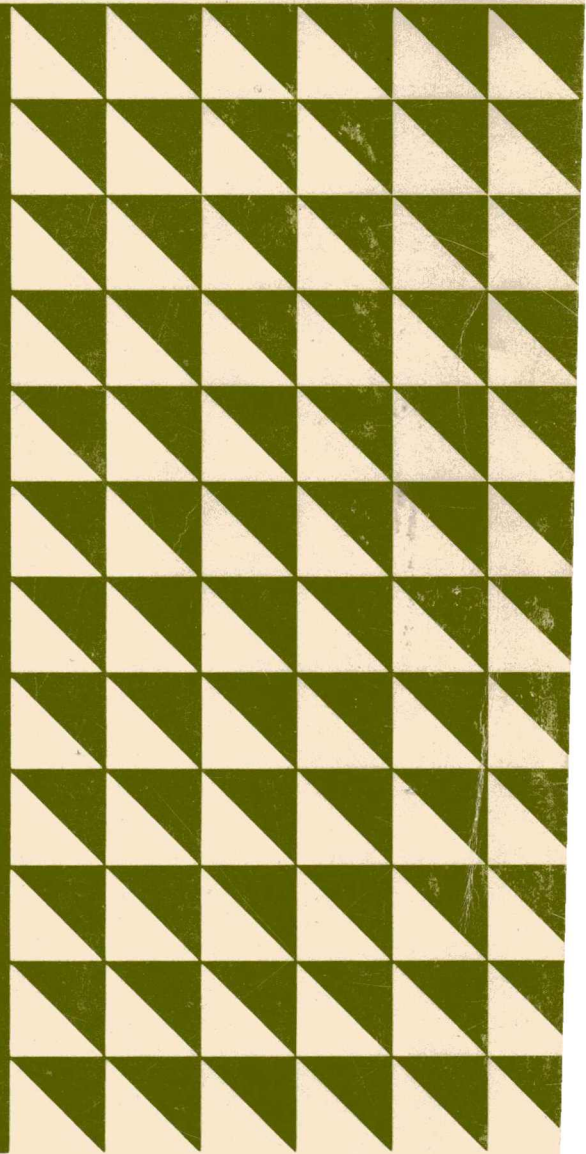




**Introduction to IBM 4300
and DOS/VSE Facilities**

Two empty rectangular boxes, one above the other, likely for a library stamp or identification number.



Student Text



Introduction to IBM 4300 and DOS/VSE Facilities

Student Text

Minor Revision, May 1980

This is a minor revision of the previous edition with small corrections.

All rights reserved. No portion of this text may be reproduced without express permission of the author.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available outside the United States.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality. Address comments concerning the contents of this publication to IBM Corporation, Publications Services, Education Center, South Road, Poughkeepsie, New York 12602

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

© Copyright International Business Machines Corporation 1980

Contents

Course Introduction and Orientation	1
Audience	1
Overview	1
Objectives	2
Duration	2
Outline	3
Exercises	3
Unit 1: Introduction to Data Processing	1-1
Introduction	1-1
Objective	1-1
Audience	1-1
Average Study Time	1-1
Topic 1 - Data Processing Overview	1-2
The Data Processing System	1-2
Functional Units	1-3
Central Processing Unit	1-3
Storage	1-4
Input and Output Devices	1-4
Exercise 1.1	1-7
Solution 1.1	1-9
Topic 2 - Data Representation	1-10
Symbols, a Communication Tool	1-10
Binary Concept	1-11
Positional Notation	1-13
Hexadecimal System	1-16
Eight-bit Alphameric Code (Extended Binary Coded Decimal Interchange Code-EBCDIC)	1-18
Exercise 1.2	1-21
Solution 1.2	1-22
Topic 3 - The Processor	1-23
The ALU	1-23
Registers	1-23
Control Section	1-23
Machine Cycles	1-24
Exercise 1.3	1-27
Solution 1.3	1-28
Topic 4 - Main and Auxiliary Storage	1-29
Auxiliary Storage	1-29
Main Storage	1-30
Exercise 1.4	1-33
Solution 1.4	1-34
Topic 5 - Input and Output Devices	1-35
Control Units	1-35
Channels	1-35
Types of Channels	1-36
Byte Multiplexer Channels	1-36
Selector Channels	1-36
Block Multiplexer Channels	1-37
Exercise 1.5.1	1-39
Solution 1.5.1	1-40

Direct Access Storage Devices (DASD)	1-41
Magnetic Tape Units	1-42
Tape Records, Interblock Gap and Tapemark	1-42
Load-Point and End-of-Reel Markers	1-43
Card Readers	1-43
Card Punches	1-44
Printers	1-45
Wire Matrix Printer	1-45
Chain Printer	1-45
Typewriter	1-46
3800 Printer	1-46
Visual Display Unit	1-46
Console	1-47
Exercise 1.5.2	1-49
Solution 1.5.2	1-50
Input/Output Operations	1-51
Topic 6 - Stored Program Concepts and Compilers	1-52
Introduction	1-52
Instructions	1-52
Program	1-53
Programming Languages	1-53
Source Program	1-53
Object Program	1-53
Linkage Editor	1-54
Languages	1-54
Exercise 1.6	1-55
Solution 1.6	1-56
Topic 7 - The Operating System	1-57
Introduction	1-57
Operating System	1-57
Job Management	1-57
Supervisor	1-58
Program Library	1-58
Data Management	1-58
Recovery Management	1-59
Exercise 1.7.1	1-61
Solution 1.7.1	1-62
Multiprogramming	1-63
Spooling	1-64
Summary	1-64
Exercise 1.7.2	1-65
Solution 1.7.2	1-66
Topic 8 - Data Communications	1-67
Introduction	1-67
Elements of a Data Communication Network	1-68
Host Processor	1-68
Communication Control Devices	1-68
Modulation/Demodulation Devices	1-69
Communication Lines	1-69
Configuration	1-69
Transmission Direction	1-69
Type	1-69
Transmission Mode	1-69
Codes	1-70
Line Disciplines	1-70

Terminals	1-70
Data Management Support	1-70
Data Communication Applications	1-71
Exercise 1.8	1-73
Solution 1.8	1-74
Topic 9 - Virtual Storage Concepts	1-75
Introduction	1-75
Why Virtual Storage?	1-75
IBM Virtual Storage Implementation	1-76
Summary	1-80
Exercise 1.9	1-81
Solution 1.9	1-82
Unit 2: Hardware Overview	2-1
Objectives	2-1
Study Time	2-1
Topic 1 4300 processor. Introduction to 4300 processors	2-2
Operational modes of 4300 processors	2-2
ECPS:VSE Mode	2-2
ECPS:VSE Storage Addressing	2-2
System/370 Mode	2-3
System/370 Storage Addressing	2-3
Error detection and maintenance	2-3
Error detection	2-3
Maintenance	2-3
Remote Support Facility (RSF)	2-3
Compatibility	2-3
Exercise 2.1.1	2-5
Solution 2.1.1	2-6
4331 Overview	2-7
Programming Support	2-7
Processor Storage	2-7
I/O Device Attachment	2-8
DASD Adapter	2-8
8809 Magnetic Tape Unit Adapter	2-8
Communications Adapter	2-8
5424 Adapter	2-8
Display/Printer Adapter	2-8
Channels	2-8
Exercise 2.1.2	2-9
Solution 2.1.2	2-10
4341 Overview	2-11
Programming Support	2-11
Processor Storage	2-11
I/O Device Attachment	2-12
Channels	2-12
Operator Console	2-12
Exercise 2.1.3	2-13
Solution 2.1.3	2-14
Topic 2. Input-Output Devices	2-15
Introduction	2-15
Direct Access Storage Devices	2-17
3880 Storage Facility	2-18
3830 Storage Control Unit	2-18
3340 Disk Storage Facility	2-19
3350 Direct Access Storage	2-20

3310 DASD	2-20
3370 DASD	2-21
Diskette Input/Output Device	2-22
Printing Devices	2-24
3203 Printer Model 5	2-24
3262 Line Printer	2-25
3211 Printer 3811 Printer Control Unit	2-25
3800 Printing SubSystem	2-26
Card Readers and Card Punches	2-26
3505 Card Reader	2-26
3525 Card Punch	2-27
2540 Card Read Punch	2-27
5424 Multi-function Card Unit	2-28
Magnetic Tape Units	2-29
3400 Devices	2-29
3410 Magnetic Tape Unit and 3411 Tape Unit and Tape Control	2-30
3420 Magnetic Tape Unit	2-31
3803 Tape Control	2-31
The 8809 Magnetic Tape Unit	2-32
Terminals	2-33
IBM 3270 Overview	2-33
3270 Display Units	2-33
3270 Printers	2-34
3270 Control Units	2-34
IBM 3770 Overview	2-34
3770 Fixed Function Terminals	2-35
3770 Programmable Terminals	2-35
Exercise 2.2	2-36
Solution 2.2	2-38
Unit 3: Introduction to DOS/VSE	3-1
Introduction	3-1
Objectives	3-1
Average Study Time	3-1
Topic - 1 DOS/VSE System Control Programs	3-2
The IPL Program	3-2
The Supervisor	3-3
The Job Control Program (JCP)	3-4
Exercise 3.1	3-9
Solution 3.1	3-10
Topic - 2 DOS/VSE Storage and Multiprogramming	3-11
Storage Organization	3-11
Partitions and I/O Resources	3-11
Multiprogramming	3-11
Processor Usage in Single-Partition Operation	3-12
Processor Usage in a Multiprogramming Environment	3-12
Processing Priorities	3-14
Multitasking	3-14
The Shared Virtual Area	3-15
Exercise 3.2	3-17
Solution 3.2	3-18
Topic - 3 DOS/VSE Virtual Storage Implementation	3-19
Page Data Set	3-19
Page In	3-20
Page Out	3-20
Page Pool	3-20

Real Mode	3-21
Exercise 3.3	3-23
Solution 3.3	3-23
Topic - 4 DOS/VSE Libraries and Processing Programs	3-25
Core Image Library	3-26
Relocatable Library	3-27
Source Statement Library	3-27
Procedure Library	3-27
Library Structure	3-27
Private Libraries	3-28
Processing Programs	3-29
Application Programs	3-29
Service Programs	3-29
Librarian programs	3-29
Linkage Editor Program	3-29
System Utility Programs	3-29
Data Management Routines	3-29
Language Translators	3-30
Exercise 3.4	3-31
Solution 3.4	3-32
Topic - 5 VSE/POWER	3-33
Concepts and Functions	3-33
Spooling	3-33
Remote Job Entry	3-36
Job Entry Control Language	3-36
Operator Commands	3-36
Exercise 3.5	3-37
Solution 3.5	3-38
Topic - 6 DOS/VSE Data Management	3-39
Data Organization and Access Methods	3-39
Data Organization	3-39
Access Method	3-39
Sequential Access Method (SAM) and Organization	3-40
File Organization	3-40
Data Access	3-40
Indexed Sequential Access Method (ISAM) and Organization	3-40
Index and File Organization	3-40
Direct Access Method (DAM) and Organization	3-41
VSE/Virtual Storage Access Method (VSE/VSAM)	3-41
File Organization	3-41
VSAM Catalogs	3-42
VSE/VSAM Space Management for SAM Feature	3-42
VSE/VSAM Device Independence	3-42
ISAM Interface Program (IIP)	3-42
Data Security/Data Integrity	3-42
VSE/VSAM Summary	3-43
Telecommunication Access Methods	3-43
Protection of Data	3-43
File labeling	3-43
Protection Against Duplicate Assignments	3-44
File Security	3-44
Track Hold Specification	3-45
Exercise 3.6	3-47
Solution 3.6	3-48
Topic 7 Unit Summary	3-49

Unit 4: ICCF Facilities	4-1
Introduction	4-1
Objective	4-1
Average Study Time	4-1
Topic - 1. Interactive Computing	4-2
Program Development With Batch Facilities	4-2
Interactive Program Development	4-3
VSE/ICCF	4-5
Exercise 4.1	4-7
Solution 4.1	4-8
Topic - 2. Organization of ICCF	4-9
ICCF Control Program	4-9
ICCF Command Processors	4-9
Terminal Control Facility	4-9
ICCF Interactive Partitions	4-10
Exercise 4.2	4-11
Solution 4.2	4-12
Topic - 3. End User's View of ICCF	4-13
ICCF Submit Facility	4-14
User Interface	4-14
Exercise 4.3	4-15
Solution 4.3	4-16
Topic - 4. The ICCF Library File	4-17
System Record	4-17
User Profile	4-17
Spool Areas	4-17
Libraries	4-17
Library Categories	4-17
Common Library	4-18
Private Library	4-18
Shared Library	4-18
Library Usage	4-18
Exercise 4.4	4-21
Solution 4.4	4-22
Topic - 5. Modes of Operation	4-23
Command Mode	4-23
Input Mode	4-23
Edit Mode	4-24
Update Mode	4-24
Execution Mode	4-24
Asynchronous Execution	4-24
Dynamic Space Allocation	4-24
List Mode	4-24
Exercise 4.5	4-25
Solution 4.5	4-26
Topic - 6. The Editors	4-27
Context Editor	4-27
Full Screen Editor	4-27
Split Screen Facility	4-28
Exercise 4.6	4-29
Solution 4.6	4-30
Topic - 7. The Command Language	4-31
System Commands	4-31
Context Editor Commands	4-31
Full Screen Editor Commands	4-31

Job Entry Statements	4-31
Dump Commands	4-31
Procedural and Macro Commands	4-32
Exercise 4.7	4-33
Solution 4.7	4-34
Topic - 8. Procedures and Macros	4-35
Topic - 9. Debugging Assistance	4-36
ICCF Dump Program	4-36
Interactive Debug	4-36
Exercise 4.8 and 4.9	4-37
Solution 4.8 and 4.9	4-38
Topic - 10. Data Security	4-39
Exercise 4.10	4-41
Solution 4.10	4-42
Topic - 11. Interactive Usability Aids	4-43
Exercise 4.11	4-45
Solution 4.11	4-46
Topic - 12. Summary	4-47
Unit 5: Data Systems Environment	5-1
Introduction	5-1
Objectives	5-1
Average Study Time	5-1
Topic 1: Background	5-2
Value of Data	5-2
Traditional Approach	5-3
Data Base Approach	5-8
Online Processing	5-9
Data Systems Environment	5-10
Exercise 5.1	5-13
Solution 5.1	5-14
Topic 2: Data Base	5-15
Data Base Manager	5-15
Data Language/I	5-16
Exercise 5.2	5-17
Solution 5.2	5-18
Topic 3: Data Communications	5-19
Data Communications Manager	5-19
CICS/DOS/VS	5-21
Exercise 5.3	5-23
Solution 5.3	5-24
Topic 4: Data Administration	5-25
Data Dictionary	5-28
Data Base Administration	5-30
Summary	5-31
Exercise 5.4	5-33
Solution 5.4	5-34
Topic 5: Data Delivery	5-35
Prewritten Application Programs	5-36
Online Interactive Tools	5-36
Application Development Tools	5-36
DMS/CICS/VS	5-36
Summary	5-37
Exercise 5.5	5-39
Solution 5.5	5-40
Topic 6: Summary	5-41

Unit 6: Data Systems Products	6-1
Introduction	6-1
Objectives	6-1
Average Study Time	6-1
Topic 1. Data Language/I DOS/VS (DL/I DOS/VS)	6-2
Data Base Structure and Organization	6-2
Logical Data Structure	6-2
Elements of a DL/I Hierarchical Structure	6-4
Physical Data Structure	6-6
Hierarchical Sequential	6-7
Hierarchical Direct	6-7
Data Base Definition	6-8
DBD (Data Base Description)	6-8
PSB (Program Specification Block)	6-9
Using The DBD and PSB	6-9
Data Base Program Execution	6-9
Additional DL/I Capabilities	6-10
Multiple Partition Support (MPS)	6-10
Program Isolation	6-10
Field Level Sensitivity	6-11
Secondary Indexing	6-11
Logical Relationships	6-11
Logging Facility	6-14
Utility Support Programs	6-15
Data Base Recovery System Utilities	6-15
Summary	6-15
Exercise 6.1	6-17
Solution 6.1	6-18
Topic 2. CICS/DOS/VS	6-19
Basic Functions	6-20
CICS/DOS/VS Management Functions	6-20
Program Interface	6-22
CICS/DOS/VS Processing Flow	6-23
Additional Functions	6-25
Sign-on	6-25
Master Terminal	6-25
System Statistics	6-25
Execution Diagnostic Facility	6-25
Program and Data Error Recovery	6-26
Summary	6-26
Exercise 6.2	6-27
Solution 6.2	6-28
Topic 3. Development Management System/CICS/VS	6-29
Organization and Basic Functions	6-29
Application Generation	6-30
DMS Transactions	6-32
Additional Functions	6-35
User Exits	6-36
Application Security	6-36
Summary	6-36
Exercise 6.3	6-37
Solution 6.3	6-38
Topic 4. DB/DC Data Dictionary	6-39
Organization and General Functions	6-40
Data Storage Techniques	6-41

Information Retrieval Techniques	6-41
Additional Optional Functions	6-42
SCAN Report	6-42
Security Facility	6-42
Extensibility Feature	6-43
Program Access Facility	6-43
Summary	6-43
Exercise 6.4	6-45
Solution 6.4	6-46
Unit 7: DOS/VSE System IPO/E	7-1
Introduction	7-1
Objectives	7-1
Average Study Time	7-1
Topic 1. DOS/VSE System IPO/E Structure and Content	7-2
DOS/VSE System IPO/E - Batch Interactive Contents	7-3
DOS/VSE System IPO/E - DC Contents	7-3
DOS/VSE System IPO/E - DB/DC Contents	7-3
Exercise 7.1	7-5
Solution 7.1	7-6
Topic 2. Overview of the Interactive Productivity Facility	7-7
Communicating With The Interactive Productivity Facility	7-7
Hierarchical Panel Structure	7-7
System Management	7-8
System Use	7-8
First Use Tutorial	7-9
Interactive Productivity Facility Display Panels	7-9
Sign-On Panel	7-9
Panel Identification	7-10
Instruction Line	7-11
Descriptive Text	7-12
Menu Selection List	7-13
User Input Area	7-14
Miscellaneous Commands	7-15
Data Entry Panels	7-16
Panel Format	7-16
Data Entry Area	7-17
Command Line	7-19
Explain Panels	7-20
Panel Format	7-20
Explain Text	7-21
Command Line	7-22
Exercise 7.2	7-25
Solutions 7.2	7-26
Topic 3. Communicating With The Interactive Productivity Facility	7-27
Exercise 7.3	7-45
Solution 7.3	7-46
Unit 8: The Data Processing Installations	8-1
Introduction	8-1
Objectives	8-1
Average Study Time	8-1
Topic 1. Perspective on Hardware	8-2

Course Introduction and Orientation

Audience This course is designed for new data processing personnel who require a general understanding of the hardware and software components of an IBM 4300 DOS/VSE data processing system.

Overview This self-study text describes the IBM 4300 hardware and the functions and features of DOS/VSE, VSE/POWER, and VSE/ICCF. The student is also introduced to the data systems environment and the facilities of CICS/VS, DL/I DOS/VS, DB/DC Data Dictionary and DMS/CICS/VS. In addition, the facilities provided by DOS/VSE System IPO/E are described.

Objectives At the completion of this course, you should be able to:

- Understand the basic terminology used to describe the hardware and software components of a data processing system.
- Understand the basic hardware functions provided by processors, channels and I/O devices.
- Describe the 4300 Processors.
- Understand the requirements for an operating system.
- Describe the facilities provided by DOS/VSE including the system control programs, multiprogramming, storage management, data management and library support.
- Understand the facilities provided by VSE/POWER and describe the spooling concept.
- Describe how ICCF supports interactive users.
- Describe the data systems environment.
- Describe the basic facilities provided by the IBM programs that support the data systems environment. These products include CICS/VS, DL/I DOS/VS, DB/DC Data Dictionary, and the Development Management System/CICS/VS.
- Understand the facilities provided by DOS/VSE System IPO/E.

Duration Self-Study 24-30 hours.

Outline**Unit**

1. Introduction to Data Processing
2. Hardware Overview
3. Introduction to DOS/VSE
4. ICCF Facilities
5. Data Systems Environment
6. Data Systems Products
7. DOS/VSE System IPO/E
8. The Data Processing Installation

Exercises

At various points throughout the course, you will be asked to do an exercise. These exercises have been inserted throughout the course to test your understanding of the material and point out material that should be reviewed for better understanding. Therefore, it is in your own best interest to take each exercise as it occurs, and not refer to the solution until you have completed the exercise.

The solutions to the exercises are located on the pages following the exercises.



Unit 1:

Introduction to Data Processing

Introduction This unit describes the major components of a data processing system. It is intended to familiarize the student with the functions of these components and to provide an understanding of the basic data processing terminology used to describe them.

Objective Upon completion of this unit, the student should be able to:

- Describe data representation in a data processing system.
- Identify the functions of the central processing unit.
- Understand the functions of auxiliary and main storage.
- Identify the various types of input and output devices and describe their functions.
- Describe the functions of programs and program languages.
- Identify and describe the major functions provided by system control programs.
- Describe the basic components of a communications network.
- Understand the basic functions of virtual storage.

Audience No prior knowledge of data processing is assumed. This unit is intended for those persons who wish to gain a basic understanding of data processing and the terminology used in this field.

If you think you do not need to study the entire unit you may take the review test at the end of each topic. Examining any questions you did not answer correctly will provide a guide to which subjects you should review.

Average Study Time 3 to 4 Hours

Topic 1 - Data Processing Overview

This topic is an overview of the material that will be covered in more detail in the topics which follow.

Read through the text provided for the topic. There may be some terms you do not understand initially, but their meaning should be clear when you have completed the text.

When you have finished reading the material, answer the review questions and review the text for any question you are unable to answer correctly. (Answers are provided on the page following the questions).

The Data Processing System

Data processing is a series of planned actions and operations upon information to achieve a desired result. The procedures and devices used constitute a data processing system (Figure 1.1.1). The devices may vary; all operations may be done by machine, or the devices may be only pencil and paper. The procedures, however, remain the same.

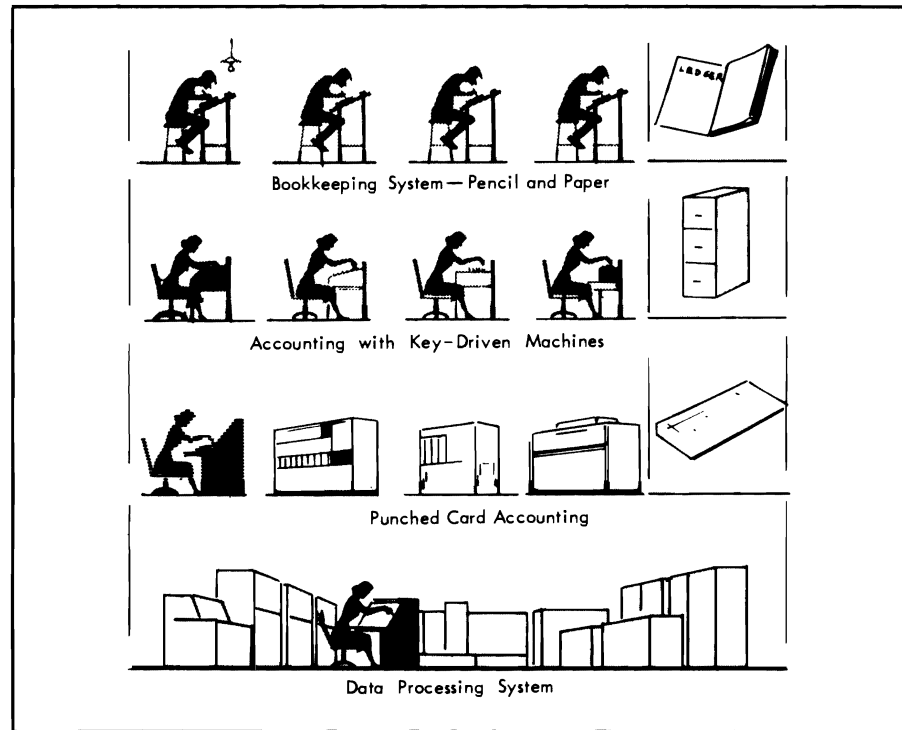


Figure 1.1.1 Data Processing Systems

The many types of IBM data processing systems vary in size, complexity, speed, cost levels of programming systems, and application. But, regardless of the information to be processed or the equipment used, all data processing involves at least three basic considerations:

1. The source data or INPUT entering the system.
2. The orderly, planned PROCESSING of the source data within the system.
3. The end result or OUTPUT from the system.

INPUT may consist of any type of data: commercial, scientific, statistical, engineering, and so on.

PROCESSING is carried out in a predetermined sequence of instructions followed automatically by the computer (Figure 1.1.2). The instructions are of human origin written in a form which can be read into the computer through an input device.

By calculation, sorting, analysis, or other operations, the computer arrives at the end results used for further processing or OUTPUT such as reports or sets of data. The DP system hardware, such as input/output (I/O) devices and the central processing unit (CPU), provide the environment for processing the data.

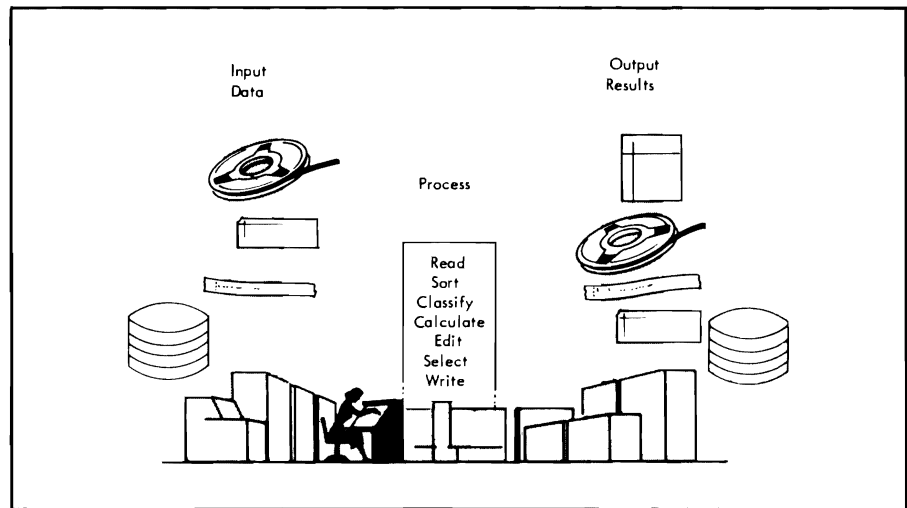


Figure 1.1.2 Data Processing by Computer

Each data processing system is designed to perform a specific number and type of operations. It is directed to perform each operation by an INSTRUCTION. The instruction defines a basic operation to be performed and identifies the data, device, or mechanism needed to carry out the operation. The entire series of instructions required to complete a given job is known as a PROGRAM. The term SOFTWARE refers to the programs used in a DP system.

Functional Units

Data processing systems can be divided into FOUR types of FUNCTIONAL UNITS: central processing unit, storage, input devices, and output devices. These units are physical devices and are often referred to as hardware.

Central Processing Unit

The central processing unit (Figure 1.1.3) is the controlling center of the entire data processing system. Some of the key functions performed by the CPU include the execution of instructions and machine service functions.

The processor contains processor storage, control storage, the system control panel, and a set of instructions to perform arithmetic functions and logical processing of data.

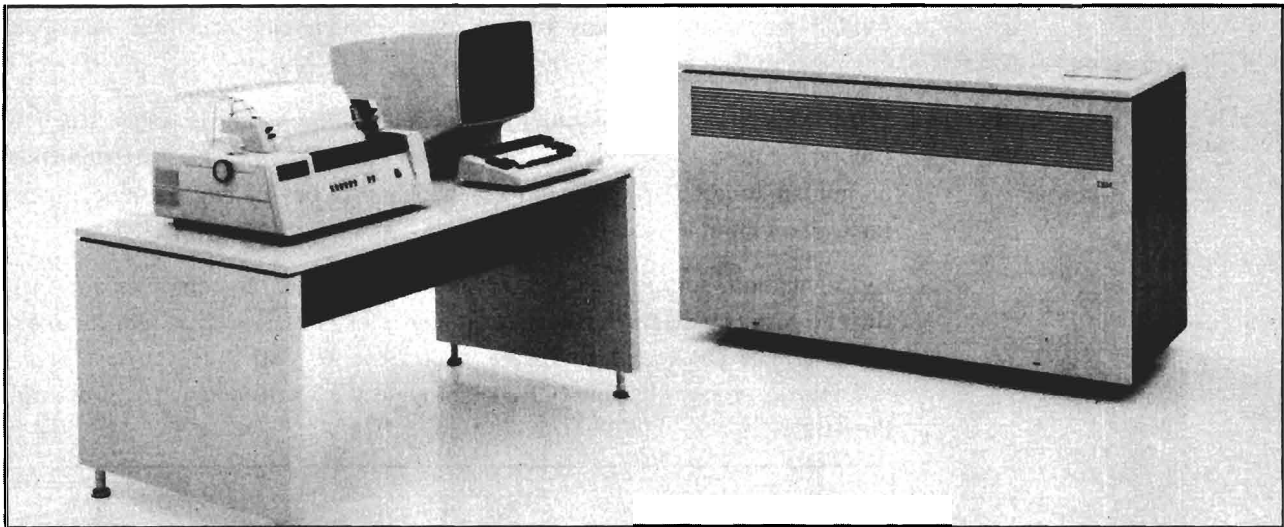


Figure 1.1.3 4331 Processor With Display Console and Keyboard

The Support Processor provides the machine servicing functions. These functions are necessary for monitoring and maintenance, and problem diagnosis and repair of the CPU.

Storage

Storage is like an electronic filing cabinet, completely indexed and accessible to the computer.

All data must be placed in storage before it can be processed by the computer. Information is read into storage by an input unit and is then available for internal processing. Each position or section of storage has a specific location called an address, so that the stored data can be readily located by the computer when needed.

The computer may rearrange data in storage by sorting or combining different types of information received from a number of input units. The computer may also take the original data from storage, calculate new information, and place the result back in storage.

The size or capacity of storage determines the amount of information that can be held within the system at any one time. In some computers, storage capacity is measured in millions of characters (bytes), providing space to retain entire files of information. In other systems, storage is smaller, and data is held only while being processed.

Additional storage called SECONDARY or AUXILIARY storage is provided by direct access storage devices (DASD) which are described later in this unit.

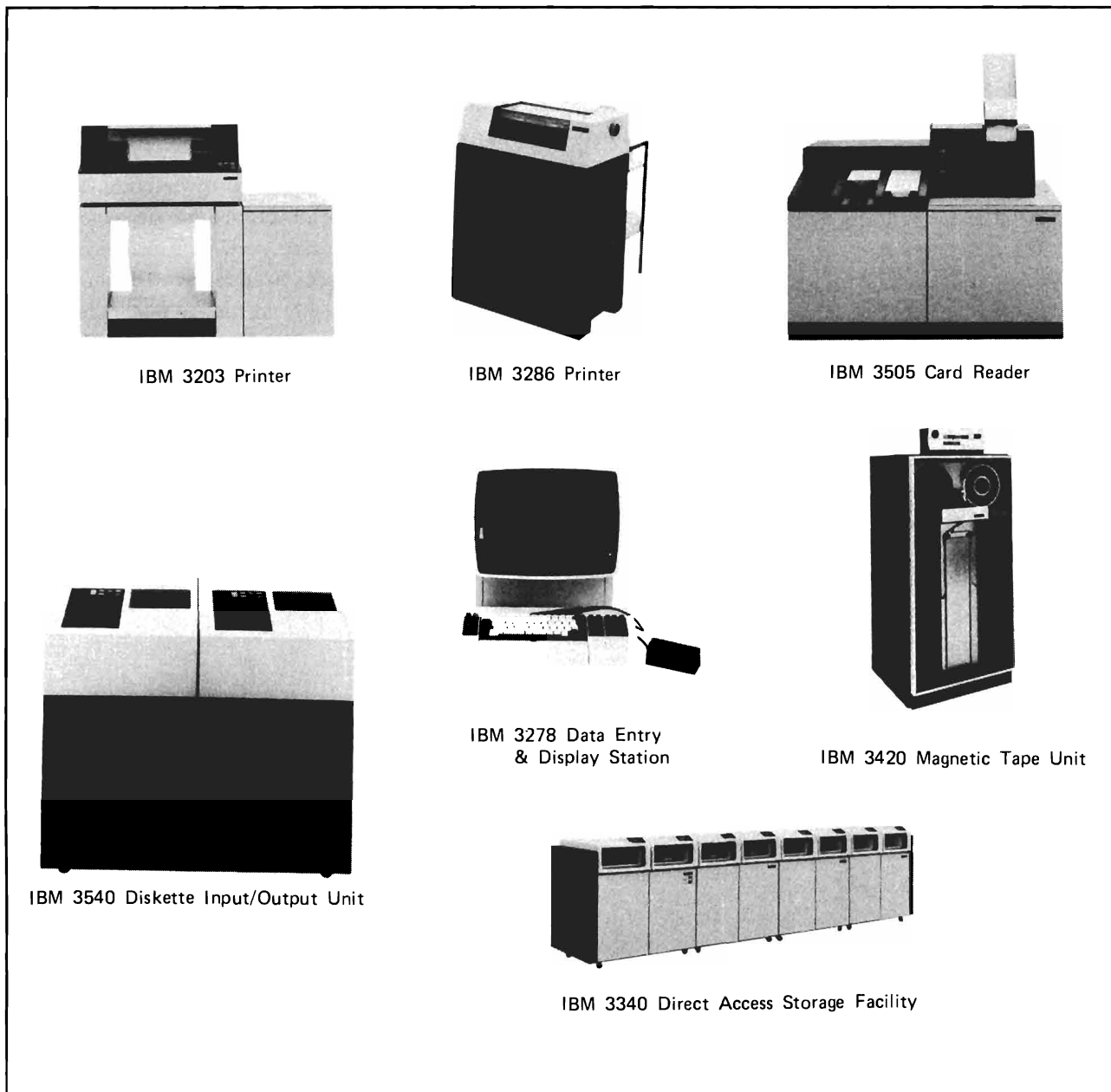
Input and Output Devices

The data processing system requires as a necessary part of its information-handling ability, the capability to enter data into the system and record data from the system. These functions are performed by input/output devices linked directly to the system.

Input devices read or sense coded data that is recorded on a prescribed medium and make this information available to the computer. Data for input is recorded in cards and paper tape as punched holes, and on magnetic tape and disk as magnetized spots. Data can also be entered from keyboard devices resembling typewriter keyboards.

Output devices record or write information from the computer on cards, paper tape, magnetic tape, and disks; they print information on paper; generate signals for transmission over teleprocessing networks; produce graphic displays, microfilm images, and take other specialized forms. The number and type of input/output devices connected directly to the computer depend on the design of the system and its applications.

Figure 1.1.4 shows several input and output devices used in IBM data processing systems.



IBM 3203 Printer

IBM 3286 Printer

IBM 3505 Card Reader

IBM 3278 Data Entry
& Display Station

IBM 3420 Magnetic Tape Unit

IBM 3540 Diskette Input/Output Unit

IBM 3340 Direct Access Storage Facility

Figure 1.1.4 Input/Output Devices

Exercise 1.1

1. All data processing involves at least three basic activities; they are:

- a. performing complex calculations
 - b. input entering the system
 - c. planning a series of actions
 - d. processing data in the system
 - e. outputting data from the system
 - f. writing specific operations
2. A data processing system is directed to perform an operation by a/an _____, a series of these, necessary to accomplish a given task, is called a/an _____.
- a. program
 - b. operation
 - c. procedure
 - d. instruction
 - e. definition
3. Functionally data processing systems consist of these four units:

- a. central processing unit
- b. circuitry
- c. storage
- d. instructions
- e. input devices
- f. output devices

4. In IBM systems the central processing unit has two primary parts: the _____, which contains control areas, and facilities for performing arithmetic and logical processes; and the _____, which provides monitoring, maintenance, and problem diagnosis and repair functions.
 - a. instruction processor
 - b. channel circuitry
 - c. support processor
 - d. I/O devices
5. The instructions to complete a task are executed by the processor.
True or False
6. Information is first read into computer _____ by an input unit before it can be processed and the results removed from _____ via an output unit.

Now check your answers against the answers provided on the following page.

Solution 1.1

1. B, D, E
2. D, A
3. A, C, E, F
4. A, C
5. True
6. Storage, Storage

Below is a reference indicating which section in this topic relate to specific questions in the quiz.

Question #	Topic 1 - Section Heading
1,2	The Data Processing System
3	Functional Units
4,5,6	Central Processing Unit
7	Storage

Topic 2 - Data Representation

Symbols, a Communication Tool

This topic provides the student with a general understanding of the numbering systems used in data processing and the manner in which data is maintained.

Symbols convey information. The symbol itself is not the information but merely represents it. The printed characters on this page are symbols that convey one meaning to some persons a different meaning to others, and no meaning to those who do not know their significance (Figure 1.2.1).

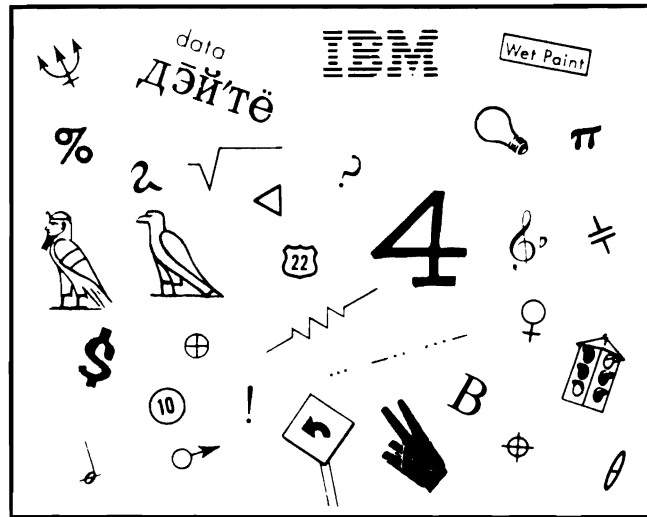


Figure 1.2.1 Symbols for Communication

Presenting data to the computer system is similar in many ways to communicating with another person by letter. The intelligence to be conveyed must be reduced to a set of symbols. In the English language, these are the familiar letters of the alphabet, numbers and punctuation. The symbols are recorded on paper in a prescribed sequence and transported to another person who reads and interprets them.

Similarly, communication with the computer system requires that data be reduced to a set of symbols that can be read and interpreted by data processing machines. The symbols differ from those commonly used by people, because the information to be represented must conform to the design and operation of the machine. The choice of these symbols (and their meaning) is a matter of convention on the part of the designers. The important fact is that information can be represented by symbols which become a language for the communication between people and machines.

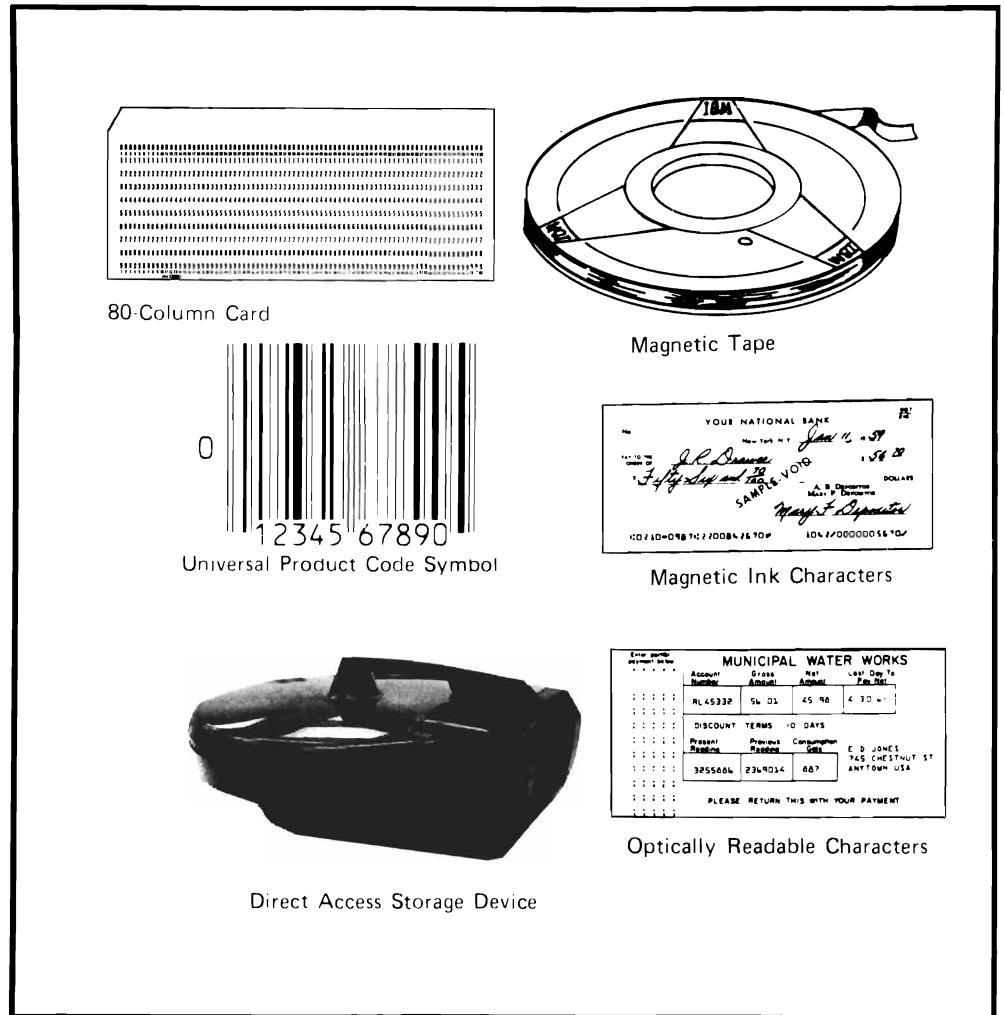


Figure 1.2.2 Data Recording Media

In the computer, data is represented by many electronic components, as shown in Figure 1.2.2. The storage and flow of data through these components are represented as electronic signals. The presence or absence of these signals in specific circuitry is the method of representing data much as the presence of holes in a card represents data.

Binary Concept

The numeric symbols we are most familiar with are those in the decimal system. However, the decimal system may not be the one best suited to the task of designing the most efficient computing machines.

In data processing we make use of other numbering systems using greater or fewer symbols than the ten used by the decimal system. For example, the hexadecimal system uses 16 symbols, 0-9, just like decimal does, plus the alphabetic characters A-F. The binary number system uses only 2 symbols, 0 and 1, to convey its values.

The binary system is more convenient to use with computers, where the information handled is in binary form - switches that are on or off, punches that are present or absent, and the like.

Computers function in binary states. This means that the computer components can indicate only two states or conditions. For example, the ordinary light bulb operates in a binary mode, that is, it is either on or off. Likewise, within the computer, transistors are either conducting or nonconducting, magnetic materials are magnetized in one direction or in an opposite direction and specific voltage potentials are present or absent (Figure 1.2.3). The binary states of operation of the components are signals to the computer, as the presence or absence of light from an electric light bulb can be a signal to a person.

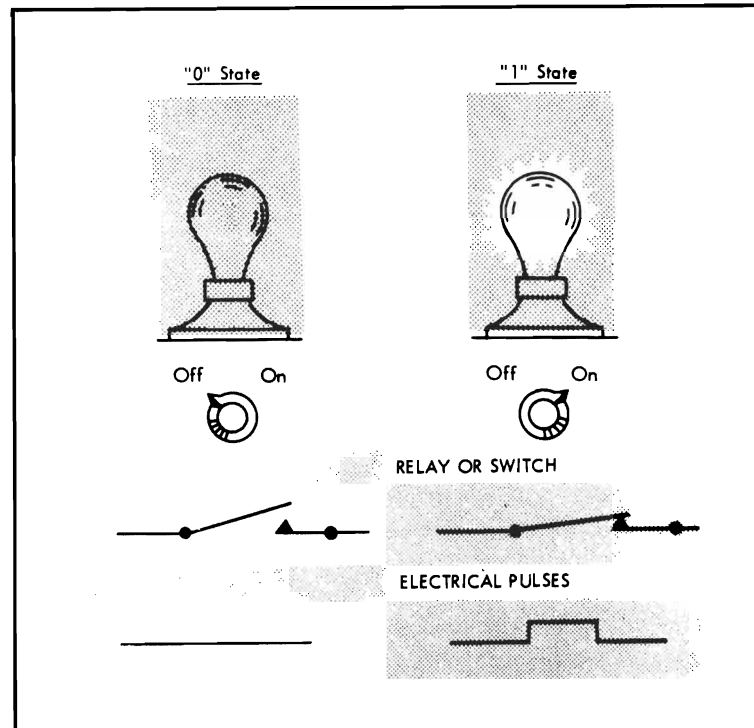


Figure 1.2.3 Binary Components

For example, suppose that a friend's house light is always on when he is home and always off when he is away. How many messages does the light being on or off communicate to you? Two.

Now suppose you have two lights, one on the left and one on the right. Both can be on, both can be off, or one can be on and the other off. If you want to send a message by using all the patterns of the two lights, how many messages can you send? _____

Four (both on, both off, left on and right off, left off and right on).

Now suppose that instead of having lights on and off, we think of representing numbers with either of two binary digits: 0 and 1. Let's figure out how many messages we could send with three binary digits, or bits (this is where the word BIT comes from). Here is an easy way to think of it. You have three black boxes of different sizes. The box on the left is big enough to hold 4 marbles. The box in the middle will hold 2 marbles. The box on the right will hold 1 marble. You have two signs to put on top of each box, a 0 and a 1. To show that a box is full of marbles, you put a 1 on it. To show that a

box is empty, you put a 0 on it. If the numbers from left to right read 101, you know there are 4 marbles + 0 marbles + 1 marble, or a total of 5 marbles. See Figure 1.2.4.

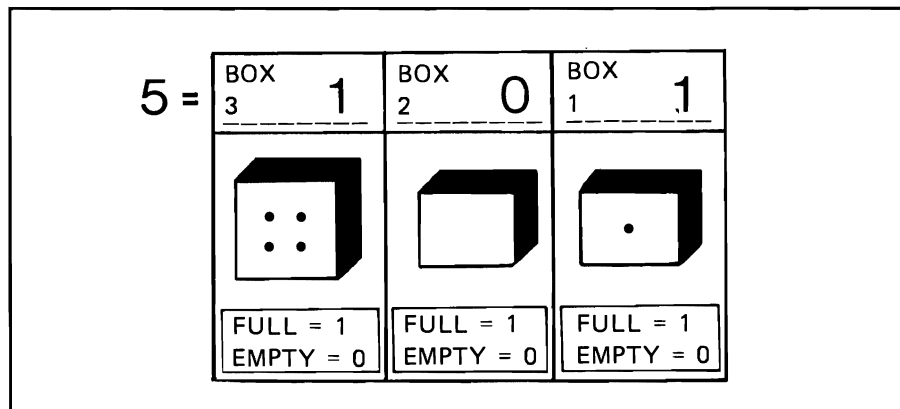


Figure 1.2.4 Binary Representation of 101

Let's read from right to left. The right most box has space for 1 marble. The next box on the left has room for 2 marbles. The next box on the left has room for 4 marbles. The pattern is that for each box we add to the left we double the number of marbles. If we add a fourth box on the left, it can hold 8 marbles. See Figure 1.2.5.

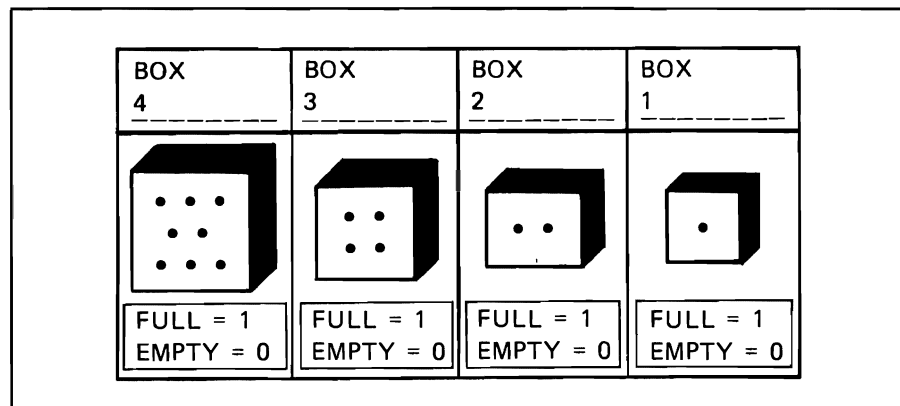


Figure 1.2.5 Binary Representation Showing 1111

If the signs on top of the boxes show 1000, how many marbles are there? Box 4 is full and all the other boxes are empty so if you said 8 you are right. In any one position the 0 represents the absence of a related or assigned value and the 1 represents the presence of a related or assigned value.

Positional Notation

A number is basically a string of symbols. The decimal system uses the symbols 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9. Each of these ten symbols has a fixed value one higher than that of the symbol before it in the progression from smallest to largest. The name commonly used with these symbols is "digit".

When several digits are combined into a number, its value depends upon not only the value of the digits, but also upon the relative position of each digit. This principle of numeric position is called positional notation. In a system

that uses positional notation, the digit position on the extreme right is the one of least value (or lowest order) and is called the "least significant digit". The digit on the extreme left is the one with the highest value and is called the "most significant digit". This is shown in Figure 1.2.6.

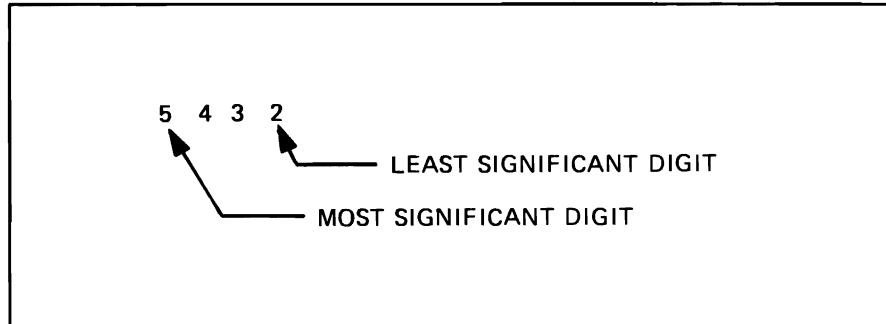


Figure 1.2.6 Digit Names

The value of each digit position increases from right to left (least significant to most significant). How much it increases depends upon the base, or radix, of the number system being used. The decimal system has a base of 10 since it has 10 different symbols, and thus each digit position is increased by a power of 10. For example, the value of the number in Figure 1.2.6. is immediately apparent to you. However, the notation 5432 actually signifies 5 thousands, plus 4 hundreds, plus 3 tens, plus 2 units. See Figure 1.2.7.

$$5432 = \left\{ \begin{array}{r} 5000 \\ 400 \\ 30 \\ + \underline{2} \\ \hline 5432 \end{array} \right.$$

Figure 1.2.7 Decimal Positional Meaning

The binary system also groups its numbers within units to represent a value. For 4300 systems, the smallest addressable unit of data is a BYTE. A byte is a grouping of 8 bits. The bits within the byte have a place or positional significance related to the binary number system. That is, the place position of a bit within the byte determines the value of the bit. In the binary number system, the decimal values of the places (from right to left) are 1, 2, 4, 8, 16, 32, 64 and so on as shown in figure 1.2.8.

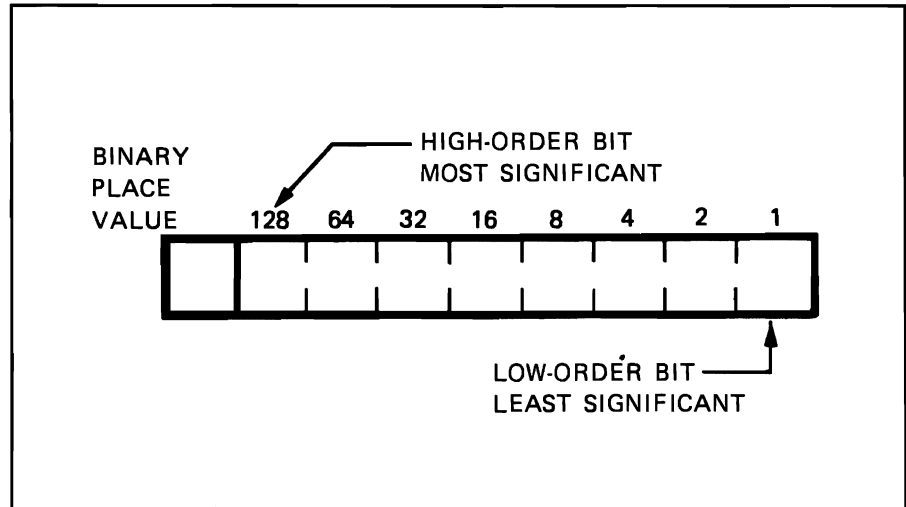


Figure 1.2.8 Place Significance in Binary Numbering

Four bytes constitute a "word" consisting of 32 consecutive bit positions of information which are interpreted as a unit, much as a character or a digit in other systems.

Although the place values of the bits of a word are always those of the binary number system, they can be interpreted or processed in such a way as to represent other than a binary number. For example, a 32 bit word (Figure 1.2.9) can be interpreted as one 32 place binary number, as an eight-digit hexadecimal number, as four alphameric characters (which may be alphabetic or numeric), or as any predetermined representation established by the programmer.

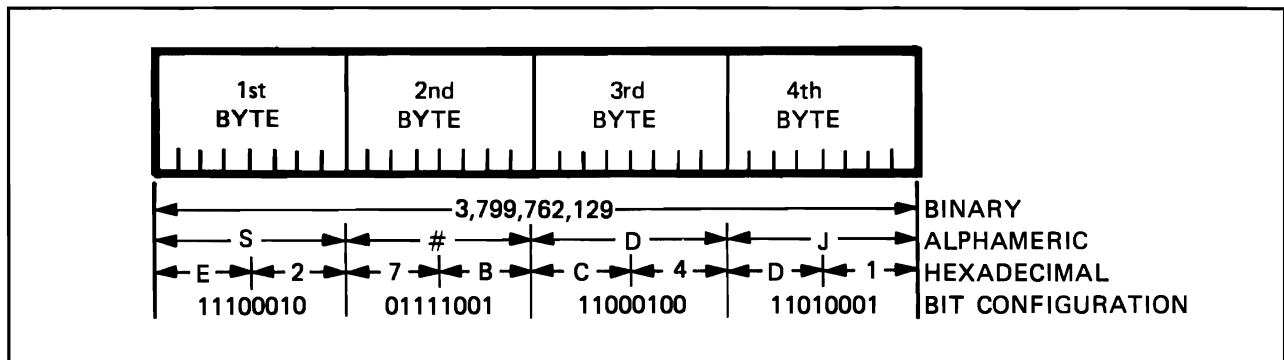


Figure 1.2.9 The 32-bit Word

Counting in binary involves setting the correct bit positions (remember, bit = binary digit) to cause the place values to add up to the decimal count desired. Figure 1.2.10 shows the decimal numbers 0 to 15 and their binary equivalents.

DECIMAL	BINARY
	<u>8421</u> ← place value
0	0000
1	0001
2	0010
3	0011 ← place value 1 + 2 = 3 in decimal.
4	0100
5	0101 ← place value 1 + 4 = 5 in decimal.
6	0110
7	0111 ← place value 1 + 2 + 4 = 7 in decimal.
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

Figure 1.2.10 Decimal and Binary Equivalents

It follows, then, that if you know the place values of binary bits, you can decode the decimal value of the binary number simply by adding the place values of the bits that are set to one.

It is apparent that binary numbers require several times as many positions as decimal numbers to display the equivalent number. In talking and writing, these binary numbers are bulky. A long string of ones and zeros cannot be effectively transmitted from one individual to another. Some shorthand method is necessary. The hexadecimal number system fills this need. Because of their simple relationship to binary, numbers can be converted from one system to another by inspection.

Hexadecimal System

Although all IBM computers use a binary system, the basic man-machine communication is in the hexadecimal numbering system. For example, programs usually list the contents of storage in hexadecimal notation.

Hexadecimal means 16. The hexadecimal system uses binary bits to count up to 16, carry, and then start counting again. It does not, however, count to 16 in numbers. It counts from 0 through 9 in numbers; then 10 is A, 11 is B, 12 is C 13 is D, 14 is E, and 15 is F, in hexadecimal representation. It takes four binary bit positions to count to F (15) in a computer. Figure 1.2.11 shows how it is done.

<u>DECIMAL SYSTEM</u>	<u>HEXADECIMAL SYSTEM</u>	<u>BINARY SYSTEM</u> 8 4 2 1 Bit values
0	0	0 0 0 0
1	1	0 0 0 1
2	2	0 0 1 0
3	3	0 0 1 1
4	4	0 1 0 0
5	5	0 1 0 1
6	6	0 1 1 0
7	7	0 1 1 1
8	8	1 0 0 0
9	9	1 0 0 1
10	A	1 0 1 0
11	B	1 0 1 1
12	C	1 1 0 0
13	D	1 1 0 1
14	E	1 1 1 0
15	F	1 1 1 1

Figure 1.2.11 Relationship Among Decimal, Hexadecimal, and Binary Systems

Since IBM systems have eight bits in each byte of storage, each byte can be thought of as being two hexadecimal-system digits; for example:

Decimal	8	
Binary	1111	1000
Hexadecimal	F	8

Note the hexadecimal F and the binary 1111, the left most characters. They represent the sign of the digit on the right, in this case a positive sign. (1100 or D, would be a negative sign)

Remember that the hexadecimal system is simply a shorthand notation used to express the binary bit patterns within a computer. In extended binary coded decimal interchange code (EBCDIC), the eight-bit character (called F8 in hexadecimal) would be a positive 8. It could have other meanings in other codes, but, regardless of the code meaning, it is expressed as F8 in the hexadecimal system.

To convert binary integers to hexadecimal, simply mark the number into groups of four bits, starting from the right, and then replace each group with the corresponding hexadecimal digit. If the left-hand group is incomplete, fill in zeros on the left as required. Figure 1.2.12 gives an example of this procedure.

111110011011010011 = 0011/1110/0110/1101/0011
= 3 E 6 D 3

Figure 1.2.12 Binary/Hexadecimal Conversion

***Eight-bit Alphameric Code
(Extended Binary Coded
Decimal Interchange Code
- EBCDIC)***

This code (Figure 1.2.13) uses eight binary positions for each character format. By using eight bit positions, 256 different characters can be coded. This code permits, for instance, the coding of uppercase and lowercase alphabetic characters, a wide range of special characters, and many control characters that are meaningful to certain input/output devices. At present, many bit patterns have no assigned function (control or graphic). They are reserved for future assignment. This is the primary code used in IBM data processing systems.

Bit Configuration		Bit Configuration		Bit Configuration	
EBCDIC	Configuration	EBCDIC	Configuration	EBCDIC	Configuration
NUL	0000 0000		0100 0101		1000 1010
SOH	0000 0001		0100 0110		1000 1011
STX	0000 0010		0100 0111		1000 1100
ETX	0000 0011		0100 1000		1000 1101
PF	0000 0100		0100 1001		1000 1110
HT	0000 0101	⌘	0100 1010		1000 1111
LC	0000 0110		0100 1011		1001 0000
DEL	0000 0111	<	0100 1100	j	1001 0001
	0000 1000	(0100 1101	k	1001 0010
RLF	0000 1001	+	0100 1110	l	1001 0011
SMM	0000 1010		0100 1111	m	1001 0100
VT	0000 1011	&	0101 0000	n	1001 0101
FF	0000 1100		0101 0001	o	1001 0110
CR	0000 1101		0101 0010	p	1001 0111
SO	0000 1110		0101 0011	q	1001 1000
SI	0000 1111		0101 0100	r	1001 1001
DLE	0001 0000		0101 0101		1001 1010
DC1	0001 0001		0101 0110		1001 1011
DC2	0001 0010		0101 0111		1001 1100
TM	0001 0011		0101 1000		1001 1101
RES	0001 0100		0101 1001		1001 1110
NL	0001 0101	!	0101 1010		1001 1111
BS	0001 0110	\$	0101 1011		1010 0000
IL	0001 0111	*	0101 1100		1010 0001
CAN	0001 1000)	0101 1101	s	1010 0010
EM	0001 1001	:	0101 1110	t	1010 0011
CC	0001 1010]	0101 1111	u	1010 0100
CU1	0001 1011	-	0110 0000	v	1010 0101
IFS	0001 1100	/	0110 0001	w	1010 0110
IGS	0001 1101		0110 0010	x	1010 0111
IRS	0001 1110		0110 0011	y	1010 1000
IUS	0001 1111		0110 0100	z	1010 1001
DS	0010 0000		0110 0101		1010 1010
SOS	0010 0001		0110 0110		1010 1011
FS	0010 0010		0110 0111		1010 1100
	0010 0011		0110 1000		1010 1101
BYP	0010 0100		0110 1001		1010 1110
LF	0010 0101	7/12	0110 1010		1010 1111
ETB	0010 0110		0110 1011		1011 0000
ESC	0010 0111	%	0110 1100		1011 0001
	0010 1000		0110 1101		1011 0010
	0010 1001	>	0110 1110		1011 0011
SM	0010 1010	?	0110 1111		1011 0100
CU2	0010 1011		0111 0000		1011 0101
	0010 1100		0111 0001		1011 0110
ENQ	0010 1101		0111 0010		1011 0111
ACK	0010 1110		0111 0011		1011 1000
BEL	0010 1111		0111 0100		1011 1001
	0011 0000		0111 0101		1011 1010
	0011 0001		0111 0110		1011 1011
SYN	0011 0010		0111 0111		1011 1100
	0011 0011		0111 1000		1011 1101
PN	0011 0100	6/0	0111 1001		1011 1110
RS	0011 0101	:	0111 1010		1011 1111
UC	0011 0110	#	0111 1011	PZ 7/11	1100 0000
EOT	0011 0111	@	0111 1100	A	1100 0001
.	0011 1000	'	0111 1101	B	1100 0010
	0011 1001	=	0111 1110	C	1100 0011
	0011 1010	:	0111 1111	D	1100 0100
CU3	0011 1011		1000 0000	E	1100 0101
DC4	0011 1100	a	1000 0001	F	1100 0110
NAK	0011 1101	b	1000 0010	G	1100 0111
	0011 1110	c	1000 0011	H	1100 1000
SUB	0011 1111	d	1000 0100	I	1100 1001
SP	0100 0000	e	1000 0101		1100 1010
	0100 0001	f	1000 0110		1100 1011
	0100 0010	g	1000 0111	⌋	1100 1100
	0100 0011	h	1000 1000		1100 1101
	0100 0100	i	1000 1001	⌋	1100 1110

Figure 1.2.13 Examples of EBCDIC Code

Now you know that data processing systems maintain information in a binary form internally and the hexadecimal system is used in IBM computers for translating internal data to printed information. We looked at the code structure used by IBM data processing systems, EBCDIC and saw that this code, by using 8 binary digits, provides 256 different characters representations.

The quiz on the following page provides a review of your understanding of the primary concepts covered in this section. Take the quiz now and review your answers using the answers given on the page following the quiz.

Exercise 1.2

1. Three numbering systems described in this topic are _____ with which you are most familiar, _____ which uses 16 symbols, and the _____ which uses only 2 symbols.
2. The components in a computer can indicate only two states or conditions and for this reason we say it functions in a _____ state.
3. Positional notation means that when digits are combined into a number, the value of the number depends not only on the value of the digits but also on the relative _____, of each digit.
4. In a system which uses positional notation, the digit position on the extreme left is called the _____ digit while the digit on the extreme right is called the _____ digit.
5. The smallest addressable unit of data in 4300 computers is the _____.
6. A _____ is a grouping of 8 bits, or binary digits.
7. Knowing the place value of the binary digits (Shown in Figure 1.2.8) will help you determine the decimal value of this binary number: 00001010.

Write the decimal value: _____

8. Assume this binary number, 11010111, needed to be rewritten as a hexadecimal number, what would be its hexadecimal symbol. Remember, each group of 4 binary digits is a single hexadecimal number. (you may use the chart in Figure 1.2.11).

Binary 1101 0111

Hexadecimal _____

9. Using Figure 1.2.13 what EBCDIC symbol would the binary code 11000011 represent _____.
10. The hexadecimal system uses six characters for number values which the common decimal system does not use. What are they?

Solution 1.2

1. A - decimal
B - hexadecimal
C - binary
2. binary
3. position (or place)
4. A - most significant (or high order)
B - least significant (or low order)
5. byte
6. byte
7. ten
8. binary 1101 0111
hexadecimal D 7
9. EBCDIC C
10. A (10), B (11), C (12), D (13), E (14), F (15)

Question #	Section Heading
1.2.7	Binary concept
3,4,5,6	Positional notation
8,10	Hexadecimal System
9	Eight bit alphameric code

Topic 3 - The Processor

Upon completion of this topic, the student should be familiar with the functional components of processors and the features they provide.

The Central Processing Unit (CPU) is the work and control center of the data processing system. Figure 1.3.1 shows the functional structure of 4300 processors. The important thing to note is that the CPU is functionally divided into two sections, instruction processing and system control.

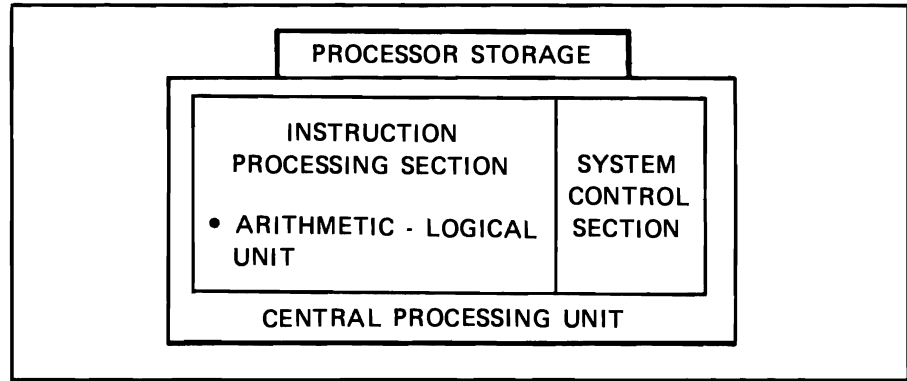


Figure 1.3.1 4300 Processor Structure

The ALU

Of primary importance within the instruction processing section is the Arithmetic Logical Unit or ALU. The function of the arithmetic logical unit is to perform the operations defined in the instructions being executed. In general, the ALU contains the circuitry for arithmetic operations, such as add, subtract, multiply, and divide. It also contains the circuitry necessary for logical operations, such as, comparing two data fields to determine which has the higher value. Other ALU instructions provide the capability to move data from one place in storage to another.

The ALU uses several different approaches to handling data. The approach used in any particular case depends on the type and location of data that is involved in the operation to be performed. Some of the methods of data manipulation involve the use of high speed storage devices called registers.

Registers

4300 processors provide 16 registers, called GENERAL PURPOSE REGISTERS, for use by both the system and programs. They are used in arithmetic or logical operations to either temporarily hold data or to locate data in storage. For example an instruction may call for the addition of two data fields. The value of the fields may be temporarily placed in registers and the result also stored there. Operations involving registers are extremely fast because of the high speed circuitry used in registers. 4300 processors contain other, specialized types of registers such as CONTROL REGISTERS which are used by the system for system control functions.

Control Section

The second functional area of the processor, the control section, directs and coordinates all operations called for by instructions. This involves control of input/output devices, entry or removal of information from storage, and

routing of information between storage and the arithmetic/logical unit. Also within the control section are the facilities necessary to monitor, maintain and service the system. Through the action of the control section, automatic, integrated operation of the entire computer system is achieved.

In many ways the control section can be compared to a telephone exchange. All possible data transfer paths already exist, just as there are connecting lines between all telephones serviced by a central exchange (Figure 1.3.2)

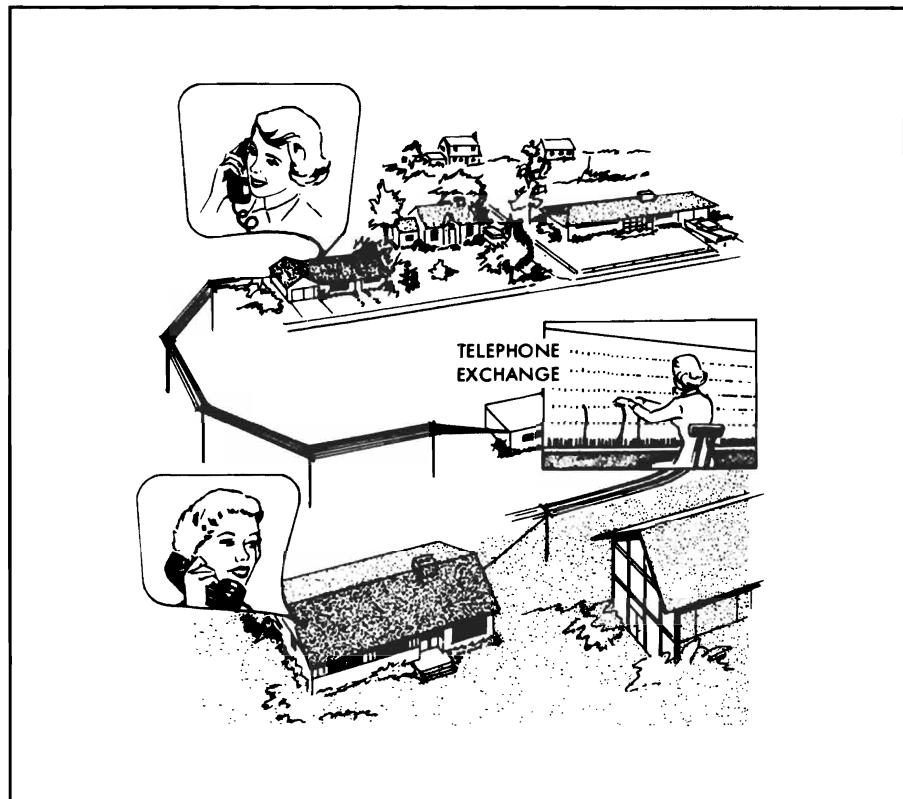


Figure 1.3.2 Telephone Exchange System

The control section is the central processing unit's internal communication center.

Machine Cycles

Activity within the Central Processing Unit must be designed and coordinated to make the most efficient use of the processor. To accomplish this all computer operations take place in fixed intervals of time. These intervals are measured by regular pulses emitted from an electronic clock at frequencies as high as several million per second. A fixed number of pulses determines the time of each basic machine cycle. This varies among processor types.

In computer usage, time references are stated in such terms as milliseconds, microseconds, and nanoseconds. These terms may convey no meaning unless it is realized just how short an interval a millisecond is. For example, the blink of an eye takes about one-tenth of a second or 100 milliseconds.

The following table establishes some additional terms and abbreviations:

0.1	= 1/10 second = 100 milliseconds
0.001	= 1/1,000 second = 1 millisecond (ms)
0.000001	= 1/1,000,000 second = 1 microsecond (usec)
0.000000001	= 1/1,000,000,000 second = 1 nanosecond (ns)

Within a machine cycle, the computer can perform a specific machine operation. The number of operations required to execute a single instruction depends on the instruction. Various machine operations are thus combined to execute each instruction.

To receive, interpret, and execute instructions, the central processing unit must operate in a prescribed sequence, which is determined by the specific instruction and is carried out during a fixed interval of timed pulses. One of the measurements of processor performance is the time required to perform its various operations.

Exercise 1.3 Match the components on the left with one or more of the phrases on the right. Use each phrase only once in the most appropriate place.

- | | |
|------------------------------------|--|
| a. General purpose register | 1. The time required by a processor to perform a machine operation. |
| b. CPU | 2. 16 of these are provided for both system and program usage. |
| c. ALU | 3. Can be functionally divided into two areas. |
| d. Machine cycle | 4. The blink of an eye takes about this long. |
| e. 1/10 second or 100 milliseconds | 5. Contains circuitry to perform arithmetic as well as logical operations. |

Solution 1.3

- a. 2
- b. 3
- c. 5
- d. 1
- e. 4

Question #

a,b,c
d,e

Headings

The processor
Machine cycles

Topic 4 - Main and Auxiliary Storage

Upon completion of the topic, the student should be able to describe the different types of storage available in a data processing system, understand why different types are necessary and how they are used.

All data to be processed must pass through some storage media. Instructions which operate on data must retrieve that data from storage and the results generated must be returned to storage. Even the instructions reside in storage. Storage or memory is classified as main or auxiliary, (Figure 1.4.1) Main storage includes all processor storage.

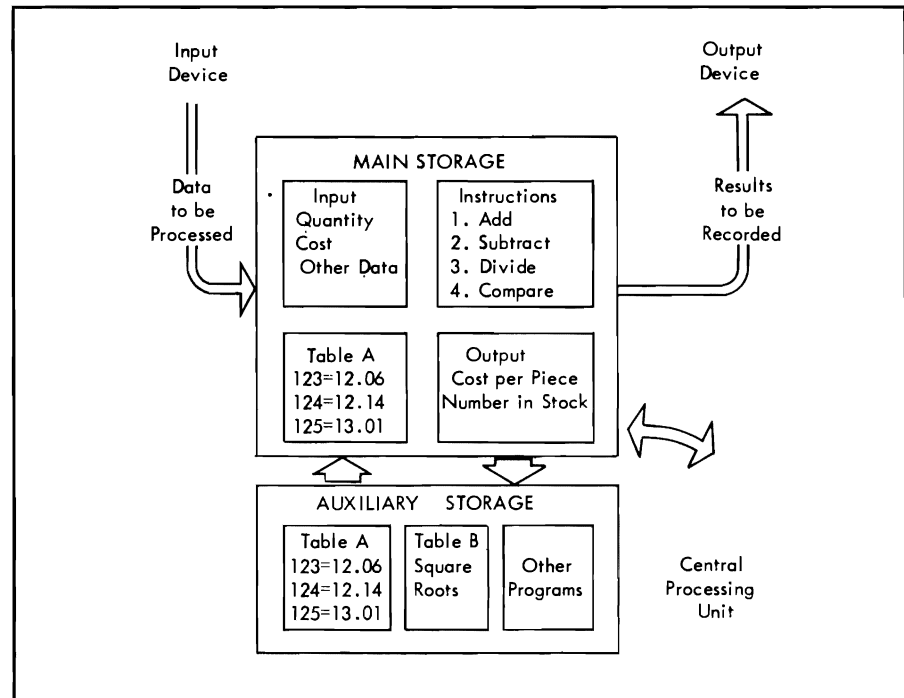


Figure 1.4.1 Schematic, Main, and Auxiliary Storage

Auxiliary Storage

Auxiliary storage refers to all other storage and is of two types:

1. Direct Access - Direct Access Storage Devices (DASD) are devices in which records can be accessed without having to read from the beginning of a file to find them.
2. Sequential - Sequential storage devices are devices in which all records must be read from the beginning in order to read or write a desired record. An example of this type of device is a magnetic tape unit.

Disk storage, provides IBM data processing systems with the ability to record and retrieve stored data sequentially or randomly (directly). It permits immediate access to specific areas of information without the need to sequentially examine all recorded data. By contrast, magnetic tape searching must start at the beginning of the tape reel and continue sequentially through all records until the desired record is found.

For an example of the application of direct access operations, as compared to sequential operations, consider the search for a word in a large unabridged

dictionary. If the contents of the dictionary were stored on magnetic tape, the complete dictionary could be machine-read in about two minutes. A wide range of individual words would require an average of one minute to be found and read by the magnetic tape sequential method of searching. Using the dictionary, a human being would average about 1/5 of a minute per word, simply because he would limit his search for each word to an appropriate portion of the whole dictionary. That is, he would immediately go to a specific letter rather than start at the beginning of the dictionary and check each entry. This concept of limiting a search to a small section of the whole would permit direct access storage to perform the dictionary word search in a few thousandths of a second.

Main Storage

Processor or main storage accepts data from an input unit, exchanges data with and supplies instructions to the central processing unit, and furnishes data to an output unit. All data to be processed by any processor must pass through main storage. This unit must therefore have capacity to retain a usable amount of data and the necessary instructions for processing. See Figure 1.4.2.

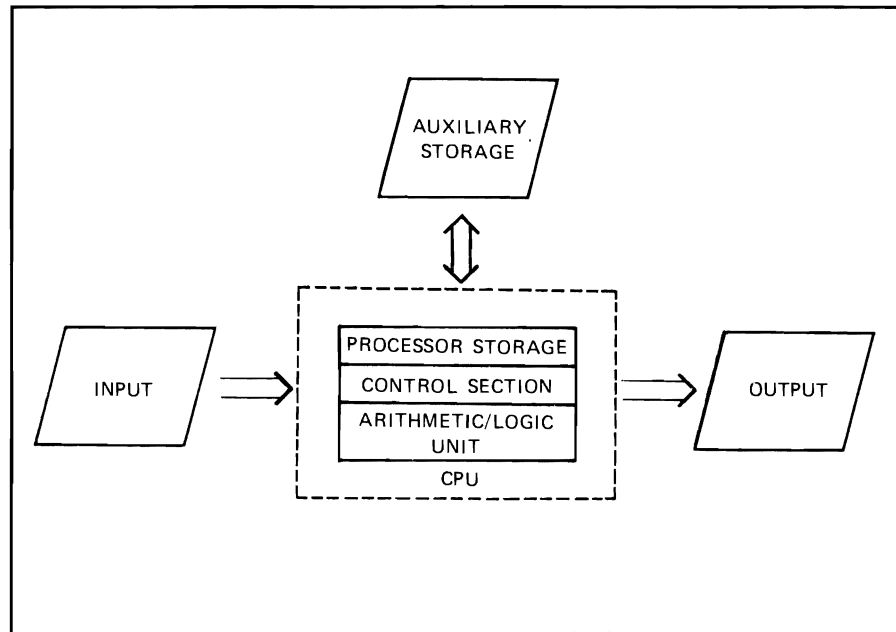


Figure 1.4.2 CPU With Storage

Storage is arranged somewhat like a group of numbered mail boxes in a post office (Figure 1.4.3).

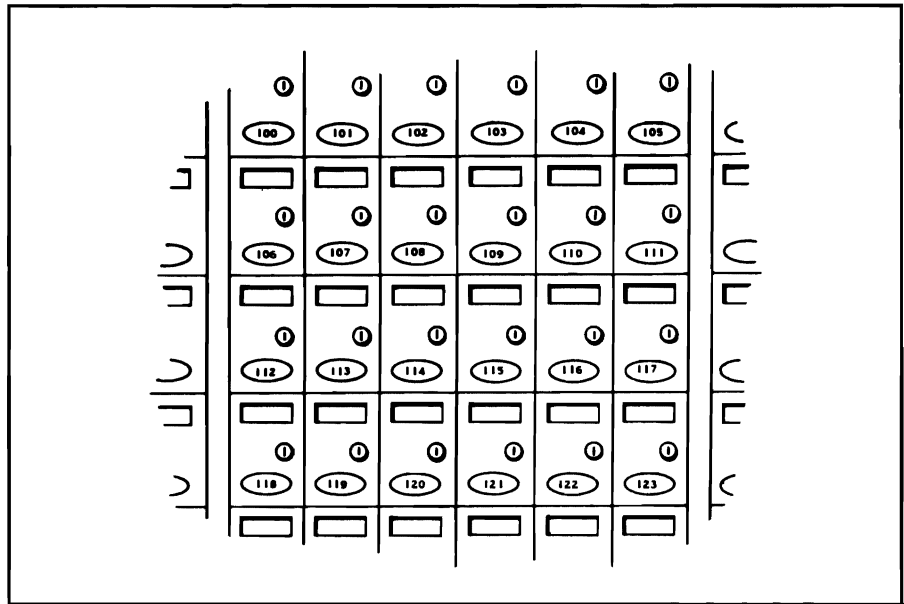


Figure 1.4.3 Post Office Mail Boxes

Each box is identified and located by its number. In the same way, storage is divided into locations, each with an assigned address. Each location holds a specific unit of data. Depending on the system, the unit of data may be a character, an entire record, or a word. To insert or remove data at a location, the address must be known.

When information is placed into a storage location, it replaces the previous contents of storage at that location. However, when information is taken from a location, the contents remain unaltered. Thus once located in storage, the same data may be used many times.

So we see that a data processing system (Figure 1.4.4) contains auxiliary storage, processor storage, and a central processing unit to process data into information. It also must incorporate devices for the transfer of data into and out of the CPU.

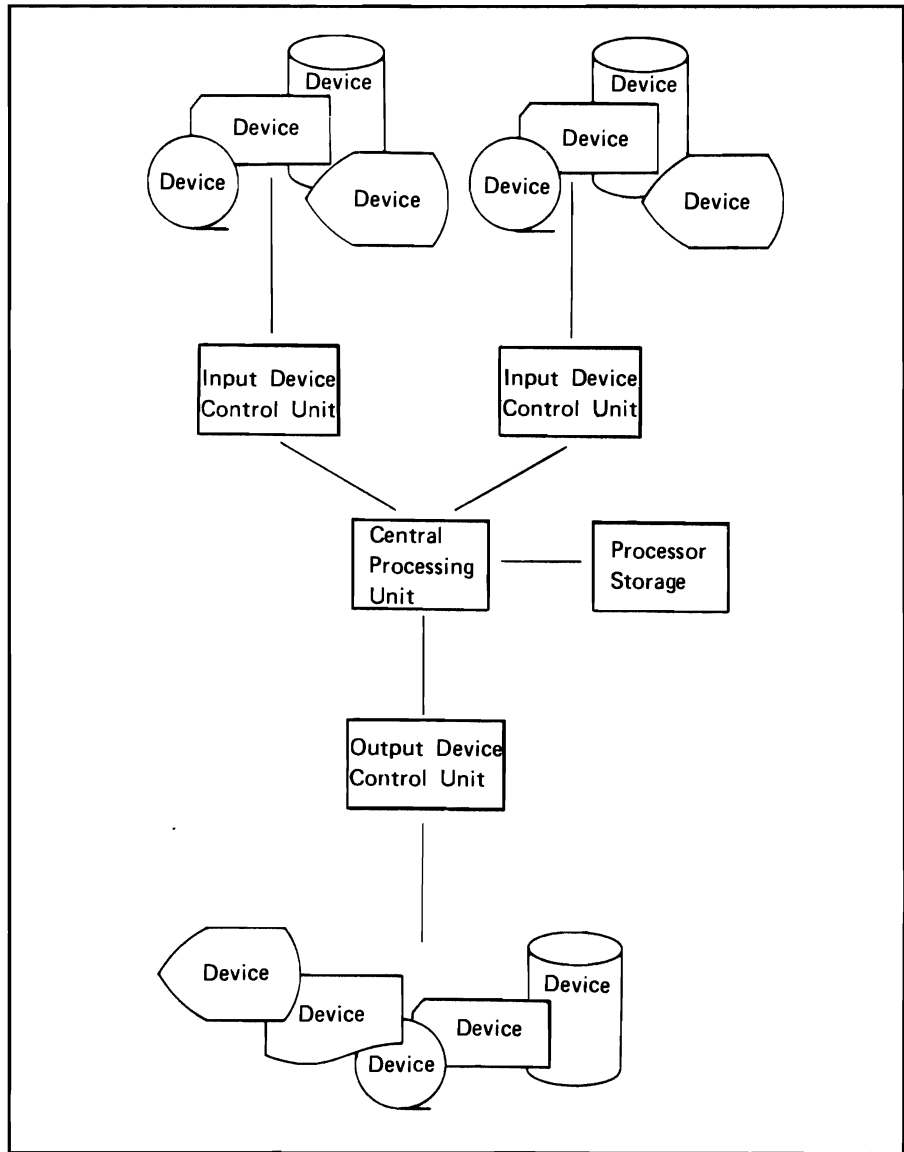


Figure 1.4.4 Units in the Data Processing System

Exercise 1.4

1. Match the words with the phrases which best describe them.
 - A. Processor storage 1. Permits both random and sequential access
 - B. Disk storage 2. Permits only sequential access
 - C. Magnetic tape 3. Also called main storage
2. All data to be processed by any system must pass through main storage. True or False
3. Processor storage is divided into locations, each of which has an address known to the computer system. True or False

Solution 1.4

1. A - 3, B - 1, C - 2
2. True
3. True

Question #	Heading
1	Main and auxiliary storage
2,3	Main storage

Topic 5 - Input and Output Devices

The input/output unit is a device in a data processing system by which data may be entered into the system, received from the system, or both.

Many input/output devices function with an external document, such as a punched card or a reel of magnetic tape. Others such as keyboard/display terminals and channels, handle only electrical signals. Usually, device operation is initiated by a program instruction which generates a command to a channel. The channel passes the command to a control unit which interprets the command and starts operation of the device.

This topic describes various types of input and output devices and the functions of each. It also provides a general understanding of how the devices operate together in a data processing system.

Control Units

A control unit is a device that controls input/output operations at one or more devices. These operations can be the reading, writing or displaying of data. The control unit function may be contained within the input/output device or in the CPU, or a separate control unit may be used. In all cases, the control units provide the logic circuitry and the storage areas (BUFFERS) needed to operate the attached input/output devices.

Channels

The electronic path over which data is transmitted between main storage and the input/output devices (and their associated control units) is called a channel. Channels do more than transmit information. A channel is similar to a small computer. It is controlled by a set of instructions just as the CPU is. The instructions that direct the channel are called **COMMANDS** (Figure 1.5.1) in order to distinguish them from instructions for the CPU. Commands reside in main storage, just as CPU instructions do. Just as CPU instructions are put together to form programs for the CPU to execute, channel commands are put together to form channel programs for the channel to execute.

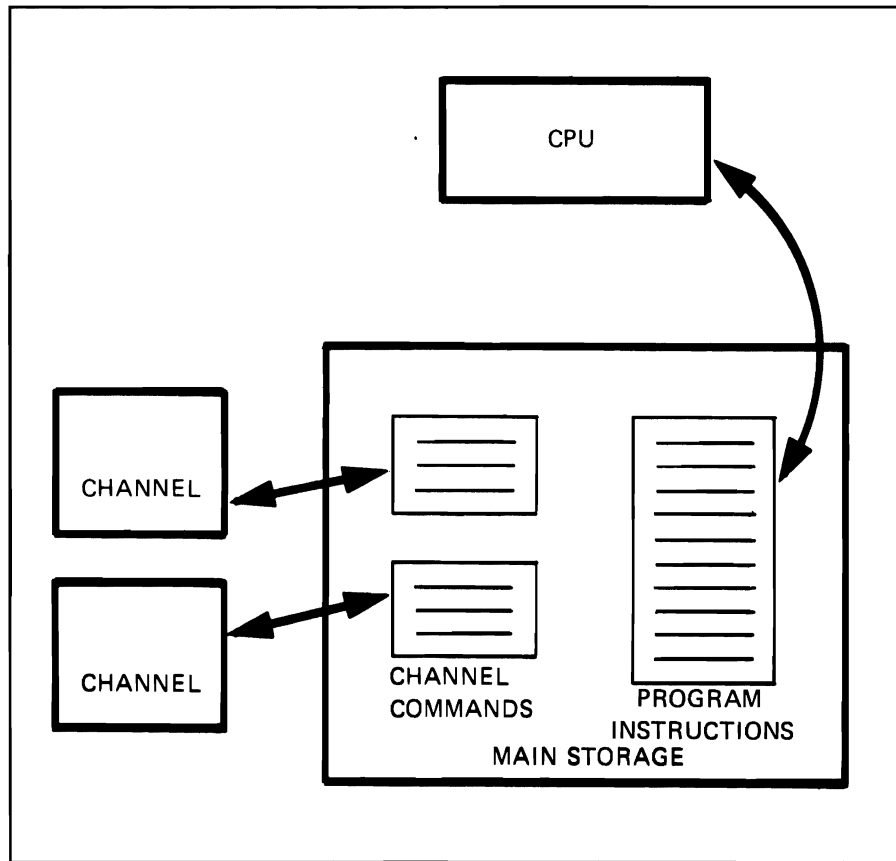


Figure 1.5.1 The CPU Executes Instructions; a Channel Executes Commands

The CPU is free to continue program execution once it has given an instruction to start a channel on its independent program of commands. Likewise, a channel is free to execute other commands during the time that the control unit and I/O device are performing their functions.

