default format used by AOS/VS CLI
2. ANSI Format Levels 1, 2, or 3	Use only for interchange from/to non-AOS/VS systems
3. IBM Format Levels 1 or 2	

ADVANTAGES OF LABELED TAPE

1. Standard formats allows communication with other operating systems.
2. You can reference your file by name rather than physical file number.
3. A logical file may span several physical reels of tape.

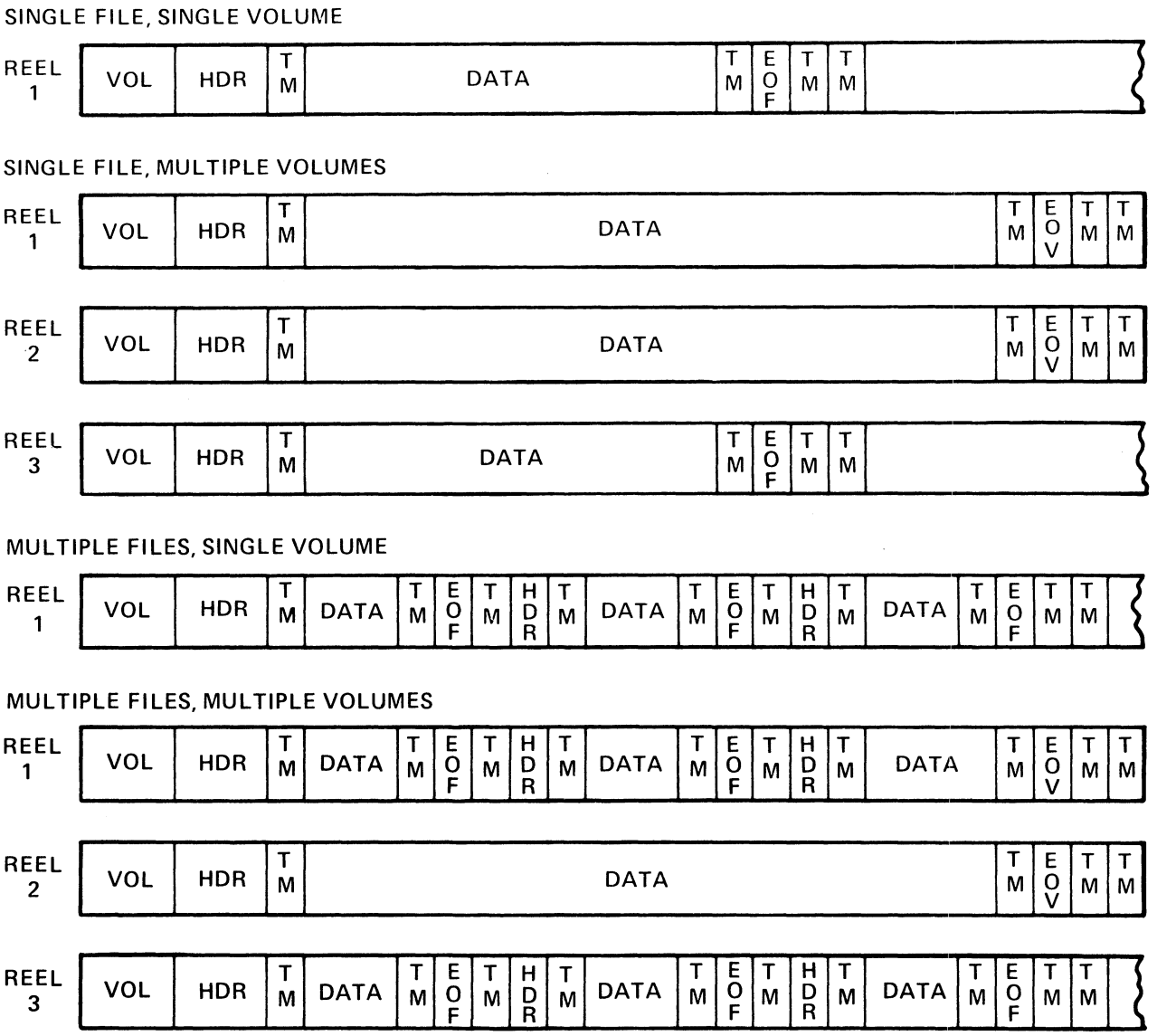
TYPES OF LABELS

1. VOLUME (VOL) LABEL - Identifies the reel of mag tape. Occurs only at the start of the reel.

2. FILE HEADER (HDR) LABELS - Identifies the mag tape file and its characteristics. Occurs at the beginning of each mag tape file.

3. END OF FILE (EOF) LABELS - Identifies the end of a file. Occurs at the end of every file.

4. END OF VOLUME (EOV) LABELS - Indicates that a file continues on another reel of tape. Occurs at the end of a reel.



CS-01397

Figure 8.4 Labeled Magnetic Tape

LEGAL LABEL CHARACTERS

1. Upper case letters A-Z
2. Digits 0-9
3. Characters SPACE ! " % & ' ( ) * +
   , - . / : ; < > ?

LABEL UTILITY

)X LABEL device-name volume-id

@MTA0 six characters

/I Create IBM labels

/S Scratch the tape

NOTE: When referencing labeled tapes, the device-name becomes @LMT.

PRIMING EXEC

Before EXEC can be operated on a master logical disc, the QUEUE structure must be initialized.

