IBM Field Engineering Education
Student Self-Study Course

DOS Maintenance Facilities

PREFACE

This publication is primarily intended for use by FE customer engineers enrolled in course 50220.

IBM is grateful to the American National Standards Institute (ANSI) for permission to reprint its definitions from the American National Standard Vocabulary for Information Processing, which was prepared by Subcommittee X3K5 on Terminology and Glossary of American National Standards Committee X3.

ANSI definitions are preceded by an asterisk. The symbol '(SCI)' at the beginning of a definition indicates that it has been discussed and agreed upon at meetings of the International Organization for Standardization Technical Committee 97/Subcommittee 1, and has also been approved by ANSI and included in the American National Standard Vocabulary for Information Processing.

Those wishing to reprint the asterisked ANSI definitions are advised to seek permission from the American National Standards Institute.

ANSI definitions appear in this course only when called from the glossary by the student using the "go to define" feature.

Fourth Edition (September 1973)
This is a major revision of, and makes SR25-5640-2 obsolete.

Issued to: _____

Branch Office: _____ No: _____

Address: _____

_____

If this manual is mislaid, please return it to the above address.

Address any comments concerning the contents of this publication to: IBM, Field Support Documentation, Dept 927, Rochester, Minnesota 55901.

CONTENTS

GENERAL INFORMATION

LEGEND

BG	Background
CAW	Channel Address Word
CCB	Command Control Block
CCW	Channel Command Word
CE	Customer Engineer
CI	Core Image
COMREG	Communications Region
CPU	Central Processing Unit
CSW	Channel Status Word
DASD	Direct Access Storage Device
DEBE	File to file utility
DITTO	File to file utility
DMF	DOS MAINTENANCE FACILITIES
DOS	Disk Operating System
DSERV	Directory Service
DSPLY	Display
EBCDIC	Extended Binary Coded Decimal Interchange Code
EOB	End of Block
EOJ	End of Job
EQE	Error Queue Entries
EREP	Environmental Recording Editing and Printing Program
EXEC	Execute
F1	Foreground 1
F2	Foreground 2
GMT	Greenwich Mean-Time
IMPL	Initial Microprogram Program Load
IPL	Initial Program Load
JCC	Job Control Command
JCS	Job Control Statement
JECL	Job Entry Control Language
LUB	Logical Unit Block
OS	Operating System
PDAID	Problem Determination Aids
PIK	Program Interrupt Key
PIOCS	Physical Input Output Control System
POWER	Priority Output Writers, Execution Processors and Input Readers
PSW	Program Status Word
PUB	Physical Unit Block
RAS	Reliability Availability Serviceability
SADP	Serviceability Aids and Debugging Procedures
SAL	Stop After Log
SEREP	System Error Recording and Edit Print Program
SIO	Start Input/Output
SPOOL	Simultaneous Peripheral Operations Online
SRL	System Reference Library
SSC	Self-Study Course (This manual)
SVC	Supervisor Call

SYSGEN	System Generation
TOD	Time of Day
TXT	Text
UCS	Universal Character Set
UPSI	User Program Switch Indicator
VTOC	Volume Table of Contents

COURSE DESCRIPTION

This course is designed to teach the CE's DOS Operating Procedures and the use of Problem Determination Aids, Utility Programs and DOS Messages for locating hardware problems.

SPECIAL INSTRUCTIONS

The figures and appendixes called out in this course are found in the Supplementary Course Material manual, form SR25-5673-1.

Prerequisites

50009 System Introduction

BASIC SKILLS

1. Be able to use JCL to define the I/O devices required to perform a job and initiate the execution of that job.
2. Be able to use the system reconfiguration facilities.
3. Invoke, control the execution of, and interpret the output of programs used for on-line maintenance in a POWER environment.
4. In the event of an ABEND, use the system messages, dumps, and the edited output of I/O error recording facilities to isolate the malfunction to software, hardware, or media.
5. Using messages, dumps, and/or other available indicators, determine the cause of a system wait.
6. Perform the necessary steps to execute system recovery procedures such as: IPL, Warm Start, Checkpoint Restart.
7. Using applicable and available facilities, execute recovery procedures from I/O and storage media failures.
8. Interpret messages and take indicated action for software failure.
9. Interpret error messages to determine: Hardware Failures, Storage Media Failures, Program Errors, JCL Errors.
10. Be able to interpret a disk dump.
11. Identify relationship and use of VTOC and labels to the data sets on a volume.
12. Determine the status of any I/O device by using the appropriate system I/O control blocks.
13. Use an assembler language listing and linkage editor map to locate, in a core dump, any given instruction or routine of a simple one-phase program.
14. Use the system control panel and/or the console typewriter to communicate with IPL and job control to assemble, linkedit, and execute a simple one-phase program.
15. Operate stand-alone/system programs to display the information on any I/O device.
16. Operate the SEREP program to determine the system status when the DOS goes to the wait state.
17. Use the PIOCS macros to accomplish I/O operations on a card, printer, tape or DASD device(s).
18. Utilize those problem determination aids provided by the latest release of DOS that will assist in defining hardware problems. The skills include, but are not limited to, the ability to utilize: (a) Interrupt and I/O Trace (b) Formatted Dump.
19. Obtain and utilize a system dump as an aid in problem definition.
20. The above skills must also make the man capable of operating in a RMS, POWER, and TP environment.

The preceding basic skills are a general description of the student job in relation to the subject. For more detailed information, refer to the objectives located within each session of this manual.

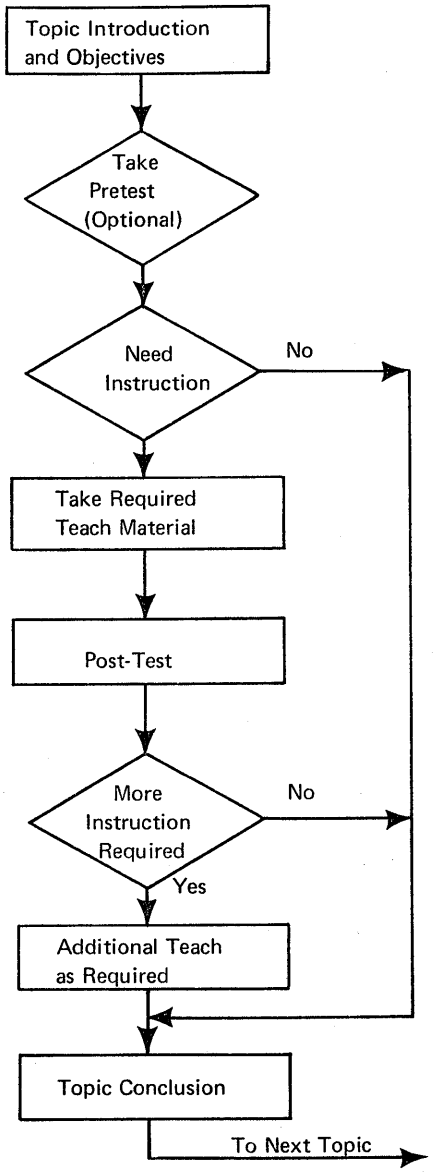
MATERIALS REQUIRED

Your course administrator or monitor will provide you with the necessary materials to complete this course.

INSTRUCTIONS TO THE STUDENT

The instructional method used for each session is designed so that you will receive only the instruction required for your individual needs. Below is a flowchart of the method used. The chart is explained on the next page.

There is a Course Map included in this book. It indicates the average time for each session. There is a column to record your actual time on and off the terminal. Please record this information.



Session Introduction and Objectives

At the beginning of each session, you will be given the session introduction and the objectives for that session. Pay particular attention to the objectives because they tell you what you must be able to do in order to satisfactorily complete the session. The pretest and post-test questions relate directly to the objectives.

Pretest

After receiving the session introduction and objectives, you will be given the choice of taking or skipping the pretest. The purpose of the pretest is simply to determine which parts of the session you already know and which parts you need to be taught. If you feel that you are totally unfamiliar with the subject, we suggest that you bypass the pretest. In that case, you would be taught the entire session.

If, on the other hand, because of prior training or experience you feel that you are already familiar with portions of the session, then it would be to your advantage to take the pretest. The results of the pretest will be used to determine which portion of the session you will be taught. If the results indicate that you need no further instruction, you will bypass all of the teaching material for that session.

CAUTION: Do not guess at answers on the pretest. If you are not sure of an answer, send an EOB. This will enable you to receive the teaching material for that question.

You should answer the questions by entering either a, b, c, d, e, T (TRUE) or F (FALSE). Any other entry will be counted as an incorrect answer.

The pretest merely enables you to move through the session more quickly if you have prior knowledge or experience.

Teaching Material

In the teaching material, you receive the amount of instruction needed to achieve the objectives for the session. You may be taught all of the material or only a portion, depending on the results of the pretest.

Post-Test

The post-test checks to see if you understand what was taught in the teaching material. Normally, you will not have to answer all the post-test questions for a session. The terminal will tell you which post-test questions to answer. Answer the questions in the same manner as you did in the pretest.

Session Conclusion

If you need no further instruction after the post-test, you will be given the session conclusion and go to the next session of study.

Additional Teach as Required

The terminal will type a short critique for each post-test question that you answered incorrectly. If you have missed only a few questions, you will go to the session conclusion. If the results of the post-test indicate you have missed several important concepts, you will be given more instruction. Because of this, you should answer each question on the post-test to the best of your ability.

SAFETY - FIS COURSES

An individual may be exposed to two conditions--HAZARDOUS and NON-HAZARDOUS. All FIS course requirements are considered to be NON-HAZARDOUS. In a NON-HAZARDOUS condition an "IBM employee may work alone providing arrangements are made for at least an hourly check relative to his well-being. This may be accomplished by telephone or direct personal communication." You should insure that local arrangements are made to comply with this safety precaution.

DOS MAINTENANCE FACILITIES — Course Name

50220 — Course Number

Course Length: 15.7 Low | 32.4 High | 27.0 Average

Session/Topic		Student On-Terminal Time (Estimate)	Student Off-Terminal Time (Estimate)	Student Actual Time (Filled in by student)	Monitor Terminal Time (Estimate)	Media Requirements (See legend below)	Training Machine Required*
1.	INTRODUCTION TO AN OPERATING SYSTEM	2.0					
2.	SYSTEM CONCEPTS	2.0					
3.	SYSTEM RESIDENCE ORGANIZATION	1.5					
4.	SYSTEM/370 CPU OPERATION	.5					
5.	DOS CONCEPTS AND OPERATING PROCEDURES	3.0					
6.	MAPS/STORAGE PRINTS/SYSTEM MESSAGES	1.0					
7.	PHYSICAL INPUT OUTPUT CONTROL SYSTEM (PIOCS) MACROS	1.0					
8.	PHYSICAL INPUT OUTPUT CONTROL SYSTEM (PIOCS) MACRO USAGE	2.0					
9.	I/O MESSAGES	1.0					
10.	SUPERVISOR CONCEPTS	2.0					
11.	SUPERVISOR TABLES	2.5					
12.	STANDALONE AND SYSTEM UTILITY PROGRAMS	1.5					
13.	PROBLEM DETERMINATION AID (PDAID) PROGRAMS	2.0					
14.	POWER CONCEPTS	1.5					
15.	POWER OPERATION	1.5					
16.	EVALUATION - FINAL QUIZ	2.0					
	Time Totals	27.0					

Legend

A. 16mm Motion Film	E. Microfiche	J. Simulator
B. 8mm Motion Film	F. Video Tape	K.
C. Filmstrip	G. Audio Tape	L.
D. Slides	H. Mock-Up	M.

*Check if B/O Training Machine Required

SESSION 1 - INTRODUCTION TO AN OPERATING SYSTEM

This session requires approximately 2.0 hours to complete.

This session reviews concepts of an operating system. The concepts of the operating system discussed will not be oriented toward any particular operating system. The purpose of this course and rationale for training hardware CEs on an operating system will be discussed.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

1. Given a list of components, select those required to make up a computer system.
2. Define an operating system and identify the types of programs that make up an operating system.
3. Given a list of programs, determine if the program is a control program or a processing program.
4. Place in the correct sequence, the steps required to implement a program.
5. Identify the purpose of the following programs when operating in an operating system environment.
   a. IPL
   b. Job Control
   c. Supervisor
   d. Assembler
   e. Linkage Editor
6. Identify the correct sequence of operation and interaction between the following programs when executing an assembly.
   a. IPL
   b. Job Control
   c. Supervisor
   d. Assembly

Highlights

- An operating system is made up of control and processing programs.
- The function of an operating system is to make optimum use of system resources.
- System control programs consist of:
  IPL
  Job Control
  Supervisor

- IPL initializes the system and loads the supervisor.
- Supervisor is core resident after IPL and acts as a monitor for all other programs.
- Job control is in core between job steps and provides job-to-job transition.
- The linkage editor is a service program, and prepares object programs for execution.
- An operating system executes programs from the Core Image Library.