Types of Channels

IBM systems have three types of channels; byte multiplexer, selector and block multiplexer.

Byte Multiplexer Channels

A multiplexer channel is a channel designed to operate simultaneously with a number of input/output devices. Several input/output devices can transfer records at the same time. A byte multiplexer channel transfers records one byte at a time. This process is slow relative to the other two channel types. A byte multiplexer is usually used for slower devices such as card devices and printers.

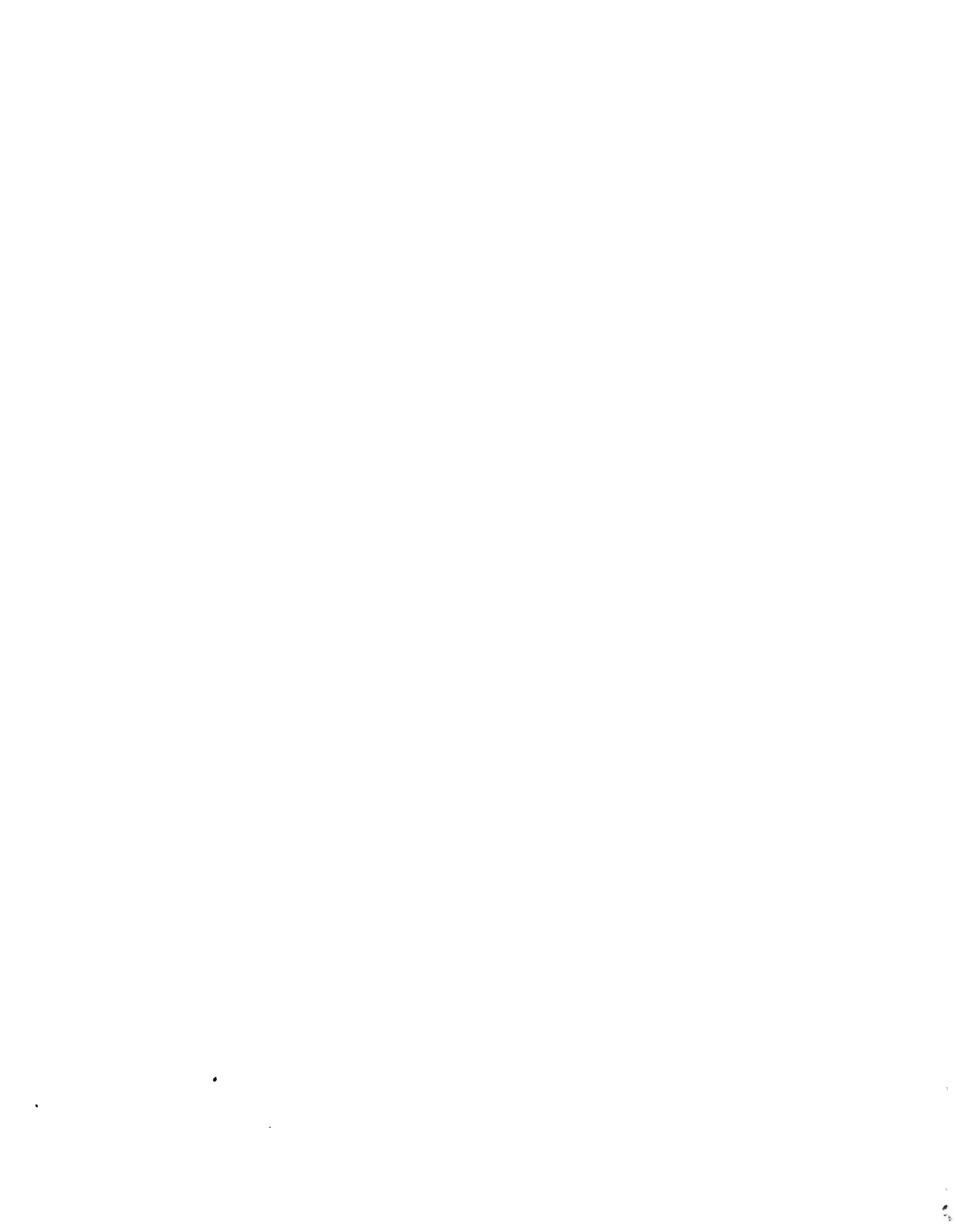
Selector Channels

Selector channels transmit data to or from a single input/ output device at a time. Once the input/output device is selected a complete record is transferred one byte at a time. The data transfer rate of selector channels is much

faster than that of a byte multiplexer channel. A selector channel can be used for most high speed devices such as DASD.

Block Multiplexer Channels

The block multiplexer channel is a multiplexer channel that transfers blocks of data instead of bytes. A block is a string of records, a string of words, or a character string formed for technical or logic reasons to be treated as an entity. Block multiplexer channels can operate with much faster input/output devices, and transfer larger quantities of data per transmission. The data transfer rate of a block multiplexer channel is faster than either of the other two channel types.



Exercise 1.5.1 Before you go on, make sure you understand the function of the devices just discussed. Can you correctly answer these questions?

1. The electronic path over which data is transmitted is called a _____.
 - a) control unit
 - b) channel
 - c) I/O device
2. True or False. A control unit is a device that controls I/O operations at one or more devices.
3. A _____ is similar to a small computer in that it executes instructions (called commands).
4. Channels operate at different speeds. The fastest is the _____, the next fastest is the _____ and the slowest is the _____.

Solution 1.5.1

1. channel
2. True
3. channel
4. a block multiplexer, selector, byte multiplexer

The input/output device is the computer's door to the outside world. Let's take a look at the primary input/output device types.

Direct Access Storage Devices (DASD)

These devices, which can access data directly as well as sequentially, use media such as magnetic disks.

The magnetic disk is a thin metal disk coated on both sides with magnetic recording material. Disks are mounted on a central shaft and are slightly separated from one another to provide space for the movement of read/write heads. The shaft revolves, spinning the disk (Figure 1.5.2).

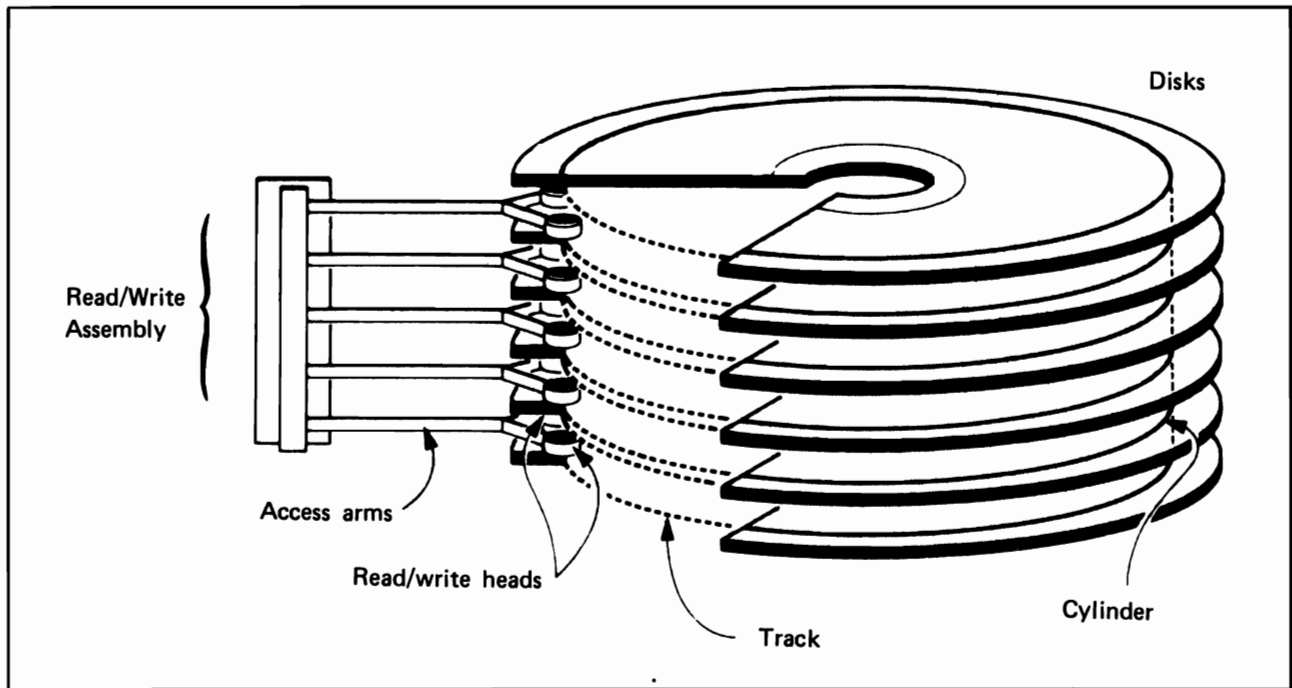


Figure 1.5.2 Magnetic Disk Structure

Data is stored as magnetized spots in concentric tracks on each surface of the disk. The tracks are accessible for reading and writing by positioning the read/write heads between the spinning disks.

The magnetic disk data surface can be used repetitively. Each time as new information is recorded and stored on a track, the old information is erased. The recorded data may be read as often as desired; data remains recorded until written over. Information is recorded on all DASD devices in a format which is prescribed by the control unit. Each track contains certain "nondata" information (such as the address of the track, the length of each record, and gaps between records) as well as data information.

The diskette is a single magnetic disk, sealed in a plastic jacket about 8 inches square. It weighs less than 2 ounces, is reusable, is interchangeable with other diskette units, and can be easily corrected and updated. The diskette can store as many as 1898 128 character records, equivalent to as many characters as can be put on 3,036 80 column cards.

Magnetic Tape Units

Magnetic tape units (Figure 1.5.3), with their dual capability of input and output, provide the capability to store data on reels of magnetic tape. The data stored on magnetic tape may be read by any tape unit with the capability to read data in the recorded format.

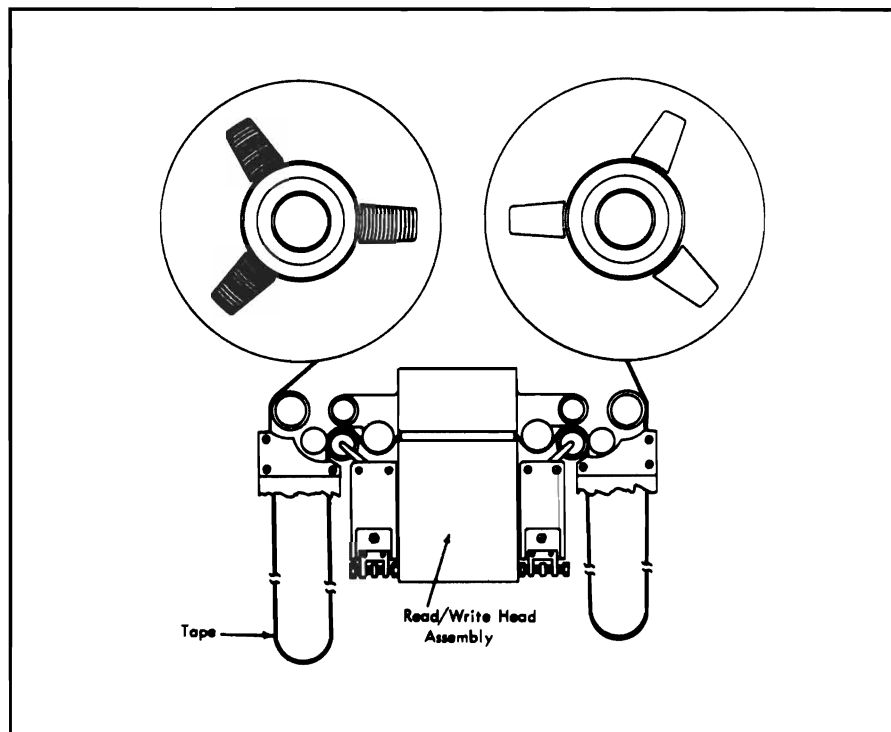


Figure 1.5.3 Tape Unit Structure

All magnetic tape units are basically similar in operation, but there are functional differences which affect the speed and ease of operation.

Writing on magnetic tape is destructive, that is, as new information is written, old information is destroyed. Reading is nondestructive; the same information may be read again and again.

Information is written on tape by magnetizing areas in parallel tracks along the length of the tape. Among tape units, the major performance considerations are the speed at which tape is moved across the read/write head and the density of information on tape.

Tape Records, Interblock Gap and Tapemark

Records on tape are not restricted to any fixed length of characters, fields, words, or blocks. Records may be any practical size.

Blocks of records, including blocks consisting of a single record are separated on tape by an interblock gap, a length of blank tape (Figure 1.5.4). During writing, the gap is automatically produced at the end of each block of records. During reading, the block begins with the first character sensed after a gap and continues without interruption until the next gap is reached.

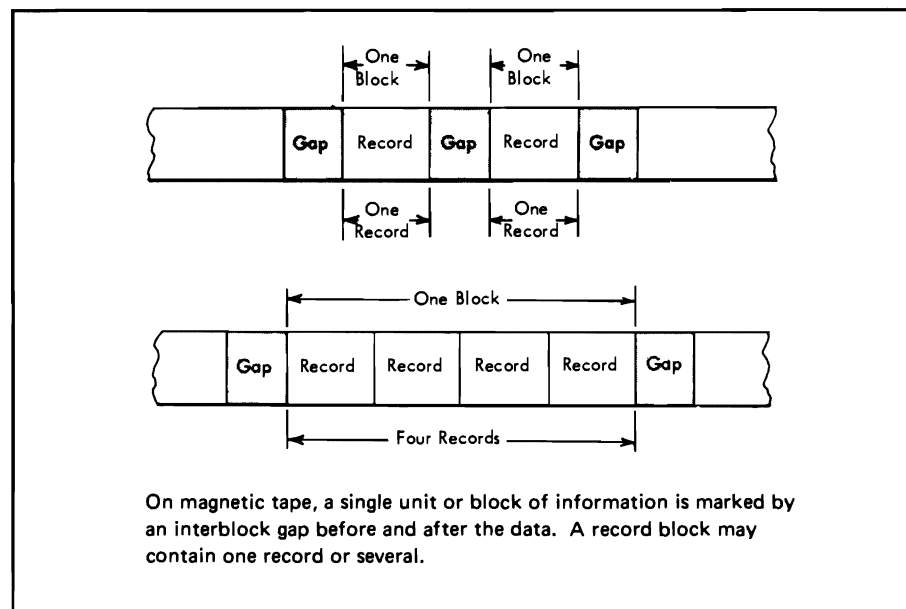


Figure 1.5.4 Blocks of Records on Tape.

On magnetic tape, a single unit or block of information is marked by an interblock gap before and after the data. A record block contains one record or several.

Load-Point and End-of-Reel Markers

Magnetic tape must have blank space at the beginning and end of the reel to allow threading through the feed mechanism. Reflective markers are used to enable the magnetic tape unit to sense the beginning and the end of the usable portion of tape. Photoelectric cells in the tape unit sense the markers as either the load-point marker (where reading or writing is to begin) or the end-of-reel marker (where writing is to stop).

Card Readers

Card reading devices introduce IBM punched card data into the computer system. The card reader moves or feeds cards past a reading unit that converts the data on the card into an electronic form. Two types of reading units are used: reading brushes or photoelectric cells.

In the brush type reader, cards are mechanically moved from a card hopper, through the card feed unit, and under reading brushes. The reading brushes electrically sense the presence or absence of holes in each column of the card (Figure 1.5.5).

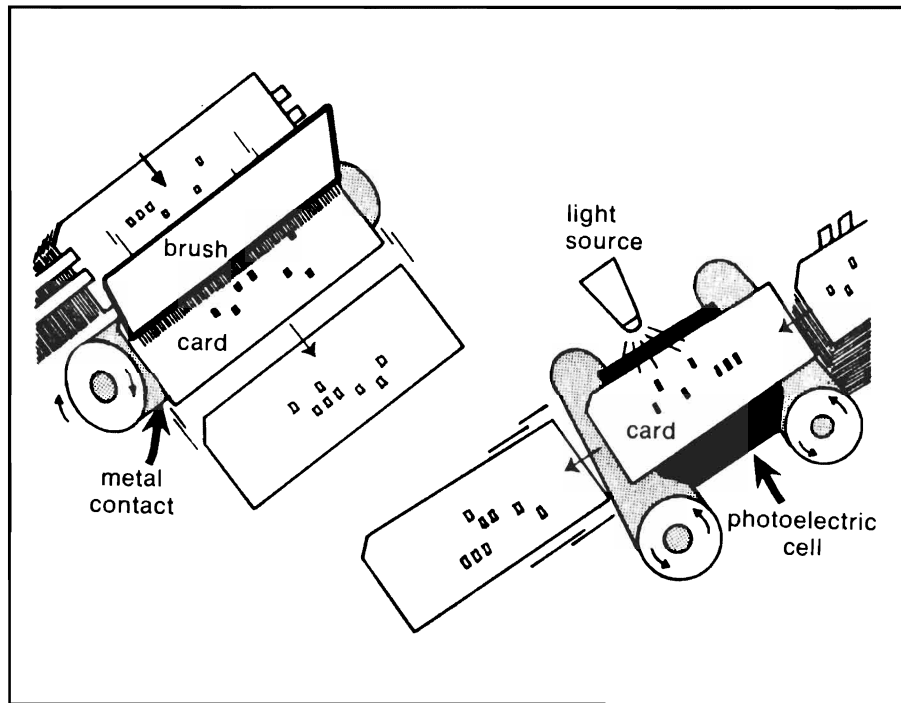


Figure 1.5.5 Card Reader

This electric sensing converts the information in the card to electrical impulses that can be detected by the card reader circuitry and stored as data. After cards are read, they are moved from the card feed unit and placed in the card stacker in the same sequence in which they were fed into the reader. Some readers have two sets of reading brushes. As a check on the validity of the reading process, each card can be read twice as it moves through the card feed unit.

The photoelectric type of card reader performs the same functions as the brush type; the difference is in the method of sensing the hole. Photoelectric cells are activated by the presence of light. As the punched card is passed over a light source in the card reader, light passing through the punched holes activates photoelectric cells.

Card Punches

Output from the computing system is recorded in cards by a card punching device. The card punch automatically moves blank cards one at a time, from the card hopper, under a punching mechanism that punches data received from storage (Figure 1.5.6). After the card is punched, it is moved to a checking station where the data is read and checked with the information received at the punching station. The card is then moved to the stacker.

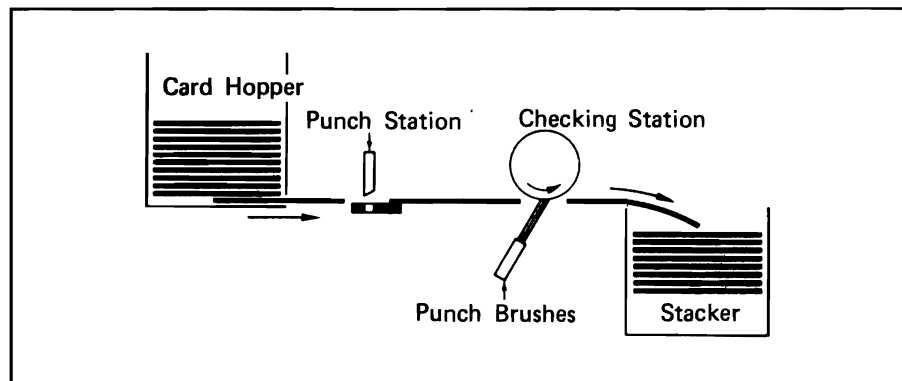


Figure 1.5.6 Card Punch

Printers

IBM printing devices provide a permanent visual record of data from the computer system. As an output unit, the printer receives data, symbolized in electronic form, from the computer system. The electronic symbols enter appropriate circuitry and cause printing elements to be actuated. All printing devices have a paper transport that automatically moves the paper as printing progresses.

The major printing devices consist of the wire matrix printer, chain printer, the typewriter, and the electrophotograph and laser beam.

Wire Matrix Printer

In the wire matrix printer each character is printed as a pattern of dots formed by the ends of small wires, arranged in five-by-seven rectangle (Figure 1.5.7). By extending selected wires, the patterns may be arranged in the shape of 47 different characters, including all letters of the alphabet, the digits 0 to 9, and eleven special characters used for punctuation and report printing. Selected wires are pressed against an inked fabric ribbon to print the characters on paper.

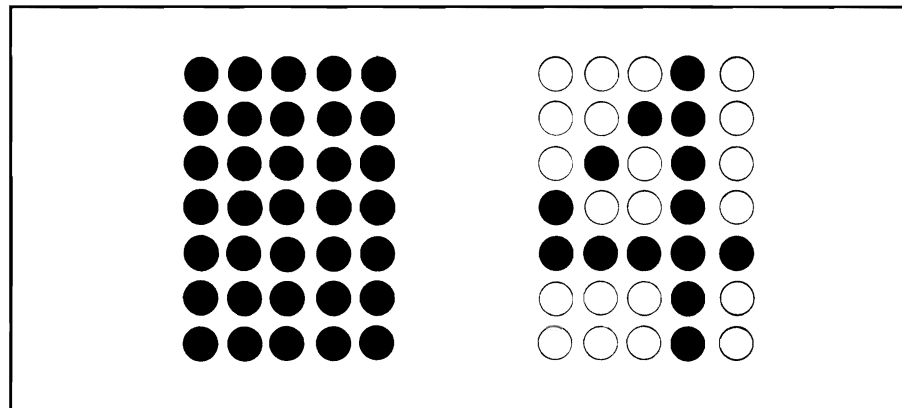


Figure 1.5.7 5x7 Dot Pattern

Chain Printer

The chain printer is an electromechanical line printer using engraved type. Alphabetic, numeric and special characters are assembled in a chain (Figure

1.5.8). As the chain travels horizontally, each character is printed as it is positioned opposite a magnetically actuated hammer that presses the paper against one piece of type in the moving chain. The print chain can be easily changed to provide a choice of print fonts.

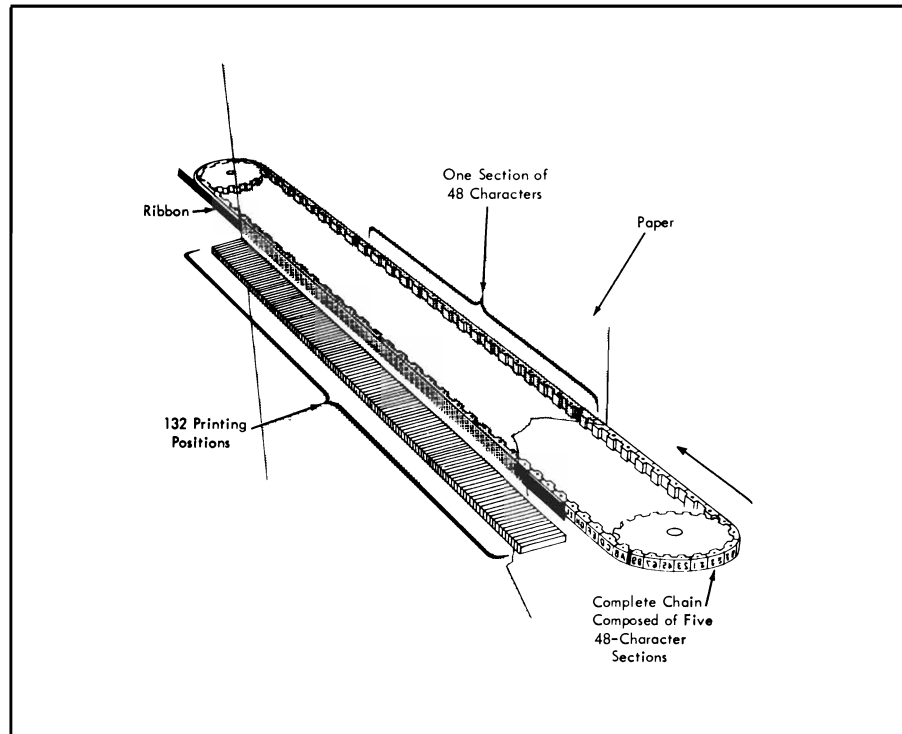


Figure 1.5.8 Print Chain

Typewriter

The typewriter that is used as an output device is similar to the one used manually. The major difference is that control of the typewriter and the printing occurs automatically as directed by the computer.

3800 Printer

The 3800 Printer uses electrophotographic and laser technology to reach very fast printing speeds. It prints on one-ply continuous paper, and uses its high speed to repeat-print multiple copies of original quality. Since the paper is always single ply, the cost of carbon paper, and the expense and delay involved in removing it are eliminated. It can use pre-printed forms or forms generated by the 3800 printer itself through either forms overlay or formatting technique.

Visual Display Unit

Visual display units or terminals (Figure 1.5.9), permit the user of a computing system to display information on a cathode ray tube device that is similar to a television screen. In a typical application, the display terminal is used to display a customer account record while the user is discussing the account with the customer. The application may also provide the capability for the user to modify the customer record by using the terminal's keyboard to type in the modified data.

This type of application, where the user has rapid access to data through a terminal, is often referred to as an *on-line* application.

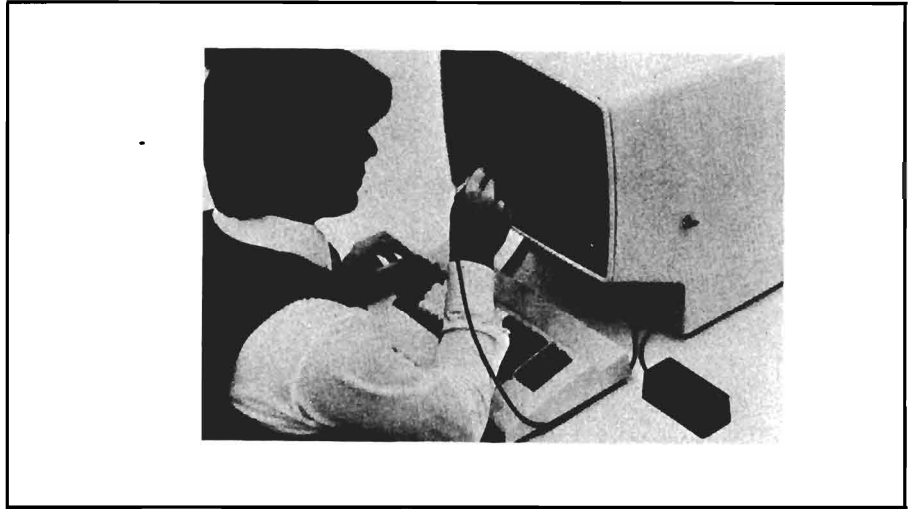


Figure 1.5.9 Visual Display Unit

Console The console (Figure 1.5.10) is an input/output device that provides external control of the data processing system. Switches turn power on or off, start or stop operation, and control various devices in the system. Data may be entered directly by manually depressing keys. Consoles may include a keyboard with a visual display and/or a printer. The primary difference between a console and other visual displays is that the console is used to control the data processing system. On some systems, a printer and keyboard combination is used for the system console (Figure 1.5.10).

The console device may display or print messages signaling the end of processing or an error condition. It may also display or print totals or other information that enables the operator to monitor and supervise operation, or it may give instructions to the operator.

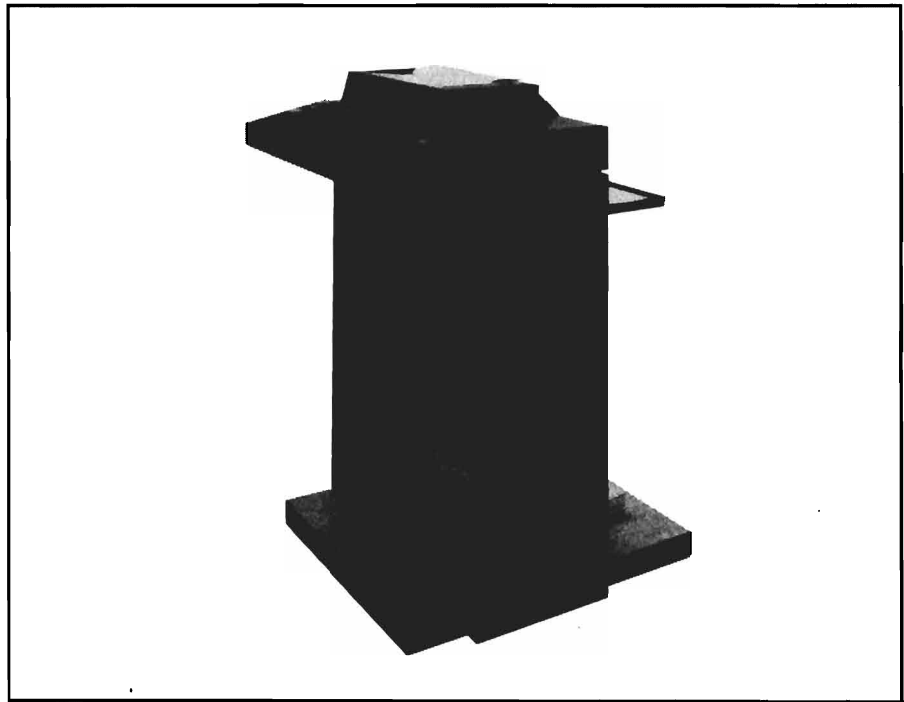


Figure 1.5.10 Console

Exercise 1.5.2 We have talked about many different types of input and output devices. Let's review them.

Use the list below to answer the "fill in the blank" questions.

- a. channel(s)
 - b. control unit(s)
 - c. card reader
 - d. card punch
 - e. DASD
 - f. magnetic tape
 - g. gap
 - h. printer(s)
 - i. eletrophotographic
 - j. print chain
 - k. console
 - l. terminal
1. A device which can access data directly and sequentially using magnetic disks is _____.
 2. Among _____ units, the major performance consideration is the speed at which the medium is moved across read/write heads and the recording density of the data on the medium.
 3. The function of a _____ is to separate different areas either on tape or disks.
 4. The _____, is an input device which senses holes punched in IBM punch card while the _____ is an output device which records data from the computer in IBM punch cards.
 5. The output device which provides a permanent visual record of data from the computer system is a _____.
 6. Printing devices record data by a variety of methods, by pressing wires against inked ribbon, by a hammer pressing the paper against one piece of type in a moving chain or by a very fast laser and _____ process.
 7. A visual display unit or _____ allows a user of a computer system to see data from the computer much faster than a printer or card punch.
 8. The input/output device which provides external control of the data processing system is the _____.

Solution 1.5.2

1. e
2. f
3. g
4. c, d
5. h
6. i
7. l
8. k

Input/Output Operations

Now let's see how all of the units work together to perform an input/output operation involving transfer of data.

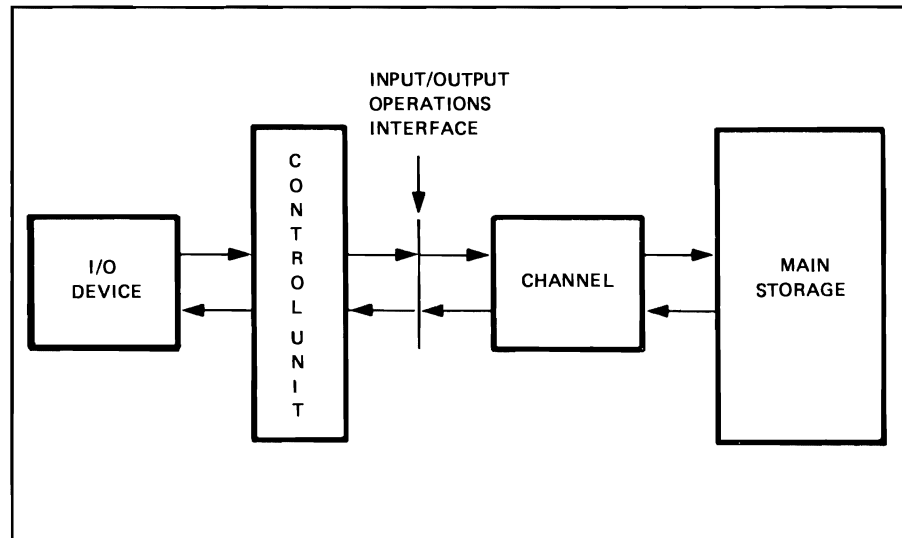


Figure 1.5.11 Data Transmission To And From Storage

An input/output operation is initiated by a CPU instruction, such as the start input/output instruction. This instruction starts the channel and specifies which input/output device is to be used. The channel then signals the control unit to start the specified device. Once started, a channel performs an input/output operation by sequencing through a set of channel commands. The commands specify such things as how much data is to be transmitted, where in main storage the data is to be placed, the rewinding of magnetic tape and the positioning of a disk access mechanism. When the input/output operation is completed, the channel causes an input/output interrupt (stop a process in such a way that it can be resumed) and passes status information to main storage. The status information describes the conditions of I/O completion.

Once the channel has been started it can operate independently of the CPU. Thus, the CPU is free to continue executing instructions. This means that the input/output operation can take place concurrently with internal processing. Not only can the channels operate independently of the CPU, but they can operate independently of each other. This makes it possible to overlap input/output operations with one another as well as with internal processing. Overlapped processing means that reading, computing and writing of data can take place concurrently. For example, data can be transferred between main storage and the input/output devices while the CPU is executing instructions.

Topic 6 - Stored Program Concepts and Compilers

Upon completion of this topic the student should know what a program is, what instructions are and understand their function in data processing systems. The student should also have a general understanding of the compiler and linkage editor functions and should be aware of the various compilers and languages available for 4300 systems.

Introduction

By now, you should have a good idea of the primary functions of the hardware units of a data processing system. I/O units such as card readers, magnetic tape, disk units and printers are the hardware units used to get data into and/or from the data processing system. The CPU contains the electronic circuits required to perform arithmetic and logical operations necessary for processing the data. Now let's consider how the hardware is controlled and directed to process data.

The computer may have the operation of multiplication built into its circuits in much the same way as the electronic hand calculator but, there must be some means of directing the computer to perform multiplication just as the calculator is directed by depressing keys.

There must also be a way to instruct the computer where in storage it can find the factors to multiply. Further, the comparatively simple operation of multiplication implies other activity that precede and follow the calculation. The multiplicand and the multiplier must be read into storage by an input device. Once the calculation is performed, the product must be returned to storage at a specified location, from which it may be written by an output device.

Any calculation, therefore involves reading, locating factors in storage, performing the calculation, returning the result to storage, and writing out the completed result. Even the simplest portion of a procedure involves a number of planned steps or operations that must be spelled out to the computer if the procedure is to be accomplished.

Instructions

The computer is directed to perform each of its operations by an instruction - a unit of specific information located in main storage. This information is interpreted by the central processing unit as an operation to be performed.

If data is involved, the instruction directs the computer to the data. If some device is to be controlled - a magnetic tape unit, for example - the instruction specifies the device and the required operations.

An instruction usually consists of at least two parts:

1. An operation part that designates read, write, add, subtract, compare, move data, and so on.
2. An operand that designates the address of the information or device that is needed for the specified operation.

During an instruction cycle, an instruction is selected from storage and analyzed by the central processing unit. The operation part indicates the operation to be performed. This information is coded to have a special meaning for the computer. For example, the letter "A" may be interpreted as "add", and the letter "C" as "compare".

The operand further defines the function of the operation. For example, to perform arithmetic, the storage location of the factors involved is indicated. For input or output devices, the unit to be used is specified. For reading or writing, the area of storage for input or output records is indicated.

Program A series of instructions pertaining to an entire procedure is called a program. In modern data processing systems, the program is stored internally, and the system has access to the instructions at electronic speeds. Such programs are called stored programs.

Programs are often classified as being either *batch* or *on-line*. A batch program or application is one that executes without interacting with a user through a terminal device. Batch programs usually read *all* input data from some pre-recorded media such as punched cards, diskette, tape and disk files.

An on-line program or application is one which provides an interface for interactive input and output with the user. The interface is usually a video display or some other terminal device. The on-line program or application is usually invoked through the terminal and provides instant access to information stored in the computers DASD devices. Most on-line applications require a communications control program that provides for the control and sharing of system resources with other terminal users.

Programming Languages

The computer executes instructions that are presented to it in machine language. In the IBM 4300, machine language instructions are in binary format. Even with the use of hexadecimal equivalents, writing a program in machine language is extremely difficult, costly, and time consuming. Furthermore, such a program is likely to contain numerous errors. Subsequent corrections may be practically impossible to make without introducing other errors. Instead of coding a program in machine language, the programmer writes the program in one of many languages that are more easily understood by the programmer and are a more convenient means of specifying the problem. Since the expressions of the particular language cannot be executed directly by the computer, the expressions are translated into machine language by the computer using previously written language translator programs for the specific language.

A programming language can be thought of in two parts:

- (1) the language itself, with associated rules of grammar, and
- (2) a machine language program (the compiler), whose main function is to translate the language of the programmer into machine language.

Source Program

The input to a compiler is called the source program. A source program is written by a programmer in the programming language of the system. It consists of instructions and other information that the programmer has written to describe the operations required to perform the job.

Object Program

The output from the compiler is the object program (or object module), the translation of the source program from the programmer's machine language to

the language of the computer system on which the program will be used.

Linkage Editor

In some systems the object program may be executed. In others, the object program is not in executable format but must be processed by a program called the Linkage Editor which will produce a load program (or load module) which is an executable program.

Languages

There are several programming languages in wide use today. COBOL (Common Business Oriented Language), RPG II (Report Program Generator), PL/1 (Program Language/1), and FORTRAN (Formula Translation), are some of the programming languages used with the IBM 4300.

The instructions written by a programmer using one of these languages bears little resemblance to machine language.

As an example of how the COBOL programmer might program a simple problem, assume we wish to increase the value of an item called INCOME by the value of an item called DIVIDENDS. The COBOL language allows us to specify the addition by writing the following sentence:

ADD DIVIDENDS TO INCOME.

This simple, English like instruction, directs the computer to perform the required calculation.

Not only are stored programs those which are written to complete a task for a specific organization but also programs which perform generalized functions like compilers and linkage editors. In the next topic we will talk about another type of stored program - the system control program, but first review what you have learned by answering the questions on the next page.

Exercise 1.6

1. An _____ is a means of directing the computer to perform an operation.
2. An instruction consists of at least two parts: an operation, designating what is to be done and an operand designating the location of the device or data for the operation. True or False
3. A series of instructions pertaining to an entire procedure is called a _____.
4. Because modern data processing systems permit internal storage of programs where the instructions can be accessed at electronic speeds, the programs are called stored programs. True or False
5. Programmers usually write programs in machine language. True or False
6. High level languages available for programming in 4300 data processing systems include:
 - a) _____
 - b) _____
 - c) _____
 - d) _____
7. A programming language consists of the language itself and a machine language program whose main function is to translate the high level language of the programmer into machine language. The translator program is called a _____.
8. The program input to the compiler is called a _____ program; once it has been translated, the compiler output is the _____ program or module.
9. While the object module from the compiler can be executed in some systems, in others the compiler output must be processed by another system program called a _____ to produce an executable program.
10. The executable program produced as described in question 9 is called a _____ module or program.

Solution 1.6

1. instruction
2. True
3. program
4. True
5. False
6. a) COBOL, b) RPG | | , c) PL/I, d) FORTRAN
7. Compiler
8. source, object
9. linkage editor
10. load

Question #	Heading
1,2	Instructions
3,4	Program
5,6,7,8,9,10	Programming languages

Topic 7 - The Operating System

At the completion of this topic the student should be able to recognize the software components of an operating system and complete sentences relating to how they interact in the data processing environment.

Introduction

The hardware components of a data processing system can be thought of in terms of resources capable of performing tasks required in the processing of data.

A typical data processing system consists of the following hardware resources:

- The Central Processing Unit
- Card Reader
- Card Punch
- Printer
- Tape Units
- Disk units
- Display Terminals

At any given time, a single program may only be using a fraction of these resources. For example, a payroll program to print pay checks may only require one tape unit, the printer and the CPU. In addition, when the program requests a read from the tape unit or a write to the printer, the CPU is idle while the read or write is taking place. Also, more likely than not, the payroll program only requires a fraction of the processor storage. When the payroll program ends, the entire data processing resource is idle until the operator starts the next program. The problem of ineffective resource utilization illustrated in this example is addressed by the implementation of an operating system.

Operating System

An operating system is a program or group of programs used to manage the resources of the data processing environment to maximize their use.

Some of the primary facilities available in most operating systems are:

- Job Management
- Supervisor
- Program Libraries
- Data Management
- Recovery Management
- Multiprogramming
- Spooling

Job Management

A major facility provided by an operating system is job management. A job is one or more programs which are logically related. For example, a payroll job might include programs to read time cards and compute gross pay, print

summary reports, print the payroll checks, and then store updated payroll information.

A key facility of job management is the ability to automatically handle program-to-program and job-to-job transition. New programs and new jobs can be initiated without intervention. This results in better CPU utilization and minimizes operator intervention.

In the example of printing payroll checks, the time between the end of the program and the start of another program would be minimized by this facility.

The job management facility is directed by information supplied by the programmer. He supplies this information via job control cards coded according to the specifications of a job control language or JCL facility.

In its simplest form JCL would signify to job management the beginning and ending of a job. The use of JCL would also provide other capabilities as you will learn later.

The operating system contains a special program called a "job control program" to read and interpret the job control cards. The job control program is invoked by the supervisor or "heart" of the operating system.

Supervisor

The supervisor is the interface between the user program and the resources of the system. It is that part of the operating system that coordinates the use of the resources and maintains the flow of CPU operations. When an executing job requires use of any system resource, it passes control to the supervisor. The supervisor in turn controls the use of the requested system resources.

Program Library

A program library is an area on DASD that is used to store programs. These programs can be loaded into processor storage for execution by facilities provided by the operating system. Using a program library further minimizes operator-machine interactions by providing an automatic way to load a program without the operator doing it from cards.

Data Management

The term "data management" describes the operating system functions that provide access to data, enforce data storage conventions and regulate the use of I/O devices. Data management refers to the CONTROL, STORAGE and RETRIEVAL of information that is to be processed by a computer.

"Control" is the supervision of the data management process. It establishes the user's right to access or modify the information in the system insuring data integrity (no data loss).

"Storage" is the technique for representing the information on a storage device such as DASD, tape or punched card. It includes the order in which the information is stored, the way it may be addressed and the method of representing the data itself.

"Retrieval" is the process of locating, formatting and sequencing information for the program. Locating information includes determining what data is required and where it may be found.

Data management supervises the transfer of data between the CPU and the I/O device and presents it to the user's program in the format that is desired. DATA MANAGEMENT is a major facility of an operating system that involves organizing, locating, storing and retrieving data.

Recovery Management

Recovery management provides a means of insuring a minimal loss of resources in the event of hardware or software failure. It provides facilities that gather information about hardware reliability and allows retry of operations that fail because of CPU, I/O device or channel errors. The information accumulated can be used to facilitate diagnosis of a malfunction and to service the hardware in less time. As you can see in Figure 1.7.1 some recovery management support is in the supervisor, but it is contained in hardware features and other systems software as well.

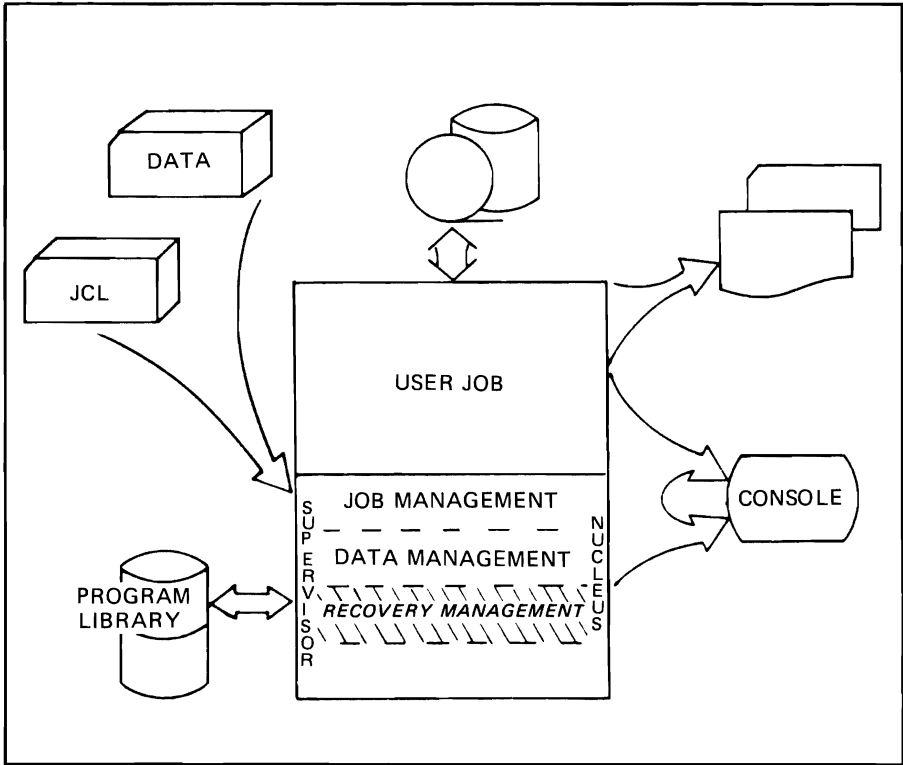


Figure 1.7.1 Operating System Structure

Exercise 1.7.1 Before we continue, use these questions to review what we have covered so far.

1. Job Management, data management, recovery management and task management are important functional areas of an operating system. True or False
2. Program to program and job to job transition is the responsibility of the _____ function.
3. The job control program interprets JCL to determine what user program is needed to execute a job. True or False
4. Data management is responsible for the _____, _____, and _____ of information that is to be processed by the computer.
5. The facilities for gathering information about hardware reliability and software malfunctions are controlled by _____.
6. The supervisor is the interface between user programs and system resources. True or False

Solution 1.7.1 Check your answers against the answers given below. If you incorrectly answered any questions, review the preceding pages before you go on.

1. True
2. job management
3. True
4. control, storage, retrieval
5. recovery management
6. True

Multiprogramming

Multiprogramming provides increased productivity or throughput by making better use of the CPU and I/O resources. Multiprogramming is defined as the ability to run more than one job at a time. In doing so, the relatively fast CPU and the slower I/O are brought into balance to yield greater throughput.

For example, when a single executing job requests an I/O operation, it cannot continue processing until the I/O operation is complete. During that time, the CPU is in a "wait" state and is idle. With multiprogramming, the CPU is available for use by another job. Multiprogramming, therefore, allows the I/O operations of one job to be overlapped with the processing of another job. When a job has to wait for the completion of an I/O operation, the supervisor sets that JOB in a wait state and selects another job for execution (on the basis of its priority and "readiness" to run).

This supervisory process is called "task management". A "task" is defined as a unit of work for the CPU. The task is the basic multiprogramming unit. Programs (or parts of programs) that are executed concurrently are called tasks. In a nonmultiprogramming environment, there is only one unit of work to be performed at any time.

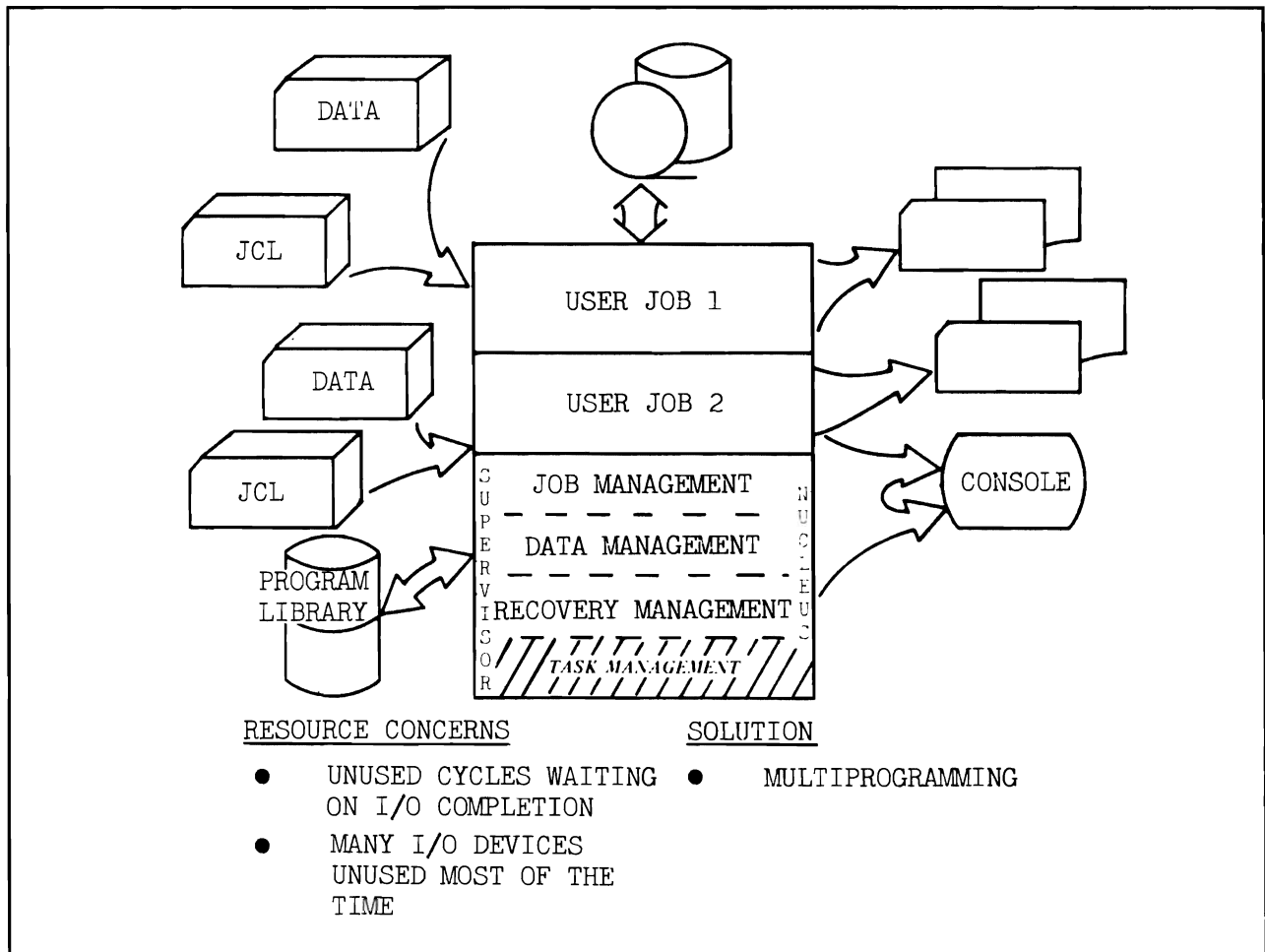


Figure 1.7.2 Resource Management Using Multiprogramming.

The task management facility gives us the ability to multiprogram. With multiprogramming, many concerns of resource usage are eliminated or minimized as shown in Figure 1.7.2.

Spooling

Multiprogramming provides the capability to run several programs at once but what about programs that read or punch cards and print reports. Most application programs require at least one of these unit record functions. Multiprogramming means that each job must have its "own" unit record I/O devices during execution. Additional unit record equipment to support many jobs is an expensive solution. The concept of "spooling" addresses this problem.

Spooling (Simultaneous Peripheral Operations On-Line) provides the ability to use ONE card reader, ONE printer and ONE card punch to satisfy the unit record I/O needs for MANY concurrently executing jobs. For a multiprogramming environment, card input for a job has to be available at any time for possible "reading" by the running job. So, all input (job control and data cards) for a job is spooled (written to DASD) before job execution begins.

Spooling is also used for output. One punch and printer for all executing jobs mean that print and punch output must be stored on DASD during job execution. The print or card output for the job can be printed or punched while other programs are executing.

Spooling improves the throughput of a computing system by separating unit record input and output operations from the internal computing. This results in overlapping the unit record I/O operations with the execution of application programs.

Summary

The implementation of an operating system effectively coordinates the use of all the resources available in the data processing environment. Job management brings jobs into the system and gets them started according to the job control language specification. Data management insures that the data required by the job is available when it is requested. Task management oversees and schedules the use of the CPU and the other hardware and software resources available in the data processing environment. The recovery management functions monitor the operation of the entire data processing system and if a malfunction occurs, provide facilities for correcting and servicing.

Multiprogramming and spooling provide additional capabilities for efficiently using both the processor and the I/O devices by overlapping their operations.

Exercise 1.7.2 The following quiz is an evaluation of your understanding of the preceding text on multiprogramming and spooling.

1. Multiprogramming provides increased productivity by making better use of the CPU and I/O resources; it is defined as _____.
 - a. Doing input and output operations at the same time.
 - b. The ability to run more than one job at a time.
 - c. Executing a task.
2. The increased productivity in multiprogramming is obtained by _____ the I/O operations of one job with the processing of another job.
 - a. separating
 - b. overlapping
3. The multiprogramming facilities are a function of _____.
 - a. the CPU
 - b. operator intervention
 - c. task management
 - d. job management
4. Spooling allows a data processing system to use _____ card reader(s), _____ card punch(es), _____ printer(s), to satisfy the unit record I/O needs for concurrently executing jobs.
 - a. one
 - b. many
5. When spooling is in use both input and output are _____, where it is available at any time for reading or writing by the system.
 - a. stored in memory
 - b. held in a program
 - c. written to a DASD
 - d. saved at the I/O device

Solution 1.7.2

1. b
2. b
3. c
4. a (one), a, a
5. c

Topic 8 - Data Communications

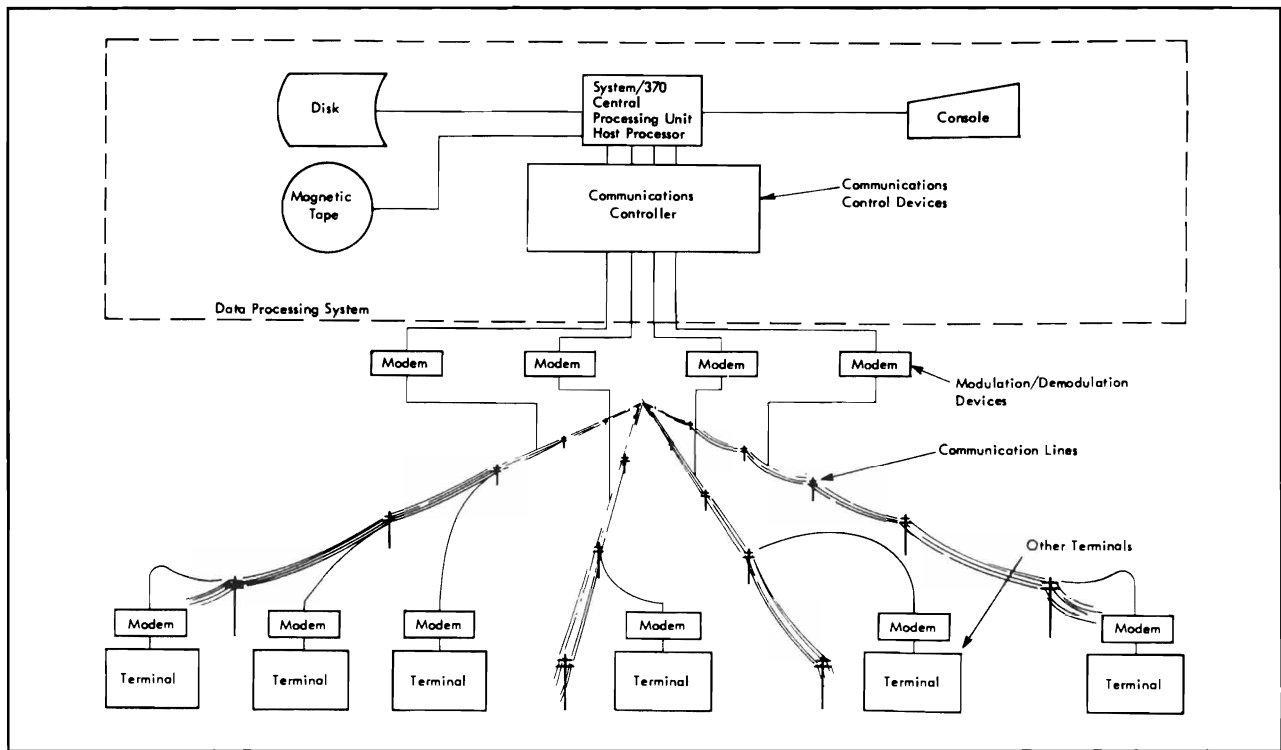


Figure 1.8.1 A Data Communication Network

Introduction

Data communications (teleprocessing) is the processing of data that is received from or sent to remote locations by way of communication facilities.

A data communication network consists of a number of communication lines (communication facilities) connecting a data processing system with remote data communication devices (Figure 1.8.1). Such devices can be terminals, control units, or other data processing systems. In this overview, any machine or group of machines capable of generating and/or receiving signals transmitted over communication lines will be referred to as a terminal or terminals. Thus, terminals may be data processing systems, communication systems such as the IBM 3270 Information Display System, or a single unit such as the IBM 2740 Communications Terminal.

As an example of how a data communication network functions, a clerk in an insurance company branch office receives a telephone call asking for information about an insured's account. Asking the caller to hold the line, the clerk enters the information request into a terminal, and the request is sent over a communication line to the IBM processor at the insurance company main office. When the request reaches the computer, several things happen. The computer interrupts processing whatever job it is working on and saves all necessary data and instructions so that it can resume processing the job at the exact point of interruption. As the information is received over the communication lines, the communication module in the control program converts the

data into machine language, stores it in a buffer area, and checks to see that it was transmitted correctly.

The nature of the request may dictate that a number of operations be performed. To process the request, the data communication program directs the IBM processor to examine the appropriate policy file and bring the insured's record from storage. The program then searches the record for the information requested and sends it over the communication lines to the clerk who originated the request. The clerk reads the information as it is displayed or typed out at the terminal and relays the information to the policy holder or adjuster waiting on the telephone.

At the main office, the control program has returned the processor to its status prior to the interruption, and the computer has resumed processing. As a result of the telephone inquiry, the clerk in the branch office may update the insured's record and transmit this information to the IBM processor at the main office at a later time.

Elements of a Data Communication Network

The elements of a data communication network (Figure 1.8.1) consist of a host processor, communication control devices, modulation/ demodulation devices (modems), communication lines, other terminals, and programming systems. Three of these elements, the communication control devices, modems, and communication line, constitute a DATA LINK.

Host Processor

IBM processors are able to serve as the host processor in a data communication network. Requirements for the host processor include multiprogramming capability, adequate storage capacity, and adequate speed for the applications required.

The host processor for a data communication network must be able to handle random and unscheduled input, as well as serialized and scheduled input.

Communication Control Devices

Communication control devices are hardware components that link the communication lines to the host processor. These devices can be external to the processor, such as the IBM 3704 or 3705 Communications Controller, or they can be a part of the processor, such as the communications adapter feature.

The transfer of data requires noninformation transmission for setting up, controlling, checking, and terminating information exchange. The noninformation exchanges constitute *data link control*. Communication control devices handle data link control; thus, functions of these devices include:

- Synchronization (getting the receiver in step with the sender)
- Identifying the sender and receiver
- Delimiting the beginning and ending of information (code translation)
- Error detection and recovery

In order for a host processor to send data over communication lines, the data must be converted (serialized) to a serial stream of binary digits. Likewise, when the host processor receives data from a remote terminal, this data must be reconverted (deserialized) into machine language for processing. Control devices perform this function.

Modulation/Demodulation Devices

After data which is to be transmitted is serialized by the control device, the binary signals must be converted to audio-frequency signals (modulated) for transmission over communication lines and reconverted (demodulated) at the other end. A modulation/demodulation device or *modem* performs this function. One modem is required at each end of a data link. Data sets and line adapters have the same functions as a modem. A modem can be an integral part of a control device or terminal, or it can be an external unit.

Communication Lines

Communication lines are classified according to:

- Configuration
- Transmission direction
- Type
- Transmission mode

Configuration

Two basic communication line configurations are

- Point-to-Point (connects two terminals).
- Multipoint (connects multiple terminals). In a multipoint configuration, one terminal must always be designated as the primary (control) terminal, and all others are secondary (tributary) terminals.

Transmission Direction

A communication line that transmits data in either direction, but not simultaneously, is called *half-duplex*. A line that transmits in both directions at the same time is *duplex* or *full duplex*.

Type

Basically, two types of communication lines are available: *switched* and *nonswitched*.

Switched lines (also called dial) connect terminals by means of common carrier exchange equipment. Dialing establishes a connection, which is maintained only while data is being transmitted. Switched lines are half-duplex only.

Nonswitched lines are available for use at any time, and dialing is not required to make a connection. Nonswitched lines may be either *leased* or *private* lines.

Leased lines are leased from a communication common carrier and are usually telephone or telegraph lines. Private lines are privately owned and may be supplied by the data communication equipment company. Duplex transmission requires leased or private lines (nonswitched).

Transmission Mode

Communication lines can transmit in *asynchronous mode* (also called serial start-stop mode) or *synchronous mode*.

Asynchronous transmission requires the use of start and stop bits to designate

the beginning and ending of transmission.

Synchronous transmission is transmission in which the receiving terminal on a communication line operates in step with the transmitting terminal through the recognition of a specific bit pattern (syn pattern) at the beginning of each transmission. Synchronous mode, therefore, eliminates the need for start and stop bits and permits continuous uninterrupted transmission, increasing transmission speed and reducing turnaround time.

Codes

A variety of codes can be used to represent data characters when transmitting over communication lines. Two of the most commonly used are ASCII (American National Standard Code for Information Interchange) and EBCDIC (Extended Binary Coded Decimal Interchange Code). ASCII offers 128 possible characters. EBCDIC has 256 possible code combinations, of which 17 are used for control purpose, 96 are used for text characters, and the remaining are unassigned.

Line Disciplines

A line discipline provides a set of rules for the orderly transfer of data from one location to another using communication facilities. Two line disciplines currently used are binary synchronous communications (BSC) and synchronous data link control (SDLC).

Terminals

The type of terminal used for handling data flow in a data communication network depends upon the application requirements. An application may or may not require printed output for example.

Terminals can be keyboard displays, keyboard printers, a communication system such as the IBM 3270 Information Display System, or a 4300 System. Terminals can prerecord data on diskettes or tape for batch transmission or mailing. Terminals can accommodate asynchronous (start-stop) and synchronous (BSC and SDLC) modes of transmission.

Data Management Support

Data management support for communication devices provides a link between the application program and remotely connected data communication terminals. This support is available from several access methods (only one is selected for implementation by the user). Two of the available access methods are the Virtual Telecommunications Access Method (VTAM) and the Basic Telecommunications Access Method (BTAM).

These access methods direct the transmission of data between the host processor application program and terminals. They provide the following basic capabilities:

- Receive messages
- Send messages
- Dial and answer (switched lines)
- Detect and correct errors
- Perform code translation

Stated simply the programmer can use either VTAM or BTAM to control terminal input/output operations.

*Data Communication
Applications*

The types of applications which utilize a data communication network are many and varied. Some of the most widely used applications include:

- Inquiry. The location and displaying of a selected data record from a data file on DASD.
- Data Entry. Entry of data from a remote terminal into a host processor via a communication link by a remote terminal.
- Record Update. Modification, deletion, or addition of data contained on existing data files stored at the host processor site via a communication link from a remote terminal.
- Remote Job Entry. Entry of jobs from a remote terminal to be executed at the host processor location via a communication link.
- Message Switching. The ability to relay a message from one remote terminal to more remote terminals via a host processor and a series of communication links.
- Time Sharing. The allocation of host processor resources so that many remote terminals may execute programs concurrently and may interact with the programs during execution.

This concludes the discussion of data communications. Do the review exercise that follows then continue in this text.



Exercise 1.8

1. The device which converts data to be transmitted over communication lines to audio frequency signals (modulation) and reconverts at the other end (demodulation) is called a _____.
2. The communication line configuration used to connect two terminals is called _____ to _____. The _____ communication line configuration is used to connect multiple terminals.
3. A communication line that transmits data in either direction, but not simultaneously is called _____. A line that transmits in both directions at the same time is called _____.
4. Communication lines can transmit in asynchronous mode or synchronous mode. Which transmission mode does not require the use of start and stop bits?
5. A line discipline provides a set of rules for orderly transfer of data from one location to another using communication facilities. Name two line disciplines.
6. Data management support for communication devices provides a link between the application program and the remotely connected communication terminal. Name two access methods that provide this support.

Solution 1.8

1. modem
2. point-to-point, multipoint
3. half-duplex, duplex (full duplex)
4. synchronous mode
5. Binary Synchronous Communications (BSC) and Synchronous Data Link Control (SDLC)
6. Virtual Telecommunications Access Method (VTAM) and Basic Telecommunications Access Method (BTAM)

Question #	Section Heading
1	Modulation/Demodulation Devices
2	Configuration
3	Transmission Direction
4	Transmission Mode
5	Line Disciplines
6	Data Management Support

Topic 9 - Virtual Storage Concepts

Introduction

IBM virtual storage management is a storage management technique which uses DASD, system software, and processor hardware facilities to allow the execution of programs that are larger than the available processor storage.

This topic provides a general description of IBM's implementation of virtual storage. The primary purpose is to familiarize the student with the functions of virtual storage and provide a general understanding of its implementation.

Note: It is common practice to use the letter *K* in conjunction with a decimal number as a "shorthand" method of representing a number that is 1,024 times the decimal number. For example, 2K bytes is used to mean $2 \times 1024 = 2048$ bytes. This practice will be used throughout this text.

Why Virtual Storage?

In order to be able to better understand the concepts of virtual storage, consider some of the problems faced by the user of a data processing system without virtual storage.

One of the constraints of a non-virtual system is that the user must write programs that fit into the available processor storage. If the program is too large, it must be re-written to fit the available storage. There are several techniques a programmer can use to solve this problem, but all involve additional time and effort. Another problem that is often experienced in data processing systems without virtual storage is storage fragmentation. *Storage fragmentation* is a condition where the available processor storage is divided (fragmented) into several areas separated by executing programs.

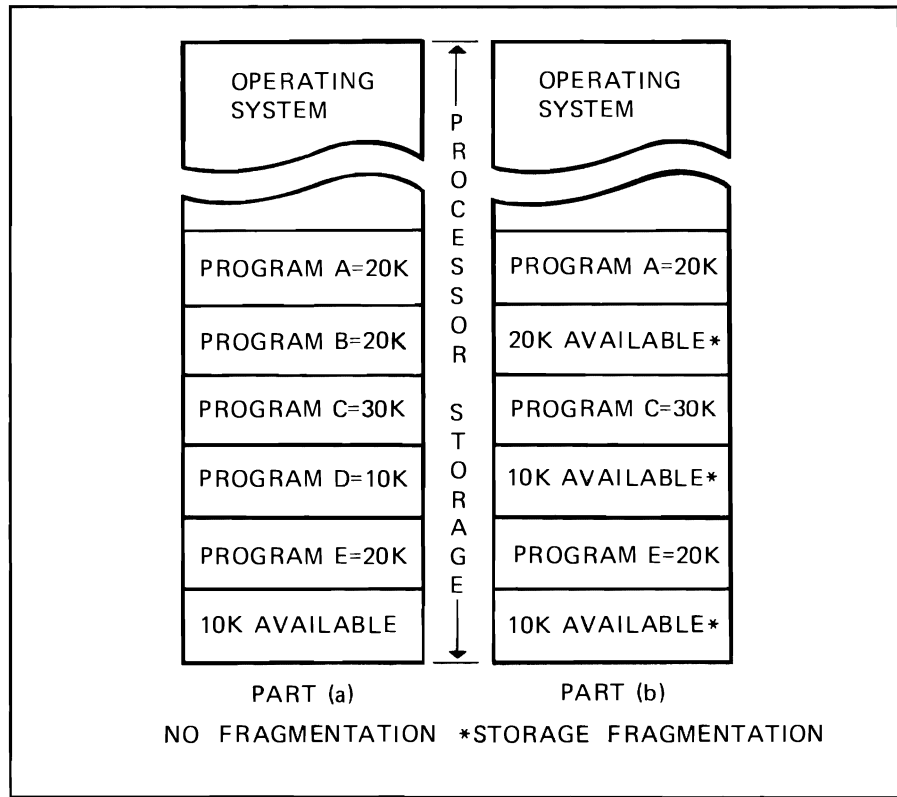


Figure 1.9.1 Storage Fragmentation

Figure 1.9.1 illustrates how fragmentation can occur. In part (a) of this illustration, programs A, B, C, D and E are in storage and there is no storage fragmentation. In part (b), programs B and D have completed execution. The storage that they occupied is now available to run other programs. There are now three non-contiguous areas of storage available. This condition is referred to as storage fragmentation.

The problem with fragmented storage is that in order to load a program into processor storage, the program must fit into a contiguous storage area. Therefore, in our example if we are ready to run a 30K program we must wait until a contiguous storage area of at least 30K becomes available. This is true even though the total available storage is 40K.

Virtual storage addresses these and other problems.

IBM Virtual Storage Implementation

In the IBM implementation of virtual storage, all programs are automatically divided into 2K byte sections called *pages*. In addition, processor storage is divided into 2k byte sections called *page frames*. Figure 1.9.2 illustrates a program divided into pages and real storage (processor storage) divided into page frames.

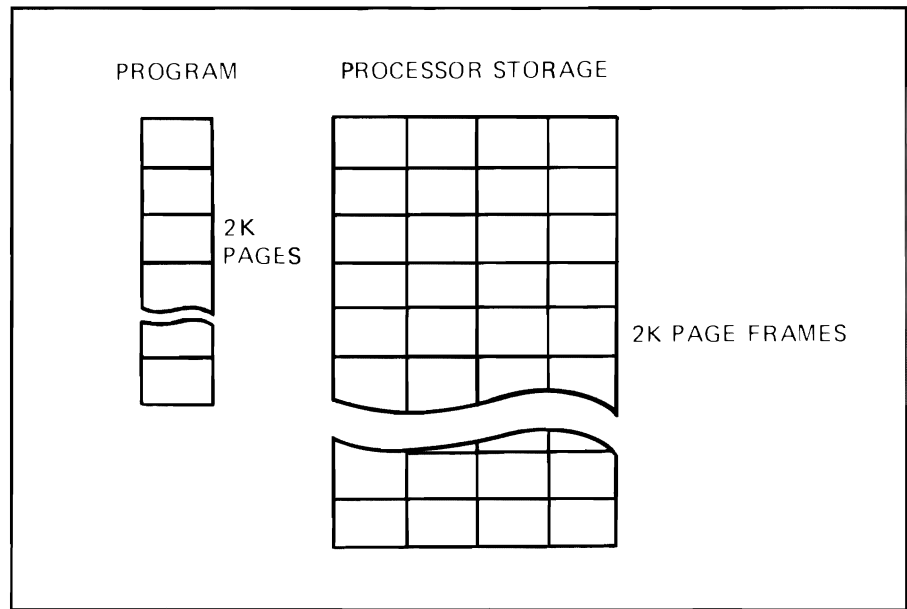


Figure 1.9.2 Program Pages and Real Storage Page Frames

One of the key concepts of this implementation is the fact that the program pages that are involved in the processing of the current instruction are the only program pages that must occupy page frames in real storage. The remaining pages may occupy other page frames if sufficient page frames are available. If there are an insufficient number of page frames, the system will automatically write the remaining programs pages to a disk file called the *page data set*. Figure 1.9.3 shows pages from the program named **PROGX** being placed into real storage page frames. When there are not enough page frames to accommodate all of **PROGX**, the system stores the contents of some page frames on the page data set. The remaining pages of **PROGX** can then be placed into the available page frames.

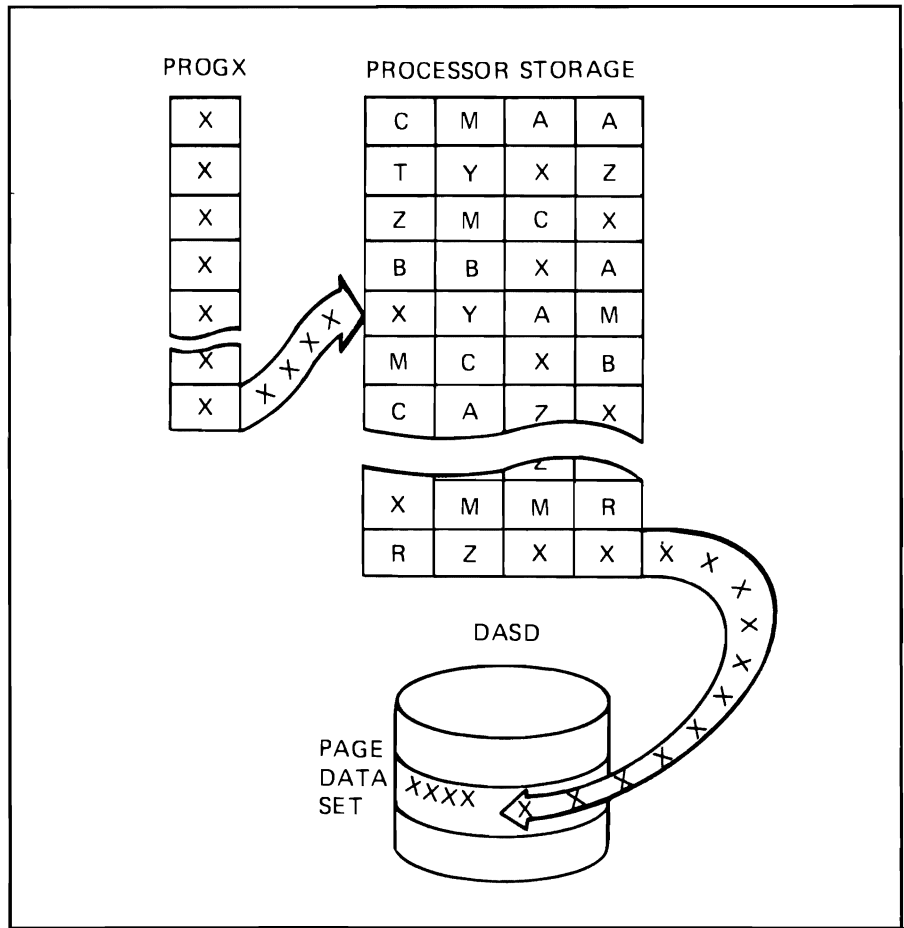


Figure 1.9.3 PAGE DATA SET

If, during program execution, a reference is made to an instruction or data that is not currently located in real storage, the system will automatically determine the location of the required program page on the page data set, retrieve the page and place it into an available page frame. This process is called a *page in*. Figure 1.9.4 illustrates a page from PROGX being paged into an available page frame in real storage.

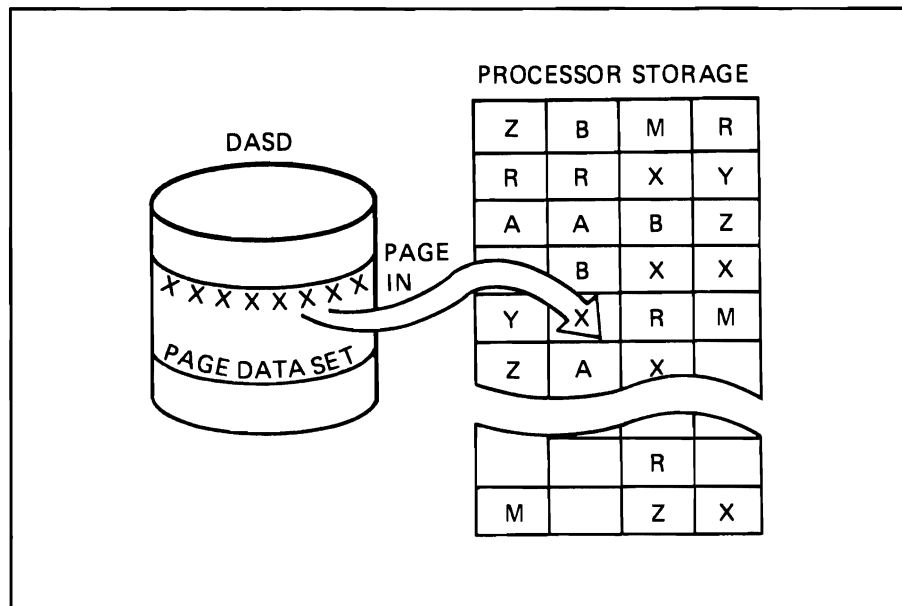


Figure 1.9.4 Page In

If no page frame is available in real storage to accommodate a program page needed for continued execution of a program the system automatically frees a page frame. To do this, the system selects a program page not recently referenced and writes it to the page data set if necessary. This operation (also called *page-out*) is necessary if the page was altered while in real storage or if it had not been written to the page data set previously.

When a program page is brought into real storage, it is placed into an available page frame which is not necessarily adjacent to other page frames used for that program. Therefore, we can picture the pages of a program as being randomly scattered throughout real storage along with the pages from other programs (Figure 1.9.5).

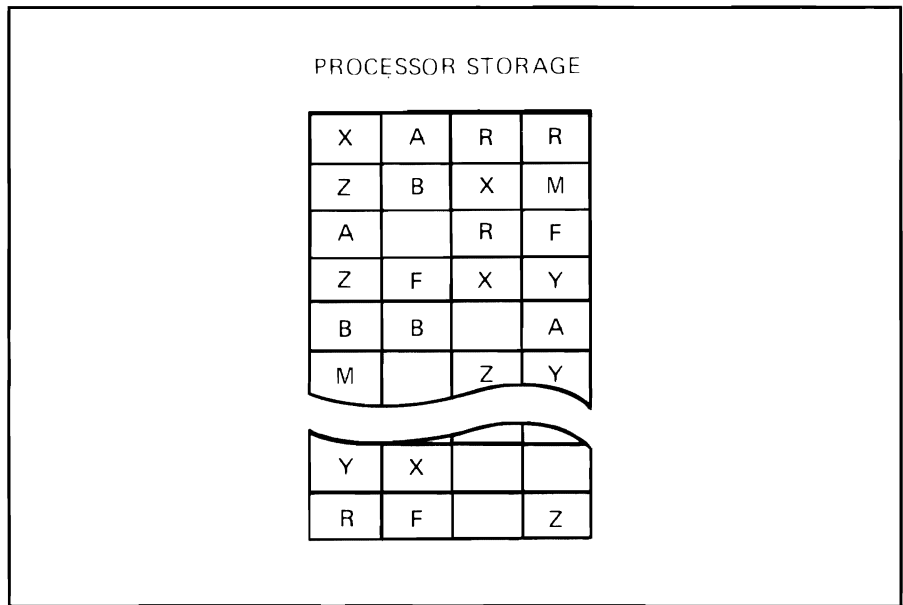


Figure 1.9.5 Program Pages In Real Storage

Since the program's pages are not necessarily contiguous in real storage, the system must provide a way to locate the instructions and data contained in these pages when they are referenced. In addition, the program addresses used to locate instructions and data will often reference storage locations that are higher than the locations of real storage. The system must be able to translate the address references in the program (referred to as *virtual addresses*) to real storage locations (addresses). This is accomplished by a high speed hardware feature called *Dynamic Address Translation (DAT)*. DAT is used in conjunction with information maintained by the operating system to provide the capability to translate virtual addresses as high as 16 million bytes.

Users of DOS/VSE with 4300 processors may select the extended control program support mode (ECPS:VSE). ECPS:VSE mode provides an improved virtual address translation capability that does not use the DAT feature. The DAT feature is used on 4300 processors when operating in the System/370 mode.

Summary

Virtual storage paging provides the capability to execute a program without having to have the entire program in real storage. In addition, it translates virtual storage addresses in the program to addresses in real storage. These capabilities allow the programmer to write programs as large as 16 megabytes.

With dynamic address translation, the pages of a program do not need to be contiguous in real storage. Therefore, storage fragmentation is not a problem in an IBM virtual storage system.

In general, Virtual Storage organization is meant to be "transparent" to the user which means you act and write programs as if you had much more storage available than you really do and let the system handle the implementation.

Exercise 1.9

1. In a virtual storage environment programs are sectioned into _____ divisions called _____.
 - A. 2K, Slots
 - B. 3K, partitions
 - C. 16 million byte, partitions
 - D. 2K, pages
2. Real processor storage is also divided into _____ sections, these are called _____.
 - A. 16 million byte, partitions
 - B. 2K, pages
 - C. partitions, frames
 - D. 2K, page frames
3. When there are insufficient page frames to contain the pages of a program the remaining pages are written to an area on DASD called the _____.
4. The maximum amount of virtual storage available in IBM's implementation is _____.
5. When the system must perform a page out in order to acquire a page frame for a page in, it selects a page that _____ been recently referenced during program execution.
 - A. has
 - B. has not
6. What is the hardware feature that is used to translate virtual addresses in the program to real storage addresses?

Solution 1.9 Check you answers against the answers given below. Review any portions of this topic pertaining to questions answered incorrectly.

1. D
2. D
3. page data set
4. 16 million bytes
5. B
6. dynamic address translation(DAT)

Unit 2:

Hardware Overview

In this unit we will discuss the architecture and features of the 4300 processors and key input/output devices associated with 4300 data processing systems. The input/output devices we will discuss are specific examples of device types described in Unit I such as direct access devices, magnetic tape units, terminals, printers, card readers and card punches.

The student should not try to memorize all the details presented in this unit. The primary purpose is to make the student aware of the available functions and configuration requirements. When these hardware devices are referenced in the study units which follow, the student may wish to use this study unit as a reference.

Objectives

Upon completion of this unit the student should be able to:

- Describe the significant features of 4300 processor architecture
- Describe the two operational modes of 4300 processors
- Describe the significant features of the 4331 processor
- Describe the significant features of the 4341 processor
- Describe significant features of the primary input-output devices associated with 4300 data processing systems

Study time

60 to 90 Minutes

Topic 1 4300 processors

Introduction to 4300 processors

The 4300 processors are general purpose processors which incorporate an integrated technology and new design concept which reduces the power, cooling, and floor space requirements while providing data processing solutions applicable to many application environments. An example of a 4300 installation is shown in figure 2.1.1

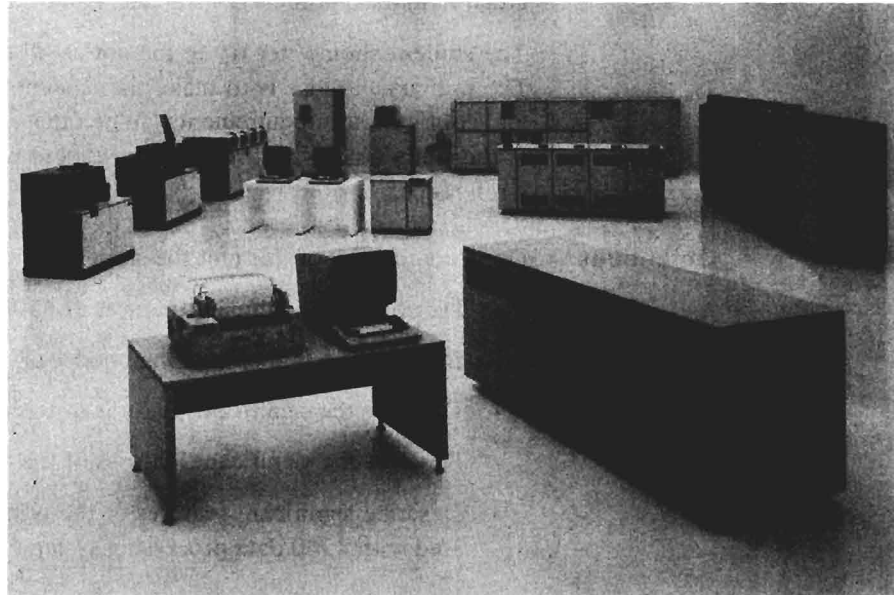


Figure 2.1.1 4300 processor Installation

Operational modes of 4300 processors

The processors operate in either of two modes selectable by the system operator. The modes are *Extended Control Program Support : Virtual Storage Extended (ECPS:VSE)* mode and *System/370 (S/370)* mode.

ECPS:VSE Mode

ECPS:VSE is an operational mode of 4300 processors which uses an address translation mechanism different from that employed by System/370 for virtual storage. When ECPS:VSE mode is chosen, there is a single address space with a maximum size of sixteen million bytes of virtual storage. All storage references by a user program as well as a channel program are made to virtual storage locations. Use of this mode results in faster address translation of virtual storage references.

ECPS:VSE Storage Addressing

In ECPS:VSE mode translation to the actual processor storage address is provided by ECPS:VSE using a Machine Storage Directory (MSD). The MSD is a table containing one entry for each 2K virtual storage page. Each entry describes whether a page is paged in or paged out of processor storage. If the page is paged in, the entry includes the processor storage location.

System/370 Mode

For compatibility with previous IBM processors, System/370 mode with its Dynamic Address Translation (DAT) function is available on 4300 processors. System/370 mode requires both a real and a virtual address space. The real address space is equal to the processor storage size and the virtual address space size is user determined.

System/370 Storage Addressing

In this mode the DAT hardware together with the software facilities of the virtual operating system are used to translate virtual addresses to the real storage addresses for user program and operating system references. However channel programs in System/370 mode still reference real storage locations as in System/370.

Error detection and maintenance**Error detection**

Instruction retry, limited error recording and error checking and correction (ECC) in processor storage are standard on 4300 processors.

Maintenance

A new concept in maintenance is provided with the 4300 processors. A separately powered subsystem, the **Support Processor**, has environmental monitoring and recording capabilities to assist in the recognition and resolution of environmentally caused problems, such as power fluctuations and temperature variances. This subsystem also provides support functions for the operation of directly attached I/O devices and for the Remote Support Facility.

Remote Support Facility (RSF)

The RSF is a customer option which enhances hardware maintenance on the 4331 and 4341 processors. The RSF allows IBM field technical support center specialists to remotely monitor and/or control problem diagnosis on the processor. Monitoring is implemented, after customer authorization, via a customer provided telephone line to an IBM field technical support center. While the 4300 system is operating in this maintenance status, all remote console screen activity can be observed at the customer's display terminal.