1.)PROC/DIR=@/DEF :UTIL:EXEC

   Wait for the message announcing EXEC then:

2.)CONTROL @EXEC CREATE PRINT LPT

)CONTROL @EXEC OPEN BATCH_INPUT

)CONTROL @EXEC OPEN BATCH_OUTPUT

)CONTROL @EXEC OPEN BATCH_LIST

)CONTROL @EXEC OPEN LPT

   Wait until EXEC displays the time of day five times, then:

3.)TERMINATE OP:EXEC

   At this point, the permanent EXEC spool queues have been created in the file :QUEUE:Q.

EXEC CREATION

- The EXECUTIVE PROCESS is created as a swappable process with all privileges granted to it.
- Only one process running the EXECUTIVE program is permitted.
- To create a process running the EXECUTIVE program:

```
)PROC/DIR=@/DEF/NAME=EXEC :UTIL:EXEC
```

SAMPLE UP.CLI

```
[!EQUAL,1,2]
[!EQUAL,([!NEQUAL,002,[!PID]]ERROR[!END]%/%%-%),()]
SEARCHLIST :UTIL : :PER
PUSH
SUPERUSER ON
ACL @MT- +,WARE
PROCESS/DEFAULT/DIRECTORY=@/NAME=EXEC EXEC
PAUSE 2.000
CONTROL @EXEC ENABLE @CON([CONSOLES])
PAUSE 10.000
CONTROL @EXEC ((CONTINUE 1) VERBOSE)
CONTROL @EXEC START (BATCH_<OUTPUT LIST> LPT) @LPB
CONTROL @EXEC CONTINUE @LPB
POP
PROMPT TIME CHECKTERMS
PROC/RES/SUPERU/ACCESSDEVICES/IPCUSAGE/NAME=GSMGR/DIR=@&
/DEF/DATA=MAGIC.SYNC :GSMGR
EXECUTE LOCK_CLI
[!ELSE]
WRITE *ERROR*
WRITE %0\% IS VALID WHEN INVOKED BY THE ROOT CLI
WRITE REQUIRED ARGUMENTS: NONE
WRITE OPTIONAL SWITCHES:,,NONE
[!END]
[!ELSE]
WRITE NON-EXECUTABLE SAMPLE...YOU CAN EXECUTE %0\% BY MAKING
WRITE THE ARGUMENTS IN THE COMMAND ON THE FIRST LINE EQUAL AND
WRITE MOVING THIS MACRO TO THE ROOT DIRECTORY.
[!END]
```

EXEC START-UP DIALOG

```
FROM PID 3 :  (EXEC) 8:24:03
8:24:06
)
FROM PID 3 :  (EXEC) GROWING TO 18 PAGES

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON1

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON2

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON3

FROM PID 3 :  (EXEC) GROWING TO 19 PAGES

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON4

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON5

FROM PID 3 :  (EXEC) STREAM_1 CONTINUING

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON6

FROM PID 3 :  (EXEC) STREAM_1 [IDLE]

FROM PID 3 :  (EXEC) GROWING TO 20 PAGES

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON7

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON8

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON9

FROM PID 3 :  (EXEC) @LPB CONTINUING

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON10

FROM PID 3 :  (EXEC) @LPB PHYSICAL UNIT OFFLINE <------  (note)

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON11

FROM PID 3 :  (EXEC) 8:24:39

FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON12

FROM PID 3 :  (EXEC) @LPB CO-OPERATIVE TERMINATED <------ (note)
```

```
FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON13
FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON14
FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON15
FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON16
FROM PID 3 :  (EXEC) ENABLED CONSOLE, @CON17
FROM PID 3 :  (EXEC) GROWING TO 21 PAGES
```

```
[!EQUAL,1,1]
[!EQUAL,([!NEQUAL,002,[!PID]]ERROR[!END]%/%%-%),()]
TERMINATE/2=ERROR OP:EXEC
PAUSE 2.000
TERMINATE/2=ERROR OP:GSMGR
PAUSE 2.000
CHECKTERMS
[!ELSE]
WRITE *ERROR*
WRITE %0\% IS VALID ONLY WHEN INVOKED BY THE ROOT CLI
WRITE REQUIRED ARGUMENTS:  NONE
WRITE OPTIONAL SWITCHES:,,NONE
[!END]
[!ELSE]
WRITE NON-EXECUTABLE SAMPLE...YOU CAN EXECUTE %0\% BY MAKING
WRITE THE ARGUMENTS IN THE COMMAND ON THE FIRST LINE EQUAL AND
WRITE MOVING THIS MACRO TO THE ROOT DIRECTORY.
[!END]
```

EXEC sends unsolicited messages to console of parent process:

- Errors
- Logon/off
- Batch job begin/end
- Mount/Dismount

Any process with same user name as EXEC (user_name = OP) can issue commands to EXEC.

Responses from EXEC are sent to the console of the commanding process.

EXEC-AOS/VS INTERACTION

AOS/VS

- Enforces privilege limits
- Enforces memory and disc limits
- Protects files by username
- Heuristic scheduling of swappable processes
- Records resource usage

EXEC

- Monitors specified consoles
- Links password with username
- Creates swappable processes
- Only recognizes users with profiles
- Listens for control commands
- Logs extra statistics
- Runs batch streams
- Performs device spooling

MODULE 9

OPERATOR INFORMATION UTILITIES

MODULE 9
OPERATOR INFORMATION UTILITIES

OBJECTIVES

Upon successful completion of this module, the student will be able to:

1. Invoke PED to monitor system activity.

2. Control the SYSLOG facility and execute the REPORT generator program to provide SYSLOG information.

OPERATOR INFORMATION UTILITIES

PED - Process Environment Display - ACTIVE PEDS ON SYSTEM
XPEDS/ALL FOR ALL PEDS

SYSLOG - System Log

REPORT - Report System Log Information

PROCESS ENVIRONMENT DISPLAY: PED

The AOS/VS system utility PED is used to display information about all processes on your AOS/VS system. PED can be called with or without arguments, and will display various defaulted information.