Activity

REMINDER: All figures and Appendixes referred to in this manual can be found in the Supplementary Course Material manual, form SR25-5673-0.

NOTE: The term 'DOS' and 'DOS/VS' are synonymous.

Read all of Section 1-1, 'COURSE INTRODUCTION', in the 'SUPPLEMENTARY MATERIAL' portion of this manual, and when you have finished, answer the study questions at the end of the section.

PRETEST/POST-TEST QUESTIONS (SESSION 1)

1. (True/False) A hardware system consists of I/O devices, a CPU, and Programs.

2. Select two components from the list below that most correctly describe a computer system.

   a. Hardware
   b. CPU
   c. Software
   d. Disk
   e. Reader

3. Select the definition below that best describes an operating system.

   a. Organized method to accomplish a task.
   b. CPU, reader, disk, printer, and console.
   c. It is a system (software) that makes the hardware system easier to use.
   d. Instructions to do an I/O operation.

4. An operating system is made up of:

   a. A hardware system and a software system.
   b. CPU and hardware I/O.
   c. Control programs and hardware system.
   d. Control programs and processing program.
   e. Hardware system and processing program.

5. (True/False) A control program controls or monitors the use of a computer system.

6. Major hardware resources that the operating system monitors and optimizes to produce more throughput for the customer are:

   a. Direct access storage space.
   b. Main storage space.
   c. CPU time.
   d. I/O devices.
   e. All of the above

7. The control program that provides job-to-job transition with no operator intervention is:

   a. IPL.
   b. Supervisor.
   c. Job Control.
   d. Linkage Editor.
   e. Assembler.

8. (True/False) The customer writes all the processing programs in an operating system.

9. Programming languages that a programmer can effectively use, that do not require in-depth knowledge of a hardware system are:
   (More than one answer is required)

   a. FORTRAN
   b. PL/I
   c. COBOL
   d. ASSEMBLER
   e. MACHINE language

10. A macro statement when processed by the ASSEMBLER, generates:

    a. Many microinstructions.
    b. Many machine instructions.

11. Arrange the following items in the correct numerical sequence required to implement a program.

    ____ Linkedit time
    ____ Execute time
    ____ Program development
    ____ Compile time

    Select one of the following (a, b, c, d, or e) when you input your answer to question 11.

    a. 1, 2, 3, 4
    b. 3, 4, 2, 1
    c. 3, 1, 4, 2
    d. 3, 4, 1, 2
    e. 1, 3, 2, 4

12. All processing programs are loaded into storage for execution by the:

    a. Job control.
    b. Linkage editor.
    c. System loader, a portion of the linkage editor.
    d. System loader, a portion of job control.
    e. None of the above.

13. The program that provides for relocation of object modules prior to execution is:

    a. IPL.
    b. SUPERVISOR.
    c. Job Control.
    d. Linkage Editor.
    e. Assembler.

14. The IBM SORT program is: (More then one answer is required)

    a. A customer written program.
    b. A language translator.
    c. A service program.
    d. A control program.
    e. A processing program.

15. The _____ program places programs in the Core Image Library and the _____ program takes the programs from this library and loads them in core for execution.

    a. Supervisor, supervisor
    b. Linkage editor, linkage editor
    c. Supervisor, linkage editor
    d. Linkage editor, supervisor
    e. Utility, supervisor

16. Match the following programs with their associated function.

	Function	Programs
____	a. Loads the supervisor	1. IPL
____	b. Loads job control	2. Job Control
____	c. Loads linkage editor	3. Supervisor
____	d. Translates symbolic to machine language	4. Assembler
____	e. First DOS control program loaded into the system	5. Linkage Editor
____	f. Converts object module from language translator output to executable format	
____	g. Resides on SYSRES	
____	h. Is a service program	
____	i. Provides job-to-job transition	
____	j. Handles I/O operations	

Select one of the following (a, b, c, d, or e) when you input your answer for question 16.

a. a=1; b=3; c=3; d=4; e=1; f=5; g=1,2,3,4,5; h=5; i=2; j=3
b. a=2; b=3; c=2; d=4; e=1; f=5; g=1,2,3,4,5; h=5; i=2; j=3
c. a=1; b=3; c=3; d=4; e=1; f=5; g=1,2,3,4; h=5; i=2; j=3
d. a=1; b=3; c=3; d=4; e=1; f=5; g=1,2,3,4,5; h=5; i=3; j=3
e. None of the above.

17. Arrange the following programs according to the numerical sequence in which they are loaded into core, when a system is IPLed and a program is assembled.

____ Supervisor
____ Assembler
____ IPL
____ Job Control

Select one of the following (a, b, c, d, or e) when you input your answer to question 17.

a. 1,2,4,3
b. 2,3,1,4
c. 1,4,2,3
d. 3,4,1,2
e. 2,4,1,3

SESSION 2 - SYSTEM CONCEPTS

This session requires approximately 2.0 hours to complete.

This session introduces the concepts of system operation unique to the Disk Operating System (DOS). Discussed are such items as:

(1) Available reference publications with emphasis on the function of specific manuals.
(2) System communications including message formats and the operator responses to them.
(3) The operating procedures and control statements required to IPL a DOS system.
(4) An introduction to multiprogramming and system generation (SYSGEN).
(5) The use of symbolic references as they apply to I/O operations.
(6) An introduction to the concepts of virtual storage.

This session provides a basic understanding of the DOS environment which you will use throughout the remainder of the course.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

1. Choose the correct manual to use as a reference when performing selected operator functions.
2. Identify the operation of Job Control (when processing an ASSGN statement) as it relates to the Logical Unit Block (LUB) and the Physical Unit Block (PUB).
3. Determine which program has priority when given a multiprogramming system configuration.
4. Select from a list, the most correct statement relative to virtual storage operation.
5. Select from a list, those items which require the customer to do a SYSGEN.
6. Given a DOS error message, identify the action indicator, message number, IBM program, and partition that issued the message.

*7. Submit the control commands required to IPL a DOS system.

*8. Given an error message as a result of an incorrect IPL control statement, identify the action to be taken and correct the error.

* Partially or wholly supported by follow-on lab.

Highlights

- Many manuals are available as reference for operating DOS/VS.
- I/O device assignments can be modified via an ASSGN statement.
- Multiprogramming provides the facility of three programs executing in core at the same time.
- System Generation (SYSGEN) is that process whereby the customer builds an operating system to fit his needs.
- IPL is the first operating procedure performed by the operator to bring up DOS.
- Virtual storage allows better utilization of real storage.

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 2)

1. Choose 3 manuals from the list that are used primarily by a system operator using DOS/VS. (Three answers are required.)

   a. DOS/VS SADP
   b. DOS/VS Operating Procedures
   c. DOS OLTEP SRL
   d. DOS/VS Messages
   e. DOS/VS System Control and Service

2. The ASSGN statement changes the pointer in the:

   a. PUB
   b. LUB
   c. Job Control Program
   d. IPL Program

3. The hardware address of an I/O device is located in the:

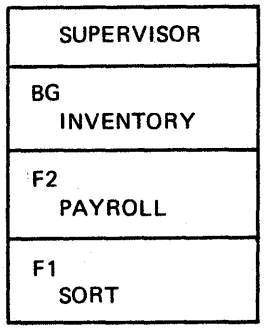
   a. PUB
   b. LUB
   c. Job Control Program
   d. IPL Program

4. The LUB and PUB are located in the:

   a. IPL Program
   b. Job Control Program
   c. Supervisor
   d. ASSGN Statement

5. Determine the problem program with the highest priority in the following figure (assume default options).

   a. Supervisor
   b. Inventory
   c. Payroll
   d. Sort



6. What 2 items must be saved when control is transferred from one partition to another? (Two answers are required)

   a. The OLD PSW's
   b. The current PSW of the interrupted partition
   c. The problem program I/O data
   d. The contents of the general purpose registers

7. Select the most correct statement.

   a. All programs in the virtual address area are divided into 1K blocks of code.
   b. The supervisor always operates in virtual mode and occupies the lower part of real storage.
   c. The job control program is always executed in virtual mode, even if the program it prepares for execution is to run in real mode.
   d. A page name is a 1K block of storage.

Use the following Job Control statement and error message to answer questions 8 through 11.

```
ASSIGN SYS003,X'281'
BG 1S01D INVALID STATEMENT
```

8. The failing operand is:

   a. BG
   b. ASSIGN
   c. SYS003
   d. x'281'

9. The partition that issued the message is:

   a. F1
   b. Supervisor
   c. ASSIGN
   d. BG

10. The action indicator means:

   a. Action
   b. Decision
   c. Information

11. The control program which issued the message is:

    a. IPL
    b. Job Control (JCL)
    c. Supervisor
    d. None of the above

12. Select the correct DOS IPL statement (command) that would initialize a system to the time of 9:00 PM on February 2, 1972.

    a. SET CLOCK=09/00/00, DATE=02/19/72
    b. SET DATE=02/02/72, CLOCK=21/00/00
    c. SET DATE=02/02/72, CLOCK=09/00/00
    d. SET CLOCK=21/00/00, DATE=02/02/72
    e. SET DATE=02/19/72, CLOCK=09/00/00

13. A DOS system is IPLed and the reader is made ready. One card is read and the system enters the wait state. Your next action should be to:

    a. Check bytes 0 through 4 for an error message.
    b. Run diagnostics on the reader.
    c. Run diagnostics on the disk.
    d. IPL again.
    e. Check the console switches to determine if you have IPLed the correct device.

14. Select from the following list those items which would require a customer to perform a new SYSGEN. (More then one answer is required)

    a. A failing disk drive.
    b. A need for multiprogramming capability.
    c. The addition of a new tape sub-system to the present system configuration.
    d. A need to use the facilities of a new language compiler.

SESSION 3 - SYSTEM RESIDENCE ORGANIZATION

This session requires approximately 1.5 hours to complete.

The system residence pack (SYSRES) is the major element of the DOS system. It contains the necessary software components (called libraries) for system operation. The purpose, size, and format of these libraries are discussed. It is important to gain at least a conversational knowledge of the SYSRES organization so that you might be better able to service your customers running DOS.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

1. Identify the four major libraries of the system residence file and the directories associated with each library.
2. Identify the relative location on SYSRES assigned to each of the following system residence components.
   a. Core image directory
   b. Source library directory
   c. Relocatable directory
   d. Procedure directory
   e. Core image library
   f. Source library
   g. Relocatable library
   h. Procedure library
   i. IPL records
   j. Sub-directories
   k. User area
   l. VTOC
3. Identify the library from which programs are executed by the DOS system.
4. Identify the library, on the system residence pack, where object decks from all IBM compilers can be stored.
5. Identify the library that can be used as input by the ASSEMBLER or COBOL language translators.
6. Identify the function of:
   a. The volume label.
   b. The VTOC.
   c. The data file (Format 1) label.

Highlights

- The system residence pack (SYSRES) contains components called libraries.
- Core image library contains programs in executable format. Programs can be executed only from the core image library.
- Relocatable library contains object modules. These object modules are the output of language translators and the object modules must be linkedited and placed in the core image library before they can be executed.

- Source library contains books in the form of source statements and can be used as input to the language translators.
- Procedure library contains commonly used sets of job control statements.
- Data file labels ensure data integrity.

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 3)

1. (True/False) The four library sizes are determined by the customer.