Compatibility

The 4300 processors have implemented the same instruction set as System/370 processors thereby making most System/370 programs compatible with the 4300.

Either the OS/VS1, DOS/VS, DOS/VSE, or Virtual Machine/370 operating system may be used with the 4300 in System/370 mode.

The DOS/VSE operating system is required in ECPS:VSE mode.

Exercise 2.1.1 Answer the following questions on the 4300 processors before you continue with the 4331 and 4341 overviews.

1. (True or False) Four significant characteristics of the 4300 processors are:

System/370 Compatibility

Two operator selectable modes of operation

The Support Processor Maintenance concept

Remote Support Facility Option.

2. The two available operational modes for the 4331 and the 4341 are _____ and _____.
3. Which 4300 mode of operation uses the DAT hardware and software to translate from the separate virtual address locations to the real address space location?
4. (True or False) The virtual storage address translation mechanism introduced with the 4300 processors is based on a single address space of up to 16 million bytes.
5. (True or False) Most System/370 user programs are compatible with the 4300 processor environment.

Solution 2.1.1 Check your answers against the answers listed here. Review the preceding topics for references to questions answered incorrectly.

1. True
2. ECPS:VSE mode, System/370 mode
3. System/370 mode
4. True
5. True

4331 Overview The 4331 processor, pictured in Figure 2.1.2, is a System/370 compatible processor which offers a full range of data processing solutions.



Figure 2.1.2 IBM 4331 processor with Input/Output Devices

Both operational modes, ECPS:VSE and System/370, are available with the 4331.

Programming Support

When ECPS:VSE mode is selected the DOS/VSE operating system is required. In System/370 mode the 4331 processor is supported by the DOS, DOS/VS, DOS/VSE, OS/VS1, VM/370 operating systems.

When System/370 mode is selected the optional ECPS:VM/370 feature may be used to provide improved performance to virtual storage operating systems operating under VM/370.

The 4331 has a 1401/1440/1460 compatibility feature which in conjunction with the new 1401 Emulator program product permits execution of 1401, 1440, or 1460 programs by the processor.

Processor Storage

Virtual storage capability is standard and the processor storage capacity depends on the 4331 model. The processor storage capacity is 524,288 bytes (1/2 megabyte) or 1,048,576 bytes (1 megabyte).

I/O Device Attachment

I/O devices are attached to the 4331 through integrated I/O adapters and channels. The adapters assist in reducing floor space requirements when using selected devices by eliminating the need for separate control units.

DASD Adapter

The DASD adapter allows the direct attachment of a variety of disk devices. Up to four strings of disks may be attached to this adapter. Devices which may be attached include 3310, 3370, 3340, and 3344.

8809 Magnetic Tape Unit Adapter

The magnetic tape unit adapter allows the direct attachment of up to six 8809 Magnetic Tape Units.

Communications Adapter

The Communications Adapter (CA) allows the direct attachment of any combination of up to eight teleprocessing lines using start/stop, synchronous data link control, or binary synchronous line control.

5424 Adapter

The 5424 Adapter is used to attach the 5424 Multi-function Card Unit for the 96 column card user.

Display/Printer Adapter

The display/printer adapter controls the required 3278-2A operator display console. The 3278-2A console provides a communication interface between the system operator and the operating system. For maintenance and service support the console can display and store the status of the 4331 processor system and provide a means for using diagnostic tools.

The display/printer adapter can also control up to seven additional displays or printers. When the display/printer adapter expansion feature is installed, a maximum of fifteen devices in addition to the system console is allowed.

Channels

One block multiplexer channel and/or one byte multiplexer channel are optionally available.

The block multiplexer channel allows attachment of up to eight control units for operation of S/370 devices with a maximum data transfer rate of 500 Kilobytes (KB) per second, such as the 3411 Magnetic Tape and Control unit.

The Byte Multiplexor channel provides attachment for up to eight control units for unit record devices like the 3203 model 5 printer.

Exercise 2.1.2 Answer the following questions to review the preceding topics.

1. The 4331 features for attachment of the 3310, 8809, 5424, 3278-2A, and the various communication devices are:
 - a. _____
 - b. _____
 - c. _____
 - d. _____
 - e. _____
2. (True or False) Virtual storage capability is an optional feature on the 4331.
3. The performance of operating systems executing under the Virtual Machine/370 facility may be improved by using the 4331 optional feature called _____.
4. (True or False) Both byte and block multiplexer channels are available on the 4331 processor.
5. Processor storage capacity in the 4331 is either _____ megabyte(s) or _____ megabyte(s).
6. The DOS/VSE operating system is required in _____ operating mode.

Solution 2.1.2 Check your answers against the answers given here. Review the preceding topics for references to questions answered incorrectly.

1.
 - a) DASD adapter
 - b) 8809 Magnetic Tape Unit adapter
 - c) 5424 Multi-function Card Unit adapter
 - d) Display/Printer adapter
 - e) Communication adapter
2. False, virtual storage capability is a standard feature of the 4331
3. ECPS:VM/370
4. True
5. 1/2, one
6. ECPS:VSE

4341 Overview

The 4341 processors (Figure 2.1.3) provide System/370 compatibility with growth opportunities for new users of intermediate systems and users in a distributed processing environment with high performance and large capacity requirements.

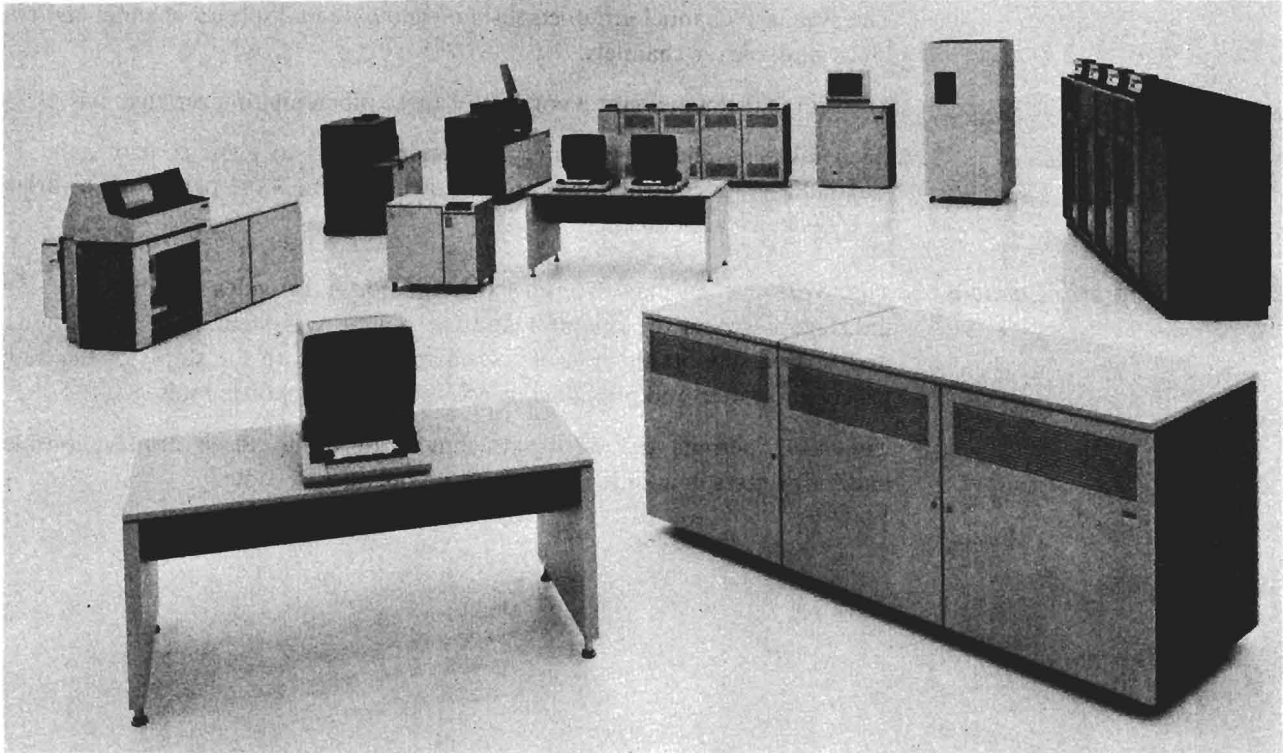


Figure 2.1.3 IBM 4341 processor with Input/Output Devices

Both the ECPS:VSE mode and System/370 mode are provided with the 4341 processor.

Programming Support

The DOS/VSE operating system is required when the 4341 is operated in ECPS:VSE mode. In System/370 mode the 4341 processor is supported by the DOS/VS, DOS/VSE, OS/VS1, and VM/370 operating systems.

When the OS/VS1 operating system is in control, ECPS:VS1 performance option provides a hardware assist that reduces the processor time to execute certain frequently used supervisor functions.

The ECPS:VM/370 feature is available to reduce the processor time needed to execute certain frequently used supervisor functions of VM/370.

Processor Storage

Virtual storage capability is standard with the 4341. Processor storage is available in either 2,097,152 bytes (2 megabytes) or 4,194,304 bytes (4 megabytes).

An 8K high speed buffer storage (CACHE) is standard. The CACHE memory provides a significant performance increase by reducing the processing time required for repetitive storage references.

I/O Device Attachment

Channels are available for attachment of all control units and I/O devices as the 4341 has no integrated adapters.

Channels

Six channels, three standard and three optional, are available.

The standard channel group consists of one byte multiplexer channel and two block multiplexer channels.

The optional channel group consists of three block multiplexer channels.

Each installed channel permits attachment of up to eight control units. A maximum of 48 control units can be attached to the 4341 permitting a variety of I/O device attachments.

Operator Console

The 3278-2A Display Console is required, it provides a communication interface between the operator and the operating system. For maintenance and service support the console can display and store the status of the 4341 processor system and provide a means for using diagnostic tools.

The 4341 permits the direct attachment of up to three display consoles and/or printers in addition to the system operator console.

Exercise 2.1.3 Answer the following questions to review the preceding topics.

1. (True or False) Virtual storage capability is a standard feature of the 4341 processors.
2. Machine storage capacity in the 4341 is either _____ megabytes or _____ megabytes.
3. A high speed buffer storage or _____ memory is provided to enhance 4341 performance.
4. (True or False) Two features are available to reduce the processor time required to execute certain supervisor functions, they are ECPS:VS1 and ECPS:VM/370.
5. (True or False) The DASD devices may be attached to the 4341 through the integrated DASD adapters.
6. (True or False) The 4341 permits the attachment of up to three display and/or printer units in addition to the 3278-2A console.

Solution 2.1.3 Check your answers against the answers given here. Review the topic if you missed more than one question.

1. True
2. two, four
3. cache
4. True
5. False. There are *no dasd adapters* on the 4341.
6. True

Topic 2. Input-Output Devices

Introduction

In this section we will discuss characteristics of some key devices which may be attached to 4300 processors.

Figure 2.2.1 shows many of the I/O devices which can be attached to a 4300 processor. The general characteristics of these devices were covered in Unit 1.



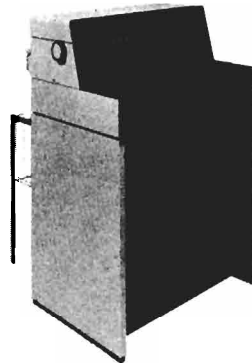
3278 Display Station



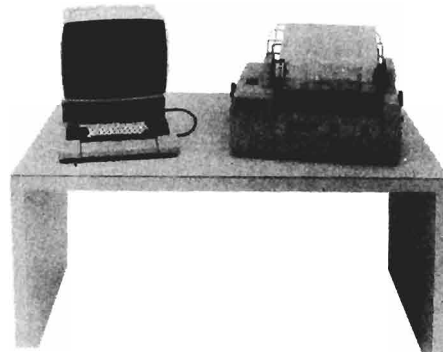
3289 Line Printer



3274 Control Unit



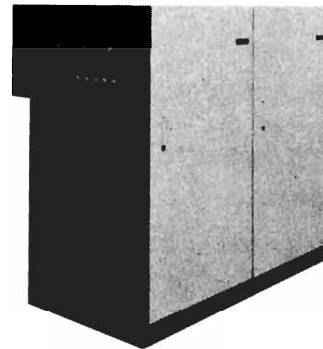
3288 Line Printer



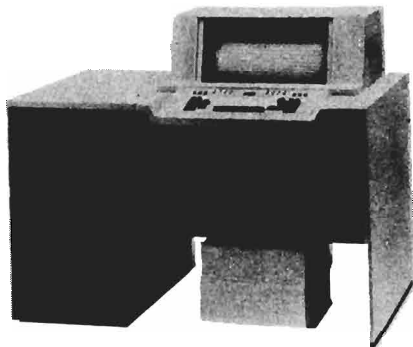
3276 Control Unit Display Station with
3287 Printer



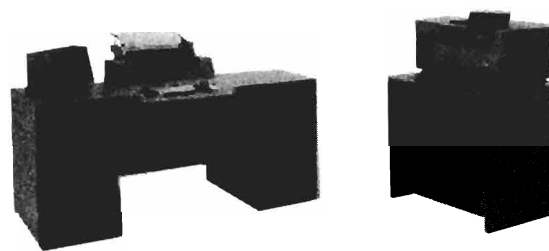
3262 Line Printer



3705 Communications Controller



3776 Communication Terminal



3774 Communication Terminal

Figure 2.2.1 4300 Compatible I/O Devices

Additional information on all the devices compatible with 4300 Systems is available in the publication, IBM 4300 processors Summary and Input/Output Data Communications Configurator (GA33-1523).

The devices in this section can be attached to 4300 processors through integrated adapters or through channels via external control units.

A data storage technique called count, key, and data (CKD), is used to maintain records on some DASD device. This technique is based on records being grouped in blocks of user determined size for disk storage. To select the most efficient blocksize for data storage on CKD devices, the user must be aware of the disk cylinder and track capacity of the device being used.

The 3310 and 3370, DASD devices, highlight the variety of storage devices available with the 4300 processors. These new devices utilize the Fixed Block Architecture (FBA) supported by the 4300 processors.

By utilizing the new fixed block architecture, files are stored in a consistent format on all DASD devices. They are then device independent, thus allowing greater flexibility and simplifying job control language. In the FBA technique all data is stored in fixed blocks of 512 bytes. The fixed block concept, by allowing specification of DASD space in groups of blocks, frees the user from concerns with the physical structure of the DASD device.

Direct Access Storage Devices

The Direct Access Storage Device (DASD) consists of the disk drive and a recording medium: a removable disk pack, data module, non removable disk, or diskette.

The use of a DASD device requires one or more units to perform the control functions for the DASD. The control function may be either integrated with the processor as the DASD Adapter is on the 4331 or external to the processor such as the 3830 and 3880 Storage Control Facility.

3880 Storage Control Facility

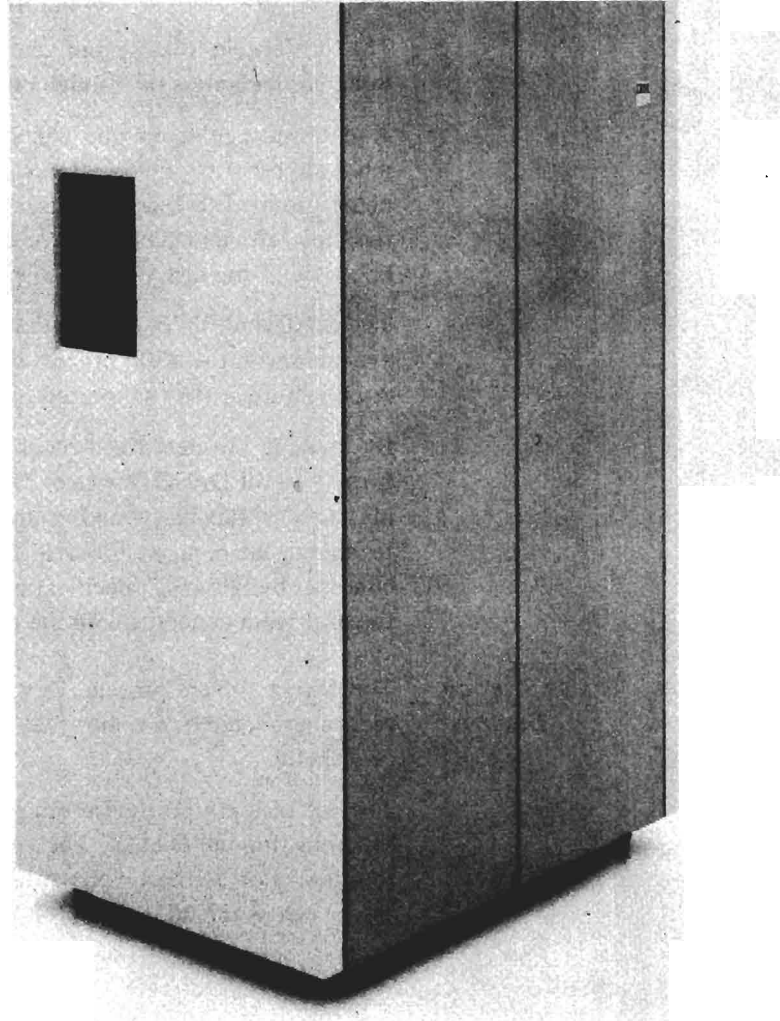


Figure 2.2.2 3880 Storage Control Facility

The 3880 Storage Control facility provides two independent data paths and the control functions for attachment of the Fixed Block Architecture devices or certain Count Key Data devices to the 4341 processor.

3830 Storage Control Unit

The 3830 Storage Control unit, provides a single data path for CKD DASD attachment to the 4341 processor.

3340 Disk Storage Facility

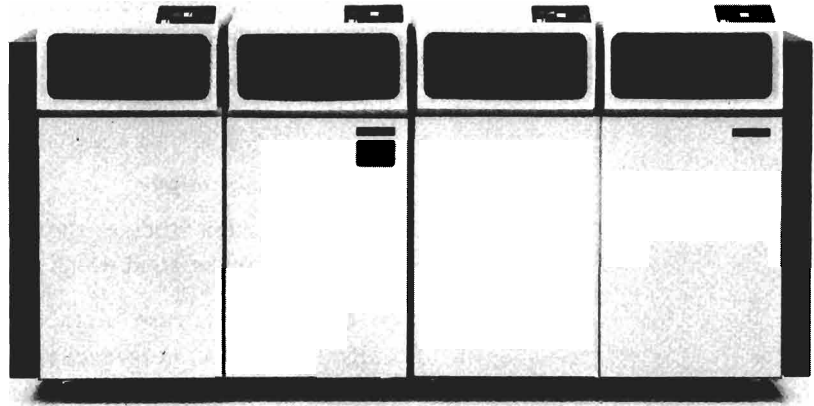


Figure 2.2.3 IBM 3340/3344 Direct Access Storage Facility

The 3340 Direct Access Storage Facility together with the 3344 Direct Access storage (see figure 2.2.3) offers large capacity, high performance, and data reliability.

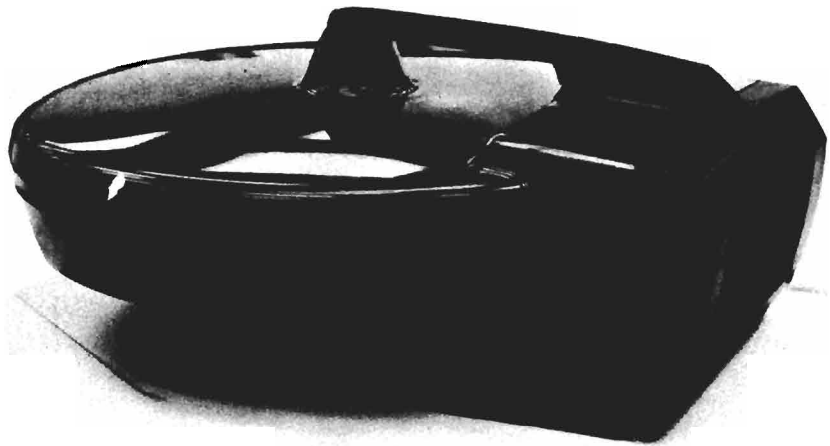


Figure 2.2.4 IBM 3348 Data Module

The sealed disk concept illustrated by the 3340 data module has the following advantages:

- Preventive maintenance of the heads, disks, and spindle is eliminated by reducing the exposure to outside contamination.
- Reliability is improved by dedicated read/write heads. Each head reads only the data it previously wrote.

In the 3340 the recording medium (see figure 2.2.4) is the *removable* sealed 3348 data module, available in both thirty-five million byte and seventy

million byte capacity. The 3344 uses a fixed or *non-removable* sealed disk with a two hundred eighty million byte capacity.

The 3340 attaches to the 4341 through an IBM 3880 Storage Control Facility. The 3340 must attach via the 4331 DASD Adapter when used on a 4331. The 3344 is attached to the 4331 and 4341 processors through the 3340 Model A2.

3350 Direct Access Storage

The 3350 direct access storage, shown in figure 2.2.5, is a large capacity, faster access DASD subsystem which uses a non-removable sealed disk.

With the continued increase in the use of data communications and the significantly larger disk storage capacities available, a growing number of installations rarely change many of their disk packs. For all practical purposes, once they mount a disk pack containing application data, it remains online for as long as they plan to use it. The 3350 is ideally suited to this purpose.

Each 3350 drive provides storage for a maximum of 317.5 million bytes of data.

Either the 3830 storage control unit or the 3880 Storage Control facility is used to attach the 3350 to a 4341 processor.

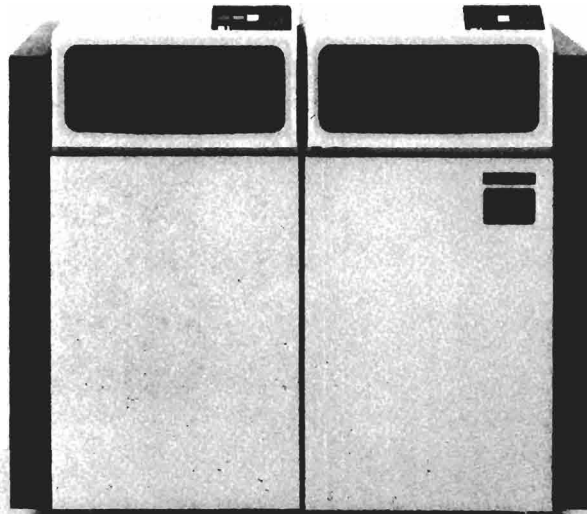


Figure 2.2.5 IBM 3350 Direct Access Storage Device

3310 DASD

The 3310 direct access storage device is a low cost intermediate capacity non-removable disk device. It offers the advantages of faster data transfer, lower cost, and improved reliability over the 3340 direct access storage. The device is specifically designed to use the new fixed block architecture (FBA) on the 4331.

Each 3310 unit consists of one or two drives with a capacity of 64.5 megabytes (MB) per drive. A maximum of 16 devices can be attached to the 4331.

There are features available on the 4331 which allow 2311/2314/2319 devices to be emulated on the 3310, thus easing the transition and avoiding the need for total file conversion.

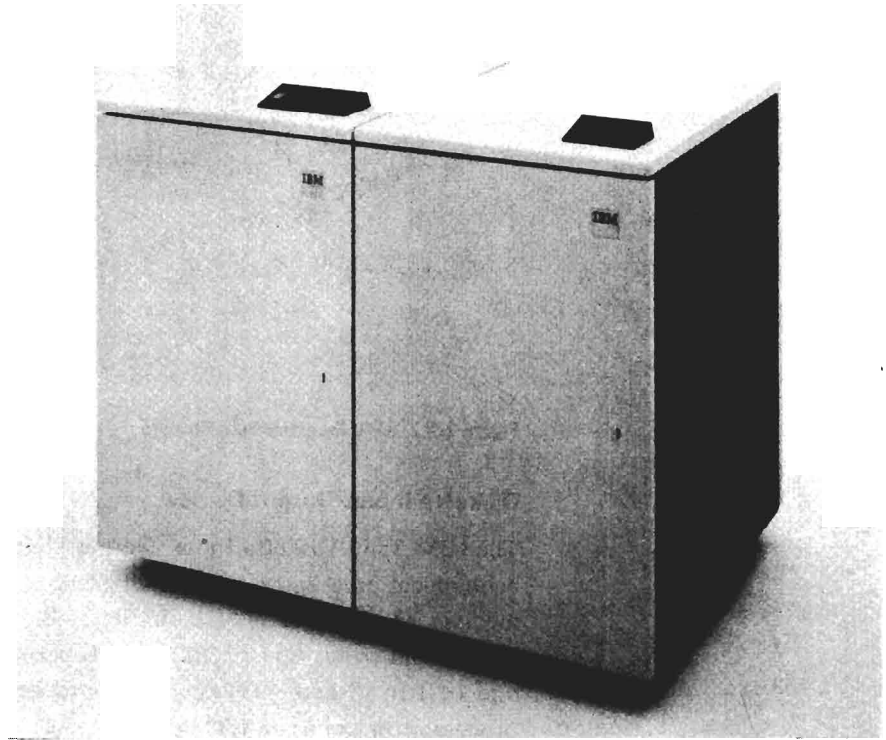


Figure 2.2.6 3310 Direct Access Storage

3370 DASD

The 3370 DASD (shown in figure 2.2.7), will provide large capacity storage using the new Fixed Block Architecture. The 3370 is also a high density fixed disk device.

Each 3370 Direct Access Storage unit has a single 571.3 MB spindle of disk cylinders accessed by two independent, movable access mechanisms (actuators). Positioning of either actuator may be overlapped with reading and/or writing of the other actuator. Each actuator accesses one half the data area of the spindle.

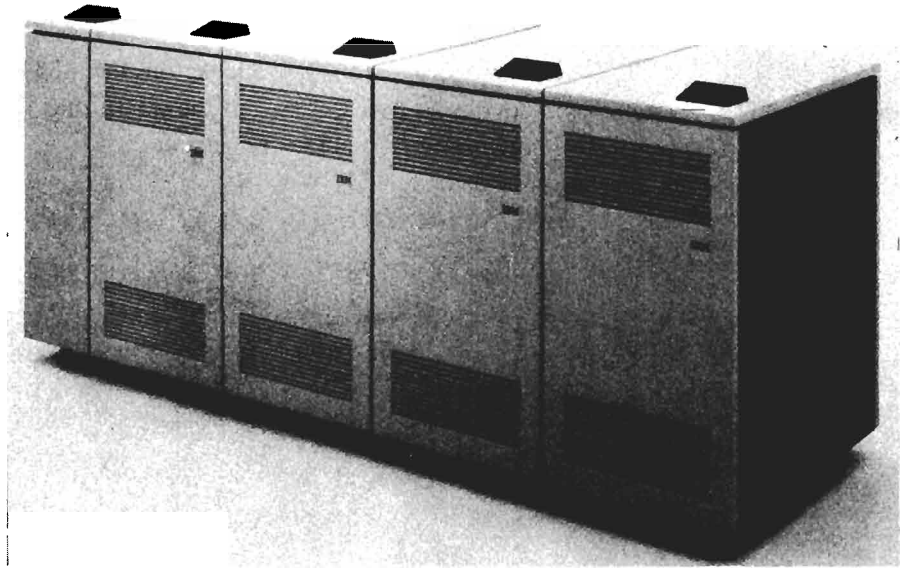


Figure 2.2.7 3370 Direct Access Storage

Diskette Input/Output Device

The IBM 3540 Diskette Input/Output Unit (Figure 2.2.8) is an efficient and economical data entry and output device. Using the diskette as a recording medium (Figure 2.2.9), the 3540 reads up to 3,600 diskette records per minute and writes up to 2,200 records per minute. The 3540 is available with one or two diskette drives and is attached to the channel of either 4300 processor for use as an I/O device.

The 4331 also has a Diskette Drive feature which provides a single diskette reader recorder. This diskette feature can read data created by certain other IBM diskette systems.

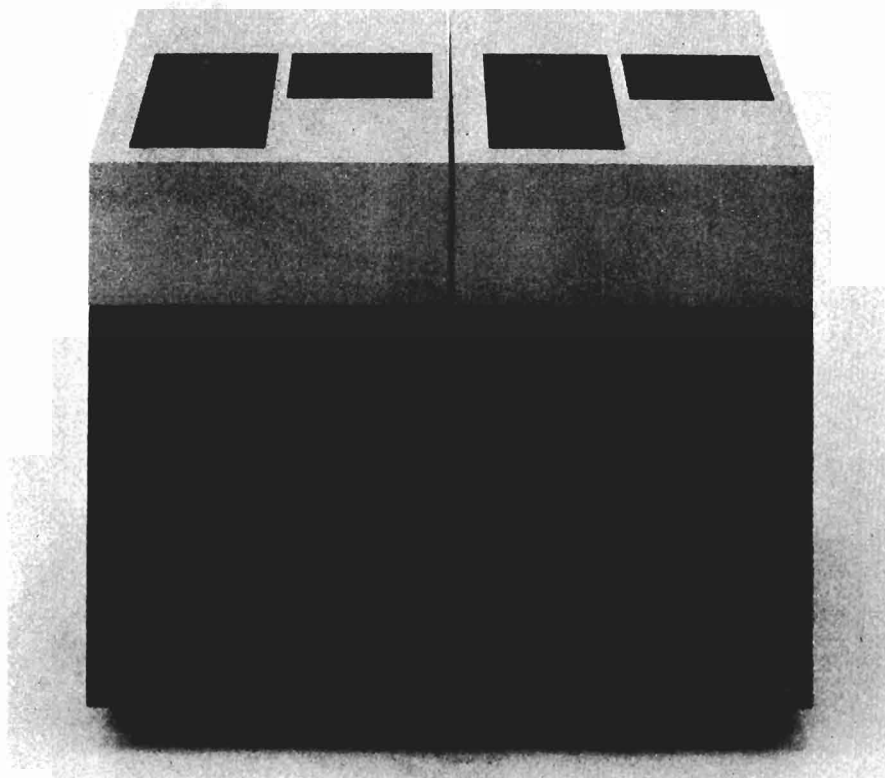


Figure 2.2.8 IBM 3540 Diskette Input/Output Unit

The diskette is a single magnetic disk, sealed in a plastic jacket. It weighs less than 2 ounces, is reusable, and may be used with other IBM diskette units. The diskette can store as many as 242,880 characters, equivalent to as many characters as can be put on 3,036 fully punched 80 column cards.

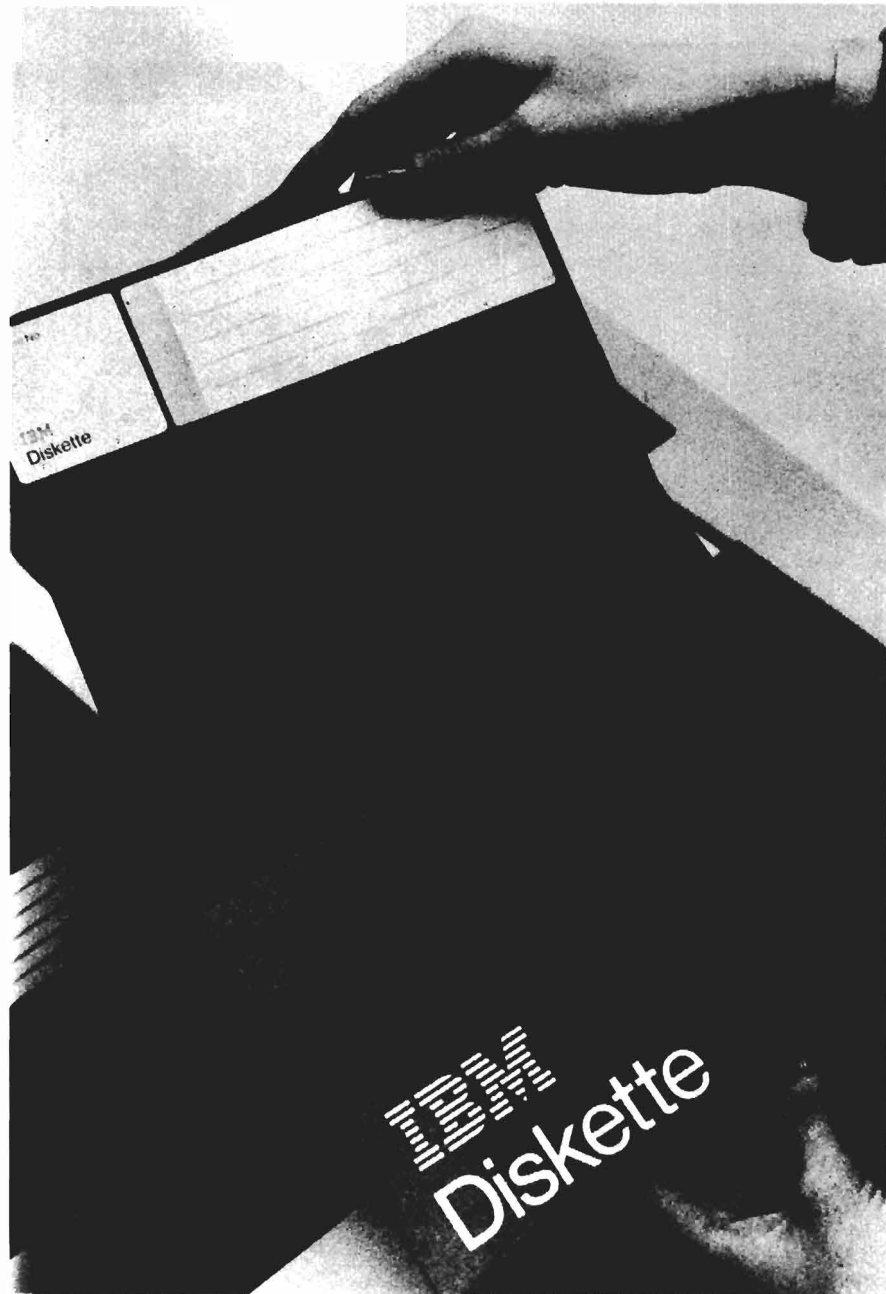


Figure 2.2.9 Diskette

Printing Devices

IBM printing devices provide a permanent visual record of data from the computer system.

3203 Printer Model 5

The IBM 3203 Printer Model 5 (Figure 2.2.10) offers medium speed, quiet operation, and high-quality printing. It contains an integrated control unit for attachment to a 4300 Processor channel.

The model 5 prints at a rate of 1200 lines per minute.

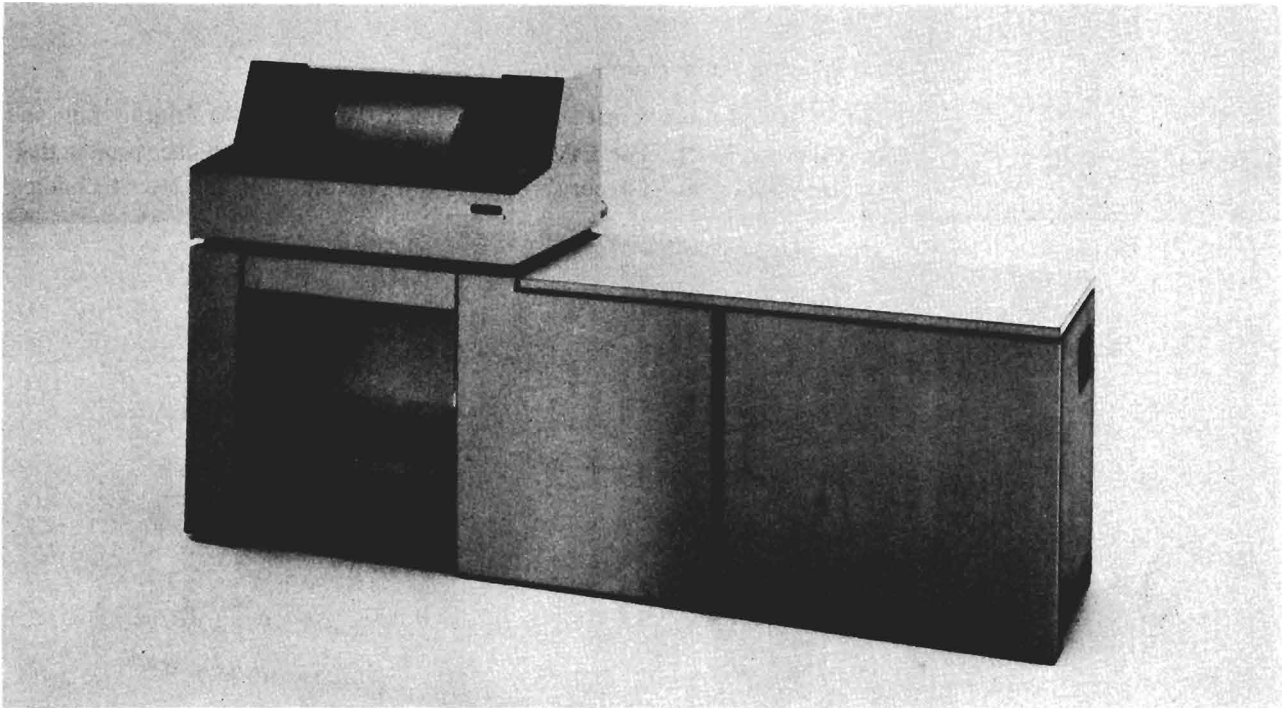


Figure 2.2.10 IBM 3203 Printer

3262 Line Printer

The IBM 3262 line printer is designed to provide quality printed output for the 4331 processor system. Maximum printing speed is 650 lines per minute with a 132 print position line. A maximum of two 3262 printers can be attached to the 4331 Display/Printer Adapter.

3211 Printer 3811 Printer Control Unit

The IBM 3211 Printer (Figure 2.2.11) is a high-speed printer which can print up to 2,000 lines per minute.

The 3211 is controlled and buffered by the 3811 Printer Control Unit which requires channel attachment to a 4300 processor.

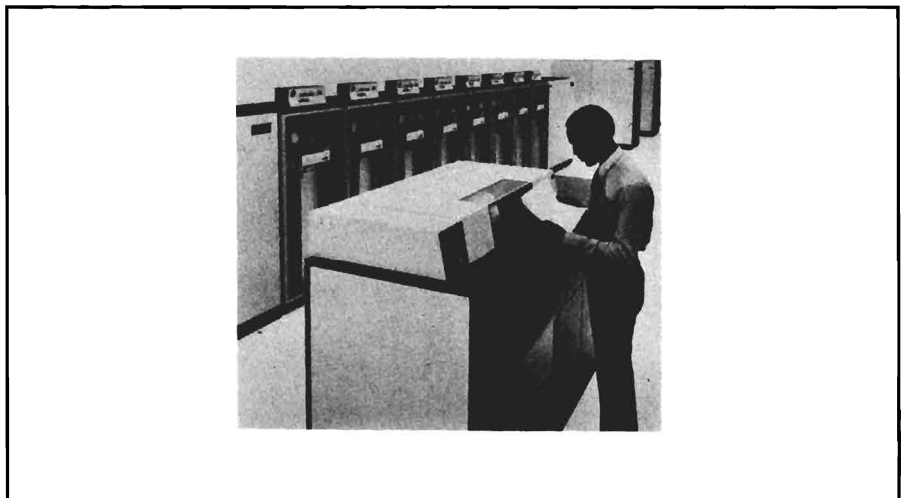


Figure 2.2.11 IBM 3211 Printer and IBM 3811 Printer Control Unit

3800 Printing Subsystem

The fastest and most versatile IBM printer is the IBM 3800 Printing Subsystem (Figure 2.2.12). The IBM 3800 is a high-speed, non-impact printer that produces characters on paper through electrophotographic and laser technology.

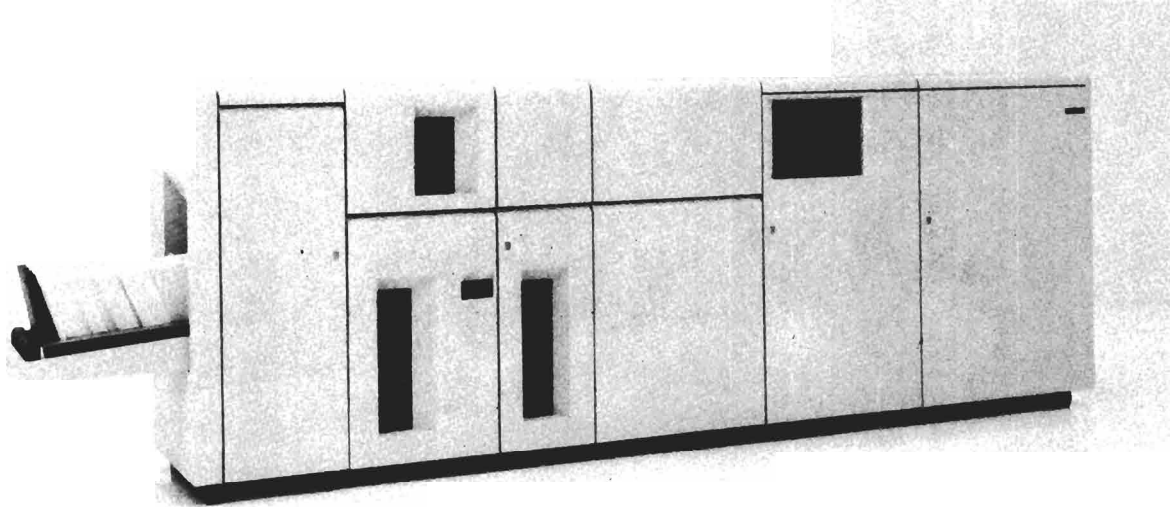


Figure 2.2.12 IBM 3800 Printing Subsystems

The 3800 printing speed is as high as 13000 lines per minute. Channel attachment to the 4300 processor is required.

Card Readers and Card Punches

Cards punched by either a keypunch or a card punch can be read by a card reader to provide input data to a data processing system.

3505 Card Reader

The IBM 3505 Card Reader (Figure 2.2.13) is a high-speed device available in two models which can read cards at a maximum rate of 800 and 1200 cards per minute.

The self contained control unit of the 3505 attaches to either one of the IBM 4300 processors via a channel.

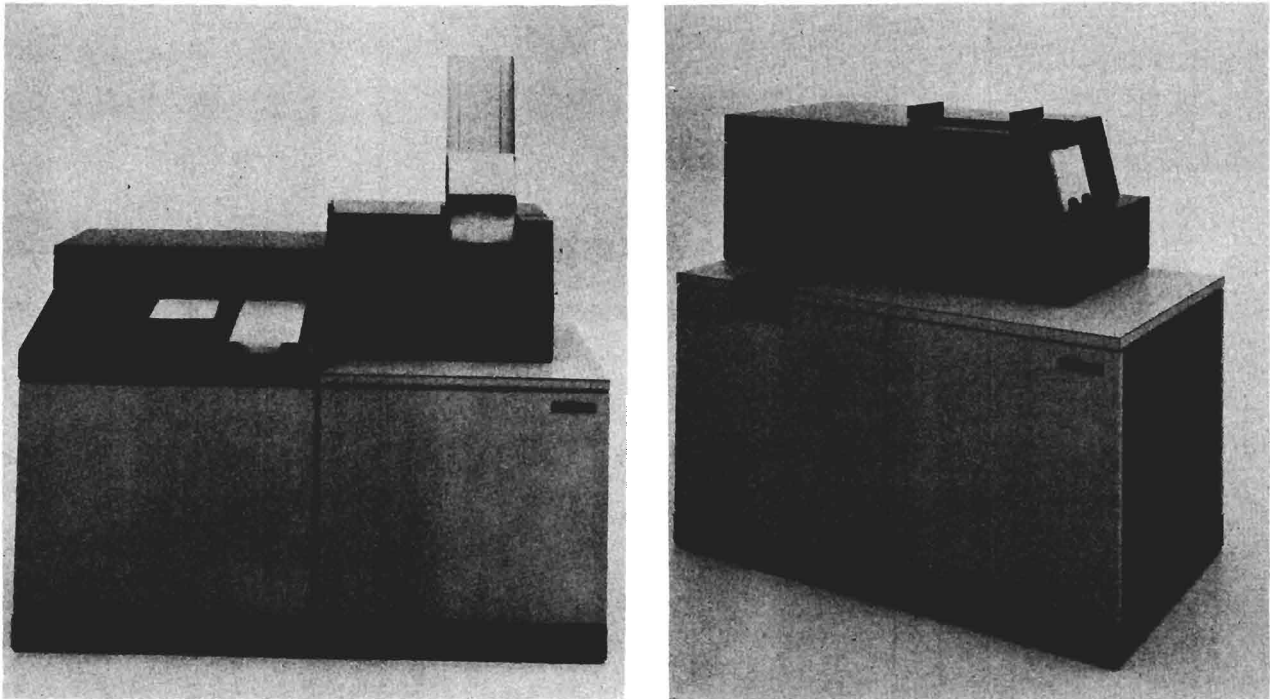


Figure 2.2.13 IBM 3505 Card Reader and 3525 Card Punch

3525 Card Punch

The 3525 (figure 2.2.13) is the card punching device associated with the 3505 card reader. It operates at up to 300 cards per minute. The 3525 has optional features for reading and printing. It is attached to a 4300 processor through the 3505.

2540 Card Read Punch

The IBM 2540 Card Read Punch (Figure 2.2.14) reads cards at a maximum rate of 1,000 per minute, and punches cards at a maximum rate of 300 per minute. The card reading and punching sections are separate entities, and reading and punching can take place simultaneously.

The 2540 is controlled, buffered, and attached to the processor channel through a 2821 control unit.

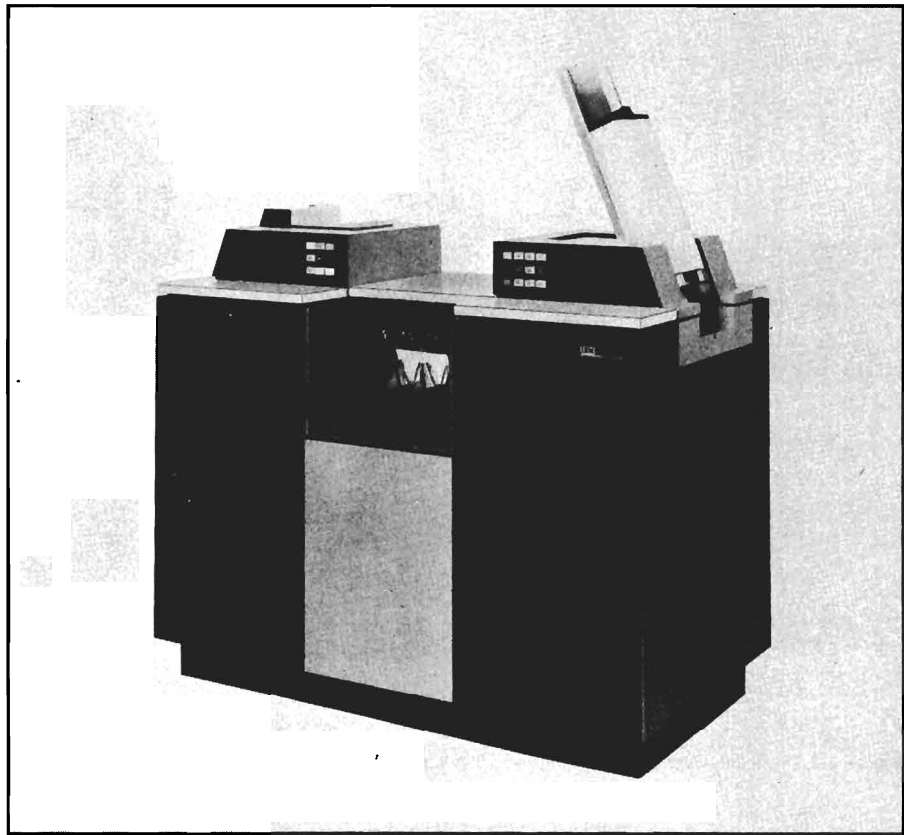


Figure 2.2.14 IBM 2540 Card Read Punch

5424 Multi-function Card Unit

The IBM 5424 Multi-function Card Unit (Figure 2.2.15) combines the functions of a card reader and a card punch using 96 column cards and attaches to the 4331 via the 5424 adapter.

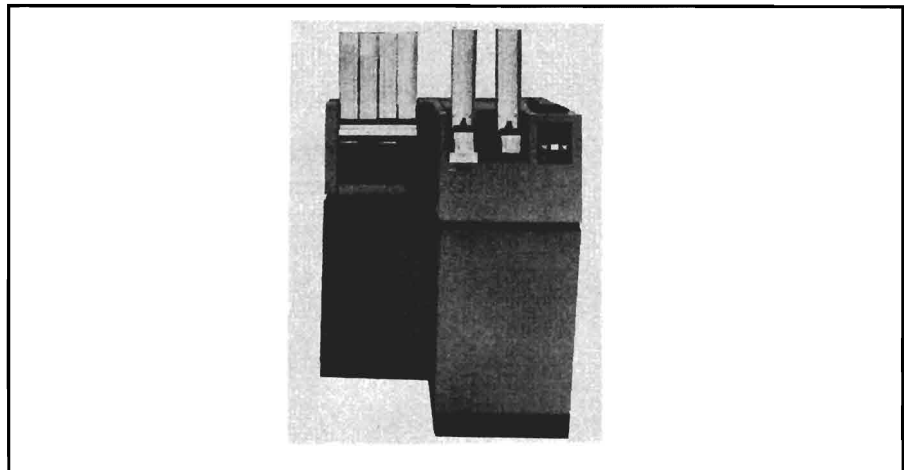


Figure 2.2.15 IBM 5424 Multi-Function Card Unit

Magnetic Tape Units

Magnetic tapes are used extensively for the off-line storage of large volumes of data, for active processing of sequential data files, and for copying or backing-up important files for protection against data loss.

3400 Devices

The IBM 3400 series tape units are devices with several features to permit ease of operation, high volume storage, and reliability.

3410 Magnetic Tape Unit and 3411 Tape Unit and Tape Control



Figure 2.2.16 IBM 3410 and 3411 Tape Unit and Control Magnetic Tape Unit

The 3410 and 3411 (Figure 2.2.16), are both available in three models. Each contains one tape unit, but the 3411 also contains the common control unit and power supply. The 3410 attaches to the 3411 and the 3411 is attached via channel to either 4300 processor.

Both the 3410 and the 3411 have a *dual-density* feature which allows reading and writing in either 800 or 1600 bits per inch (BPI) mode. Data transfer rates are from 20 to 80 Kilobytes (KB) per second depending on the model.

3420 Magnetic Tape Unit

The 3420 (Figure 2.2.17), available in six models, offers faster data transfer rates than the 3410 and 3411.

A choice of tape densities and track formats is available. Using the *single-density* feature, 3420 models operate at 1,600 bpi. Using the *dual-density* feature tape operations are performed at 800, 1,600, or 6,250 bpi. The 3420 has data transfer rates of 120.2 to 320 KB per second.

Control and diagnostic functions for the 3420 are contained in the separate 3803 Tape Control unit.

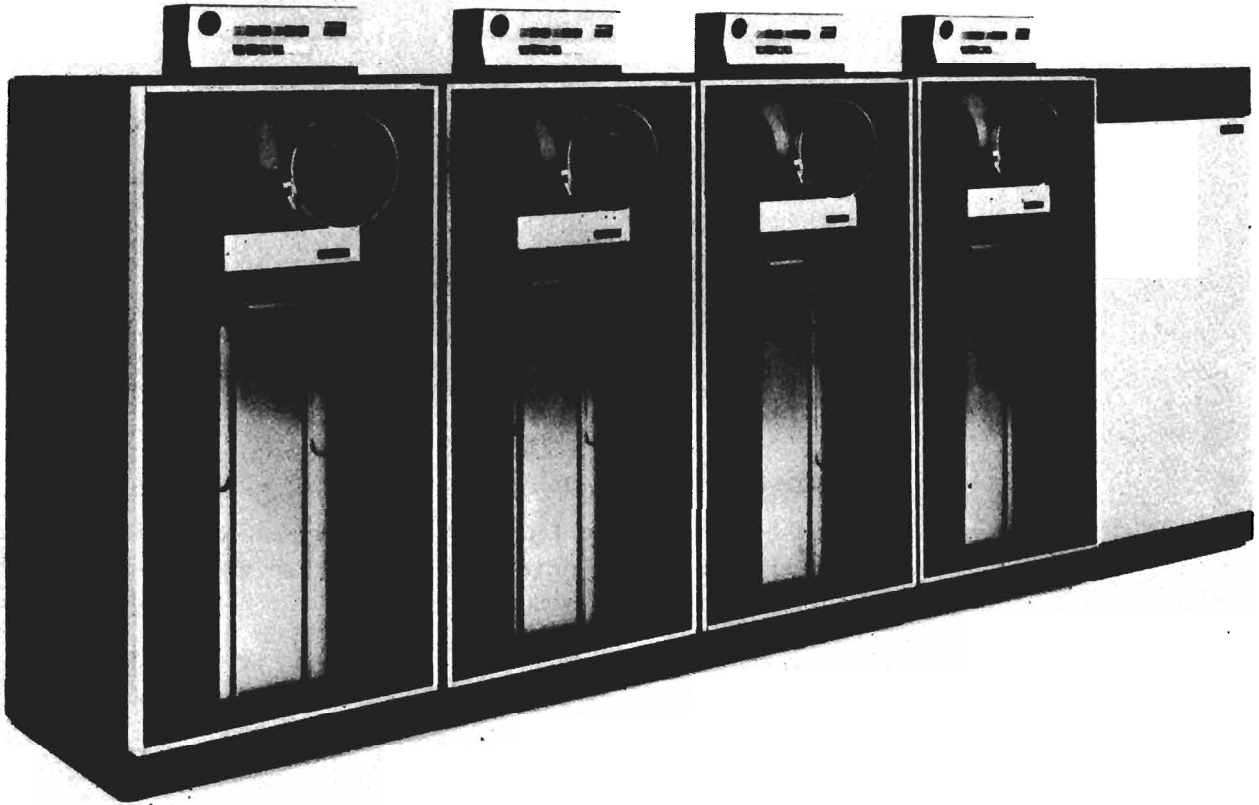


Figure 2.2.17 IBM 3420 Magnetic Tape Unit and IBM 3803 Tape Control

3803 Tape Control

The IBM 3803 Tape Control unit (Figure 2.2.17) provides control functions and diagnostic capabilities for the 3420 tape units.

Up to eight 3420's can be connected to a 3803. The 3803 is used to attach all models of the 3420 to the 4341 channel and four 3420 models to the 4331 channel. The 3420 models 3, 4, 5 and 7 are the only models which can attach to the 4331.

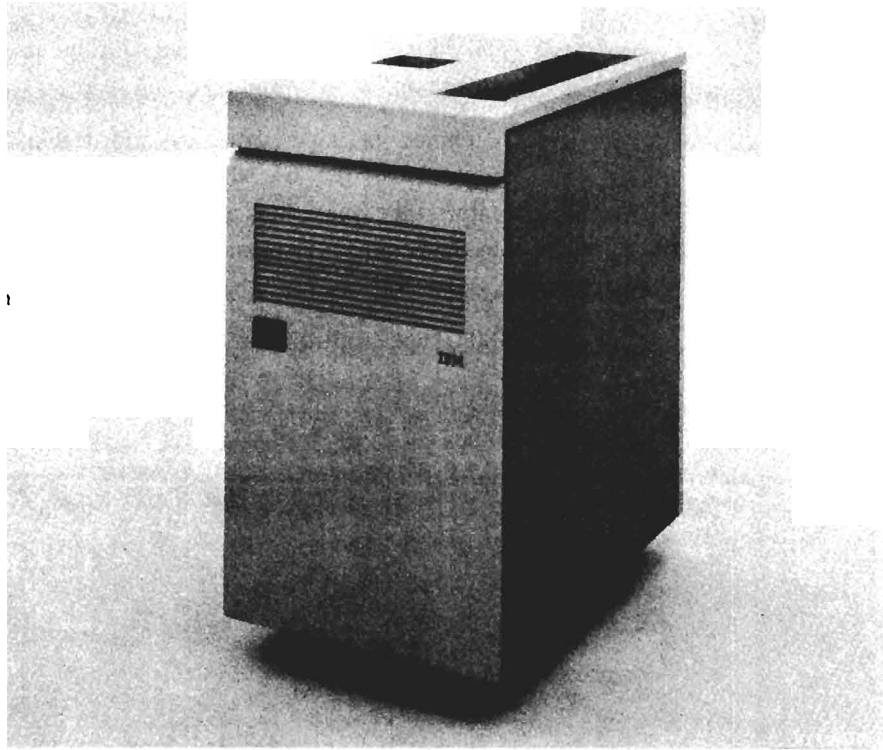


Figure 2.2.18 8809 Tape unit

The 8809 Magnetic Tape Unit

The IBM 8809 Magnetic Tape Unit (see Figure 2.2.18) introduces a new design in magnetic tape drives. Tape tension and velocity are controlled electronically rather than mechanically.

A maximum of six 8809 Magnetic Tape Units may be attached to the integrated adapter of the 4331 processor.

Dual speed capability is achieved by offering two operating modes.

Start/Stop Mode

A conventional start/stop mode is used for normal tape processing. In this mode, the 8809 achieves a 20,000 byte per second data transfer rate.

Streaming Mode

A new high-speed streaming mode is used to copy (dump) and restore data contained on direct access storage devices. In this mode, the 8809 achieves a 160,000 bytes per second data transfer rate. Streaming mode is NOT available for normal tape processing.

Tapes written in either mode have an identical format. This 1600 BPI tape format permits the compatible interchange of tapes with IBM 3400 tape subsystems.

Terminals There are many terminal types ranging from simple low cost keyboard printer terminals to sophisticated programmable devices. Some of the terminals can serve in the dual capacity of communications terminals and stand-alone data processors. We will discuss only the 3270 Information Display System and the 3770 Data Communication System.

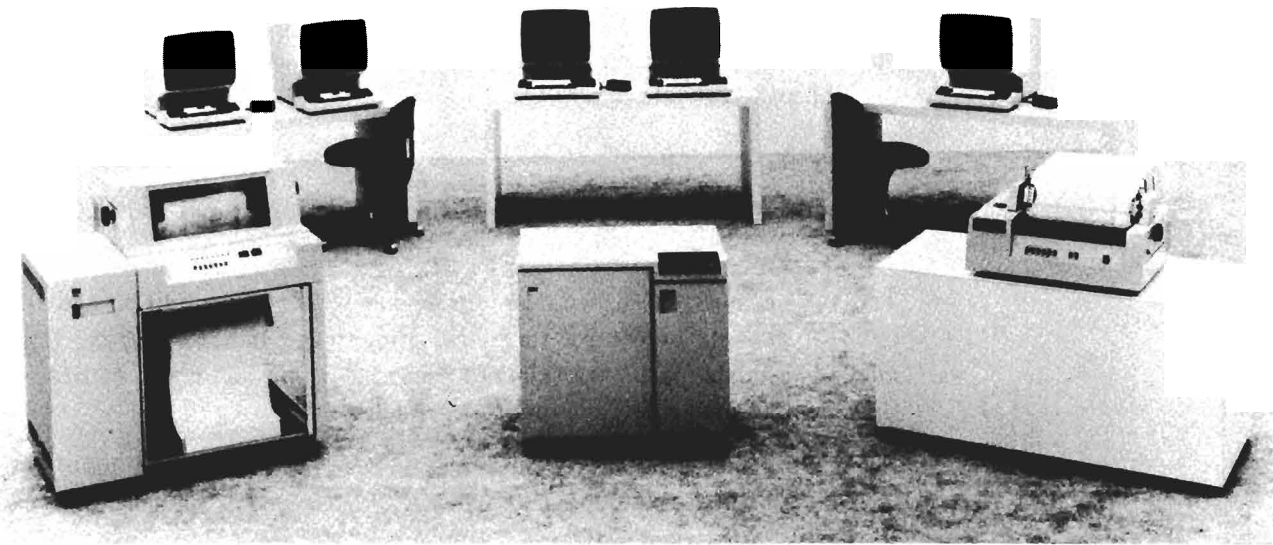


Figure 2.2.19 IBM 3270 Information Display System

IBM 3270 Overview The IBM 3270 Information Display System (see Figure 19) provides a series of *local or communication* terminal products, consisting of displays, printers, keyboards and auxiliary equipment which the user can combine into various configurations to meet different requirements.

The IBM 3270 terminal family includes a multitude of devices some of which are: the 3276 control unit/display station, the 3277 and 3278 display stations, the 3279 color display station, the 3289 printer, the 3287 standard printer and the 3287 color printer, and the 3274 control unit.

The 3270 Family can be attached either locally or remotely to the processor. The local attachment is via connection to channels. Certain models of the 3270 displays and printers can be attached to the 4331 display/printer adapter. The remotely located 3270 configuration is attached via communication lines through a communication controller such as the 4331 communications adapter or the 3704/3705. Either BSC or SDLC line control may be used.

3270 Display Units

The displays can have either 480, 960, 1920, 2560, or 3440 character capacity, with either size providing high quality displays of alphabetic, numeric and special character images.

The displays provide a special symbol, called a CURSOR, which is used to indicate where the next character entered from the keyboard will be displayed on the screen.

Program Function (PF) keys provide the capability of giving the terminal operator access to user program functions by depressing a PF key on the terminal keyboard. Depending on the terminal model as many as 24 program functions keys are available. The function of each PF key depends on the definition that was provided by the application programmer.

The displays also provide a selector pen capability, which allows the terminal operator to select information from the display screen with a light pen rather than through keyboard entry.

3270 Printers

Within the 3270 family are printers having a variety of speeds and characteristics.

The 3287 is available in two desk top models which print up to 80 and 120 CPS. It consists of control functions, printer, and indicator lights. Each model is available in both a standard single color unit and a unit capable of single and/or multicolored output for each line.

The 3289 model 4 printer provides the 4331 with print capability up to 400 lines per minute when operated under DOS/VSE. It is attached to the Display/Printer Adapter of the 4331 processor.

3270 Control Units

The 3274 control unit can control a group or cluster of up to 32 display/printer terminals locally or remotely. The 3276 control unit/display station can attach up to seven 3278 model 1, 3287 and 3289 devices in any combination in a remote environment.

IBM 3770 Overview

The IBM 3770 Data Communication System (shown in figure 2.2.20) consists of a family of *communication terminal* offerings that combine a keyboard and printer with a selection of terminal input/output components and communication features to provide many general purpose, fixed function and programmable configurations.

Additional function is provided by attaching card reading and punching units such as the 3501 Card Reader and the 3521 Card Punch to the configuration. A 155 line per minute print capability can be included by adding a 3784 line printer to a 3770 configuration.

The host system attachment requirements and programming support is the same for all of these terminal offerings, that is via communication control unit and communication lines. The 3770 configuration may be operated using SDLC or BSC line control techniques.

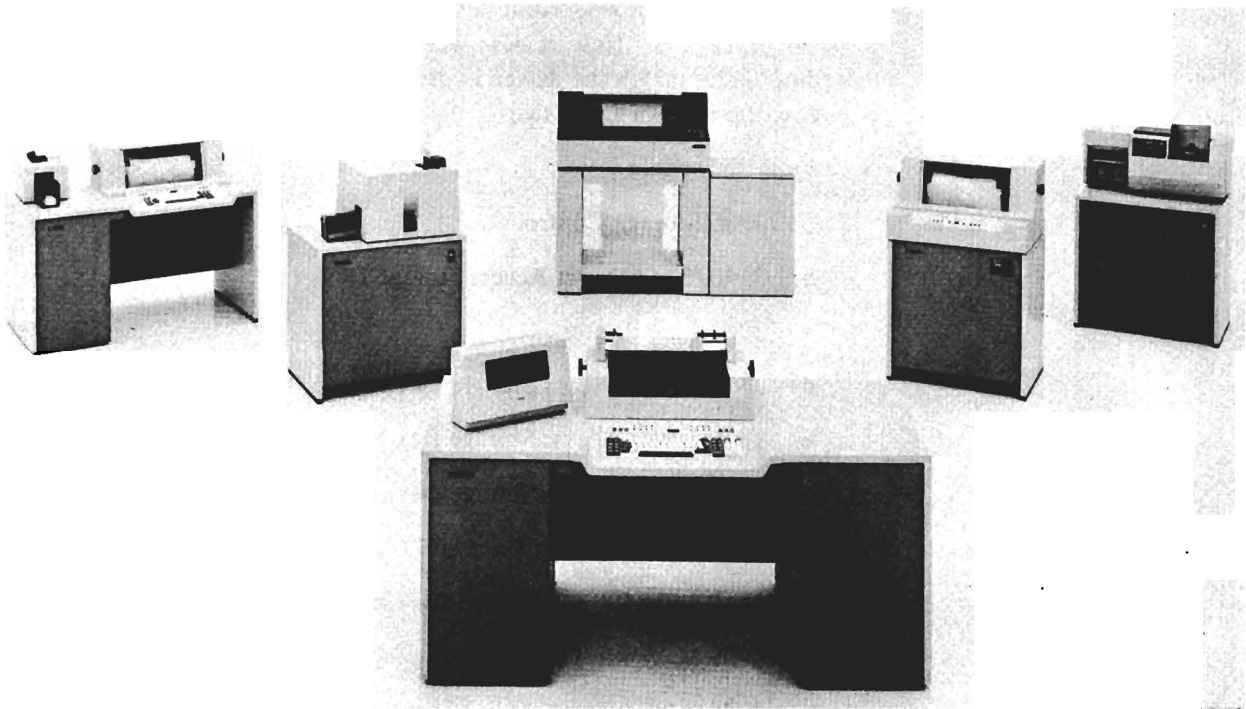


Figure 2.2.20 3770 Terminals

3770 Fixed Function Terminals

The fixed function models of the 3770 provide limited capability and flexibility of operation. Basically, they permit such "data processing" functions as data input, data storage, data movement and data output to occur. However, if the user's application requires any arithmetic or logic activity upon the 3770 data, it is necessary for the data to be transmitted to the host processor. It is at the host processor that the various application programs are stored and where the arithmetic and logic capability of the system resides.

3770 Programmable Terminals

The programmable models of the 3770 permit all the functional capabilities of the fixed function models, and in addition, permit extensive arithmetic and logic operations to be performed. This permits the programmable 3770's to operate in the dual capacity of both a data terminal and as a stand-alone remote processor.

Programmed operation of the 3770 is controlled through the application programs stored at the 3770. These application programs vary depending upon what the user wishes to do with the data.

Exercise 2.2 We have discussed many different devices which enhance the effectiveness of the 4300 processors. Now review what we have covered. To review the preceding topics match the device in group 1 with the key feature or appropriate description from group two.

- Group 1 -Devices
 - a. 3880 Storage Control
 - b. 3340/3344 Direct Access Storage Devices
 - c. 3262 Line Printer
 - d. 3350 Direct Access Storage Device
 - e. 3310 Direct Access Storage Device
 - f. 3370 Direct Access Storage Device
 - g. 5424 Multi-function Card Unit
 - h. 3410 and 3420 Magnetic Tape Units
 - i. 8809 Magnetic Tape Unit
 - j. 3278-2A Display Unit
 - k. 3540 Diskette Input/Output Unit
 - l. 3270 Display/Printer Family
 - m. 3770 Communication Terminal Family
- Group 2 -Descriptions and Features
 1. _____ Is a unit which reads and punches 96 column cards. This unit is attached by a special adapter to the 4331 processor.
 2. _____ May be operated on the 4331 in either streaming mode or in a stop/start mode. Is directly attached via a special 4331 adapter.
 3. _____ Provides single and dual density with a choice of 800, 1600, or 6250 BPI operation. These units offer a choice of data transfer rates from ranging from 20 to 80 and from 120.2 to 320 kilobytes per second. The faster units require a separate control unit while the slower unit has a model with a self contained control function.
 4. _____ Is required as a system console for the 4331 and the 4341 processors.
 5. _____ Is a control unit with two separate data paths each capable of handling both CKD or FBA devices.
 6. _____ Provides a family of printers, display terminals, and control units which may be operated in either a local or remote environment. This group of devices can be used with either BSC or SDLC line control.

7. _____ Is a fixed or non-removable intermediate capacity FBA DASD specifically designed to attach to the 4331 processor DASD Adapter.
8. _____ Are 35, 70, or 280 million byte capacity sealed disk units. These CKD devices can be attached to the 4331 DASD Adapter or to a 4341 via a 3880 control facility. The smaller unit uses a removable data module and the larger capacity unit uses a non removable disk as the recording medium.
9. _____ Is a large capacity fixed disk FBA device which has two ACTUATORS that can independently access half the data area available on a spindle.
10. _____ Is a 317.5 million byte capacity non-removable sealed disk CKD device which requires the 3880 Storage Control for attachment to the 4341 processors.
11. _____ Is designed to provide printed output for the 4331 at a maximum speed of 650 lines per minute. A maximum of two of these devices may be attached to a special 4331 adapter.
12. _____ Is an input and output device which uses the IBM diskette as a recording medium.
13. _____ Is a family of communication devices which can be configured as a standalone subsystem providing card input and output, print capability, display and key data entry over communication lines using either BSC or SDLC line control.

Solution 2.2 Review the preceding topics for answers to any matches you missed after you check the answers given below.

1 -g

2 -i

3 -h

4 -j

5 -a

6 -l

7 -e

8 -b

9 -f

10 -d

11 -c

12 -k

13 -m

The end of Unit 2.

Unit 3:

Introduction to DOS/VSE

Introduction The Disk Operating System/Virtual Storage Extended (DOS/VSE) is a set of programs and libraries designed to support the operation of the IBM 4300 and System/370 data processing systems. Multiprogramming and system management facilities are provided to optimize the productivity of the system and simplify its use. Additional multiprogramming support and extended system management and useability facilities are provided by the optional VSE/Advanced Functions program product.

VSE/POWER, another program product, provides spooling facilities which optimize the productivity of unit record devices. VSE/POWER also enhances system performance and operator control of the system. In addition, VSE/POWER supports communications with and provide spooling for Remote Job Entry terminals.

This unit describes the major functions of DOS/VSE and VSE/POWER. Selected features of VSE/Advanced Functions are also discussed.

Objectives Upon completion of this unit, the student should be able to:

- Identify the DOS/VSE system control programs and describe their major functions.
- Describe DOS/VSE storage organization.
- Understand the DOS/VSE implementation of multiprogramming.
- Explain how multiprogramming enhances the productive use of CPU cycles.
- Describe the functions of the shared virtual area (SVA).
- Describe the DOS/VSE implementation of virtual storage.
- Identify and define the functions of DOS/VSE libraries.
- Describe key functions of the DOS/VSE librarian programs and linkage editor.
- Identify some of the key features of DOS/VSE Advanced Functions.
- Describe the features of VSE/POWER.
- Describe the basic data management support provided by DOS/VSE and VSE Virtual Storage Access Method (VSE/VSAM).

Average Study Time 3 to 4 Hours

**Topic - 1 DOS/VSE
System Control
Programs**

The DOS/VSE system control programs, IPL (Initial Program Load), Supervisor, and Job Control Program, reside in a library known as the system Core Image Library (CIL). This library is located on a disk designated as the system residence device or SYSRES.

The IPL Program

The IPL program is used to start the DOS/VSE system. The IPL process begins with the loading (into processor storage) of a small program called the "bootstrap" which is located at a fixed address of SYSRES (Figure 3.1.1).

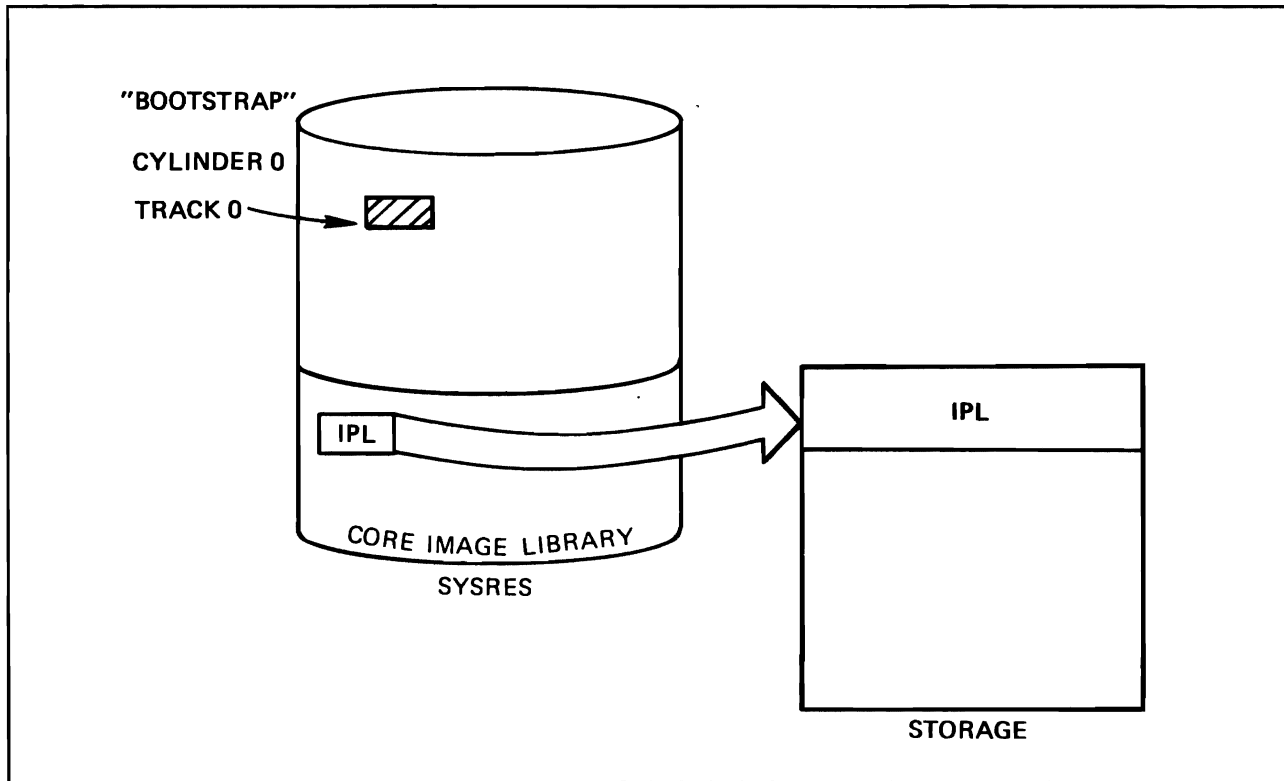


Figure 3.1.1 The IPL Function

The operator initiates the IPL function through the system console. The bootstrap program locates the IPL program in the Core Image Library and loads it into processor storage (Figure 3.1.1). The IPL program loads the DOS/VSE Supervisor into processor storage (Figure 3.1.2) and performs system initialization functions.

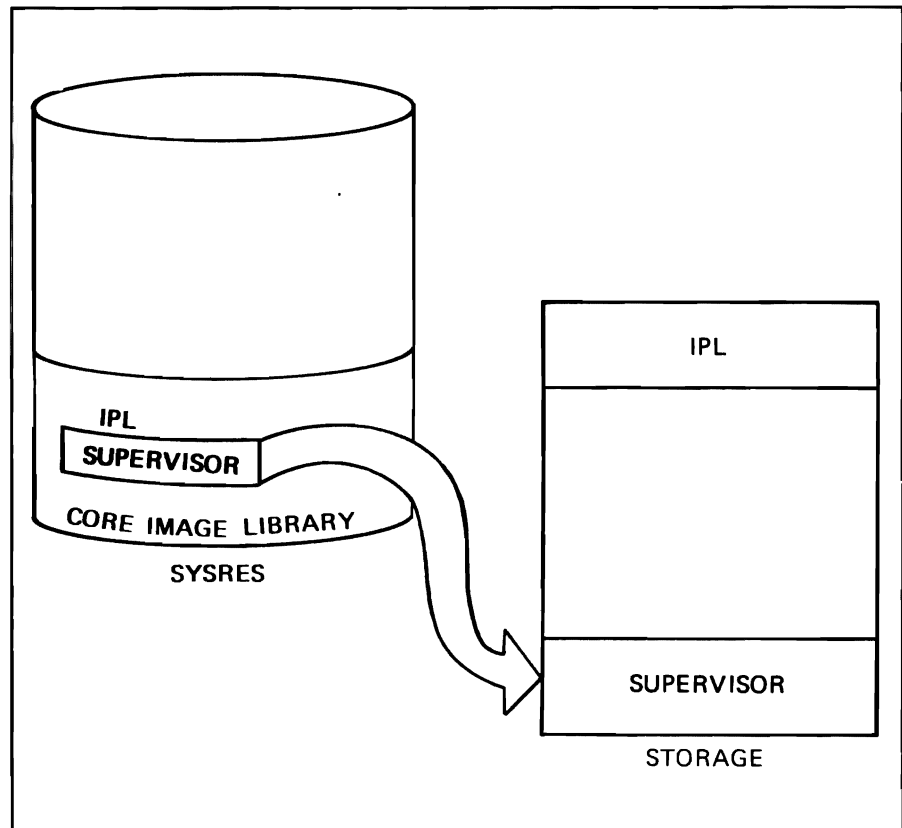


Figure 3.1.2 The DOS/VSE Supervisor

Some of the key initialization functions are:

- Furnish the supervisor with the addresses and characteristics of the I/O devices in the current hardware configuration.
- Prepare the page data set for virtual storage and provide its location to the supervisor.
- Set the time of day clock.

If VSE/Advanced Functions is installed in the system, the initialization process can be almost completely automated by using the Automated System Initialization (ASI) facility. With this facility, the IPL program can read the data required for system initialization from a library on disk. Without ASI, the data must be supplied by the operator through the console or a card reader.

The Supervisor

The supervisor loads into storage beginning at location 0. Once loaded, it remains in storage continuously during system operations. The supervisor initiates the loading and execution of all other programs and handles the input and output of data that is processed by any program executed in the system.

The supervisor controls main storage utilization and the use of CPU cycle time. It handles communications to and from the operator, error recovery, and input/output operations. In a word, the supervisor controls the system. Later in this unit, we will describe some of the key supervisor management functions.

The Job Control Program (JCP)

The job control program is loaded by the supervisor to begin execution of user programs. The job control program acts on information you supply, such as program name and main storage and device requirements, to allocate resources and prepare DOS/VSE to execute your programs. The interaction between the job control program and a user program is shown in Figures 3.1.3 through 3.1.7.

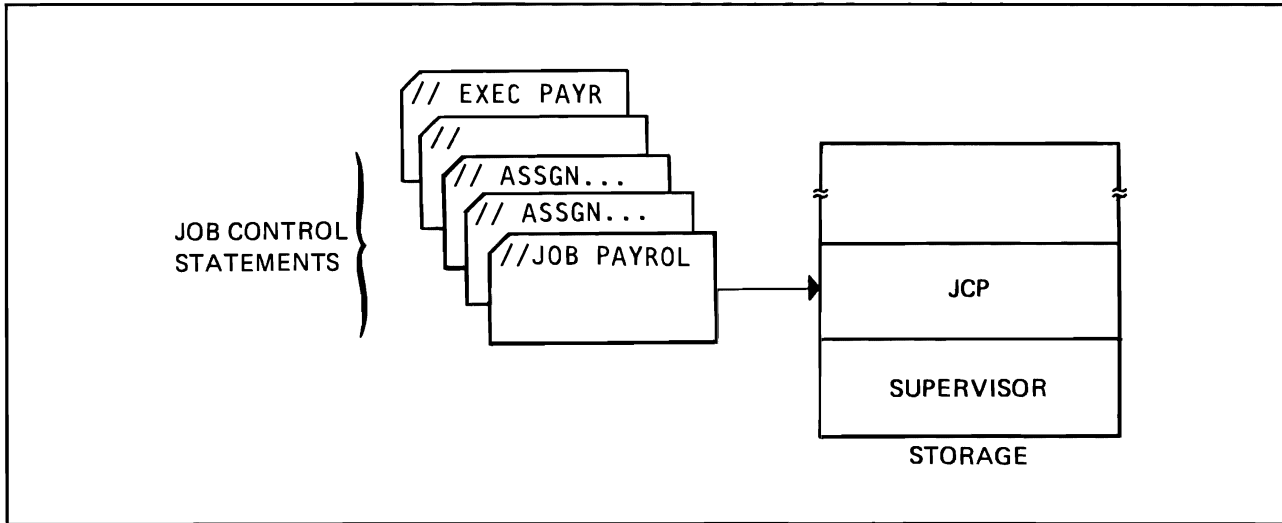


Figure 3.1.3 The Job Control Program Reads Job Control Statements.

1. The job control program reads job control statements written in the DOS/VSE job control language. These user prepared statements identify the job, designate the devices that will be used by the program for its input and output and provide information required for identifying and locating data stored on direct access storage devices and tape. Figure 3.1.3 shows the job control program reading job control statements for the execution of the "PAYR" program.

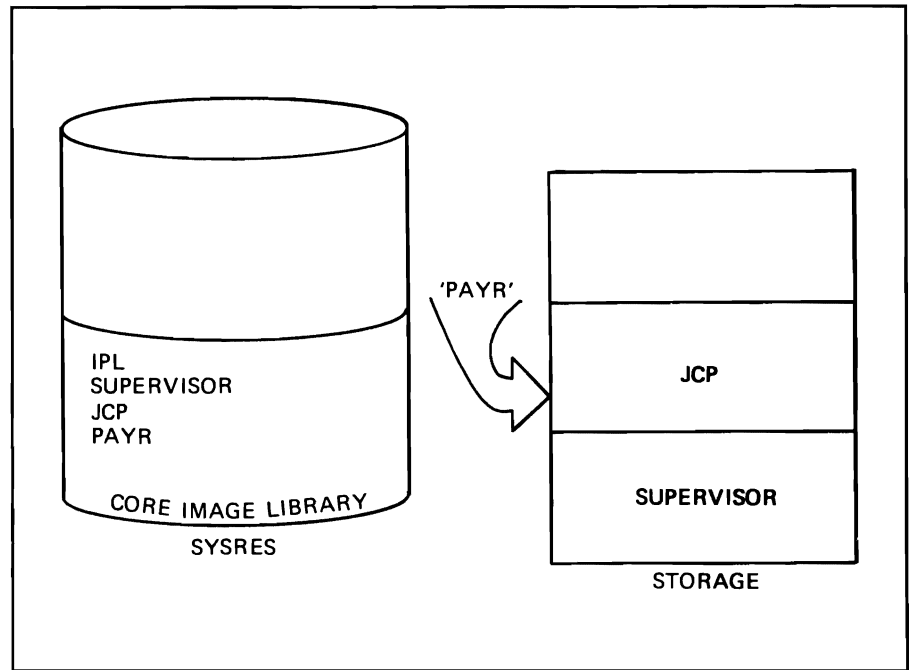


Figure 3.1.4 Initial State

2. The job control program is informed that we wish to run a program named PAYR. This is done by specifying the program name, PAYR, on an EXEC job control statement. Notice that PAYR exists in the Core Image Library (Figure 3.1.4).

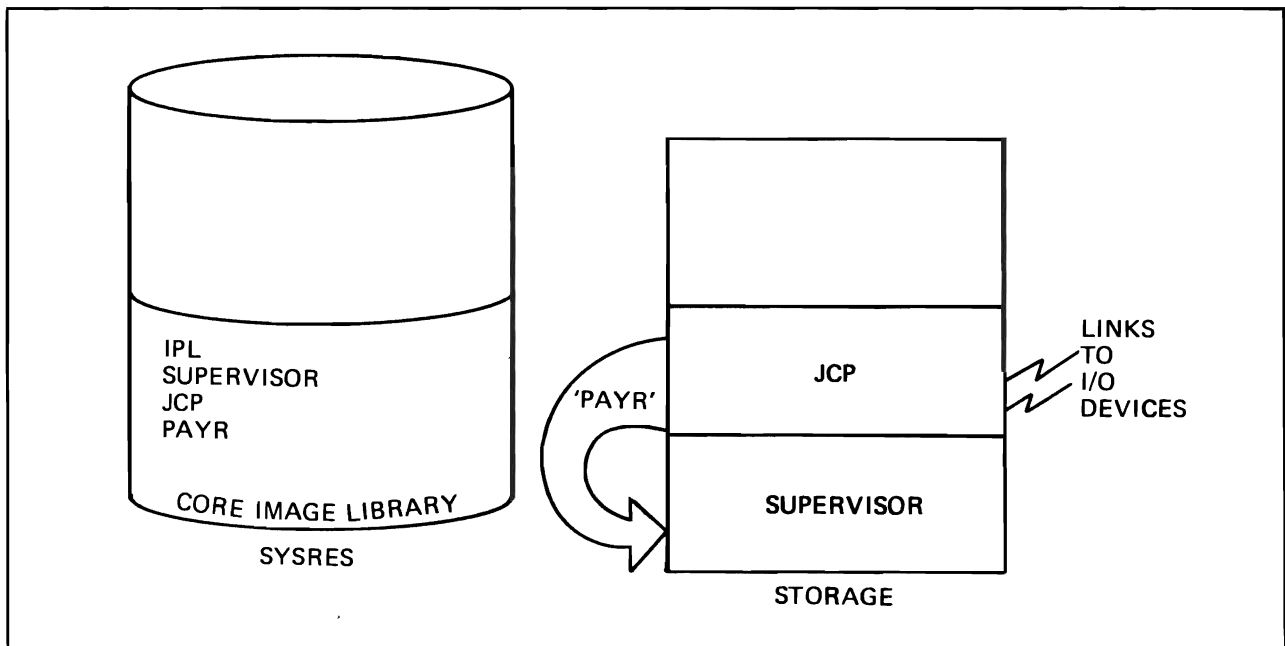


Figure 3.1.5 The Job Control Program Prepares for PAYR

3. The job control program makes sure that the storage and device requirements of PAYR are satisfied, then informs the supervisor that PAYR is to be executed (Figure 3.1.5).