PED accepts single letter commands to alter its operation. The following commands are supported by PED:

A	Toggle the status of the /ALL switch - WHILE RUNNING
B	Bye
E	Exit
H	Help
Q	Quit
R	Refresh the screen (useful after sends, etc)
^S	Freeze display
^Q	Resume display (unfreeze)
V	Decrement Minpid by 5 > SCROLL SCREEN BY 5
^	Increment Minpid by 5 >
<	Decrease the sleep time by one second
=	Reset the sleep time to 10 seconds
>	Increment the sleep time by one second

SCREEN UPDATES EVERY 10 SECONDS

The default time between screen updates is 10 seconds. Users wishing to modify this parameter should use the /CYCLE=N switch, where N is the number of seconds to sleep. The minimum cycle time is one second.

PED displays elapsed and CPU time in the following format:

DD-HH	4 days, 3 hours	would be displayed as 4-03
HH/MM	3 hours, 27 minutes	would be displayed as 3/27
MM:SS	27 minutes, 3 seconds	would be displayed as 27:03
SS.hh	3 and 1/2 seconds	would be displayed as 3.50

Table 9.A PED Switches

SWITCH	DESCRIPTION
/ALL	DISPLAY ALL PIDS IN USE (DEFAULT IS ACTIVE PIDS)
* /BS	DISPLAY THE BLOCKED AND SWAPPED STATUS
* /CPU	DISPLAY CPU TIME
/CPUS	DISPLAY CPU TIME USAGE RATE
/CYCLE=<n>	SLEEP FOR <n> SECONDS BETWEEN UPDATES
* /ELAPSED	DISPLAY ELAPSED TIME
/FATHER	DISPLAY PROCESS FATHER
/FLAG<n>	DISPLAY FLAG WORD <n>, WHERE <n> IS IN THE RANGE 1-5
* /FTA	DISPLAY (LOGICAL + PHYSICAL) PAGE FAULTS
/FTAS	DISPLAY (LOGICAL + PHYSICAL) PAGE FAULT RATE
/FTL	DISPLAY LOGICAL PAGE FAULTS
/FTLS	DISPLAY LOGICAL PAGE FAULTS RATE
/FTP	DISPLAY PHYSICAL PAGE FAULTS
/FTPS	DISPLAY PHYSICAL PAGE FAULTS RATE
* /IO	DISPLAY I/O USAGE
/IOS	DISPLAY I/O USAGE RATE
/IREC	DISPLAY NUMBER OF PROC'S TASKS SUSPENDED ON ?IREC
/LISTFILE(=FILE SPEC)	DIRECT PED OUTPUT TO A FILE
/MINPID=<n>	DISPLAY NO PIDS LESS THAN <n>
/PAGESECONDS	DISPLAY PAGE-SECONDS
* /PID	DISPLAY PROCESS ID
/PNQ	DISPLAY PRIORITY ENQUEUE FACTOR
/PRIORITY	DISPLAY PROCESS PRIORITY
/PRIVBITS	DISPLAY PROCESS PRIVILEGE WORD
* /PROCESS	DISPLAY PROCESS NAME
* /PROGRAM	DISPLAY PROGRAM NAME
/PSW	DISPLAY PROCESS STATUS WORD
/SH	DISPLAY CUMULATIVE NUMBER OF SHARED PAGES IN RINGS 3-7
* /SH<n>	DISPLAY NUMBER OF SHARED PAGES IN RING <n> (RINGS 3-7) DEFAULT DISPLAY SHOWS RING 7
/SUBSLICES	DISPLAY NUMBER OF SUBSLICES IN TIME SLICE
/SUPERMODE	DISPLAY SUPER<PROCESS,USER> MODE IN CLI PROMPT FORMAT
/SUPERPRIVILEGE	DISPLAY SUPER<PROCESS,USER> PRIVILEGES IN CLI PROMPT FORMAT

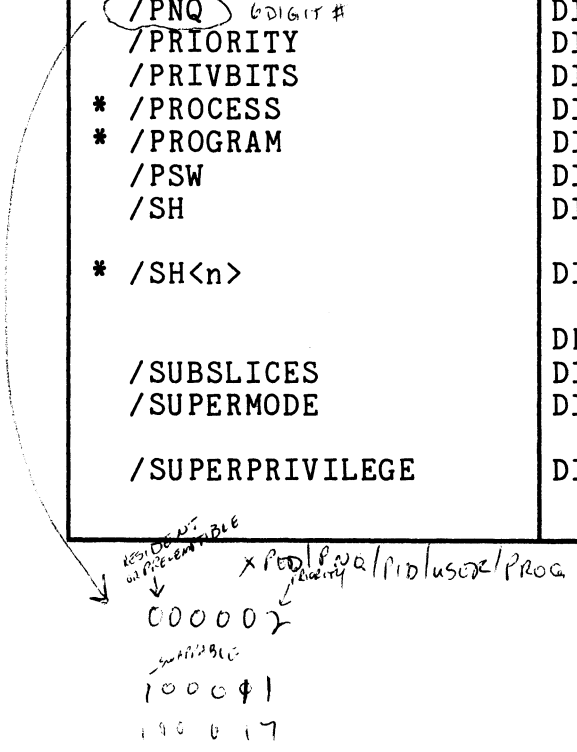


Table 9.A PED Switches (Cont.)