2. The optional library (libraries) on the system residence pack are:

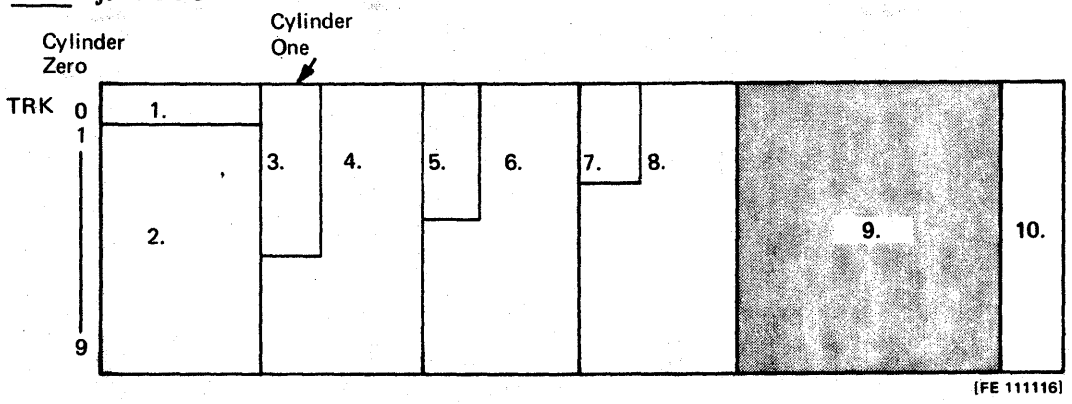
   a. Source, relocatable, and procedure.
   b. Relocatable and core image.
   c. Core image, source, relocatable, and procedure.
   d. Relocatable.
   e. Core image and source.

3. Private libraries can be associated with:

   a. Relocatable library.
   b. Core image library.
   c. Source library.
   d. All of the above.

4. The figure below is a representative layout of 2311 residence file. Match the numbered areas using the identifying letters from the given list. (Procedure library and directory not shown.)

   ____ a. Core image directory
   ____ b. Relocatable directory
   ____ c. Source library
   ____ d. Source directory
   ____ e. Core image library
   ____ f. Relocatable library
   ____ g. IPL records
   ____ h. Sub-directories
   ____ i. User area
   ____ j. VTOC



5. Executable programs are contained in which one of the following libraries?

   a. Relocatable
   b. Source
   c. Core image

6. Object decks produced by language compilers may be stored in the _____ library.

   a. Relocatable
   b. Source
   c. Core image

7. When a compiler has need to generate a MACRO, it obtains the MACRO from which one of the following libraries?

   a. Relocatable
   b. Source
   c. Core image

8. The pointer to the VTOC is contained in:

   a. The volume label.
   b. The first record in the VTOC.
   c. The SYSRES EXTENT.
   d. None of the above.

9. A Format 1 label contains:

   a. A pointer to the VTOC.
   b. Name and extent limit of the file.
   c. A pointer to the volume label.
   d. All of the above.

10. Which of the following describes the relationship between the volume label, the VTOC, a Format 1 label, and a data file?

    a. The VTOC points to the volume label.
       The volume label contains a Format 1 label.
       The Format 1 label defines a data file.
    b. The volume label points to the VTOC.
       The VTOC contains a Format 1 label.
       The Format 1 label defines a data file.
    c. The Format 1 label points to the VTOC.
       The VTOC contains the volume label.
       The volume label defines a data file.
    d. The data file contains the volume label.
       The volume label points to the VTOC.
       The VTOC defines a data file.

SESSION 4 - SYSTEM/370 CPU OPERATION

This session requires approximately 0.5 hour to complete.

This session presents the information required by an operator for normal system operation. This information will be used during the lab portion of the DMF course. During this Session, you will be exposed to information on various System/370 CPU Models that you may possibly encounter in follow-on lab training.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

*1. Operate the System/370 using DOS/VS SADP manual (GC33-5380) to perform the following functions.
   a. Display/alter storage using the Documentary console.
   b. Display/alter general registers, using the Documentary console.
   c. Trace an instruction loop.
   d. Perform the Store Status (ST) function.

* Partially or wholly supported by follow-on lab.

Highlights

- The documentary console can be used to alter/display the following.
  - Main Storage
  - Storage Key
  - General Registers
  - Floating-Point Registers
  - Current PSW
- Power-on of the CPU will automatically load the microprogram for the system if the console file is ready.

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 4)

1. What switch is used to instruction step a loop through a program on a System/370 Model 145?

   a. Address Compare
   b. Check Control
   c. Rate
   d. Storage Select
   e. Diagnostic/Console File Control

2. (True/False) On a System/370 Model 145, pressing the Power-On button will initiate an automatic microprogram loading sequence, provided the Diagnostic/Console File Control switch is in the Read position.

3. Select from the list provided, the items necessary to display general purpose register 10 on a 3210 printer connected to a System/370 Model 145.

   _____ 1. Press alter/display
   _____ 2. Type DF
   _____ 3. Type DM
   _____ 4. Type DG
   _____ 5. Type A
   _____ 6. Type 10
   _____ 7. End

   Select one of the following (a, b, c, d, or e) when you input your answer for question 3.

   a. 1 - 2 - 5 - 7
   b. 1 - 3 - 5— 7
   c. 1 - 4 - 6 - 7
   d. 1 - 2 - 6 - 7
   e. 1 - 4 - 5 - 7

4. On the System/370 Model 125, pressing the MODE SELECT and entering C will display:

   a. Protection Key
   b. Control Registers
   c. Main Storage Real
   d. General Registers

5. On the System/370 Model 125, to alter a main storage address, what function must be performed?

   a. Type in the desired change data.
   b. Type A and the desired change data.
   c. Type the desired change data and press ENTER.
   d. Move the cursor under the desired digit and type the new data.

6. Which state(s) must a System/370 Model 145 be in to alter/display from the Documentary console?

a. Wait
b. Manual
c. System
d. All of the above

7. Select the items necessary to trace a loop through a program, one instruction at a time (System/370 Models 135 and 145).

____ 1. Press stop
____ 2. Press system reset
____ 3. Set check control switch to hard stop
____ 4. Set rate switch to instruction step
____ 5. Set rate switch to single cycle
____ 6. Press start key
____ 7. Press set address and display

Select one of the following (a, b, c, d, or e) when you input your answer for question 7.

a. 1 - 2 - 6 - 7
b. 1 - 3 - 7
c. 1 - 4 - 6
d. 1 - 5 - 6
e. 1 - 2 - 4 - 6

8. Select from the list provided, the items necessary to alter floating-point register 4, using the printer-keyboard on a System/370 Model 135.

____ 1. Press alter/display
____ 2. Type AG
____ 3. Type AF
____ 4. Type AC
____ 5. Type AK
____ 6. Type 4
____ 7. End

Select one of the following (a, b, c, d, or e) when you input your answer for question 8.

a. 1 - 2 - 6 - 7
b. 1 - 3 - 6 - 7
c. 1 - 4 - 6 - 7
d. 1 - 5 - 6 - 7

9. On a System/370 Model 125 system the entry U is made while in a MODE SELECT operation. This will:

a. Store Status
b. Store GP and Control Registers
c. Save Usage Counters
d. Unload Tapes

SESSION 5 - DOS CONCEPTS & OPERATING PROCEDURES

This session requires approximately 3.0 hours to complete.

This session assembles all the elements of DOS (some of which were presented in the previous sessions), and shows how they fit together during system operation. Job execution is discussed; including the details of Job Control statements, with specific emphasis on how they are used to handle job-to-job transition. Checkpoint/Restart- a method of restarting a job from some point other then the beginning is covered. An understanding of system operation, including job control statements and messages, is essential for you to service hardware systems running DOS.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

*1. Correct any errors in job control cards used to do an assembly, linkedit, and execution of a problem program (excluding label cards).
*2. Use error messages and the operator's guide, to correct any operator error encountered during the assembly and execution of a problem program.
*3. Use the console and proper job control statements to obtain a system dump, either during execution or on an abnormal termination of a problem program.
*4. Given an error message as a result of an incorrect job control statement, use the error message to locate and correct the error.
5. Identify the relationships between the IPL program, Job Control program, Supervisor, and problem program in a batched job processing environment.
6. State the purpose of the DOS Checkpoint/Restart procedure.

* Wholly or partially supported by follow-on lab.

Highlights

- A // JOB statement indicates the beginning of a job.
- A job contains one or more job steps.
- Checkpoint/Restart provides the facility to recover from an error without re-starting the job from the beginning.
- Each time a checkpoint is taken, the number of the checkpoint is printed on the console typewriter.

Activity

Return to the terminal and enter an EOB to continue.

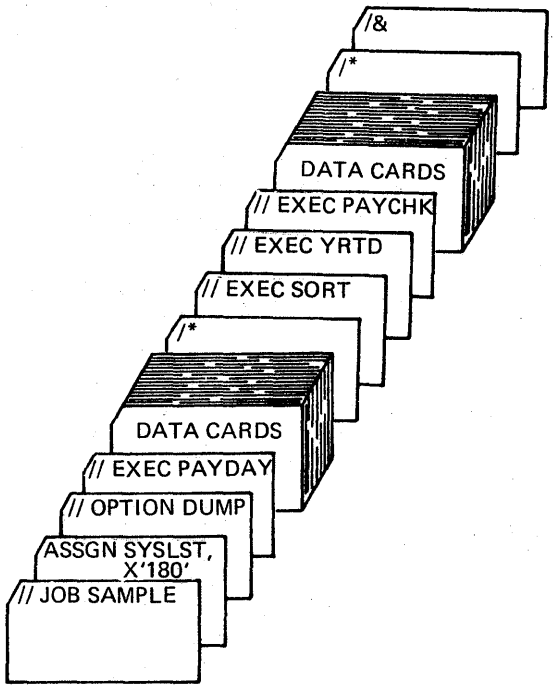
PRETEST/POST-TEST QUESTIONS (SESSION 5)

1. Under DOS, the terms job, job step, and their interrelationship may be most accurately described as:

   a. A job is one or more related job steps run as a unit. It is initiated by a // JOB control card, and terminated by a /* control card.
   b. A job step is a series of related jobs.
   c. A job step is the execution of one processing program. It is initiated by the // JOB control card.
   d. A job is one or more related job steps run as a unit. The unit is initiated by a // JOB control card and terminated by a /& control card.
   e. None of the above.

2. How many job steps are there in the following job stream?

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



3. Assume the operator was running a job and it was cancelled due to a program check. There was no system storage print outputted on SYSLST. Which of the following is the most likely reason for the absence of a storage printout?

   a. SYSLST is unassigned.
   b. The printer is inoperable because of print checks.
   c. There was a disk error when the storage print program was being loaded in core.
   d. There was no // OPTION DUMP job control statement.

4. Choose the command that will provide a listing of the symbolic assignments for a Disk Operating System on SYSLST.

   a. // LISTIO BG
   b. LISTIO ALL
   c. // LISTIO ALL
   d. LISTIO BG

5. The job control program is the unit in the disk operating system that handles job-to-job transition. It:

   a. Reads job control cards from SYSIPT.
   b. Is disk resident and must be loaded into core whenever its functions are needed.
   c. Is core storage resident and its functions are requested via an SVC (supervisor call instruction).
   d. Is loaded, as a deck of cards, from the reader, and in turn loads the supervisor from a disk pack.

6. At the time execution of a processing program is requested, it, like all programs run under control of the disk operating system, is:

   a. Resident in core storage at all times.
   b. A module in the relocatable library.
   c. Loaded, by the system loader, from SYSIPT.
   d. Relocated, loaded and executed by linkage editor.
   e. Fetched from the core image library.

7. Select appropriate cards and decks from the following list, and place them in the sequential alphabetical order required to do an assembly, linkedit, and execution of a problem program. (Items in the list may be used more than once.)

   a. ____
   b. ____
   c. ____
   d. ____
   e. ____
   f. ____
   g. ____
   h. ____
   i. ____
   j. ____

   1. // JOB TEST
   2. // EXEC LNKEDT
   3. /*
   4. // OPTION LINK
   5. OBJECT DECK
   6. /&
   7. SOURCE DECK
   8. // EXEC ASSEMBLER
   9. DATA DECK
   10. // EXEC
   11. // EXEC ASSEMBLY
   12. // EXEC TEST
   13. // EXEC LINKEDT

8. The following ASSGN statement resulted in the error message shown:

```
// ASSGN SYS002,X'181'
1A04 INVALID I/O ASSIGNMENT
```

What action would you take if you were the system operator?

a. Cancel the job and give the message to the programmer.
b. Take a LISTIO printout and make the correct assignment.
c. Call the CE and have the LISTIO printout and error message available for him.
d. Type in IGNORE and continue processing.