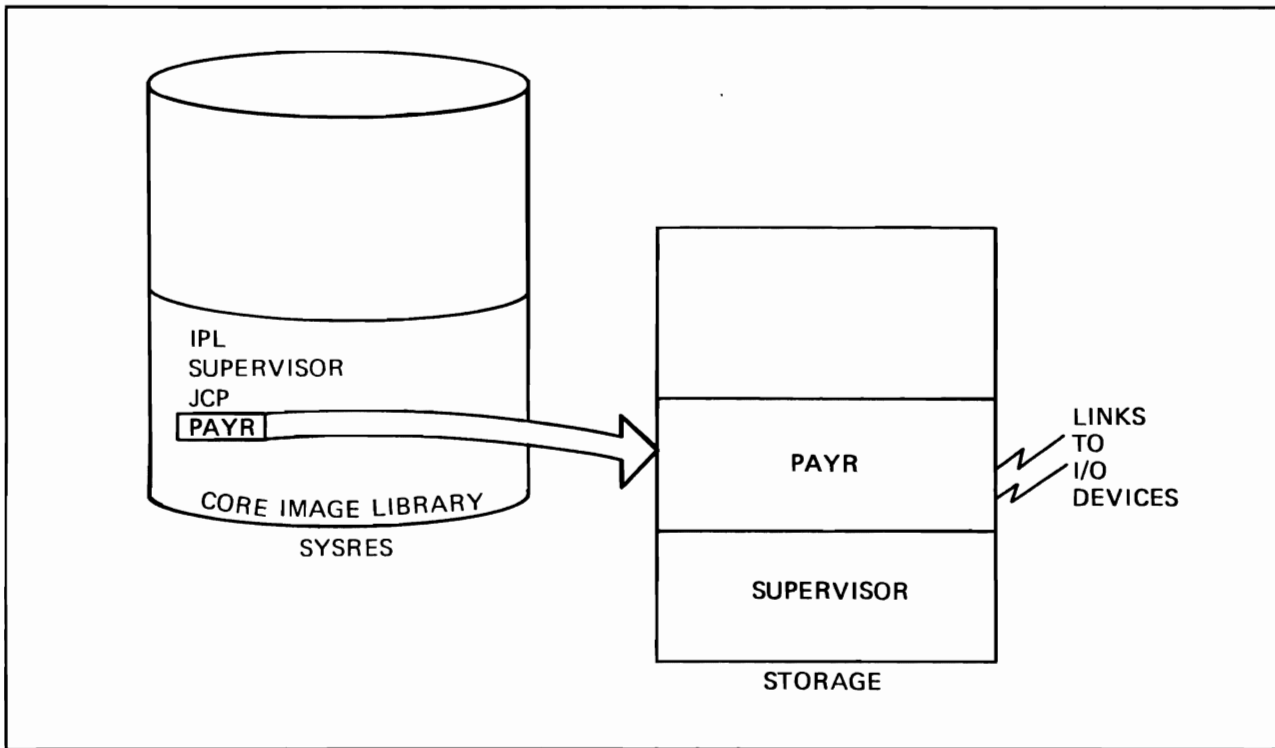


Figure 3.1.6 PAYR is Loaded for Execution

4. The supervisor loads PAYR and allows it to begin execution. Figure 3.1.6 shows that PAYR is loaded into the same storage location as the job control program. This is possible because the job control program is not needed in this area of storage during the time that PAYR is executing.

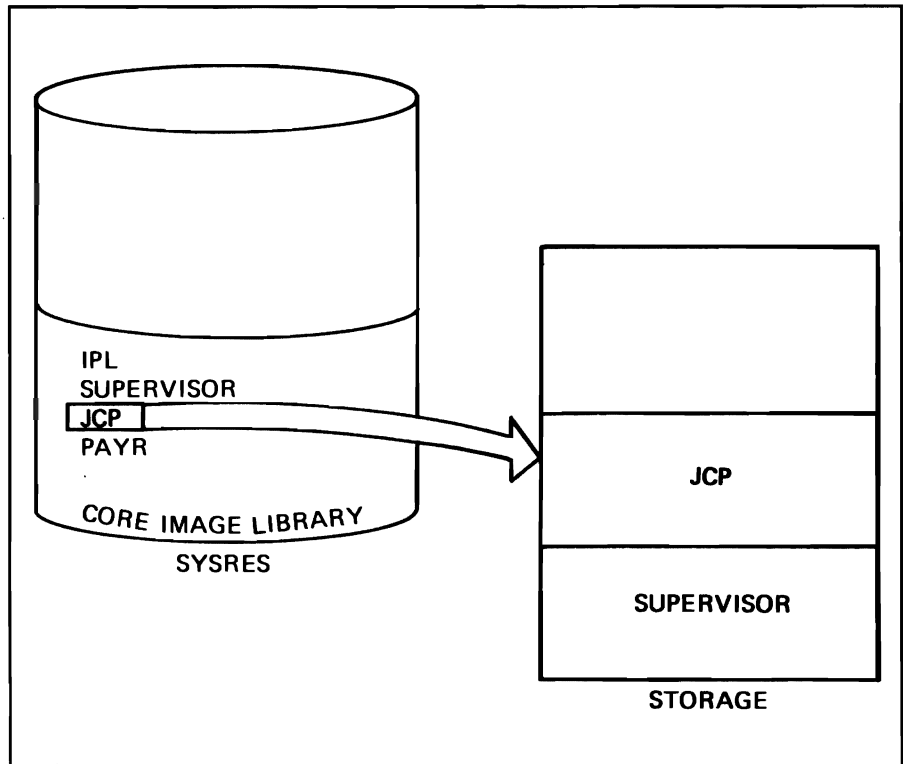


Figure 3.1.7 Final State = Initial State

5. The supervisor knows when PAYR has completed its processing. It reloads the job control program, overlaying PAYR. The cycle is complete and will be repeated to set up for execution of the next user program.

Notice in this last diagram that the I/O links are gone. This is because they were needed only for PAYR. The next program will probably have a different set of requirements which will in turn be set up by the job control program. This sequence of events is called job-to-job transition.

Exercise 3.1

1. The DOS/VSE supervisor is activated by the _____.
 - a. job control program
 - b. operator
 - c. SYSRES device
 - d. IPL program
2. An application program is loaded into storage for executing by the _____.
 - a. supervisor
 - b. IPL program
 - c. job control program
 - d. programmer
 - e. SYSRES device
3. The _____ controls main storage utilization and the use of CPU cycle time.
 - a. operator
 - b. supervisor
 - c. job control program
 - d. channel
4. The _____ allocates resources and prepares DOS/VSE to execute your program.
 - a. operator
 - b. job control program
 - c. IPL program
 - d. CPU
5. The user prepares _____
to identify the job and designate the I/O devices for the job.

Solution 3.1

1. d
2. a
3. b
4. b
5. job control statements

**Topic - 2 DOS/VSE
Storage and
Multiprogramming**

One of the primary goals of an operating system is to optimize resource utilization. This topic describes DOS/VSE storage organization and multiprogramming which work together to optimize the utilization of storage and CPU cycles.

Storage Organization

In DOS/VSE, the virtual storage space reserved for active programs is divided into areas called partitions. From two to five partitions can be defined (see Figure 3.2.1). If VSE/Advanced Functions is installed, the maximum number of partitions is increased to twelve.

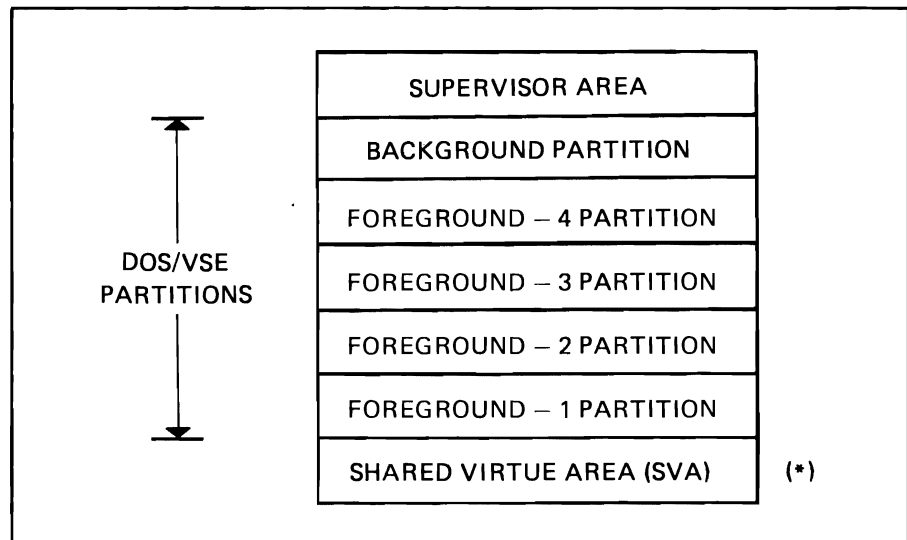


Figure 3.2.1 DOS/VSE Storage Organization.

Normally, a program is executed in each of these partitions. A hardware feature known as storage protection is used by DOS/VSE to protect the storage in one partition from being altered by a program running in another partition. Without this protection, one program could accidentally alter another program and cause erroneous results.

Partitions and I/O Resources

I/O resources are allocated to partitions by the job control program. Some I/O resources such as disk devices can be allocated to more than one partition at a time. Others like card devices, printers and tape devices can only be allocated to one partition at a time. (Later we will see that VSE/POWER will allow the user to assign card and printer devices to multiple partitions.) In order to run a program in a partition, all resources required by the program must be allocated to the partition.

Multiprogramming

Recall from Unit 1 of this text that once an input or output request has been started for a program, the I/O operation is directed by a program that is executed by the channel rather than by the CPU.

Compared to the CPU's internal speed, the rate of data transfer between an I/O device and processor storage is very slow. The reading or writing of data involves mechanical as well as electronic actions. Positioning of disk access mechanisms, punching cards, and the like are time consuming operations that

slow down a program's performance. If a given program is the only user of the CPU, then the CPU is essentially idle while that program is waiting for completion of its I/O operations. Rather than waste the expensive resource of CPU time, it is preferable to allow more than one program to be active simultaneously. This process is called multiprogramming. The programs in a multiprogramming system are independent of each other. While Program A, for example, is waiting for an I/O operation to complete, Program B can be processing.

Remember, only one thing can take place in the CPU at any one time, so Program A and Program B cannot both be using CPU cycles at the same instant. Either one of them, however, can process "in the gaps" created by the wait time involved from the other's I/O activity. In order to demonstrate how multiprogramming improves the utilization of CPU cycles time, let's compare two operating environments.

Processor Usage in Single-Partition Operation

In single-partition operation, only one program can be in storage at a time. When, for example, the program needs input or output, it issues an I/O request to the system supervisor. The supervisor passes this request to a channel, which then executes the I/O operation. During most of this time interval, the central processor remains idle, or in the wait state (Figure 3.2.2).

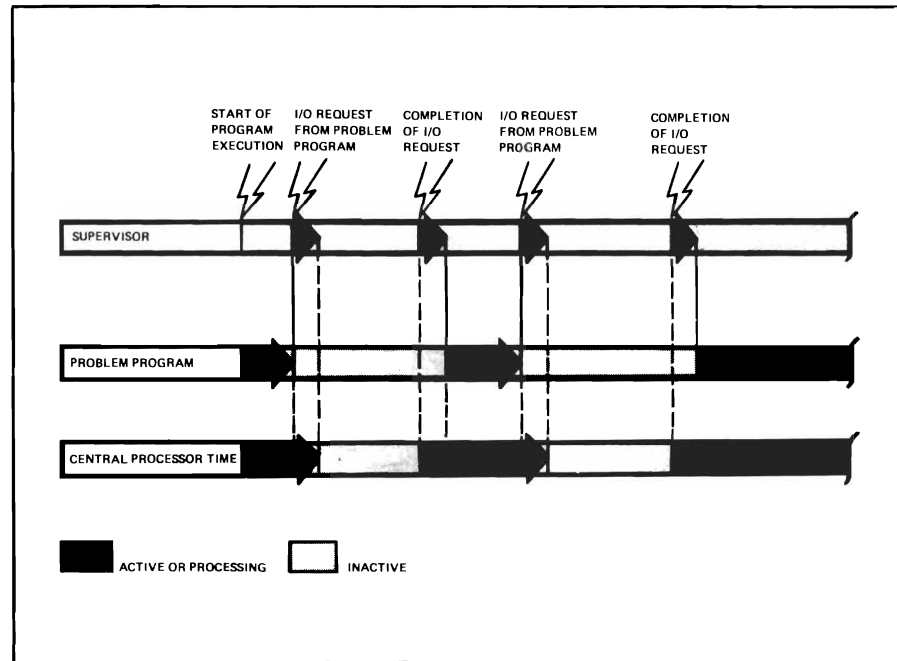


Figure 3.2.2 Processor Usage in a Single-Partition Operation.

Processor Usage in a Multiprogramming Environment

Figure 3.2.3 illustrates the passing of control among programs in a multiprogramming environment. Note how much less processor time is left unused or idle when compared to processor usage in a single-partition system (refer to Figure 3.2.2).

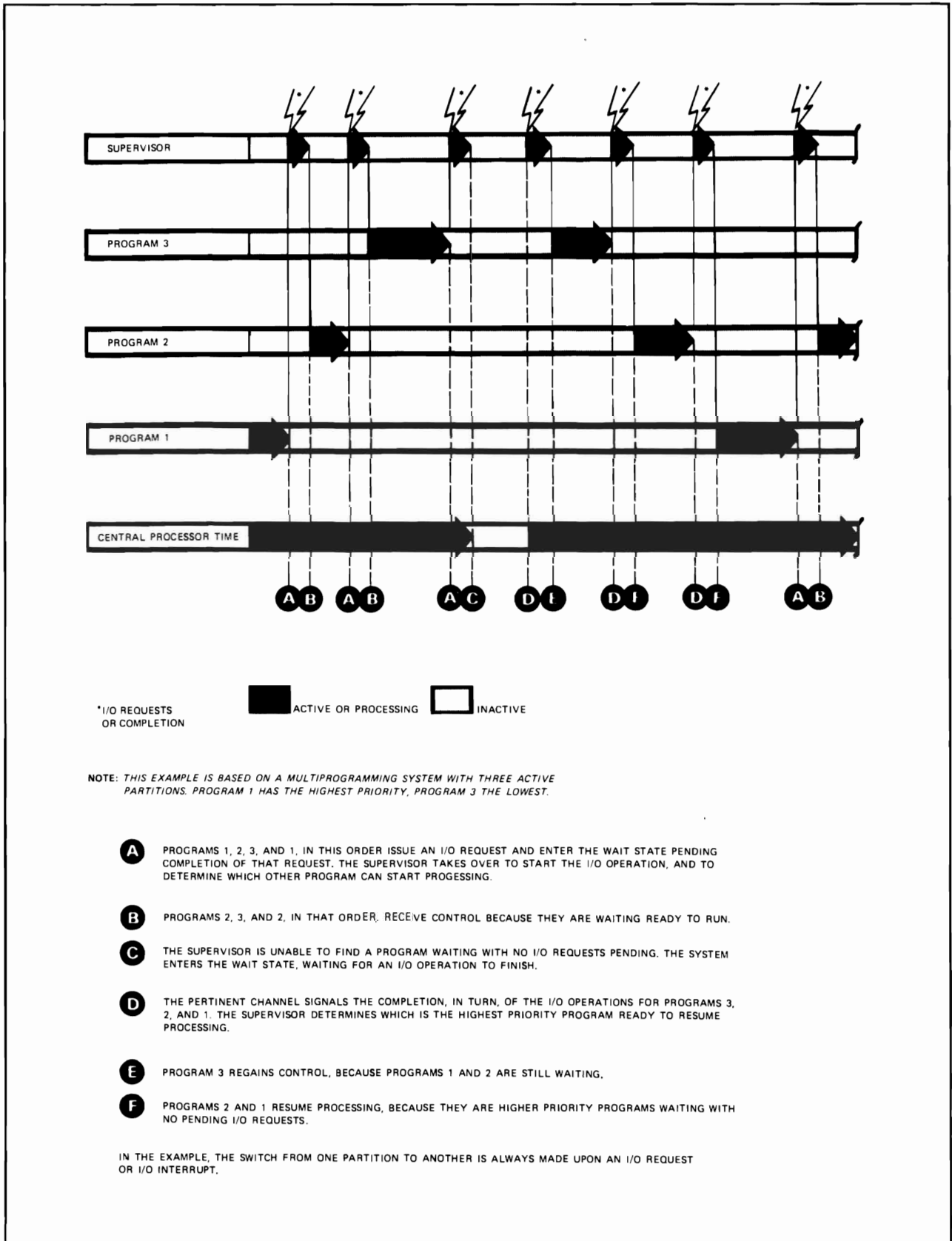


Figure 3.2.3 Processor Usage in a Multiprogramming DOS/VSE

Processing Priorities

With programs taking turns executing in a multiprogramming environment, processing must obviously proceed according to a set of priorities. The priority of a program for receiving processor resources is dependent upon the processing priority of the partition in which the program resides. The supervisor, of course, always has the highest priority.

The default priorities for the partitions are shown in Figure 3.2.4. However, DOS/VSE allows you to change these default values by means of an operator command. By assigning a job to a certain partition, a programmer or an operator, therefore, assigns the priority of that partition to the job as well.

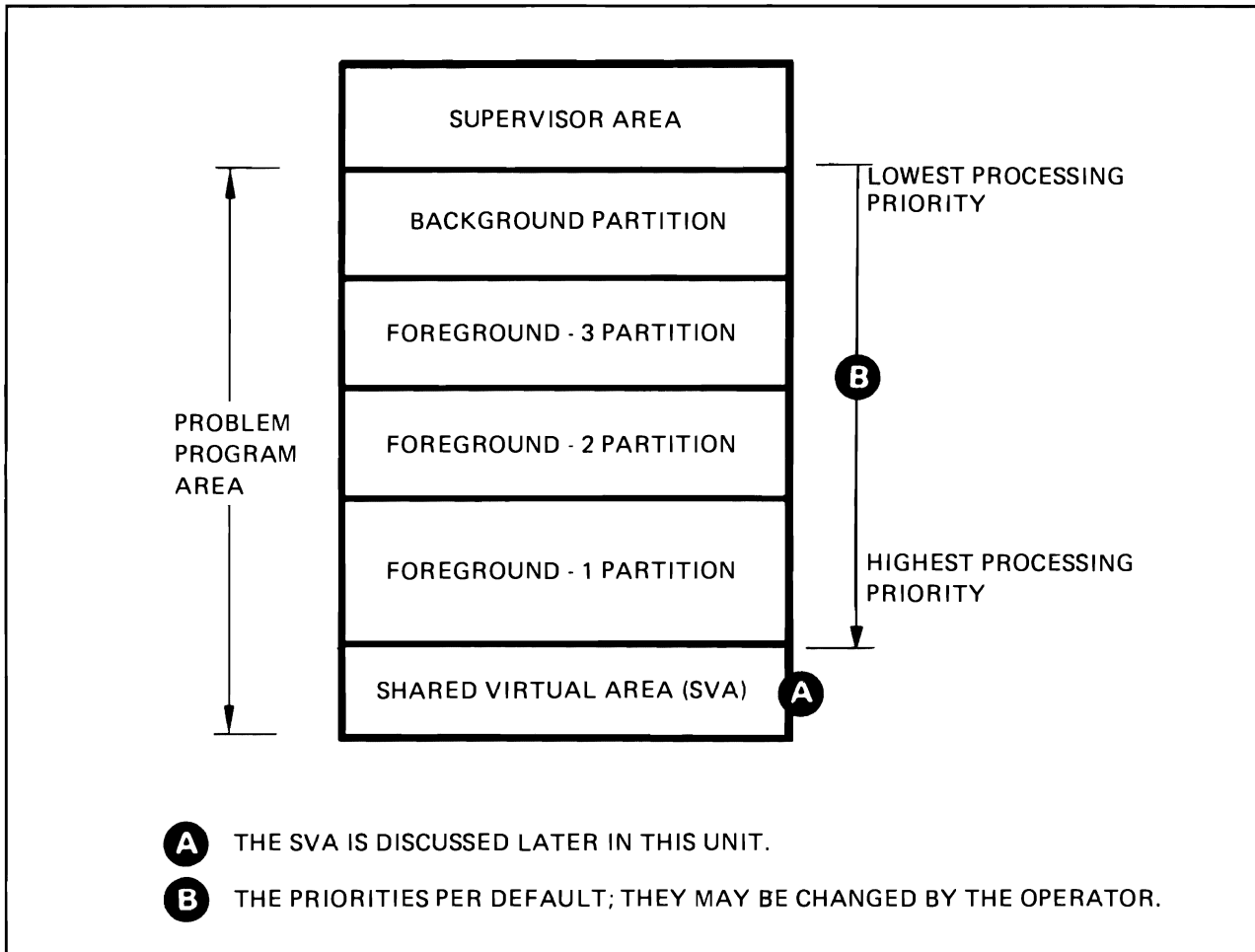


Figure 3.2.4 Storage Organization of and Processing Priorities for Partitions in a 4-partition DOS/VSE

Multitasking

Multitasking, a special form of multiprogramming, allows concurrent execution of two or more sections of a program, called task, within one partition. The purpose of this facility again is to make more efficient use of processor time.

With multitasking, one main program, the main task initiates or attaches one or more subprograms, or subtasks. The main task and its attached subtasks always reside in the same partition.

The total number of tasks in a system depends on the number of partitions. The sum of all subtasks attached at any one time and all partitions must not exceed 15. A 4-partition DOS/VSE, for example, allows you to have a maximum of eleven subtasks attached at any point in time. If VSE/Advanced Functions is installed, the total number of tasks is extended to 208. Although many users never write multitasking programs, there are program products that use multitasking that are in wide use.

The Shared Virtual Area

In addition to the partitions specified when the DOS/VSE system is installed the user must also specify the mandatory shared virtual area (SVA). This area of storage is used for three purposes:

1. To hold reenterable program phases. These are routines (instructions and/or data) that are available for use by any program (or programs) that are executing in any of the partitions at any time. SVA routines must be reentrant, that is, they must allow for shared use by one or more programs. Thus, even though only one copy of the routine exists in the SVA, it can be shared by any number of programs. In this way, the SVA routine does not have to be physically duplicated in each program where it is needed.

This capability allows the IBM supplied modules of the Virtual Storage Access Method (VSE/VSAM) to be loaded and executed from the SVA rather than from the user's partition. This means that if different partitions are using VSE/VSAM at the same time, only one copy of the VSE/VSAM code (in the SVA) will be needed to service these users.

2. To hold the system directory list (SDL). This directory contains entries (consisting of program names and locations in the SVA) of each SVA routine. In addition, the SDL may contain entries providing the locations of selected programs in the core image library (CIL). When one of these programs must be loaded, the supervisor locates it by using the SDL entry rather than the CIL directory entry. Program loading time is reduced by avoiding the usually required access to the CIL directory on DASD for these programs. The SDL used in this way can be thought of as an index to selected programs.
3. To hold the system GETVIS area. This GETVIS area is used by SVA programs that were written to dynamically acquire virtual storage when it is required. When one of these programs requests virtual storage, it is obtained from the GETVIS area. When the storage is no longer required, it can be dynamically released for use by other programs.

Many users never write programs that run in the shared virtual area. There are, however, many IBM provided program phases that do use the facilities of the SVA.

Exercise 3.2

1. The _____ feature prevents programs from interfering with each other's operations under DOS/VSE.
 - a. background protect
 - b. supervisor call
 - c. storage protect
 - d. multipartition
2. Multiprogramming is designed to make efficient use of _____.
 - a. I/O devices
 - b. CPU cycles
 - c. program interrupts
 - d. operator intervention
3. DOS/VSE allows the user to divide the problem program area into as many as _____ partitions.
 - a. 2
 - b. 3
 - c. 4
 - d. 5
4. Explain in your own words the reason for a priority system in a multiprogramming environment. Hint: What if two programs are ready to run at the same time?
5. By default, the highest priority partition under DOS/VSE is the _____ partition.
6. An I/O bound program is one in which there is a great deal of I/O activity, while a CPU bound program is one in which there is a minimum of I/O activity and a great deal of CPU calculations.

In a multipartition system would you chose to put I/O bound or a CPU bound program in the highest priority partition? Explain your answer.
7. The default priorities may be changed by _____.
 - a. an operator command
 - b. a partition override
 - c. the supervisor
 - d. a problem program

Solution 3.2

1. c
2. b
3. d
4. In an environment where more than one program may be active simultaneously, a priority system is necessary to resolve situations when more than one of these programs is ready to use the CPU at the same time.
5. Foreground - 1 or F1
6. An I/O bound program should go into the highest priority partition. The I/O bound program allows for interrupts to occur as it waits for completion of its input/output requests, and the lower priority partitions can get control of the CPU at these times.

If a CPU bound program were in the highest priority partition, it would not allow for a sufficient number of interrupts to take place so that other partitions sharing the CPU could gain control within reasonable periods of time.

7. a

**Topic - 3 DOS/VSE
Virtual Storage
Implementation**

The concepts of virtual storage were discussed in Unit 1 of this text. This topic reviews these concepts and describes the DOS/VSE implementation of virtual storage.

Page Data Set

Programs to be executed in a virtual storage environment are sectioned by DOS/VSE into 2k byte blocks called program pages. If the available processor storage can not contain all of the pages of a program, DOS/VSE writes pages not currently required for program execution to a disk file called the page data set. Figure 3.3.1 shows the relationship of the page data set and processor storage; the illustration may help you to understand the subsequent discussion. Notice that virtual storage is divided into partitions.

You should keep in mind that virtual storage does not exist in a physical sense. Virtual storage is a concept. When implemented, it provides the capability to execute programs that are larger than the available processor storage. The page data set is part of the implementation of virtual storage. The storage capacity of the page data set is equal to the size of virtual storage.

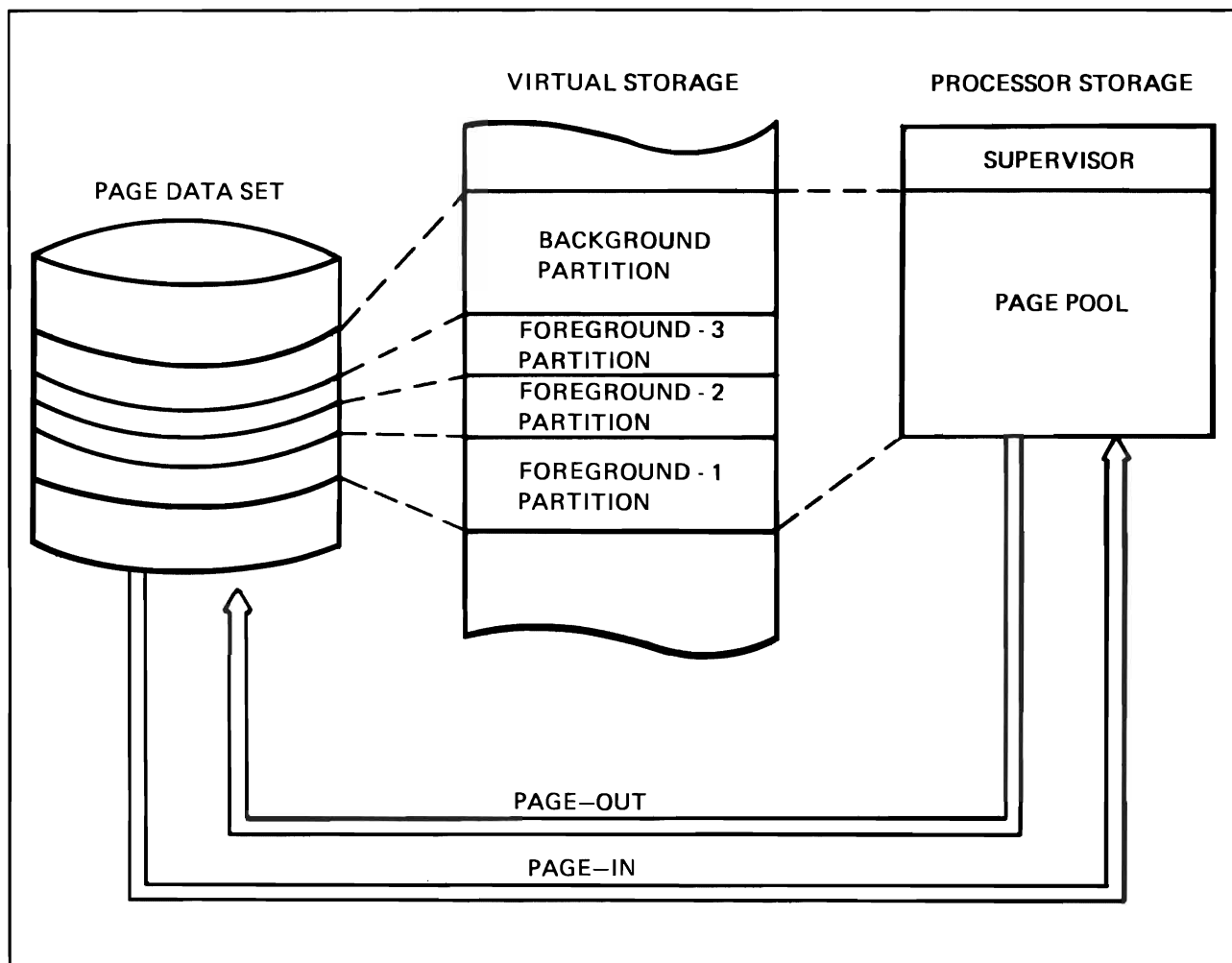


Figure 3.3.1 Relationship of Page Data Set, Virtual Storage, and Processor Storage.

- Page In* If, during program execution, reference is made to a program page which is not in processor storage, DOS/VSE automatically suspends execution of the program, retrieves that page from the page data set and places it into a 2K-byte section (page frame) of processor storage. Once this operation, which is referred to as page-in, is complete, execution of the interrupted program continues.
- Page Out* If no page frame is available in processor storage to accommodate a program page needed for continued execution of a program, DOS/VSE automatically frees a page frame. To do this, DOS/VSE selects a program page not recently referenced and writes it to the page data set if necessary. This operation (also called page-out) is necessary if the page was altered while in processor storage or if it had not been written to the page data set previously.
- Page Pool* Page frames that are not used by the supervisor nor are needed for the execution of programs in real mode - (real mode is discussed below) - comprise the system's page pool. Figure 3.3.2 illustrates the execution of four programs in a virtual storage environment. Note that while programs in virtual storage are each one continuous block of instructions and/or data, their pages are scattered randomly throughout processor storage. Keeping track of the pages of a program in processor storage is a function of DOS/VSE and a feature of the processor called dynamic address translation (DAT). When DOS/VSE is executing in a 4300 processor in ECPS:VSE mode the DAT feature is not used. ECPS:VSE mode uses an improved virtual address translation capability.

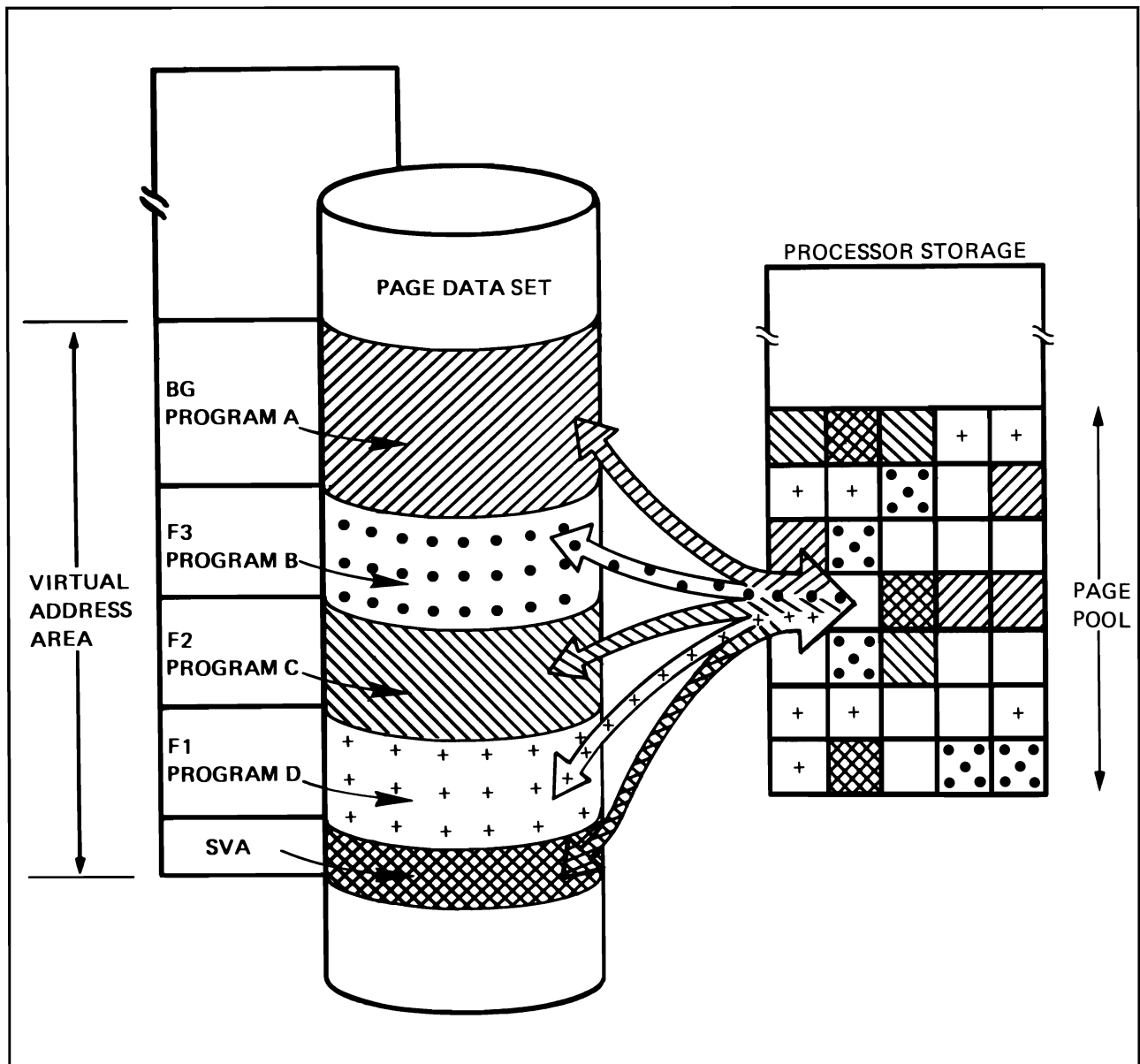


Figure 3.3.2 Four Programs Executing in a Virtual Environment.

Real Mode

For reasons that are beyond the scope of this text, some programs cannot tolerate page-in or page-out operations. For these programs, execution in real mode is required.

For the execution of a program in real mode, you must allocate a sufficiently large area of processor storage to the partitions in which the program is to run. The page frames used by that program are taken away from the page pool; they are therefore no longer available for the execution of programs which run in other partitions in virtual (not real) mode.

In general, virtual storage organization is meant to be "transparent" to the user, which means you act and write programs as if you had much more storage available than you really do and let DOS/VSE handle the implemen-

tation. The application programmer does not usually need to be concerned with any greater level of detail than has been presented here.

Exercise 3.3

1. The disk file where program pages are written in order to make room in processor storage for pages required for program execution is called _____.
 - a. page in file
 - b. page out file
 - c. page data set
 - d. SYSRES
2. Page frames that are not used by the supervisor and are not required for execution of programs in the real mode are referred to as the _____.
 - a. program pool
 - b. page pool
 - c. virtual space
 - d. page data set
3. True or False - The page frames used for program execution must be contiguous.
4. With DOS/VSE, processor storage is divided into 2 k byte sections called _____.
 - a. virtual storage
 - b. page frames
 - c. pages
5. True or False: The DOS/VSE page data set is divided into partitions.

Solution 3.3

1. c
2. b
3. False
4. b
5. True

**Topic - 4 DOS/VSE
Libraries and Processing
Programs**

DOS/VSE provides libraries for storage of programs and information used in the operation and control of the data processing system. This topic describes the DOS/VSE library facilities and defines the types of processing programs that reside in these libraries.

There are four libraries available to the DOS/VSE user. These libraries are called the core image, relocatable, source statement, and procedure libraries. When any of these libraries reside on the portion of the SYSRES pack devoted to system functions (within the SYSRES area) it is known as a system library. When it exists outside this area, usually on another disk pack, it is called a private library. The only exception to this is that there can be no private Procedure Library unless VSE/Advanced Functions, is installed. The structure, organization and format of private libraries is otherwise identical to that of system libraries.

Figure 3.4.1 shows how private libraries can exist on SYSRES as well on other disk packs.

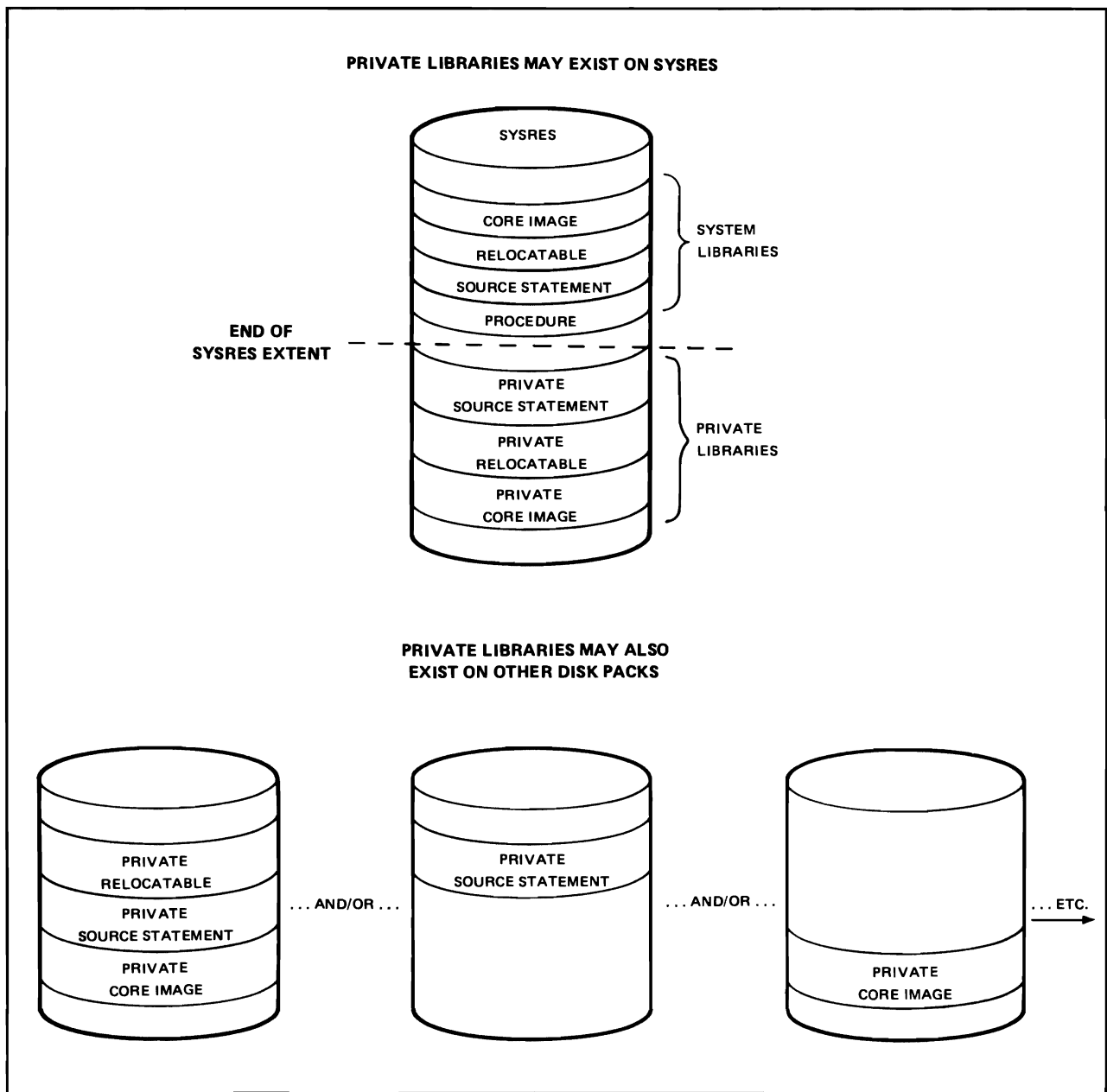


Figure 3.4.1. System and Private Libraries

Core Image Library

The system core image library is the only library that is always required. It contains the IPL, supervisor, and job control programs, as well as all the other programs that make up your DOS/VSE system.

Programs in the CIL are in executable format and are called phases. Any program to be executed under DOS/VSE must first be stored in a CIL. The supervisor will load programs for execution only from a CIL, be it system or private.

Relocatable Library When a source program is processed by a language translator (any of the compilers or the Assembler), a machine language version of the program called an object module is produced. The DOS/VSE library used for the storage of object modules is the relocatable library (RL). IBM supplied data management routines for data transfer to and from I/O devices are normally stored in the relocatable library. These I/O logic modules are automatically included as part of your program by the linkage editor program which is the IBM supplied program used to construct the executable phases that reside in the CIL.

Source Statement Library When you write a new program, you can place the source code in the source statement library (SSL). Programs stored in the SSL are called books. Books can be copied from the SSL into source programs, and individual statements may be added to, deleted from, or updated within books in the SSL. This means that a source program may be maintained in the SSL and that maintenance of the program in card deck form (or on diskettes) is not necessary.

Procedure Library Frequently used sets of job control statements may be stored in the procedure library. They are then called procedures. A procedure can be invoked from the procedure library with a single job control statement. For those jobs that are run frequently, procedures reduce the volume of control cards to be read into a system. This reduces card handling by the system operator and improves availability of your system's card readers.

Library Structure The four libraries are structured as illustrated in Figure 3.4.2.

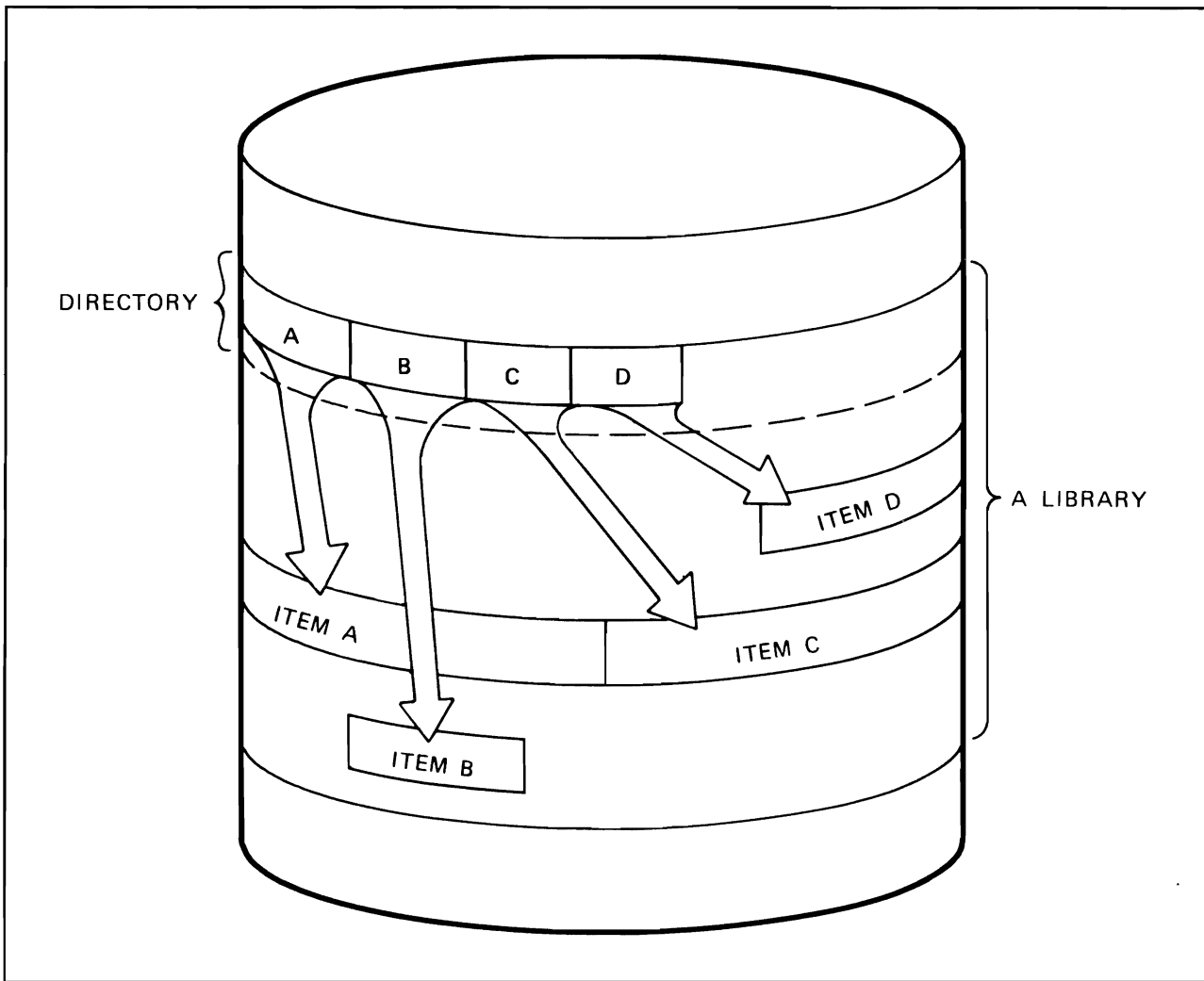


Figure 3.4.2 Structure of the Libraries

A directory at the beginning of each library contains an entry for each member in the library. Thus, the CIL directory contains an entry for each phase in the core image library, the RL directory contains an entry for each module in the relocatable library, and so forth. Directory entries contain the name of the member they reference and the location of that member in the library proper. A directory entry is made automatically when a member is added to a library.

Private Libraries

Private libraries can be viewed as extensions of the system libraries. Individual users or user departments can maintain phases, modules, or books in separate private libraries that are independent of each other and the system libraries. In this way there is no threat of running out of room in the system libraries.

Programs being written or revised and undergoing testing can be kept in private libraries without interfering with any production programs (programs that are already tested and are operational) that reside in the system libraries. Once development is completed and the programs are ready for operational status, they may be retained in the private library, or they may be copied into

the system library. Whether a given program is to reside in a system or private library is the decision of individual users.

Processing Programs

The DOS/VSE system control programs (IPL, the supervisor, and job control) control the execution of another class of programs called processing programs. These consist of application programs, service programs, and language translators.

Application Programs

An application program (user program) is a program that applies to the work in your installation. Most application programs are written by DOS/VSE users, while some are obtained from software vendors, such as IBM, on a purchase or lease basis.

Service Programs

Service programs assist in the use of DOS/VSE without contributing directly to the control of the system. Examples are the librarian programs, the linkage editor, and the system utility programs.

Librarian programs

Librarian programs are used to maintain the DOS/VSE libraries and to display their contents. Some of the functions provided by the librarian programs include:

- Allocation of space for a library on disk.
- Addition of modules, books and procedures to their respective libraries.
- Deletion of phases, modules, books and procedures from their respective libraries.
- Printing the contents of the library directories.

Linkage Editor Program

The linkage editor processes object modules to produce phases, which it places in a core image library.

System Utility Programs

These programs perform functions such as copying data from one volume to another and preparing volumes for the storage of data.

Data Management Routines

These routines are available for use by your programs whenever they need to write or read data to or from an external storage device. They consist of Physical Input/Output Control System (PIOCS) routines and Logical Input/Output Control System (LIOCS) routines.

PIOCS

The PIOCS routines are located in the supervisor. They work in conjunction with the LIOCS routines to perform the work associated with transferring data between external storage (tapes, disks, printers, etc.) and main storage. It is the PIOCS routines that actually issue the commands that cause this data transfer to take place.

LIOCS

These routines do the blocking and de-blocking of records, and pass requests to PIOCS to perform data transfers to and from I/O devices. The LIOCS routines supplied by IBM are normally maintained in the system relocatable library, and are automatically included as part of your program by the linkage editor program that prepares your program for execution.

Language Translators

Language translators convert source language programs to machine language programs called object modules. The Assembler language translator is included with the DOS/VSE system. Other translators are available on a rental or lease basis.

Exercise 3.4 Complete this exercise before going on to the next topic. If you miss any of these questions, review the appropriate material in the text.

1. The library that cannot exist on a private basis unless VSE/Advanced Functions is installed is the _____.
 - a. Core Image Library
 - b. Relocatable Library
 - c. Source Statement Library
 - d. Procedure Library
2. To be loaded for execution a program must reside in a _____.
 - a. Core Image Library
 - b. Relocatable Library
 - c. Source Statement Library
 - d. Procedure Library
3. The output of a language translator is a module that could be placed in the _____.
 - a. Core Image Library
 - b. Relocatable Library
 - c. Source Statement Library
 - d. Procedure Library
4. Frequently used sets of job control statements may be stored in the _____.
 - a. Core Image Library
 - b. Relocatable Library
 - c. Source Statement Library
 - d. Procedure Library
5. The linkage editor and the librarian are examples of _____.
 - a. Application Programs
 - b. Service Programs
 - c. Control Programs
 - d. Language Translators

Solution 3.4

1. d
2. a
3. b
4. d
5. b

Topic - 5 VSE/POWER

VSE/POWER (Virtual Storage Extended/Priority Output Writers, Execution Processors and Input Readers) is a program product that provides spooling for the DOS/VSE user. This topic describes how VSE/POWER spooling optimizes the utilization of unit record I/O devices. Other VSE/POWER facilities including remote job entry support are also discussed.

Concepts and Functions

In all computing systems, there is a large discrepancy between central processor execution speeds, which are electronic and therefore very high, and the speeds of unit record I/O devices such as card readers, punches, and printers, which are largely mechanical and relatively slow. This discrepancy has a potentially negative impact on throughput (the amount of work accomplished in a unit of time). VSE/POWER is designed to improve throughput in a DOS/VSE multiprogramming environment by separating internal computing operations from the unit record I/O functions.

Spooling

VSE/POWER decreases the execution time of unit record I/O-bound jobs by servicing input and output requests addressed to such devices at disk I/O speed. The reading and punching of cards and the printing of listings is done by VSE/POWER during the execution of other jobs. The additional processor time used by VSE/POWER is negligible.

To achieve this, VSE/POWER stores all unit record input and output in disk queues. When a unit record device is available to read input or punch or print output, VSE/POWER either reads the data into an input queue or routes punch or print output from an output queue to the appropriate output device. When an application program in a VSE/POWER controlled partition requests input, VSE/POWER supplies the next record in the input stream for this particular job from the input queue. When an application program writes an output record to a unit record device under its control, VSE/POWER intercepts the output request and writes the record to an output queue.

This process which is called "spooling" (an acronym derived from the term "Simultaneous Peripheral Operations On-Line"), results in overlapping the unit record I/O operations for all controlled partitions with the execution of application programs in these partitions. Figure 3.5.1 describes the spooling process in an example where only one partition is controlled by VSE/POWER.

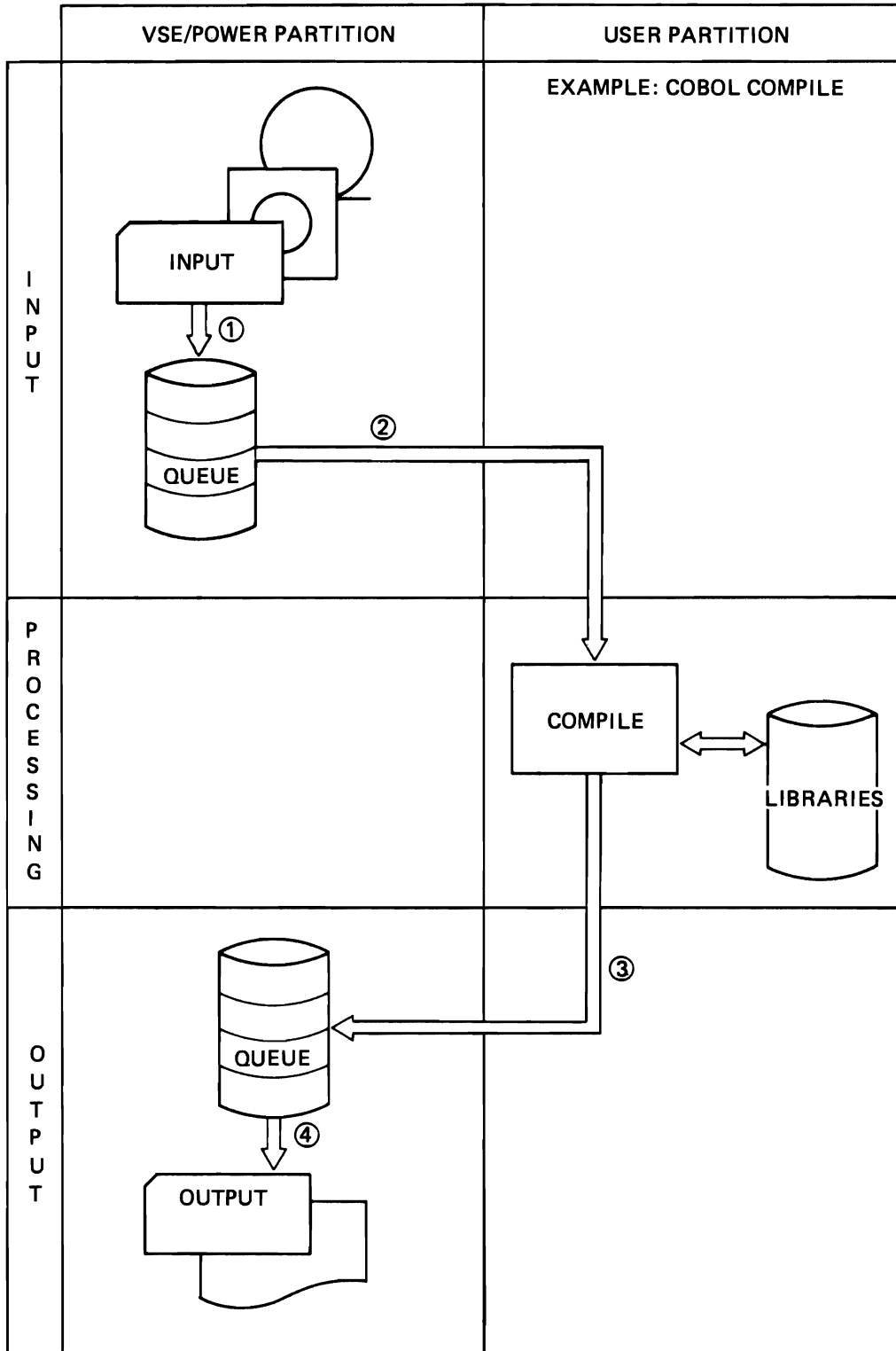


Figure 3.5.1 DOS/VSE Operation with VSE/POWER

The example shows VSE/POWER reading a COBOL source program from a either a card reader or diskette. The input is written to a VSE/POWER input queue and held for execution. When the user partition becomes available, the program can be released for execution.

During execution, the COBOL compiler reads the source program from the input queue, spools the printed output to the printer output queue and spools the object program to the punch output queue.

The output is held in the printer and punch output queues until the printer and punch are available. Printing and punching is performed after the program has completed execution.

The spooling of unit record I/O has another throughput advantage in that VSE/POWER can operate these devices at or near their maximum rated speed. This is often not the case when the unit record I/O is not spooled.

In addition to increasing the user's job throughput, VSE/POWER offers the DOS/VSE user the advantage that he needs to provide only one group of unit record devices for all partitions under its control. The unit record devices used by VSE/POWER can provide the unit record I/O requirements for all the partitions that are being serviced by VSE/POWER.

Figure 3.5.2 shows the operation of a DOS/VSE controlled IBM data processing system within a given period of time. In its upper part, this figure illustrates the three COBOL compiler executions without VSE/POWER assistance. The lower part of Figure 3.5.2 shows the same three executions under control of VSE/POWER.

Without the VSE/POWER function, the compiler runs in one partition that has unit record input and output devices associated with it. During data input and output (I1, I2, I3, O1, O2, O3), the CPU is not used by this partition and is therefore available to other partitions. However, the input and output devices, when idle, are not available to other partitions. The total processing time is indicated by the time tags A (beginning) and E (end).

With the VSE/POWER function, the compiler runs in one partition that has no unit record input and output devices associated with it. These devices are associated with the partition where VSE/POWER is executed. This figure shows the overlapping operations -- input and output in the VSE/POWER partition and processing in the COBOL (example) partition. Again, the CPU time not used by the tasks X1, X2, and X3 (compiles) is available to other partitions. In addition, the unit record input and output devices are available to other VSE/POWER controlled partitions; the card reader between the time tags C and E, and the printer between A and B and D and E. Finally, processing of all three jobs is finished at tag D, that means, earlier than in an identical computing system environment not controlled by VSE/POWER.

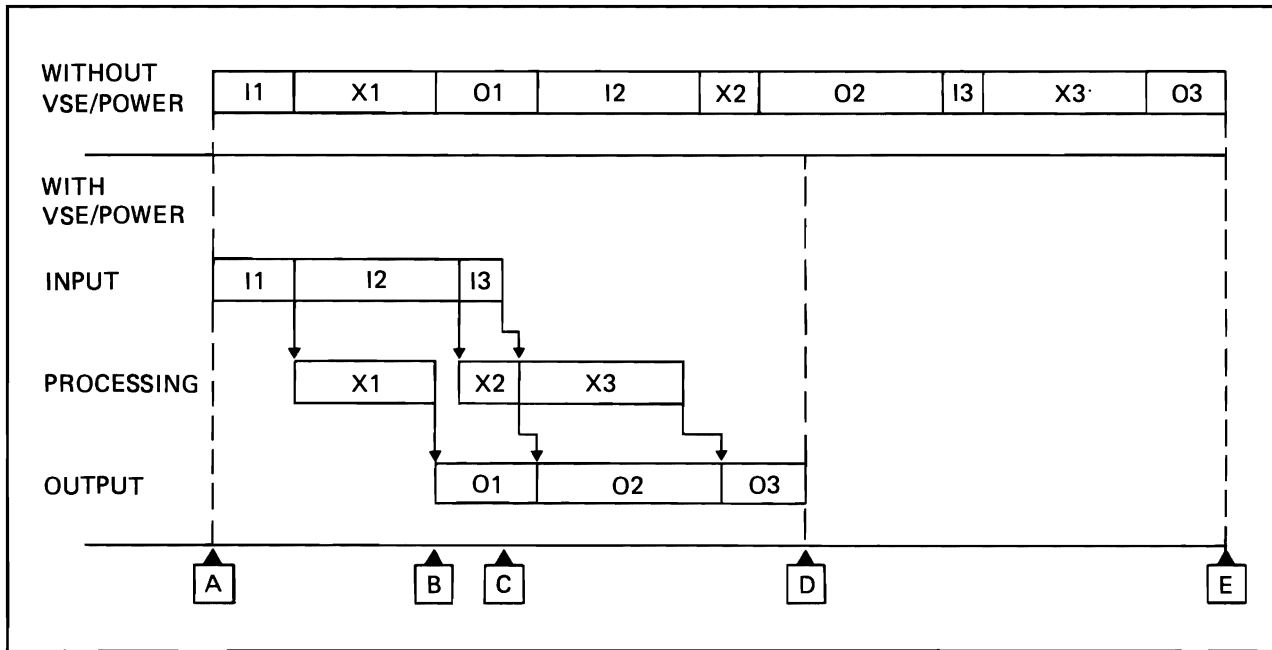


Figure 3.5.2 Overlapping Execution with VSE/POWER

Remote Job Entry

VSE/POWER also allows the user to expand his local job processing environment with input entered from remote terminals. This process is called Remote Job Entry (RJE). VSE/POWER, further offers the user the possibility to decide where the output from a remotely entered job should be directed; to the originating terminal, to another terminal, or to a central unit record output device. VSE/POWER offers these and other advantages without requiring the user to modify his DOS/VSE jobs.

Job Entry Control Language

The VSE/POWER job entry control language (JECL) can be used to write commands to describe individual job execution or I/O requirements. These commands are included with the input job descriptions that are spooled to disk. Some of the functions that can be controlled by JECL commands include:

- Specification of the partition in which the job is to be executed.
- Assignment of job execution priorities.
- Specification of the number of print copies required.

Operator Commands

From the reading of input until the final printing of output, the operator may start and stop the job execution, change the output characteristics of jobs in the output queues, or start and stop physical I/O devices. These functions are accomplished by the entry of VSE/POWER operator commands through the system console device.

Exercise 3.5

1. Briefly describe how VSE/Power can service the printing requirements for two or more programs when there is only one printer.
2. True or False: If the system printer is disabled, the jobs running under the control of VSE/Power can continue to execute.
3. VSE/Power Job Entry Control Language (JECL) can be used to:
 - a) specify the partition in which a job will be executed
 - b) assign job execution priorities
 - c) specify the number of printed copies
 - d) all of the above
 - e) none of the above
4. True or False: Operator commands can be used to change the output characteristics of the jobs in the output queues.
5. VSE/Power enhances throughput by:
 - a) overlapping program execution with unit record I/O
 - b) overlapping program execution with disk I/O
 - c) all of the above
 - d) none of the above

Solution 3.5

1. Program print requests are written to disk queues (spooling). VSE/Power prints output for the jobs one at a time. VSE/POWER can continue to spool program I/O for other programs while it is printing.
2. True. The printed output is spooled to disk during execution.
3. d
4. True
5. a

Topic - 6 DOS/VSE Data Management

Input and output requirements for programs are handled by program routines called access methods. These access methods transfer data between main storage and external storage devices. DOS/VSE data management provides a set of access methods for frequently required I/O functions. These IBM supplied access methods relieve the programmer from the details of I/O programming.

This topic provides a basic description of the DOS/VSE access methods and the Virtual Storage Access Method (VSAM) program product. This topic also describes some of the data protection facilities of DOS/VSE.

Data Organization and Access Methods

Data Organization

Data organization refers to the techniques used in placing records on an auxiliary storage device (magnetic tape or disk, for example). It involves such considerations as:

- The choice of storage media best suited to the processing requirements of the data.
- The sequence of records in the file. For example: the records could be sorted on one control field or on several, they could be in ascending or descending order.
- The length of records. Records in a file can be of fixed length or of variable length; they can be relatively short or relatively long.
- The blocking factor for the file. This determines how many logical records constitute a block of records (for transfer from the central processor to the I/O device and vice versa); and it may be an important factor in storage media utilization and in processing efficiency.
- The use of indexes with a file on a direct access device to provide an efficient means of randomly selecting specific records.
- The use of programmed addressing techniques to determine where a record is stored on a direct access device and the location from which the record can subsequently be retrieved for processing.

Access Method

Access method refers to the routines which assist the programmer in transferring records in a particular data organization format between processor storage and an I/O device. It is important to understand the relationship between data organization and access methods. Broadly speaking, how data is organized and the type of device on which this data is stored largely determine the access method that can subsequently be used to retrieve that data.

DOS/VSE provides several methods of data organization. For each of these, there is an access method which allows one or more techniques of file creation and retrieval. These data organizations and their associated access methods are briefly described below.

*Sequential Access Method
(SAM) and Organization*

File Organization

Sequential organization means that records physically follow one another in a sequence usually determined by one or more control fields within each record. Examples of control fields are employee number in a personnel file, or part number in an inventory file.

Sequential organization is supported for all device types except telecommunication terminals. Card files, print files, diskette files, and magnetic tape files are always organized sequentially, simply because the physical characteristics of those devices require the reading or writing of one record after another. Files on disk may also be organized sequentially, in control number sequence.

Data Access

If required, records are sorted into their prescribed sequence prior to creating a sequentially organized file. The Sequential Access Method (SAM) can create a sequential file from sorted records and subsequently retrieve those records for sequential processing.

Records from sequential disk files may be updated, meaning that each record may be written back onto its original location after having been changed by the program. By contrast, in order to update records stored on tape, a new tape must be created by a program that reads the old tape and re-writes the records to the new tape. The records that are updated are changed before they are written to the new tape.

*Indexed Sequential Access
Method (ISAM) and
Organization*

The physical characteristics of a disk make it practical to retrieve a record from any location in a file instead of having to go from one record to the next starting at the beginning of the file until the desired record has been located. DOS/VSE exploits these characteristics through the indexed sequential access method and organization.

Index and File Organization

An indexed sequential file is made up of (1) records in logical sequence by control field (or key) and (2) indexes that are built when the file is created. Each index entry is composed of the key of a data record and the physical address at which the record is located on the disk. The programmer may process indexed sequential files sequentially when the definition of the programming application requires this approach, or he may process them randomly, retrieving a particular record from the file, if this approach is better adapted to the requirements of the job. He may even combine both approaches in the same program. Both retrieval methods are easy from the programmer's viewpoint. For example:

- For sequential retrieval, he simply issues the appropriate I/O statement at the point in his program where he requires the next record, and ISAM makes the record available to the program.
- For random retrieval, the programmer merely provides the key of the record he needs, issues the appropriate statement and ISAM presents

the specific record to the program for processing. Index searching, deblocking records, and handling device requirements are all accomplished internally by the access method.

***Direct Access Method
(DAM) and Organization***

While ISAM offers both sequential and random retrieval, DAM concentrates on random retrieval only and provides an access method that can accomplish this function efficiently. DAM requires, however, that you establish a direct relationship between the keys of the records and their physical addresses on disk. This means that the program, by using the key of a record, can calculate or look up in a table the corresponding record address, and either directly store the record (on output) or directly retrieve it (on input). Greater burden and responsibility is placed on the programmer; the benefit is a potentially faster record retrieval for specialized applications.

In a DAM file:

- Records normally are not in logical sequence by key.
- Tables to accomplish retrieval functions are not built and maintained by the access method.
- The physical location of each record on the disk is determined by an algorithm in the program that accesses the file. The programmer must write this algorithm.
- Depending on the addressing technique used by the programmer, gaps may exist; also, the addressing technique could produce synonyms, which are multiple records for the same address. The programmer is responsible for solving these problems.

DAM is available for programs that process data stored on DOS/VSE supported count-key-data disk devices.

***VSE/Virtual Storage
Access Method
(VSE/VSAM)***

The VSE/VSAM program product is an access method covering a maximum of possibilities for direct and sequential processing of fixed and variable length records on direct access devices. VSE/VSAM offers considerably more functions than the access methods available with DOS/VSE. It provides improved data security and integrity and a more flexible data organization.

File Organization

The records in a VSAM file can be organized in logical sequence by a key field (key-sequenced file), in the physical sequence in which they are written into the file (entry-sequenced file), or according to the relative record numbers in the file (relative record file).

VSE/VSAM allows retrieval, storage, update, and deletion of records; it can handle variable length as well as fixed length records. The access can be either sequential or direct. It can be by record key, record address, or relative record number. The key can be that of an individual record or it can be a generic key specifying a group of individual records. Blocking and deblocking of records is done by the access method which optimizes block length to suit the device on which the file is written.

With a key-sequenced file, several records in sequence can be inserted as a group at one point in the file (this is faster than inserting them one at a time with direct access, as ISAM requires). Also, VSE/VSAM allows accessing several records in key-sequence and then have VSE/VSAM skip to another portion of the file and access more records in sequence without having to search the entire index to find the new group of records (this is called skip-sequential access). For keyed or addressed sequential (but not skip-sequential) processing, there is also an option to process records backwards.

VSAM Catalogs

VSE/VSAM keeps control over the creation, access, and deletion of files and over the direct-access storage space allocated to those files. This is done by keeping information on file characteristics in a VSAM catalog. There are two kinds of VSAM catalogs, master and user catalogs. One master catalog is required with VSE/VSAM; any number of user catalogs are optional. User catalogs increase data integrity and security.

VSE/VSAM Space Management for SAM Feature

With this optional feature, the control functions of the VSAM catalog are made available to sequential access method (SAM) users. SAM files may be defined and processed within VSAM data space, thereby providing automatic space management of the file. In addition, automatic secondary space allocation is possible for SAM files. This feature makes it possible to increase the file size without having to recreate the file. Also, SAM files can be accessed by VSE/VSAM, thereby enhancing programming flexibility. Through the use of VSE/VSAM Space Management, the SAM user is provided with the improved data security features available from VSE/VSAM.

VSE/VSAM Device Independence

With VSE/VSAM, all device related data for a file is stored in the VSAM catalog which is maintained separately from application programs. This means that a VSAM file can be moved to a different DASD device type without having to change the programs that access that file. This capability is called device independence.

ISAM Interface Program (IIP)

Programs that were written to process data through ISAM can also process data contained in VSAM files without changing the program. VSE/VSAM uses the ISAM Interface Program (IIP) to provide this capability.

Data Security/Data Integrity

VSE/VSAM offers improved data security and integrity. A file can be protected against unauthorized use through passwords. The various password options available grant authority to read a file, read and update a file, or to read and update both a file and the VSAM catalog.

Data integrity is improved by minimizing data movement and index updating when records are added, by preserving both new and old indexes to the data until an update is completed, and by a provision for recovery from damage to the catalog.

VSE/VSAM Summary

In conclusion, VSE/VSAM provides the most complete data access method available for processing DASD. VSE/VSAM was designed to make optimum use of the facilities provided by the DOS/VSE operating system. With VSE/VSAM, the other IBM access methods are not necessary for DASD unless the user desires to continue processing existing files that were created using one of the other access methods.

***Telecommunication
Access Methods***

In telecommunication, input and output of data to and from the computer is performed using terminals, comparable to I/O devices, which are connected to the installation's central processor through communication lines or local control units.

The telecommunication access methods you would normally use under DOS/VSE are: the Basic Telecommunication Access Method - Extended Support (BTAM-ES) and the Advanced Communications Function/Virtual Telecommunications Access Method Entry (ACF/VTAME). Both of these access methods are available as licensed IBM programming support.

Normally, you would use the services of these telecommunications access methods through other IBM licensed programming support such as VSE/Interactive Computing and Control Facility (VSE/ICCF) or the Customer Information Control System (CICS/VS) or both. This licensed programming support relieves the programmer of the burden of coding telecommunication access routines and yet allows for utilizing all of your computing system's services via a terminal.

Protection of Data

It is important, to protect an installation's data, programs and files against accidental or intentional misuse or destruction. Under DOS/VSE, programs may attempt to retrieve and update the same file and or even the same record simultaneously; moreover, the installation's online disk storage may contain classified data that must be protected.

DOS/VSE provides facilities that help to protect your data against misuse or destruction. Some of these facilities are:

- File labeling
- Protection against duplicate assignment
- Secured DASD files
- Track Hold Specification

File labeling

File labels assist in achieving file protection. File labels are records associated with the files stored on magnetic tape, on disk, or on diskette. These labels provide unique identification for the files, and the volumes of these files. (A volume is a storage device such as a disk or tape.)

For data on magnetic tape, all labels are optional; for data on disk or diskette, each volume must have one volume label, and each file must have a file label.

For the creation of a file, you specify the contents of the labels. Then, when the file is processed as input (see Figure 3.6.1.), you again specify, in job

control statements, the contents of the label, so that the data management routines can compare the specified data with the actual label. If data management detects a mismatch between the actual label and the label information, the job is terminated.

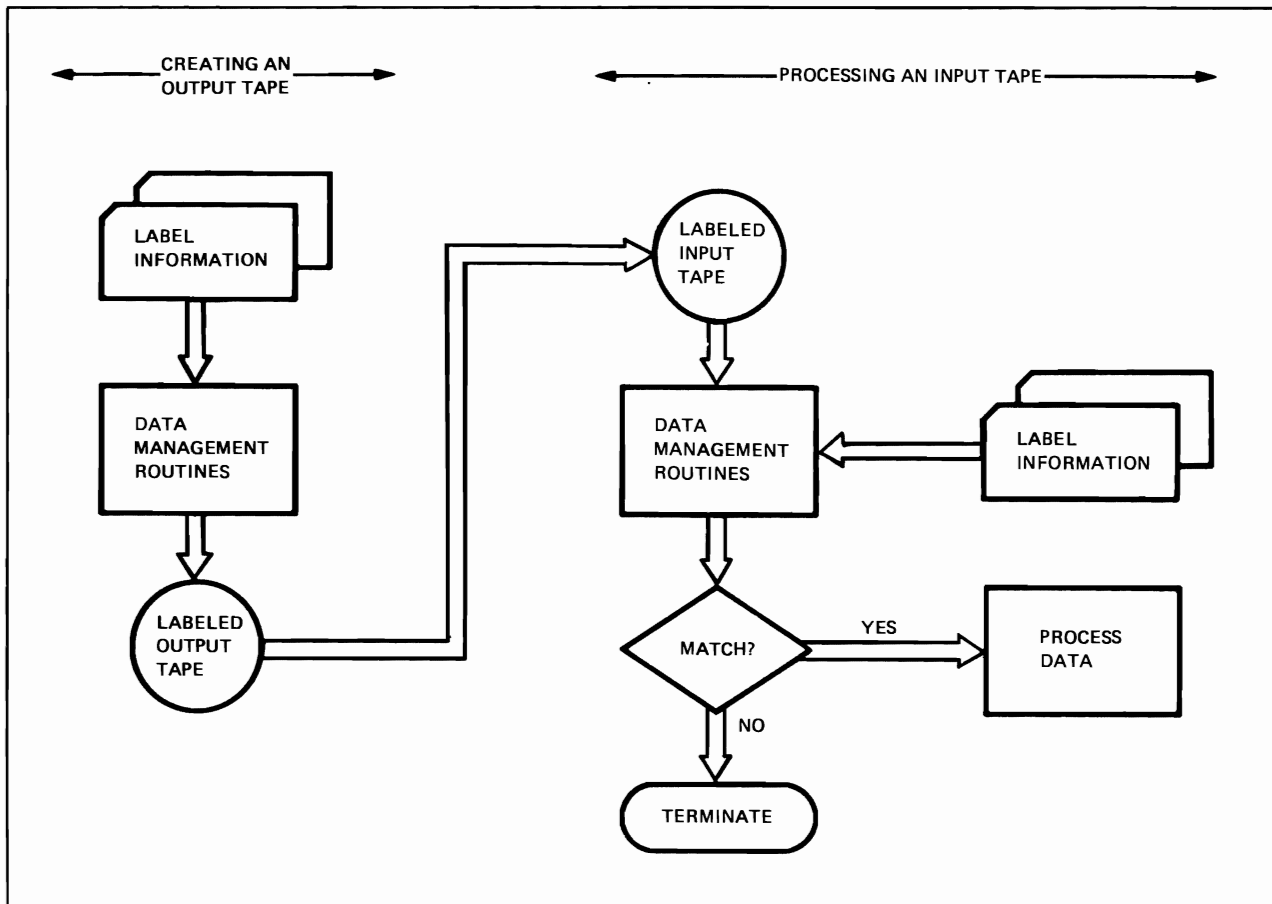


Figure 3.6.1 Label Processing

Protection Against Duplicate Assignments

In a multiprogramming environment, it is essential that no two programs in different partitions gain access to the same device, except for disk units. A single disk unit may contain a number of files; therefore, disk units can contain files for processing by programs in more than one partition. DOS/VSE prevents the assignment of non-disk devices to more than one partition. The only exception is that card and printer devices can be assigned to multiple partitions under the control of VSE/POWER.

File Security

When you create an output file on disk, you can request, by means of a job control statement, that the file is to be secured. The operator is notified with a message when a secured file is being used as input. He can then make the decision as to whether the file should be processed.

Track Hold Specification

This DOS/VSE facility contributes to data integrity. It prevents two or more different programs from updating the same track (or block) on a DASD concurrently. The track (or block) is then held for the first program that accesses it and is not freed until that program has finished processing for this track or block.

Exercise 3.6

1. Program routines used to transfer data between main storage and external storage devices are called _____.
 - a) data routines
 - b) ISAM methods
 - c) access methods
2. The file organization that supports all device types except telecommunication terminals is called _____ organization.
3. The access methods that offer both sequential and random retrieval are _____ and _____.
4. True or False: Disk and diskette files require labels but tape files do not require labels.
5. The access method that keeps control over the creation, access, and deletion of files and controls the direct access space allocated to those files by keeping information on file and space characteristics in a catalog is _____.

Solution 3.6

1. c
2. sequential
3. Indexed Sequential Access Method (ISAM), Virtual Storage Access Method (VSAM)
4. True
5. Virtual Storage Access Method (VSAM)

Topic 7 Unit Summary

DOS/VSE is a disk operating system for virtual storage designed to efficiently manage the facilities of the IBM 4300 and System/370 data processing systems.

Three system control programs - IPL, the supervisor, and job control - are major components of DOS/VSE. The system is started by the IPL program; operation is controlled by the supervisor; and processing programs are prepared for execution by the job control program.

The SYSRES pack is mandatory for system operation and contains the four DOS/VSE libraries: Core Image, Relocatable, Source Statement and Procedure. The libraries respectively contain "phases", "object modules", "books", and "procedures". Except for the Procedure Library, they may be classified as system or private libraries.

The job control language is used to communicate processing requirements to DOS/VSE. Each job and its required resources must be defined by job control statements. This enables DOS/VSE to perform the basic functions of automatic job-to-job transition, assignment of I/O devices, and loading programs for execution.

Application programs, service programs, and language translators comprise a classification called processing programs; all may be efficiently controlled by DOS/VSE.

Your valuable CPU resource is handled efficiently by the facilities of multiprogramming and virtual storage organization. Multiprogramming allows multiple users (multiple programs) to share the CPU at the same time. DOS/VSE permits up to five partitions to execute programs concurrently with VSE/Advanced Functions up to twelve partitions can execute programs concurrently.

Virtual storage frees the application programmer from constraints previously imposed by the amount of machine storage physically present on the computer.

VSE/POWER provides spooling of unit record I/O. With VSE/POWER one set of unit record devices can be used to handle the card reading, punching and printing requirements for all programs running in partitions under its control.

DOS/VSE data management provides access methods for transferring data between main storage and external storage devices. Facilities are also provided to help protect an installation's files against accidental or intentional misuse or destruction.



Unit 4:

ICCF Facilities

Introduction This unit describes interactive computing and the productivity gains attainable through performing application program development, personal computing and problem solving, and system programming functions in an interactive environment. Interactive computing is provided to the DOS/VSE user by the Virtual Storage Extended/Interactive Computing and Control Facility program product. The major purpose of this unit is to describe the basic concepts and facilities of VSE/ICCF.

- Objectives** At the completion of this unit, you should be able to:
- Describe the process of interactive program development
 - Explain the interactive partition concept of ICCF
 - Describe the facilities available to the end user during a terminal session
 - Describe the structure and use of the ICCF libraries
 - Identify the six modes of operation available to the user to perform the basic functions of ICCF
 - Distinguish between the capabilities of the context editor and the full screen editor
 - Discuss the six categories of ICCF commands
 - Explain the purpose of ICCF procedures and macros
 - Describe the debugging assistance available to ICCF users
 - Discuss the data security features of ICCF
 - Describe the functioning of the interactive usability aids

Average Study Time 2 to 3 Hours

Topic-1. Interactive Computing

The process of an individual directly utilizing a computer via a terminal is defined as interactive computing.

Interactive computing differs from conventional batch computing in that it allows an individual to enter input into a computer from a terminal and to receive output, or responses to that input, back at the terminal. In effect, the terminal keyboard and display screen replace the card reader, printer and card punch of a batch computing system. Instead of entering input on cards through a card reader, an individual types the input into the keyboard of the terminal and, after processing by the computer, the output is displayed on the terminal screen instead of being printed on paper or punched into cards.

While the concept of interactive computing has almost unlimited applications, it has proved most useful in the areas of program development and maintenance, individual problem solving and education. Interactive computing allows a programmer, engineer, financial planner and computer operator to utilize the same central processing unit simultaneously.

To better understand the advantages of interactive computing, let's take a look at a programmer doing program development. This development may take the form of creating new programs or modifying code for programs that have been previously written.

Program Development With Batch Facilities

Program development in a batch environment is illustrated in Figure 4.1.1.

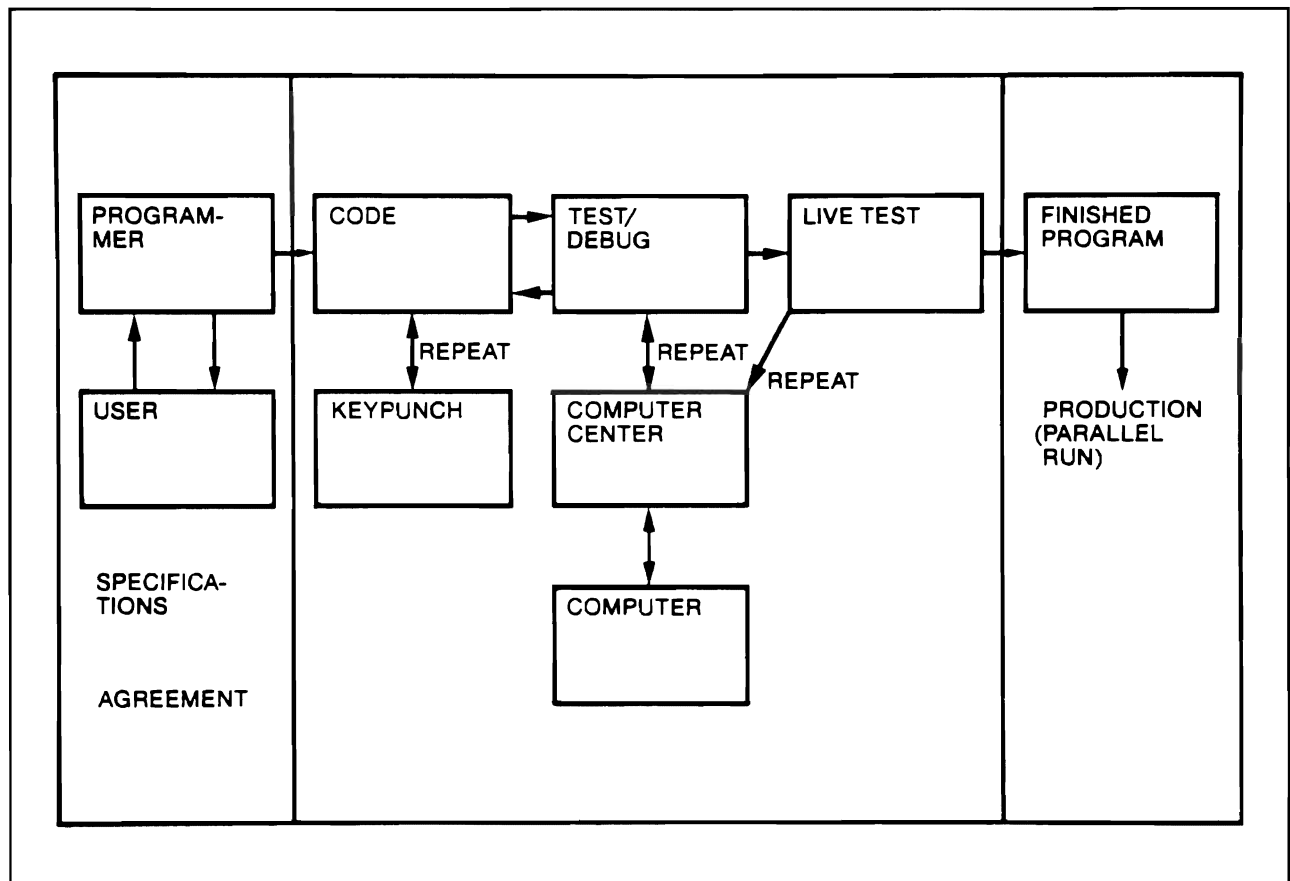


Figure 4.1.1 Traditional Program Development

The individual user requesting the program has to discuss with the programmer precisely what he wants done. Once they have reached agreement the programmer has to translate these ideas into a program by writing instructions on a coding sheet. The information contained on this coding sheet is then keypunched and verified. The programmer then schedules test time in order to compile the program. This compilation competes for computer time with other scheduled jobs within the data processing installation. Therefore, the results of the compilation may not be available to the programmer for a lengthy period of time. Upon receiving the output from the compiled program, the programmer corrects any errors and recompiles the program. After successful compilation, the programmer schedules test time and tests the program. Errors from the test are corrected and the program retested until the test is successful.

Interactive Program Development

Let's contrast program development in the batch environment with its implementation in an interactive environment. (Figure 4.1.2)

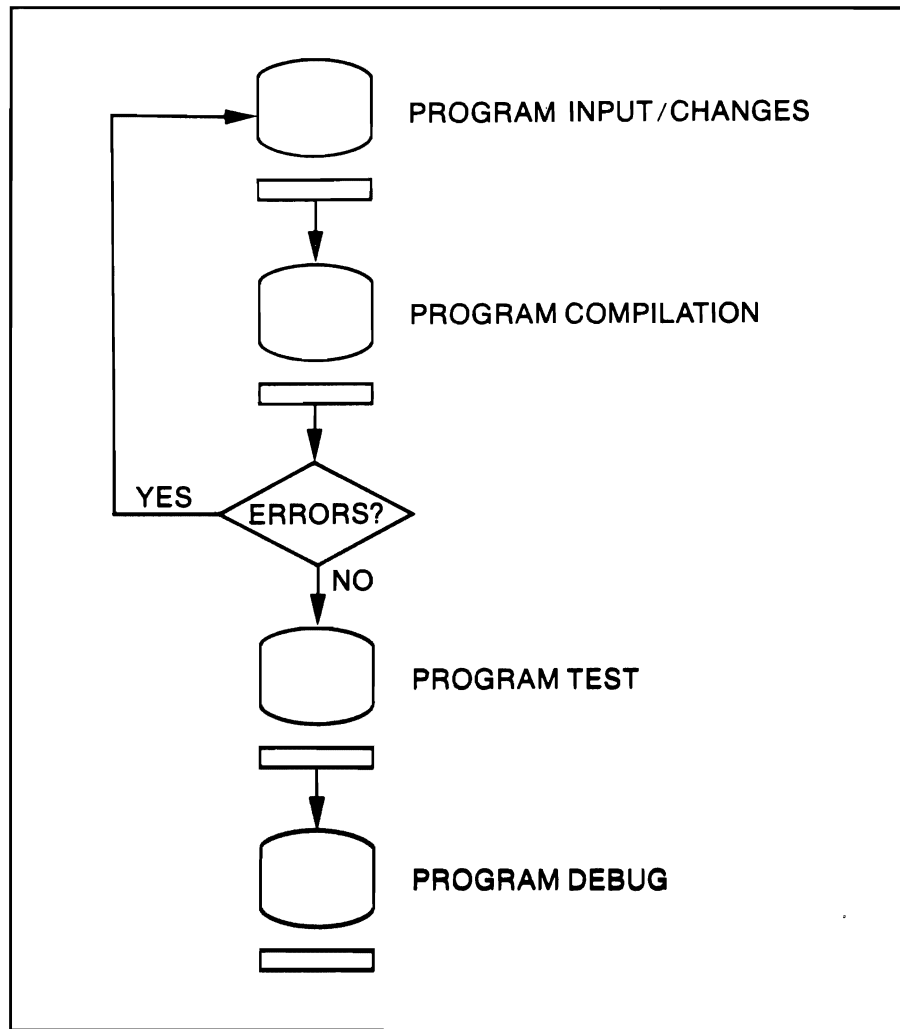


Figure 4.1.2 Interactive Program Development

The programmer is put in direct communication with the computer via a terminal. This terminal may be located at his desk or some other convenient area where the terminal may be shared. Rather than write code on a sheet of paper and have it keypunched, the programmer enters the coding directly into a file which is stored in the computer. Once the programmer is satisfied that the coding is correct, he merely selects the program from the file and asks the timesharing system to compile it. No wait, no delay. When the program has been compiled, the programmer may review the compilation output at the terminal. Now, what was formerly an overnight compilation job can be completed in minutes. The programmer may now select test information from the file which he created on the terminal, associate the data with the program, and test it. Should an error occur during program testing, the programmer can view the output of the traditional dump directly at the terminal. Corrections are made. The program is recompiled and testing continues. There is no waiting to schedule test time. No waiting for the compile. No waiting for the listing to be returned. All this is done with the programmer sitting at the terminal.

The preceding discussion clearly points out the productivity gains of interactive computing to the Data Processing professional during program development. Interactive computing provides similar productivity benefits to non Data Processing professionals in performing their jobs faster and more efficiently.