SWITCH	DESCRIPTION
/TSE	DISPLAY TIME SLICE EXPONENT
/US	DISPLAY CUMULATIVE NUMBER OF UNSHARED PAGES IN RINGS 3-7
* /US<n>	DISPLAY NUMBER OF UNSHARED PAGES IN RING <n> (RINGS 3-7) DEFAULT DISPLAY SHOWS RINGS 3 AND 7
* /USERNAME	DISPLAY USER NAME
* /WSS	DISPLAY CURRENT WORKING SET SIZE
/WSSMAX	DISPLAY MAXIMUM WORKING SET SIZE
/WSSMIN	DISPLAY MINIMUM WORKING SET SIZE

Invoking PED with no arguments is equivalent to typing:

```
PED/CYCLE=10/PID/USER/PROGRAM/PROCESS/WSS/FTA/IO/CPU/ELAPSED/US7/SH7/BS
```

SAMPLE "PED" DISPLAY (DEFAULT)

```
)X PED

PID USERNAME PROCESS  PROGRAM    ELAPS    CPU  BS  SH7  US7   I/O  FTA  WSS
  1 PMGR     PMGR     PMGR        1/21 38.83       0    1    29  164   53
  2 OP       OP       LOOK_CLI    1/21  1.01  B   18    2   108  125   61
  3 OP       EXEC     EXEC        1/20  2.42  B    1   22   626   53   65
  4 OP       LPB      XPLT        1/19  1.79       3    3   123   21   52
  5 OP       INFOS    INFOS_II16  1/19  1.00       9   21   168    7   75
  6 TOM      CON7     CLI        14:50  0.07  B   18    2     8   46   54
  7 JACK     007      PED         5:40  3.60  B   17  495     0   55   65
  8 JACK     CON4     CLI        14:14  0.11  B   18    2     0   40   51
  9 TOM      009      PED         1.00  0.05      17  495     0   43   53
10 CHRIS     CON3     CLI        12:59  1.39  B   18    2   191   99   62

PED REV 1.20.00.00 FRIDAY 13-MAR-81  10:19:25 AM  AOS/VS REV 1.20.00.00
```

SAMPLE "PED"DISPLAY (SWITCHES)

```
)X PED/ALL/PID/USERNAME/FATHER

PID USERNAME  PROGRAM      FP
  1 PMGR      PMGR          0
  2 OP        LOOK_CLI      0
  3 OP        EXEC          2
  4 OP        XLPT          3
  5 OP        INFOS_II16    2
  6 TOM       CLI           3
  7 JACK      PED           8
  8 JACK      CLI           3
  9 TOM       PED           6
 10 CHRIS     CLI           3

PED REV 1.20.00.00 FRIDAY  13-MAR-81  10:25:15 AM  AOS/VS REV 1.20.00.00
```

The CLI command SYSLOG permits the collection of information on various kinds of system activities. The following information is collected in a log file in the root directory:

- Usernames of system users
- Process elapsed time
- CPU time
- I/O disc blocks read or written
- Integral of page usage
  = CPU time x number of main memory pages
- Hardware errors

Any additional user-specified information, if specified in SYSLOG format, is also collected in a log file.

When EXEC is running, the following is also recorded:

- Disc space used by each user
- Console connect time for each user
- Tape and disc mount requests
- Privileged user logons

)SYSLOG	See current setting.
)SYSLOG/START [filename]	Start system logging.
)SYSLOG/STOP	Stop logging.
)SYSLOG filename	Rename :SYSLOG to filename

NOTE: To rename, SYSLOG must be stopped. :SYSLOG will be recreated when the log is restarted.

The AOS/VS system utility REPORT is used to generate system statistics from one or more SYSLOG files.

```
XEQ REPORT[/SWITCHES] [PATHNAME] ...  [NL]
```

The following switches are used to modify the default values and produce detail reports:

```
/L

/L=pathname

/USERS=user1+..+userN

/DEVICES=dev1+...+devN

/BEFORE=dd-mmm-yy[:hh:mm:ss]

/AFTER=dd-mmm-yy[:hh:mm:ss]

/ATU
/C
/CT
/ERCC
/EV
/FA
/FATAL
/FE
/HANG
/I
/IC
/IOC
/MT
/NA
/NE
/PP
/PT
/PR
/PW
/RA
/SA
/SC
/SCP
/TA
/TE
/X
```