9. Select from the list provided the items that are saved in the checkpoint record.

1. PSW
2. Floating Point Registers
3. General Purpose Registers
4. The Problem Program Area
5. Physical position of the I/O devices
6. Supervisor

Select one of the following (a, b, c, d, or e) when you input your answer for question 10.

a. 1-2-3-4-6
b. 1-3-4-5
c. 1-2-4-5
d. 2-3-4-5
e. 1-2-3-5-6

10. (True/False) Checkpoints are taken to eliminate the need to start a program at the beginning after a failure.

SESSION 6 - MAPS/STORAGE PRINTS/AND SYSTEM MESSAGES

This session requires approximately 1.0 hours to complete.

This session will show the format and purpose of the Linkedit Map, and the format of standalone and system dumps. It will also show you how to interpret system cancel messages. Included is a review of the various fields of an Assembler listing and their uses. You will use this information in follow-on lab to solve trouble analysis problems.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

1. Given a storage print, linkedit map and an assembler listing, locate any instruction of a problem program.
2. Given a standalone or system storage print, locate the following:
   a. General Registers
   b. CAW
   c. CSW
   d. PSWs (old)
   e. PSWs (new)
3. Given a system dump, use the system message to determine the cause of the cancellation.
4. Use the assembler listing to debug program errors.

Highlights

- A Linkedit Map is produced each time a // EXEC LNKEDT run is made.
- An Assembly listing contains:
  - External Symbol Dictionary
  - Source and Object Coding
  - Relocation Dictionary
  - Cross Reference
  - Diagnostics

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 6)

1. In an assembler listing, what is the purpose of the diagnostic section?

   a. Contains the literals that appear in the listing.
   b. Contains information concerning symbols, where they are defined and used in the program.
   c. Contains error messages.
   d. Contains the relocation dictionary information.

2. (True/False) In a linkedit map, the XFR-ADR and LOCORE address will always be the same.

3. (True/False) A linkedit map is printed at the conclustion of every successful // EXEC LNKEDT job-step.

4. The difference between the storage address a program is assembled at, and the actual storage address a program resides at in core, is found in the field of the linkedit map designated as:

   a. XFR-ADR
   b. LOCORE
   e. DSK-ADR
   d. LOADED
   e. REL-FR

Reference Figure 6.4 (parts 1 through 11) for Questions 5 through 8.

5. When the program is loaded into storage, the actual address of the RDCCW (stmt 45) will be:

   a. 376A
   b. 3760
   c. 3660
   d. 3750
   e. 3650

6. In the dump provided [Figure 6.4 (Part 5 of 11)], what information is contained in GP reg 1?

   a. 000036A8
   b. 00003750
   c. 444E1FF0
   d. 000036DC
   e. 0A0407F1

7. In the dump provided [Figure 6.4 (Part 5 of 11)], what is the contents of the CAW?

   a. 000029D0
   b. 000029C0
   c. 00000000
   d. 00040000
   e. 00000324

8. The instruction at 0036C8 in the dump is located at which statement number in the assembler listing?

   a. 5
   b. 10
   c. 12
   d. 13
   e. 20

9. Which of the following conditions would cause a 0P81I error message?

   a. Job cancelled due to I/O operator option.
   b. Job cancelled due to I/O error.
   c. Job cancelled due to Invalid address.
   d. Job cancelled due to CPU failure.
   e. Job cancelled due to Channel failure.

10. What would be the cause for the following message?

    0T03I ERROR ON RECORDER FILE AT B9192

   a. Unrecoverable I/O error on the recorder file.
   b. Recorder file not opened.
   c. Recorder file full.
   d. Recorder file located at invalid extent.

SESSION 7 - PHYSICAL INPUT OUTPUT CONTROL SYSTEM (PIOCS) MACROS

This session requires approximately 1.0 hours to complete.

This session introduces the coding used by all problem programs to issue an I/O request, and to show the relationship of the physical I/O control blocks within the problem program and core resident supervisor.

How many times have you as a CE, worked on a supposedly solid hardware problem, and ran all diagnostics on a device without any trouble showing up? The device is turned back to the customer and the device immediately fails.

This usually causes the hardware CE to ask himself some of the following questions?

- What do I do now?
- Why is this device still failing, when the diagnostics ran correctly?
- Is it a hardware problem?
- Is it a software problem?
- How can I make it fail so I can fix it?
- Where are the CCW's in storage that caused the failure?
- How can I loop it on the failing CCW chain?
- Can I write a little program to make it fail?
- Where is the information that the program is analyzing to determine it had a problem?

A knowledge of PIOCS will allow the CE to answer many of these questions. In DOS all problem programs request I/O operations exactly the same way, no matter if the program is written in Assembler, COBOL, PL/I, FORTRAN, etc.

This session will explain how the programs request I/O operations, how the supervisor handles these requests, the control blocks used in the problem program and the machine instructions required.

It is not our intent to make you an expert in coding PIOCS macros, but the easiest way to understand how these macros function, is to be able to use them in a program.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

1. Identify the function of the following macros when used in a problem program:
   a. EXCP macro
   b. WAIT macro
   c. CCB macro
2. Identify the functions of the Physical Input Output Control System.
3. Identify the methods used to transfer control between the supervisor and problem program and vice versa.

4. Identify the relationship between the following items when related to an I/O operation:
   - EXCP macro
   - WAIT macro
   - CCB macro
   - CCW instruction
   - I/O Area

Highlights

- The Physical Input Output Control System (PIOCS) handles all I/O requests in a DOS environment.
- The problem program requests I/O operations from the supervisor using the following PIOCS macros:
  - EXCP (Execute Channel Program)
  - WAIT
  - CCB (Channel Control Block)

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 7)

1. The Supervisor transfers control to the problem program by a:

   a. EXCP macro
   b. WAIT macro
   c. Supervisor Call (SVC) instruction
   d. Load PSW instruction
   e. Branch unconditionally instruction

2. The Supervisor can determine the function requested by a problem program from analyzing:

   a. General Register 0
   b. General Register 1
   c. Interruption code of the SVC new PSW
   d. Interruption code of the SVC old PSW
   e. CCB

3. The channel scheduler (PIOCS) routines perform the following functions. (More than one answer is required)

   a. Schedules I/O request on each channel
   b. Starts input/output operations
   c. Loads problem programs
   d. Handles end-of-job
   e. Interval timer services

4. The Physical Input Output Control System is a portion of the:

   a. Problem program
   b. Supervisor
   c. EXCP (Execute Channel Program) macro
   d. CCB (Command Control Block)
   e. CCW (Channel Control Word)

5. Match the PIOCS macros with their associated function.

   a. ___ EXCP
   b. ___ WAIT
   c. ___ CCB

   1. Initiates an I/O request from the problem program.
   2. Area the input/output record occupies in storage.
   3. Contains the SIO instruction for the I/O request.
   4. Suspends program operation until an I/O data transfer has been completed.
   5. Problem program constants (parameters) required for an I/O request.

6. The CCB (Command Control Block) contains: (More than one answer is required)
   a. Address of the CCW(s) to be used for an I/O request.
   b. Hardware channel and device address the I/O request is for.
   c. Address of the I/O area.
   d. CCW's
   e. Symbolic device address

7. The WAIT macro's operand is the name of the:

   a. EXCP macro
   b. CCB macro
   c. I/O area
   d. CCW(s) chain
   e. Symbolic device

SESSION 8 - PHYSICAL INPUT OUTPUT CONTROL SYSTEM (PIOCS) MACRO USAGE

This session requires approximately 2.0 hours to complete.

In this session the actual coding to use the Physical Input Output Control System (PIOCS) macros will be shown. Also covered will be the PIOCS macro format and several examples of how they are used. Special emphasis will be placed on the interpretation of the CCB and its contents and how the information can be helpful to a CE when trying to identify a problem.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

1. Identify the function of a macro within the Disk Operating System.
*2. Use notation conventions to correctly code a macro.
*3. Identify the general purpose registers that have operating system restrictions.
*4. Code any PIOCS macros required for an input/output operation on a unit record, tape or DASD device.
*5. Given a storage print due to an I/O error, locate the CCB and the following items in the CCB as an aid to problem determination:
   a. Transmission information
   b. CSW status bytes
   c. Symbolic unit address
   d. CCW address
   e. CCW address in CSW
*6. Correct any improper logic in the use of the EXCP, WAIT and CCB macros when used to control any card reader, printer, console typewriter, or tape.
*7. Identify in CCB, those bytes or bits that are used during an error free I/O operation for any unit record type file.
*8. Correct any improper logic in the use of the CCB, WAIT, EXCP macros, and the test under mask on branch condition instructions to sense carriage overflow on a printer and skip to one.
*9. Given a system error message due to an invalid device assignment, use the LISTIO operator command to determine the assignment status of any physical or symbolic device.

* Partially or wholly supported by follow-on lab.

Highlights

- The CCB contains descriptive information that can be useful when attempting to locate an I/O problem.
- The WAIT macro will issue a SVC 7 to place the CPU in the wait state, if CPU processing must be suspended while waiting for completion of an I/O operation.
- General Register 1 normally points to the CCB being used or last used for an I/O operation.
- The EOJ macro is used to terminate problem program processing.

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 8)

1. The Assembler will access the _____ library to generate the instructions required for a macro.

   a. core image
   b. relocatable
   c. source
   d. program
   e. assembler

2. The Assembler will _____ the items specified by the programmer into the macro instructions and generate assembler mnemonic instructions.

   a. eliminate
   b. delete
   c. modify
   d. substitute
   e. subtract

3. In the Assembler listing the mnemonic instructions generated for a macro are identified with a:

   a. *
   b. +
   c. M
   d. G
   e. /

4. Match the notation convention with its description:

_____ a.	[ ] (brackets)	1. required parameter
_____ b.	. . . (ellipsis)	2. substitute decimal number
_____ c.	{ } (braces)	3. invalid parameter
_____ d.	____ (underlined)	4. optional parameter
_____ e.	n (lower case letter)	5. variable number of parameters may be supplied
		6. substitute general register number
		7. assumed parameter

5. General registers _____ can be used with no restrictions by the problem program.

   a. 0 - 12
   b. 2 - 12
   c. 0, 1, 13, 14, and 15
   d. 0 and 1
   e. 13, 14, and 15

The following CCB will be used for questions 6 through 8:

```
READ CCB SYS005,READER
```

6. The operand of the EXCP and WAIT macros that refer to this CCB must be:

   a. READ
   b. READCCB
   c. CCB
   d. SYS005
   e. READER

7. The name of the CCW chain associated with this CCB is:

   a. READ
   b. CCB
   c. READCCW
   d. READER
   e. SYS005

8. (True/False) This CCB when used as the operand of an EXCP macro, will cause an I/O operation to occur on SYSRDR.

9. Place the following instructions in the correct sequence (1 through 7) to perform a continuous read/print operation.

   (Assume all housekeeping instructions, device assignments, I/O areas, and CCW's have been defined correctly.)

```
_____ a.                WAIT   READ
_____ b.  READLOOP      EXCP   READ
_____ c.  READ          CCB    SYS005,READER
_____ d.                EXCP   PRINT
_____ e.                WAIT   PRINT
_____ f.  PRINT         CCB    SYSLST,PRINTER
_____ g.                BC     15,READLOOP
```

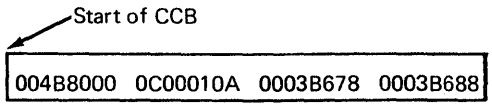
10. The mnemonic instructions expanded for the EXCP macro will load the address of the CCB into general register:

    a. 0
    b. 1
    c. 2
    d. 14
    e. 15

11. The WAIT macro will issue a SVC _____ to suspend problem program processing in a batch job environment.

    a. 0
    b. 1
    c. 4
    d. 7
    e. 14

Use the following storage print of a CCB to answer questions 12 through 14.



12. This CCB is for the symbolic unit:

    a. SYSREC
    b. SYSLST
    c. SYS000
    d. SYS010
    e. SYS075

13. The address of the CCW chain to be executed for an I/O request for this CCB is:

    a. X'4B8000'
    b. X'0C0001'
    c. X'00010A'
    d. X'03B678'
    e. X'03B688'

14. (True/False) The 'channel end'; 'device end' and 'traffic bit' are on (1) in this CCB.