VSE/ICCF Interactive computing is available to all DOS/VSE users of the 4300 Processors through the Virtual Storage Extended/Interactive Computing and Control Facility program product (referred to as VSE/ICCF or ICCF). ICCF provides sophisticated computing facilities to allow the Data Processing professional to write, edit (change or update), compile and test programs without leaving the terminal. ICCF also enables the non Data Processing professional to solve complex problems at the terminal. The remainder of this unit will be a discussion of the functions and facilities of ICCF.

Exercise 4.1

1. _____ allows an individual to enter input into a computer from a terminal and to receive output back at the terminal.
2. (True or False) Interactive program development is the process of writing, editing, compiling and testing a program directly from a terminal attached to the computer.
3. Interactive computing is available to all DOS/VSE users of the 4300 Processors through _____.

Solution 4.1

1. Interactive computing
2. True
3. VSE/ICCF

Topic-2. Organization of ICCF

Let's first discuss ICCF's organization and its place in the DOS/VSE installation. ICCF executes in a DOS/VSE partition. Generally, ICCF executes in the highest priority partition or, if under control of VSE/POWER, in the second highest priority partition. The ICCF partition is composed of four major components as illustrated in Figure 4.2.1.

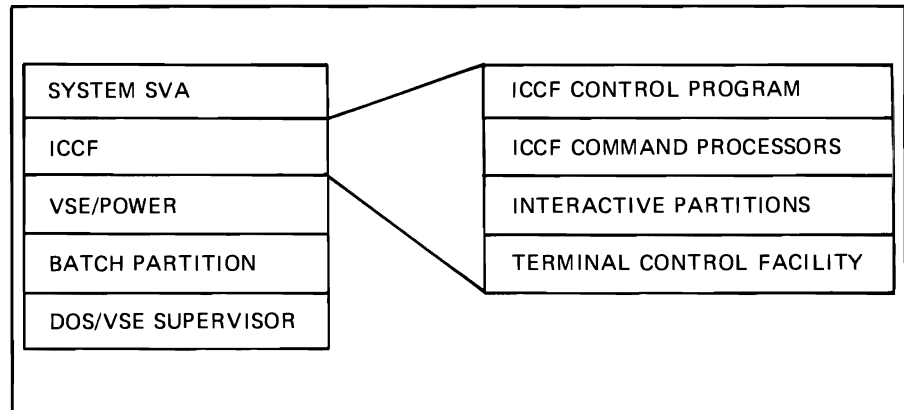


Figure 4.2.1 VSE/ICCF in the DOS/VSE Installation

ICCF Control Program

The first component is the ICCF control program. This is the system traffic cop supervising the activities of all ICCF components, except terminal I/O operations. The responsibilities of the control program include the scheduling of jobs for execution and the maintenance of the ICCF library file.

ICCF Command Processors

The ICCF command processors read and interpret commands (instructions to ICCF) and data from the terminal and perform requested functions. These functions could include the inputting, updating and listing of a program at a terminal. The functions handled by the command processors require little CPU time and, therefore, execute at a high priority. The command processors interact with the terminal control facility in performing these functions.

Terminal Control Facility

The terminal control facility provides the telecommunications interface between the terminal user and the terminal. The terminal control facility can be either the Customer Information Control System/Virtual Storage (CICS/VS) program product or the Terminal Transaction Facility (TTF) of ICCF. The Terminal Transaction Facility is provided as a separate program within ICCF.

ICCF Interactive Partitions

The final component of the ICCF partition is the interactive partitions. These interactive partitions are blocks of virtual storage where ICCF terminal users' jobs can be executed. Interactive partitions have characteristics similar to DOS/VSE partitions. Each interactive partition may have up to four classes of jobs associated with it to provide efficient job scheduling and system resource utilization. For example, some interactive partitions may be of a standard size, while others may be larger to accommodate special programs. ICCF can support up to 35 interactive partitions, plus one for the terminal control facility. A job executing in an interactive partition will contend for resources with other active jobs in interactive partitions. ICCF, through its control program, will attempt to balance the time and resources allocated to each active job within an interactive partition.

Exercise 4.2 The following choices are to be used in answering questions 1 through 6.

- a. ICCF control program
 - b. ICCF command processors
 - c. terminal control facility
 - d. interactive partition
1. The _____ read and interpret commands and data from the terminal and perform the requested functions.
 2. A block of virtual storage within the ICCF partition, where ICCF users' jobs are executed is called an _____.
 3. The _____ schedules jobs for execution and maintains the ICCF library file.
 4. The telecommunications interface between the terminal user and the terminal is provided by the _____.
 5. The _____ attempts to balance the time and resources allocated to each active job within an interactive partition.
 6. The _____ interact with the terminal control facility in performing their functions.

Solution 4.2

1. b - ICCF command processors
2. d - interactive partition
3. a - ICCF control program
4. c - terminal control facility
5. a - ICCF control program
6. b - ICCF command processors

Topic-3. End User's View of ICCF

Now that the organization of ICCF has been described, let's look at the facilities and capabilities of ICCF from the end user's view. Who are some typical users of ICCF?

- Application programmers using ICCF for interactive program development
- System programmers using ICCF for system software installation and maintenance
- Operations personnel using ICCF for submission and control of production jobs
- Non Data Processing professionals using ICCF for problem solving

To describe the user's perception of ICCF, let's assume that you, as the user, are an application programmer doing program development.

Upon logging onto ICCF from your terminal, you have the facilities, illustrated in Figure 4.3.1, available to you. Let's see how these facilities might be used.

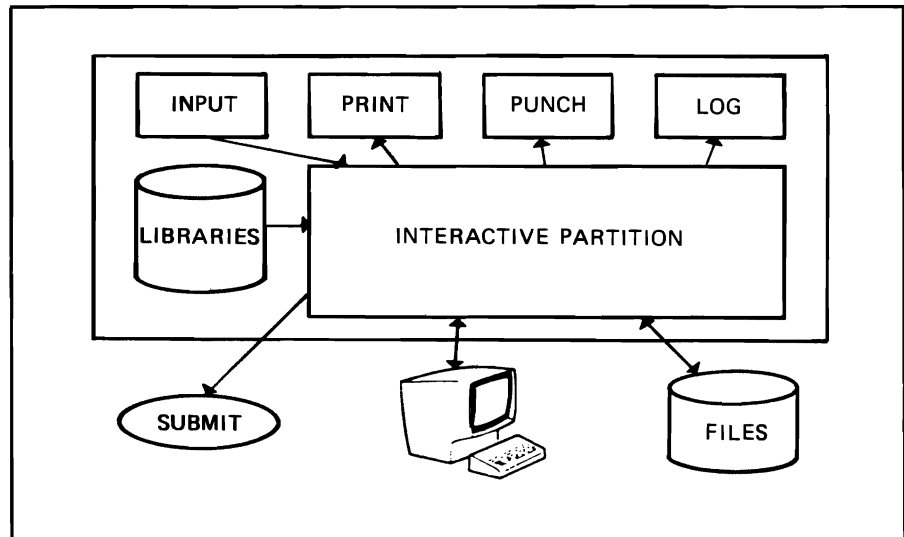


Figure 4.3.1 User's Perception of ICCF

To initiate the ICCF facilities, you key in an identification and a password. This process is referred to as logging on. You then key in instructions or data for your new program from the terminal keyboard. These instructions are placed into a temporary disk storage area, called the INPUT AREA. Whenever you are logged onto ICCF you automatically have access to four temporary disk storage areas: an INPUT, PRINT, PUNCH and LOG area.

After completing your data entry, you instruct ICCF to move your program from the INPUT AREA to your library. Once the program is in your library, you can display and edit it at your terminal. You prepare your program for compilation and testing by adding the necessary job entry statements (job control). You then request an interactive compilation of your program. ICCF schedules your job for execution in an available interactive partition. ICCF moves your job stream to your INPUT AREA where it becomes input

for the compilation. The printed output from your compilation is temporarily stored in your PRINT AREA until it can be displayed at your terminal. The punched output (the object program) is temporarily stored in your PUNCH AREA for later use in your job stream or for saving in your library.

If you want to record details of your terminal session, you may specify logging for your terminal, which causes the input/output from and to your terminal to be stored in your LOG AREA for later display at your terminal.

During execution of your compile and test job stream you have access to all DOS/VSE libraries and DASD files that have been authorized for your use.

ICCF Submit Facility

Besides enabling you to run batch jobs in an interactive environment, as was previously described, ICCF also allows you to submit jobs to DOS/VSE batch partitions for processing. Using the ICCF SUBMIT facility, you can enter jobs into the VSE/POWER input queue. The printed output from such a job may be routed to the main system printer, or it may be displayed at your terminal before being routed to the system printer. You can request the status of these jobs at any time.

User Interface

During your terminal session you communicate with ICCF through commands and control statements entered from your terminal. These commands will be discussed in more detail in a separate topic entitled 'The Command Language'. ICCF allows you to write and execute programs in its interactive environment in the following programming languages used with the IBM 4300 processors: ASSEMBLER, COBOL, BASIC, PL/I, FORTRAN and RPG II.

Exercise 4.3

1. Name three typical users of ICCF.
2. To initiate the ICCF facilities, the terminal user logs on by keying in an _____ and a _____.
3. (True or False) Each ICCF user has four temporary disk storage areas available for use during the execution of a job in an interactive partition.
4. A programmer keys in instructions for a new program from the terminal and ICCF places these instructions into the temporary storage area called the _____.
5. (True or False) There is no facility within ICCF for submitting jobs for execution in DOS/VSE partitions.
6. (True or False) A job executing in an ICCF interactive partition can have access to DOS/VSE libraries and DASD files.

Solution 4.3

1. Application Programmers
Systems Programmers
Operations Personnel
Non Data Processing Professionals
2. identification
password
3. True. Each user has an INPUT, PRINT, PUNCH and LOG area.
4. INPUT AREA
5. False. The ICCF SUBMIT facility allows a user to submit jobs to DOS/VSE batch partitions for processing.
6. True

Topic-4. The ICCF Library File

The ICCF library file is a direct access data set which is composed of several component parts as illustrated in Figure 4.4.1.

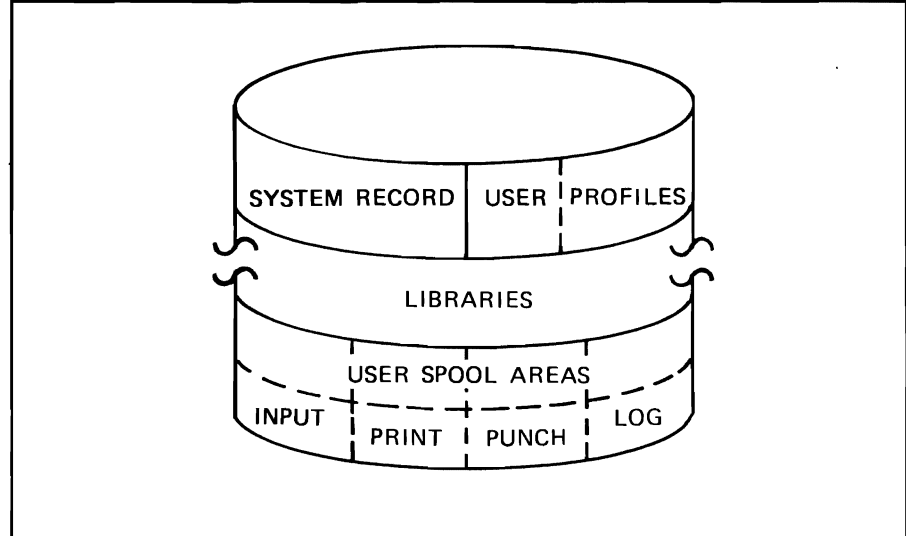


Figure 4.4.1 ICCF Library File

System Record The system record contains control information about the library file, such as the number of user profiles and the number of libraries.

User Profile There is a user profile record for each user of ICCF which contains specific information about the user. The user's identification and password and his main and alternate libraries are examples of the information provided in a user profile record.

Spool Areas Each user of ICCF has four spool areas. These are the temporary storage areas (INPUT, PRINT, PUNCH, LOG) discussed earlier.

Libraries The major portion of the ICCF library file is composed of the libraries themselves. As a user of ICCF you have access to one or more of these libraries, which you may own exclusively or share with other users. Each library has a directory containing an entry for each member in the library.

A single file of data in the library is called a 'member'. Each member is represented by a member name and location in the directory. These members can be either source programs, object programs, data, job streams, or procedures. Each member in a library is stored as private, public or common data and can also be further protected by requiring that the user specify a password. Library members can be updated directly so that changes take place immediately. Or, a copy of the member can be placed in the INPUT AREA and updated there, leaving the original library member unchanged. A new library member can be built from existing members by inserting the required statements from those members into the new member. This facilitates the creation of a new program where the required routines already exist in other programs in the library.

Library Categories These libraries fall into three different categories as illustrated in Figure 4.4.2.

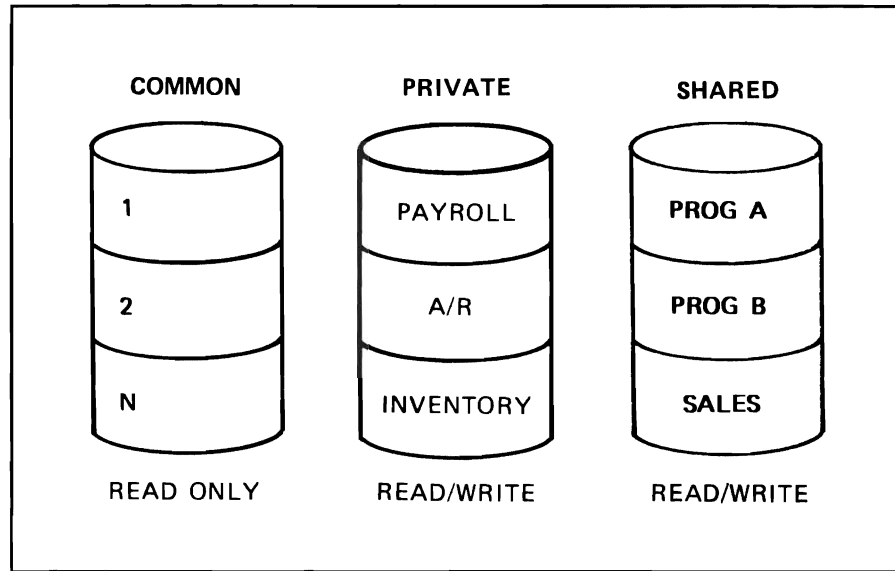


Figure 4.4.2 Categories of ICCF Libraries

Common Library

The first is the COMMON library. This library contains members which are generalized routines or procedures that all users may access in read only mode. There is only one COMMON library per ICCF library file.

Private Library

The second category is the PRIVATE library. This library contains members that are unique to a specific user. The user has the ability to read and update this library.

Shared Library

The third library category is a shared library. A shared library can be a PUBLIC library to which ALL users can have read/write access or a PRIVATE library to which selected users can have read/write access. For example, programs and data unique to the Sales Department are stored in that department's private library. All Sales Department personnel have access to the data in their shared library. However, the data is not available to the Accounts Receivable Department.

Library Usage

Let's look at how you can make use of these different libraries. (Refer to Figure 4.4.3)

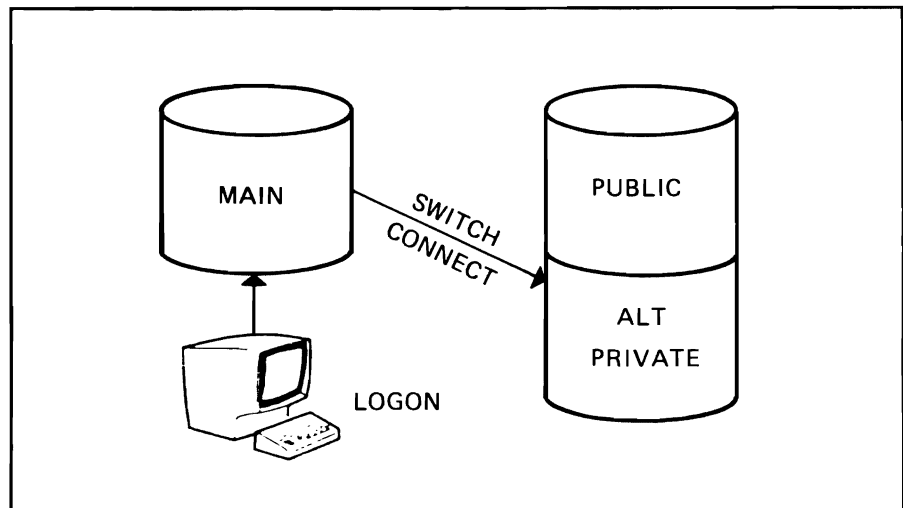


Figure 4.4.3 Library Usage

After logging on to ICCF you have read/write access to your main library (specified in your user profile). This library contains your own programs and/or data. You can access a shared library by switching or connecting to it. A switched-to library (alternate) replaces your main library.

When you connect to a library you have access to your main library and the shared library. At all times you have access to the COMMON library. You may only have read/write access to one library at a time. This is your main library or an alternate to which you have switched.

Exercise 4.4

1. The identification and password for each ICCF user is maintained in the _____ record for that user.
2. What are the four spool areas available to an ICCF user?
3. A library member is stored as _____, _____ or _____ data.
4. What are the three categories of ICCF libraries?
5. (True or False) An ICCF user may have read/write access to more than one library at a time.

Solution 4.4

1. user profile
2. INPUT, PRINT, PUNCH and LOG
3. public, private or common
4. Common, Private and Shared
5. False. An ICCF user may have read/write access to only one library at a time. (main library or alternate switched to).

Topic-5. Modes of Operation

During a terminal session your terminal is always in one of six modes of operation. These operational modes allow you to perform the basic functions of creating and updating library members, compiling programs and executing jobs. (Refer to Figure 4.5.1).

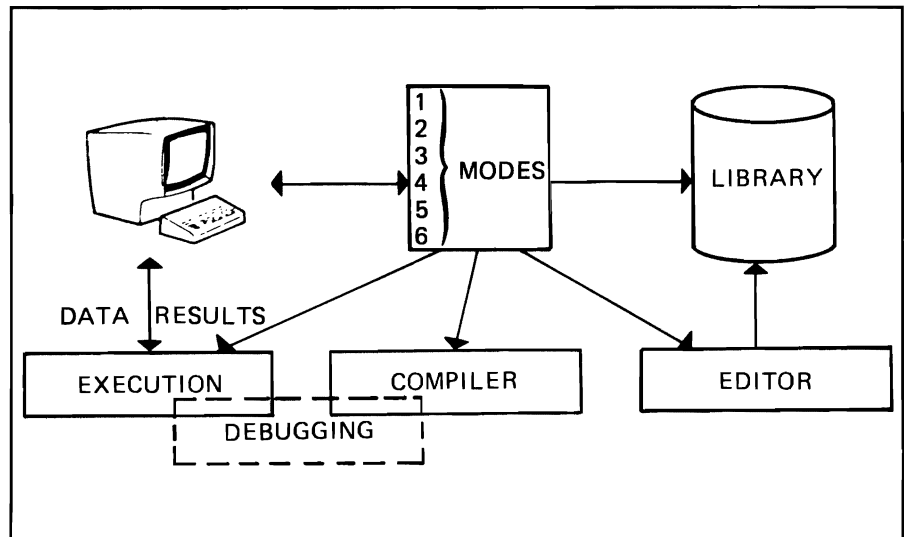


Figure 4.5.1 Basic Functions of ICCF

Associated with each mode of operation are a series of commands and/or control statements which instruct ICCF to perform various functions. These commands will be discussed in the topic entitled 'The Command Language'.

The six operational modes are described as follows:

Command Mode

1. **COMMAND** mode is the controlling mode of operation. While in command mode you may enter ICCF commands to:
 - Select another operational mode
 - Display library member names and owners
 - Assign meaning to program function keys
 - Submit jobs to a batch DOS/VSE partition for execution

Input Mode

2. **INPUT** mode enables you to enter programs and data. The data records are stored in your **INPUT AREA** and later saved in your library. In constructing a program in your **INPUT AREA**, you have the option of including instructions stored in an existing library member rather than rekeying them.

Edit Mode

3. EDIT mode is used to update data in the INPUT AREA and in the library. In this mode the data is located and modified by text or content. New data can be entered as well as existing data. There are two editing facilities available in EDIT mode. These facilities are described in detail in the topic entitled 'The Editors'.

Update Mode

4. In contrast to Edit Mode where you can readily change anything, you need to make your modification by specific line number in UPDATE mode.

Execution Mode

5. EXECUTION mode is entered whenever a job is being executed in an interactive partition or when a job is queued for execution. This mode remains in effect until the job ends.

In execution mode you can execute programs from a DOS/VSE core image library as well as the ones you are creating at your terminal.

Also, in execution mode, you may execute batch programs, which normally read cards and print a report, from your terminal. The input data is keyed in from your terminal and the printed output is displayed at your terminal.

Asynchronous Execution

While your job is executing in an interactive partition, you can temporarily disconnect your terminal from this job and enter another mode of operation to perform other terminal work, such as editing another file. This is referred to as asynchronous execution mode. When your job has finished execution, you may re-synchronize your terminal and review the output.

Dynamic Space Allocation

A feature of ICCF utilized during execution mode is dynamic space allocation. At ICCF installation time a pool of disk space is allocated to be used by ICCF to dynamically allocate user work files to jobs executing in interactive partitions. These dynamically allocated files may be permanent or temporary, that is, they can be retained from day to day or be accessible only during execution of the job or job step using them. This feature eliminates the need for predefining individual user work files.

List Mode

6. LIST mode is always in effect while a display to the terminal is in progress. This display could be output from an interactive partition or the VSE/POWER output queue. It also could simply be a display of a library member.

If printed copy of the displayed information is required, ICCF provides the capability to print the information on a 3270 hard copy device, for example, the 3287.

Exercise 4.5

1. Which one of the following is not a valid ICCF mode of operation?
 - a. Command
 - b. Input
 - c. Debug
 - d. Edit
 - e. Execution
2. (True or False) While in EDIT mode, a user can update data in the library but cannot update data in the INPUT AREA.
3. (True or False) A user is not able to execute programs from a DOS/VSE core image library while in EXECUTION mode.
4. In _____ mode a user may enter ICCF commands to select another operational mode.
5. (True or False) While a job is executing in an interactive partition, a user's terminal can be temporarily disconnected from the executing job and used in another mode of operation.
6. The _____ feature of ICCF eliminates the need for predefining individual user work files.

Solution 4.5

1. c - Debug
2. False. While in EDIT mode a user can update data in the library *or* in the INPUT AREA.
3. False. A user *can* execute programs from a DOS/VSE core image library while in EXECUTION mode.
4. COMMAND
5. True. This is called asynchronous execution mode.
6. dynamic space allocation

Topic-6. The Editors

The ICCF editor is divided into two editing facilities: the context editor, which can be used with any input/output terminal supported by ICCF, and the full screen editor, which can only be used with the 3270 information display system.

Context Editor

The *context editor* is a general purpose text manipulation program that enables you to create and modify *card image* files (80 character records) from the terminal. These files can reside in your library or in your INPUT AREA. The context editor consists of a group of commands which allow you to:

- Locate and modify data within a file
- Reposition the line pointer, which indicates what line you are operating on, to another line
- Replace or insert complete or partial lines of data

All context editor commands have an immediate effect upon the data being edited, that is, the change is made to the actual file or member when it is requested.

The context editor allows global changes to be made to a line, to a number of lines or to an entire member. A global change is shown in Figure 4.6.1. The word SALES is misspelled more than one time in the Sales Analysis program (member name SLSANAL).

<pre style="margin: 0;"> BEFORE: ADD SALS TO DISTRICT ADD SALS TO REGION ACTION: /EDIT SLSANAL CH/SALS/SALES/*G AFTER: ADD SALES TO DISTRICT ADD SALES TO REGION </pre>

Figure 4.6.1 Edit Mode - Global Change

Using the context editor 'change' command (CH) in conjunction with *G (to indicate a global change) *all* occurrences of the incorrect spelling 'SALS' are replaced with the correct spelling 'SALES'.

Full Screen Editor

The *full screen editor* is designed to use the full range of 3270 features, such as user defined program function key options and minimum data transmission (only changed data is transmitted, not the entire screen). The full screen editor includes all the functions of the context editor and has the added advantage of allowing you to change and insert data anywhere on the screen simply by positioning the cursor and making the change. This is contrasted to the context editor where changes are made by commands.

Figure 4.6.2 shows the basic screen layout of the full screen editor.

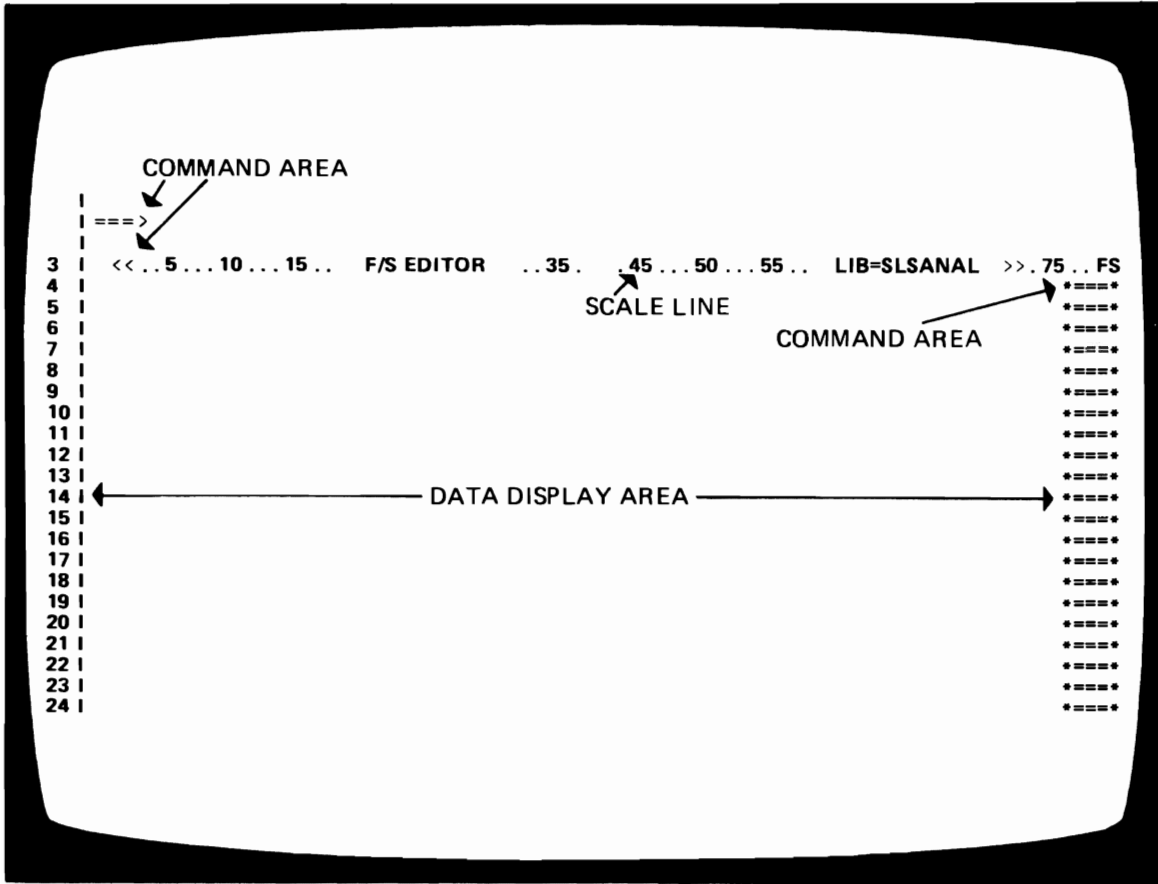


Figure 4.6.2 Full Screen Editor Screen Layout

The edit commands entered into the command area at the top left of the screen manipulate the screen and act upon the entire member being edited. Commands entered in the command area to the right of the screen are special short commands which apply to an individual line or multiple individual lines currently on the screen. The scale line indicates the column number for easy reference and provides the name of the file or member being displayed (SLSANAL) in the data display area and its file type (LIB-library member).

Split Screen Facility

In addition, the full screen editor has a split screen facility. This facility allows you to display and edit multiple members or files or different areas of the same file on the screen at the same time. For example, corrections to a source program may be made on one portion of the screen while the error messages from an interactive compilation of that program are displayed in another portion of the screen.

Exercise 4.6

1. The _____ editor is a general purpose text manipulation program that enables an ICCF user to create and modify *card image* files from the terminal.
2. The _____ editor allows an ICCF user to change and insert data anywhere on a 3270 screen simply by positioning the cursor and making the change.
3. (True or False) ICCF context editor commands do not immediately change the file being edited.
4. (True or False) Global changes can be made to a line, to a number of lines or to an entire member while using either editor.
5. The _____ facility of the full screen editor allows an ICCF user to make corrections to a source program on one portion of the screen while displaying the error messages from an interactive compilation of that program in another portion of the screen.

Solution 4.6

1. context
2. full screen
3. False. ICCF context editor commands have an immediate effect on the file being edited.
4. True. Using the context editor 'change' command (CH) in conjunction with *G, global changes can be made to members from the context and full screen editors.
5. split screen

Topic-7. The Command Language

Throughout the discussion of the six modes of operation there has been reference to ICCF commands and control statements. Terminal operations with ICCF are controlled by commands and control statements entered from the terminal. Certain commands can only be entered in certain modes of operation. Together these commands and statements make up the ICCF command language. Let's now review the six categories of commands.

System Commands

1. *System Commands* direct general system functions, for example, the /LOGON and /LOGOFF commands begin and end a user terminal session. The /SAVE command instructs the system to save the current contents of the INPUT AREA in the library. Using the /RUN command you can cause a job currently in your INPUT AREA to be executed.

Context Editor Commands

2. *Context Editor Commands* relate to the scanning or manipulation of data in your INPUT AREA or library. Like system commands, context editor commands have an immediate effect on the data they refer to. For example, the INPUT command allows you to enter data while still in edit mode; the SEARCH command searches the file for a given set of characters; and the CHANGE command changes one character string (one or more characters) to another.

Full Screen Editor Commands

3. *Full Screen Editor Commands* perform three different functions. They control the content of the screen; they locate areas anywhere within the file and change or add data within the current line; and they allow you to edit individual records currently on the screen with special short commands.

For example, the VIEW command is used to re-arrange, format and view the 80 character records on the screen; and the LOCATE command locates a string of characters within a file.

Job Entry Statements

4. *Job Entry Statements* direct and control the execution of jobs. For example, the /LOAD statement defines the start of a job and indicates which compiler or program phase is to be loaded.

Dump Commands

5. *Dump Commands* enable you to display information from programs that have ended abnormally. For example, the DISPLAY command is used to view the contents of virtual storage associated with the terminated program or general registers.

*Procedural and Macro
Commands*

6. *Procedural and Macro Commands* are library members that contain executable statements, commands and/or data. Their purpose is to perform frequently used functions, such as storing object programs. See the topic on 'Procedures and Macros' for further details on these facilities.

Exercise 4.7

1. Which two of the following commands are not system commands?
 - a. /SAVE
 - b. /LOAD
 - c. /LOGON
 - d. CHANGE
 - e. /RUN
2. _____ commands relate to the scanning or manipulation of data in a user's INPUT AREA or library.
3. _____ direct and control the execution of jobs.
 - a. Procedural commands
 - b. Dump commands
 - c. Full screen editor commands
 - d. Job entry statements
 - e. System commands
4. (True or False) With special short commands of the full screen editor, an ICCF user can edit individual records currently on the screen.

Solution 4.7

1. /LOAD is a job entry statement.
CHANGE is a context editor command.
2. Context editor
3. d - Job entry statements
4. True

Topic-8. Procedures and Macros

Procedures and macros are ICCF library members which contain a sequence of ICCF statements that together perform frequently used functions such as compilation, loading and execution of programs and storing object programs. The statements in a procedure or macro may be ICCF system commands, editor commands or job entry statements and look just like commands and data lines entered from a terminal.

The purpose of such procedures and macros is to save you from having to re-enter the same series of commands and statements each time you want a particular function. Once the procedure or macro has been created and stored, you can later invoke it by issuing only the procedure or macro name. (Refer to Figure 4.8.1).

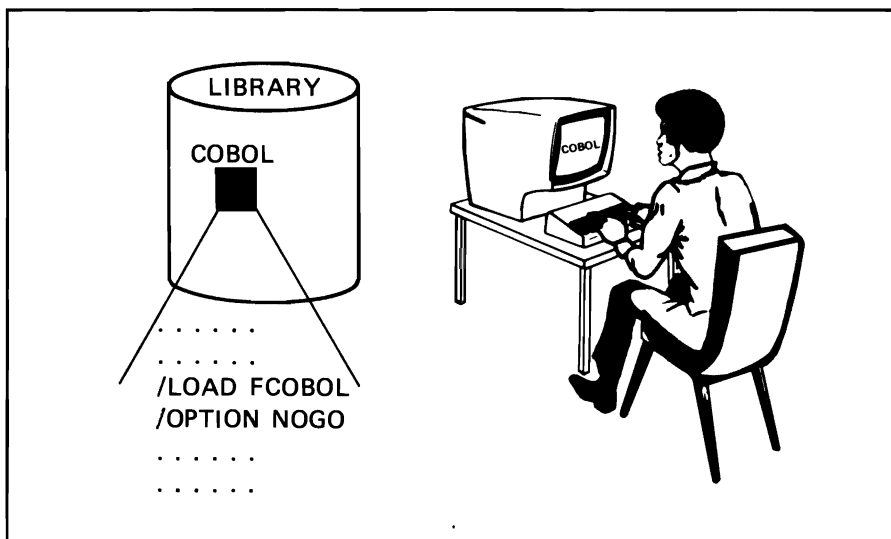


Figure 4.8.1 ICCF Procedures & Macros

IBM supplied macros and procedures are available with ICCF, but facilities are also provided in the system that enable you to write your own procedures and macros. A typical IBM procedure is COBOL which contains all the statements to compile a COBOL language program and produce an object program.

Topic-9. Debugging Assistance

Very few programs work the first time they are tried. Debugging, the process which includes finding errors, correcting them and retrying, is a time consuming task. To speed the testing of programs, ICCF provides debugging assistance in two areas.

ICCF Dump Program

When a program abnormally terminates, the contents of virtual storage associated with the program contains information useful in problem determination. A display of this storage is referred to as a program dump. The ICCF dump program displays various items of information about the termination, such as data fields associated with the terminating instruction, general registers and other storage areas of importance to the terminated program.

Interactive Debug

An optional Field Developed Program named DOS/VS Interactive Debug Facility has been designed to work with ICCF to extend the productivity benefits of online programming. This program will allow you to look at your errors, modify storage, and continue execution until you have a working program.

Exercise 4.8 and 4.9

1. _____ and _____ are ICCF library members which contain a sequence of ICCF statements that together perform frequently used functions.
2. (True or False) A procedure or macro can be invoked by simply issuing its name.
3. (True or False) The ICCF dump program allows a user to modify storage during the process of debugging a program.

Solution 4.8 and 4.9

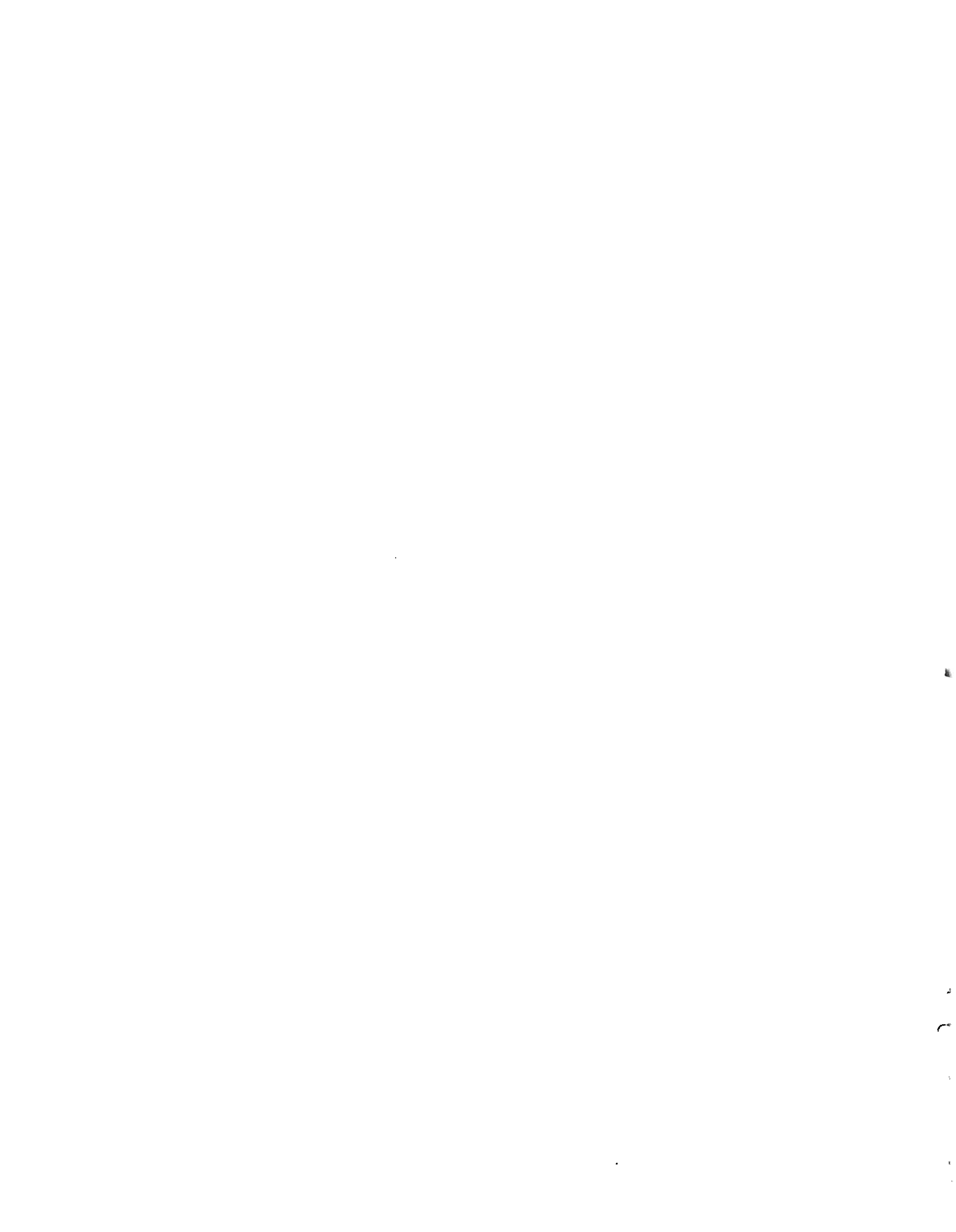
1. Procedures and macros
2. True
3. False. The ICCF dump program only allows a user to *display* storage areas associated with a terminated program. With the optional Field Developed Program named DOS/VS Interactive Debug Facility, a user can *modify* as well as *display* storage.

Topic-10. Data Security

ICCF provides a number of security features to safeguard your installation's data. In order to log on to ICCF, you must supply a correct identification and password. Once logged on, you can create library members. These members can be either public or private. If the member is private, only the user who created the member can update it. A public member can be viewed and updated by any ICCF user who has access to the library it is stored in. A further security for library members is provided by the assignment of member passwords.

In addition to ICCF library member protection, user access to DOS/VSE programs, files and libraries can be restricted.

ICCF security features can be extended to DOS/VSE batch operations through the use of the VSE/Access Control-Logging and Reporting program product. This program working in conjunction with DOS/VSE and ICCF monitors the access to restricted resources used under DOS/VSE and protects the data processing installation from unauthorized usage. A disk log is maintained by this program and ICCF to record accesses to restricted (protected) resources as defined in ICCF tables. These protected resources can be files, libraries or programs. This log can be used as an audit trail since it can contain, in addition to security violations, all successful accesses of protected resources.



Exercise 4.10

1. A _____ library member can be viewed and updated by any ICCF user who has access to the library it is stored in.
2. (True or False) Library members may be assigned passwords to provide additional security.
3. (True or False) ICCF only provides protection for members stored in an ICCF library.
4. ICCF security features can be extended to DOS/VSE batch operations through the use of the _____ program product.
 - a. VSE/Access Control - Logging and Reporting
 - b. VSE/Interactive Problem Control System
 - c. DOS/VS Interactive Debug Facility
 - d. Terminal Transaction Facility
 - e. CICS/VS

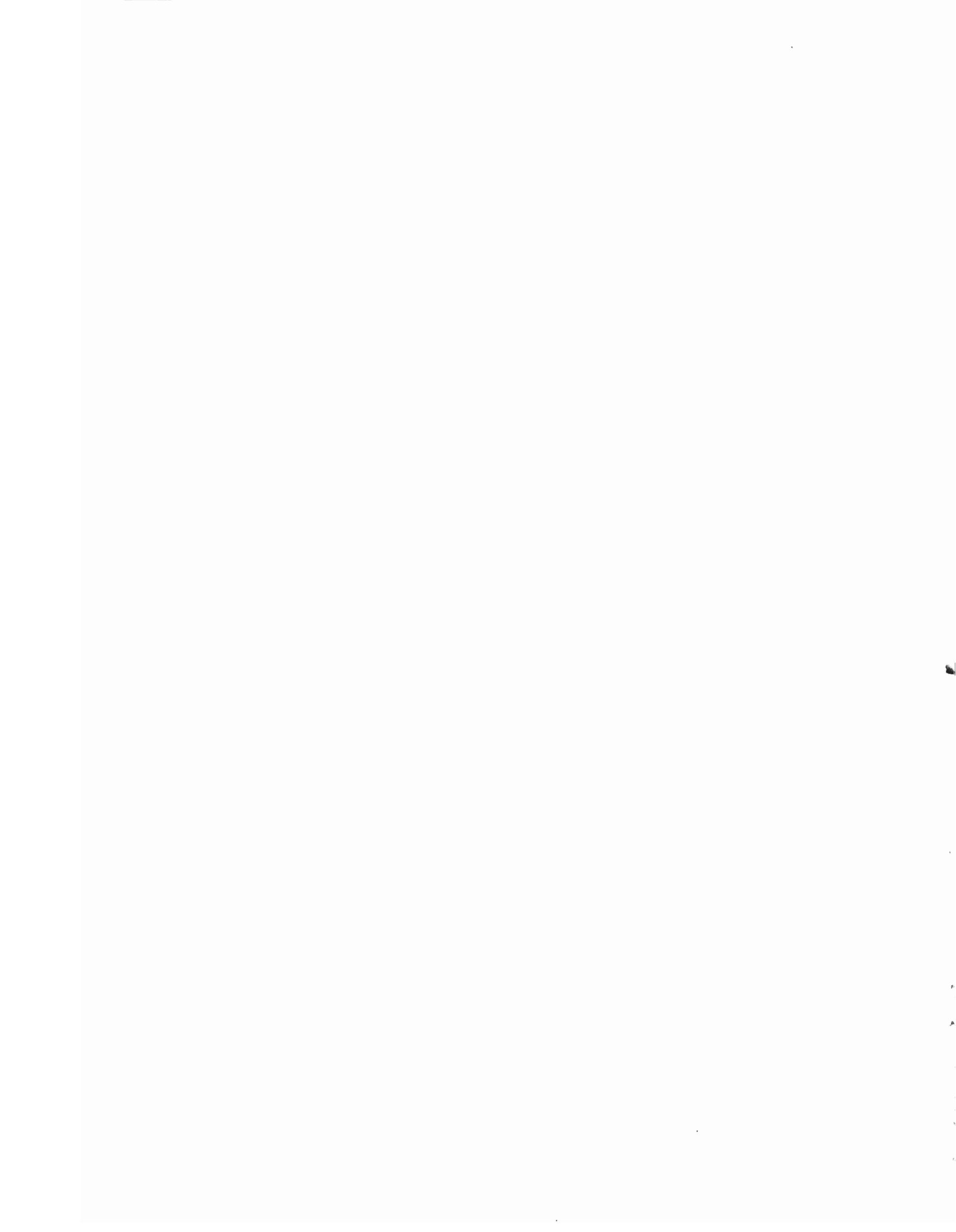
Solution 4.10

1. public
2. True
3. False. ICCF also provides protection against unauthorized user access to DOS/VSE programs, files and libraries.
4. a - VSE/Access Control - Logging and Reporting

Topic-11. Interactive Usability Aids

ICCF provides a series of interactive usability aids which assist you in preparing and submitting jobs for execution in ICCF and DOS/VSE. These usability aids, or prompters, make working with ICCF easier and quicker. They act as an interface between you and the system, freeing you from having to know, or to look up in manuals, details of ICCF and DOS/VSE, such as the format and sequence of control statements needed to execute a particular job.

The ICCF usability aids are application programs executing in interactive partitions that work conversationally with you via messages to the screen. They guide you through the necessary steps to run your job by displaying questions and alternatives at your terminal and from your responses, building the control statements that will perform the required function. The prompter output can be executed in an interactive partition or in a DOS/VSE batch partition or saved in your ICCF library for later use. Thus, these aids not only "prompt", guide and advise you, they also take over much of the task of actually creating and running your jobs.



Exercise 4.11

1. ICCF provides a series of _____ which assist the user in preparing and submitting jobs for execution in ICCF and DOS/VSE.
2. (True or False) The ICCF prompters free the user from having to know the format and sequence of control statements needed to execute a particular job.
3. (True or False) The ICCF prompter output must be immediately executed in an interactive partition.

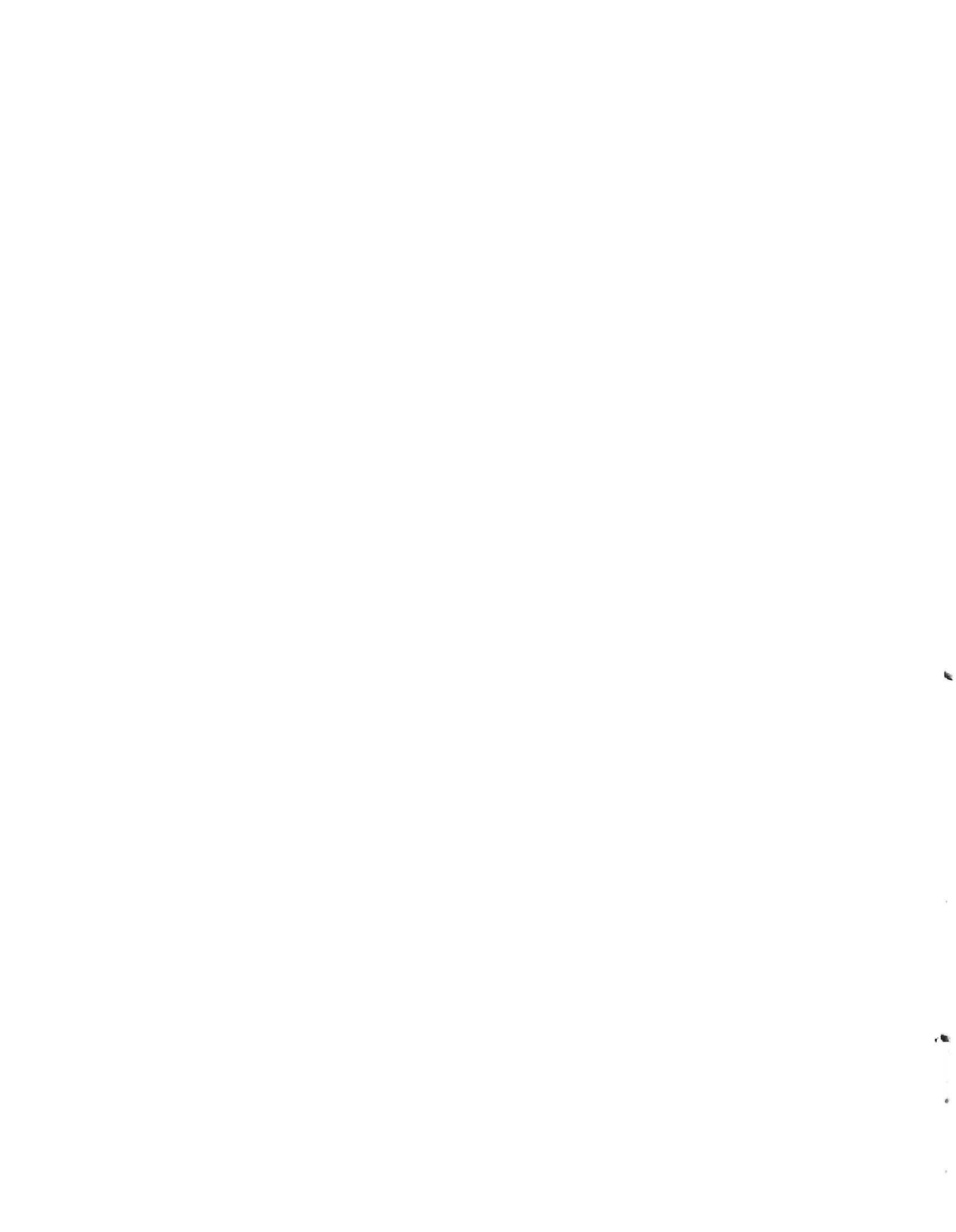
Solution 4.11

1. interactive usability aids or prompts
2. True
3. False. The prompter output can be executed in an interactive partition or in a DOS/VSE partition or saved in an ICCF library for later use.

Topic-12. Summary

In this unit you have been presented with a functional overview of the DOS/VSE interactive computing system, VSE/ICCF. This product offers significant productivity aids to programmers performing application program development and end users performing personal computing and problem solving functions.

As an ICCF user, you can perform editing, compilation, execution and debugging functions using six ICCF modes of operation. You can execute your compilation or job in an ICCF interactive partition or in a DOS/VSE batch partition through the ICCF SUBMIT facility. You have access to the ICCF library file and any DOS/VSE library or DASD files authorized for your use. Finally, ICCF provides a series of interactive usability aids or prompters which assist you in preparing and submitting jobs for execution in ICCF and DOS/VSE.



Unit 5:

Data Systems Environment

Introduction This unit deals with data, its use, management and control. Data is the most basic element of a data processing system and is the essential ingredient in the operation of a business. In order for that data to be processed and to produce the desired results, it must be accessed easily and efficiently. There also need to be effective methods for controlling the use (data security) and preserving the content in its intended form (data integrity).

Data security and integrity could be implemented through manual techniques, but an automated process is more accurate and productive. The automated methods of handling and managing data as provided by IBM are presented in the six topics of this unit. The first topic discusses the attributes and value of data and the two most typical approaches to its management. The next four topics describe the Data Systems Environment and the IBM software products of which it is comprised. Topic 6 is a summary of the entire unit.

- Objectives** At the completion of this unit, the student should be able to:
- Understand the value and uses of data
 - Contrast the traditional approach to organizing and managing data with the Data Systems approach
 - Contrast data communications with the batch mode of processing data
 - Name and differentiate the 4 basic elements of a Data Systems Environment
 - Identify the basic functional characteristics of IBM's Data Base/Data Communications products
 - Understand the basic terminology (vocabulary) used in a Data Systems Environment

Average Study Time 2 to 3 Hours

Topic 1: Background

In this first topic we will examine the concepts of data and the environment in which data is processed. This will lay the foundation for topics which follow.

Value of Data

- CASH
- ACCOUNTS RECEIVABLE
- MATERIALS
- EQUIPMENT
- PLANT

Figure 5.1.1 Assets of a Business

In Figure 5.1.1 are the assets familiar to all business - the items that normally appear on a corporate balance sheet. Any prudent business would have controls to protect these assets - how they are dispensed, who uses them, how much, and so forth. However, data does not normally appear on the balance sheet and has not traditionally received the recognition as a corporate asset that it deserves.

Yet, data is used every day at all levels of management in every functional area of a business.

For operational management the mission is to run the day by day functions and this requires data like balances, inventories, credit ratings, attendance, and work-in-process. For optimum operations the data must be accurate and up-to-date.

Middle management, concerned with measurement and control also needs timely data. They need to know things like reject rates, material requirements, and class schedules.

The challenge of top management to make accurate future plans requires current, valid data. Information like the number of cumulative claims exceeding \$1000 or a list of the most profitable customers might not be readily available or easily accessible, but is a necessary base for management decisions.

- DATA — — — A BASIS FOR MANAGEMENT DECISIONS
- DATA — — — A KEY CORPORATE ASSET
- DATA PROCESSING — — — A SIGNIFICANT CORPORATE ASSET
 - PERSONNEL
 - HARDWARE
 - APPLICATION PROGRAMS

Figure 5.1.2 Value of Data

It can be concluded that data, and thereby data processing, represents a significant corporate investment in a number of ways.

- There are the people in the data processing area that understand both data processing and the business.
- There is the data processing hardware that is installed, the CPU, memory, storage devices, and terminals required to run application programs.
- There are the application programs that are necessary to run a business and provide the information needed by management to make decisions concerning all the corporate assets listed earlier.
- There is the data itself. The application programs are useless if their data becomes lost or damaged. As bad as a data source that is lost, is one that is only slightly damaged, but the damage is not known. Company decisions will continue to be based on its contents until a problem occurs like an empty stock location, a full-loading dock, or something worse.

Traditional Approach

Given that data is to be considered a valuable asset, let's examine the ways in which it is managed. There are two current methods, the traditional approach and the data base approach.

In the traditional approach, data sets are designed to serve individual applications, such as inventory control, payroll, accounts receivable, or purchasing. Each data set is specifically designed for its own application and stored separately in the computer, on tape or disk. Quite often, the data sets of different applications contain common data fields. This repeated data causes an extra problem for the user because it becomes very difficult to keep it consistent. For example, in addition to the payroll data set with its normal applications there might be a need for the retention of data about the employees and their jobs for use in the hiring, promoting, and skill inventory functions. As shown in Figure 5.1.3, a second data set is created to contain this data, but since both it and the payroll data set contain data about employees they require the same basic descriptive fields. That is *redundant data* with the problems of wasted storage space and the need to perform update operations against both data sets when the data changes.

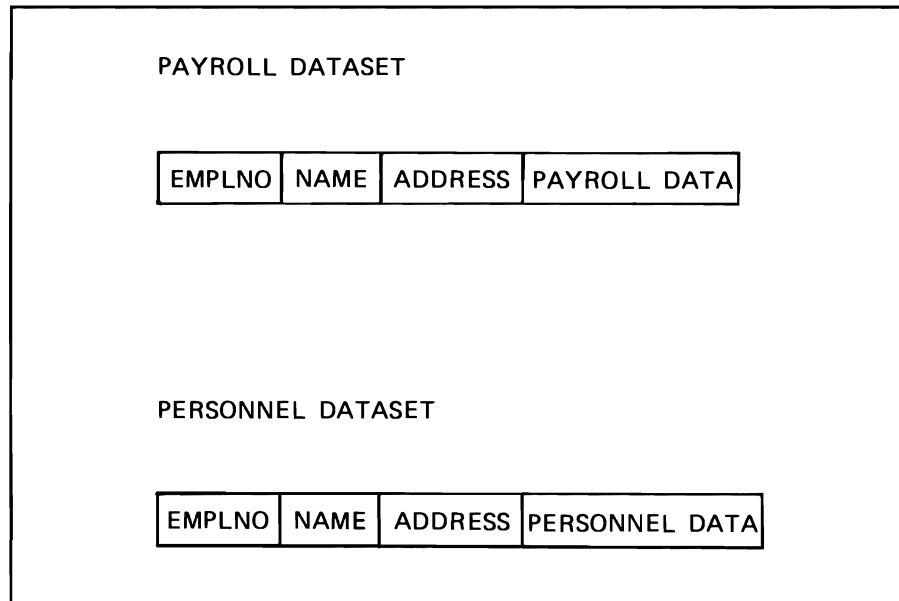


Figure 5.1.3 Redundant Data

Furthermore, the same data in different data sets can often have different formats. This variance in the format of common data means that application programs become tailored to specific methods of physically storing the data and even to specific physical devices. When new applications, data management techniques, or devices are introduced, the application programs normally have to be changed. As a result many application programs may be in a perpetual state of change, adding appreciably to the overall cost of data processing.

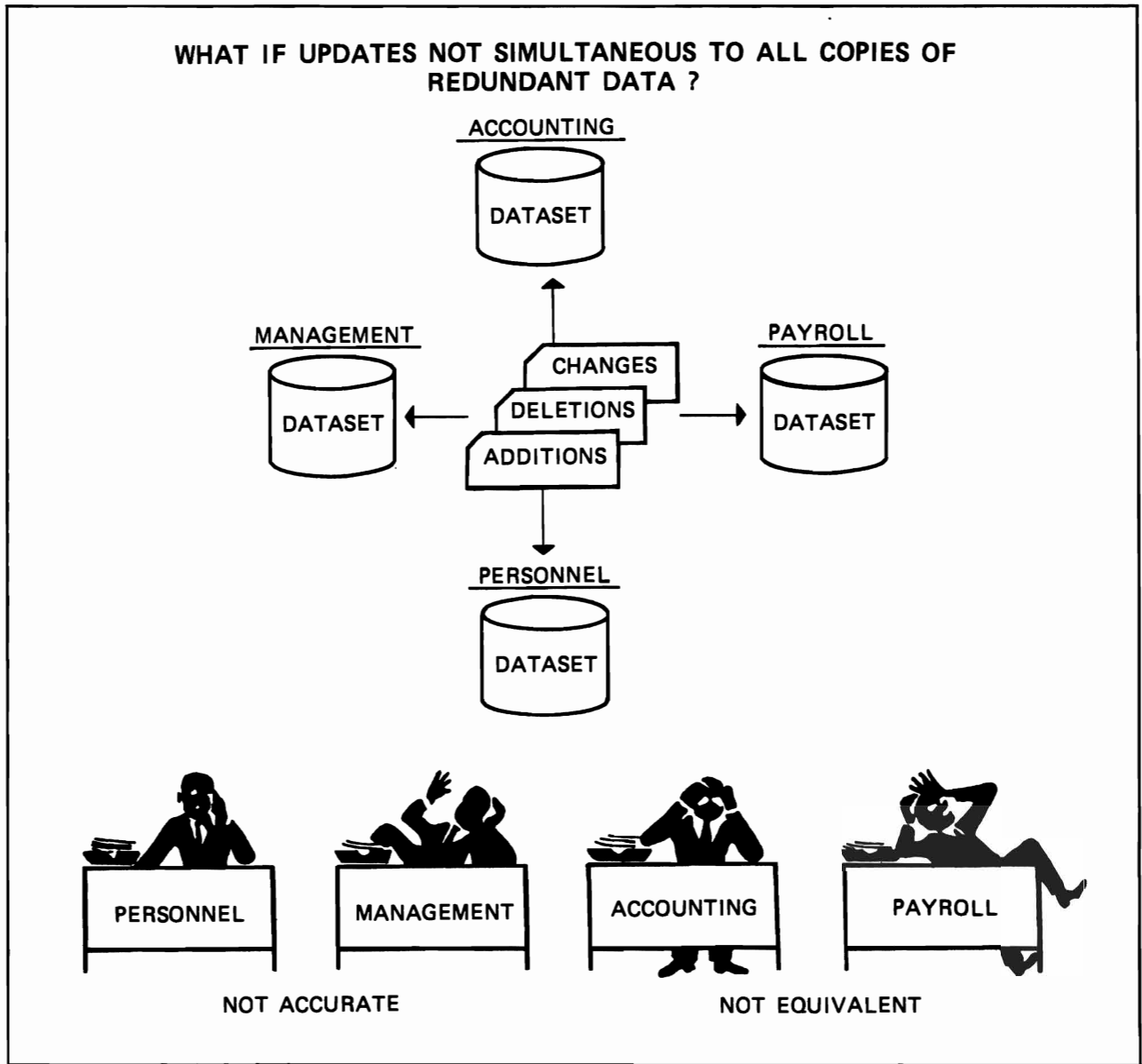


Figure 5.1.4 Data Set Maintenance Concerns

What can be done to reduce these problems? The data sets could be combined as shown in Figure 5.1.5.

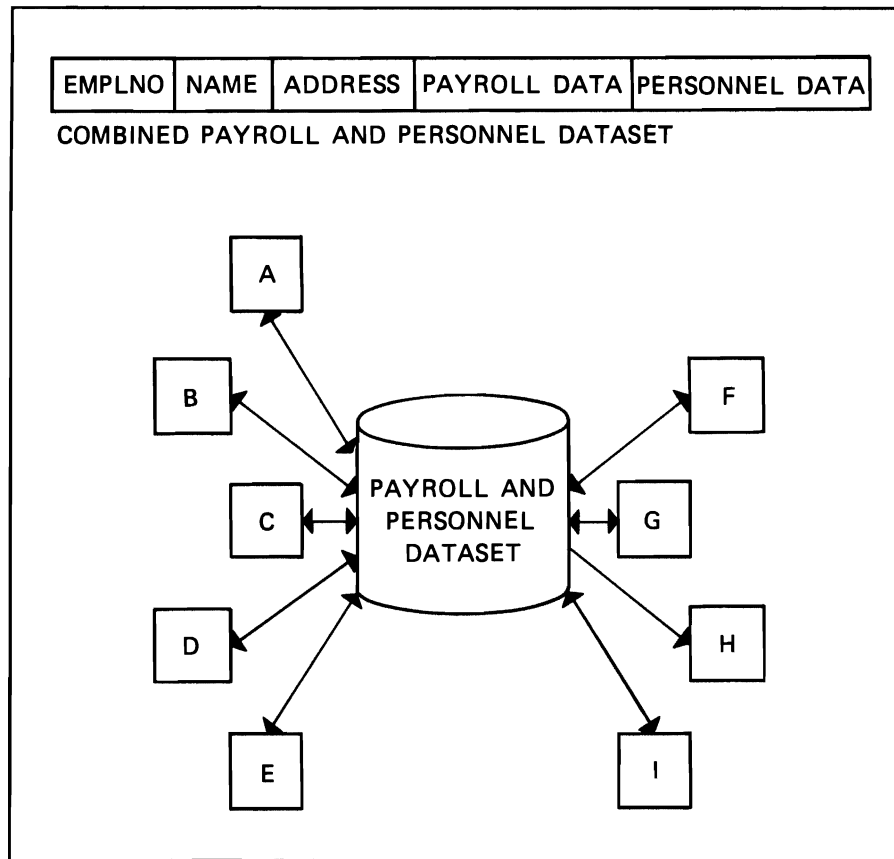


Figure 5.1.5 Combined Data Sets

As you see at the top of the figure, the combined data set consists of employee number, name and address followed by the fields for payroll and personnel applications. The result will be that the problem of data redundancy is solved. On the other hand, new problems have been introduced.

First, confidential information among the personnel fields is suddenly available to the "payroll people" using the combined data set. Also, confidential payroll information is available to the "personnel people". In other words, *data security* has been reduced.

Second, what happens if the combined data set is destroyed or erroneously updated by the personnel application? This will cause problems not only for the personnel area but also for payroll, if for instance, it causes a delay of the pay checks. Combining the data sets causes the reduction of *data integrity* for the previously redundant data.

In addition, combined data sets will affect maintenance. If new data needs to be added to the personnel application then the contents of the combined data sets will change. This will require changes to all of the programs in the personnel system, but will also require the payroll programs to be modified to reflect the presence of the new data. This means increased costs for *data set maintenance*.

Let's summarize the characteristics of combined files.

The advantage is that data redundancy can be reduced. This means that multiple copies of the same data can be avoided. The result is more accurate and reliable data.

As discussed previously, combined files can reduce data security and integrity when data is accessible by many programs and increase program maintenance when the format or contents of combined data sets need to be changed.

The question now arises: Is there a way to get the benefits of reduced data redundancy without the negative effects from the combined files? The answer is that there indeed is a way.

Data Base Approach

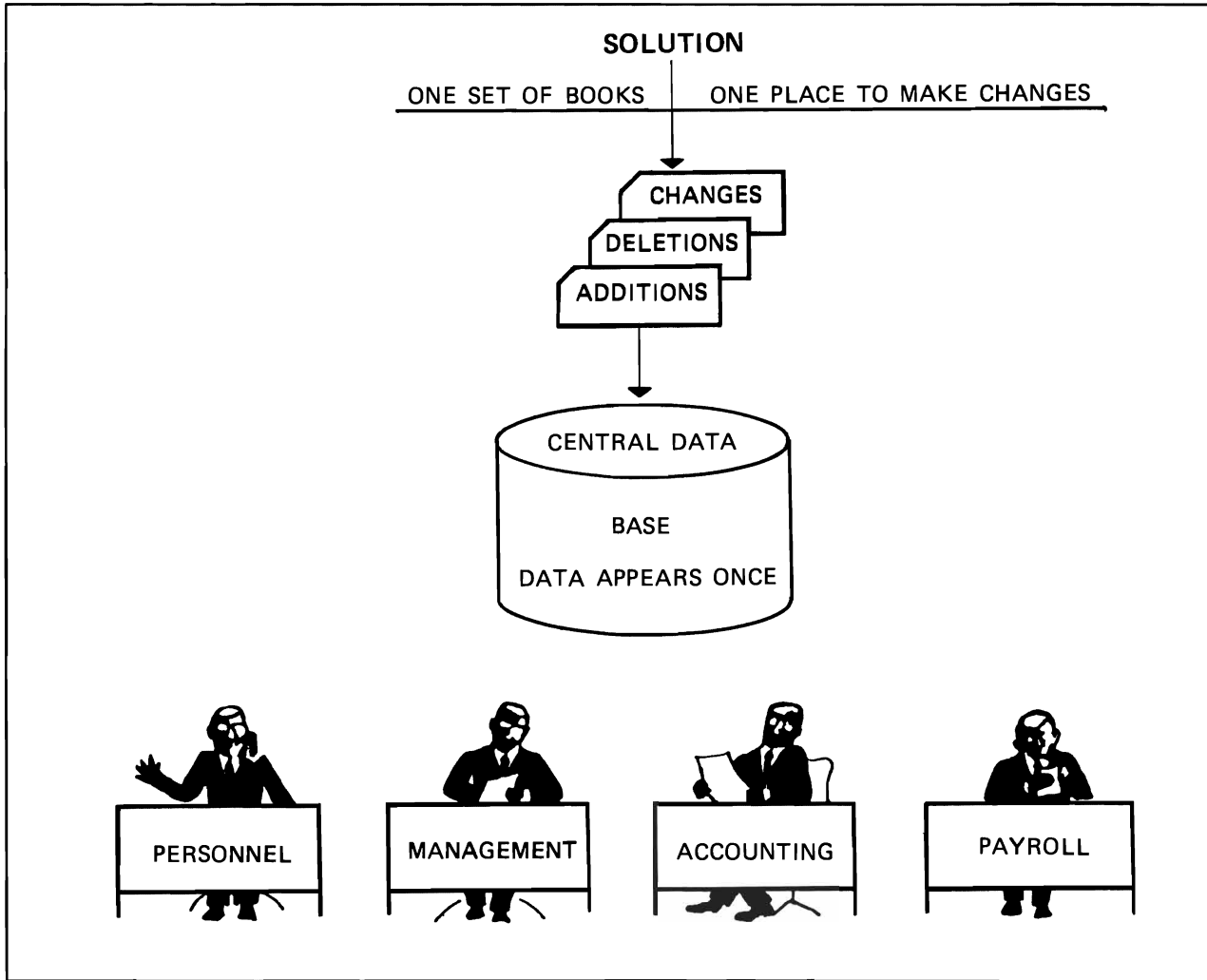


Figure 5.1.6 Solution to Redundancy and Maintenance Concerns

A solution is an organization known as a data base, a collection of interrelated data elements processable by one or more applications. The objective is to eliminate the undesirable attributes of combined data sets and provide for the integration, sharing and control of the common data.

This is accomplished through the implementation of a software program called a data base manager, which provides methods of organizing, accessing, and manipulating data. It provides a data structure and the facilities necessary to maintain and use that structure. We will examine the functions of a data base manager later in this unit.

Online Processing

As was presented in the first part of this unit, data is a valuable asset; but data residing in a computer system, however efficiently organized and managed, can not be valuable until it is accessed by the user. The traditional method of satisfying the user's requests for information is to design, code, test, and execute a program. This method is not always responsive to the needs of the user. A manager waiting for the information on which to base a decision would prefer the answer in seconds rather than overnight.

As another example, suppose we had the sequence of events in Figure 5.1.7.

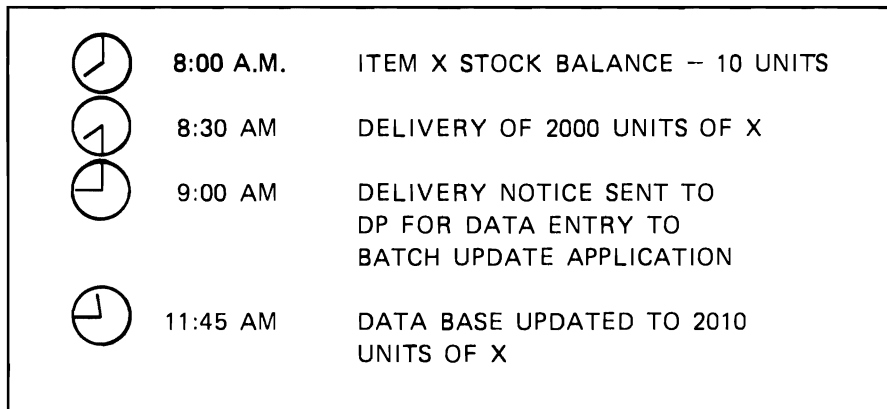


Figure 5.1.7 Inventory Update Process

If a request for the availability of 100 units of item X arrived at 10:00 am, the response would be negative and the result would be a lost sale. This illustrates very clearly that data decreases in value with the time required to provide it to the user.

The data base approach has provided *data availability* but has not resolved the need for *data currency*. The provision of data currency is known as *online* (or *real-time*) processing. It is accomplished by the implementation of a software program known as the *data communications manager*.

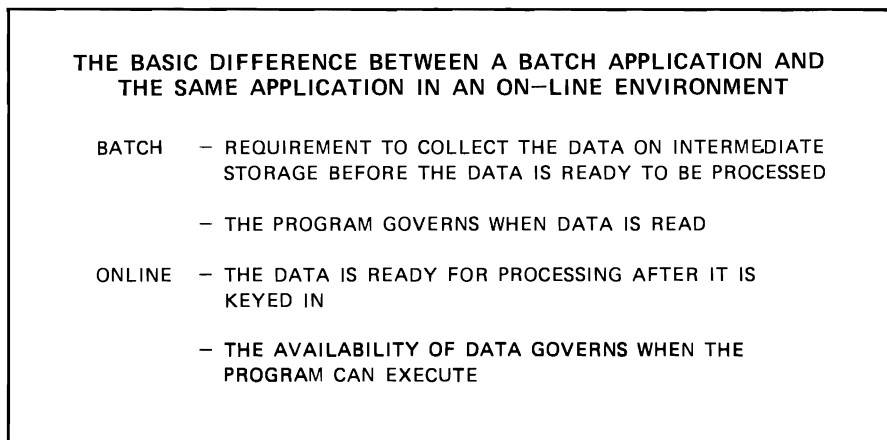


Figure 5.1.8 Batch vs Online

In addition to providing data currency, online processing allows immediate processing of the data. This means that the time from the collection of data

to the delivery of output from the processing of that data, known as turn-around time, can be measured in minutes instead of hours. It also can reduce the volume of printed reports due to the ability to access portions of data bases or records as needed. The specific functions of a data communications manager will be presented later in this unit.

The net result, then, of processing data online is to provide better service to users, whether they be within an organization or those it serves.

As shown in Figure 5.1.9 we have now defined what is called the *data base/data communications (DB/DC) approach* to meeting the informational needs of a user.

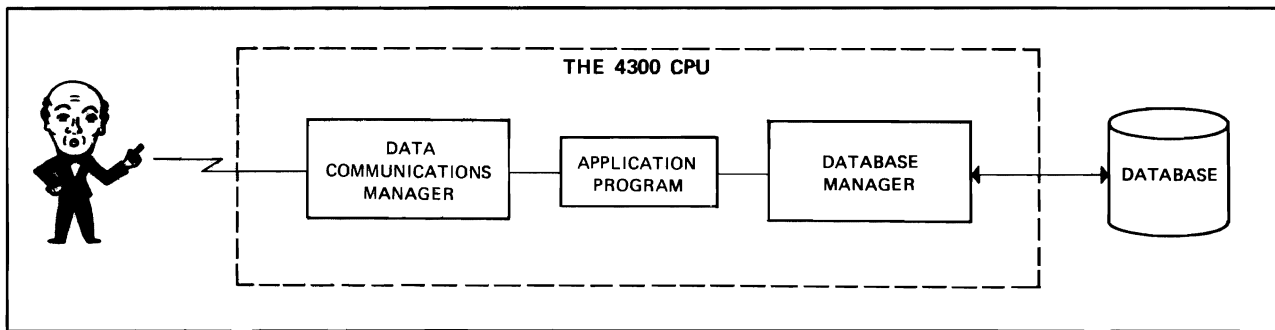


Figure 5.1.9 DB/DC Approach

Data Systems Environment

In a DB/DC environment, the data processing system must be oriented to the collection, management, and delivery of data. It must provide facilities for accomplishing these functions as needs arise and events occur rather than through regularly scheduled processing.

To adequately meet these challenges of today's environment, the data processing system must provide facilities in each of the following areas:

- Data administration
The controls needed by the DP organization to ensure adequate protection and proper usage of all data.
- Data base
An approach to the storage and retrieval of data in the computer system.
- Data communications
The means of making the data available to users on a timely and convenient basis.
- Data delivery
The additional effort needed to convert raw data into useful information for specific users.

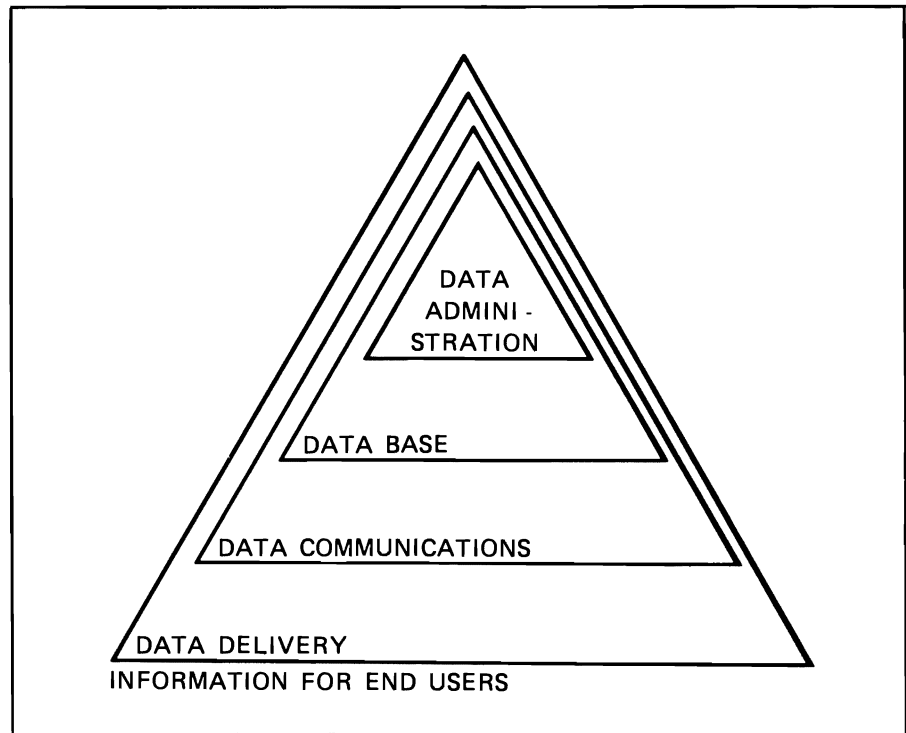
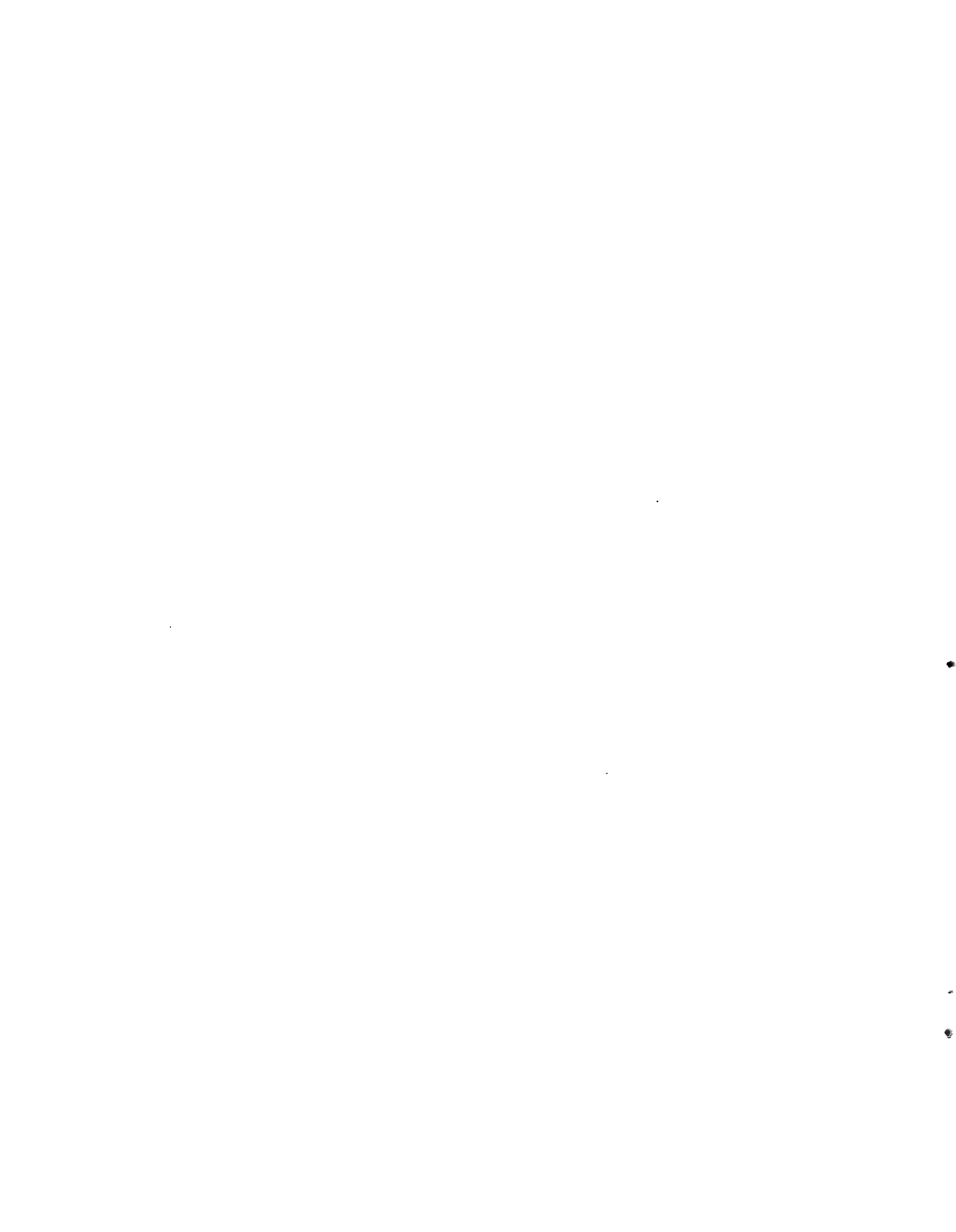


Figure 5.1.10 The Complete Solution

As illustrated in Figure 5.1.10, these four areas are the elements of a *Data Systems Environment*. The requirements could be fulfilled by each DP organization writing its own programs; however, there are IBM software products available in each of the areas which would allow the programmers to concentrate on items which are unique to their organization.

The remainder of this unit will examine in more detail the functions of each of these areas and the facilities of the IBM products available to accomplish those functions.



Exercise 5.1

1. The two approaches to managing data are _____ and _____.
2. The primary advantage of combined data sets is a reduction of _____.
3. Combining data sets creates possible exposures in the areas of _____, _____, and _____.
4. The solution to the concerns associated with combined data sets is a collection of interrelated data elements processable by one or more applications. This is known as a _____.
5. In addition to data availability, another key requirement for meeting the informational needs of business is data _____. This requirement is provided by _____ processing.
6. The four elements of a Data Systems Environment are _____, _____, _____, and _____.

Solution 5.1

1. a) traditional
b) Data Base or DB/DC
2. Redundant data or data which is common to all files involved
3. a) data security
b) data integrity
c) maintenance
4. data base
5. a) currency
b) online
6. a) data administration
b) data base
c) data communications
d) data delivery

Topic 2: Data Base

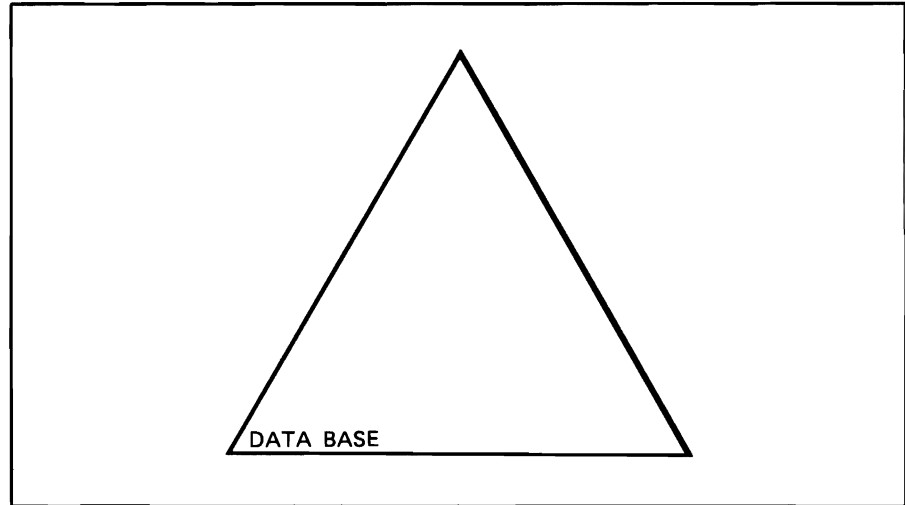


Figure 5.2.1 The Data Systems Environment

Let's look at the first of our four basic elements in the total Data Systems Environment - the data base - and see how it can help.

Data Base Manager

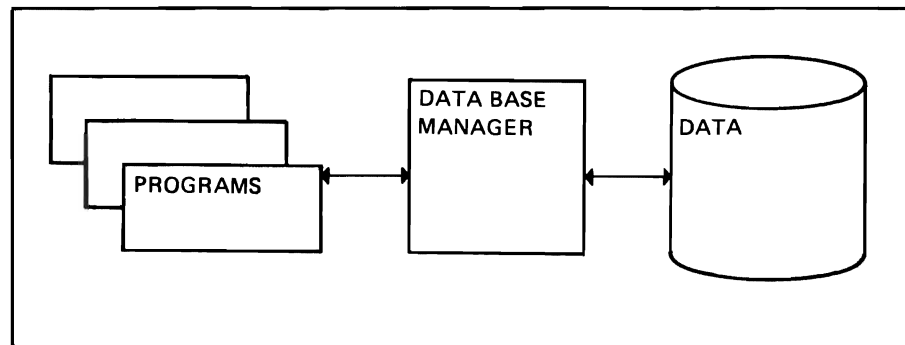


Figure 5.2.2 Data Base Manager

The data base manager is a piece of software that fits between programs and data. Because the data base manager takes over management of data storage and manipulation, it does away with the need for programs that contain both logic and a description of the way the data was structured. This is termed *data independence* and allows the changing of the data structure or the addition of new data types without affecting existing programs. It also provides *device independence* or the separation of program logic from the details and characteristics of the physical device on which the data is stored.

Another attribute of a data base manager is that data can be restricted on a "need to know" basis. An employee data base may range in content from military history to salary history, yet all that an employee mailing program is allowed to obtain is name and address information. It cannot, for instance, access the salary information. With this approach programming becomes

easier, because the programmer has no need to get involved in the intricacies of the entire file, and data confidentiality is improved.

The data base manager also includes the capability to recover data in the event of application failures, software failures, or file failures of any kind. This increases the availability of the data without the need for programming complex recovery and maintenance routines.

Data Language/I

In the 4300 environment the data base manager provided by IBM is Data Language/I DOS/VS (DL/I DOS/VS). This program product provides facilities such as the control of data redundancy, the reduction of duplicate data maintenance, data and device independence, and data recovery. With DL/I less time is spent in maintaining application programs and in the programming of new applications. The use of DL/I also eases the conversion to new storage devices (for example, from 3340 to 3370).

DL/I allows a choice of the type of data structures to be used. These structures, called organization methods, are designed to meet the varying needs of different applications without the need for reprogramming.

Application programs written in Assembler, COBOL, PL/I, or RPG/II run with the DL/I program in a batch partition of DOS/VSE. DL/I also includes an interface which allows data base programs to run in an online environment. The same data base can be accessed concurrently by application programs in different partitions. For example, there can be online inquiries to a data base while a batch program updates it.

Payroll, personnel, accounts receivable, hospital records, petroleum well records, and manufacturing are only a few of the applications which have been implemented using the facilities of DL/I DOS/VS to meet their data structuring and handling needs.

In summary, DL/I DOS/VS is a data base management control system that improves an installation's ability to implement and maintain batch and online applications. It permits the writing of data independent application programs and provides program and data base integrity. It is a program running in a DOS/VSE partition that supports application programs written in various languages.

DL/I DOS/VS allows concurrent access by more than one user to the same data base. Application programs may utilize this concept in conjunction with a data communications manager to access DL/I DOS/VS data bases in an online environment.

It is the tool to be used in the control and management of that valuable corporate asset known as data.

Exercise 5.2

1. The software that provides a storage and manipulation interface between data and programs is called a _____.
2. DL/I DOS/VS provides _____ and _____ independence which allows the programmer to concentrate on the logical functions of the program.
3. (True or False) DL/I with its facilities for data integrity, data security, data recovery, data maintenance, and data access improves the ability of an installation to control the data used in any application system.

Solution 5.2

1. data base manager
2. a) data
b) device
3. True

Topic 3: Data Communications

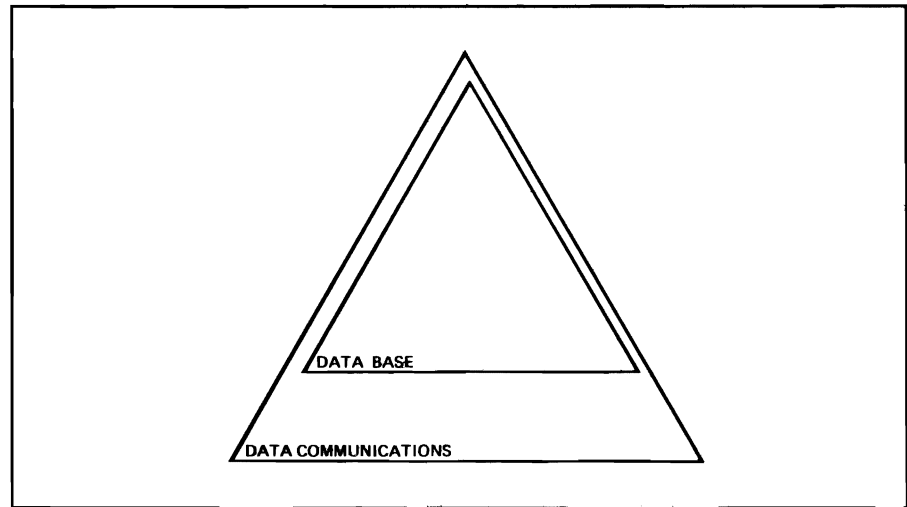


Figure 5.3.1 The Data Systems Environment

The second major element - data communications - can help solve the data currency problem. It can be even more effective when used with a data base manager to ensure that the data is consistent and protected from misuse.