DEFAULT REPORT PRINTOUT

5-FEB-81 13:47:14 REPORT REV 03.30

USER SUMMARY FROM FILE(S): :SYSLOG

USERNAME	CONNECT TIME CONSOLE	UNIT	PAGES PRINTED	CPU TIME	I/O BLOCKS	PAGE SECS
	0:00	0:33	0	0:00:00.000	0	0.000
ASN.S220	0:05	0:00	0	0:00:02.843	1	61.927
AXGAIL	0:01	0:00	0	0:00:04.378	192	140.140
AXJANET	2:34	0:00	11	0:00:21.659	252	526.338
AXNANNETTE	4:05	0:00	49	0:04:00.608	5541	6744.282
AXSHEILA	0:01	0:00	0	0:00:00.127	0	2.710
*BACKUP	0:49	0:00	15	0:02:08.664	63298	3919.625
CALDWELL.S220	0:18	0:00	0	0:00:16.601	119	261.076
CDL.S220	2:13	0:00	0	0:00:58.442	168	964.207
DAVE.S220	1:32	0:00	0	0:00:30.769	295	595.751
FIRST.S220	0:42	0:00	0	0:01:00.016	511	1106.373
FOO.S220	0:03	0:00	0	0:00:01.181	6	25.879
HIBLER.S220	1:12	0:00	0	0:00:21.421	38	290.862
JAM.S220	1:34	0:00	0	0:01:05.204	638	1140.852
*NUNES	4:03	0:00	9	0:04:41.140	2160	8287.540
OOO.S220	0:07	0:00	0	0:00:01.884	8	40.990
OP	0:00	0:00	15	0:02:02.382	4267	2319.555
PED	6:32	0:00	0	0:02:12.741	8	1323.121
*ROSANNE	4:22	0:00	68	0:12:54.644	34508	23842.336
*TOM	2:11	0:00	0	0:01:48.822	1229	1958.829
*TXAUG	16:49	0:00	29	0:05:02.021	1842	8908.117
*TXJACKW	5:21	0:00	70	0:01:42.576	1674	2540.873
*TXMASON	3:34	0:00	11	0:01:33.859	2659	3148.950
*TXPETERSEN	2:32	0:00	12	0:00:13.032	276	326.967
VXAL	2:07	0:00	0	0:03:12.291	540	5162.855
VXCOSTLEY	1:08	0:00	29	0:00:36.517	324	1187.490
VXGARYS	5:45	0:00	0	0:09:56.514	14315	20174.097
VXKATZ	1:09	0:00	0	0:00:53.408	537	1571.431
VXTAILBY	1:38	0:00	8	0:00:28.603	330	897.219
VXWAGNER	0:35	0:00	0	0:00:02.746	31	62.547
WXANN	1:14	0:00	87	0:08:32.054	346	13308.301
WXSOUTHLAND	0:54	0:00	0	0:01:59.852	1546	3221.337
*XXCAROL	4:41	0:00	445	0:04:33.656	8565	9501.082

5-FEB-81 13:47:14 REPORT REV 03.30

DEVICE SUMMARY FROM FILE(S): :SYSLOG

DEVICE CODE	UNIT	ERRORS HARD	SOFT
22	0	2	6

OTHER REPORT PRINTOUTS

*** /PT ***

5-FEB-81 14:01:05 REPORT REV 03.30

PROCESS TERMINATION DUMP

	PROCESS NAME	ELAPSED TIME	CPU TIME	I/O BLOCKS	PAGE SEC
4-FEB-81 13:02:10	VXGARYS:019	0:01:46	0:00:04.698	511	155.383
4-FEB-81 13:26:31	TOM:030	0:05:25	0:00:25.750	42	386.021
4-FEB-81 13:27:24	TOM:CON2	0:06:36	0:00:00.831	93	19.240

*** /DE ***

DEVICE ERROR DUMP

	DEVICE CODE	UNIT	STATUS	RETRIES	TYPE
4-FEB-81 16:12:13	22	0	106101	1	SOFT
5-FEB-81 12:30:35	22	0	106101	1	SOFT
5-FEB-81 12:44:21	22	0	106107	13	HARD

*** /CT ***

CONSOLE CONNECT TIME DUMP

	USERNAME	CONNECT TIME	DEVICE
4-FEB-81 13:27:25	TOM	0:06	CON2
4-FEB-81 13:31:05	VXWAGNER	0:34	CON20
4-FEB-81 13:39:49	WXANN	0:32	CON4

*** /MT ***

UNIT MOUNT DUMP

	USERNAME	CONNECT TIME	DEVICE
4-FEB-81 16:39:51		0:33	MTA0

*** /PR ***

PRIVILEGED USER LOGON DUMP

	USERNAME
4-FEB-81 12:52:52	XXCAROL
4-FEB-81 12:53:05	ROSANNE
4-FEB-81 13:20:48	TOM

*** /PP ***

PAGES PRINTED DUMP

	USERNAME	PAGES	DEVICE
4-FEB-81 13:36:31	VXCOSTLEY	22	LPB
4-FEB-81 14:06:41	WXANN	38	CON17
4-FEB-81 14:28:04	OP	2	LPB

MODULE 10

SYSTEM BACKUP

MODULE 10
SYSTEM BACKUP

OBJECTIVES

Upon successful completion of this module, the student will be able to:

1. Explain the concept of a logical disc unit.
2. Graft a logical disc into a system.
3. Use the PCOPY utility to back up and restore entire discs.
4. Use the DUMP utility to back up user files.
5. Use the LOAD utility to restore user files.

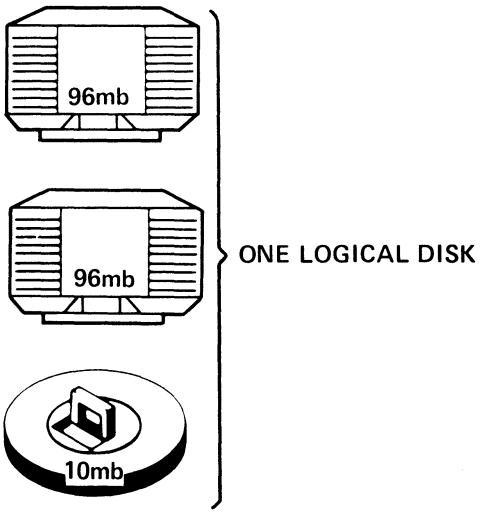
LOGICAL DISC (LD) - Grouping of one or more physical discs which the system treats as a logical entity.

Each LD is a file hierarchy or free-standing tree structure.

Each LD has a local root directory.

The logical disc containing the system root directory (:) is the Master LD.

The logical disc must have all packs mounted before it can be initialized.



CS-01398

Figure 10.1 A Logical Disc

The above LD would have been created by the Disc Formatter utility.

GRAFTING LDs

You can graft one or more LDs to the MASTER LD, and graft other LDs to those.