15. The WAIT macro will always wait until _____ time, when using the following CCB: CCB SYSLST, PRINTCCW, X'0400'

    a. Channel-end
    b. Device-end
    c. Sense channel "12" on a 1403 carriage tape
    d. Start I/O

16. What value should be used in the third parameter of the CCB if the WAIT macro is issued to WAIT until device-end?

    a. X'8000'
    b. X'0004'
    c. X'0800'
    d. X'0400'
    e. X'FBFF'

17. If a programmer wishes to test for channel 12 on the printer, he must code the CCB to wait for _____ (channel end/device-end) and TM after his WAIT, TM CCB+___, X'___' before executing a skip to channel 1.

    a. Channel end, +4, 02
    b. Device end, +4, 02
    c. Device end, +4, 01
    d. Channel end, +4, 01
    e. Device end, +5, 02

18. The macro that must be coded by the programmer to terminate processing of a problem program is:

    a. STOP
    b. QUIT
    c. END
    d. TERMINATE
    e. EOJ

SESSION 9 - I/O MESSAGES

This session requires approximately 1.0 hours to complete.

This session covers the use of '0P' type messages to determine if an I/O failure is hardware or software. System low core messages will be discussed. The importance of system messages (printed or low core) cannot be over emphasized. The printed message is usually the first visual indicator to the system operator that a problem has occurred. The operator will probably show the CE the message if it is an I/O error message and ask him to fix the problem. DOS places all the pertinent information about the I/O failure in the error message. This information can be extremely helpful to the CE.

Another way an error may be indicated is when a system operator observes that the system is in the wait state. Prior to entering the wait state (due to a device error) the supervisor, if possible, will place a message in low core that describes the problem.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

*1. Use the physical I/O error message to locate any CCB for any I/O device.
*2. Given a physical I/O error message, locate the following items in the message.
   a. Message number
   b. Action code
   c. Operator code
   d. Message text
   e. Logical unit that failed
   f. Device address that failed
   g. Command code of the last CCW executed
   h. CSW
   i. Sense bytes
   j. CCB address
   k. Seek address is DASD
*3. When the system enters the WAIT state, locate and interpret the low core message and take necessary corrective action. (With the exception of the SEREP interface)

* Wholly or partially supported by follow-on lab.

Highlights

- When an error occurs while running under DOS, an error message is issued.
- There are two forms under which the error message occurs. They are the printed message and the low core message.
- The printed message format may contain information identifying the partition from which the error was initiated and the action code which identifies the action taken. The operator code - indicates action(s) required by the operator and may identify the failing unit.
  A low core error message code is placed in bytes 0 to 4 when DOS recognizes that it cannot issue a printed message. The system enters an uninterruptable wait state. (System must be IPLed)

Activity

Return to the terminal and submit an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 9)

1. If the system message '0P71I' is issued, the address of the CCB for the device that failed can be found:

   a. By displaying register 0 on the CPU.
   b. In the I/O old PSW.
   c. In the I/O new PSW.
   d. By obtaining a storage print and looking in register 1.
   e. None of the above.

Use the following error message for questions 2 and 3.

```
BG ØP11D   R DATA CHECK SYSLNK=190
           CCSW=1D00002EC80E400008 CCB=002E50 SK=000000BE0002
           SNS=080000C80000
```

2. Match the elements of the message with their corresponding values.

____	a.	Message number	1.	2EC8
____	b.	Message describing I/O error condition	2.	190
____	c.	Symbolic unit that failed	3.	11
____	d.	Address of hardware device	4.	080000C80000
____	e.	Sense data	5.	DATA CHECK
____	f.	CCB address	6.	SYSLNK
____	g.	Disk address that failed	7.	1D
____	h.	CCW command code that failed	8.	2E50
____	i.	Address of CCW that failed	9.	SYS190
			10.	08
			11.	000000BE0002
			12.	2EC0

3. The acceptable responses that an operator could make to this message are: (More than one answer is required)

   a. CANCEL
   b. IGNORE
   c. RETRY
   d. END/EOB
   e. Any of the above responses

4. Assume the system was running and it entered the wait state with interrupts masked off. The low core message in bytes 0-3 was displayed as follows: 07 E6 00 00 in hex. The reason the system entered the wait state was due to which of the following:

   a. Machine check
   b. I/O error queue has overflowed
   c. Channel failure
   d. IPL error
   e. Program check in the supervisor

5. Assume the system entered a hard wait with a low core message of X'C1E2E200'. Select the items below that are indicated by this message. (More than one answer is required)

   a. Unrecoverable machine check.
   b. Unrecoverable channel failure.
   c. Error recording was unsuccessful on SYSREC.
   d. Error recording was partially successful on SYSREC.
   e. Error recording was successful on SYSREC.

6. If a low core message of X'08C1001F' was displayed when the system entered the wait state the problem is probably:

   a. A data check on the card reader (00C).
   b. A data check on the console typewriter (01F).
   c. A busout check on the card reader (00C).
   d. A busout check on the console typewriter (01F).
   e. The console typewriter (01F) probably out of forms.

SESSION 10 - SUPERVISOR CONCEPTS

This session requires approximately 2.0 hours to complete.

This session describes the facility of the supervisor to handle I/O requests. The tables and control blocks used by the supervisor will be discussed. The linkage from the problem program to the supervisor and I/O device will be shown. Flowcharts will be used to show the basic concepts of a SVC 0 and the interrupt handling facility of the Supervisor.

This session is an extension of PIOCS macros and the I/O operation will be addressed in the supervisory area versus the problem program. The control blocks and tables that will be discussed contain information that can be helpful to the CE. This session will stress how the tables and control blocks are used for I/O operations and the next session will address how the CE locates and uses them.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

1. Identify the purpose of the following control blocks when used in the DOS/VS environment.
   a. PUB
   b. LUB
   c. Channel queue
   d. Communications region
   e. CCB
2. Given a layout of the components of an I/O operation, identify the sequence and logic flow of the following:
   a. PUB
   b. LUB
   c. CCB
   d. I/O device

*3. Identify the contents and/or usage of each item within the following tables:
   a. PUB table
   b. LUB table

* Partially or wholly supported by follow-on lab.

Highlights

- The supervisor's prime purpose is to control I/O operation through the use of tables.
- Channel Control Block (CCB) is used to pass information between problem program and supervisor.
- Logical Unit Block (LUB) reflects symbolic address and contains pointers to a physical unit address.
- Physical Unit Block (PUB) contains physical unit addresses and information relating to that device.
- Channel queue is used by the supervisor to remember requests for I/O operations, by placing the address of the CCB associated with an I/O request into this table.

Activity

Return to the terminal and enter an EOB to continue.

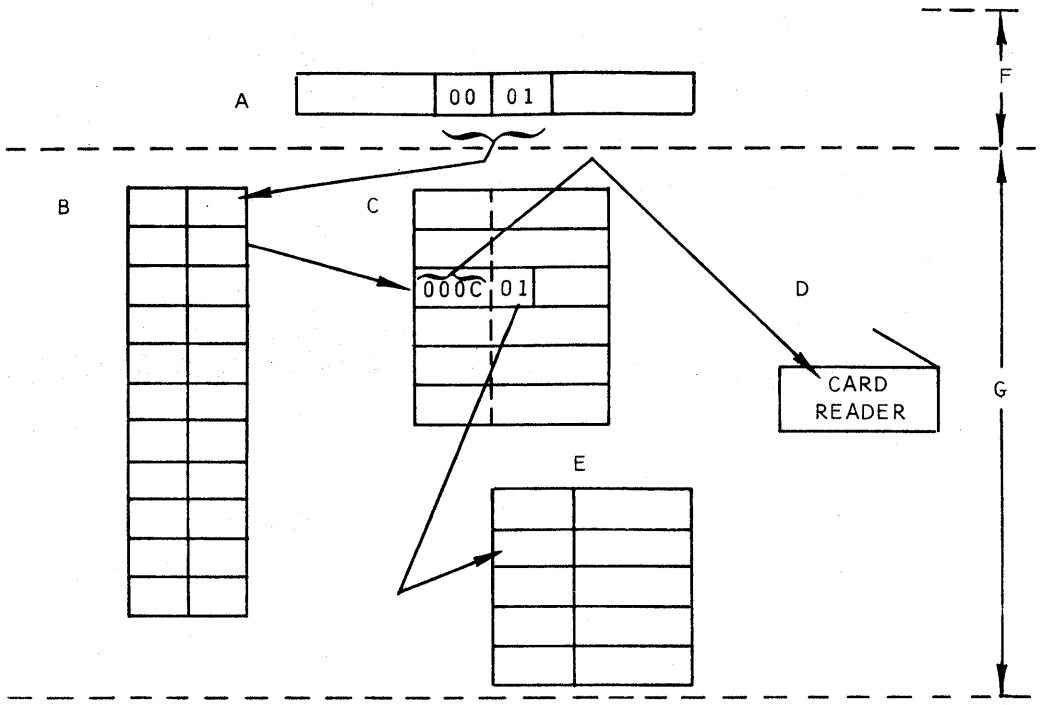
PRETEST/POST-TEST QUESTIONS (SESSION 10)

NOTE: Appendix A in the student guide contains a detailed layout of control blocks and tables.

1. Match the following blocks with their function as used by the supervisor.

____ a. PUB	1. Used to store the CCB address.
____ b. LUB	2. Contains the device address (CUU).
____ c. Channel queue	3. Contains addresses of the PUB and LUB table.
____ d. Communications region	4. Contains information concerning an I/O request.
____ e. CCB	5. Contains a pointer to the PUB table.

Use the following figure to answer questions 2 through 5.



2. Match the list of supervisor components to indicated block in the figure.

____ a.	Block A	1. Supervisor
____ b.	Block B	2. Problem program
____ c.	Block C	3. I/O device
____ d.	Block D	4. CCB
____ e.	Block E	5. Channel queue
____ f.	Block F	6. PUB
____ g.	Block G	7. LUB

3. The value of the pointer in table B to table C is:

a. '00'
b. '01'
c. '02'
d. '03'
e. '04'

4. The _____ is stored in bytes 1, 2, 3 of table E.

a. PUB address
b. CCB address
c. LUB address
d. Channel queue address
e. Device address

5. Block A is pointing to symbolic unit:

   a. SYSRDR
   b. SYSIPT
   c. SYS000
   d. SYS001
   e. SYSRES

6. In the PUB table, byte 6-bit 0 (device busy), for a device will be reset (0) when _____ occurs.

   a. unit check
   b. unit exception
   c. channel end
   d. device end
   e. a successful SIO instruction for a device

7. Byte _____ of the PUB can be checked for a non X'FF' condition to determine if an I/O request is queued up.

   a. 0
   b. 2
   c. 4
   d. 5
   e. 6

8. The address of the communications region (COMREG) can be found in storage at location:

   a. X'00'-X'03' in supervisor (low storage)
   b. X'14'-X'17' in supervisor (low storage)
   c. X'1000'-X'1003'
   d. X'00'-X'03' of the problem program
   e. X'14'-X'17' of the problem program

9. The storage protect key for the foreground 3 partition is _____.

   a. X'00'
   b. X'10'
   c. X'20'
   d. X'30'
   e. X'40'

SESSION 11 - SUPERVISOR TABLES

This session requires approximately 2.5 hours to complete.

This session teaches you how to locate components of the supervisor in a storage print. The contents of the communications region will be shown. How to determine the status of the I/O devices at an Abend will be shown. The format and how to use the error queue to locate failing devices will be discussed. The procedure to determine the program in control at an Abend will be discussed.

Information is provided to assist you in determining the cause of a problem. This session contains a project that will require you to locate all I/O components and should reinforce the supervisor concepts.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

*1. Given a storage print of a program that enters the wait state or aborts due to an I/O error, determine:
   a. Physical device that failed.
   b. Symbolic unit that failed.
   c. An entry in the Error Recovery Block.
   d. Failing CCW chain.
   e. CCB for the device that failed.
   f. I/O area for the device that failed.
   g. Last device started on a channel.
   h. PUB for any device.
   i. LUB entry for a symbolic unit.
   j. Partition in control of the system.

* Partially or wholly supported by follow-on lab.

Highlights

- The communications region contains addresses and pointers to various blocks used by DOS. The address of the communications region can be found by examining bytes 14 through 17 (hex) in storage.
- The error queue is used to save information received on a unit check. It contains entries of 44 bytes each.
- Channel bucket contains the PUB address of the last devices successfully started on a channel.
- Partition Save Area contains the PSW and registers for the last time a program was interrupted (lost control of the CPU).
- Partition Identification Key (PIK) indicates the partition in control of the system.

Activity

Return to the terminal and enter an EOB when you are ready to continue.