An online system requires terminals, a central processor, and direct access storage. When the terminals are located remotely, additional communications facilities may also be needed.

Data Communications Manager

The programming required in an online environment includes user created application programs that process the data required by the user as well as an operating system that provides basic system-wide programming services. There must also be additional programming that extends the operating system capabilities to create an online environment for the application programs. This additional programming is known as a data communications manager.

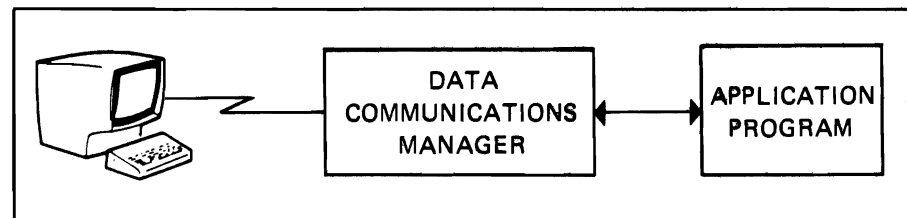


Figure 5.3.2 Data Communications Manager

Like the data base manager, the data communications manager is a software package. It provides many of the functions common to handling a terminal network that would otherwise have to be written into each online program.

The data communications manager supports online processes of the following general types:

- Inquiry

Provides the user with ready access to information in centralized files (the data base). In response to an operator's inquiry request, the program sends back a reply message containing data retrieved from the central files.

- Inquiry-With-Update

Extends the inquiry type of process to the point where the user can alter, delete, and store data in centralized files. This application is more complex than simple inquiry because the integrity of the central files must be protected; safeguards are needed against system failure during the update process and against multiple users concurrently changing the same record.

- Data Entry

Allows lines of data to be keyed at speed, with minimum interruption from the computer, and is similar to a card punching operation. It is used to create files at the central location for subsequent processing or access by other users.

- Remote Batch Processing

Allows data to be sent to the computer for batch processing - for example to perform a periodic update of the central files. This type of application is often associated with remote terminals that can store data - on disk, diskette or on cards. Data originally entered at the terminal can thus be stored and transmitted at high speed during off-peak hours. This technique is referred to as "batch store and forward".

- Message Switching

Handles the transfer of messages from one terminal to other terminals in the network. No data processing or access to central files is required. If necessary, the application stores a message until each destination terminal is able to receive it.

Examples of applications that will use one or more of these processes include customer service requests, student records and class information, hospital patient records, credit checking and authorization, and control of stockholder records.

An online system can be set up quickly and efficiently, because the data communications manager provides a standard set of services for the control functions. Those who set up the system can concentrate on the application functions: the ones directly concerned with their data and their business needs.

CICS/DOS/VS

As in the data base environment, IBM has a program product available for the online environment to build on the functions provided by DOS/VSE.

The Customer Information Control System/DOS/VS (CICS/DOS/VS) is an IBM program product that controls online Data Base/Data Communication (DB/DC) applications. CICS/VS provides most of the standard functions required by application programs for communication with remote and local terminals and controls the resource usage and data access of multiple users. CICS/VS and the application programs under its control run under DOS/VSE.

CICS/VS is a general purpose DB/DC control system that can be tailored to the needs of most online applications. Furthermore, the data files that are used by the online application may also, in general, be used by batch programs in other partitions.

CICS/VS controls and services the system in the following manner. Whenever a request for some use of the system is sent from a terminal, CICS/VS identifies the application program required for the particular job. It loads the application program (if it is not already loaded), and starts a task to execute the application program. The application program can then send messages to the terminal and receive replies from the terminal until it has finished processing. When the application is complete, the program ends the task and returns the terminal to a standby state. During execution of the application program, CICS/VS facilities will have been requested by the program to handle such jobs as transmission of data between the program and the terminal, and accessing the data base. Thus CICS/VS is both controlling the overall flow of the online processing and supplying an interface between the online programs and the operating system of the computer (DOS/VSE).

Some of the attributes and characteristics of CICS/VS include:

- System modularity which allows an installation to choose or add only those functions needed to support a particular set of applications.
- System recovery functions to facilitate the recovery of the system from a variety of possible errors.
- Data security facilities to help prevent unauthorized access of information.
- Device independence and data independence, for most applications, allowing the application programmer to be concerned only with the coding to perform the required application logic function.
- A command-level (high level) interface to COBOL, PL/I, RPG/II, or Assembler Language to allow programmers to issue CICS/VS commands which are similar in appearance and usage to the language being used.

It should be apparent that one of the major advantages of CICS/VS is that of simplifying the programming of online applications. Without CICS/VS, all terminal handling, security, program loading, formatting, and data retrieval would have to be coded by the programmer.

While this simplified programming can lower the skill level needed in a programming staff and reduce the costs of implementing online applications, there is still a need for system resources to be used effectively.

As application demands expand, the terminals and application programs required to service them must expand. So the initially simplified use of CICS/VS can become complex and a natural concern is the continued responsiveness of the system and the impact on other applications. In recognition of this, CICS/VS is built for responsiveness and efficiency. A number of users are serviced at the same time in many CICS/VS installations, because it has sophisticated capabilities that allow many different application programs to be in operation at the same time in the same DOS/VSE partition.

- SIMPLIFIES APPLICATION PROGRAMMING
- MAKES EFFICIENT USE OF SYSTEM RESOURCES
- ENHANCES RESPONSIVENESS
- PROVIDES FOR PRIORITY PROCESSING

Figure 5.3.3 CICS/VS Features

CICS/VS can make the implementation of an online system faster, easier, and more effective by simplifying application programming as well as efficiently using the resources of the system. At the same time, it provides responsiveness to user needs and priority processing based on the needs of the installation.

Exercise 5.3

1. The software that provides an online environment by interfacing between the operating system and the terminal user is known as a _____.
2. The IBM data communications software product that controls online environments is _____.
3. (True or False) A data base may be concurrently accessed from an online program and a batch program. .
4. As with DL/I DOS/VS, the IBM data communications product provides _____ and _____ independence.

Solution 5.3

1. data communications manager
2. CICS/DOS/VS
3. True
4. a) data
b) device

Topic 4: Data Administration

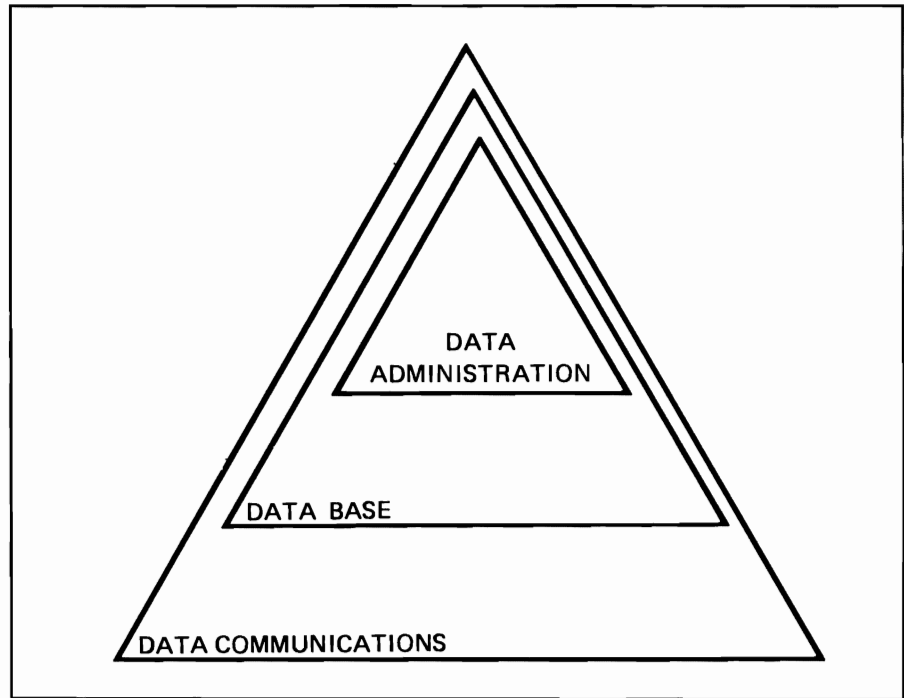


Figure 5.4.1 The Data Systems Environment

The implementation of the third element of our data systems environment, data administration, requires both software and organizational functions.

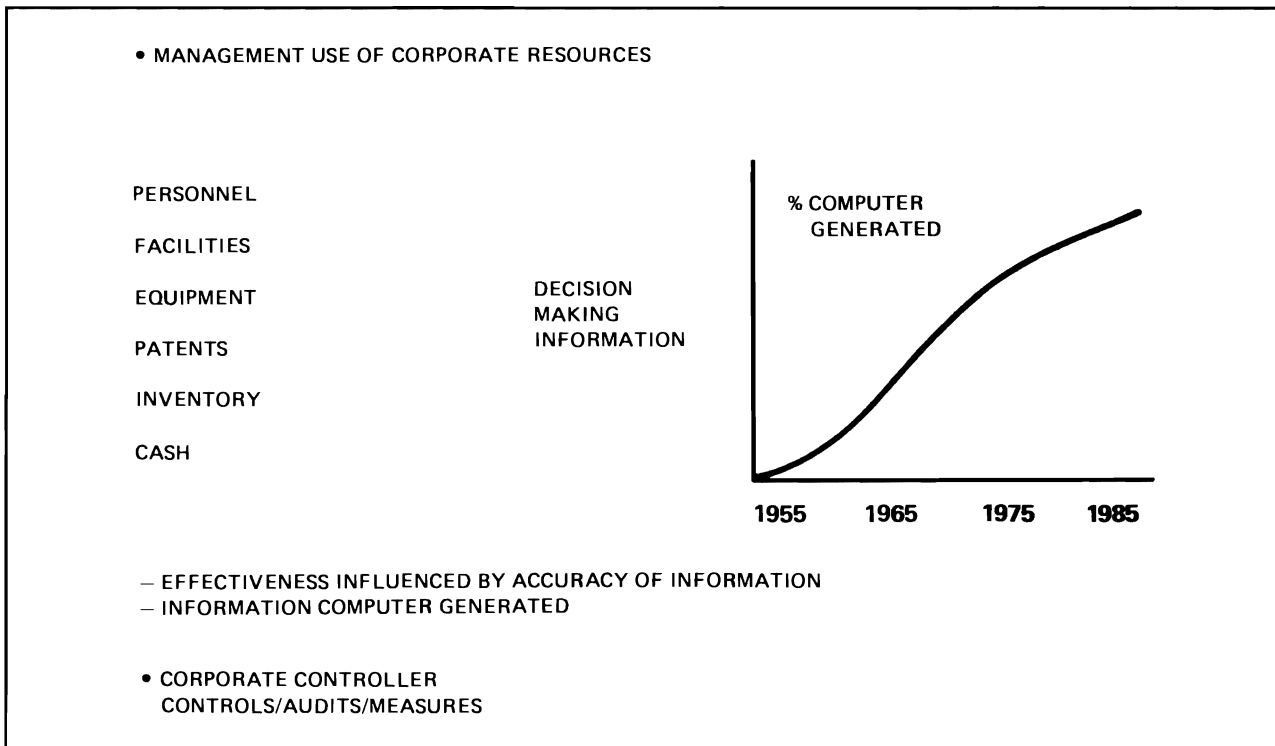


Figure 5.4.2 Management Use of Corporate Resources

Figure 5.4.2 identifies some of the more obvious resources of a business or corporation. Management attempts to use these resources as effectively as possible to maximize the profits and growth of the business. Recognizing the value of these resources, management institutes numerous plans, practices, and controls to both protect and develop them. An example of this management control is the functions performed by the office of the corporate controller, which monitors, measures, and evaluates just how effectively the resources are being used. It is interesting to note that while data processing is not listed as one of the obvious corporate resources (possibly because the DP budget is typically less than 5% of the total corporate budget), it provides most of the information that management uses in making decisions concerning all the other resources of the corporation. In a sense, data processing and the information generated by data processing, control the business. But, how does the business control the validity of the data being processed?

Before answering this question let's step back a moment and examine a basic concept about data. Any piece of data can be thought of as having two dimensions - content and description of content and usage. For example, an inventory balance may have a content value of 1,000 and a description that specifies "3/4-inch galvanized stove bolts, contains 5 numeric characters, is accessed by the shipping department, and is used in INVENUPDATE (inventory update) and INVENQUERY (inventory query) programs".

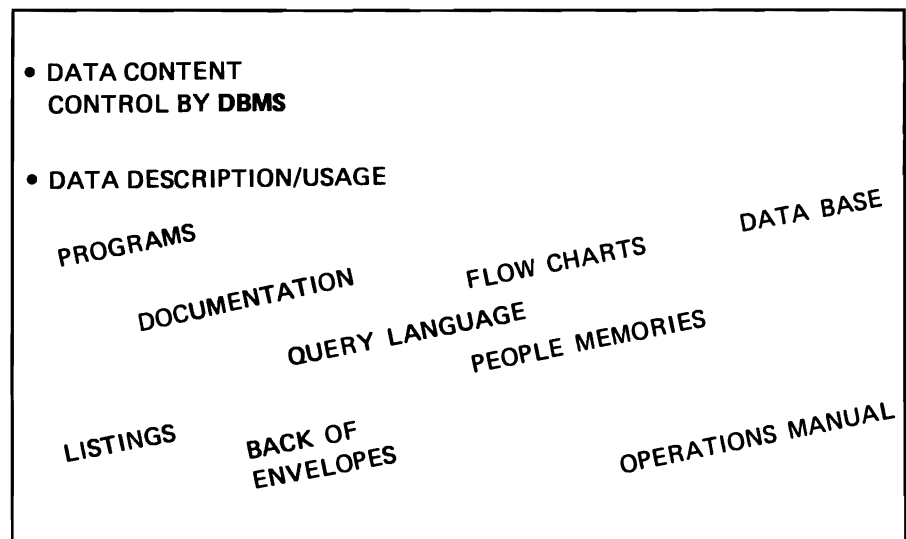


Figure 5.4.3 Data Usage and Content Description

As a partial answer to the question, the data base manager can be used to place controls around the content of the data to provide confidentiality and integrity. The need to do this is generally well accepted.

However, not nearly as well understood or accepted is the need for controlling the description and usage side of data. Data descriptions and usage tend to be dispersed. There is no single authoritative place to turn.

You may ask, "Why is this a problem?" First of all, if the descriptions exist in a half dozen places, then, whenever a change is made, someone has to make sure that all these descriptions are updated. If this isn't done, then inconsistencies are created that no one is aware of. Is item number the same thing as part number? And what about UNITCOST as marketing defined it and as accounting defined it? Are they really the same?

All this leads to unproductive work, which cannot be solved by the use of DB/DC alone.

What is needed to complete the answer to the question is a method of controlling and maintaining the description of data, allowing all parties to access the descriptions. In other words, a dictionary of data descriptions and usages.

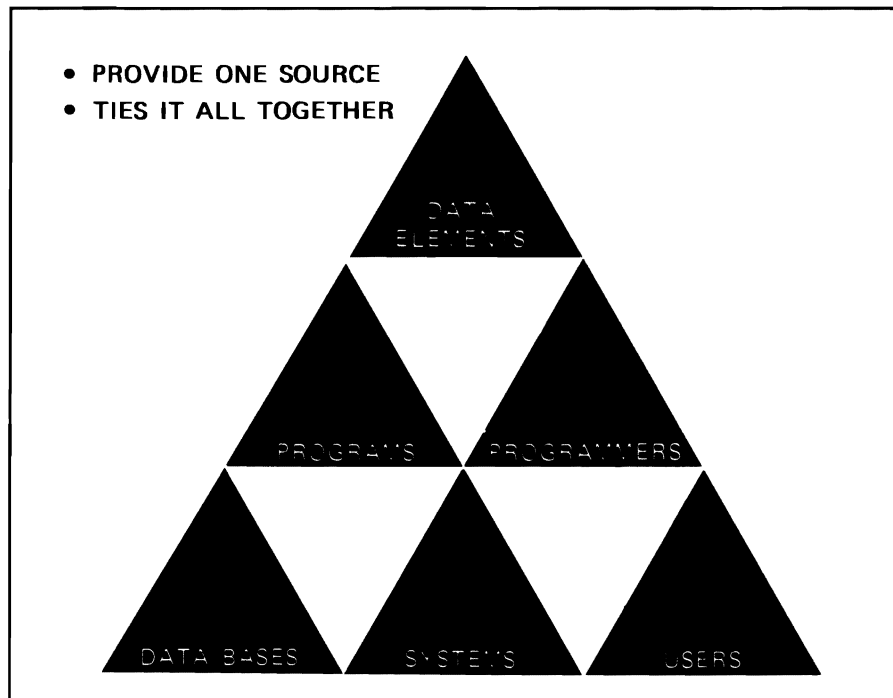


Figure 5.4.4 The Solution: Data Dictionary

Data Dictionary

A data dictionary is a central repository of information about all data processing activities. It contains information about items such as jobs, programs, transactions, and reports. An automated data dictionary takes advantage of the power of a computer and a data base management system to provide efficient facilities for the creation, manipulation and viewing of this information.

Every installation utilizes some form of a dictionary whether it be maintained manually or as a computer application. Every lead programmer in every application area maintains a dictionary that contains descriptions of records and fields, input transactions, and reports. The major problem with these individual application dictionaries is that they usually are out of date. There will be inconsistencies in the information maintained between the individual application areas depending on the time frame in which they were developed, the skill of the developer, and the care taken to document the changes to the data by the developing programmer. Finally, with few exceptions they will almost never be complete. In practice, a central automated data dictionary has obvious advantages for keeping track of the complex, changing data in data base systems.

For this reason, the IBM DB/DC Data Dictionary program product was developed as a management tool designed to assist in managing and controlling data processing resources. It is a software product which interfaces to CICS/VS to provide online access to the dictionary information in its files.

This data dictionary program can do for the description and usage of data what the data base manager did for the content. The Data Dictionary provides what can be thought of as a series of predefined compartments for the storage of data descriptions. It can also keep track of relationships that may

exist between them. For example, what data elements are used by which programmers or which users are affected by a change in a given data base.

It is the one authoritative source where all definitions and cross-relationships of data can be stored and controlled.

This product can maintain similar information about data processing; information such as which jobs are in production, what data base records they access, or who is responsible for them.

The IBM Data Dictionary complements DL/I in several ways. First, DL/I does not provide any documentation for data bases, programs, transactions, and users. Two data bases could be 95% redundant, and DL/I would not recognize this condition nor take advantage of it. In addition, what information DL/I has is written in programming languages and is understandable only by programmers.

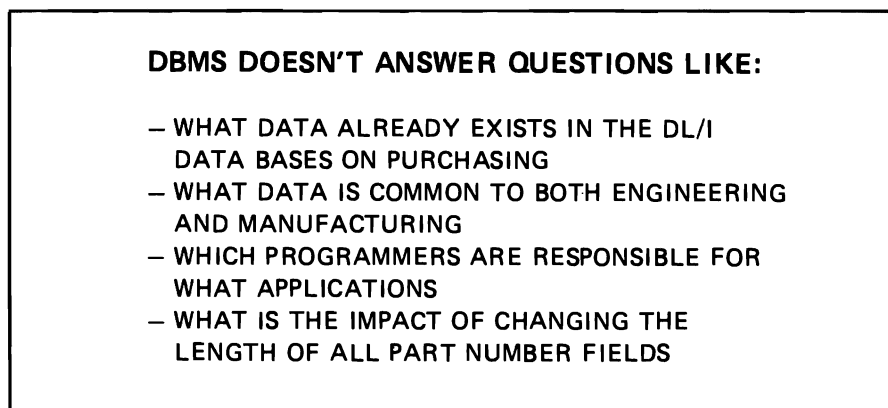


Figure 5.4.5 Questions about Data

If we look at the questions listed in Figure 5.4.5, DL/I would not provide answers to any of them. An installation needs to answer questions like these in order to manage its data processing resources. Hopefully, the answers are available. The question is how many different places must we look and how many different people must we ask in order to get the answers? It is the purpose of a data dictionary product to provide answers to questions like these.

With the help of the dictionary the impact of proposed changes can be more readily predictable. And with a good understanding of where data redundancies occur the dictionary can help identify application areas that should be included in data base designs.

Many customers utilize the data dictionary just for the simplification and productivity it provides for DL/I usage. Other reasons and benefits are shown in figure 5.4.6.

	MANAGEMENT	DATA BASE ADMIN.	SYSTEM ANALYSTS	PROGRAMMERS	END USERS	DP OPERATIONS
• DATA DOCUMENTATION	X	X	X	X	X	X
• DATA REDUNDANCY		X	X	X	X	
• APPLICATION DEVELOPMENT		X	X	X	X	
• DATA BASE DESIGN		X	X	X	X	
• DATA CONTROL	X	X			X	
• MAINTENANCE		X	X	X		
• IMPACT OF CHANGE		X	X	X	X	
• RECOVERY/RESTART		X	X	X		X
• COMMUNICATIONS	X	X	X	X	X	X
• STANDARDS	X	X	X	X	X	
• AUDITING	X	X			X	
• DEPENDABILITY OF DATA	X	X	X	X	X	X

Figure 5.4.6 Data Dictionary Benefits Potential

The IBM Data Dictionary program product provides a control vehicle to be used to assure the documentation of data processing is current, consistent, and complete. But to do the entire job of data administration requires more than software packages.

Data Base Administration

In a Data Systems Environment, recovery capabilities are essential to help ensure acceptable data processing availability. Greater control is needed over who has access to which portions of the data bases.

Potential conflicts that might result from many programs attempting to access the same file at the same time must be overcome. Finally, in an environment where data is shared, it is even more important to be able to pinpoint and isolate update responsibility.

As we have seen, the IBM DB/DC products offer functions that can assist in the handling of these tasks and responsibilities.

But who should actually perform these tasks? If they are assumed by individual application programmers, then decisions, such as the location of update and access capabilities within an organization, will be made from individual application viewpoints rather than from a corporate viewpoint.

Whose responsibility is it to control, audit and protect this corporate resource and other data processing resources in an organization? Is it the responsibility of the data processing manager or is each application area responsible for developing the rules and procedures necessary to protect their data bases?

The best answer is a central data base administration function with the authority to cut across department lines to establish access and update requirements. Physically, this may consist of an individual, a department, or an assignment for an existing individual. This person or department is responsible for executing the control tools provided by the data systems software. This removes the need for each application programmer to learn and use these control tools.

The functions of data base administration include:

- providing standards for controlling the administration of the data bases and their use.
- providing guidance, review and approval of data base design.
- determining the rules of access to the data bases and monitoring their security.
- controlling the data base integrity and availability, monitoring the necessary activities for reorganization and back-up/recovery.
- enforcing procedures for accurate, complete, and timely updates of the data bases.
- approving the operation of new programs with existing production data bases.

A data administration function usually does the initial planning for the installation's use of the dictionary product, including establishment of installation standards, naming conventions, and controls. Another task of the administration function might be the education of programmers and users on the use of the automated dictionary.

The automated data dictionary is an aid to the execution of all of these central data base administration responsibilities.

Summary

In summary, we see that a data dictionary does the expected task of documenting fields and records, and is often utilized just for that purpose. The data dictionary can also be extremely valuable as a tool for management understanding and control of the data processing operation. It provides access to the information required for effective management and protection of valuable corporate data. These benefits that the data dictionary provides an installation are greater when data processing management uses the dictionary product as the tool to manage the corporate data resources.

But with all it provides, the data dictionary is only a tool which must be properly utilized. This requires an organizational function, Data Base Administration. The primary purpose of this function is to coordinate and control the management and usage of the centralized data.

Although data base administration is the primary user of the dictionary, others find it valuable too. Application designers search the definitions to determine which subjects have been defined, what their current status is, and whether to make or recommend changes. They often use the dictionary to design new applications, with existing or modified definitions, and maintain installation standards of consistency and completeness of documentation. Application programmers use the dictionary to obtain data definitions for

insertion into new programs. Data processing managers and technical personnel use the dictionary to obtain varied reports on their current data resources.

End users of the data - those involved in daily activities of the enterprise - are able to use the dictionary output to become familiar with the data available to them. This familiarity allows them to more easily define their data requirements to the data processing group.

DB/DC Data Dictionary is a product which can reduce the complexity of managing data resources both inside and outside data processing. It is also flexible enough to be used for other purposes such as keeping track of reports, transactions, communications, and hardware. In short, the dictionary can become a tool to help manage all aspects of an entire installation.

Exercise 5.4

1. All data has two dimensions, _____ and _____.
2. A _____ is a central repository of information about data.
3. The software tool provided by IBM to help with the administration and management of data is _____.
4. (True or False) The information maintained by the IBM software product for Data Administration is accessible from a terminal through an online program.
5. The coordination and control of the usage of data is a function of _____.

Solution 5.4

1. a) content
b) description (of content and usage)
2. data dictionary
3. IBM DB/DC Data Dictionary
4. True
5. Data Base Administration

Topic 5: Data Delivery

The terms "data" and "information" have appeared throughout this unit. As a working definition, information is data that has been processed, arranged, and presented to the user in an immediately usable format.

We have examined the concepts of a data base (a collection of data), a data communications function (a way to transmit data/information), and a data dictionary (recording and controlling data descriptions), but have not talked about the data becoming information. This usually requires data processing professionals to write and implement application programs.

As previously explained, there are products available to help us effectively manage, store, transmit, and control the data which exists in the organization. Also available are ways to improve the process of turning data into information.

There are two basic ways in which information can be made available with less effort.

One is a shortcut in the effort required to do conventional programming. The other is to use tools that make it possible for users to do more of the work themselves.

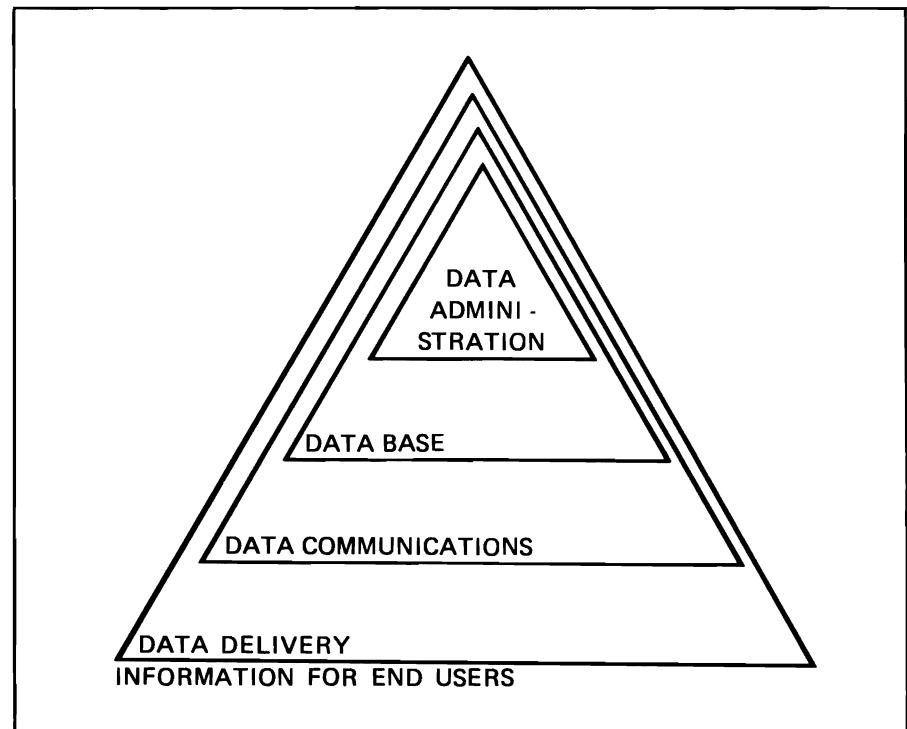


Figure 5.5.1 The Data Systems Environment

And that brings us to the final element making up the Data Systems Environment - data delivery. IBM has a variety of software products which assist in changing data into useful information for users. Some are for programmers, others are for users themselves.

Prewritten Application Programs

For shortening the work programmers have to do, there are prewritten IBM application programs built around a DB/DC base. A good selection of these already exist for personnel, manufacturing, health, and finance applications and their number is growing.

Online Interactive Tools

The programming staff also can make use of a variety of online interactive tools designed to help speed the program development process. This includes the Interactive Computing and Control Facility (ICCF) described in another unit of this document.

For users, there is a selection of products designed for direct interactive problem solving by staff professionals, such as engineers and financial analysts. These include PLANCODE, BASIC, and A Programming Language (APL).

Application Development Tools

Also there are rapid application development tools, based on a DB/DC environment, that allow online transactions and reports to be developed in a fraction of the time needed for conventional programming.

DMS/CICS/VS

The most significant application development tool for a DOS/VSE user is the Development Management System/CICS/VS (DMS/CICS/VS). It simplifies the implementation of a DB/DC system by allowing the programmer to select pre-programmed functions in the sequence necessary to produce the desired information on a video terminal. This is viable due to the fact that the code required to move data from a data set or data base to an output terminal is essentially the same regardless of the content of the data or the application. The variables are the format of the data, the manipulations of the data, and the format for display. These variables are all that it is necessary for the programmer to specify and he is provided a series of "fill in the blanks" forms to accomplish this.

This fill-in-the-blanks technique provides a fast and efficient method of defining applications. Programming effort for implementation, maintenance and documentation is reduced, freeing programmers to concentrate their efforts on other applications.

The ability to use predefined functions to perform simple calculations and to edit data through the use of calculation/edit statements simplifies application development by significantly reducing the need for conventional programming language skills.

DMS/CICS/VS allows access to existing data bases and data sets in the online environment and provides the capability for the user to inquire into a data base in a simple, straightforward manner and to display the results in a user specified format. The user can also interactively update and add to the data base. In addition, messages that originate at one terminal can be directed (switched) to other terminals. Output can be sent to printing terminals and files can be rebuilt from prior versions and audit trail logs.

DMS/CICS/VS provides online data integrity and security by selective access to display screens and protection against the concurrent changing of a data record by both DMS/CICS/VS transactions and other CICS/VS applications.

DMS/CICS/VS provides for a simplified application design which allows user specification of application dependent information. The simplified design also allows implementation by nonprogramming personnel.

The DMS/CICS/VS program product operates under control of CICS/VS which provides terminal, storage, data, program, and file management services in a multi-tasking environment. The DMS/CICS/VS user also has access to all other CICS/VS facilities. DMS/CICS/VS is a powerful tool for the development and maintenance of online systems and is one of several significant software tools to help convert the data into the information requested by the user and to help do that conversion in an expedient manner.

Summary The data delivery element of a Data Systems Environment consists of a series of aids, IBM software products, grouped into the areas shown in Figure 5.5.2.

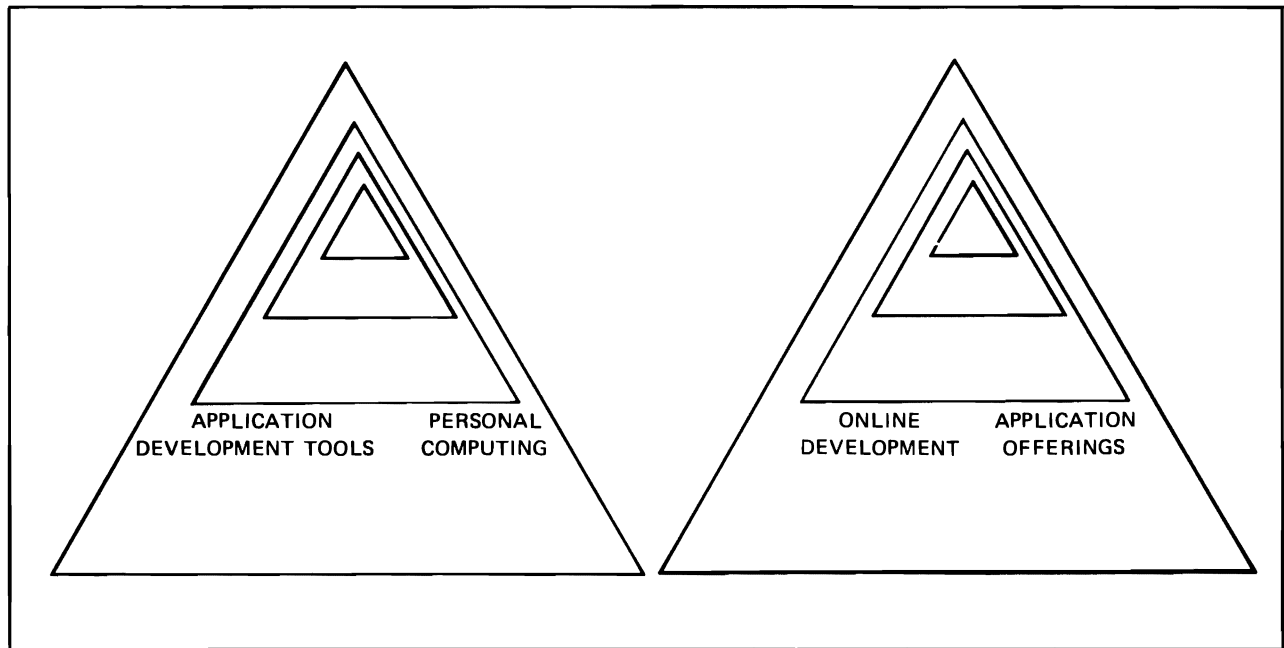
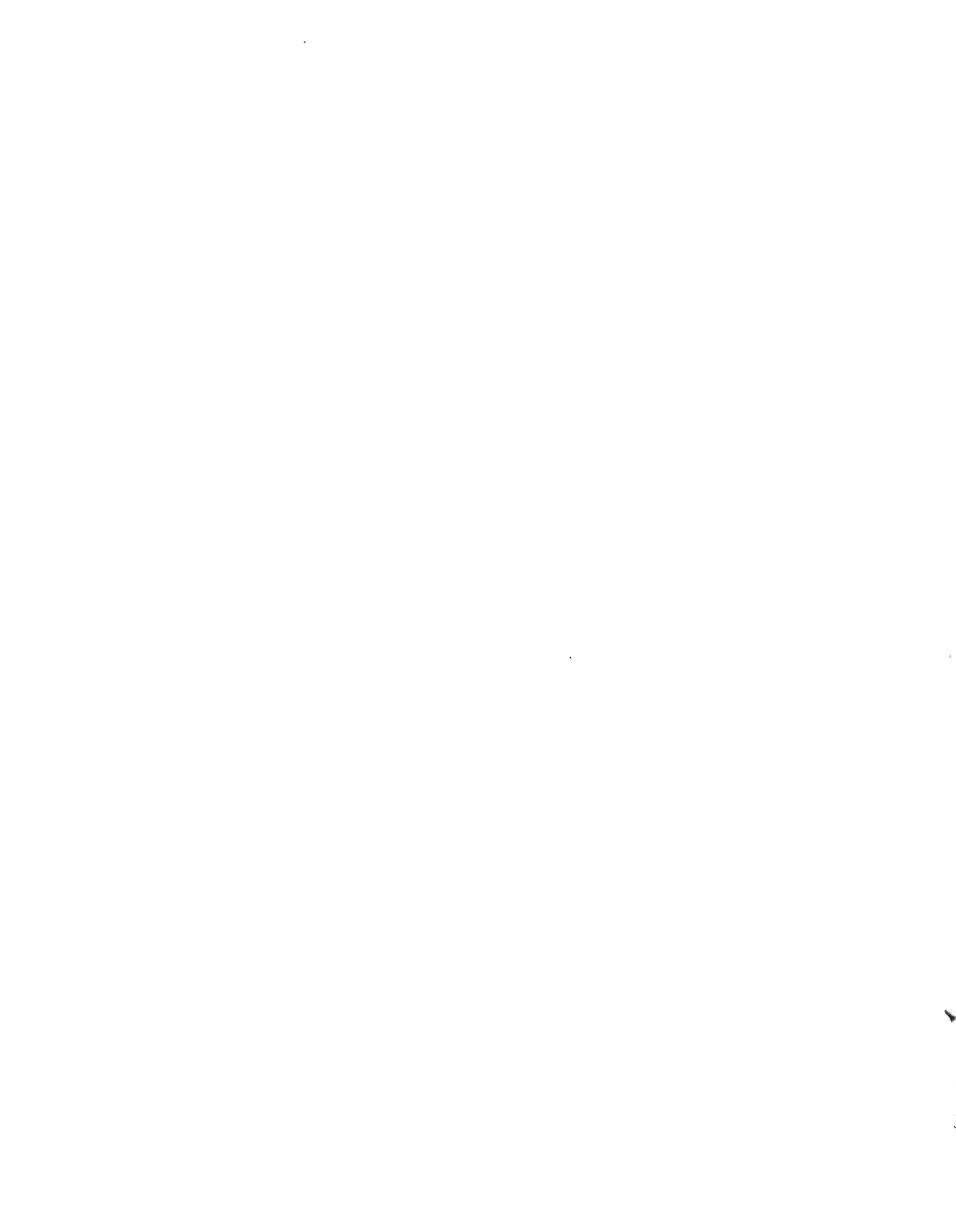


Figure 5.5.2 Data Delivery Products

The use of these aids should enable a reduction of the imbalance which typically exists between program/application maintenance and new application development.



Exercise 5.5

1. State the relationship between "Data" and "information" as they are used in this text.

2. The process of providing the user with information resulting from the processing of raw data is called _____.
3. From the list below, select the items which are used to shorten and simplify the process of providing information.
 - a. Pre-written application programs
 - b. Online interactive tools
 - c. ICCF
 - d. User application program
 - e. APL
 - f. DMS/CICS/VS
 - g. All of the above
 - h. None of the above
4. (True or False) The program code needed to move data from a data base to a user terminal varies significantly from one installation to another.
5. (True or False) DMS/CICS/VS includes a pre-written online program which performs functions based on user-specified application dependent variables.
6. (True or False) Providing current, reliable information in an expedient manner is the basic purpose of data processing.

Solution 5.5

1. Information is data that has been processed and is available for use.
2. data delivery
3. All of the above
4. False. It is because the code required to move data to and from a terminal or file is essentially the same in most environments that a product such as DMS/CICS/VS is possible.
5. True
6. True

Topic 6: Summary

In the beginning of this unit, the idea of the value of data as a corporate asset was presented. This led to the need for efficient storage and management techniques for the control of that valuable asset: data. Then it was apparent that some method of timely accessing of that data would be imperative for its proper use. From these requirements and the expansion of data processing techniques and knowledge evolved the concept of a Data Base/Data Communications (or online) system. To minimize the effort involved in implementing a DB/DC system, IBM developed two program products, DL/I and CICS/VS. As depicted in figure 5.6.1, these system software programs are extensions of the operating system which interface with the application program to provide the functions described earlier in this unit.

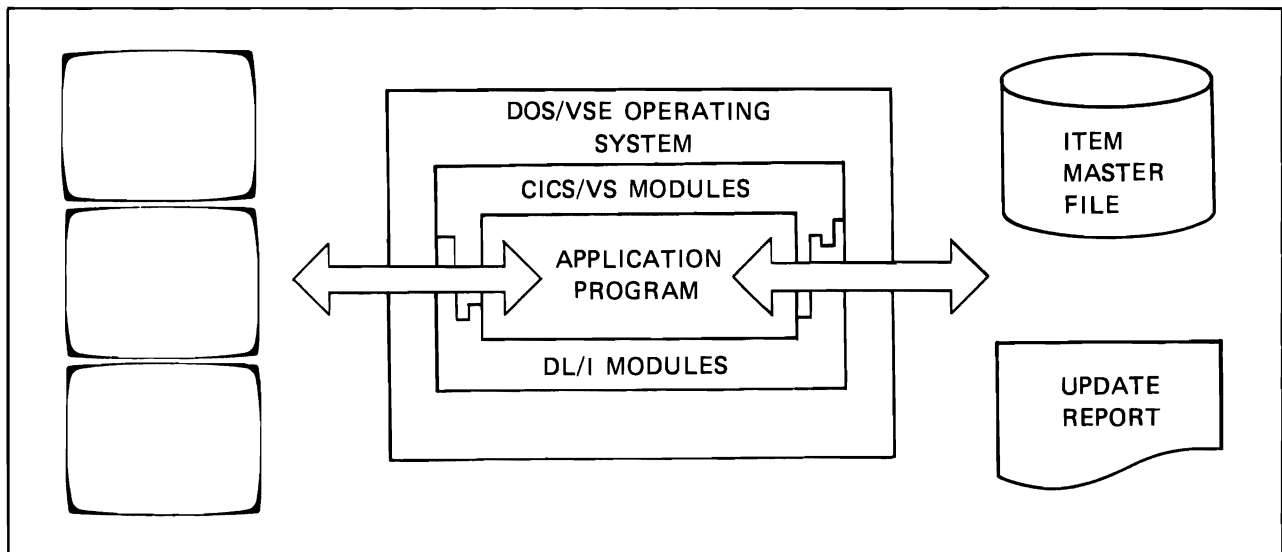


Figure 5.6.1 The Data Systems Environment

Corporations are now evaluating computer systems, not only with regard to programming systems and hardware, but also in relation to the information needs of the total corporate environment. Demands for programming applications that interrogate and maintain large centralized information files are increasing. From these increased demands and attention of corporation management, arose new problems; the standardizing and control of the data management environment. The need for defining and tracking the many usages of data has become imperative. IBM again produced a software product (the Data Dictionary) to use as a tool in this function of controlling the data and its uses.

In today's fast-paced business environment, there is a constant need for massive amounts of information and it quickly loses its value if delayed in delivery to the user. This has caused the need for additional products to assist the programmer in his functions of providing the application programming. These tools are of many types and uses, but all of them are aids in the data delivery process.

The result of these changing needs and the products developed to meet them is the latest stage of the continuing evolution of data processing: the Data Systems Environment. The implementation of IBM software products in each

of the four data systems environment elements is the way to satisfy the informational needs of business today.

Unit 6:

Data Systems Products

Introduction Unit 5 presented the data systems environment and identified four major IBM program products used to implement that environment. This topic addresses those four products (DL/I DOS/VS, CICS/DOS/VS, DB/DC Data Dictionary, and DMS/CICS/VS) in more detail.

The sequence of the topics presented in this unit follows the logical flow or relationship between these four products. DL/I DOS/VS is used to manage the large collections of data needed by the corporation. CICS/DOS/VS provides the functions allowing timely access of the data maintained by DL/I and functions to control the online environment. DMS/CICS/VS provides an effective and productive technique to generate the application programs in the CICS/DOS/VS environment. The last product discussed, DB/DC Data Dictionary, is a tool to assist in the management of the usage of the data and other resources in the DB/DC environment.

Objectives The objectives of this unit are:

1. To gain an understanding of the organization and basic functions of each of the major Data Systems products.
2. To be able to relate these basic functions to the benefits provided by each of the products.
3. To understand the manner in which these four products interrelate.
4. To learn basic terminology and expressions used in the literature on each product.
5. To gain an appreciation for where and when to use each product.

Average Study Time 3 to 4 Hours

Topic 1. Data Language/I DOS/VS (DL/I DOS/VS)

The establishment of data base applications involves both design and implementation activities. The design activities determine the data and data organization which will satisfy the application requirements and meet the installation's objectives for data security, integrity, and redundancy. Once designed, the application programs and the data base(s) they access must be created and maintained. To assist this implementation activity, IBM has developed a data base manager called DL/I DOS/VS.

DL/I provides methods of data organization which simplify the design tasks. It also provides an interface which accepts data base access requests from application programs and invokes the appropriate DOS/VSE data management services to satisfy those requests. This interface is provided for programs in both batch and online environments.

The remainder of this topic will examine the data structures and organizations processable by DL/I. Also presented is the interface between DL/I and an application program and some additional features and functions included in the DL/I DOS/VS program product.

Data Base Structure and Organization

DL/I provides device and data independence, two of the basic attributes of the data base concept. This means that the user's programs are separated from the access methods, storage organizations, and storage devices used for the data. Since the data is stored and retrieved through organized methods known by DL/I, there must be a symbolic representation of the data for the user program to reference. Application programs deal with *logical data structures*, where *logical* refers to the manner in which the data is seen (or viewed) by the program. On the other side, DL/I deals with *physical data structures* where *physical* refers to the manner in which the data is stored on a tape or direct access device.

Logical Data Structure

The logical representation of the data in a data base is called a *hierarchical structure*. This structure, sometimes called a tree structure, represents the dependency relationships of data and subordinate data. This structure is similar to the organization chart within a business where officers are subordinate to other officers as shown in Figure 6.1.1.

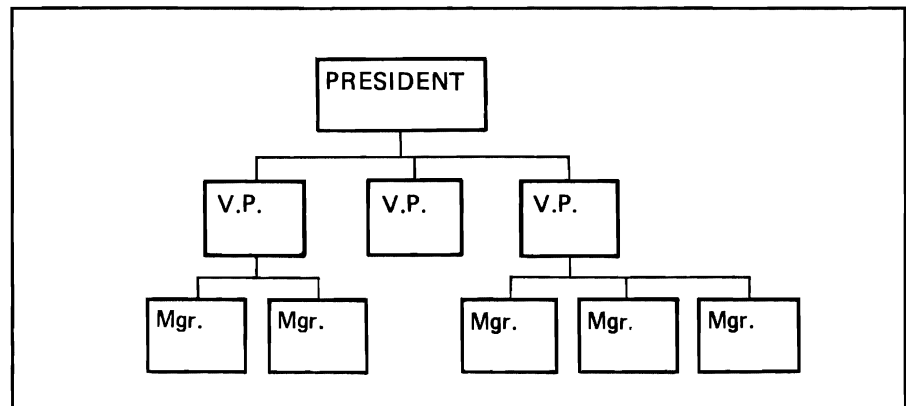


Figure 6.1.1 Hierarchical Structure of an Organization

Another way to illustrate the logical structure of a DL/I data base is to take a look at a typical programming application and compare the methods used to handle it (that is, data base versus non-data-base). Figure 6.1.2 shows a sequentially organized employee master file on disk storage. This file, containing information about each employee, is used to process a payroll application of 30 programs.

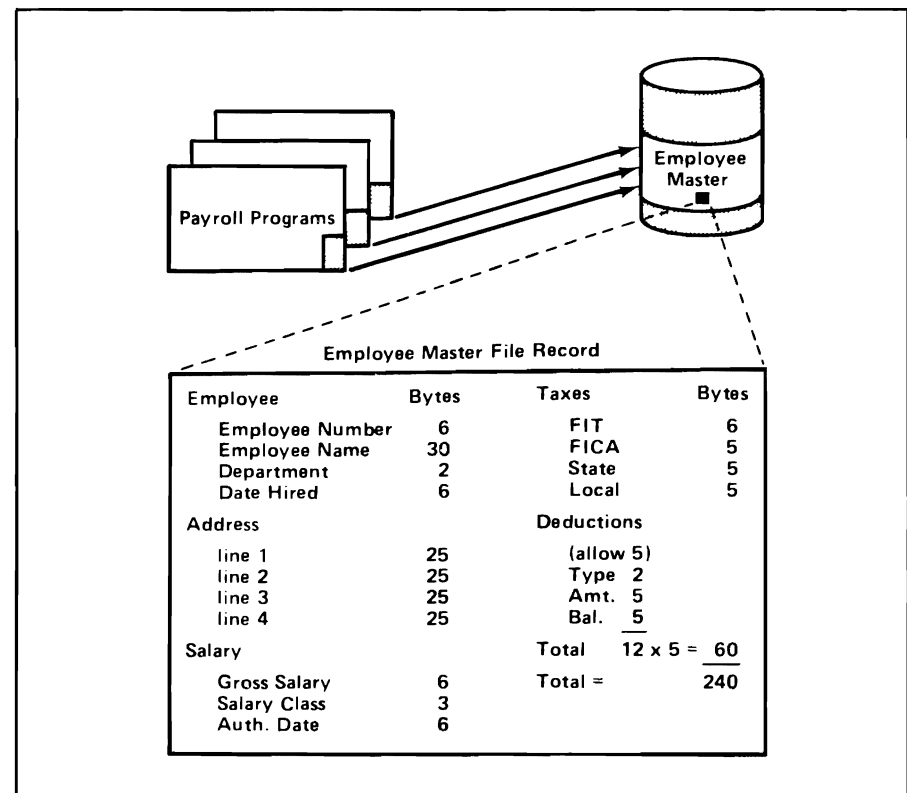


Figure 6.1.2 Sample Payroll Application

Its contents are typical. The control information includes employee number, name, department, and date hired. The address can be up to four lines. Payroll information includes salary and year-to-date tax data, with provisions made for five deductions.

Data can be thought of as having the same kinds of relationships as in the organization structure of Figure 6.1.1. For example, the employee master file could be structured as shown in Figure 6.1.3 to show salary as information which is subordinate to employee because it is impossible to have a salary that isn't tied to an employee. Also, an address all by itself would be meaningless if it were not tied to a particular employee. So in this case the employee information is the "boss", if you will, of both salary and address. Likewise, taxes cannot logically exist unless there is a salary.

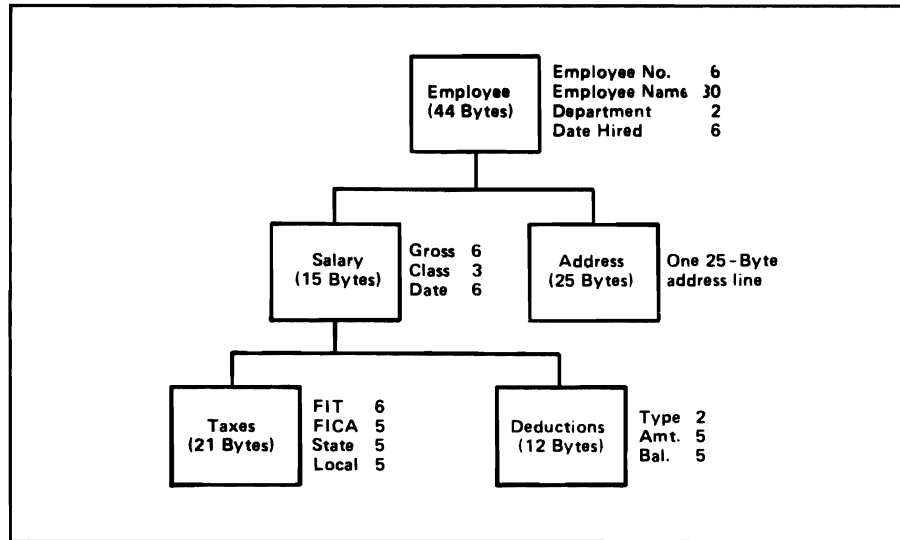


Figure 6.1.3 DL/I Hierarchical Data Structure

Notice that the data fields of the record are identical but have been separated and placed into these logically structured groupings. Address and deduction groups of data are only shown once because all lines of address and all deductions have the same fields in the same sequence.

Elements of a DL/I Hierarchical Structure

Each box shown in Figure 6.1.3 is known in DL/I terms as a *segment*. In this case, the structure depicts five different types of segments (Employee, Salary, Address, Taxes, and Deductions). Each segment consists of one or more *data fields*. The Employee segment, for example, has the employee number, name, department number, and date of hire. In this example, the Employee segment contains the identifying information for the data structure (name and number), and it is called the *root segment*. There is only one root segment in a data hierarchy. All others are *dependent* segments. The Salary and Address segments relate to the Employee segment, and depend on the Employee segment for their full meaning (Whose salary? Whose address?).

Each type of segment can vary in length (Employee is 44 bytes, Salary is 15 bytes, etc.), and segments of the same type may also vary in length.

There can also be many *levels* of dependency. For example, the Tax and Deduction segments are subordinate to and depend on the Salary segment. The Salary segment, in turn, depends on Employee, the root segment. This defines the Tax and Deduction segments as a part of the third level of the hierarchy.

The basic building element of a hierarchical data structure is the *parent/child relationship* between segments of data. The root segment (Employee) is also the *parent* of all the segments (that is, Salary and Address) that depend on it. The dependent segments, Salary and Address, are called *children* of the parent segment (Employee).

The same relationship exists down the structure (Salary is the parent of Taxes and Deductions; Taxes and Deductions are children of Salary).

Each *occurrence* (or instance) of a parent segment has associated with it any number of occurrences of a child segment type. Note the distinction between a *segment type* (the kind of segment), and the *segment occurrence* (the segment and its particular contents and location). In Figure 6.1.4 the parent segment, Employee, has four occurrences of the child segment type, Address. Each child segment type has associated with it one occurrence of a parent segment. Different occurrences of a particular segment type under the same parent, such as address lines, are called *twins*.

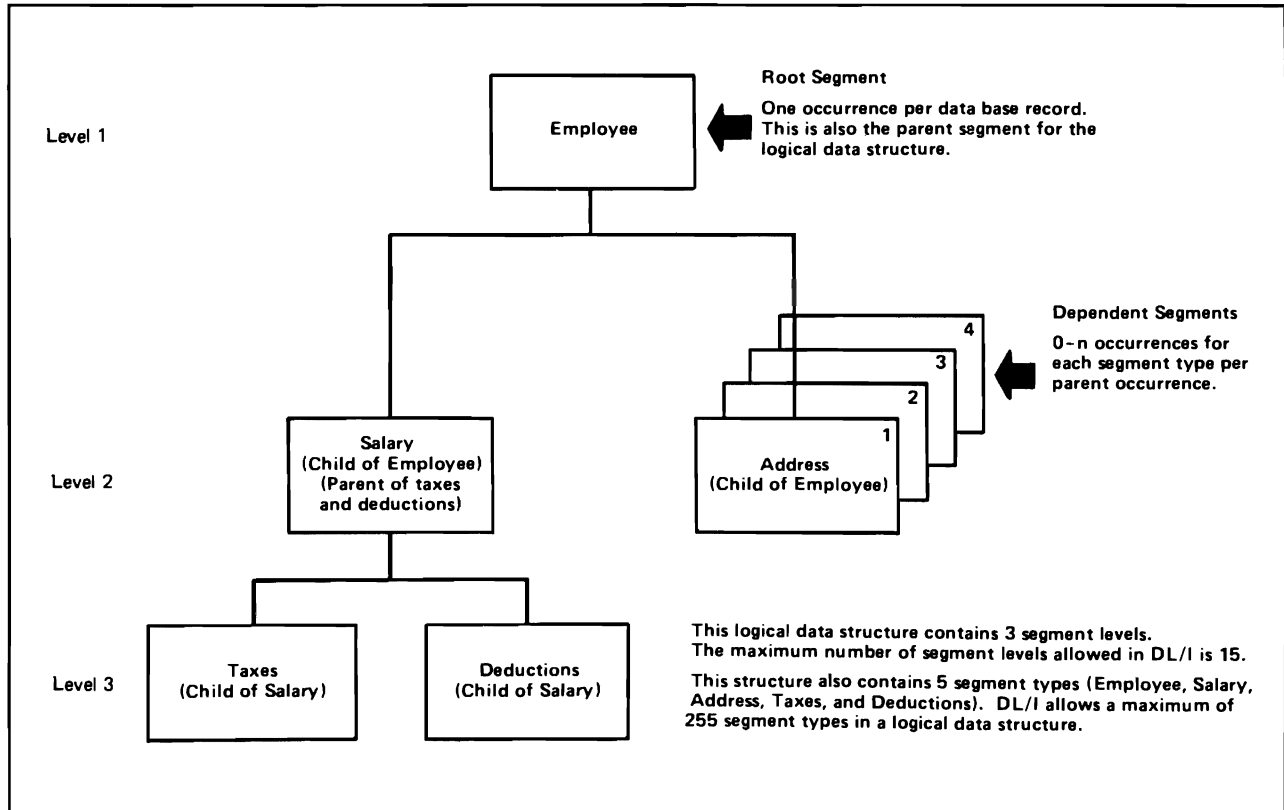


Figure 6.1.4 Logical Data Structure, The Program View

All the parent/child occurrences for a given root segment are grouped together to form a *data base record*. In this example, although the logical data structure contains 5 segment types, the data base record actually contains 8 segments because of the multiple occurrences of the address segment. Data base records for a given logical data structure may vary in size according to the number of occurrences of a given segment. The collection of all these like-data-base-records is a *DL/I data base*.

Since each dependent segment in the hierarchy has only one parent (immediate superior segment) the hierarchical data structure is sometimes called a *tree structure*. Each branch of the tree is called a *hierarchical path*. A hierarchical path to a segment contains all consecutive segments from the top of the structure down to that segment.

DL/I allows a wide variety of hierarchical data structures. The maximum number of different segment types is 255 per hierarchical data structure. A maximum of 15 segments levels can be defined in a hierarchical data struc-

ture. There is no DL/I restriction on the number of occurrences of each segment type.

The significance of the hierarchical structure is important in application programming. In DL/I, the user program no longer sees a physical record. It sees a series of segments in a parent/child relationship. User programs request segments of data, rather than physical records. The concept of segment *sensitivity* allows a user program to be restricted to seeing (or viewing) only those segments of information that are relevant to the processing being performed. For example, an application program could be written to see only the Employee and Address segments of the data base record shown in Figure 6.1.3. The program need not be aware of the existence of the Salary segment and its children. This also means that the data which is not a part of the user program's view is not accessible by the program.

Based on the previous discussion of the structure of the DL/I data base, the following definitions apply:

- **Segment.** A data element containing one or more logically related data fields. A segment is the basic data element that is transferred between the application program and DL/I and upon which the user can define sensitivity. A segment may be fixed length or under certain conditions variable length.
- **Sensitivity.** A means by which the user defines which subset of the data within the data base can be accessed by an application program and what operations may be used in accessing that subset of the data base. This defines a logical view of a data base.
- **Logical data base record.** A hierarchical structure of related segments of one or more segment types. Each segment type may have a unique length and format. As viewed by the application program, the logical data base record is always a hierarchical tree structure of segments.

Physical Data Structure

What has been described above is the logical data structure as viewed by the program. As explained earlier in this topic, there is a physical data structure which is managed by DL/I.

The data described by the DL/I logical data base structure in Figure 6.1.3 could be physically stored as shown in Figure 6.1.5.

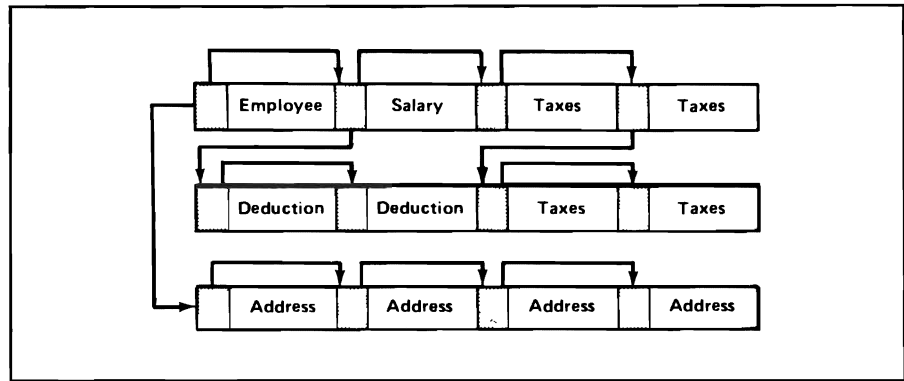


Figure 6.1.5 Physical Data Structure

Employee, the root segment, points to its children (Salary and Address). Salary also points to its children (Taxes and Deductions). The multiple occurrences of segments are also linked. It is through these pointers that DL/I retrieves segments of data. The application programmer, however, need not be aware of the physical storage.

Physical storage is accomplished through the use of two unique DL/I storage organizations: *hierarchical sequential* and *hierarchical direct*. These organizations primarily differ in the manner by which segments are related and the methods of data access.

Hierarchical Sequential

In a hierarchical sequential organization the segments of a data base record are placed on the physical storage device physically adjacent to each other and in the same sequence as defined in their logical structure.

DL/I has four data access methods utilizing the hierarchical sequential organization. Simple Hierarchical Sequential Access Method (Simple HSAM) and Hierarchical Sequential Access Method (HSAM) are used for sequential storage on tape or disk and use the DOS/VSE Sequential Access Method (SAM) for data management services. The other two DL/I access methods using the sequential organization, Simple Hierarchical Indexed Sequential Access Method (Simple HISAM) and Hierarchical Indexed Sequential Access Method (HISAM), include an index to the data base records for faster and more efficient random access. HISAM and simple HISAM use VSE/VSAM to provide the data management services.

Hierarchical Direct

In the hierarchical direct organization the segments of a data base record are placed on the physical storage device in a random manner. There is a *direct physical address* associated with each segment denoting its location on the physical device. This address is used by DL/I as a pointer which links the scattered segments in their logical sequence.

There are two DL/I data access methods which utilize the hierarchical direct organization, Hierarchical Direct Access Method (HDAM) and Hierarchical Indexed Direct Access Method (HIDAM). HDAM requires the user program to provide the direct physical address of the required segment while HIDAM

allows access through an index containing the direct physical addresses. Both HDAM and HIDAM utilize VSE/VSAM for operating system data management services.

Data Base Definition

At this point we have described the purposes of DL/I DOS/VS, the concept of logical structures and physical structures, and the physical methods used to store and process the data represented by the logical structures. In order for DL/I to manage the physical structure or for the user program to refer to the logical structure, these structures must be defined and the definition stored for repeated use.

The entire hierarchical structure (names of segments, sizes of segments, and hierarchy) as well as its physical attributes (the fact that it is on disk, organized sequentially, blocked 5) is kept in two tables external to the user programs. These tables are prepared during the design process and created and maintained independently of the application programs. The separation of this information from the application program forms the basis for data and device independence in DL/I user programs. Each non-DL/I user program includes a description of the physical attributes of the data whereas a DL/I user program has no knowledge of the physical structure, storage methods, or devices used for the data.

DBD (Data Base Description)

The first table is called the *DBD* (data base description). It describes most of the file characteristics that normally are included in a non-data base DOS/VSE application program. Each DBD is created from statements which define the hierarchical data structure and physical organization of the data base.

The DBD contains a description of the data base including the names of the fields, the names of the segments, their hierarchical relationship, and its physical organization and characteristics. The DBD is the master description of everything that is in the data base.

The DBD provides DL/I with the correlation between the logical data structure used in the application program and the physical organization of the data used by DOS/VSE. Since the data structure is external to the application program, it can be changed to a different physical organization without application program modification. New application data can also be added to this data base and not require a change to the original application programs. The concept of the DBD reduces application program maintenance caused by changes in the data requirements of the application.

The DBD is created from definition statements provided to a DBD generation program included with DL/I. This process of generating a DBD is called data base description generation (*DBDGEN*).

PSB (Program Specification Block)

The other table is called the *PSB* (program specification block). The PSB defines which segments of the data base a specific program requires (the logical data structure required by that application program). A PSB contains one or more *PCBs* (program communication blocks).. one for each hierarchical data structure the program intends to use. Each PCB defines the hierarchical (sub)structure of the physical data base. It specifies for each segment the kind of access allowed by the program (read only, update, insert, initially create, and delete). Every program that uses the data has a PSB; more than one program may use the same PSB. The PSB describes the logical data needed for the program (usually a subset of the entire data base).

The PSB is created from statements provided to a program known as the PSB generation utility. This process of generating a PSB is called program specification block generation (*PSBGEN*).

Using The DBD and PSB

Figure 6.1.6 shows that at execution time, DL/I uses the combination of the DBD describing the data base, and the PSB describing the data needs of a specific program to satisfy the requests of the application programs. DL/I acts as an intermediary between the application program and the data itself.

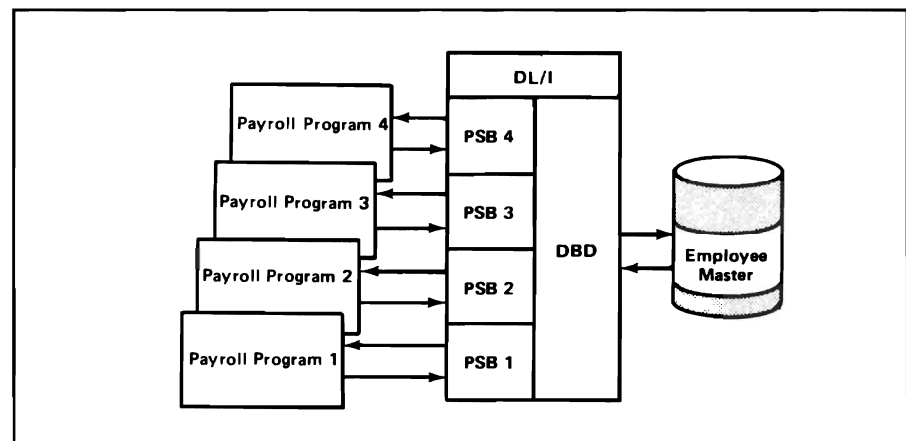


Figure 6.1.6 DL/I is the intermediary between the application program and the data base.

When a program needs a segment of information, it issues instructions to DL/I using a standard request format. DL/I, using the PSB and DBD, interprets the request. When necessary, DL/I issues the appropriate I/O command to obtain data from the data base and return the requested segment to the program.

Data Base Program Execution

In order to process a data base, the following program modules are required:

- The user application program containing DL/I requests.
- For each application program, a PSB (program specification block) that identifies each DL/I data base used by this program and describes how each can be processed by this program.

- For each DL/I data base, a DBD (data base description block) that describes the physical data base structure, the file organization, and the device type on which the data base resides.
- The DL/I processing modules.

In a batch environment, all these program modules are in the same DOS/VSE partition. The application program operates as a sub-program to DL/I. In the online environment, DL/I executes in the same partition as CICS/VS and services data base requests from CICS/VS user tasks through a functional part of DL/I called the *DL/I online processor*. The online processor establishes the connection between DL/I and CICS/VS for the accessing of data bases from online programs.

When a DL/I user program is to be executed, the DL/I modules are placed in a partition by DOS/VSE and they begin an initialization process. DL/I completes the initialization process by loading the user program into the partition and allowing it to begin execution. The application program can communicate with DL/I by issuing requests in a standard format called a *DL/I call statement*. A call statement can be a request for the addition, deletion, or changing of the data in the data base segments.

After processing a call statement, DL/I always returns information to the user program to indicate successful or unsuccessful completion of the request and to identify the segment processed.

Additional DL/I Capabilities

In addition to the previously described basic functions of DL/I there are additional features included in the product. Each feature increases the capability of DL/I as a full function data base management system.

Multiple Partition Support (MPS)

An enhancement to basic DL/I is *MPS (Multiple Partition Support)* which enables application programs executing in different DOS/VSE partitions to access the same data base concurrently. For example, online applications may retrieve data base records while a batch DL/I program is updating the same data base. These MPS functions are implemented using the multitasking capabilities of CICS/DOS/VS. This means that CICS/DOS/VS is a prerequisite for MPS.

Program Isolation

If multiple users in an online or MPS environment perform concurrent updates to the same data base record, there is a possibility that the changes made by each user would be reset by the next one to complete. Only the last changes would be accepted causing the resulting record to be incorrect. The various possibilities of this type of error affect the *physical integrity* or accuracy of the data. That is, the data is not preserved on the physical device in the same way intended as a result of some activity.

To insure the physical integrity of data, DL/I contains a feature called *program isolation (PI)*. It records the accessing of a particular segment for update purposes and prevents other user programs from accessing that same segment until the update is completed.

Field Level Sensitivity enhances data base integrity and security by allowing an application program to see only portions of a segment rather than the entire segment. This provides data independence and sensitivity at the field level instead of only at the segment level. The fields accessible by a program are specified in the PSB associated with that program.

Secondary Indexing To identify and to provide access to a particular data base record and its segments, DL/I uses *sequence fields* (or keys). Each segment normally has one of its fields denoted as the sequence field. However, not every segment type need have a sequence field defined. Particularly important is the sequence field for the root segment, since it serves as the identification for the data base record. Normally, DL/I provides a fast, direct access path to the root segment of the data base record based on this sequence field. This direct access is extended to lower level segments if sequence fields of the segments along the hierarchical path are also specified.

An additional capability in DL/I is the ability to define a *Secondary Index* in order to be able to retrieve records in a sequence other than the defined sequence fields of the segments. For example, we might want to be able to access a data base on employee name information or department number information instead of the employee number which is specified as the sequence field. The problem is that the programmer must develop an indexing approach and be sure to maintain the indices accurately. DL/I can automatically develop and maintain such indices. For example, if we request the records of people who are named Adams, DL/I will look first at the secondary index for Adams and then use the direct pointers contained in the index to obtain those records for us. This indexing capability can also be applied to subordinate segments, or even combinations of fields from different segments.

Logical Relationships In addition to the DL/I facilities discussed so far, DL/I provides a facility for HDAM and HIDAM data bases to interrelate segments from different data base hierarchies. For example, suppose that a new application, Job Incentives, needs the information in the Employee data base. One of the requirements of the new application is to produce information on the skills of employees including a description of each skill and job-standards data. The data required by the Job Incentive application is shown in Figure 6.1.7.

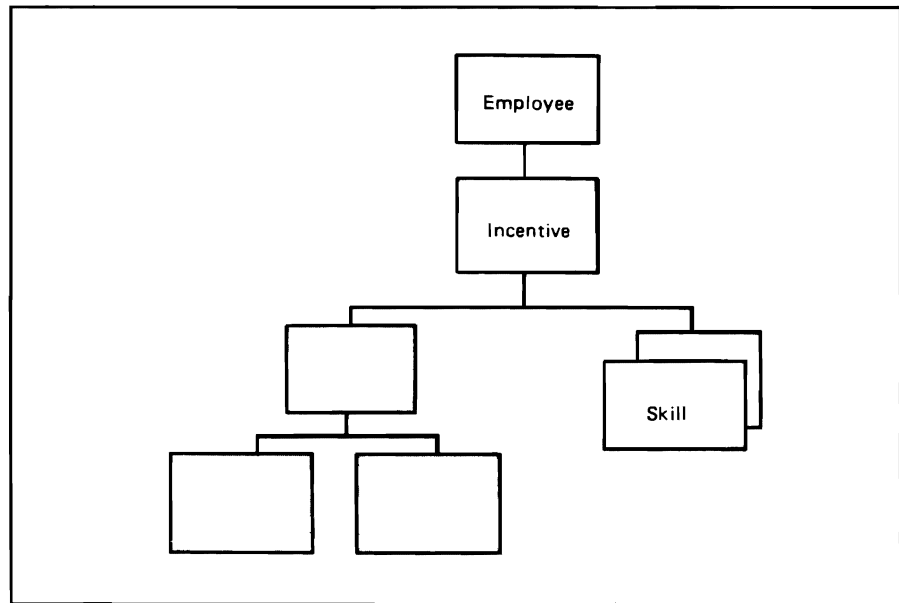


Figure 6.1.7 Job Incentive Application to be Included in Employee Data Base

Let's also assume the existence of another application, Skills Inventory, whose data base is as shown in Figure 6.1.8.

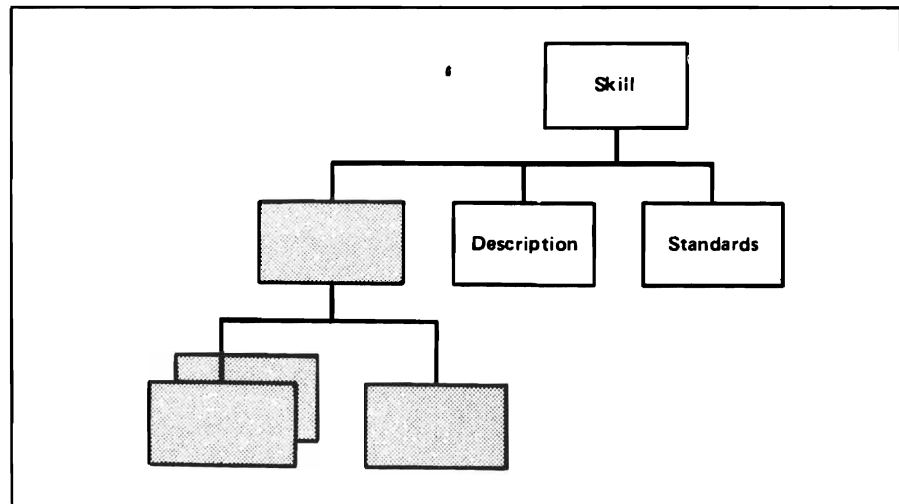


Figure 6.1.8 Skills Inventory Data Base

As you can see in Figure 6.1.8 the data that is needed by the Job Incentive application is contained in three of the segments of the Skill data base: Skill, Description, and Standards. The new application needs data that exists in two different data bases and must use some method to obtain that data.

One way might be to copy the skill data into the record of each employee who has that skill. This does away with the need to interrelate the Employee and Skill data bases, but it creates redundant data. For example, several employees could have the same skill, and this data would be repeated over and over again.

Another approach might be to replace the Skill segment in the Employee data base with a segment that contains the direct physical address of the Skill segment in the Skill data base. Let's call this segment, Skillnum. Now when the information is needed, the programmer accesses the Skill data base using the pointer in the Skillnum segment. Under this arrangement, the Skillnum segment is called a *pointer* segment and the Skill segment is called a *target* segment. Of course the data in the two data bases and the relationship between the two must be kept up to date.

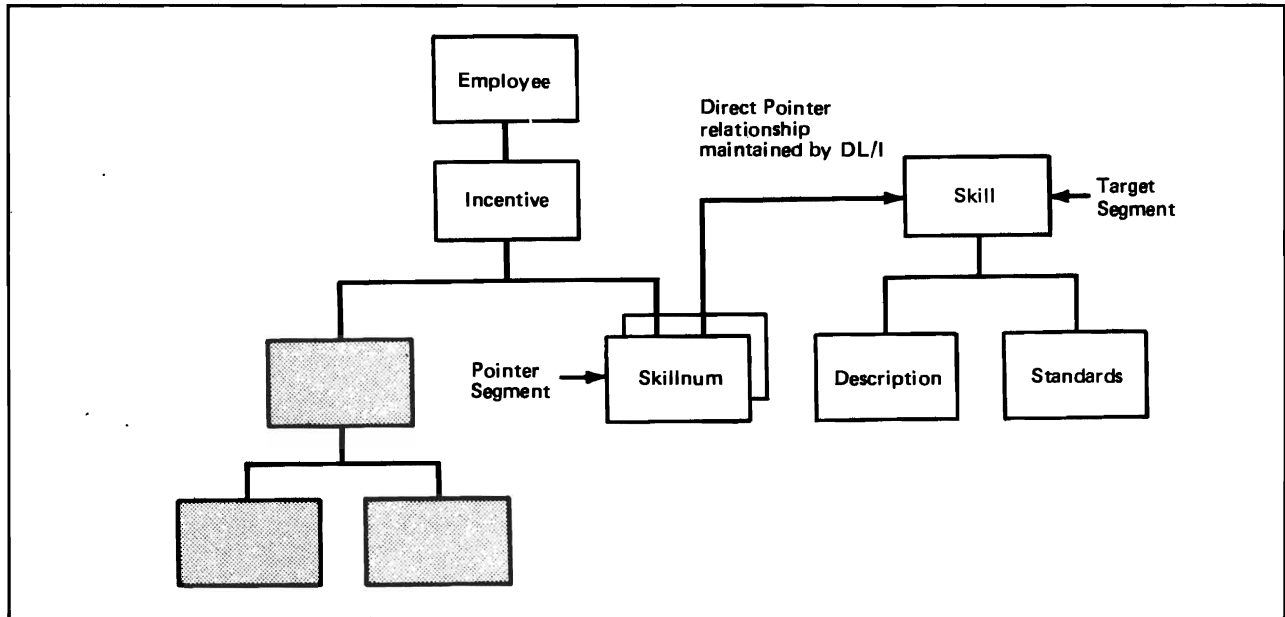


Figure 6.1.9 Relating Two Data Bases

With DL/I the programmer doesn't have to worry about maintaining this relationship. Once the logical connections between segments have been specified (in the DBD), DL/I maintains the relationship between the two data bases automatically. This capability is known as *logical relationships*.

The logical relationship between the two data bases is actually viewed by the programmer as one logical data structure as shown in Figure 6.1.10. When a programmer asks for a Skill segment for an employee, DL/I retrieves the correct segment from the Skill data base and the application programmer doesn't even have to be aware that this data base exists.

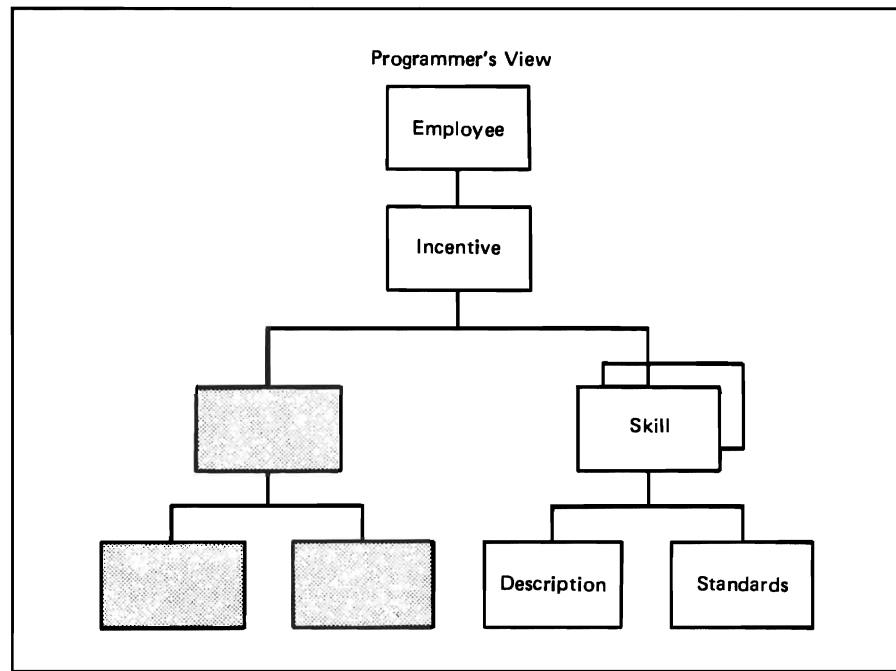


Figure 6.1.10 Logical Relationship Seen As One Logical Data Structure

This enhanced capability provides flexibility in application design, because the data can be viewed in a natural and logical way regardless of the physical structure to which it belongs.

Logging Facility

In any data processing environment some type of failures or errors inevitably occur that damage the data. Successful recovery and restart from these errors and failures is critical to the interrupted application.

Error detection and correction in a data base environment is complicated by the various interrelationships inherent in the data base. For example, if a data base participating in a logical relationship is damaged or destroyed, simply restoring a copy of the data base from the last backup copy does not replace the interrelationships between the data bases. Nor can the jobs executed since the last backup copy because the related data bases already reflect the relationships created by the original execution of those jobs. To recover in the traditional manner, all related data bases have to be restored from earlier backups. Unless the backups of all the related data bases are taken at the same time, it is impossible to recover from the failure in the traditional manner.

Among the facilities provided by DL/I to assist in this error recovery task is the *DL/I Logging Facility*. With this facility, DL/I records both "before" and "after" images of all segments changed by application programs. Application program requests may cause segment changes that are "invisible" to the application program. For example, if an application program deletes a segment, all the segment's children are also deleted, even if the application program is not sensitive to them. This same delete request may also cause pointers in related segments to be modified. All these changes to the data base are recorded on the log, even though they may be invisible to the appli-

cation program. The use of these log records is explained in the section on Data Base Recovery System Utilities.

Utility Support Programs

DL/I DOS/VS provides the following utility programs to support a data base environment. These programs execute in a batch, non-MPS partition to accomplish their specified function.

- PSB Generation
- DBD Generation
- Data Base Reorganization - used to unload, reorganize, and reload data bases
- Data Base Recovery System - See next section

Data Base Recovery System Utilities

The data base recovery system comprises several utility programs and is designed to provide a rapid, accurate, and easy-to-employ means of restoring the contents of a physical data base after destruction.

- Data Base Backout Utility

This utility removes changes made to data bases by application programs. After backout, the status of the data base is the same as if the job or transaction was never executed.

- Data Base Data Set Image Copy Utility

The data base data set image copy utility is used to produce a copy of the data base on tape or disk in a format suitable for use by the data base data set recovery utility.

- Data Base Change Accumulation Utility

The data base change accumulation utility sorts records from the data base log file and combines all records that update the same segment. The result is a sequential file that contains a condensed description of all changes to the data bases.

- Data Base Data Set Recovery Utility

This accomplishes the restoration of a data base using an image copy and the accumulated changes.

- Log Print Utility

This utility prints the contents of DL/I log files.

Summary

The data in a DL/I data base is defined to the application program as a series of segments arranged in a hierarchical structure. The data is actually stored in a physical structure defined to DL/I. This separation of the physical organization of data from the structure accessed by an application can allow the expansion of the data base without affecting existing application programs. This reduces the cost of making changes and reduces data redundancy when new applications are implemented.

- During execution, an application program may issue a DL/I call statement requesting a particular segment of the logical structure as defined in the PSB

associated with that program. The process which occurs to satisfy that request is depicted in Figure 6.1.11.

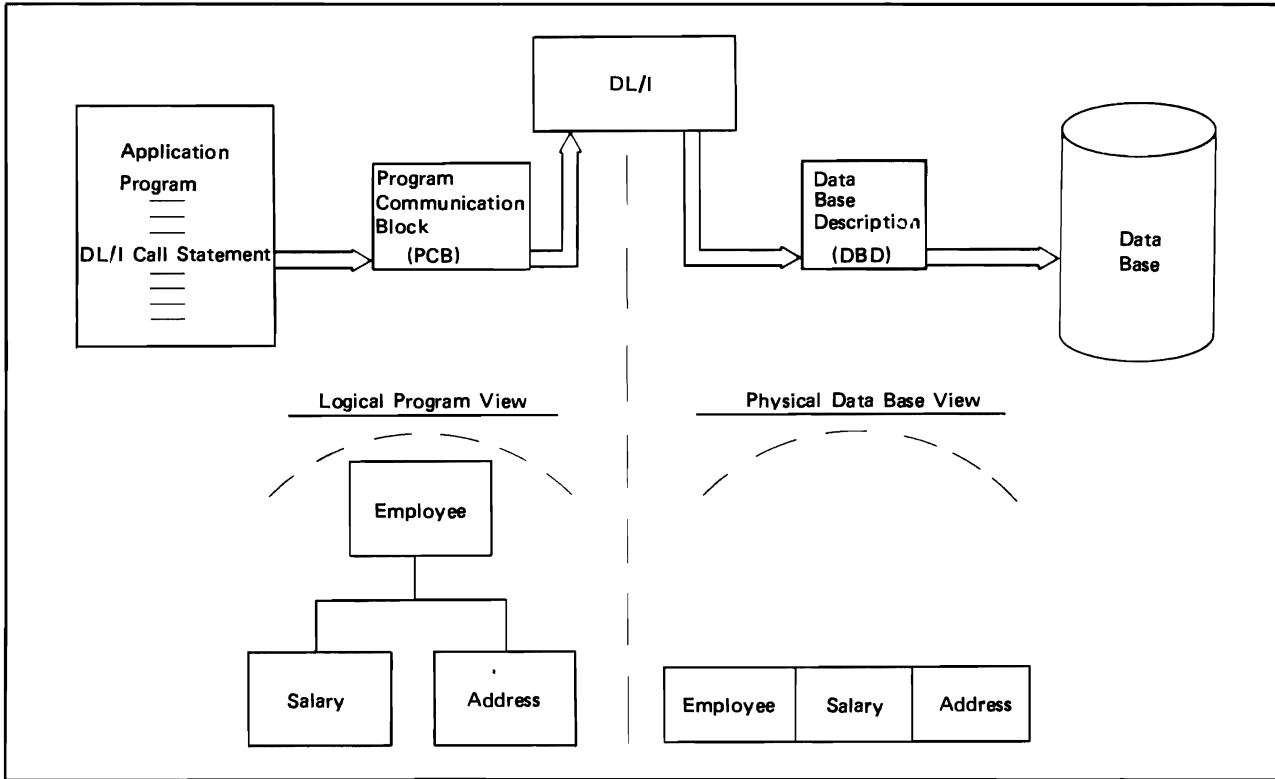


Figure 6.1.11 Accessing A Data Base From An Application Program

The ability to use PSB's to define sensitivity to different segments or fields of the logical hierarchical structure provides data confidentiality and security. In addition, this ability to limit the data accessible to that needed by a user program increases the productivity of the programmer.

When data existing in one data base is required by a completely separate application with its own data base, DL/I provides logical relationships. DL/I maintains a pointer between the data bases so that the logical access of data existing in another data base is handled as if the two were one data base. This reduces data redundancy between applications and eliminates the programming task of maintaining the pointers.

DL/I also includes facilities to log changes to the data and to use these logs to restore the validity of the data in the event of errors. This protects the physical integrity of the data.

Exercise 6.1

1. To provide device and data independence, DL/I references a _____ structure while the application program deals with a _____ structure.
2. These structures are defined to DL/I in two tables stored external to the user programs. They are the _____ and the _____.
3. The basic element of a DL/I heirarchical structure is the _____.
4. Match the appropriate description with each of the following terms:
 Parent _____ Twins _____ Child _____
 Dependent _____ Root _____
 Data Base Record _____
 - a) first segment in a DL/I hierarchical structure.
 - b) multiple occurrences of the same segment type.
 - c) all the related segments of a hierarchical structure.
 - d) all segments in a structure except the root.
 - e) segment type with a subordinate segment type.
 - f) subordinate segment type.
5. DL/I contains two storage organizations, _____ and _____.
6. Name the six DL/I access methods.
 - a) _____
 - b) _____
 - c) _____
 - d) _____
 - e) _____
 - f) _____
7. When DL/I maintains a link between separate data bases, they are said to be involved in a _____.
8. List the following items in the normal sequence of an application program requesting data from a data base:
 - a) DL/I
 - b) DOS/VSE file management
 - c) PSB
 - d) DL/I call statement
 - e) DBD

Solution 6.1

1. a) logical, b) physical
2. a) DBD, b) PSB
3. segment
4. Parent - e, Twins - b, Child - f,
Dependent - d, Root - a,
Data Base Record - c
5. a) Hierarchical Direct (HD)
b) hierarchical sequential (HS)
6. a) HSAM, b) SHSAM, c) HISAM, d) SHISAM, e) HDAM, f) HIDAM
7. logical relationship
8. d - c - a - e - b

Topic 2. CICS/DOS/VS

As information requirements of a business expand, the data processing time required to produce the information becomes critical. This requires the implementation of methods which reduce the processing time. One such method is an online or data communication system which provides direct communication between the user and the computer processing the application data. With an online approach, the required portion of information is available upon request. This is contrasted to waiting for scheduled batch processing and manually extracting the required portion of information from a large, inclusive report.