The grafted LDs file hierarchy now becomes part of the existing file structure.

Each LD in the disc hierarchy must be initialized for AOS/VS to see all the physical discs in the LD.

The root directory of the MASTER LD has execute access and cannot be changed. The local root directories of grafted LDs can have rights that were set up when the disc was originally formatted.

GRAFTING LOGICAL DISCS

GRAFTING AN LD - the ability to add an LD to the existing directory tree structure.

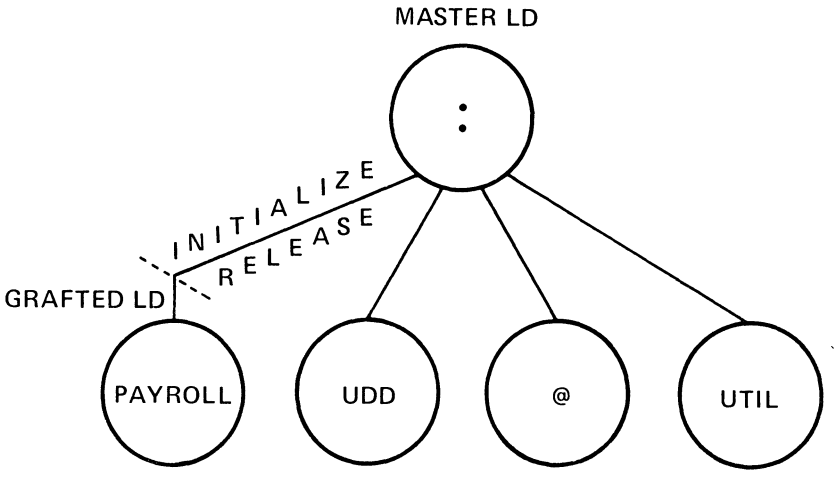


Figure 10.2 Grafting Logical Discs

PCOPY

This utility is a fast, stand-alone disc DUMP/LOAD program.

PCOPY does not dump unused disc blocks, nor any block occupied by BOOT.SV.

It is faster than using CLI commands.

It lets you dump onto successive reels of labeled magnetic tape, disc, or 1.26 megabyte diskette.

You can use PCOPY even if your logical disc has not been fixed with FIXUP.

An entire logical disc must be dumped or restored.

Specific directories or files cannot be selected.

A copy of the utility is kept in the root directory. The complete pathname to it is :PCOPY.

A copy of this program has to be loaded into low core for execution to begin.

SAMPLE PCOPY DIALOGUE DISC TO TAPE

TO BOOT FROM MAG TAPE:

```
SCP-CLI> reset
SCLP-CLI> boot 22
FROM MTB0:1
```

```
AOS/VS DISK COPIER, REV XX.XX


ENTER TODAY'S DATE (MM DD YY)?   11 30 81
IS SOURCE A LOGICAL DISK (D) OR LABELED TAPE(T)   d

SPECIFY SOURCE LDU
ENTER NAME OF EACH UNIT IN THE LDU (<NL>) WHEN DONE
DISK UNIT NAME? dpf0
DEVICE CODE?  27
DISK UNIT NAME?

COPY TODISK (D) OR LABELED TAPE (T)? t
TAPE UNIT NAME?  mtb0
SPECIFY VOLUME ID     pc00000
TAPE UNIT NAME?

EXPIRATION DATE (DEFAULT IS mm dd yy)   12 31 81
COPY TO DISK FROM TAPE.  PLEASE CONFIRM (N/Y)?   y
CONFIRMED

GENERATION NUMBER [current]?  1234


DONE!

CPU HALTED

SCP-CLI>
```

9-53 + 9-54 INAOS/VS B/G MANUAL

The combined commands DUMP and LOAD can do basically the same thing as the PCOPY utility.

The DUMP command will create a DUMPFILE which will contain the directory information and data for each file specified.

The LOAD command will create files in the working directory according to the information supplied in the DUMPFILE.

The DUMPFILE is normally on magnetic tape.

To create a DUMPFILE:

```
)DUMP dumpfile [source-pathname]...
```

If no source-pathname is specified, # is assumed.

By default the blocks written to magnetic tape will be 2048 bytes in size. To pack more data on the magnetic tape you can override the blocksize with the switch /BUFFERSIZE=n where n is a number in the range of 1 to 8192.

To load one or more previously-dumped files into the working directory:

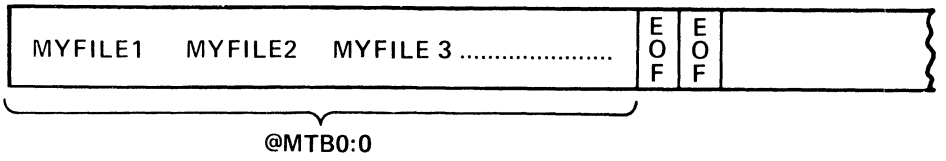
```
)LOAD dumpfile [source-pathname]...
```

If no source-pathname is specified, all files in the dumpfile will be loaded, provided they do not already exist.

By default, CLI will reserve a buffersize of 2048 bytes into which it will read the blocks from the magnetic tape. If the blocksize is larger than 2048 bytes, the buffersize switch must be used.