Directions to the Student for Pretest/Post-Test

Appendix D in the SCM contains a stand-alone storage print taken by a second shift operator when the system entered the wait state. The system was reading job control cards and printing the job control statements on both the 1403 and 1050. After the system entered the wait state, the operator took the storage print in Appendix D. The operator then tried to IPL the DOS system and it kept entering the wait state.

The first shift operator came in this morning and set the system up and has been running with no problems.

The shift supervisor gives you the storage print in Appendix D and asks what caused the system to stop during second shift. This is all the information that is available. Communications region address is X'278' taken from bytes X'16' and X'17' of storage.

Answer the following Pretest/Post-Test questions and when you have completed them, enter your answers via the terminal.

NOTE: Appendixes A and D are placed in SCM SR25-5673 to make it more convenient to answer pretest/post-test questions without flipping pages.

You should also refer to the DOS/VS SADP manual as required for descriptions of tables and control blocks.

PRETEST/POST-TEST QUESTIONS (SESSION 11)

1. The second shift operator, during his problem determination action, made the following mistake:

   a. Should have taken a DOS system dump.
   b. Should have done a LISTIO ALL
   c. Should have recorded the low-core message.
   d. Should not have re-IPLed.
   e. Should have taken a checkpoint.

2. The address of the PUB table in this supervisor is: (use COMREG)

   a. X'3234'
   b. X'38F0'
   c. X'3448'
   d. X'22EF'
   e. X'4D94'

3. The device with the address 00E in the PUB table is a:

   a. 1403 printer without UCS
   b. 1403 printer with UCS
   c. 3211 printer
   d. 1404 printer
   e. 1443 printer

4. The partition identification key (PIK) in the storage print indicates that the _____ partition has control of the system.

   a. Foreground 1
   b. Foreground 2
   c. Foreground 3
   d. Foreground 4
   e. Background

5. There is one device with an I/O request pending in the PUB table. It's hardware address is:

   a. 01F
   b. 00C
   c. 02F
   d. 151
   e. 282

6. The channel scheduler flags in byte 6 of the PUB entry for the device in question 5 indicates:

   a. device busy
   b. Switchable device
   c. I/O error queued for recovery
   d. Operator intervention required
   e. Device end posting required

7. The first error queue entry in the error recovery block indicates the address of the failing PUB entry is:

   a. X'5B58'
   b. X'2260'
   c. X'34B8'
   d. X'1606'
   e. X'6048'

8. In the LUB table, what symbolic unit(s) is assigned to the hardware device 150 in the PUB table. (More than one answer required.)

   a. SYSLNK
   b. SYSRES
   c. SYSREC
   d. SYS001
   e. SYS002

9. The first error queue entry indicates that the message _____ would be printed for this error.

   a. 0P10
   b. 0P14
   c. 0P15
   d. 0P56
   e. 0P40

10. The CCB address for the failing device is at location _____ in the storage print.

    a. X'34B8
    b. X'5B58'
    c. X'1510'
    d. X'00FF'
    e. X'6048'

11. The CCW operation code that failed is:

    a. X'00'
    b. X'01'
    c. X'03'
    d. X'07'
    e. X'09'

12. The EBCDIC contents of the first several bytes in the I/O area associated with the CCW that failed is:

    a. * STOPPED
    b. I AM LOOPING
    c. * RIGHT
    d. PAYROLL
    e. 00/00/00

13. Locate the 'channel buckets' (Sys Com X'30'). The last devices successfully started on channel 0 and channel 1 were: (two answers required)

a. 00E
b. 01F
c. 150
d. 180
e. FF0

SESSION 12 - STANDALONE AND SYSTEM UTILITY PROGRAMS

This session requires approximately 1.5 hours to complete.

This topic will describe the SEREP program operation and interface. The output of SEREP will be discussed to determine the type of failure that caused the system to enter the wait state. The operation of DITTO, DEBE and DASD print programs will also be shown. These programs can be used to operate devices under program control and to display the contents of disk and tape data files. The operation of the OS/DUMP restore program will be covered. This program can be used to copy information from disk to tape or tape to disk. DSERV, which is a DOS Librarian Service Function, can be used to display the directories of the various libraries. This information will tell you what programs are contained in the CI library.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

*1. When the system enters the wait state, determine if the SEREP interface has been set up; and if it has, run SEREP and interpret the output.
*2. Operate the DITTO program to reproduce a deck of cards.
*3. Operate the DEBE program to list a deck of cards.
*4. Use the stand-alone DASD print program to display any area of a 2319 disk pack and locate the following in the printout.
    a. Home Address
    b. Record Zero (Count and Data)
    c. Record 1 (Count, Key and Data)
*5. Use the OS/DUMP restore program to copy from tape to disk or disk to tape.
*6. Use the DSERV function to display the programs in the CI Library.

* Wholly or partially supported by follow-on lab.

Highlights

- When DOS enters the wait and byte 1 of storage contains X'E2', SEREP should be run to print out the failure indicators.
- DITTO is a type 3 program that functions as a multiple utility program.
- DEBE is either a standalone or system program that functions as a multiple utility program.
- The DASD Print Program can be used to display the contents of the disk packs.
- The OS/DUMP restore program can copy from tape to disk or disk to tape.
- The DSERV function can be used to display the directories of the libraries.

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 12)

1. Select from the list provided the items that describe the function of the SEREP.

   1. Provides a hardcopy listing of data pertinent to an error.
   2. Provides a limited storage print.
   3. Prints out the data stored on the recording file.
   4. Records System environment.
   5. Provides a storage print of control storage.

   Select one of the following (a, b, c, d, or e) when you input your answer for question 1.

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

2. What will byte X'73' of storage contain for a device error?

   a. 2F
   b. 1F
   c. 0F
   d. 3F
   e. FF

3. (True/False) If a run request is not present, SEREP checks bytes 3 and 4 of the I/O old PSW to see if a Stop After Log or Machine Check has occurred.

Refer to Figure 12.2 for the next question.

4. Select the correct run request that SEREP found when it was IPLed and produced this printout.

   a. device error
   b. channel error
   c. test channel failure
   d. device not operational

5. Select the correct DITTO function to perform a card to card with sequence numbers and deck name.

   a. CC
   b. CP
   c. CCS
   d. CH
   e. CTR

6. Select the correct DEBE control commands to punch data from a tape into cards. Tape drive 180 (9 track), and card punch 00D.

   a. CT and 00180
   b. CT and 180
   c. TC and 00180
   d. TC and 180
   e. TD and 00180

Refer to figure 12.10 for the following question.

7. Which of the following is the first data byte in R1.

   a. 00
   b. C7
   c. 04
   d. F4
   e. C8

8. (True/False) The restore function of the OS Dump/Restore program is used to copy from tape or card to disk.

9. Select the correct entry for the operand field of the DSPLY control card to display the core image library directory.

   a. TD
   b. CD
   c. RD
   d. SD

SESSION 13 - PROBLEM DETERMINATION AID (PDAID) PROGRAMS

This session requires approximately 2.0 hours to complete.

Problem Determination programs are available to help in determining the cause of errors. Two of these programs PDAID, with the I/O trace option; and the DUMPGEN program, are of special interest to the hardware CE. I/O trace has proven very helpful in locating intermittent I/O problems. DUMPGEN provides DOS standalone dumps which can format the data (Supervisor Tables and Low-core Data) that is extremely valuable in locating errors.

This topic will discuss the operation and outputs of:

- DUMPGEN Program
- Location of data in a DUMPGEN produced Formatted Standalone dump.
- PDAID with the I/O trace option.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

*1. Identify the control cards and parameters needed for the DUMPGEN program when given the user specifications.
2. Interpret the output of a formatted dump which was generated from the DUMPGEN program.
*3. Initialize and terminate the I/O trace PDAID via either the console typewriter or card reader.
*4. Interpret the output data that was produced as a result of an I/O event when running PDAID I/O trace.
5. Identify when to use the PDLIST program.
*6. Locate any I/O trace event which was recorded as the result of using the PDAID core-wrap mode.

* Wholly or partially supported by follow-on lab.

Highlights

DUMPGEN

- Formatting or non formatting standalone dump programs tailored to system requirements can be generated thru the use of the DOS DUMPGEN program.
- Output from the DUMPGEN program can be to either tape or card.

FORMATTED DUMP

- Data from low-storage locations and supervisor tables are formatted for ease of access, to aid in problem determination.
- A standard display of main storage precedes the formatted information.

PDAID (I/O Trace)

- The PDAID I/O trace (interrupt trace) can be used to record in sequence, all I/O interrupts and CSW's stored.
- Trace output can be contained in core through the core-wrap option.
- Trace data can be outputted to either tape or print.
- The PDLIST program is provided to print trace data recorded on tape.

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 13)

1. (True/False) If PPOOL=NO is specified in the DUMPGEN option statement, the boundary box contents will not be printed.

2. To have the DUMPGEN program create a stand-alone dump on tape, the desired tape drive must be assigned to:

   a. SYS001
   b. SYS005
   c. SYSLST
   d. SYSPCH
   e. SYSOUT

3. (True/False) The stand-alone dump will load into a non-critical area of the supervisor. The LOAD ADDRESS of this area is determined at generation time by DUMPGEN making it unique to that system.

4. (True/False) The operands of the DUMPGEN OPTN statement are separated by commas.

5. The data formatted in the output of the formatted standalone dump is/are:

   a. Communication Region
   b. Partition save area
   c. PCIL LUBS
   d. Error recovery block
   e. All of the above

6. (True/False) The PDLIST program when executed will print the data the PDAID program outputted to tape.

7. What response to "PDAID =" is needed to terminate the PDAID that is presently running.

   a. an EOB or END signal from the console typewriter
   b. XX
   c. IT
   d. STOP
   e. END

8. The displacement from the start of the PD area to the pointer which contains the address of the first entry when using the I/O trace.

   a. X'C0'
   b. X'F4'
   c. X'D0'
   d. X'198'
   e. X'114'

SESSION 14 - POWER CONCEPTS (PRIORITY OUTPUT WRITERS, EXECUTION PROCESSORS AND INPUT READERS)

This session requires approximately 1.5 hours to complete.

This session introduces DOS in a POWER environment. POWER provides the facility of spooling unit record I/O operations to disk or tape, which will be printed or punched at a later time. Many DOS customers are using POWER, therefore, an understanding of POWER, and how to service a hardware system where POWER is used, is essential.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

1. Select from a list of statements, the advantages of using POWER.
2. Identify from a given configuration of core storage, the partition(s) that are supported by POWER.
3. Arrange a list of items in the sequence required to run a job under POWER.

Highlights

- POWER increases job throughput.
- Unit record I/O is performed at disk speeds during job execution.
- Up to two partitions can be serviced by POWER.

Activity

Return to the terminal and enter an EOB to continue.

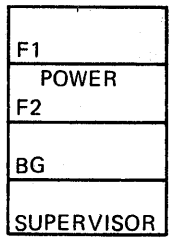
PRETEST/POST-TEST QUESTIONS (SESSION 14)

1. Select from the list below the advantages from using POWER. (More than one answer is required.)

   a. less CPU time is required to process a job.
   b. less core storage is required to process a job.
   c. less time is required to process a job.
   d. more jobs can be processed in a given amount of time.

2. Using the following core storage configuration, select the partition(s) that are supported by POWER.

   a. F1
   b. F2 and BG
   c. BG
   d. BG and SUPERVISOR
   e. F2



CORE STORAGE

3. Place the statements in the following list in sequential order to execute a card to print job under POWER.

   ____ a. the problem program signals POWER that it has completed processing.
   ____ b. the problem program issues a read a card command.
   ____ c. POWER intercepts the unit record I/O request and spools the print record on the PRINT QUEUE.
   ____ d. the Printer Task prints the records in the PRINT QUEUE.
   ____ e. POWER analyzes the job control statements in the INPUT QUEUE.
   ____ f. the problem program issues a print command.
   ____ g. the system loader loads the program from the core-image library into storage.
   ____ h. POWER signals the Printer Task that output in the PRINT QUEUE is available for printing.
   ____ i. the Reader Task builds the INPUT QUEUE with the data read from the card reader.
   ____ j. POWER intercepts the unit record I/O request and causes the data to be transferred from the INPUT QUEUE to the problem program.