As the information becomes more available, the currency of the data is an important concern. This leads to updating of the data as events occur, which in turn causes concerns about the validity of the data. The number of users and applications utilizing online processing continues to grow. Eventually, the major concern of data processing is the management and control of the online system. The solution is a data communications manager.

The implementation of a data communications manager separates the online system into two components, the *control process* and the *application process*. The application process is the conversion of available data into the required information. It is a data delivery process involving the formats for information display, data manipulation, and the update of data bases. The control process or component includes the movement of data between the data base and the application program, and between the user's terminal and the application program. The control process also includes the management of resources such as storage and time. The application process varies with each user and with different businesses; however, the control process is very similar in all environments. Because of the common nature of the control process, it is possible to develop a data communications manager which can be tailored or customized to fit each different environment.

IBM offers the *Customer Information Control System DOS/VS (CICS/DOS/VS)* to relieve each data processing installation from the task of developing their own data communications manager. CICS/DOS/VS is a program product which runs in a DOS/VSE partition to provide an interface between the operating system and the application program. It provides commands to be used in an application program to request services such as reading and writing of data bases, reading and writing to the user terminal, and requesting system resources. CICS/DOS/VS provides these functions and saves the time, cost, and risk of creating them. It simplifies and reduces the effort to implement an online application.

This topic will explain the basic functions of CICS/DOS/VS, the program interface, the general flow of operation, and several additional optional functions included in the basic program product.

Basic Functions

With CICS/DOS/VS, data processing is brought into the user department where the input data is available and the output data is needed. In order for this to happen, an input/output (I/O) device(s) must exist in the user department. This online I/O device, called a *terminal*, can be any one of a wide range of devices but is generally a video display with a keyboard such as the IBM 3278 or 3279. The person who utilizes this device is called the *terminal operator*. This terminal operator is not necessarily aware of CICS/DOS/VS, but interacts directly with the application program designed for the function being performed. This interaction or application processing session is called a *transaction*. The transaction is initiated by the terminal operator inputting a preset *transaction identification code (transid)*. This transid causes CICS to invoke the appropriate application program which then interacts with the terminal operator. The specific execution of a program to interact with a particular terminal operator is called a *CICS task*. The distinction between a transaction and a task is that a

- *Transaction* is an *application* program (or programs) that can be used by a terminal operator. A given transaction can be used concurrently by *one or more* operators.
- *Task* is an *execution* of a transaction for a particular operator. A given task can relate to *only one* operator.

A CICS/DOS/VS environment consists of multiple terminal operators entering transactions and creating multiple tasks. Each task is operating concurrently and is requesting services and sharing the resources of the system (storage, programs, files). CICS/DOS/VS employs a modular approach to the control and servicing of this environment. It is divided into several management functions or control programs, each of which has a specific purpose. The primary modules of CICS/DOS/VS and their functions are presented in the following paragraphs.

CICS/DOS/VS
Management Functions

provides for the communication between terminals and user-written application programs. It checks terminals to see if they have data to transmit and checks to see if they are available and ready to receive output data. It also keeps track of which task is associated with which terminal and handles any terminal I/O errors which may occur.

Task management provides dynamic control of the multiple tasks which are executing concurrently. It uses a multi-tasking scheme similar to the one used by DOS/VSE to keep track of the status of all executing tasks. A transaction is usually not processed to completion without interruption. For instance, if it needs data from a file, it must wait for the file access request to be processed by CICS and DOS/VSE. While that task is waiting for its data, task management will allow another task to process. It also will check the validity of a transaction code entered by the terminal operator and keep track of the use of application programs in a valid transaction.

Program Management tracks which application programs are in use and their location in main storage. If a requested application program is not located in main storage, program management will access the core image library to load it into storage. It also handles any errors caused by the application program

and prevents them from causing the termination of CICS/DOS/VS and other transactions being processed.

Storage management controls the main storage allocated to CICS/DOS/VS taking advantage of virtual storage concepts. It processes requests from application programs for additional storage to be used for data from files or unique program requirements. It also handles requests from other management programs. For example, when program management needs storage for a program to be loaded from the core image library. Storage management keeps track of the storage in use and contains procedures for handling errors that may occur during processing.

File Management accepts requests from application programs for data which is stored in a VSE/Virtual Storage Access Method (VSE/VSAM) data set or a Basic Direct Access Method (BDAM) data set. It issues the proper request to VSE/VSAM or DOS/VSE file management routines and formats the returned data for the application program. It also provides the same interface for the updating of these data sets. File management provides file protection functions to prevent errors due to multiple concurrent online tasks accessing the same data set.

The use of these management functions or control programs is depicted in Figure 6.2.1.

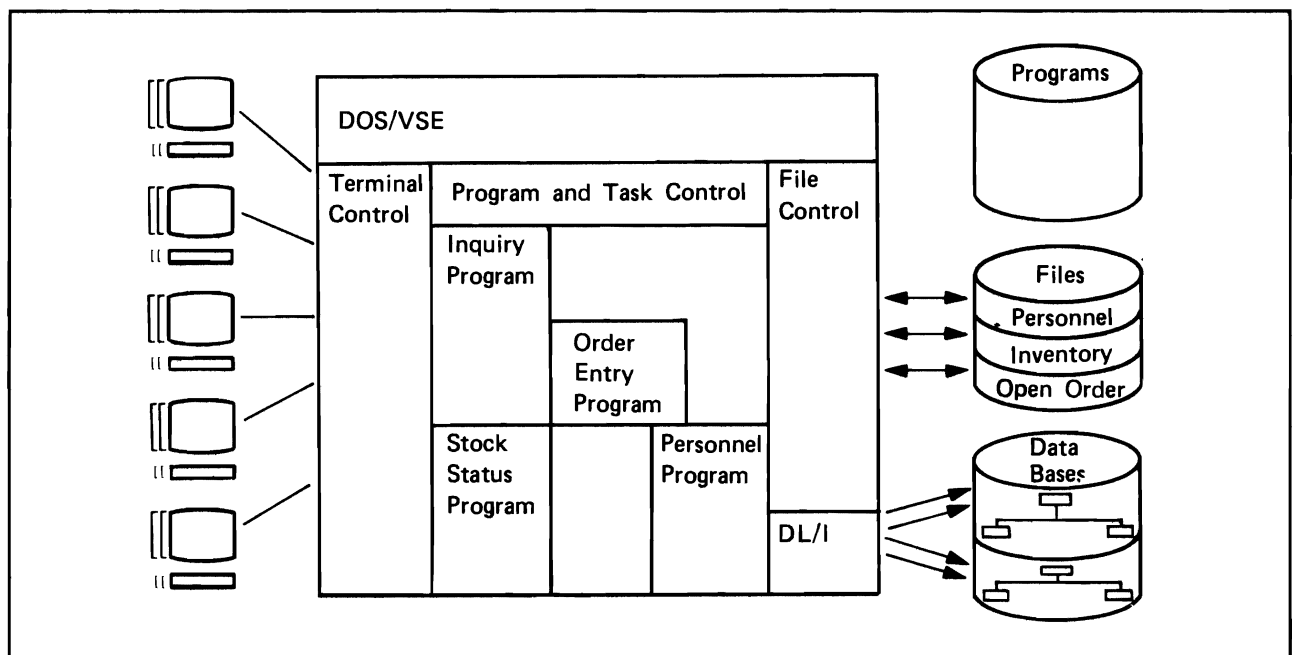


Figure 6.2.1 CICS/DOS/VS Management Functions

Each data processing installation differs in such areas as the terminal devices used, names of application programs, the codes used to identify transactions, and the type of data set access methods employed. To allow these variations to exist and still utilize the benefits of CICS/DOS/VS, there must be a method of defining or specifying the characteristics of each environment. CICS/DOS/VS uses a technique in which the installation unique items are specified in a prescribed format called a *table*. The data specified in the table

is then stored in the core image library for reference during execution of CICS.

There is a *file control table* containing the record formats, access methods, and other characteristics of the data sets processable by the file management function. It also contains the names of the data bases to be processed by DL/I. Another, the *terminal control table*, identifies the terminal devices used with CICS/DOS/VS. The *processing program table* includes the name and storage location of each program to be used by transactions. This table is used by the program management function.

The last of the primary tables prepared by the user is called the *program control table*. It is associated with the task management function and defines each transaction that may be processed. It contains a transaction code (transid), the name of the program which processes the transaction, a security code, and a value representing the priority or relative significance of the transaction. It is accessed by task management to determine if a transaction identification code entered by a terminal operator is valid. If the transid is not in the table then the transaction is invalid.

There are other tables used by CICS/DOS/VS to allow the user to specify desired optional functions and to specify variable information required by CICS/DOS/VS. All of these tables and the management functions which use them provide the flexibility and adaptability for growth and change of the data communications environment without impacting the application programs.

Program Interface

Online application programs used to process transactions can be written in COBOL, RPGII, PL/I, or Assembler language. They are similar in content to batch programs with the exception of the requests for data input and output to terminal operators and files. These requests are submitted to a CICS/DOS/VS function known as the *high level programming interface (HLPI)*. The programmer communicates the requests for service through the HLPI by coding special statements called *commands*. The CICS commands are structured in a format which is similar to the programming language being used. They are simple to use and allow the programmer to utilize the services provided by CICS/DOS/VS without knowing its internal procedures.

Online programs are smaller than typical batch programs because many of the necessary functions are provided by CICS/DOS/VS through its commands. This allows quicker implementation of application processes and facilitates the use of online programming to provide current readily available information.

Included in CICS/DOS/VS is a feature called *Basic Mapping Support (BMS)*. The use of BMS accelerates application program development. Through it, the formats to be used in displaying information to the terminal operator may be specified in advance and stored in the core image library. Then the programmer simply identifies the desired format and merges it with the application data to be displayed. With this procedure, there is a measure of device independence as the programmer does not need to be cognizant of the type of terminal device being used. This also reduces programming effort by freeing the programmer from rewriting a common format in every program using it.

**CICS/DOS/VS
Processing Flow**

To better understand the interaction of an application program, a terminal operator, and the CICS/DOS/VS management functions, Figures 6.2.2 through 6.2.6 describe the general flow of a transaction. The example is based on a simple inquiry application in which the terminal operator enters a transaction identification code and the account number of a particular customer. The application is designed to access a data set to obtain the current balance in the specified account and then to display that information back to the terminal operator. This example assumes a typical error free flow of the transaction used to process the application.

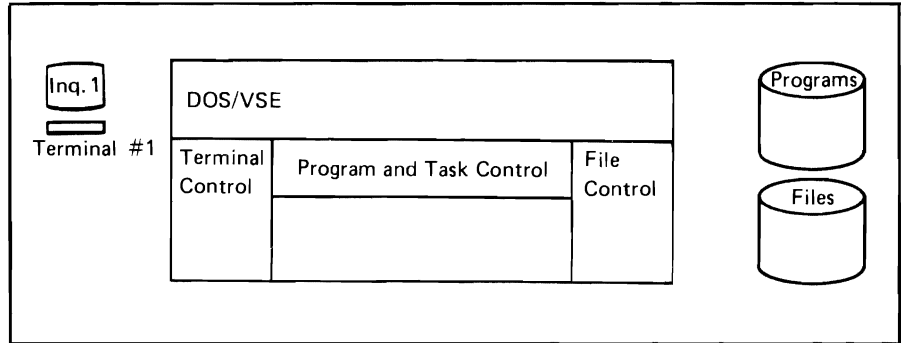


Figure 6.2.2 Terminal Management

While checking the terminals, terminal management recognizes the transaction request entered by the terminal operator. It accepts the data from the terminal and transfers it to task management.

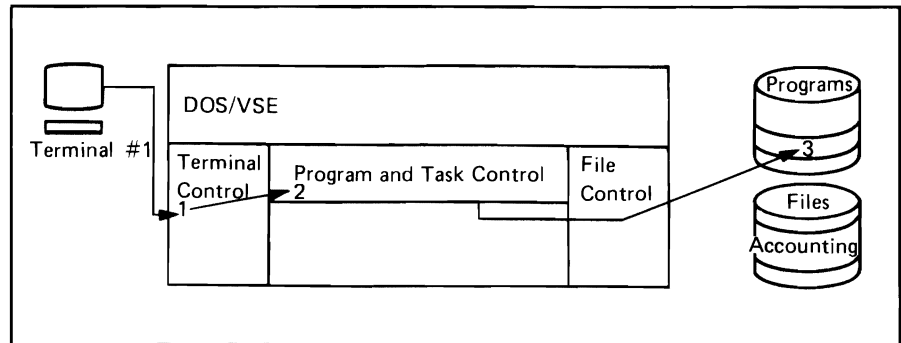


Figure 6.2.3 Task Management

Task management validates the transaction by examining the program control table. If valid, it extracts the name of the program to be executed and requests program management to locate the program.

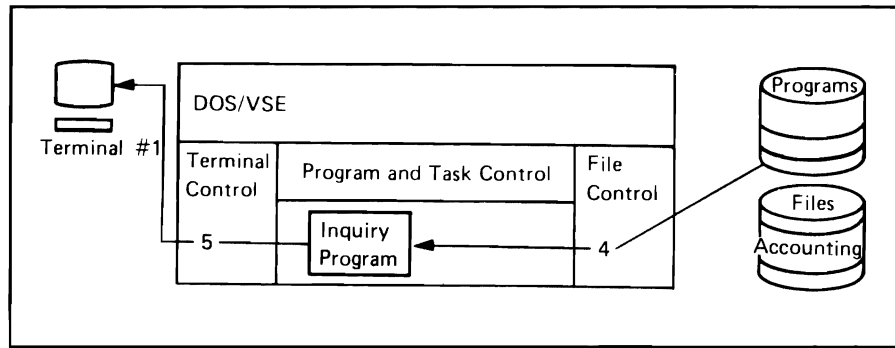


Figure 6.2.4 Program Management

Program management checks the processing program table to determine the location of the application program. If not present in storage, it will obtain the program from the core image library. Control is then given to the application program which requests the terminal operator to enter an account number.

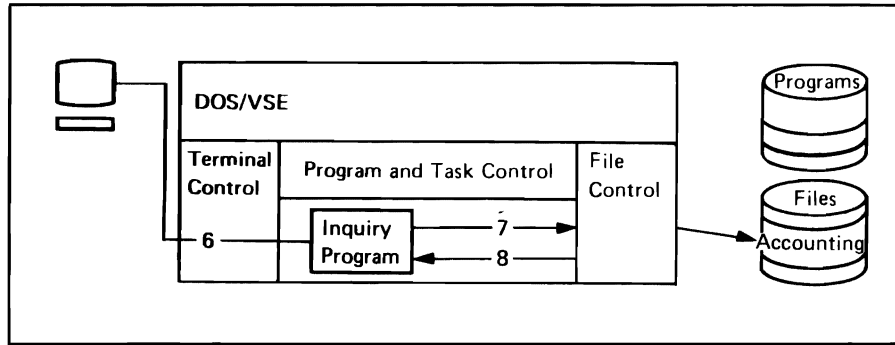


Figure 6.2.5 File Management

Terminal management sends the application program request to the terminal, accepts the data entered by the terminal operator, and provides the data to the application program. The program then issues a command to request the accounting data from file management. File management accesses the accounting file described in the file control table, utilizing the DOS/VSE services, and returns the data to the application program.

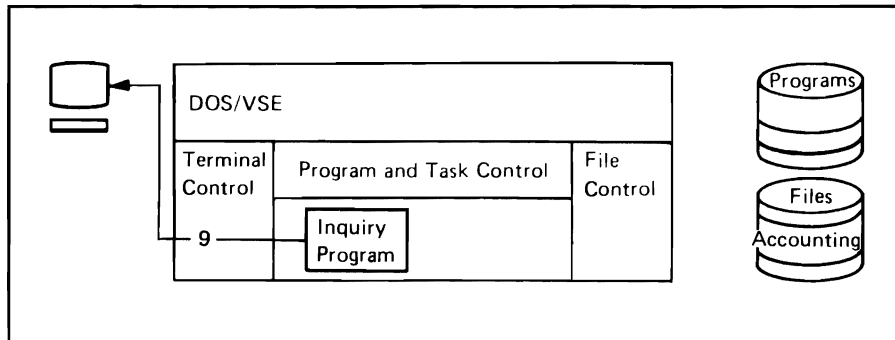


Figure 6.2.6 Terminal Management

The application program merges the data with the format previously specified through basic mapping support and sends it to the terminal operator. Terminal management performs the output to the terminal and continues to check the terminals for other transactions to be started.

Additional Functions

Up to this point we have described the basic functions used for application processing of online transactions. This section presents other commonly used functions and facilities that are available to users of CICS/DOS/VS.

Sign-on

The CICS/DOS/VS sign-on program and table optionally provide security by restricting the personnel allowed to operate terminals. All operators may be required to sign on by entering their name and a pre-arranged password before being allowed to operate terminals. Through security keys, only selected operators may be allowed to use restricted programs and have access to confidential files.

Master Terminal

The master terminal function allows control over the system during CICS/VS operation. Any terminal may be designated temporarily as a master terminal, and an individual may be authorized to perform master terminal functions through a security key. Master terminal functions include the capability to activate or de-activate facilities such as programs, files, transactions and terminals. The master terminal operator can use this capability to control these facilities as required. For example, if it is determined that one of the programs has an error, it can be de-activated by the master terminal operator without affecting other operations. Also, the master terminal operator can request information on the status of each table entry and alter various values which control the operation of the CICS/DOS/VS system.

System Statistics

Various modules accumulate statistics that are useful in analyzing and improving performance. The statistics include the number of transactions initiated, the number of times a program was used, and the number of file operations. CICS/DOS/VS also provides other statistics such as the number of input and output messages per terminal which can help management to make the most effective use of the online system and to plan for future growth.

Execution Diagnostic Facility

To decrease the amount of time spent in checking for errors in a program, CICS/VS provides the Execution Diagnostic Facility (EDF). EDF is designed to allow the application program that was written using HLPI to be tested in an online, interactive manner from a terminal.

Using EDF, CICS/VS commands will be displayed at a terminal in source format before execution, thereby allowing the user to examine for errors and, if necessary, make corrections. After execution, the command and the results of the command will be displayed so that the user can see if the command yielded a correct response. Also, the flow of commands in the application program can be traced and displayed. The use of EDF by the application programmer to find and resolve the errors in an application program, should lead to shorter application development time.

*Program and Data Error
Recovery*

During the execution of CICS/DOS/VS, errors could develop within the application program which cause a task failure. If the failing task was in the process of updating a data set or data base, the data is then in error. In the event of a task failure a function called *Dynamic Transaction Backout (DTB)* can be used to cancel (or backout) the work done. This restores the affected files to the state they were in before execution of the transaction without affecting other transactions.

If a severe error occurs in an application program or a CICS/DOS/VS management function the CICS/DOS/VS system may be abruptly terminated. After a system failure of this type, a function called *Emergency Restart* can be invoked to restore all data files being processed by transactions at the time of the failure to their condition prior to the start of those transactions.

In order for the emergency restart and DTB procedures to restore the affected files, information reflecting the operational status of files and transactions at the time of failure must be available. This information is stored in a *CICS/VS journal*, a special-purpose sequential data set. The journal is maintained by the *journal management function* which writes records reflecting the task activity against data sets during the execution of these tasks.

The use of these functions provides physical data integrity for CICS/DOS/VS data sets.

Summary

An online system normally includes multiple terminals. For effective use of the online system, it must include a facility to manage those terminals. With multiple terminals and multiple applications there will exist multiple tasks in the online system. There needs to be a facility to manage the tasks and their programs so they can execute concurrently. These tasks normally access data which exists in multiple data bases or data sets accessible by the online system. A facility to manage these file accesses is required to provide for data integrity and recoverability. An effective, productive, responsive online system should include all of these components.

It is not easy for an installation to develop all these functions for their online system. The effort to implement these functions can be minimized by the use of CICS/DOS/VS. It includes these and other management functions to ease the process of developing and using online applications.

The factors which are unique to an installation are specified in various tables. These tables are then stored separately from the application program. This enables changes in terminal configuration, data bases, data sets, and application transactions to be made easily without altering the application programs. This allows more time for programmers to develop new applications instead of maintaining existing ones.

In general, CICS/DOS/VS is a general purpose DB/DC control system for online applications. It provides most of the standard functions required to communicate with the terminals in the online system and the control for multiple terminal users processing transactions containing multiple programs. It is the primary tool for providing the timely, usable information required to operate the business.

Exercise 6.2

1. The I/O device used in a data communication system is called a _____ and the person using this device is a _____.
2. A CICS/VS interaction or application processing session is called a _____ and is initiated by entering a _____.
3. The specific execution of a program to process an interaction is called a CICS/VS _____.
4. For efficient and easily maintainable operation, CICS/DOS/VS is divided into several modules called ____ file _____.
5. Two examples of the CICS/VS tables used to store installation unique items are _____ and _____.
6. List the following functions in the sequence they are used during normal CICS/DOS/VS processing.
 - a) Task management
 - b) Terminal management
 - c) Program management
 - d) File management

Solution 6.2

1. a) terminal, b) terminal operator
2. a) transaction, b) transaction identification (TRANSID)
3. task
4. management functions (or control programs)
5. a) File control, b) Terminal control, c) Processing Program or d) Program control
6. b - a - c - d

Topic 3. Display Management System/CICS/VS

The availability of data is improved in a data base environment using DL/I DOS/VS. Techniques for achieving data currency and accessibility are provided in a data communications environment with CICS/DOS/VS. An installation with a DB/DC environment has the ability to make data available and directly accessible by the users. This capability results in an increase in the requests for data to be maintained and for new online applications to access that data. This can result in a backlog of requests for data processing application programs due to the lack of available programming manpower and the time required for program development.

The reduction of this backlog can be accomplished by denying the requests, increasing the manpower, or shortening the development time. If there is a justifiable need for the information, the requests should not be refused. Increasing the manpower, if it is available, increases the expense of labor which could raise the cost of an application beyond justification. The best approach might be the implementation of data processing techniques and tools which shorten the program development time. In other words, to provide for faster data delivery to the user.

One of the tools available to accomplish faster data delivery is an IBM program product called the *Display Management System/CICS/VS (DMS/CICS/VS)*, hereafter referred to as DMS. Basically it is a group of pre-defined and pre-programmed functions to be invoked as needed by the programmer. This technique is effective because most online applications require the same logical functions such as reading data from a file, performing calculations, reformatting the data, and displaying the information to the user. They differ in which file is used, the format of the data, the sequence of data manipulation and how the data is displayed. DMS provides these common functions and techniques for easy specification of data and screen formats. The programmer is freed from programming the details of functions, thereby reducing the effort to implement an online program.

Use of these predefined functions and techniques requires less programming knowledge and skill. Many application programs can be implemented by user department personnel with a minimum of data processing skills. In a sense, this can increase the manpower for programming without increasing the labor cost. Also, the person implementing the programs is the one who understands the data and the application requirements so it eliminates the time required for a programmer to learn the application. These are a few of the advantages of using DMS to develop online programs.

Organization and Basic Functions

DMS/CICS/VS has two operating environments, online and offline. In the offline operations, the user prepares input for and executes various utility programs in a batch partition of DOS/VSE. The online portions of DMS operate under the control of CICS/DOS/VS and are accessed through a 3270 system video terminal.

DMS/CICS/VS has two separate components, the *base product* and the *application generation feature*. The base product controls the online display and file operations specified by the user application. It includes the programs which are executed when processing the application. The application generation feature is used for the defining of the application and its data. This

definition is accomplished through an online facility called the *interactive application generation facility* or through offline DMS batch programs. The offline approach to application generation uses a series of "fill in the blanks" forms to provide the input to the batch DMS programs.

The general procedure is to first define the data files to DMS and then to define the display formats and file-to-display or display-to-file operations. The next step is to define the data editing calculations to be performed. These definitions are stored by DMS to be used by the online processor when this application is activated by a terminal user. The process of defining the application data and procedures to be performed is called *application generation*. The definitions are used by a terminal user invoking a *DMS transaction*.

Application Generation

The application is defined in three stages:

Data Definition allows the identification of the file organization, the data format and field characteristics such as length and field name.

Map or Panel Definition allows the user to describe the display formats to be used at a terminal when processing the DMS transaction. Each separate format is known as a *DMS panel*.

Process Definition is used to specify which DMS functions, arithmetic operations, and content editing is to be performed on the data. It also allows specification of the panel to panel processing sequence required to complete the application.

These definitions can be performed from an online terminal through the interactive application generation facility. It will guide (or prompt) the terminal user through all the stages. As an option, the user may perform application generation by manually entering the data onto special forms. The data entered by either method is processed by the batch DMS utility programs. The generated application is the same regardless of the definition method used.

To better understand this process of application generation, Figures 6.3.1 through 6.3.3 show how the manual forms are used to define a file and specify its use on a panel. Figure 6.3.1 shows the lines of the *File Description Form* which are used to specify the characteristics of the data format and the fields of the records in this file.

DMS/VS DMS/CICS/VS File Description Form							
Application Project							
Completed by							
FILE HEADER CARD							
File Name	Type	Record Length	Rec. Blk.	Key Field Name			
1 2 3 4 5 6 7 8	9 10	11 12 13 14 15	16 17 18	19 20 21 22 23	24 25 26 27 28 29		
FIELD DESCRIPTION CARDS							
Field Name	Length	Mode	Field Name	Length	Mode	Field Name	
1 2 3 4 5 6 7 8	9 10	11	12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29	30 31 32 33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48 49 50	51 52 53 54 55 56 57 58 59 60	61 62 63 64 65 66 67 68 69 70

Figure 6.3.1 File Description

When the application is processed, some of the data fields defined during File Description may be displayed on the user's terminal. Which fields to be displayed and their location on the screen are defined on the *Panel Description Form* as shown in Figure 6.3.2.

Data Definition Cards												
	DATA FIELD											
F	Data Field Name	Row	Column	Length	At	FI	FO	V	A	R		

Figure 6.3.2 Data Definition Portion of the Panel Description Form

The data field name entered here is the same as in Figure 6.3.1 and the location where it is to be displayed is specified by identifying the row and column of the 3270 display screen. The other information (AT, FI, so forth) is used to specify special techniques to be used when displaying the data such as highlighting. The Panel Description form is also used to specify the operations which are to occur in this DMS transaction. This is specified by entering the appropriate codes for one or more pre-defined DMS *supervisor functions*. Three of the supervisor functions for files are shown in Figure 6.3.3.

DMS/VS SUPERVISOR FUNCTIONS			
Function Name	SF	Modifier	
		Code	Meaning
Inquiry/Inquiry for Update	G	6	Normal inquiry
		1	Browse (Get Next)
		2	Seq. on NRF
Record Add	H	6	Add
		1	Hold for add
		2	Inq. for update
Update	I	6	Update
		1	Delete
		2	Hold for update

Figure 6.3.3 Supervisor Functions

When the code for a supervisor function is encountered during transaction processing, DMS will perform the indicated action. Thus the DMS programmer accomplishes with one line of codes what would require many lines of instructions in a non-DMS transaction program. Similar codes are used on the *Calculation Specification Form* to cause arithmetic or field editing operations to be performed on the data prior to displaying it or writing it back to a file.

All of the information specified during application generation is stored as tables in the DOS/VSE core image library for retrieval during transaction processing.

DMS Transactions

A DMS transaction is invoked in the same way as other transactions in the CICS/DOS/VS system. The transaction identification entered by the terminal user causes the execution of the DMS online processor which then displays a panel requesting the terminal operator to select the application to be performed. This initiates the DMS transaction and the operator is ready to proceed with application processing.

A DMS application is defined in a series of panels designed in a hierarchical manner as shown in Figure 6.3.4.

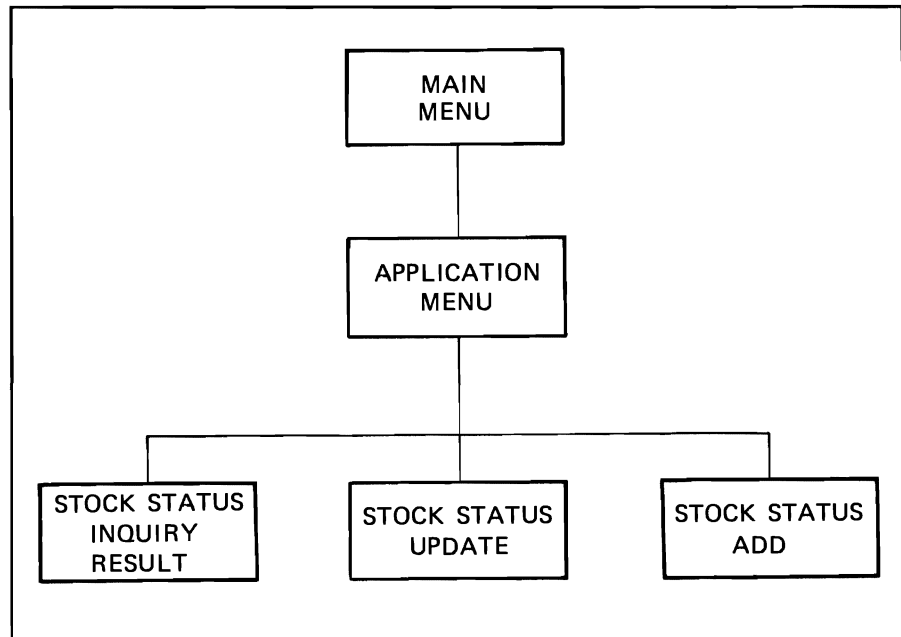


Figure 6.3.4 Panel Hierarchy For A DMS Application

The first or root panel in each application hierarchy is known as the *main menu*. This is the panel displayed by DMS for transaction initiation. The next panel would be displayed as the result of a selection by the terminal user from the main menu panel. Typically, this panel is an *application menu* which lists the possible processing that may be performed in this application. Upon selection of an application process, another panel containing the requested data or allowing for the entry of data might be displayed. Each of these lower level panels should contain instructions on their use and on the way to select another panel to be displayed. An example of this DMS application transaction flow is shown in Figures 6.3.5 through 6.3.7.

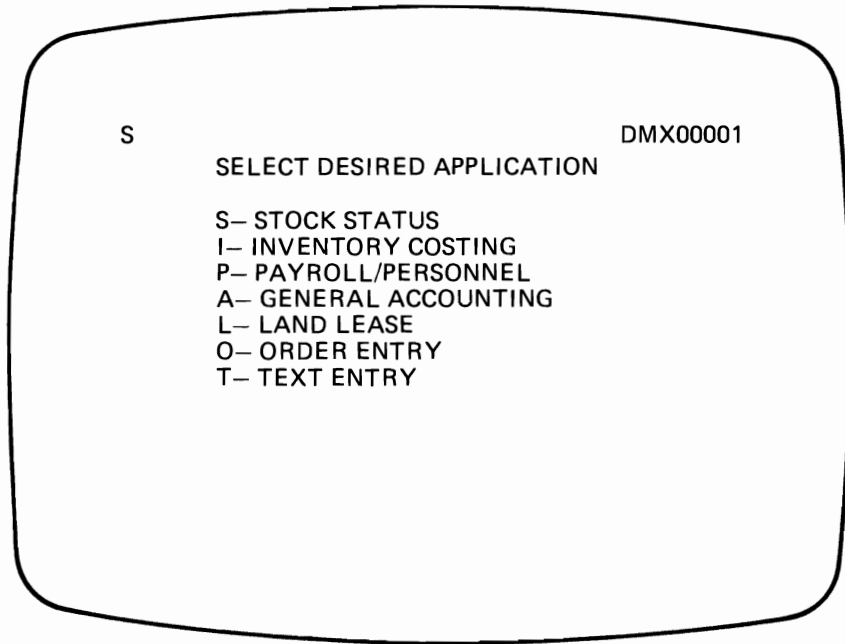


Figure 6.3.5 Main Menu Panel

The general procedure is to request the desired operation by entering the appropriate character in the upper lefthand corner of the panel displayed. This character, called a *function identifier*, is associated with the appropriate supervisor function. In Figure 6.3.5 we have selected the Inventory Stock Status application by entering an S.

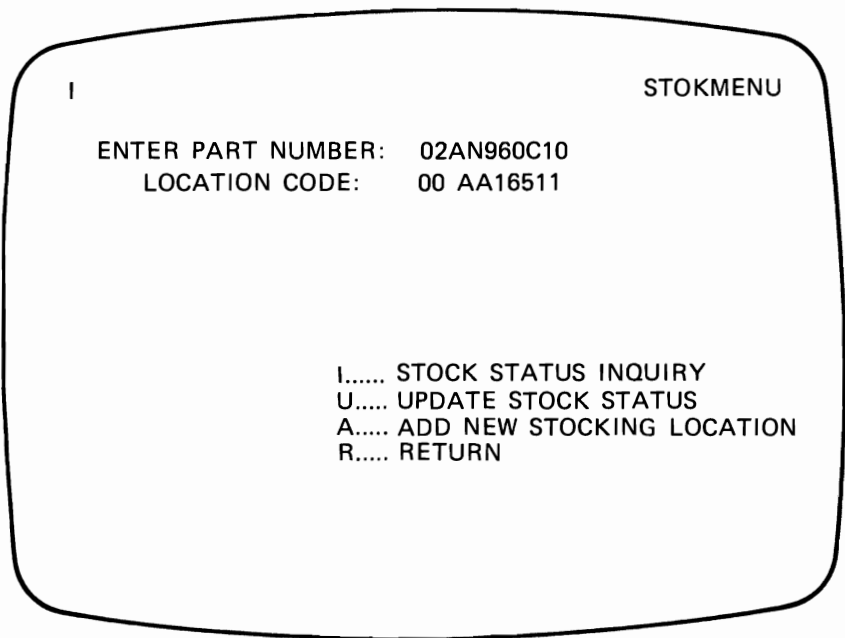


Figure 6.3.6 Panel With Data Keyed For Inquiry

The selection of the Stock Status application caused the display of the menu panel previously generated for that application. This panel as shown in Figure 6.3.6 requests the terminal user to select the operation to be performed and to enter the data which uniquely identifies the data record to be processed. In this example we have asked for an inquiry on part number 02AN960C10 in location 00 AA16511.

N	STOCK STATUS REPORT		STOKSTAT
PART NUMBER:	02AN960C10	DESCRIPTION:	WASHER
STOCK KEY:	00 AA16511	REQUIREMENTS:	131
UNIT PRICE:		ON HAND:	126
U/MEASURE:	EACH	ON ORDER:	20
L/C DATE:		DISB. PLANNED:	104
		DISB. UNPLANNED:	
AREA:		DEPT:	AA
PROJ:	165	DIV:	11
		N...	NEXT ITEM
		R...	RETURN

Figure 6.3.7 Result Of Inquiry On Inventory Data Base

The data from the specified record is obtained by DMS using CICS/DOS/VS facilities. The requested data is then displayed at the terminal as specified on the Panel Description. The inquiry result panel shown in Figure 6.3.7 offers the user the option of returning to the stock status panel of Figure 6.3.6 or of entering another part number and location for display. If the next operation to be performed is a record update on this displayed record, then it would be necessary to return to the previous panel.

The processing of this application request from main menu panel through the inquiry result panel required the person who generated the application to enter thirty-four lines of data to the DMS batch utility program. A knowledge of programming language was not required to perform this function. The same function prepared by COBOL programming would require more lines of instructions, programming skills, and development time. The DMS "programmer" is simply selecting pre-programmed groups of instructions by using codes to specify the operations to be performed.

Additional Functions

There are two significant capabilities of DMS/CICS/VS which extend its usability.

User Exits At times it is necessary to perform functions not provided in the DMS base product component. For this purpose, DMS allows the execution of user programming to perform those functions. Within the DMS online processor there are designed *user exits*. These are locations where user written programs may be inserted. The specification of the user program name to be included is entered on the panel description form. This again allows the application generator to invoke a function without using programming skills provided that the user exit program was pre-written by a programmer.

Application Security A DMS transaction is controlled by the CICS/DOS/VS security techniques as is any other online transaction. In addition, DMS employs a security feature for limiting access to applications within the DMS transaction. When initiating a DMS transaction, the terminal operator enters a two character security code. This code is matched against the one assigned to the panel requested by the operator. Access to the panel is allowed if the codes match. The applications or functions which are to be restricted can be specified on panels with security codes so that users cannot perform those functions without knowing the specific security code for that panel.

Summary The data required to operate the business is made available with DL/I DOS/VS and made readily accessible by CICS/DOS/VS. DMS/CICS/VS is the data delivery component designed to reduce the effort and time required to develop the programming to access that available data.

The fill-in-the-blanks technique used in either batch or interactive application generation provides a fast and efficient method of defining, implementing and maintaining online applications. The capabilities of DMS/CICS/VS make it possible to implement many online applications with little or no conventional user programming. The ability to perform simple calculations and data edits simplifies application development by significantly reducing the need for conventional programming skills. These features result in higher programming productivity and expanded use of the online system.

Exercise 6.3

1. DMS/CICS/VS has two separate components, the _____ and the _____.
2. (True or False) DMS includes CICS/DOS/VS programs which process all DMS transactions.
3. The process of defining the work to be accomplished by DMS is called _____.
It involves three stages of definition: _____, _____ and _____.
4. A DMS application panel hierarchy begins with a _____.
5. When presented with a DMS panel, the desired operation is accomplished by entering a _____ in the upper left corner of the panel.
6. Either _____ or the _____ can be used to provide the input for the offline DMS/DOS/VS utilities used to generate an application definition.

Solution 6.3

1. a) base product, b) application generation feature
2. True
3. a) application generation,
b) data definition, panel definition,
process definition
4. Main menu panel
5. function identifier
6. a) offline batch forms, b) interactive application generation facility

Topic 4. DB/DC Data Dictionary

Data is valued as an important resource in many organizations. The benefit of storing it in a flexible, easily accessible manner is understood and implemented as a DL/I data base. The importance of employing security, integrity, and recoverability techniques to protect the data is recognized and accomplished through the use of appropriate DL/I functions. The need for fast data access, data currency, and productive delivery of information is satisfied by establishing an online environment with CICS/DOS/VS, DMS/CICS/VS, and other data delivery products.

These concerns and characteristics of data collection, storage, and usage are important, but so is the proper management of the environment in which they exist. As data becomes centrally maintained and readily accessible in variable formats, its use by data processing and by multiple user departments increases. The same piece of data might be used in different ways by each of several users and given a different name by each user. One item of data can be processed by many different application programs and, likewise, one application program processes many items of data. In such an environment, it becomes increasingly important to pose and answer the following questions concerning the management and control of the data and its usage.

Is it easy to determine who uses any given piece of data in any application? Who should be advised if any changes are made in it? Who has maintenance responsibility for it? Who is to be contacted if it is erroneous? Or, in the event a particular program is known to have logic errors, what data bases might have been affected? Which other programs were affected? Who should be notified? What about the cross-relationships between programs and data? It may be fairly easy to identify all the data any given program uses, but, for a given piece of data, is it easy to identify all the programs (both in development and production) that use it?

There's another cross-relationship that must be controlled between data element name and content. How often do different names exist for what is really the same data? And how often is the same name given to data that is defined and used differently? Suppose it is necessary to identify all data now in the system that has something to do with tax credit. Can it be done easily?

The inability to answer these questions and others like them may result from insufficient attention to the need for management of data as a resource. It might also indicate data duplications, redundancies, and unclear data definitions, all of which serve to slow application development and increase cost.

If these questions are not easily answerable, there exists a need for a data dictionary. A data dictionary is a repository of all information relating to data as an entity and is a single consistent source of information for an entire organization. To allow productive collection, storage, and access of this information about data IBM offers a program product known as the *DB/DC Data Dictionary*. It is a tool to be used by those who are responsible for managing data.

In this topic, we will look at the general organization and function of the IBM DB/DC Data Dictionary product (hereafter called the Dictionary), its techniques for storage of data about data, the types of information retrieval provided by the product, and some optional functions for special situations.

Organization and General Functions

Like any dictionary, a data dictionary contains *definitions*. In this case, the definitions give the names and characteristics of the data items in an organization, and of the application system components that use the data. The Dictionary is implemented with DL/I as the data base manager to take advantage of the benefits of a data base environment. Like all DL/I applications, it consists of two major parts, *data* in data bases and *programs* to operate on the data. The programs are the means by which users enter and modify definitions and obtain reports on the contents of the data bases.

The Dictionary is designed to keep track of the major resources of an organization. These resources are called *subjects*. The subjects defined in the Dictionary are grouped into one of ten pre-established dictionary *subject categories*. Three subject categories correspond to the basic units of data in a DL/I system, but can be used for non-DL/I applications as well.

- Data base (or data sets)
- Segment (or record)
- Field (or element)

The next five correspond to the components of the application system which uses the data.

- System (application)
- Job
- Program
- Module (within a program)
- Transaction

The last two subject categories represent the sets of statements in DL/I systems that define and control the application program's use of data.

- PSB
- PCB

The Dictionary also maintains information on the *relationships* between subjects and the *attributes* which characterize both subjects and relationships. A *relationship* represents the interaction between subjects. For example, one relationship between a program and a data base would be the indication that the data base is accessed by that program. *Attributes* describe the properties of a subject or relationship. Examples of attributes are size, frequency of relationship, or the date a relationship was established.

As an example of the type of information stored in the Dictionary data base the following items are kept for an element or field. Other subject categories contain similar information.

- Name
- Subject-code (identifies the type of subject to which the field belongs, such as a COBOL program, PSB, or DBD)

- Status (one of thirty different, predefined codes to represent the use or ownership such as Payroll, formal tests, or production)
- Occurrence (the number of duplicate names)
- Attributes (start position, length)
- Descriptive text
- Relationships
- Alias names (other names for the same field as defined by different users)

Data Storage Techniques

To maintain the subject categories, relationships, and attributes with integrity, security, and flexible access, the basic Dictionary is implemented as five data bases. There exists one each for fields (elements), segments (records), data bases (data sets), and PCB's. All the other subject categories are grouped together into a system data base. There is a sixth, optional data base called the extensibility file. The use of this data base is explained in the section on optional and additional functions. These six data bases are associated with each other by DL/I logical relationships.

The data definitions, relationships, and attributes are entered into the appropriate data base through the use of a special batch program *command language* or the *Interactive Display Forms Facility*. The batch command language has extensive capabilities to initially create, add, change, or delete the information in Dictionary data bases. These commands also allow the Dictionary to accept the information for initial creation from existing DL/I DBD's, DL/I PSB's, or the section in COBOL or PL/I application programs which describes a record. This saves the time and effort of rewriting the same information for dictionary purposes.

The Interactive Display Forms Facility has the same powerful capabilities as the batch input commands. The primary difference and value is that it is an efficient, easy to use online interface which simplifies the task of maintaining the data definitions in the dictionary. This interface is designed to be used by all installation personnel, not just programmers. This is accomplished through a "fill-in-the-blanks" technique which allows effective use without the terminal user needing to know the commands and their specific formats. Use of this facility included in the basic Dictionary product should increase the accuracy of the data and the productivity of manipulating the data.

Information Retrieval Techniques

With a special portion of the command language, over thirty different standard reports can be generated. For instance, one report (subject report) prints all the information stored in the dictionary for a single specified subject. This includes description text, attributes, and all relationships. There is a hierarchy report which prints a picture of the hierarchical structure of a data base including all segments and fields within segments.

In addition to these fixed reports provided by the Dictionary programs, users can access the Dictionary data bases with their own programs. The combination of standard and user provided reports will assist in achieving consistent documentation and in reducing redundancy of data elements.

The command language also provides for the retrieval of the data descriptions from the Dictionary data bases for use in DBD's, PSB's, and COBOL and PL/I application programs. The outputs from the Dictionary when placed in the DL/I DBD's, PSB's and application programs are in the same format as if they were manually entered by a programmer. As an example of the tremendous productivity of this capability, suppose that a new application being implemented adds data to an existing data base. In order to add the new segment types to the data base, changes are required to the DBD. Through the online facilities of the Dictionary, we can modify the DBD definition stored in the Dictionary data base. After editing it for accuracy, we request an output of the new updated version of the DBD and subsequently process it through a DBDGEN. This is easier and less risky than manually changing the original DBD.

For another example, a change is to be made to the length of a field. First we would use the Dictionary to identify all records and segments which are affected. Using commands, we can then specify the restructuring of each affected segment or record. This resets the start position of all other fields in those segments and the length of the segments. These updated record descriptions (structures) will replace the ones in the affected application programs.

All of these information outputs from the Dictionary data bases help the entire installation, especially the data base administration function, to achieve the necessary control of the resources in a productive and efficient manner.

Additional Optional Functions

Four significant optional functions are included in the IBM DB/DC Data Dictionary program product. They are the SCAN report, the security facility, the extensibility facility, and the program access facility.

SCAN Report

The command SCAN is used to produce reports that only include records or fields which contain data that meets specified conditions. It can be directed at any of the Dictionary data bases to perform text searching for single character groupings (strings) or a pair of groupings connected by an and/or relationship. This might be used, for instance, to locate and print all fields with a status of I and with "TAX" or "CR" in their description. The SCAN command and report make the information from the Dictionary more accessible and useful for specific purposes.

Security Facility

The Dictionary security function allows the identification of users by codes. This code is then used to control access to the Dictionary subject categories, and to define which subjects the user may view and/or update. Thus not only is the data protected by DL/I, but also the description of that data is secured by the Dictionary from unauthorized access.

Extensibility Feature

This feature adds another Dictionary data base to the five basic ones. This optional data base contains subject categories, relationships, and attributes defined by the user to meet their own particular dictionary needs. This allows the user to extend his dictionary information to include such things as personnel, hardware devices, online screen formats, and organizational responsibilities. The new subject categories may have user-defined relationships between them and the Dictionary-supplied categories. This data base can be maintained through the use of the Dictionary command language or through the Interactive Display Forms Facility. This expands the Dictionary to include any important resource of an installation.

Program Access Facility

The program access facility allows user application programs to extract information from the Dictionary data bases. These user-written report programs could be used to generate special reports not provided by the standard Dictionary report programs. They might also be used to compare and validate definitions. The use of this facility results in a Dictionary system or application which is tailored to each installation's requirements.

Summary

The data processing environment is experiencing constant growth, ever-increasing complexity, and rapidly accelerating change. This introduces a need for controlling, auditing, and measuring the growth and change which is occurring. Of significant importance is the management and control of the data processing operation and the data resource. The IBM DB/DC Data Dictionary is designed as a tool to assist in the implementation of these management and control activities.

The Dictionary is the central repository of information about resources involving data. It is the vehicle to help in ensuring that information is current, consistent, and complete.

Exercise 6.4

1. The Dictionary maintains three types of information about resources: _____, _____, and _____.
2. The resources defined by the basic dictionary are divided into ten groups called _____.
3. (True or False) These 10 pre-defined groups of resources are the only resources definable in the dictionary.
4. The data to be stored by the Dictionary is entered into the system through the use of one of two available Dictionary facilities. They are a _____ and the _____.
5. The user can produce standard reports on Dictionary contents by using the appropriate part of the batch _____.
6. (True or False) User-written programs can access the Dictionary data bases to produce non-standard reports and validate stored definitions.

Solution 6.4

1. a) definitions, b) attributes, c) relationships
2. Subject categories
3. False. Through the extensibility feature user-defined subjects may be stored by the Dictionary.
4. a) batch command language
b) interactive display forms facility
5. Command language
6. True

Unit 7:

DOS/VSE System IPO/E

Introduction DOS/VSE and the IBM program products that have been discussed in the preceding units must be installed on your 4300 or S/370 processor before they can be used. The process of installing a DOS/VSE system is referred to as *system generation*.

The system generation process includes the selection of optional system support features and library components required to meet the unique processing requirements for a given installation. This selection process is called "tailoring" the installation. Tailoring allows the creation of a system which contains only those options required. The programming that supports the options that are not selected will not be included in the tailored supervisor. This reduces the amount of processor storage required by the supervisor thereby increasing the storage available for execution of application programs.

The system generation process generally requires the installation of IBM-supplied software maintenance to one or more of the system components. The software maintenance addresses problems that were not detected until after the component was released.

The IBM DOS/VSE System Installation Productivity Option/Extended (DOS/VSE System IPO/E) is designed to enhance the productivity of both IBM and customer personnel in the installation, service, and use of a DOS/VSE System.

This topic describes the structure and facilities of DOS/VSE System IPO/E and the Interactive Productivity Facility.

- Objectives** At the completion of this Unit, the student should be able to:
- Describe the DOS/VSE System IPO/E offerings.
 - Explain how the interactive full-screen display and prompting are used for easier and more productive communication with the system.
 - Discuss how online information may contribute to reducing user prerequisite education and assists the new or infrequent user.
 - Describe the functions of the Interactive Productivity Facility.
 - Explain how the user interacts with the Interactive Productivity Facility through the use of dialogues and display panels.

Average Study Time 2 to 3 Hours

**Topic 1. DOS/VSE
System IPO/E Structure
and Content**

The purpose of *DOS/VSE* System *IPO/E* is to reduce the time and skill level required for customer and IBM personnel to install, maintain, and use *DOS/VSE* and selected licensed programs. This is accomplished by providing IBM customers with a *DOS/VSE* Base and selected licensed programs ready to use in specific operating environments.

The major System *IPO/E* facilities offered by the combination of *DOS/VSE* and selected licensed programs are:

1. Step-by-step procedures for initially installing and verifying a *DOS/VSE* system and selected components. This is designed to reduce the installation time required.
2. Prewritten routines to assist in the installation and tailoring process. This reduces the time spent in generating individual routines.
3. Installation Verification Procedures that contain sample programs and data that may be used to verify the installation and may also be used as examples for application development.
4. Assistance with the management of installed systems in the form of online procedures that aid the user in the management and maintenance functions. These functions include application of software maintenance, installation of selected licensed programs, and monitoring system performance.
5. Assistance with the use of the installed system in the form of online information that aids the user in the development and implementation of new applications. This includes information to help the user create, modify, compile, and execute programs using the Virtual Storage Extended/Interactive Control and Computing Facility (VSE/ICCF).

Each System *IPO/E* includes a base set of products and a series of optional features. The base is an integrated set of software components. These components are pregenerated for a specific operating environment and include the current level of software maintenance.

With *DOS/VSE*, three System *IPO/E*'s are available. These are: the Batch Interactive Base, the Data Communication Base, and the Data Base/Data Communications Base.

Each *DOS/VSE* System *IPO/E* base has a set of complementary program products that can be ordered as features to the base. Although ordered separately, these features have been synchronized and operationally verified with the base. Included in the base package are ease of use facilities for the System *IPO/E* features, including feature installation prompters that assist when installing features.

All program products are available under the agreement for IBM licensed programs.

Let us now look at the base components of each of the System *IPO/E*'s and their related complementary products.

DOS/VSE System IPO/E - Batch Interactive Contents The Batch/Interactive base is composed of DOS/VSE, VSE/Advanced Functions, VSE/IPCS, VSE/ICCF, VSE/POWER, BTAM-ES, BTAM-SCP, VSE/VSAM, and the Interactive Productivity Facility. To complement the Batch/Interactive base, a series of program products are available as features to the base set. The COBOL, VSE/DITTO, and RPGII program products are examples.

DOS/VSE System IPO/E - DC Contents The Data Communication base combines the Batch/Interactive base with CICS/VS. Available with the DC base, in addition to the B/I features, are a series of complementary products. The SORT/VS, DMS/CICS/VS-DOS, and VSE/POWER/RJE program products are examples.

DOS/VSE System IPO/E - DB/DC Contents The Data Base/Data Communication base combines DL/I DOS/VS with the DC Base and adds the Data Dictionary as a feature to provide a full DB/DC system.

Notice that each base of System IPO/E includes the Interactive Productivity Facility program product. The Interactive Productivity Facility provides the user with interactive full-screen displays and dialogues to assist in managing and using the system. The Interactive Productivity Facility also contains online information and, through the use of its First Use Tutorial, data processing professionals and non-DP-experienced people alike can productively use the system more quickly.

DOS/VSE System IPO/E is structured so that the user can interface with the system through the Interactive Productivity Facility. See Figure 7.1.1.

The Interactive Productivity Facility can be thought of as an insulating layer that helps insulate the user from much of the complexity of the system just as applications, languages, sorts, and utilities help insulate the user from the complexity of subsystems.

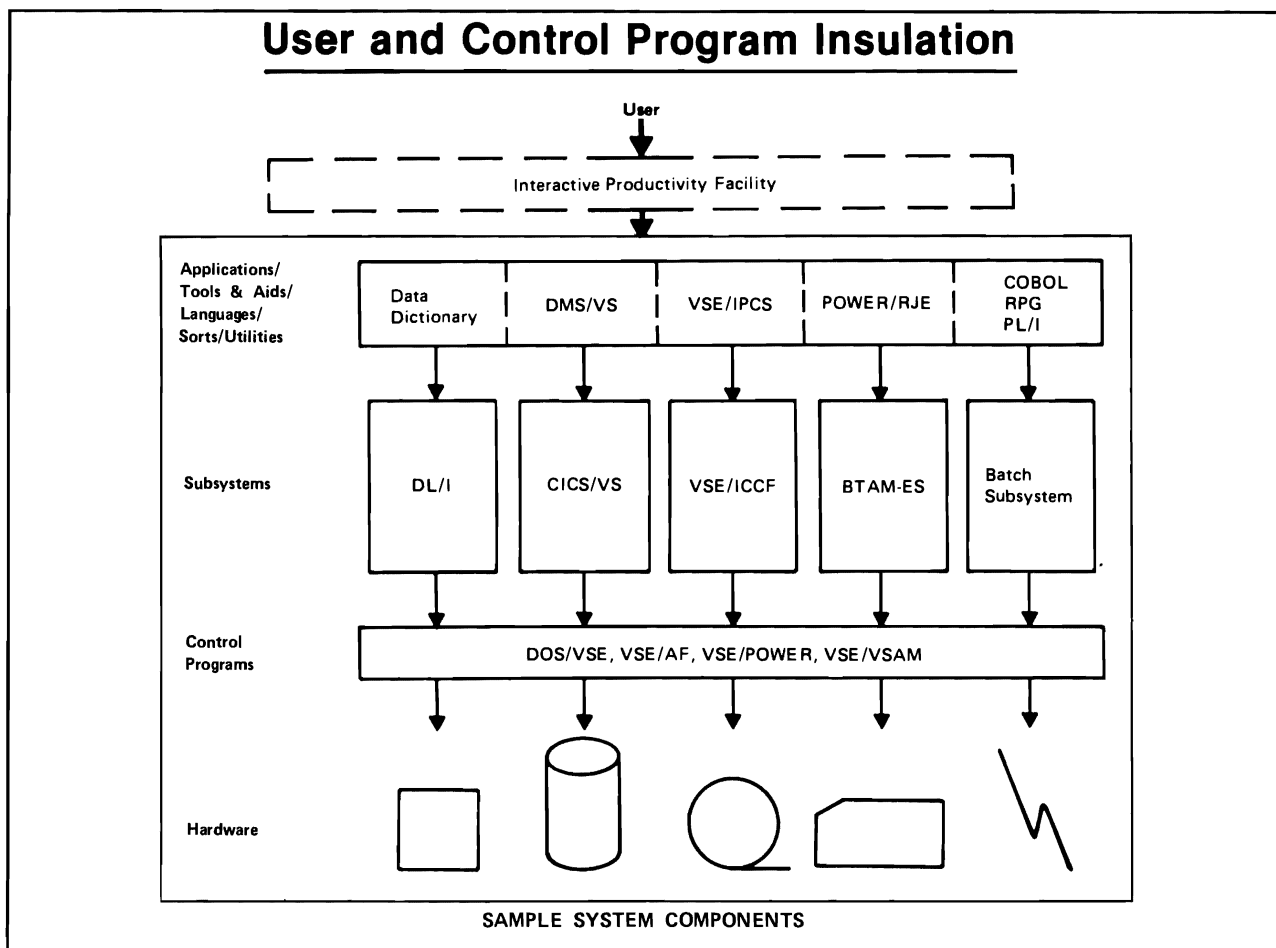


Figure 7.1.1. Sample System Components

In the next topic, you will learn more about the Interactive Productivity Facility. But first, complete this exercise. If you miss any of these questions, review the appropriate material in the text.

Exercise 7.1

1. The process of installing as DOS/VSE system is referred to as a _____.
2. The purpose of the DOS/VSE System IPO/E's is to:

3. (True or False) DOS/VSE System IPO/E's are hardware features ordered with the machine.
4. (True or False) DOS/VSE System IPO/E includes the Interactive Productivity Facility Program Product as part of its base.
5. (True or False) The DOS/VSE System IPO/E provides support for batch/interactive environments, data communications environments (CICS) and data base/data communications environments (DLI/CICS) with three separate base System IPO/E's.
6. (True or False) The Interactive Productivity Facility helps to insulate the user from the complexities of the system.

Solution 7.1

1. system generation.
2. Reduce the time and skill level required to install, service and use DOS/VSE and selected licensed programs.
3. False. System IPO/E is comprised of software products pre-configured and pre-generated for specific hardware environments.
4. True.
5. True.
6. True.

**Topic 2. Overview of the
Interactive Productivity
Facility**

The facilities of DOS/VSE System IPO/E are implemented through the Interactive Productivity Facility program product. This program product can be thought of as an insulating layer that keeps the user away from the complexity of using and managing the system. That is to say, the Interactive Productivity Facility provides the user with a friendly interface, reducing the time and skill level required to use and manage a DOS/VSE System.

***Communicating With The
Interactive Productivity
Facility***

The Interactive Productivity Facility communicates with the user through display devices using the full screen display capability of these devices. Full screen displays of data, called Display Panels, or just panels, are presented to the user.

Some panels are called "Menu Panels" for they offer several options, one of which is selected by the user. Other panels are called "Data Entry Panels." These panels require the user to enter data. A third panel type are the "Explain Panels." These panels offer explanatory information about a "Menu Panel" or a "Data Entry Panel." Later in this chapter you will learn how all three types of panels are used. For now, let's learn more about the panel structure.

Hierarchical Panel Structure

The menu panels of the Interactive Productivity Facility are arranged in a hierarchical structure so that by starting at the highest-level menu panel one can progress logically through the hierarchically arranged menus and arrive at a function that is to be executed. See Figure 7.2.1. for an example of the hierarchical structure.

For each menu panel presented, the user must respond, selecting the option required. If the response corresponds to a lower level menu panel, that menu panel is displayed. If the response corresponds to a function, that function is invoked. As shown in Figure 7.2.1., from the INITIAL MENU, the user moves through TAILORING to CICS to TRANSACTION to the function of CREATE TRANSACTIONS.

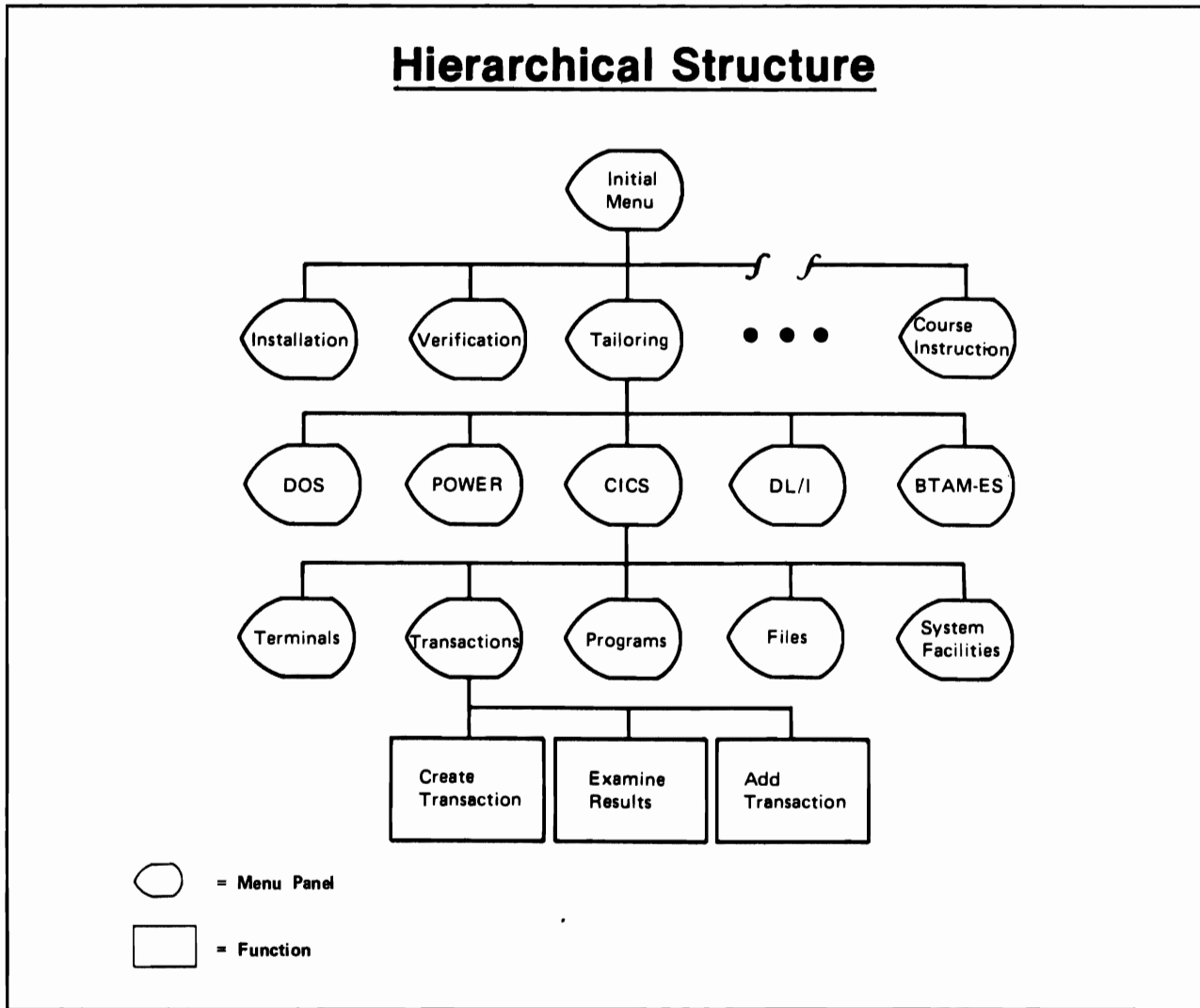


Figure 7.2.1. Menu Structure

Functions supported by the Interactive Productivity Facility can be broadly classified into two areas: System Management and System Use.

System Management

System management consist of those activities that keep a computer system operational. Examples of system management activities include modifying the Automatic Systems Initialization procedures and creating DOS/VSE job streams that invoke the DOS/VSE utilities that print the contents of VSAM data sets.

System Use

System use consist of those activities that interface directly with the end user to help create a work product. System use includes online information to help end users create or modify a program and information on how end users would submit a job to a DOS/VSE partition from their display terminal.

First Use Tutorial

In addition to providing System Management and System Use functions, the Interactive Productivity Facility also provides an online tutorial. This tutorial is designed to teach the major facilities of the Interactive Productivity Facility as implemented in the DOS/VSE System IPO/E. This feature can be of significant value to the first-time users of DOS/VSE System IPO/E.

Interactive Productivity Facility Display Panels

Before showing you an example of the hierarchical panel structure of the Interactive Productivity Facility, let's first examine the format and the different features and facilities of the panels.

The following panels constitute a demonstration of the Interactive Productivity Facility.

You will be viewing sample panels and functions that may not appear exactly as shown in the released product.

Sign-On Panel

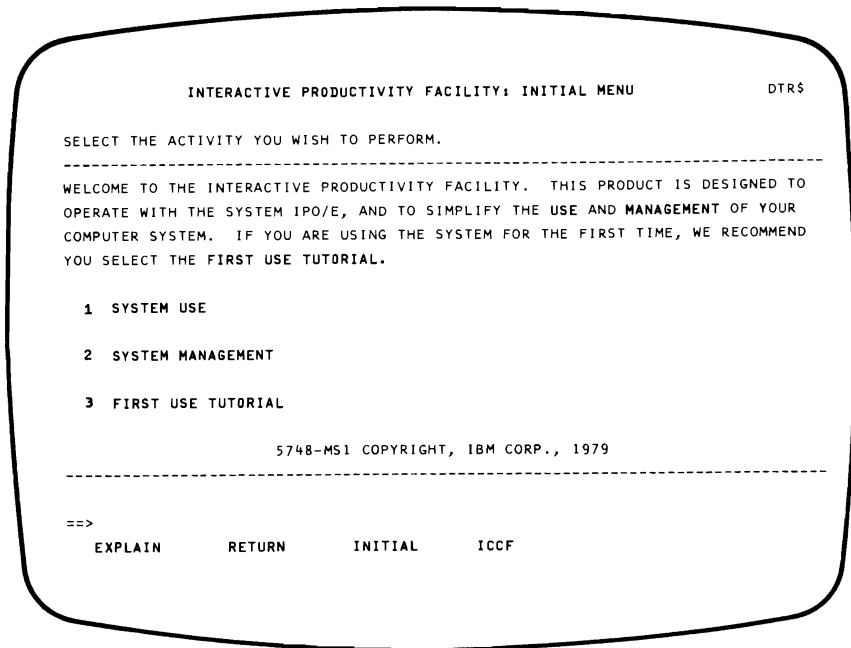


Figure 7.2.2. Sign-On Panel

Figure 7.2.2. shows the Interactive Productivity Facility Sign-On panel. This panel is a "Menu Panel." We will use it to describe the format and the different features and facilities of the panels.

Panel Identification

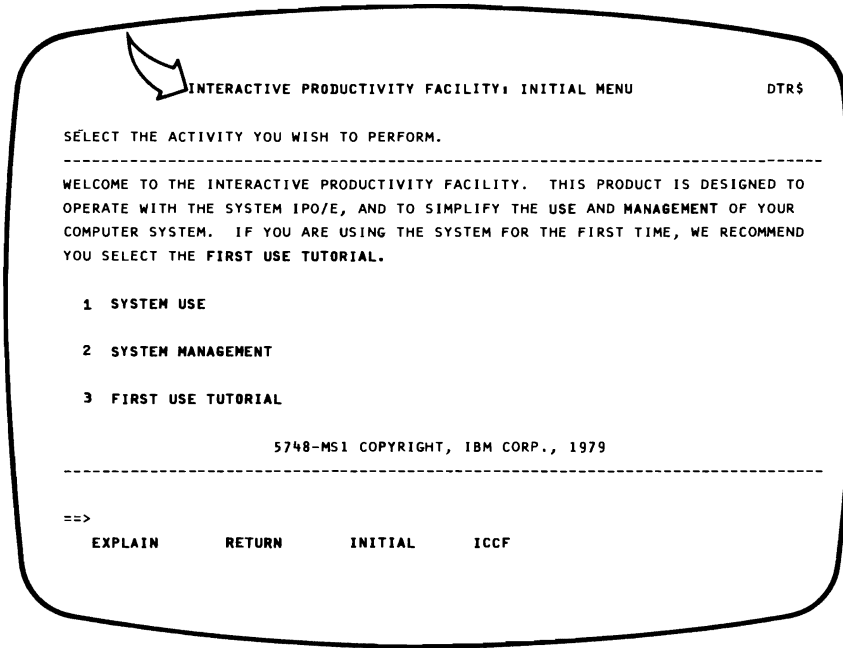
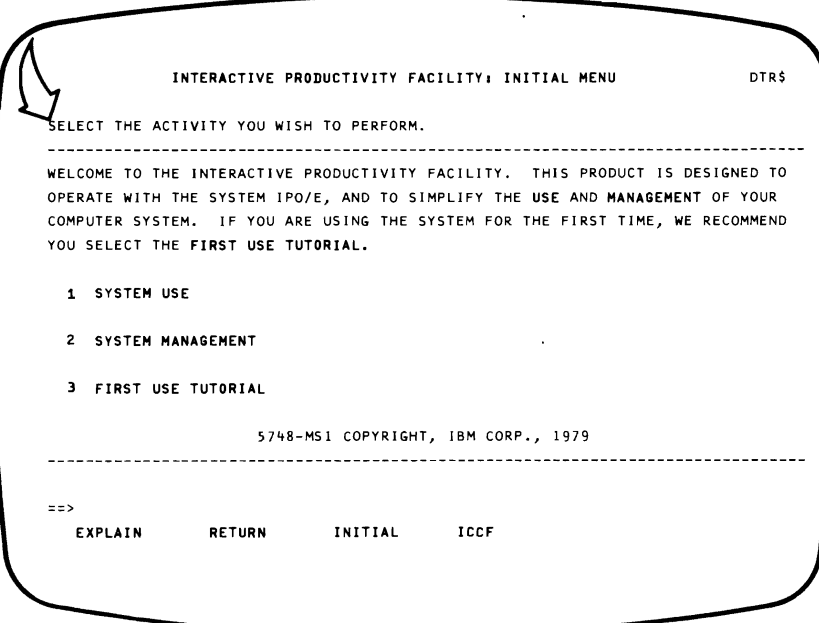


Figure 7.2.3. Sign-On Panel

Each panel has a title at the top, with the identification (ID) of the panel in the upper right-hand corner.

Instruction Line

```
INTERACTIVE PRODUCTIVITY FACILITY: INITIAL MENU          DTR$  
SELECT THE ACTIVITY YOU WISH TO PERFORM.  
-----  
WELCOME TO THE INTERACTIVE PRODUCTIVITY FACILITY. THIS PRODUCT IS DESIGNED TO  
OPERATE WITH THE SYSTEM IPO/E, AND TO SIMPLIFY THE USE AND MANAGEMENT OF YOUR  
COMPUTER SYSTEM. IF YOU ARE USING THE SYSTEM FOR THE FIRST TIME, WE RECOMMEND  
YOU SELECT THE FIRST USE TUTORIAL.  
  
1 SYSTEM USE  
  
2 SYSTEM MANAGEMENT  
  
3 FIRST USE TUTORIAL  
  
5748-MS1 COPYRIGHT, IBM CORP., 1979  
-----  
==>  
EXPLAIN      RETURN      INITIAL      ICCF
```

Figure 7.2.4. Sign-On Panel

Each panel contains an instruction line that is a brief statement of what is required on this panel. In this example, the terminal user would select System Use or System Management activities, or the First Use Tutorial.

Descriptive Text

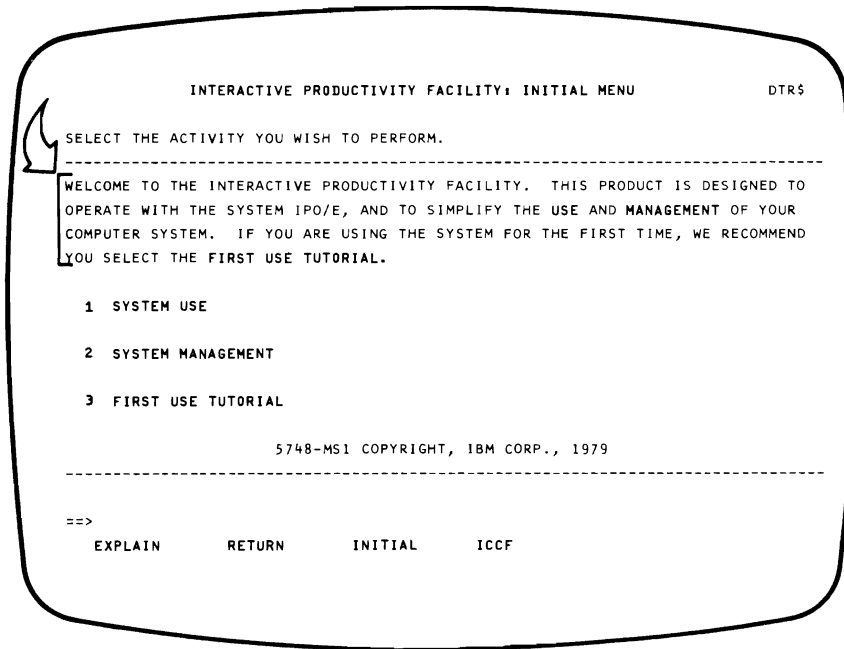


Figure 7.2.5. Sign-On Panel

Following the instruction line is descriptive text that explains in detail the instructions and actions to be taken on this panel. Keywords are highlighted for quick reference.

Menu Selection List

```
INTERACTIVE PRODUCTIVITY FACILITY: INITIAL MENU          DTR$

SELECT THE ACTIVITY YOU WISH TO PERFORM.
-----
WELCOME TO THE INTERACTIVE PRODUCTIVITY FACILITY. THIS PRODUCT IS DESIGNED TO
OPERATE WITH THE SYSTEM IPO/E, AND TO SIMPLIFY THE USE AND MANAGEMENT OF YOUR
COMPUTER SYSTEM. IF YOU ARE USING THE SYSTEM FOR THE FIRST TIME, WE RECOMMEND
YOU SELECT THE FIRST USE TUTORIAL.

 1 SYSTEM USE
 2 SYSTEM MANAGEMENT
 3 FIRST USE TUTORIAL

                    5748-MS1 COPYRIGHT, IBM CORP., 1979
-----

==>
  EXPLAIN      RETURN      INITIAL      ICCF
```

Figure 7.2.6. Sign-On Panel

This is an example of a menu selection list. The keywords and the menu numbers are intensified and light-pen detectable. The user selects the desired alternative by light pen or by keying in the option number.

User Input Area

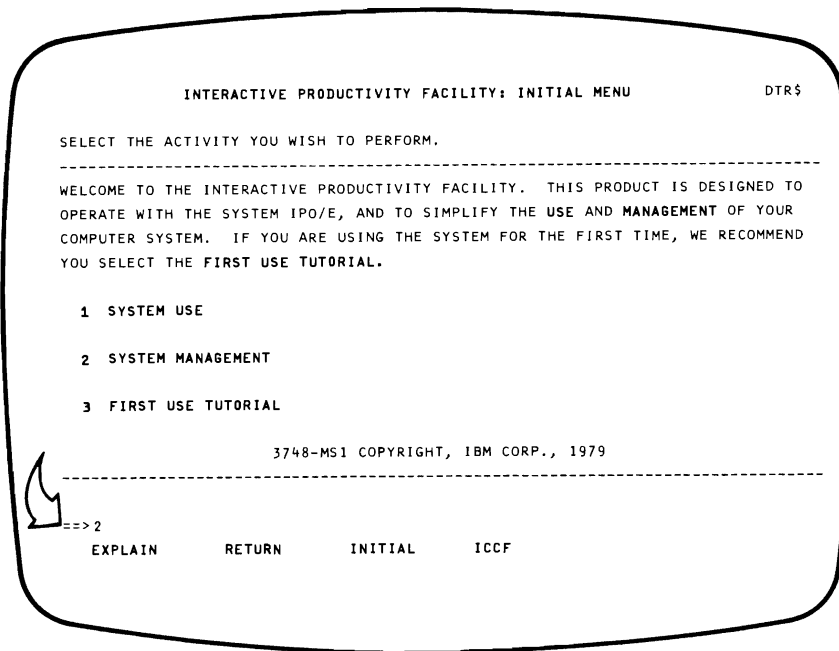


Figure 7.2.7. Sign-On Panel

Input lines are provided where the user is to supply data. The cursor is automatically set to accept keyed data following the arrow (==>). The user has selected System Management by entering a "2".

As you will see later, when the field length is pre-defined, dashes are provided to indicate the number of characters allowed. Where possible, default parameters are provided. If the user wants different parameters, then the default parameters can be typed over. In most cases, the overriding parameters become the new default parameters for that user.

Miscellaneous Commands

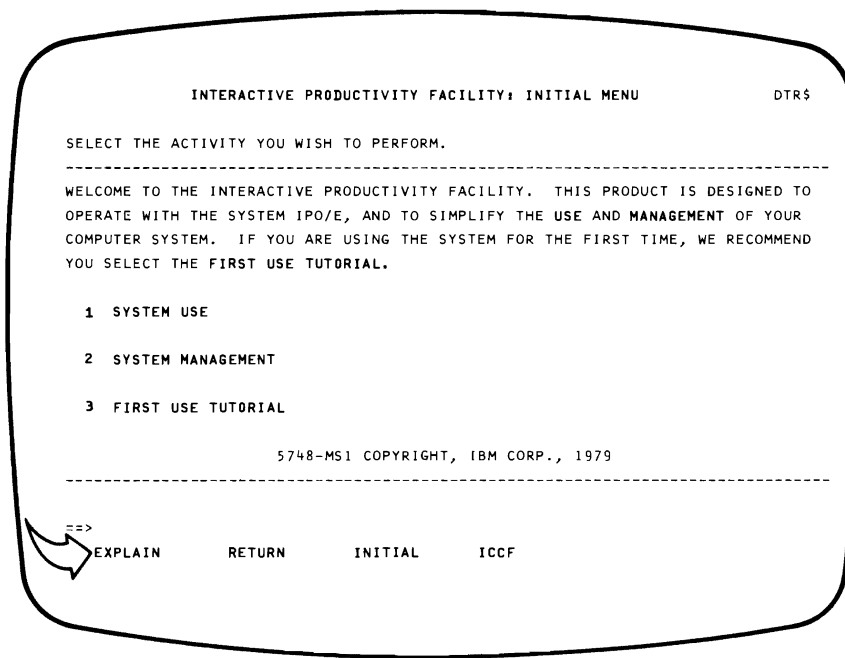


Figure 7.2.8. Sign-On Panel

At the bottom of each panel a series of commands is displayed. These commands vary according to the panel type and can be keyed or light-pen selected to invoke the corresponding functions.

Each command represents a selectable service that is valid for this panel. The following services are available for Menu Panels:

a. **EXPLAIN**

If your response is **EXPLAIN**, one or more panels of explanatory text are displayed. For certain menu panels, **EXPLAIN** panels are not available. If so, a message is given stating that no additional information is available.

If there are several **EXPLAIN** panels, you may browse through them before returning to the panel from which **EXPLAIN** was requested.

b. **RETURN**

If your response is **RETURN**, the menu immediately preceding the current menu in the hierarchy is displayed. If there is no preceding menu, the current menu is redisplayed.

c. **INITIAL**

If your response is **INITIAL**, the Interactive Productivity Facility Initial Menu is displayed.

d. **ICCF**

This selection will exit from the System IPO/E and place you in command mode under VSE/ICCF.

So far, the panels shown in the examples have been "Menu Panels." Let's look now at the other panel types; "Explain Panels" and "Data Entry Panels".

Data Entry Panels

"Data Entry Panels" are displayed after the user has selected a function from a menu panel and information must be entered to perform that function. Figure 7.2.9. gives an example of a Data Entry Panel used to add user transactions to a CICS/VS system.

```

                                CICS/VS: ADD TRANSACTIONS                                TCISAPC1
-----
COMPLETE THE REQUIRED DATA AND PRESS ENTER.
-----
THIS IS THE FIRST OF TWO PANELS USED TO ADD NEW TRANSACTIONS TO THE CICS/VS
PROGRAM CONTROL TABLE (PCT).  ENTER THE REQUESTED INFORMATION INDICATED BELOW,
AND PRESS ENTER.  A SECOND PANEL WILL THEN BE DISPLAYED, ASKING YOU FOR FURTHER
INFORMATION.

IF YOU HAVE MORE THAN ONE TRANSACTION TO ADD, THESE PANELS WILL BE RETURNED TO
YOU AS MANY TIMES AS NECESSARY.  AFTER YOU HAVE ADDED ALL YOUR TRANSACTIONS,
TYPE /END IN THE TRANSACTION ID FIELD, AND PRESS ENTER.

TRANSACTION ID      ==>  ----      A UNIQUE FOUR-CHARACTER CODE
PROGRAM NAME        ==>  -----     THE NAME OF THE PROGRAM WHICH PRO-
CESSES THIS TRANSACTION
WORK AREA SIZE      ==>  256        THE SIZE OF THE WORK AREA REQUIRED BY
THIS TRANSACTION, IN BYTES
STORAGE USED        ==>  4K         THE AMOUNT OF DYNAMIC STORAGE USED BY
THIS TRANSACTION, IN 2K MULTIPLES
-----
==>
      EXPLAIN      CANCEL      RETRY

```

Figure 7.2.9. Data Entry Panel

Panel Format

These panels have the same general format as the menu panels. As indicated by the arrows, in Figure 7.2.10., they have a panel identification, an instruction line, and a descriptive text.

```

CICS/VS: ADD TRANSACTIONS                                TCISAPC1
-----
COMPLETE THE REQUIRED DATA AND PRESS ENTER.
-----
THIS IS THE FIRST OF TWO PANELS USED TO ADD NEW TRANSACTIONS TO THE CICS/VS
PROGRAM CONTROL TABLE (PCT).  ENTER THE REQUESTED INFORMATION INDICATED BELOW,
AND PRESS ENTER.  A SECOND PANEL WILL THEN BE DISPLAYED, ASKING YOU FOR FURTHER
INFORMATION.

IF YOU HAVE MORE THAN ONE TRANSACTION TO ADD, THESE PANELS WILL BE RETURNED TO
YOU AS MANY TIMES AS NECESSARY.  AFTER YOU HAVE ADDED ALL YOUR TRANSACTIONS,
TYPE /END IN THE TRANSACTION ID FIELD, AND PRESS ENTER.

TRANSACTION ID      ==>  ----      A UNIQUE FOUR-CHARACTER CODE
PROGRAM NAME       ==>  -----     THE NAME OF THE PROGRAM WHICH PRO-
CESES THIS TRANSACTION
WORK AREA SIZE     ==>  256        THE SIZE OF THE WORK AREA REQUIRED BY
THIS TRANSACTION, IN BYTES
STORAGE USED       ==>  4K         THE AMOUNT OF DYNAMIC STORAGE USED BY
THIS TRANSACTION, IN 2K MULTIPLES
-----

==>
EXPLAIN      CANCEL      RETRY

```

Figure 7.2.10. Data Entry Panel

Data Entry Area

Data Entry Panels differ from Menu Panels in that they indicate what data fields are to be entered to perform a function. As shown in Figure 7.2.11., four data fields could be entered. The last two, "work area size" and "storage used" have defaults assigned. The user has the option of using these defaults or overriding them. To override these defaults, the user will simply key in the new values.

```

                                CICS/VS: ADD TRANSACTIONS                                TCISAPC1
-----
COMPLETE THE REQUIRED DATA AND PRESS ENTER.
-----
THIS IS THE FIRST OF TWO PANELS USED TO ADD NEW TRANSACTIONS TO THE CICS/VS
PROGRAM CONTROL TABLE (PCT).  ENTER THE REQUESTED INFORMATION INDICATED BELOW,
AND PRESS ENTER.  A SECOND PANEL WILL THEN BE DISPLAYED, ASKING YOU FOR FURTHER
INFORMATION.

IF YOU HAVE MORE THAN ONE TRANSACTION TO ADD, THESE PANELS WILL BE RETURNED TO
YOU AS MANY TIMES AS NECESSARY.  AFTER YOU HAVE ADDED ALL YOUR TRANSACTIONS,
TYPE /END IN THE TRANSACTION ID FIELD, AND PRESS ENTER.
-----
TRANSACTION ID      ==>  ----      A UNIQUE FOUR-CHARACTER CODE
PROGRAM NAME        ==>  -----     THE NAME OF THE PROGRAM WHICH PRO-
WORK AREA SIZE      ==>  256        THE SIZE OF THE WORK AREA REQUIRED BY
STORAGE USED        ==>  4K         THE AMOUNT OF DYNAMIC STORAGE USED BY
-----
                                ==>
                                EXPLAIN      CANCEL      RETRY

```

Figure 7.2.11. Data Entry Panel

Data Entry Panels also indicate the maximum length and give a brief description of the data fields to be entered. As shown in Figure 7.2.12., the first data entry field has four dashes, coaching the user to enter the correct length transaction ID of four characters. The second field, PROGRAM NAME, has eight dashes. The WORK AREA SIZE and STORAGE USED fields are already filled in with default parameters, which the user may override.

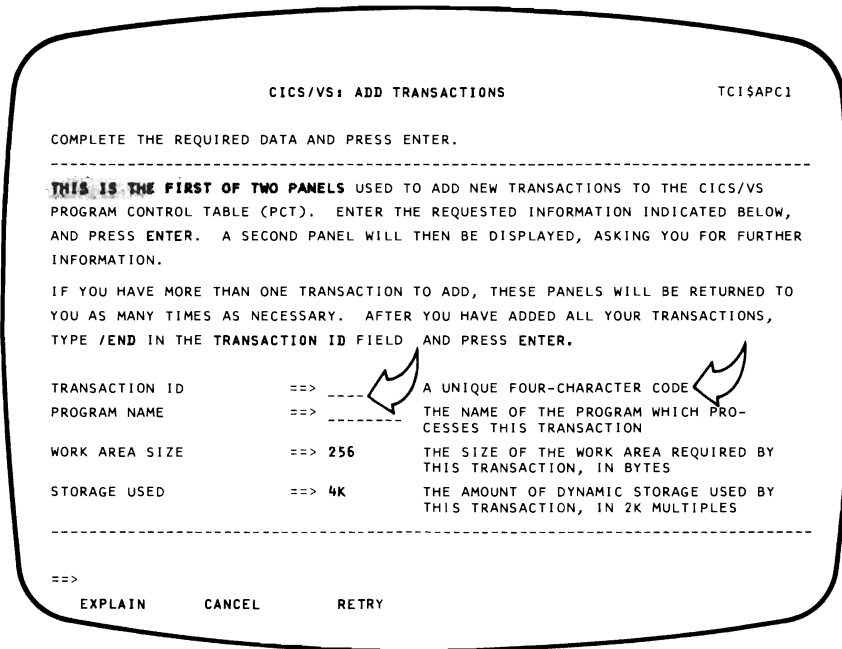


Figure 7.2.12. Data Entry Panel

Command Line

The command line differs slightly from Menu Panels.

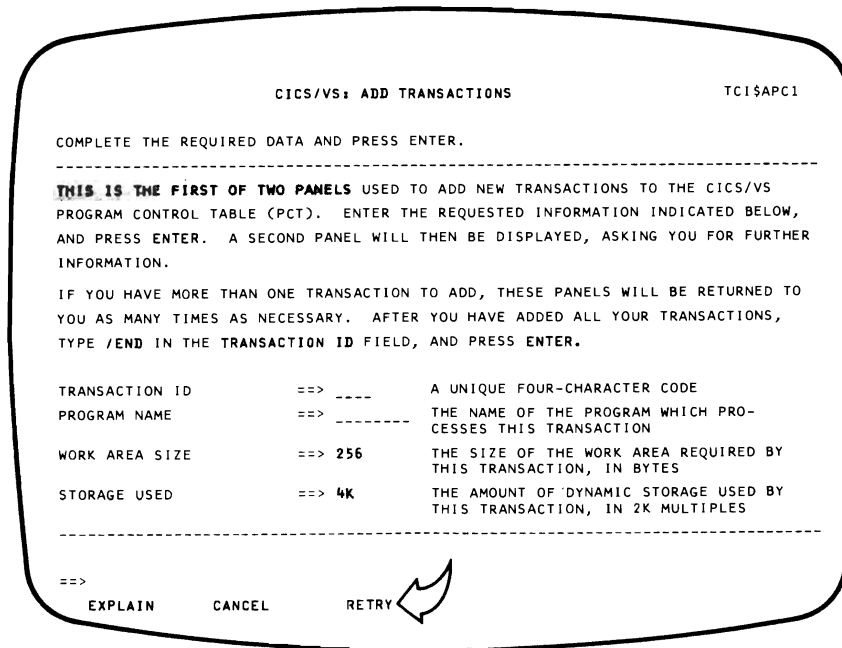


Figure 7.2.13. Data Entry Panel

The command line contains a list of services that are valid for this panel. The following services are provided in Data Entry panels:

a. **EXPLAIN**

If your response is EXPLAIN, one or more panels of explanatory text are displayed.