UNLABELED TAPE - DUMP/LOAD

```
)DIR :UDD:MYDIR
)DUMP/V/AFTER/TLM=24-NOV-80 @MTB0:0 MYFILE+
)DELETE/V MYFILE+
)LOAD/V @MTB0:0
```

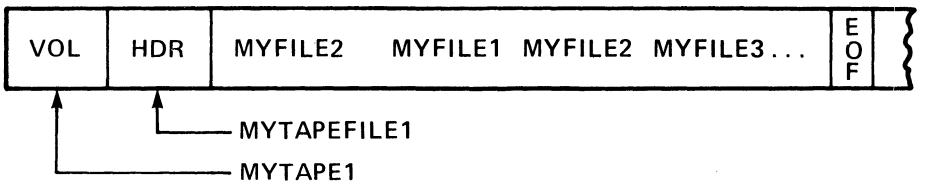


CS-01400

Figure 10.3 DUMPed Files on Unlabeled Magnetic Tape

LABELED TAPE - DUMP/LOAD

```
)MOUNT ATAPE MOUNT A SCRATCH TAPE
)X LABEL ATAPE MYTAPE
)DISMOUNT ATAPE

)DIR :UDD:MYDIR
)MOUNT/VOLID=MYTAPE ATAPE PLEASE MOUNT BLANK TAPE WITH RING
)DUMP/V/BUFFERSIZE=8192 ATAPE:MYTAPEFILE1 MYFILE+
)DISMOUNT MYTAPE
                                    OR
)DUMP/V/BUFFERSIZE=8192 @LMT:MYTAPE:MYTAPEFILE1+
)DELETE/V MYFILE+
)MOUNT/VOLID=MYTAPE ATAPE NO RING
)LOAD/V ATAPE:MYTAPEFILE1
                                    OR
)LOAD/V @LMT:MYTAPE:MYTAPEFILE1
```



CS-01401

Figure 10.4 DUMPed Files on Labeled Magnetic Tape

MODULE 11

SYSTEM PROBLEMS

MODULE 11
SYSTEM PROBLEMS

OBJECTIVES

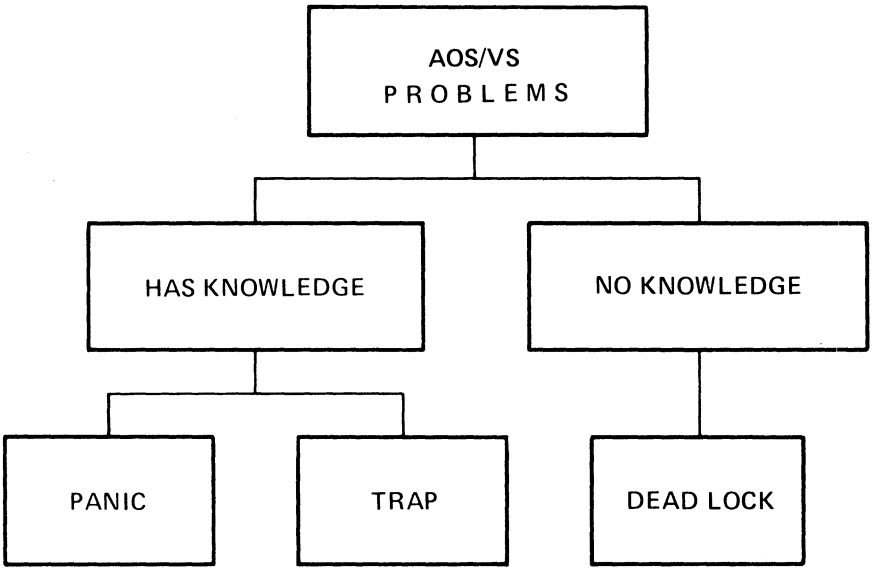
Upon successful completion of this module, the student will be able to:

1. Recognize the difference between system panics, traps, and deadlocks.

2. Respond correctly to system panics, deadlocks, and traps.

3. Use the emergency termination facility.

4. Use the file fixup procedure (FIXUP).



CS-01527

Figure 11.1 Possible AOS/VS Problems

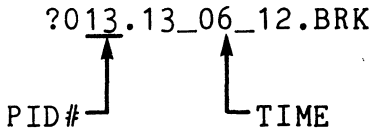
FAILURE OF A USER PROGRAM

CAUSED BY AN ATTEMPT TO:

- Read, write or execute code protected against any of these actions
- Refer to an address outside of user space or an invalid address
- Use more than 16 levels of indirect addressing
- Issue an I/O instruction while I/O mode is not enabled
- Transfer control to a lower ring illegally

AOS/VS ACTION:

- Terminates process
- Produces a "BREAK FILE"



MESSAGE:

```
*ABORT*
USER TRAP, "message"
```

C	PC	AC0	AC1	AC2	AC3
n	nnnnnnnnnnn	nnnnnnnnnnn	nnnnnnnnnnn	nnnnnnnnnnn	nnnnnnnnnnn

OPERATOR ACTION:

- Record message
- Save "BREAK FILE"
- Notify programmer

FAILURE OF:

- System data base
- System hardware

AOS/VS TERMINATES

MESSAGE:

FATAL AOS/VS ERROR MMMmmm

MMM = Major panic code number

mmm = Minor panic code number

AOSVS.PANICS.SR - List of AOS/VS panic codes

OPERATOR ACTION:

- Produce a core dump - Mirror Image of What is in Memory
- Run EMERGENCY SHUTDOWN
- Run FIXUP, if ESD fails
- Re-bootstrap the system

SYSTEM DEADLOCK

SYMPTOMS:

- No system response on user console
- Little or no CPU activity
- A process does not respond to CTRL C - CTRL B

MIGHT BE:

- A malfunctioning high-priority resident or pre-emptible process
- Poor system organization
- The system may be denying service

SOLUTION:

- Try to terminate processes causing problem
- Terminate the OP process
- Treat as a fatal error (force a core dump and ESD)

```
Hit BREAK key

SCP-CLI> HALT

SCP-CLI> .

AC0           AC1           AC2           AC3
xxxxxxxxxxx   xxxxxxxxxxx   xxxxxxxxxxx   xxxxxxxxxxx

PC            C     MAP
xxxxxxxxxxx   x     ATU


SCP-CLI> RESET

SCP-CLI> START 50


AOS/VS PROCESSING ABORTED
STRIKE "D" FOR AOS/VS DUMP OR "S" FOR AOS/VS SHUTDOWN
OR "R" FOR OPEN FILE REPORT
?  D


AOS/VS SYSTEM DUMP
LOAD TAPE ON MTB0
STRIKE "D" WHEN READY:  D

DONE.

STRIKE "S" FOR AOS/VS SHUTDOWN, OTHER KEY TO HALT
```

If CORE DUMP routine fails, the system console displays:

```
error message
BAD MEMORY DUMP
RESTART MEMORY DUMP?
```

```
SCP-CLI> CONTINUE
```

EMERGENCY SHUTDOWN (ESD)

* To force - Follow CORE DUMP routine

* Fatal AOS/VS error automatically executes CORE DUMP routine

After CORE DUMP:

```
STRIKE "S" FOR AOS/VS SHUTDOWN, OTHER KEY TO HALT.
? S

FILE SYSTEM RESTART
NOW RESTARTING DEVICE 27  UNIT 1
NOW RESTARTING DEVICE 27  UNIT 0

FLUSHING BUFFERS

OPEN FILE PROCESSING

SYSTEM SHUTDOWN
```

THEN BRING BACK UP

Note: Do not run ESD if you get:

FATAL AOS/VS ERROR 14047
or
FATAL AOS/VS ERROR 14077

You may compromise the disk !!

FIXUP

A utility program which is run to locate and, optionally, to correct errors in disc file structures.

MAY BE RUN PERIODICALLY:

- To delete temporary files
- As an early warning technique to discover potentially troublesome directory errors

MUST BE RUN AFTER:

- A fatal system error
- A system deadlock you cannot resolve
- A fatal hardware error
- An abnormal system shutdown
- ESD fails

MAY BE RUN EITHER STAND-ALONE OR UNDER AOS/VS CONTROL.

Execution:

- Under AOS/VS control

```
)XEQ FIXUP.PR
```

- Stand-Alone (Bootstrap from mag tape)

```
SCP-CLI> RESET
SCP-CLI> BOOT 22

FROM MTB0:  1
```

FIXUP dialog:

```
AOS/VS DISK FIXER REV xx.xx

SPECIFY EACH DISK UNIT IN THE LOGICAL
DISK

DISK UNIT NAME? DPF0
DEVICE CODE?
DISK UNIT NAME? DPF1
DEVICE CODE?

VERBOSITY (0-2)? 2

ERROR LOG FILE? @LPB

SHOULD I REPORT CLOSING FILES? Y

MAY I FIX IT? Y

DONE!
```

SEE (6-34 & 6-35)
6-28
IN AOS/VS

APPENDIX

SCP DTOS UTILITY

The SCP DTOS is a utility that allows you to run diagnostics on the SCP-MV/Family interface and on MV/Family circuit boards.

MV/8000

```
SCP-CLI> XEQ DTOS

*
```

MV/6000

```
Mount MV/6000 System tape.
Press the CPU LOCK switch to UNLOCK
Press CPU RESET

  BOOT DEVICE?  22
  TESTOK

  *
```

DTOS COMMANDS:

```
ACCEPT
EXIT
HELP          (MV/8000 only)
LOAD filename
RUN [n]
DIR
```

SCP DTOS also recognizes many of the standard DTOS commands.

SCP ERRLOG UTILITY

ERRLOG is a utility that allows you to display log entries recorded on the SCP-OS diskette (MV/8000) or recorded in SCP memory (MV/6000).

A log entry is either an error or key system information such as power on/off and ELOG flag on/off.

The SCP can record log entries while in either command mode or TTY mode, providing the ELOG flag is set. The SCP cannot record entries while running the DTOS utility.

SCP LOG FILES

1. ERRORLOG file - contains all hard errors and key system information.

2. IOC file - contains I/O channel parity errors.

3. MEMLOG file - contains single-bit ERCC errors.

To initiate:

```
SCP-CLI> XEQ ERRLOG

SCP-ERR>
```

ERRLOG COMMANDS:

```
CLI
HELP
INIT
LIST
TIME
```

Note: MV/6000 error files are kept in SCP memory. They will be lost when power is turned off.

SCP MCODE UTILITY

MCODE is the utility that allows you to load and verify MV/8000 microcode and the scratchpad.

TO INITIATE:

```
SCP-CLI> XEQ MCODE

SCP-MCODE>
```

MCODE COMMANDS:

```
CLI
FLAGS
HELP
LOAD    (MV/8000 only)
LSPAD   (MV/8000 only)
STATUS  (.)
VERIFY  (MV/8000 only)
VSPAD   (MV/8000 only)
```

SCP MEMORY UTILITY
(MV/8000 only)

The MEMORY utility allows you to verify and load MV/8000 main memory locations.

While in MEMORY, you can examine and modify main memory, just as in the SCP CLI.

In addition, MEMORY allows you to load programs that test peripherals.

TO INITIATE:

```
SCP-CLI> XEQ MEMORY

SCP-MEM>
```

MEMORY COMMANDS:

```
CLI
DISPLAY
EXAMINE
FLAGS
HELP
LOAD
STATUS
VERIFY
```

DataGeneral

Data General Corporation, 4400 Computer Drive, Westboro, MA 01580
(617) 366-8911