4. Select the true statement.

   a. jobs under POWER are run in the sequence in which they are entered in the INPUT QUEUE.
   b. the Printer Task spools print records on the PRINT QUEUE.
   c. a priority can be assigned to a job so it executes ahead of other jobs already in the INPUT QUEUE.
   d. the Reader Task reads records from the INPUT QUEUE and passes them to the problem program.

SESSION 15 - POWER OPERATION

This session requires approximately 1.5 hours to complete.

It was established in the previous session that the CE might have the need to run diagnostic tests, such as the On-Line-Test programs in a POWER environment. This session describes the use of the operator commands and the POWER job control statements that are required to execute these programs.

OBJECTIVE

Upon completion of this session, the student, using the maintenance documentation, should be able to:

*1. Use the Queue management console commands to:
   a. Determine the status of any job in the POWER system.
   b. Delete a job from the INPUT Queue.
   c. Hold a job in the INPUT Queue.
   d. Release a job from the INPUT Queue.

*2. Prepare the POWER JECL statements that are required to execute a job with the following conditions:
   a. execute the job in the BG partition.
   b. assign the highest priority to the job.

* Partially or wholly supported by follow-on lab course.

Highlights

- POWER JECL statements provide the facility to:
  ASSIGN priorities to jobs.
  Put jobs in the HOLD state.
  Spool output to tape instead of to disk.
  Specify the partition a job is to execute in.
  Prevent spooling of output.
- POWER Console Commands provide the facility to:
  Display the status of jobs in the Queues.
  Alter the priorities of jobs.
  Put jobs in the HOLD state.
  Delete jobs from the Queues.
  Release jobs from the HOLD state.

Activity

Return to the terminal and enter an EOB to continue.

PRETEST/POST-TEST QUESTIONS (SESSION 15)

1. Which of the following functions can be performed using a POWER JOB statement? (More then one answer is required)

   a. assign a priority to a job.
   b. delete a job from the INPUT Queue.
   c. specify the partition the job is to execute in.
   d. hold a job in the INPUT Queue until it is released.
   e. cause the output to be spooled to tape instead of to disk.

Use the following POWER JECL statement as a reference when answering question 2.

```
* $$ JOB LIST,,2
```

2. Which one of the following statements is true?

   a. the jobname is JOBLIST, having a priority of 2 and will execute in the BG partition.
   b. the jobname is LIST, having a priority of 0 and will execute in the F2 partition.
   c. the jobname is LIST, having a priority of 2 and will execute in the BG partition.
   d. the jobname is * $$ JOB LIST, having a priority of 2 and will execute in the BG partition.

Use the following POWER JECL statement as reference to answer question 3.

```
* $$ PRT DA,16,8,
```

3. Which one of the following statements is true?

   a. the print output will be spooled to the disk whose address is X'168'.
   b. the print output is grouped into class A and will print 8 copies and form 16 is to be used.
   c. the print output is grouped into class A and will print 16 copies and form 8 is to be used.
   d. the print output is grouped into class DA and will print 8 copies and form 16 is to be used.
   e. the print output is grouped into class DA and will print 16 copies and form 8 is to be used.

4. Which one of the following commands would cause the report shown below to print on SYSLOG?
(assume a one partition POWER system).

a. D BGRDR, ALL
b. L F2RDR, ALL
c. R BGRDR, ALL
d. D BGRDR, SYSLOG
e. L BGRDR, SYSLOG

REPORT:

SAMPLE	00002	3		0000015
SORT	00003	7		0000070
PAYDAY	00004	5		0000475
ACCTSPAY	00005			0000009
LIST	00006	5	*	0000100

5. What does the * indicate in the report printed in the previous question?

a. the job has been deleted.
b. the job is in the HOLD state.
c. the job is currently executing.
d. the job is the next to execute.
e. the job just completed executing.

6. Which one of the following commands will prevent the jobs with a priority of 6 from executing? (assume a one partition POWER system).

a. STOP BGRDR, 6
b. HALT BGRDR, 6
c. H BGRDR, 6
d. STOP ALL, 6
e. H ALL, 6

SESSION 1 - INTRODUCTION TO AN OPERATING SYSTEM

SECTION 1-1 - COURSE INTRODUCTION

If you are enrolled in the course as part of a "Hardware Training" sequence, you are probably asking this question: "Why do I, a hardware CE, need to take what appears to be a Programming Systems course?".

This is a valid question, and the following text will attempt to answer it.

The data processing industry has undergone many changes during the tremendous growth period of the past fifteen years. Each succeeding system generation, since the announcement of the 650 Data Processing System during the mid 50's, has offered the customer improvements in speed, memory sizes, variety of input/output devices and programming languages. But the most significant of all changes that have occurred, and the controlling factor in many related areas, has been the transition to more and more online processing activities as opposed to the more traditional batch accounting functions.

These online activities have made the availability of a data processing system a very essential factor in a company operation and a complete system failure is extremely disruptive. When a batch accounting system is down, one customer account is affected. But consider the impact of that system operating in an online teleprocessing application. A company's operation may be affected nationwide (eg, our own CAI system), or even worldwide (an airline reservation system), or many different industries may be impacted in the case of a shared system.

Because of this changing environment, there is a competitive necessity for IBM to ensure its customers greater systems availability with each new generation. During early System/370 planning, IBM developed a Reliability, Availability, and Serviceability (RAS) strategy.

While only some RAS strategy features were incorporated into System/360 (those that are implemented through software), System/370 was designed to operate in a real time environment and has incorporated an intensive group of these features. Some objectives of these features are to ensure the customer maximum system availability by reducing the frequency and impact of systems interruptions, due to hardware failures.

This is being met by the following:

- Hardware reliability is improved through use of fewer failure prone components. While maximum availability would be achieved if no hardware failures occurred, this level of reliability is not economically feasible.

- Recovery facilities (implemented through both hardware and software) are incorporated to reduce the number of times that a hardware failure will result in complete system termination.

  For many types of failures, these facilities will allow the system to continue operation, and will provide sufficient information to allow the CE to repair the failure concurrent with customer operation or to defer the repair until complete system outage will have minimum impact on the operation.

- Repair Procedures must be implemented through more online diagnosis and repair. This necessitates a CE to operate in an online multiprogramming environment, but does reduce the effect of such repairs on systems availability.

In summary:

- More reliable components should result in fewer hardware failures.
- When a failure does occur, the repair should be made concurrent with customer operation or deferred to a more convenient time.
- If the failure results in total system outage, it must be repaired as quickly as possible.

Those new skills imposed on the hardware CE by the RAS strategy, which may have formerly been considered to be software skills, are now primarily related to the concurrent/deferred maintenance philosophy.

Concurrent Maintenance

Concurrent maintenance is most applicable to the maintenance of input/output devices that the customer can do without, and still continue system operation while the CE repairs the problem. The CE will have the capability to do online diagnosis problems, effect an offline repair, and verify the repair online.

Deferred Maintenance

Deferred maintenance is most applicable to intermittent errors that a system has the ability to recover from and still continue operation (even though in a degraded mode) until the end of a job or a scheduled maintenance period.

To enable concurrent and deferred maintenance, and a rapid diagnosis of any system interruption, the following operating system facilities are extremely valuable to the hardware CE.

- I/O error messages
- Statistical data for:
  - Device failures
  - Channel failures
  - CPU failures
  - Media failures
- Wait state codes
- Interfaces to CE logout programs
- Online tests for I/O devices
- Problem determination charts and aids, primarily for the customer, but valuable to the CE.

This course will show you how to use these facilities to identify and repair failures. To effectively use these you must be able to:

- Operate the system used by your customer.
- Communicate with personnel, both customer and IBM, about the operating system environment.
- Use operating system messages, storage prints (dumps), and accumulated error data etc, as an aid in problem determination and definition.

Purpose of Course

There are several courses in your training sequence which enable you to service a System/370 in the environment and level previously discussed.

This course concentrates on the Disk Operating System (DOS) and will provide you with a knowledge of DOS concepts, organization, and operation that is required to determine the general cause of any problem in a total system operation.

Problem determination will consist of the ability to utilize available indications or data provided by the system in the form of messages, storage printouts, I/O tables, utility programs, etc, to determine if a problem is:

- Hardware, software, or storage media.
- If hardware, the type of error and the unit that failed.
- If software, same probability as to whether it was a user program or an IBM supplied program. (A program Support CE may be required for a positive determination.)

Future courses in your sequence will teach you how to use the specific maintenance aids provided by DOS that will be required to further define hardware problems and how to operate the system to perform concurrent maintenance.

Summary

IBM is providing its customers with Data Processing systems that have the hardware and software capabilities to give them the systems availability they require. But once that system is in the customer's office, it is up to the FE Division and you, the CE, to ensure that any system problems have the least possible impact on that availability.

Just as a system is a combination of hardware and software, so must your knowledge of the system be, if you are to achieve this.

Answer the following study questions and when you have completed them, return to the terminal and enter an EOB.

Study Questions (Section 1-1)

1. The RAS strategy for System/370 was implemented by improvements in the three areas of _____ , _____ , and _____ .

   a. Hardware reliability
   b. Faster hardware
   c. Recovery facilities
   d. Repair procedures
   e. Operating system reliability

2. The hardware CE needs software skills to implement the _____ / _____ maintenance philosophy.

   a. Non-concurrent/non-deferred
   b. Concurrent/deferred
   c. Non-concurrent/deferred
   d. Concurrent/non-deferred

3. Deferred maintenance is most applicable to _____ errors.

   a. Solid
   b. Intermittent

4. (True/False) The purpose of this course is to provide training for the hardware CE, so he may take trouble calls on IBM programs.

Answers for Section 1-1 Study Questions are on the next page.

Return to the terminal and enter an EOB.

Answers for Section 1-1 Study Questions

1. The correct answer is A, C, and D.
   Answer 'B' is a marketing strategy for System/370; it is not part of the RAS strategy.

   Answer 'E' has always been a concern of IBM, but is not part of the RAS strategy.

2. The correct answer is B.
   IBM will provide Concurrent/Deferred maintenance wherever possible.

3. The correct answer is B.
   Deferred maintenance is only applicable to intermittent errors. Solid errors cannot be usually deferred.

4. The correct answer is FALSE.
   This course is designed to provide you with the skills required to use (converse) with an operating system when performing concurrent maintenance.

SECTION 1-2 - SYSTEM CONCEPT REVIEW

Before proceding, there are a few terms that must be reviewed before any new material is covered.

SYSTEM

There are many definitions for the term SYSTEM in a dictionary. A few of the definitions are:

- A regularly interacting or interdependent group of items forming a unified whole (for example, a numbering system).
- A group of related natural objects or forces (for example, a system of rivers).
- A group of devices or artificial objects or an organization forming a network established for distributing something or serving a common purpose, (for example, a heating system, telephone system, or highway system).

Using the above definitions, it can be seen that a system is made up of a group (more than one) of items that interact or work together to accomplish something.

To summarize the above definitions, it can be said that a SYSTEM is an organized method used to accomplish a task.

Let's look at a typical system we are familiar with. Our first example is a car wash. Refer to Figure 1.1, and the following description.

1 This system is a car wash. This system is an organized method used to accomplish the task of washing a car. The method of doing the task is controlled by a 'black box'. The 'black box' must be informed or directed concerning the task to be completed. This is done by a coin box in this example, 2 labeled 'software'.

2 The customer selects the options desired 'wash no soap' or 'wash with soap' and provides money to initiate the logic. This information from 2 is processed by the 'black box' and the 'black box' now directs and controls the hardware I/O 3 that accomplishes the physical task of washing the car. The hardware I/O consists of a transport chain to move the car, cleaning brush, switches to monitor the operation, etc.

It can be seen that the car wash system, accomplished in a organized method, using a 'black box' that relies on software (coin box) and hardware I/O, the task of washing a car.

Also shown in Figure 1.1 is a computer system. Let's take a close look at what makes up a computer system.

4 This block represents a CPU, one of the elements of a computing system. The function of this unit is to process data under control of software programs 5 .

5 This represents the software programs that control CPU operation. The software programs inform the CPU when input data is required from the hardware I/O 6 . The CPU then processes data (calculates, edits, etc) under control of software programs.

When a record needs outputting, the software programs instruct the CPU to write the data on the hardware I/O devices.

6 This represents the hardware I/O devices that provide input data (read) or output data (write) from the CPU under control of software programs.