If there are several EXPLAIN panels, you may browse through them before returning to the data entry panel from which EXPLAIN was requested.

b. **RETRY**

If your response is RETRY, the function redisplay the data entry panel in its original state. All data entered on previous panels is still used. Normally, "RETRY" will be used if data entered on the current panel is incorrect.

c. **CANCEL**

If the response is CANCEL, the last menu panel displayed is redisplayed. All data entered since then is ignored. Normally, "cancel" will be used if the wrong function has been selected.

Explain Panels

Explain Panels are displayed if the user selects the EXPLAIN response of the command line on Data Entry or Menu Panels. These panels present you with explanatory information about a Data Entry or Menu Panel. Figure 7.2.14. gives an example of an Explain Panel.

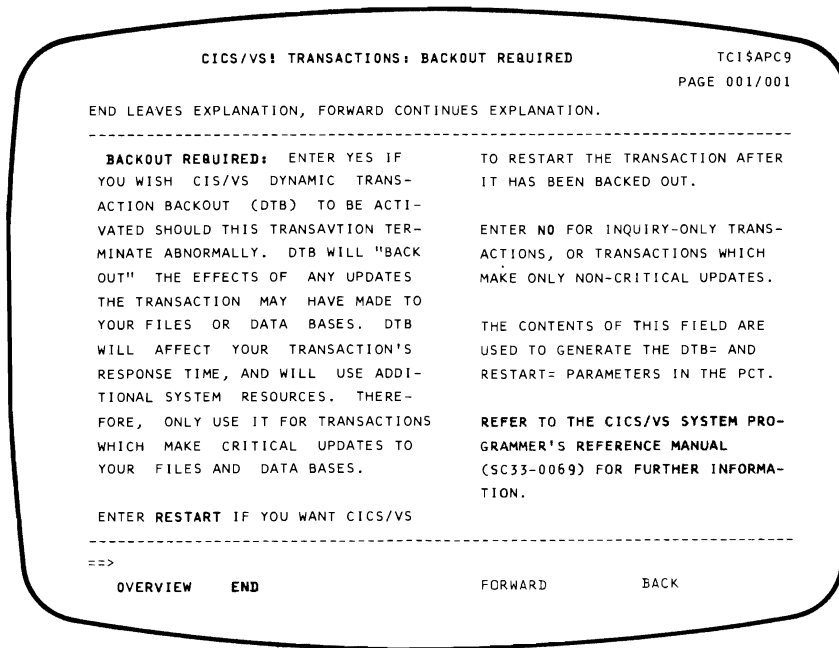


Figure 7.2.14. Explain Panel

Panel Format

As indicated by the arrows in Figure 7.2.15., Explain Panels have a panel identification and an instruction line.

Command Line

The command line differs from other panels. Figure 7.2.17. gives an example of the command line.

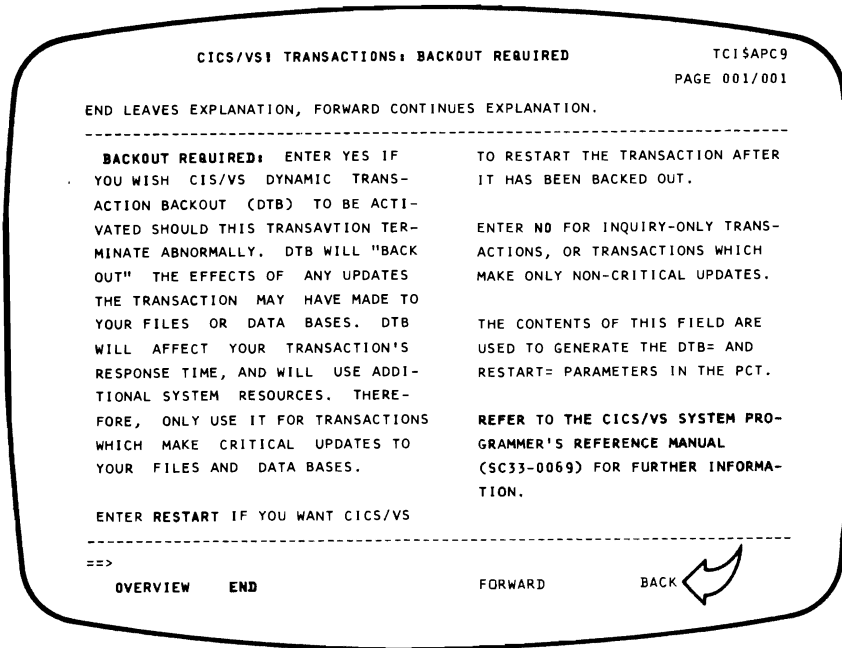


Figure 7.2.17. Explain Panel

The command line contains a list of services that are valid for this panel. The following services are provided in Explain panels:

a. **OVERVIEW**

If your response is **OVERVIEW**, a menu of explanation information available is displayed.

b. **END**

If your response is **END**, the panel from which **EXPLAIN** was requested is displayed next. The **END** service is simulated if you press the **ENTER** key when the last panel in a multi-panel sequence is being displayed.

c. **FORWARD**

If your response is **FORWARD**, the next explain panel in a multi-panel sequence is displayed. This option is not available on the last panel. If you press the **ENTER** key, the **FORWARD** service is simulated.

d. **BACK**

If your response is **BACK**, the previous explain panel in a multi-panel sequence is displayed. This option is not available on the first panel.

In the next topic you will see how the user interfaces with the Interactive Productivity Facility to perform a function and a demonstration of all the

panel types you have just reviewed. But first complete this exercise. If you miss any of these questions, review the appropriate material in the text.

Exercise 7.2

1. (True or False) The Interactive Productivity Facility is designed to reduce the time and skill level required to manage and use a DOS/VSE system.
2. (True or False) The Interactive Productivity Facility presents a hierarchical structure of menu panels to the user.
3. What are the three classifications of activities that are supported by the Interactive Productivity feature?

4. Online information to help an end user create or modify a program is provided by the _____ activity.
5. Assistance in creating DOS/VSE job streams to print the contents of of VSAM data sets is provided by the _____ activity.
6. What are the three panel types uses by the Interactive Productivity Facility?

7. _____ panels are displayed after the user has selected a function from a _____ panel requiring information to be entered.
8. _____ panels present explanatory information about _____ and _____ panels.

Solution 7.2

1. True.
2. True.
3. System Management; System Use; First Use Tutorial.
4. System Use.
5. System Management.
6. Explain Panels, Data Entry Panels, Menu Panels.
7. Data Entry, Menu.
8. Explain, Data Entry, Menu.

Topic 3. Communicating With The Interactive Productivity Facility

The objective of this topic is to familiarize you with the various panels and the Hierarchical Panel Structure of the Interactive Productivity Facility. To do this, we will follow the sequence of events and the panels displayed when a CICS/VS user adds a transaction to the Program Control Table of CICS/VS. You will see the series of Menu, Data Entry, and Explain panels contained in the Interactive Productivity Facility that support this function. Refer to the Interactive Productivity Facility User's Guide for a complete description of all other supported functions and activities.

Our example begins with the initial menu panel of the Interactive Productivity Facility.

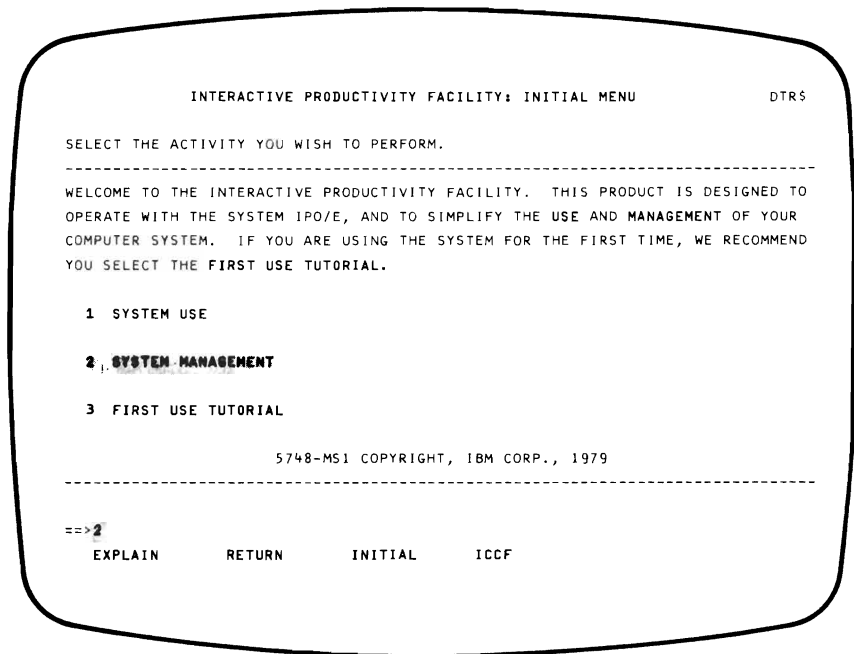


Figure 7.3.1. Initial Menu

Let's assume you have just signed onto the Interactive Productivity Facility. The menu panel shown in Figure 7.3.1. is displayed. This is the same menu panel you saw earlier. You have a choice of three menu items: SYSTEM MANAGEMENT, SYSTEM USE, and FIRST USE TUTORIAL. You want to do SYSTEM MANAGEMENT, so option 2 is keyed and ENTER is pressed. You could have light-pen selected the option.

This brings you the System Management Menu.

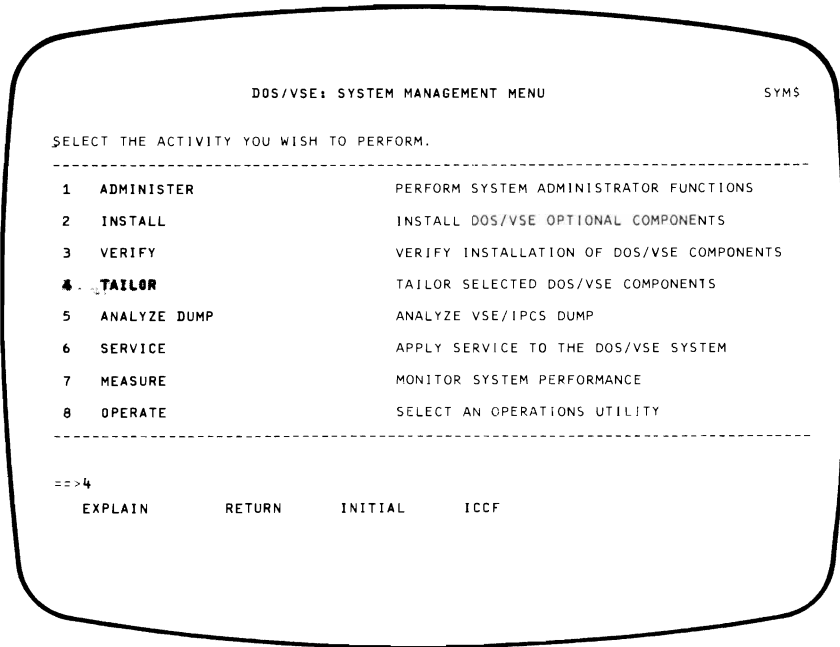


Figure 7.3.2. System Management Menu

There are several System Management activities that could be performed. The activity to be performed, as per our sample problem, is tailoring, so option 4 is selected.

This brings you another menu panel, the Tailoring Menu panel.

```

                                SYSTEM MANAGEMENT: TAILORING MENU                                TAI5
-----
SELECT THE ACTIVITY YOU WISH TO PERFORM.
-----
1  SUPERVISOR                    GENERATE DOS/VSE SUPERVISOR ASSEMBLY
2  ASI PROCEDURE                 MODIFY AUTOMATIC SYSTEM INITIALIZATION
3  VSE/POWER                     GENERATE VSE/POWER ASSEMBLY
4  VSE/POWER RJE                 GENERATE OPTIONAL VSE/POWER RJE ASSEMBLY
5  CICS/VS TABLES              MODIFY CICS/VS CONTROL TABLES
6  ADDITIONAL ACTIVITIES        SELECT FOR ADDITIONAL TAILORING ACTIVITIES
-----
=> 1
    EXPLAIN      RETURN      INITIAL      ICCF

```

Figure 7.3.3. Tailoring Menu

Here you see a number of items that could be selected.

YOU SELECT OPTION 5, CICS/VS TABLES as per our sample problem.

This causes the CICS/VS menu panel to be displayed.

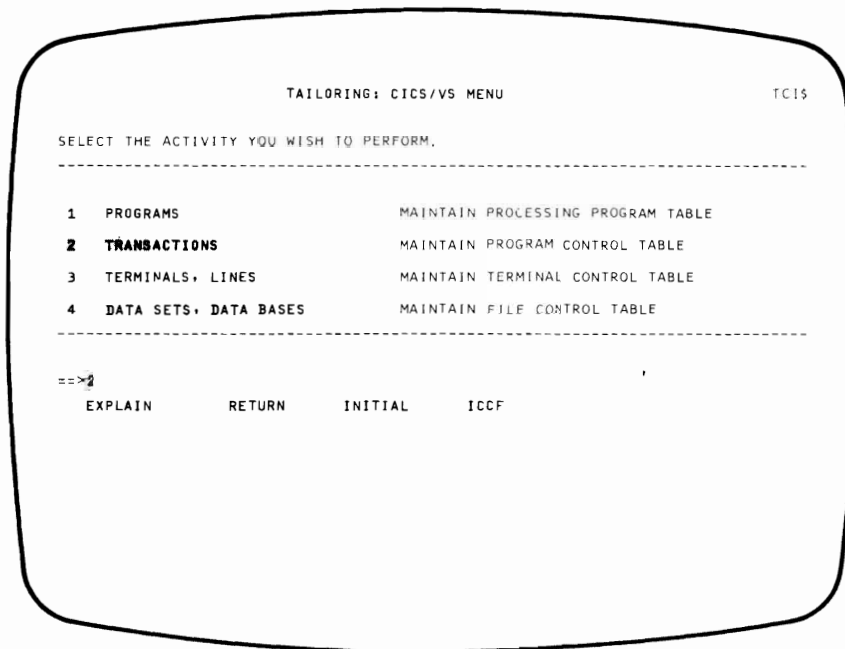


Figure 7.3.4. CICS/VS Menu

You are then given an opportunity to choose a number of CICS/VS tailoring activities. Your choice is option 2, **TRANSACTIONS - Maintain Program Control Table** . . .

. . . causing the Program Control Table Menu Panel to be displayed.

```

CICS/VS: MAINTAIN PROGRAM CONTROL TABLE                                TC152
-----
SELECT THE ACTIVITY YOU WISH TO PERFORM.
-----
THESE ACTIVITIES UPDATE THE CICS/VS PROGRAM CONTROL TABLE (PCT).
-----
1  ADD TRANSACTION                ADD ENTRY TO PCT
2  DELETE TRANSACTION             DELETE ENTRY FROM PCT
3  CHANGE TRANSACTION            CHANGE ENTRY IN PCT
4  ASSEMBLE/CATALOG PCT         ASSEMBLE AND CATALOG PCT WITHOUT CHANGES
-----
==>1
EXPLAIN      RETURN      INITIAL      ICCF

```

Figure 7.3.5. Program Control Table Menu Panel

Here, you have four functions which can be performed on the CICS/VS Program Control Table as indicated by the arrow in Figure 7.3.5.

Before continuing with the next panel, let's review what has happened. So far, you have seen five different panels. All of them were menu panels. You have made selections from each of these panels based on the activity to be performed: each selection causing the next lower panel in the hierarchical structure to be displayed. What you really are doing is, through the selection of a series of options, informing the Interactive Productivity Facility what function you want to perform.

Now, back to the example and Figure 7.3.5. Now you may choose, from four options, the function you want to perform to the Program Control Table. "Add Transaction" is required for our example, so option 1 is selected. This causes a data entry panel to be displayed. Now, you have reached the point where the Interactive Productivity Facility knows what function you want to perform. Data to execute that function is now entered.

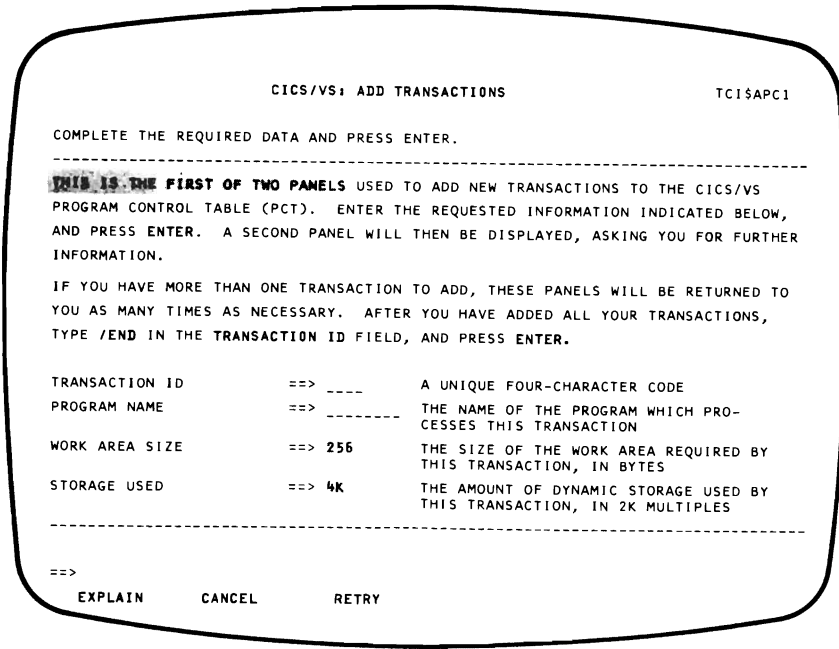


Figure 7.3.6. Add Transaction Data Entry Panel

What you see in Figure 7.3.6, is the first of two Data Entry panels for which you will be prompted for the required data to add a new transaction to the CICS Program Control Table.

On this panel, the first data entry field has four dashes, coaching the user to enter the correct length transaction ID. The second field, PROGRAM NAME, has eight dashes. The WORK AREA SIZE and STORAGE USED fields are already filled in with default parameters, which you may override.

You now key in the "Transaction ID" and the "Program Name." The default values for "Work Area Size" and "Storage Used" are valid for this transaction so no additional keying is required.

You now press enter, and . . .

```

                                CICS/VS: ADD TRANSACTIONS                                TC1$APC2
-----
COMPLETE THE REQUIRED DATA AND PRESS ENTER.
-----
THIS IS THE SECOND OF TWO PANELS USED TO ADD TRANSACTIONS TO THE CICS/VS PRO-
GRAM CONTROL TABLE (PCT).  ENTER THE REQUESTED INFORMATION INDICATED BELOW, AND
PRESS ENTER.

SECURITY CODE          ==> 1          SECURITY PROTECT TRANSACTION
PRIORITY               ==> 1          DEFINE TRANSACTION PRIORITY
TRANSACTION CLASS     ==> NO         LOGICALLY GROUP TRANSACTIONS
BACKOUT REQUIRED?      ==> NO         YES, NO, OR RESTART
DUMP REQUIRED?         ==> YES        YES, NO, OR FULL
STALL PURGE?         ==> YES        YES OR NO
I/O ERROR PURGE?     ==> YES        YES OR NO
READ TIMEOUT         ==> 0          TIME TO WAIT FOR TERMINAL INPUT
                                BEFORE PURGING TRANSACTION (MMSS)
DEADLOCK TIMEOUT     ==> 0          TIME TO WAIT BEFORE PURGING STALLED
                                TRANSACTION (MMSS)
-----
==>
      EXPLAIN      CANCEL      RETRY

```

Figure 7.3.7. Add Transaction Data Entry Panel

. . . A second Data Entry panel prompts you for the rest of the data. Notice that each data item has a default. Pressing ENTER would cause the defaults shown to be applied to the transaction you are adding. Or, you could override the defaults by keying the correct data before pressing ENTER.

It just so happens that this function, Add Transaction to Program Control Table, requires two data entry panels. Not all functions require two panels. Some require more, some only one. It all depends on the function.

Also, Figure 7.3.7. shows default values for each data item. Not all data entry panels will have a complete set of default values. As previously seen in Figure 7.3.6., only two data items had default values. Other panels of the Interactive Productivity Facility have no default values. Again, it all depends on the function.

Back to the example.

CICS/VS: ADD TRANSACTIONS		TCISAPC2
COMPLETE THE REQUIRED DATA AND PRESS ENTER.		

THIS IS THE SECOND OF TWO PANELS USED TO ADD TRANSACTIONS TO THE CICS/VS PROGRAM CONTROL TABLE (PCT). ENTER THE REQUESTED INFORMATION INDICATED BELOW, AND PRESS ENTER.		
SECURITY CODE	==> 1	SECURITY PROTECT TRANSACTION
PRIORITY	==> 1	DEFINE TRANSACTION PRIORITY
TRANSACTION CLASS	==> NO	LOGICALLY GROUP TRANSACTIONS
BACKOUT REQUIRED?	==> NO	YES, NO, OR RESTART
DUMP REQUIRED?	==> YES	YES, NO, OR FULL
STALL PURGE?	==> YES	YES OR NO
I/O ERROR PURGE?	==> YES	YES OR NO
READ TIMEOUT	==> 0	TIME TO WAIT FOR TERMINAL INPUT BEFORE PURGING TRANSACTION (MMSS)
DEADLOCK TIMEOUT	==> 0	TIME TO WAIT BEFORE PURGING STALLED TRANSACTION (MMSS)

==>		
EXPLAIN	CANCEL	RETRY

Figure 7.3.8. Add Transaction Data Entry Panel

Now, let's assume that you have accepted the defaults for SECURITY CODE, PRIORITY, and TRANSACTION CLASS. However, you are uncertain as to what is meant by "BACKOUT REQUIRED?". So, you light-pen select EXPLAIN.

Now, an Explain Menu is displayed.

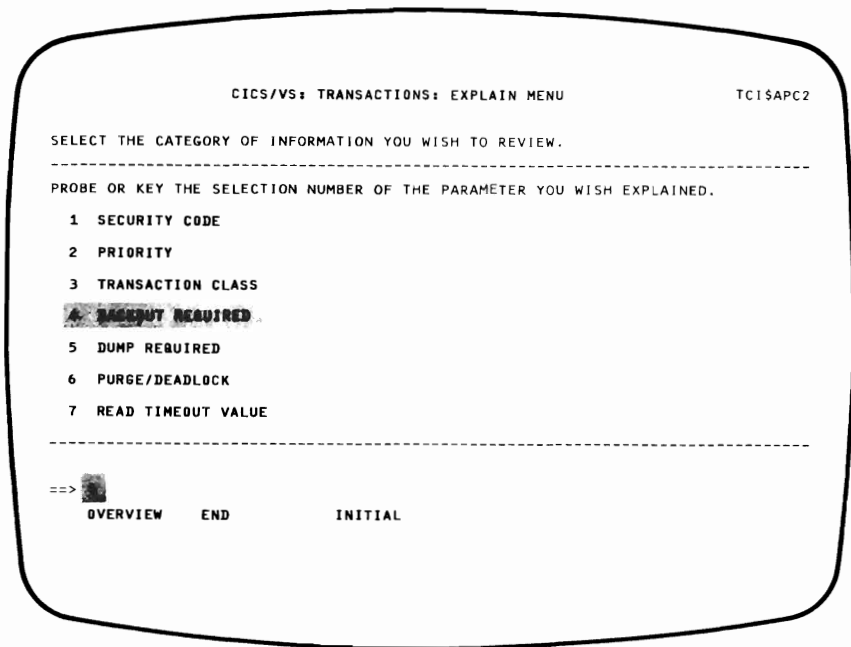


Figure 7.3.9. EXPLAIN Menu

Explain menus allow selection of various Explain information about the previous data entry panel. "BACKOUT REQUIRED" was not fully understood from the previous panel so option 4 is chosen.

An Explain panel describing "BACKOUT REQUIRED" is now displayed.

```

CICS/VS1 TRANSACTIONS: BACKOUT REQUIRED                                TC15APC9
                                                                    PAGE 001/001
END LEAVES EXPLANATION, FORWARD CONTINUES EXPLANATION.
-----
BACKOUT REQUIRED: ENTER YES IF YOU WISH CICS/VS DYNAMIC TRANS-      TO RESTART THE TRANSACTION AFTER
ACTION BACKOUT (DTB) TO BE ACTI-                                     IT HAS BEEN BACKED OUT.
VATED SHOULD THIS TRANSAVTION TER-
MINATE ABNORMALLY. DTB WILL "BACK
OUT" THE EFFECTS OF ANY UPDATES
THE TRANSACTION MAY HAVE MADE TO
YOUR FILES OR DATA BASES. DTB
WILL AFFECT YOUR TRANSACTION'S
RESPONSE TIME, AND WILL USE ADDI-
TIONAL SYSTEM RESOURCES. THERE-
FORE, ONLY USE IT FOR TRANSACTIONS
WHICH MAKE CRITICAL UPDATES TO
YOUR FILES AND DATA BASES.
ENTER RESTART IF YOU WANT CICS/VS
-----
==>
OVERVIEW   END           FORWARD   BACK

```

Figure 7.3.10. EXPLAIN Panel

This Explain panel describes the **BACKOUT REQUIRED** parameter. From this explanation you can see when and when not to use **BACKOUT**. If further explanation is required, you are referred to a specific reference manual.

You could now enter "FORWARD", causing the next Explain panel (if there is one) to be displayed.

Entering **OVERVIEW** would cause the Explain Menu to be redisplayed.

Let's assume that you now understand the **BACKOUT** parameter; you light-pen select **END** and the system returns us to . . .

```

                                CICS/VS: ADD TRANSACTIONS                                TCISAPC2
-----
COMPLETE THE REQUIRED DATA AND PRESS ENTER.
-----
THIS IS THE SECOND OF TWO PANELS USED TO ADD TRANSACTIONS TO THE CICS/VS PRO-
GRAM CONTROL TABLE (PCT).  ENTER THE REQUESTED INFORMATION INDICATED BELOW, AND
PRESS ENTER.

SECURITY CODE          ==> 1          SECURITY PROTECT TRANSACTION
PRIORITY               ==> 1          DEFINE TRANSACTION PRIORITY
TRANSACTION CLASS     ==> NO          LOGICALLY GROUP TRANSACTIONS
BACKOUT REQUIRED?      ==> YES        YES, NO, OR RESTART
DUMP REQUIRED?         ==> YES        YES, NO, OR FULL
STALL PURGE?         ==> YES        YES OR NO
I/O ERROR PURGE?     ==> YES        YES OR NO
READ TIMEOUT          ==> 0          TIME TO WAIT FOR TERMINAL INPUT
                                BEFORE PURGING TRANSACTION (MMSS)
DEADLOCK TIMEOUT      ==> 0          TIME TO WAIT BEFORE PURGING STALLED
                                TRANSACTION (MMSS)
-----
==>
      EXPLAIN      CANCEL      RETRY

```

Figure 7.3.11. Add Transaction Data Entry Panel

... the second data entry panel where you had entered EXPLAIN.

Before continuing with the example, let's discuss the other two options on the command line, RETRY and CANCEL.

If you were to select RETRY at this time, the same panel would be redisplayed without any data you might have keyed. Default values would be as shown on the original panel.

If you were to select CANCEL, all data entered in both data entry panels is ignored and the last menu panel that was displayed (Figure 7.3.5.) is redisplayed.

Let's return to the example.

You will continue to enter data or accept the defaults, then press ENTER.

The next panel displayed is the first data entry panel.


```

                                CICS/VS: ADD TRANSACTIONS                                TCISAPCI
-----
COMPLETE THE REQUIRED DATA AND PRESS ENTER.
-----
THIS IS THE FIRST OF TWO PANELS USED TO ADD NEW TRANSACTIONS TO THE CICS/VS
PROGRAM CONTROL TABLE (PCT).  ENTER THE REQUESTED INFORMATION INDICATED BELOW,
AND PRESS ENTER.  A SECOND PANEL WILL THEN BE DISPLAYED, ASKING YOU FOR FURTHER
INFORMATION.

IF YOU HAVE MORE THAN ONE TRANSACTION TO ADD, THESE PANELS WILL BE RETURNED TO
YOU AS MANY TIMES AS NECESSARY.  AFTER YOU HAVE ADDED ALL YOUR TRANSACTIONS,
TYPE /END IN THE TRANSACTION ID FIELD, AND PRESS ENTER.
-----
TRANSACTION ID      ==>  ____  A UNIQUE FOUR-CHARACTER CODE
PROGRAM NAME        ==> NEWPROG THE NAME OF THE PROGRAM WHICH PRO-
                                CESSES THIS TRANSACTION
WORK AREA SIZE      ==> 256    THE SIZE OF THE WORK AREA REQUIRED BY
                                THIS TRANSACTION, IN BYTES
STORAGE USED        ==> 4K     THE AMOUNT OF DYNAMIC STORAGE USED BY
                                THIS TRANSACTION, IN 2K MULTIPLES
-----
==>
      EXPLAIN      CANCEL      RETRY

```

Figure 7.3.12.

This indicates that all the data has been entered for the first transaction ID. Another transaction ID could be entered now, if required. Our example has only one transaction ID, therefore you have entered all the data required.

Now you are ready to execute the function.

Note the instructions that are highlighted in 7.3.12.

Now you key /END, . . .

```

                                CICS/VS: ADD TRANSACTIONS                                TC1$APC1

COMPLETE THE REQUIRED DATA AND PRESS ENTER.
-----
THIS IS THE FIRST OF TWO PANELS USED TO ADD NEW TRANSACTIONS TO THE CICS/VS
PROGRAM CONTROL TABLE (PCT).  ENTER THE REQUESTED INFORMATION INDICATED BELOW,
AND PRESS ENTER.  A SECOND PANEL WILL THEN BE DISPLAYED, ASKING YOU FOR FURTHER
INFORMATION.

IF YOU HAVE MORE THAN ONE TRANSACTION TO ADD, THESE PANELS WILL BE RETURNED TO
YOU AS MANY TIMES AS NECESSARY.  AFTER YOU HAVE ADDED ALL YOUR TRANSACTIONS,
TYPE /END IN THE TRANSACTION ID FIELD, AND PRESS ENTER.

TRANSACTION ID      ==> /END      A UNIQUE FOUR-CHARACTER CODE
PROGRAM NAME        ==> NEWPROG   THE NAME OF THE PROGRAM WHICH PRO-
                                         CESSES THIS TRANSACTION
WORK AREA SIZE      ==> 256      THE SIZE OF THE WORK AREA REQUIRED BY
                                         THIS TRANSACTION, IN BYTES
STORAGE USED        ==> 4K       THE AMOUNT OF DYNAMIC STORAGE USED BY
                                         THIS TRANSACTION, IN 2K MULTIPLES
-----

==>
    EXPLAIN      CANCEL      RETRY

```

Figure 7.3.13.

... and press Enter.

The next panel, a menu panel, prompts you for the disposition of the Program Control Table additions just entered. You are now informing the Interactive Productivity Facility what you want to do with the data you have just entered.

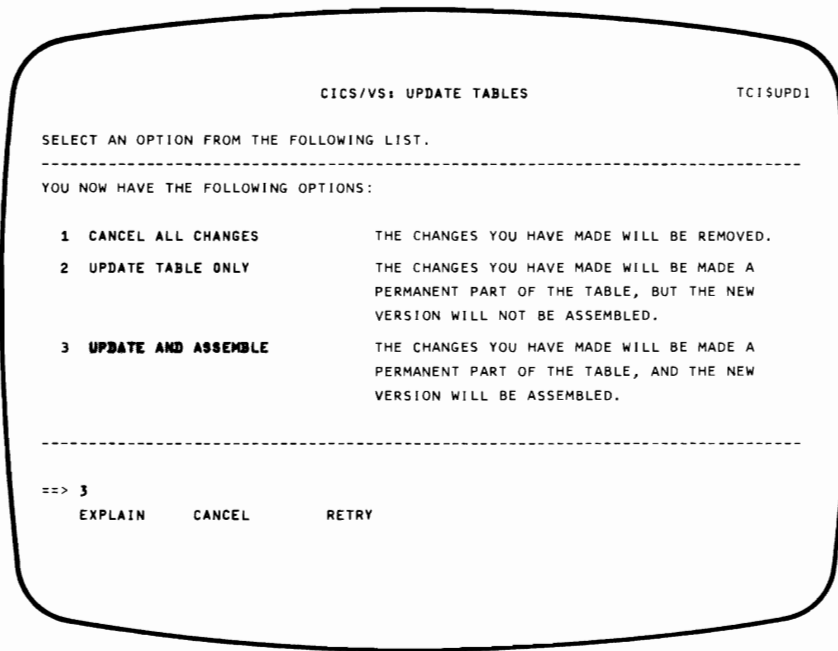


Figure 7.3.14.

Item 3 is selected; the data entered will become a permanent part of the Program Control Table.

Certain DOS/VS jobs must be executed to accomplish the update and assembly of the Program Control Table. So, after item 3 is selected and you press Enter . . .

```

                                JOB DISPOSITION                                SUB$JOB5
-----
COMPLETE THE REQUIRED DATA AND PRESS ENTER.
-----
PCT$ASTT      ENTER THE POWER JOB NAME FOR THE JOB
              TO BE FILED IN YOUR USER AREA.
==>          ENTER THE POWER JOB CLASS FOR THE JOB
              TO BE FILED IN YOUR USER AREA. IF AN
              ASTERISK IS SHOWN, NO CHANGE CAN BE
              MADE.
-----
==>
EXPLAIN      CANCEL      RETRY

```

Figure 7.3.15.

. . . this panel is displayed. You are now prompted for the job name and job class for the DOS/VSE job that will update and assemble the Program Control Table. This job will be filed in your VSE/ICCF user area for later submission to a batch partition.

Note that there are defaults assigned for both job name and job class.

By pressing Enter, you accept the default values and the following panel is displayed:

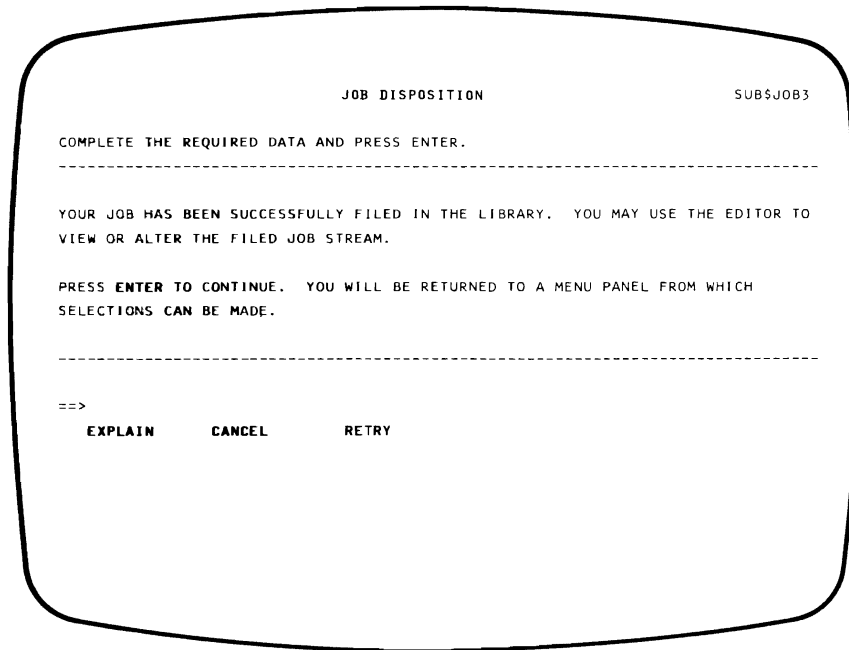


Figure 7.3.16.

This panel informs you of the status of the work just performed.

Note that jobs are not submitted directly to a batch partition, but filed in your library. You can now use the edit capability of VSE/ICCF to view and alter, if necessary, the filed job stream.

As indicated by the arrow, you are given guidance for the next task.

This completes our example.

Before you take the final exercise, let's review some important points learned from our example.

Remember, you have been shown an example of a function being performed using the Interactive Productivity Facility. Many more functions are available and are described in the Interactive Productivity Facility's User's Guide.

The Interactive Productivity Facility presents the user with a series of menu panels. The menu panels offer the user a variety of options from which to choose. If the choice corresponds to a lower level menu panel, that menu panel is displayed. This transition occurred in many places in our example. One such place is shown in Figures 7.3.3. and 7.3.4.

If the choice corresponds to a function, that function is invoked. Figure 7.3.6., in our example, is the point in which the function of adding a transaction to the Program Control Table was invoked.

Explain panels are displayed only when requested by the user and are available for most menu and data entry panels.

Now, complete the exercise. If you miss any of these questions, review the appropriate material in the text.

Exercise 7.3

1. (True or False) After the user signs-on to the Interactive Productivity Facility, a menu panel is displayed.
2. (True or False) Explain panels are displayed only at the request of the user.
3. (True or False) Data entry panels are displayed only at the request of the user.
4. (True or False) After data entry has been completed for a function, DOS/VSE job streams are created and submitted automatically to a DOS/VSE batch partition for execution.

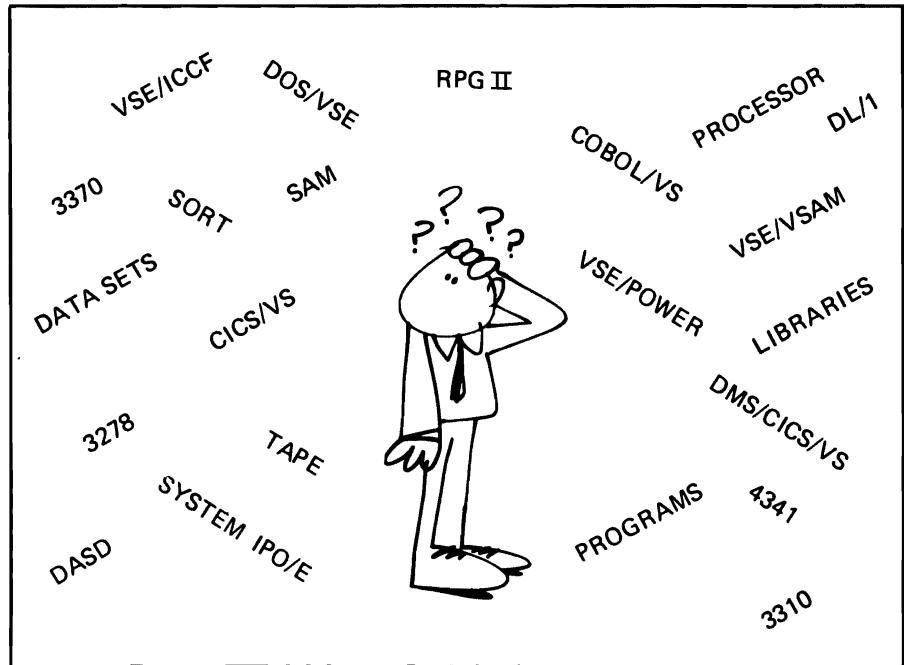
Solution 7.3

1. True.
2. True.
3. False. Data Entry panels are displayed corresponding to the normal hierarchical panel structure. These panels are normally displayed when a user response corresponds to a function.
4. False. DOS/VSE job streams are created after you have been prompted for the job name and job class. Also, they are not automatically submitted to a DOS/VSE partition for execution. The jobs are filed in your VSE/ICCF area.

Unit 8:

The Data Processing Installation

Introduction At this point you have learned about the components that go into making a modern data processing installation. In fact, with so many possible choices that can go into making the whole, you might feel like our friend below.



Sometimes it is truly difficult to *see the forest for the trees*. In this unit we are going to step back and view the *forest*.

Objectives At the completion of this Unit, you should be able to:

- Understand the four major functions of hardware.
- Understand the four general categories of software.
- Relate the material presented in the previous chapters to the data processing installation as a whole.

Average Study Time 30 to 45 Minutes

Topic 1. Perspective on Hardware

Let's return to basics for a moment and review some of the necessary items in a data processing installation. The major categories are hardware and software. We can break hardware down into 4 major functional components.

1. Processor

Hardware that executes instructions and processes data, for example a 4341 or 4331.

2. I/O Device

Hardware that reads, writes and stores data, for example a 3310 disk or 8809 tape drive.

3. Control Unit

Hardware that controls the I/O devices, for example the 3274 control unit for terminals or the display/prINTER adapters on the 4331.

4. Data Path

Hardware that transfers data from the I/O devices to main or processor storage where it can be processed, for example the byte or block multiplexor channels on the 4331 or the 4341.

Not every piece of hardware serves only one function. Some such as the IBM 3276 terminal perform two functions. This device has an integrated control unit so it performs both the I/O Device function (#2) and the Control Unit function (#3).

An I/O adapter on the 4331 takes the place of both the channel and the control unit, but only for a limited type and number of devices. It therefore, performs both the Control Unit function (#3) and the Data Path function (#4).

We can picture these four functions by showing the typical flow of data being processed on a disk as in Figure 8.1.1.

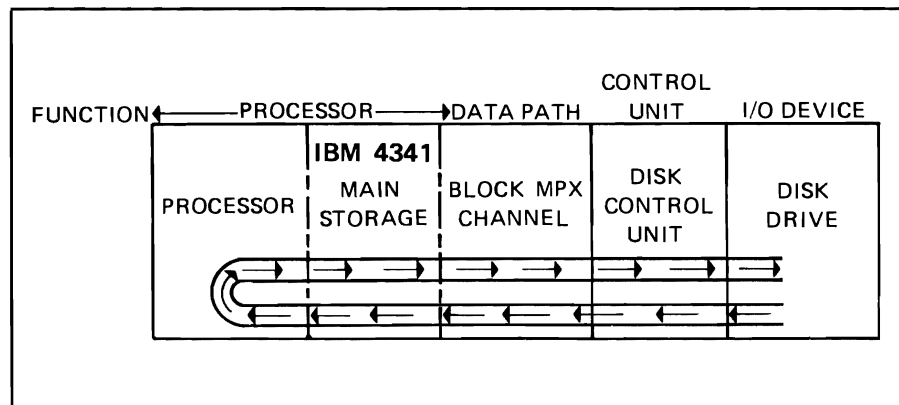


Figure 8.1.1 Flow of data being processed on disk.

A similar data flow is also true if the user is communicating with the system through a terminal that is locally attached (i.e. not over a communication line). See Figure 8.1.2.

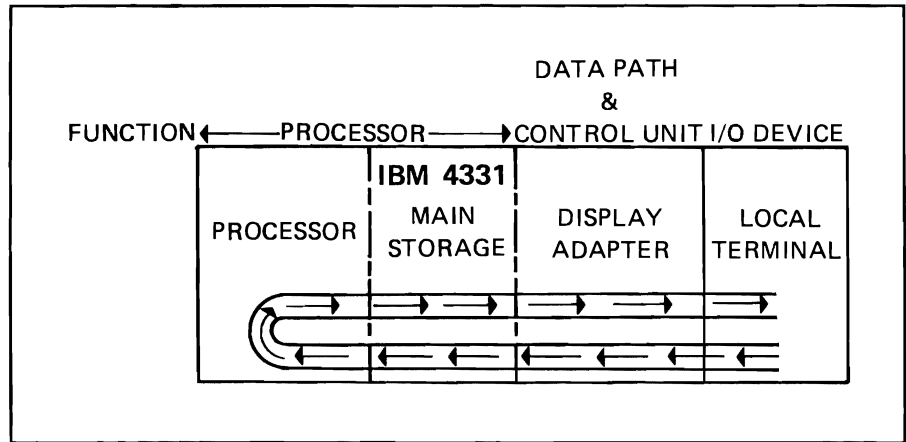


Figure 8.1.2 Flow of data being processed from a local terminal.

When remote terminals are added to the system new hardware which forms the remote data path is introduced. This new hardware controls the data as it passes over the communication line (e.g. telephone line). An example of remote data flow might consist of the communications adapter in a 4331 with a modem at either end of a telephone line. The same data transfer as pictured in Figure 8.1.2 might look like Figure 8.1.3 if the communications were with a remote rather than a local terminal.

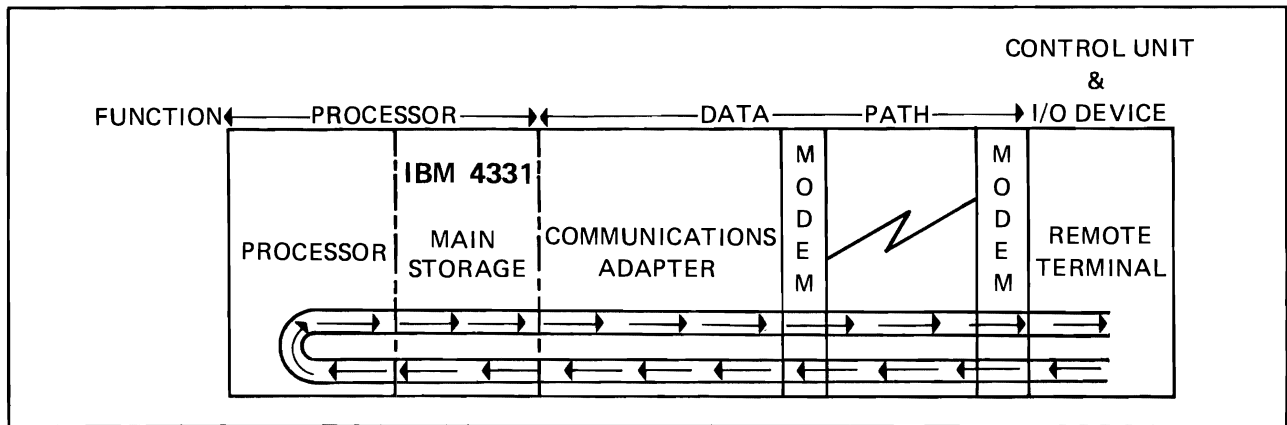


Figure 8.1.3 Flow of data being processed from a remote terminal.

Hardware is the fundamental requirement for a data processing installation. It might be thought of as the foundation upon which all the other components are based as shown in Figure 8.1.4.

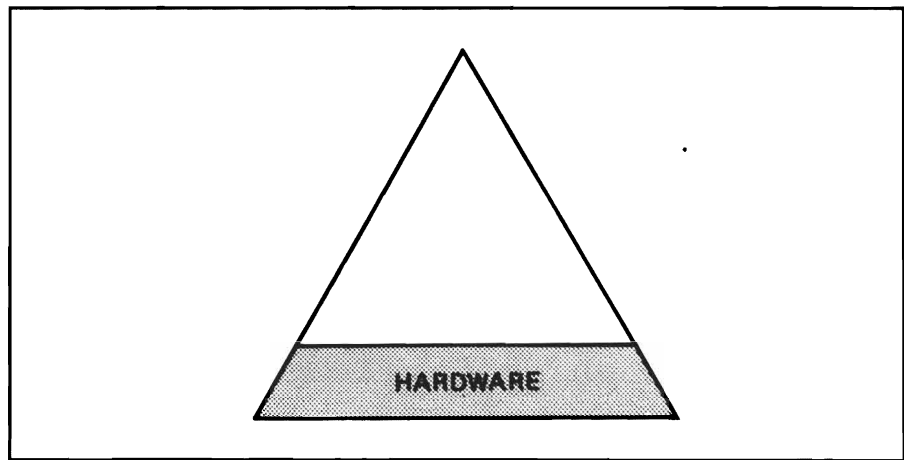


Figure 8.1.4 Hardware

Hardware can be compared to a car without a driver. A great problem solver but it needs direction and guidance.

In data processing, software is the driver.

Topic 2. Perspective on Software

Software can be divided into four general categories as follows:

1. Basic
2. Productivity
3. Installation Applications
4. Service and Utility

Basic software is IBM supplied software that is fundamental to the operation of the system. This code acts as a liaison between other programs and the hardware by actually driving the hardware and controlling the execution of all programs. It is generally known as the "operating system".

Examples are:

DOS/VSE

All access methods (VSE/VSAM, ACF/VTAME, ACF/VTAM, BTAM-ES, SAM)

VSE/POWER

If hardware is the foundation of a data processing installation then basic software is the first building block. See Figure 8.2.1.

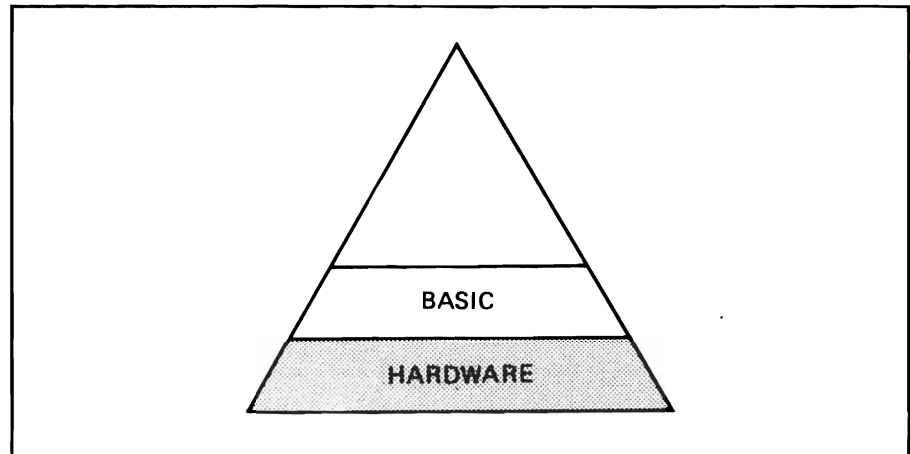


Figure 8.2.1 Hardware and Basic Software

Productivity software consists of elements that generally fit between the basic software and the user programs to increase productivity by reducing complexity, providing a high level of function, and easing maintenance.

Examples are:

CICS/VS

DL/1

DMS/CICS/VS

VSE/ICCF

See Figure 8.2.2.

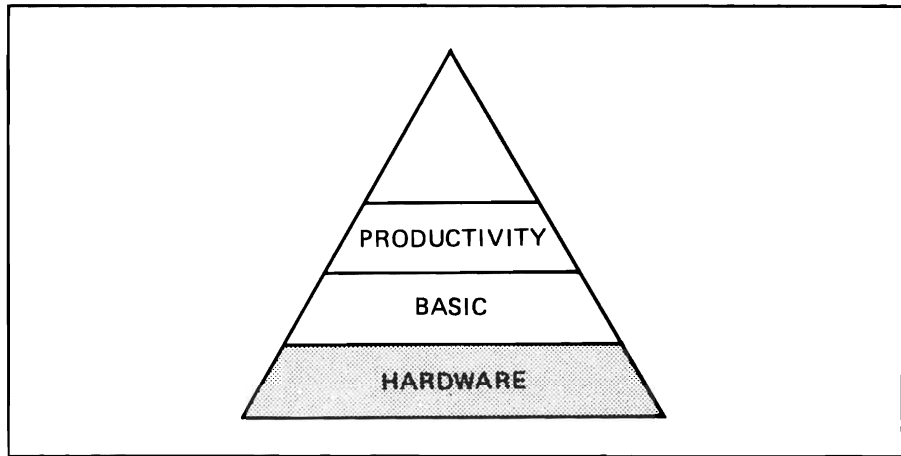


Figure 8.2.2 Hardware, Basic Software and Productivity Software

Installation applications are generally written by the data processing installation's programmers or may be Program Products (PP's), Field Developed Programs (FDP's), or Installed User Programs (IUP's) that have been purchased to perform specific tasks in that installation.

Examples are:

A payroll system

An inventory system

This gives the relationship as shown in Figure 8.2.3.

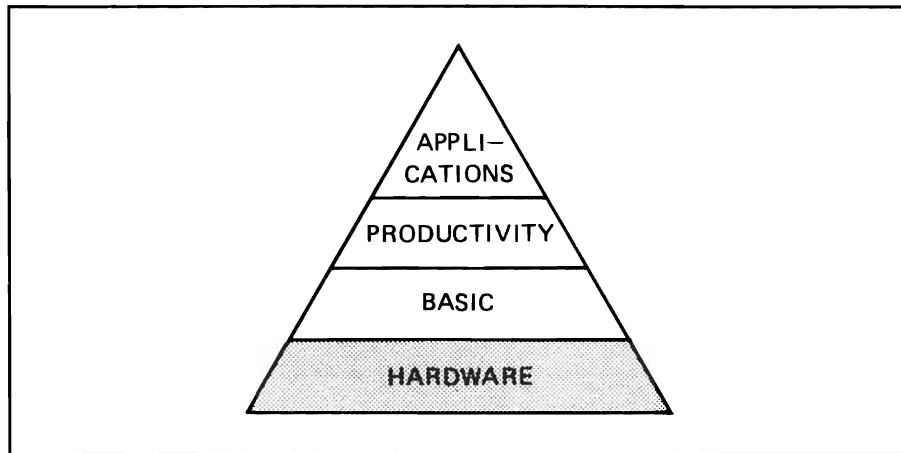


Figure 8.2.3 Hardware, Basic Software, Productivity Software and Installation Applications

Service and utility software consists of programs that are used to support the system functions, apply maintenance and perform everyday tasks. There are system service and utility programs and user service and utility programs.

Examples are:

- System

MSHP

VSE/IPCS

Interactive Productivity Facility

Language translators such as COBOL/VS, RPG II, and PL/I

- User

DOS/VS Sort

VSE/DITTO

These programs might be depicted as shown in Figure 8.2.4.

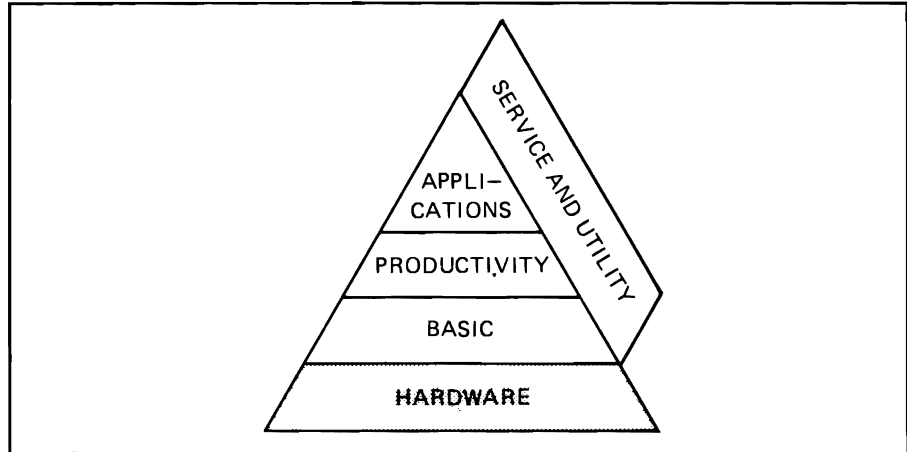


Figure 8.2.4 Hardware, Basic Software, Productivity Software, Installation Applications, and Service and Utility Software

This hierarchy of software works together to achieve the result expected by the data processing installation. With DOS/VSE and VSE/AF, for example, the installation can have from one to twelve partitions active at any one time, each performing its separate function.

The basic software is always present. Service and utility software may or may not be in use at any given time. If the user application is currently active and it uses productivity software such as CICS/VS, then both must be present. There may, of course, be more than one user application active.

The specific hardware and software used in your installation will be some combination of the individual pieces as presented in the previous chapters. Working together they will fulfill the data processing requirement for your installation.

Acronyms

ACF	Advanced Communication Function
BTAM-ES	Basic Telecommunications Access Method-Extended Support
DOS/VSE	Disk Operating System/Virtual Storage Extended
DAT	Dynamic Address Translation
ECPS	Extended Control Program Support
FBA	Fixed Block Architecture
ICCF	Interactive Computing & Control Facility
IPCS	Interactive Problem Control System
NCP	Network Control Program
SE	System Extensions
SYSTEM IPO/E	SYSTEM Installation Productivity Option/Extended
VSAM	Virtual Storage Access Method
VSE/AF	Virtual Storage Extended/Advanced Functions
VTAM	Virtual Telecommunications Access Method
VTAME	Virtual Telecommunications Access Method Extended
3203-5	Printer for 4331 and 4341 - Up to 1200 LPM
3262	Printer for 4331 - Up to 650 LPM
3278-2A	Display Console for 4331 and 4341
3289-4	Printer for 4331 - Up to 400 LPM
3310	Direct Access Storage Device for 4331 at 64.5 MB
3370	Direct Access Storage Device for 4331 & 4341 at 571.3 MB
3880	Storage Control for 3340/3344 and 3370 on 4341
4300	IBM 4300 Processors
4331	IBM 4331 Processor
4341	IBM 4341 Processor
5424	96 column card reader, punch and printer for the 4331

Glossary

access method: A technique for moving data between programs in virtual storage and input/output devices.

address: (1) The storage location of an instruction or data. (2) The part of an instruction that specified the location of an operand for the instruction.

address translation: The process of changing the address of an item of data or an instruction from its virtual address to its real storage address. See also dynamic address translation.

alternate library: An additional ICCF private library to be associated with a specific ICCF user. A user may have up to eight alternate libraries which are specified in his user profile.

application: Job(s) or function(s) to be performed or accomplished by utilizing data processing equipment and techniques.

application program: A program written by or for a user that applies to his own work.

application programmer: A programmer who designs, writes and tests programs that satisfy specific needs of a particular area of a business.

assembler language: A source language that includes symbolic machine language statements in which there is a one-to-one correspondence with the instruction formats and data formats of the computer.

asynchronous execution: The process of temporarily disconnecting an ICCF user's terminal from a job executing in an interactive partition so that the user can enter another mode of operation to perform additional terminal work.

attribute: A characteristic of a unit of data such as length, value, or method of representation.

audit trail: An accumulation of all the occurrences of a specific activity to be used for the verification and review of that activity. For example, all the accesses to a restricted resource can be accumulated for future review.

auxiliary storage: Data storage other than real storage; for example, storage on magnetic tape or disk. Synonymous with external storage, secondary storage.

BASIC: Beginner's All-purpose Symbolic Instruction Code. A programming language with a small repertoire of commands and a simple syntax, primarily designed for numerical applications.

batch: An accumulation of data to be processed.

batch computing: The processing of data or the accomplishment of jobs accumulated in advance in such a manner that each accumulation thus formed is processed or accomplished in the same computer run.

batch job: A job that is grouped with other jobs as input to a computer system.

batch partition: In DOS/VSE, a contiguous area of virtual storage associated with the execution of batch programs and jobs.

blocking: Combining two or more logical records into one block.

blocking factor: The number of logical records combined into one physical record or block.

book: A group of source statements written in any of the languages supported by DOS/VSE and stored in a source statement library.

buffer: An area of storage that is temporarily reserved for use in performing an input/output operation, into which data is read or from which data is written. Synonymous with I/O area.

byte: A sequence of eight adjacent binary digits that are operated upon as a unit and that constitute the smallest addressable unit of the system.

card image: A one-to-one representation of the hole patterns of a punched card, for example, a matrix in which a 1 represents a punch and a 0 represents the absence of a punch.

card image file: A file containing 80 character records.

card punch: A device to record information in cards by punching holes in the cards to represent letters, digits, and special characters.

card reader: A device which senses and translates into machine code the holes in punched cards.

central processing unit: A hardware unit of a data processing system that includes the circuits controlling the interpretation and execution of instructions. Abbreviated CPU.

channel: A hardware device that connects the CPU and real storage with the I/O control units.

child: A segment of a DL/I hierarchical structure which has a segment immediately above it in the structure. A segment which is dependent on another.

CICS/DOS/VS: See Customer Information Control System/Virtual Storage.

COBOL: COmmon Business Oriented Language. An English-like programming language designed for business data processing applications.

code: The representation of data in a symbolic form that can be accepted by a computer.

command: An instruction.

command area: The area on the screen of a display device reserved for the entering of ICCF editor commands.

command language: The commands and control statements that allow ICCF terminal users to communicate with ICCF and perform requested functions.

command mode: The controlling mode of operation of ICCF from which all other modes are entered.

common data: A member that appears only once within the ICCF library file but has a directory entry in every user's library. All ICCF users have read access to common data.

common library: An ICCF library containing members which are generalized routines or procedures that all users may access in read only mode. There is only one COMMON library per ICCF library file.

compile: To translate a computer program expressed in a problem-oriented language into a computer-oriented language.

compiler: A program that translates high-level language statements into machine language instructions.

component: A functional part of an operating system, program product or Field Developed Program.

configuration: The group of machines, devices, etc., which make up a data processing system.

context editor: A general purpose text manipulation program that enables an ICCF user to create and modify card image files from the terminal.

context editor commands: ICCF instruction that relate to the scanning and manipulation of data in a user's Input Area or library.

control program: A program that is designed to schedule and supervise the performance of data processing work by a computing system.

control statement: A statement which requests the performance of a specific function.

control unit: A device that controls the reading, writing, or display of data at one or more input/output devices.

core image library: A library of phases that have been produced as output from link-editing. The phases in the core image library are in a format that can be loaded into processor storage for execution.

count-key-data (CKD) device: A disk storage device storing data in the format: count field normally followed by a key field followed by the actual data of a record. The count field contains, among others, the address of the record in the format CCHHR (CC = cylinder number, HH = head or track number, R - record number) and the length of the data; the key area contains the record's key (search argument).

CPU: See Central Processing Unit

CPU busy time: The amount of time devoted by the central processing unit to the execution of instructions.

CPU time: See CPU busy time.

cursor: A movable spot of light on the screen of a display device, usually indicating where the next character will be entered.

Customer Information Control System/Virtual Storage: An IBM program product that enables transactions entered at remote terminals to be processed concurrently by user-written application programs.

DASD: Direct Access Storage Device.

data: A representation of facts, concepts, or instructions in a formalized manner suitable for interpretation or processing by human or automated means.

data availability: The existence of data storage capability and techniques allowing access of the data as needed.

data base: A collection of interrelated data elements processable by one or more applications.

data base administration: The responsibility within an installation for technically supporting the use of DL/I.

data base definition: A block of data stored in a Core Image Library which defines the hierarchical structure, physical organization, segment and field names, access methods, and all other physical characteristics of a data base.

data base manager: A software program such as DL/I DOS/VS which provides methods of organizing, accessing, and manipulating data. It is the interface between a data base and an application program.

data base record: A root segment and all its dependent segments which is all the data associated with a particular primary key.

data communications manager: A software program such as CICS/DOS/VS which provides the control of terminal devices, online application programs and resource usage necessary to maintain online processing. It is the interface between the online application program and the communication network.

data confidentiality: See data security.

data content: The actual data or data value assigned to a specific data entity. For example, a content of a data field called Salary might be 1000.00.

data description: The classification of a data value by a name or an explanation of its use. For example, the data 1000.00 could be given the description of the amount paid to an employee expressed in dollars and the name Salary.

data dictionary: A central repository of information about the data existing in an organization.

data display area: The area of the ICCF full screen editor screen layout where the individual data records of a file or library member are displayed.

data file: See data set.

data independence: The ability to access data from an application program without regard for its physical structure, organization, or location.

data integrity: Preservation of data for its intended purpose.

data management: A major function of DOS/VSE that involves organizing, storing, locating, retrieving, and maintaining data.

data recovery: The ability to determine the status at time of failure and to re-establish the data values so that processing can be resumed at that point without irreparable loss of data or of the results of processing.

data security: Prevention of unauthorized disclosure, modification, destruction, or access of data.

data set: The major unit of data storage and retrieval in an operating system, consisting of a collection of related data records arranged in one of several prescribed ways.

data set maintenance: The adding, deleting, or changing of data to reflect the effects of changes.

data systems environment: The capture, analysis, and effective use of reliable information required by authorized individuals to operate and manage an enterprise.

data transmission: The transferring of data from one place for reception elsewhere by signals over a channel.

DBD: See Data Base Definition.

DB/DC: Data Base/Data Communication.

deblocking: The action of making the first and each subsequent logical record of a block available for processing one record at a time.

debugging: The process of finding errors in a computer program, correcting the errors and retesting the program.

definition: A statement that describes the features of specific relationships of, or establishes context of data. In a data dictionary, a description of the subjects.

dependent segment: All segments of a data base record other than the root.

device independence: The ability to perform I/O operations from a program without regard to the physical characteristics and operation of the sending or receiving device.

diagnostic routine: A program that facilitates computer maintenance by detection and isolation of malfunctions or mistakes.

direct access: Retrieval or storage of data by a reference to its location on a volume, other than relative to the previously retrieved or stored data.

direct organization: Direct file organization implies that for purposes of storage and retrieval there is a direct relationship between the contents of the records and their addresses on disk storage.

directory: An index that is used to locate programs that are stored on direct access storage.

diskette: A flexible magnetic-oxide coated disk suitable for data storage and retrieval.

disk log: A data set containing information on user access to restricted resources as defined in ICCF tables. This log is maintained by the VSE/Access Control-Logging and Reporting program product and ICCF.

disk pack: A direct access storage volume containing magnetic disks on which data is stored. Disk packs are mounted on a disk storage drive.

display device: An output unit that gives a visual representation of data on a cathode ray tube.

DOS/VS: Disk Operating System/Virtual Storage.

DOS/VSE: Disk Operating System/Virtual Storage Extended.

DOS/VS Interactive Debug Facility: An IBM Field Developed Program designed to allow the ICCF user to debug a program interactively from his terminal. This program allows the user to look at application program errors, modify storage and continue execution until the program is working properly.

DTB: Dynamic Transaction Backout.

dump: (1) To copy the contents of all or part of virtual storage. (2) The data resulting from the process as in (1).

dump commands: Instructions to ICCF that enable a user to display information from a program that has abnormally terminated.

dynamic address translation (DAT): (1) The change of a virtual storage address to an address in real storage during execution of an instruction. (2) A hardware function that performs the translation.

dynamic space allocation: A feature of ICCF which provides for the dynamic allocation of user work files to jobs executing in interactive partitions. This feature eliminates the need for predefining individual user work files.

EDF: Execution Diagnostic Facility of CICS/VS.

edit: To prepare data for later operation. Editing includes the addition modification, rearrangement and deletion of data.

edit mode: The ICCF mode of operating enabling a user to locate and modify data by text or content. In edit mode new data may be entered as well as existing data being modified, rearranged or deleted.

end user: The ultimate source or destination of information flowing through a system. In the context of ICCF, an end user is any terminal user.

entry sequence: The order in which data records are physically arranged in auxiliary storage, without respect to their contents (contrast with key sequence).

entry-sequenced file: A VSAM file whose records are loaded without respect to their contents, and whose relative byte addresses cannot change. Records are retrieved and stored by addressed access, and new records are added to the end of the file.

error message: The communication that an error has been detected.

error recovery procedures: Procedures designed to help isolate, and, when possible, to recover from errors in equipment. The procedures are often used in conjunction with programs that record the statistics of machine malfunctions.

execution mode: The ICCF mode of operation in effect whenever a job is being executed in an interactive partition or when a job is queued for execution. This mode remains in effect until the job ends.

extent: A continuous space on a direct access storage device, occupied by or reserved for a particular file.

FBA: See fixed block architecture (FBA) device.

field: A specified area of contiguous characters in a record used for a particular category of data.

Field Developed Program (FDP): A licensed program normally developed by IBM branch office personnel to perform end-use, productivity and transitional functions for the user.

file: A collection of records treated as a unit. See data set.

fixed block architecture (FBA) device: A disk storage device storing data in blocks of fixed size; these blocks are addressed by block number relative to the beginning of the file.

FORTTRAN: FORMula TRANslation. A programming language primarily used to express computer programs.

full screen editor: The ICCF editor designed to use the full range of 3270 features such as user defined program function key options and minimum data transmission (only changed data is transmitted, not the entire screen).

full screen editor commands: Instructions to the ICCF full screen editor that control the content of the screen; locate areas anywhere within a file and change or add data within the current line; and edit individual records currently displayed on the screen.

general register: A register used for operations such as binary addition, subtraction, multiplication and division. General registers are used primarily to compute and modify addresses in a program.

global change: A change affecting all the occurrences of a specific character string within a file or library member. The ICCF context editor provides the facility for making global changes.

hard copy: A printed copy of machine output in a visually readable form, for example, printed reports, listings, documents, and summaries.

hard copy device: An output device providing a permanent copy of a display image that can be separated from a display device and recorded on paper.

hardware: The physical equipment that makes up the data processing system. For example, mechanical, magnetic, electrical, or electronic devices. Contrast with software.

hierarchical path: See path.

hierarchical structure: A data structure consisting of sets and subsets of data such that every subset of a set is of lower rank than the data of the set.

ICCF: See Interactive Computing and Control Facility.

ICCF command processors: ICCF programs which read and interpret commands (instructions from ICCF) and data from the terminal and perform requested functions.

ICCF control program: The system traffic cop supervising the activities of all ICCF components, except terminal I/O operations.

ICCF library file: A direct access data set which is composed of a system record, user profile records, user spool areas and the ICCF libraries. These libraries, include one COMMON library and a number of PUBLIC and PRIVATE libraries.

identification: A group of characters used to identify or name a specific user of ICCF.

index: A table made up of keys and corresponding disk addresses used to locate the records of a file.

indexed-sequential organization: The records of an indexed sequential file are arranged in logical sequence by key. Indexes to these keys permit direct access to individual records. All or part of the file can be processed sequentially.

information: The meaning assigned to data after processing by a computer or other conventions.

Initial Program Load (IPL): The initialization procedure that causes DOS/VSE to commence operation.

input area: A temporary disk storage area within the ICCF library file into which keyed-in data may be placed for modification.

input mode: The ICCF mode of operation enabling a user to enter programs or data.

input queue: In DOS/VSE a waiting list of job definitions on

a direct access storage device managed by VSE/POWER.

instruction: In a programming language, a meaningful expression that specifies one operation and identifies its operands, if any.

integrity: Preservation of data or programs for their intended purpose.

interactive computing: The process of an individual directly utilizing a computer via a terminal. This process permits frequent interchange between the terminal user and the computer during the execution of a program; generally, each entry from the terminal elicits a response from the computer, and vice versa.

Interactive Computing and Control Facility: An IBM program product for use at DOS/VSE installations as the vehicle for interactive computing with ICCF, the DOS/VSE user can perform application program development, personal computing and problem solving and system programming functions in an interactive environment.

interactive environment: A data processing environment in which there is continuous dialog between the user and the computing system.

interactive partition: A block of virtual storage within a DOS/VSE partition where ICCF terminal users' jobs can be executed.

interactive program development: The process of creating new computer programs directly from a terminal.

interactive usability aid: An application program executing in an ICCF interactive partition that works conversationally with the ICCF user via messages to the display screen. An interactive usability aid assists the user in preparing and submitting jobs for execution in ICCF and DOS/VSE. Also referred to as a prompter.

I/O: Used to mean input/output.

ISAM interface program: A set of routines that allow a processing program coded to use ISAM to gain access to a VSAM key-sequenced file with an index.

job: (1) One or more tasks designated as a unit of work for a computer. (2) A collection of related programs, identified in the input stream by a JOB statement followed by one or more EXEC statements.

job control: A program that is called into a partition to prepare each job or job step to be run. Some of its functions are to assign I/O devices to certain symbolic names, log (or print) job control statements, and fetch the first program phase of each job step.

job control statement: A statement in a job that is used in identifying the job or describing its requirements to an operating system.

job entry statements: ICCF commands that direct and control the execution of jobs.

job step: The execution of a single processing program.

job stream: The sequence of job control statements and data submitted to an operating system.

K: When referring to storage capacity, 1024 bytes.

key: One or more characters included in the data and used to identify it or control its use.

keyboard: A systematic arrangement of keys by which data is entered.

keypunch: The process of punching holes in a data carrier (card) from a keyboard activated punch.

key sequence: The collating sequence of data records, determined by the value of the key field in each of the data records. May be the same as, or different from the entry sequence of the records.

key-sequenced file: A file whose records are loaded in key sequence and controlled by an index.

label: Identification record for a tape, diskette, or disk file.

language translator: A general term for any assembler, compiler, or other routine that accepts statements in one language and procedures equivalent statements in another language.

leased facility: A circuit of the public telephone network made available for the exclusive use of one subscriber.

librarian: The set of programs that maintains, services, and organizes the system and private libraries.

library: A collection of files or programs, each element of which has a unique name, that are related by some common characteristics. For example, all phases in the core image library have been processed by the linkage editor.

line pointer: The pointer that is associated with each ICCF file displayed on the screen while the user is in edit mode. This pointer refers to a line in the file considered to be the current line. The current line is defined as the line that is being created or edited in the file.

linkage editor: A processing program that prepares the output of language translators for execution. It combines separately produced object modules and produces executable code (a phase) that is ready to be fetched or loaded into virtual storage.

list mode: The ICCF mode of operation in effect while a display of data to the terminal is in progress.

load: In DOS/VSE, to bring a program phase from a core image library into virtual storage for execution.

log area: A temporary disk storage area within the ICCF library file used to store terminal input and output for later display at the terminal.

logical relationship: A connection between segments in different data bases allowing users of one data base access to the data in another.

logging: The process within ICCF which causes the input/output from and to a user's terminal to be stored in the user's Log Area for later display at his terminal.

logging on: The process of initiating a terminal session with an interactive computing system. This is accomplished with ICCF by supplying an identification and password.

logoff: The procedure by which a user ends a terminal session.

logon: The procedure by which a user begins a terminal session.

macro: An ICCF library member which contains a sequence of ICCF statements that together perform frequently used functions, such as the storing of object programs.

main library: An ICCF private or public library that an ICCF user has immediate read/write access to after logging on to ICCF. A user's main library is specified in his user profile record.

member: A single file of data in an ICCF library. This file can be a source program, an object program, a job stream or a procedure.

MPS: Multiple partition support provides a centralized data base facility to permit multiple applications to access DL/I data bases concurrently.

multiprogramming system: A system that controls more than one program simultaneously by interleaving their execution.

multitasking: The concurrent execution of one main task and one or more subtasks in the same partition.

object code: Output from a compiler or assembler which is suitable for processing by the linkage editor to produce executable machine code.

object module: The output of an assembler or compiler that is input to a linkage editor.

object program: A fully compiled or assembled program. Contrast with source program.

online: (1) Computer hardware controlled by the central processing unit. (2) The ability of a user to interact with a computer process.

online processing: The operation of a computer to process data directly from the point of origin and to transmit results directly to where they are used while under the continual control of the computer.

operation: A well-defined action that, when applied to any permissible combination of known entities, produces a new entity. A program step undertaken or executed by a computer, for example, addition, multiplication, extraction, comparison, shift, and transfer.

operator command: A statement to the control program, issued via a console device, which causes the control program to provide requested information, alter normal operations, initiate new operations, or terminate existing operations.

operator message: A message from the operating system or a problem program directing the operator to perform a specific function, such as mounting a tape reel, or informing him of specific conditions within the system, such as an error condition.

organization method: In a data base, the technique used to arrange the data on the physical device. DL/I DOS/VS has two organization methods, Hierarchical Sequential and Hierarchical Direct.

output queue: In DOS/VSE a waiting list of control information describing system output data sets, that specifies the location and disposition of system output. This queue is managed by VSE/POWER.

page: (1) In DOS/VSE, a 2K block of instructions, data or both. (2) To transfer instructions, data, or both between processor storage and the page data set.

page data set: An extent in auxiliary storage, in which pages are stored.

page frame: A 2K block of processor storage that can contain a page.

page in: The process of transferring a page from the page data set to processor storage.

page out: The process of transferring a page from processor storage to the page data set.

page pool: The set of all page frames that may contain pages of programs in virtual mode.

paging: The process of transferring pages between processor storage and the page data set.

panel: In DMS/CICS/VS, the format or display image which is transmitted to a video terminal. It includes the definitions of data fields displayed and those entered by the terminal user.

parent: A segment of a DL/I hierarchical structure which has a segment immediately below it in the structure. A segment which has a dependent segment.

partition: In DOS/VSE, a contiguous area of virtual storage available for the execution of programs.

password: A unique string of characters that a program, computer operator or user must supply to meet security requirements before gaining access to data.

path: The chain of segments in a data base record hierarchy that leads to a specific segment. It contains only one segment occurrence for each level from the root segment to the segment being addressed.

PCB: See Program Communication Block.

peripheral equipment: A term used to refer to card devices, magnetic tape and disk devices, diskettes, printers, and other equipment bearing a similar relation to the CPU.

permanent file: A file that is retained from one day until the next or from one initial program load until the next.

phase: The smallest complete unit that can be referred to in the core image library.

PI: See Program Isolation.

PL/I: Programming Language/I. A programming language designed for use in a wide range of commercial and scientific computer applications.

POWER: A unit record spooling support available as the IBM licensed program VSE/POWER.

print area: A temporary disk storage area within the ICCF library file used to temporarily hold printed output until it is displayed at the terminal.

printer: A device that expresses coded characters as hard copy.

priority: A rank assigned to a partition that determines its precedence in receiving CPU time.

priority processing: In a multi-task environment, processing based on assigned status or ranking. A job, online task, or other processing component given precedence over other components when contending for a resource.

private data: A library member that may be read by any user, who shares that ICCF library, but may be modified only by the user who entered the data.

private library: An ICCF library containing members that are unique to a specific user. The user has the ability to read and update this library.

problem program: Any program that does not contain privileged instructions. This includes IBM-distributed programs, such as language translators and service programs, as well as programs written by a user.

procedure: An ICCF library member which contains a sequence of ICCF statements that together perform frequently used functions, such as the compilation, loading and execution of programs.

processing program: (1) A general term for any program that is not a control program. (2) Synonymous with problem program.

processor storage: The general purpose storage of a computer. Processor storage can be accessed directly by the operating registers. Synonymous with real storage.

program: A series of actions designed to achieve a certain result.

program development: The process of creating new computer programs.

program function key: A key on the keyboard of a display device that passes a signal to a program to call for a particular program operation.

program integrity: Preservation of programs for their intended purpose.

Program Isolation (PI): A facility of DL/I which allows concurrent updates on different data base segments of the same type without destroying the validity of the data.

program maintenance: The process of modifying existing computer programs.

programmer: A person who designs, writes and tests computer programs.

program product: An IBM licensed program developed to perform end-use functions for the user and usually interfaces with and relies upon system control programming or a currently available control program. A program product contains logic directly related to the user's data, and is directly usable or adaptable to meet his specific requirements.

Program Specification Block: A block of data stored in a Core Image Library which specifies for an application program the data bases to be used, type of data used, and the allowable operations on each data base.

prompter: See interactive usability aid.

protected resource: See restricted resource.

PSB: See program specification block.

public data: A library member that may be read and modified by any user who shares that ICCF library.

public library: An ICCF library containing members which are accessible to all users on a read/write basis.

Punch Area: A temporary disk storage area within the ICCF library file used to temporarily store punched output from jobs run in interactive partitions.

queue: A waiting line or list formed by items in a system waiting for service.

random processing: The treatment of data without respect to its location in auxiliary storage, and in an arbitrary sequence governed by the input against which it is to be processed.

real address: The address of a location in real storage.

real mode: In DOS/VSE, the mode of a program that cannot be paged.

real storage: The storage of a computing system from which the central processing unit can directly obtain instructions and data, and to which it can directly return results. Synonymous with processor storage.

redundant data: Identical data sorted in more than one place such as in two different data sets.

reenterable: The attribute of a load module that allows the same copy of the load module to be used concurrently by two or more tasks.

relationships: In a data dictionary, the interaction between subjects.

relocatable library: A library of relocatable object modules and IOCS modules required by various compilers. It allows the user to keep frequently used modules available for combination with other modules without recompilation.

restricted resource: Any system resource that is available only to authorized users.

root segment: The first segment in a data base record. It is at the highest hierarchical level of the structure.

routine: A set of instructions that usually has some general or frequent use.

RPG II: Report Program Generator, version 2. A commercially oriented programming language specifically designed for writing application programs that meet common business data processing requirements.

scale line: The line on the ICCF full screen editor screen layout that indicates the column number for easy reference and provides the name of the file or library member being displayed in the data display area and its file type.

screen: The display surface of a display device.

secondary indexing: A method of accessing data base records based on data in the records other than the primary key field.

security: Prevention of access to or use of data or programs without authorization.

segment: A group of related data fields within a data base record that is the basic unit of information passed between an application program and a data base.

segment occurrence: The specific existence of data for the fields defined in a segment type. A particular data base record may have no occurrences of dependent segments or it may have multiple occurrences of the same type.

segment type: A specific group of data fields identifiable as a unit and given a name. The basic unit of a hierarchical structure.

sensitivity: The ability to access a particular unit of data such as fields or segments.

sequence field: A field of a segment, the contents of which identify that segment. The sequence field of the root segment uniquely identifies the data base record and is called the primary key.

sequential organization: Records of a sequential file are arranged in the order in which they will be processed.

service program: A program that assists in the use of a computing system, without contributing directly to the control of the system or the production of results.

shared library: An ICCF public library to which all users can have read/write access or an ICCF private library to which selected users can have read/write access.

shared virtual area: An area located in the highest addresses of virtual storage. It can contain a system directory list of highly used phase, resident programs that can be shared between partitions, and an area for system GETVIS support.

software: A set of programs, concerned with the operation of the hardware in a data processing system.

source: The statements written by the programmer in any programming language with the exception of actual machine language.

source program: A computer program written in a source language that is input to a computer or language translator.

source statement library: A collection of books (such as macro definitions) cataloged in the system by the librarian program.

split screen facility: The facility of the ICCF full screen editor that allows a user to display and edit multiple library members or files or different areas of the same file on the display screen at the same time. With the split screen facility the physical display screen of an IBM 3270 can be divided into two or more output areas.

spool areas: The temporary disk storage areas within the ICCF library file. Each user has four spool areas. They are the input area, print area, punch area and log area.

storage protection: An arrangement for preventing access to storage.

subject: The elements contained in a data dictionary.

submit facility: This ICCF facility allows the terminal user to enter jobs into the VSE/POWER input queue for execution in a DOS/VSE partition.

supervisor: A component of the control program. It consists of routines to control the functions of program loading, machine interruptions, external interruptions, operator communications and physical IOCS requests and interruptions. The supervisor alone operates in the privileged (supervisor) state. It coexists in real storage with problem programs.

switched line: A communication line in which the connection between the computer and a remote station is established by dialing. Synonymous with dial line.

system: In data processing, a collection of men, machines, and methods organized to accomplish a set of specific functions.

system commands: Instructions to ICCF that direct general system functions, such as beginning and ending a user terminal session.

system control programming: IBM programming that is fundamental to the operation and maintenance of the system. It serves as an interface for program products as well as user programs and is directly involved with the management of available system resources.

system directory list: A list containing directory entries of highly used phases and of all phases resident in the shared virtual area. This list is contained in the shared virtual area.

system programmer: A programmer who plans, generates, maintains, extends, and controls the use of an operating system with the aim of improving the overall productivity of an installation.

system record: The record in the ICCF library file which contains the control information about the library file.

system residence device: The direct access device on which the system residence file is located.

system resource: Any facility of the computing system or operating system required by a job or task, and including main storage, input/output devices, the processing unit, data sets, and control or processing programs.

task: A basic unit of work for the central processing unit.

telecommunication: Data transmission between a computing system and remotely located devices via a unit that performs the necessary format conversion and controls the rate of transmission.

teleprocessing: The processing of data that is received from or sent to remote locations by way of telecommunication lines.

temporary file: A file that is accessible only during execution of the job or job step using it. It can be erased or overwritten when it is no longer needed.

terminal: A device, usually equipped with a keyboard and some kind of display, capable of sending and receiving information to/from a computer.

terminal control facility: A program or series of programs that provide the telecommunications interface between the terminal user and the terminal.

terminal operator: The person utilizing the terminal to communicate with the computer.

terminal session: The period of time during which a user of a terminal can communicate with an interactive system; usually, the elapsed time from user logon to user logoff.

Terminal Transaction Facility: A separate program within ICCF that provides the telecommunications interface between the terminal user and the terminal.

terminal user: See terminal operator.

time sharing system: A method of using a computing system that allows a number of users to execute programs concurrently and to interact with the programs during execution.

throughput: The total volume of work performed by a computing system over a given period of time.

track: The portion of a moving storage medium, such as a tape, diskette, or disk, that is accessible to a given reading head position.

transaction: An exchange between a terminal operator and a computer to accomplish a particular action.

transaction identification code: A group of 1 to 4 characters entered by a terminal operator to select a transaction.

transid: See transaction identification code.

TTF: See Terminal Transaction Facility.

turnaround time: The elapsed time between submission of a job and the return of the completed output.

twin: All occurrences of a particular segment type dependent on a specific occurrence of a parent.

unit: (1) A device having a special function. (2) A basic element example magnetic tape unit.

update mode: The ICCF mode of operating enabling a user to locate and modify data by specific line number.

user: Anyone who requires the services of a computing system.

user profile: The record in the ICCF library file which contains specific information about an individual user. There is a user profile record for each user of ICCF.

user program: See application program.

utility program: A problem program designed to perform a routine task, such as transcribing data from one storage device to another.

verify: To check the results of keypunching.

virtual address: An address that refers to virtual storage and must, therefore, be translated into a real storage address when it is used.

virtual storage: Addressable space that appears to the user as real storage, from which instructions and data are mapped into processor storage locations. The size of virtual storage is limited by the addressing scheme of the computing system and by the capacity of the page data set, rather than by the actual number of real storage locations.

virtual storage access method (VSAM): An access method (available as the licensed program product VSE/VSAM) for direct or sequential processing of fixed and variable length records on direct access devices; designed for use in a virtual storage environment.

virtual telecommunications access method (VTAM): A set of IBM programs (available as the licensed program product ACF/VTAM) that control communications between terminals and application programs.

volume: (1) That portion of a single unit of storage media which is accessible to a single read/write mechanism, for example, a diskette, a disk pack, or part of a disk storage module. (2) A recording medium that is mounted and dismounted as a unit, for example, a reel of magnetic tape, a disk pack, or a diskette.

VSAM catalog: A key-sequenced file, with an index, containing extensive file and volume information that VSAM requires to locate files, to allocate and deallocate storage space, to verify the authorization of a program or operator to gain access to a file, and to accumulate usage statistics for files.

VSE/Access Control-Logging and Reporting: An IBM program product designed to work with DOS/VSE and ICCF to monitor access to restricted resources used under DOS/VSE and to protect the data processing installation from unauthorized usage.

VSE/ICCF: Virtual Storage Extended/Interactive Computing and Control Facility. See Interactive Computing and Control Facility.

VSE/POWER: See POWER.

work file: A file on an auxiliary storage medium reserved for intermediate results during execution of the program.





International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

GR20-4666-0

Introduction to IBM 4300 and DOS/VSE Facilities Printed in U.S.A. GR20-4666-1