Summary of a Computer System

- In a computer system, we have both hardware and software.
- The hardware (CPU and I/O) accomplishes the physical task, and software provides the instructions, with various options to accomplish this task.
- The software tells the CPU from which device to get the input.
- Hardware physically makes this input available to the CPU.
- Software solves the problem (compute, edit, etc) and informs the CPU what hardware device to use for its output.

A SYSTEM is an organized method used to accomplish a task.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 1-2)

1. In Figure 1.1, the coin box item 2 of the car wash serves the same function as _____ in the computer.

   a. The hardware I/O
   b. The software programs
   c. The CPU

2. (True/False) A computing system is the combination of hardware and software working together to accomplish a task.

SECTION 1-3 - OPERATING SYSTEM

During the review of a computing system, it was established that:

Hardware
+ Software
System/370

Let's take a closer look at the SOFTWARE portion of a computing system. Most System/370s use some form of an operating system to accomplish the software functions. Two of the most common operating systems used in the field are OS (Operating System) or DOS (Disk Operating System). During this session, there will be no attempt to separate or further define these operating systems as applied to System/370 because only the basic concepts of an operating system are being covered and these concepts are the same for both operating systems.

The first question to be answered is:

What is an operating system?

An operating system is: 'A system (software) that makes the hardware easier to use, with a minimum amount of required intervention.'

Previously, we defined a 'SYSTEM' as a group of items that interact or work together to accomplish something. So it could be said that an operating system is a group of programs combining (working) together to do useful work for a customer (solves problems, prints reports, etc).

Refer to Figure 1.2 1 and observe that an operating system consists of programs. Components of an operating system are broken into two major program categories:

Control Programs
Processing Programs

CONTROL PROGRAMS

Control programs (Figure 1.2 2), as the name implies, control or monitor the use of the computer system. Normally, these programs are supplied by IBM as a part of an operating system.

PROCESSING PROGRAMS

Processing programs (Figure 1.2 3) are problem solving type programs. Processing programs are either IBM written or customer written programs. Refer to Figure 1.3 4 and observe that a customer written program could consist of a payroll program used to compute and write employee paychecks. There could also be an inventory program that controls the level and ordering of parts created or used by the company. In addition, there are IBM written processing programs; these will be covered later.

The purpose of a control program (Figure 1.3 2) is to support the programs written or used by customers using a computer system. Using a control program, 2 , processing can progress from one customer program (payroll 1), to another (inventory 1) without any operator intervention.

Advantages of an Operating System

As previously mentioned, one advantage of an operating system is a minimum amount of intervention is required. This means that when one program (payroll) has completed, the next program (inventory) will be automatically processed without operator intervention.

Another advantage of an operating system is that it makes optimum use of CPU system components (resources). See Figure 1.4. Some of the system resources managed or monitored by the operating system are:

- I/O devices
- CPU time
- Channels
- Storage
- Media (tape reels and areas on disk packs)

Due to this constant monitoring and optimization of system resources, the customer gets more work from his computer system.

The customer uses an operating system to make the best use of a computer and the people who work with it.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 1-3)

1. Control programs:

   a. Are supplied by IBM.
   b. Manage hardware resources.
   c. Monitor overall system operation.
   d. All of the above.

2. Processing programs:

   a. Are problem solving programs.
   b. Manage system resources.
   c. Monitor overall system operations
   d. None of the above.

3. An advantage of an operating system is:

   a. It is compact.
   b. It makes optimum use of system resources.
   c. That all the programs are classified as control programs.

SECTION 1-4 - OPERATING SYSTEM CONCEPTS

To make optimum use of the CPU system components, the following control programs are required:

- IPL
- SUPERVISOR
- JOB CONTROL (job management)

Control Programs

Refer to Figure 1.5 [1] and observe the control programs. The following is a brief description of their functions as relates to an operating system.

IPL

- Initializes the system (for example, clears core [2])
- Loads the supervisor control program into core [2].

SUPERVISOR

- In core [2], at all times after IPL, acts as a monitor.
- Works in conjunction with all other programs [3].

JOB CONTROL

- In core between job steps (when a program has been completed, Job Control is loaded into core).
- Provides job-to-job transition (batch processing). For example, when the customer program PAYROLL finishes, Job Control will be placed in core. Job Control can be informed (through control cards) that the next customer program to be executed is INVENTORY. Job Control will initiate the system action required to have the program INVENTORY placed in core.

SECTION 1-5 - PROCESSING PROGRAMS (other than customer written)

IBM supplies the customer with several language translators as a part of an operating system (Figure 1.5 3). These language translators are provided to make it easier for the customer to write his programs. The language translators translate programs written in a programming language to machine language. Listed on Figure 1.5 3 are the four most commonly used language translators.

- ASSEMBLER
  A machine oriented language of symbolic codes used by the programmer to designate operations that the computer is to perform. The ASSEMBLER translates the symbolic codes to machine language. Refer to Figure 1.6 1 .

- COBOL (COmmon Business Oriented Language)
  The programmer writes a program using English language statements. The COBOL compiler translates these English language statements to machine language. This compiler is used to solve commercial problems (payroll, inventory, etc). Refer to Figure 1.6 2 .

- FORTRAN (FORmula TRANslation)
  Scientific application language coded by use of formulas. These formulas are translated into machine instructions. Refer to Figure 1.6 2 .

- PL/I
  Combines the features of COBOL and FORTRAN with added features and capabilities of its own. Refer to Figure 1.6 3 .

Now let's look at the advantages and differences between these languages. Figure 1.7 shows examples of the various languages. In each example two data fields labeled (named) B and C are added together and the answer stored in another data field labeled A.

First, observe the FORTRAN/ PL/I example in Figure 1.7 1 . To perform the function described above, the programmer needs to code only one source statement. This statement is called an ASSIGN statement. It assigns the field A the sum of B + C.

The next example is a COBOL example on Figure 1.7 2 . The programmer is required to code a little longer statement to do the same function performed by FORTRAN/ PL/I 1 . This is an easy language to use and very popular with our customers because it has been designed for programming of commercial problems.

The next example in Figure 1.7 3 is familiar to all of us. This is an example of assembler language. This example points out two major differences between ASSEMBLER and the three previous languages mentioned in 1 and 2 . The most obvious difference is the number of source statements the programmer must code to solve the problem. In this case, the programmer must code three statements versus the one statement discussed in the previous statement. For assembler language, the programmer coded:

- An instruction to load the value from the field B into general register 5.
- Then the value from the data field C was added to general register 5, which contained the value from field B.
- Finally the last instruction stored general register 5, which contains the sum of B and C into the data field A.

The second difference is the amount of knowledge the programmer must know about the CPU and hardware I/O before he can code effectively. For example, if a programmer coded a program in COBOL to run on a 1401, it would be for all practical purposes, coded exactly the same in COBOL to be run on a System/370.

This is not true for ASSEMBLER language. A program written in ASSEMBLER language for System/370 wouldn't run on a 1401. One obvious reason is that a 1401 doesn't have general registers like System/370.

In summary, we can state that to code PL/I, FORTRAN or COBOL the programmer need concern himself only with writing (coding) a solution to a problem and not concern himself with the hardware aspects of the program. Literally speaking, a PL/I, COBOL or FORTRAN program, when compiled, will run on any CPU it is compiled on.

This is not so for ASSEMBLER language. The programmer must understand the CPU instruction set and hardware functions before he can effectively code in ASSEMBLER language. From all that's been said against the ASSEMBLER, it would seem there are no advantages, but this isn't so. In most cases, a program coded in ASSEMBLER language requires less core and executes faster than programs coded in PL/I, COBOL, or FORTRAN. This is because PL/I, COBOL, and FORTRAN use generalized routines that must handle multiple functions which increases the amount of machine instructions used to do any one function.

The programmer, though, can code the solution to the problem faster in PL/I, COBOL, or FORTRAN than in ASSEMBLER language.

One of the tradeoffs that a customer makes when he elects to use PL/I, COBOL, or FORTRAN instead of ASSEMBLER is that problems are coded and solved quicker by the programmer (reduce program coding time) but the programs use more core and execute slower.

The last example on Figure 1.7[4] is machine language. None of our customers code in this language but this is the language that makes the computer go. The four previously mentioned languages convert the source statements to machine language for execution in the CPU.

Answer the following study questions, and when you have completed them, input your answers on the terminal.

Study Questions (Section 1-5)

1. The language translators that are most applicable for a programmer to solve scientific problems are _____ and _____.

   a. Machine language
   b. ASSEMBLER
   c. COBOL
   d. FORTRAN
   e. PL/I

2. If a customer had a problem to solve that required fast execution and he desired to compile a program using a minimum of core, the _____ would be the best language translator to use.

a. ASSEMBLER
b. COBOL
c. FORTRAN
d. PL/I

SECTION 1-6 - MACRO/MICRO

There are two terms that are often confused when talking about programming. They are microinstructions and the assembler language macro statement.

Refer to Figure 1.8, using the following description.

1 This represents an ASSEMBLER language instruction. The ASSEMBLER translates this to a machine instruction 2 .

2 This represents one machine instruction (add Register) generated by the ASSEMBLER. This instruction will be executed within a System/370 CPU that uses a microprogram.

3 The microprogram uses many microinstructions within the CPU to perform the one machine instruction.

4 This represents an ASSEMBLER language macro statement. (NOTE: PRINT is not a mnemonic operation code). This macro statement causes the ASSEMBLER to generate many machine instructions 5 .

5 There are many machine instructions generated to perform the function of the PRINT macro. (Macros covered in depth, later in course) Each machine instruction generated requires many micro instructions (2 and 3 .) to be executed 6 by the CPU.

SECTION 1-7 - IMPLEMENTATION OF A PROGRAM

A customer uses our data processing equipment mainly to solve his problems. The problem can be in the form of printing payroll checks, solving engineering problems, printing reports for management to use for decision making, inventory control, etc.

To solve a problem, the customer uses a program that works in conjunction with the computing system to provide a quick and accurate answer.

Because of the importance a program plays in solving a customer problem, let's take a close look at how a customer implements a program in an operating system environment.

Refer to Figure 1.9. The associated text sequentially describes how a program is implemented.

1 The programmer is informed of the problem and codes a solution to the problem on a coding sheet using some programming language (this example shows ASSEMBLER and COBOL).

2 The coding sheets are given to a keypunch operator who will punch the information in IBM cards. The deck of cards, when punched, will be called a 'source deck'.

3 The deck of cards from the keypunch that contain the programmer's source statements (source deck) is input to a language translator 4.

4 The language translator (ASSEMBLER or COBOL) reads in the source deck and translates the source statements into machine instructions. The language translator then outputs an object module 5.

5 This object module may be in the form of a deck of cards. (Normally in an operating system environment, the object module is written on disk or tape.)

This object deck, produced by a language translator, is <u>not</u> in a form that can be executed directly by an operating system. Also, the object module may need to be combined with other modules or relocated to a specific core location other than the core locations assigned when run through the language translator. This function is accomplished by the linkage editor program.

6 The object module is input to the service program called linkage editor. The object module can now be combined with other object modules and be relocated to a specific core address. The output of the linkage editor 7 is in executable format and can now be brought into core.

7 This loadable program text, the output from the linkage editor, is now in a format that can be executed under control of an operating system.

8 Shown here is an operating system supervisory routine called a system loader (the name of this routine varies between operating systems). The system loader takes the output of the linkage editor and loads (places) it in CPU core 9.

9 Once in core, control is transferred to the problem program and it begins executing.

The above conceptually describes program implementation in an operating system environment. In figure 1.9, the implementation stages are broken down into four unique times. It is important that you understand and remember these four times, because they will be constantly referred to throughout the remainder of the course.

These times are as follows: (Refer to Figure 1.9)

A PROGRAM DEVELOPMENT TIME

This is the time during which a program is developed (coded) and a source deck is produced.

B COMPILE TIME

Refers to the conversion of a source deck by a language translator (ASSEMBLER, COBOL, etc.) to an object module containing machine instructions.

C LINKEDIT TIME

Refers to the time during which an object module (the output of a language translator) is converted to executable and loadable program text by the linkage editor.

D EXECUTE TIME

Refers to the time during which a loadable program text (problem program) is placed in core, receives control, and executes the machine instructions within it to solve a customer problem.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 1-7)

1. Converting a source deck into an object module is considered:

   a. compile time
   b. program development time
   c. execute time
   d. linkedit time

2. The writing of instructions on a coding form is considered:

   a. compile time
   b. program development time
   c. execute time
   d. linkedit time

3. Processing of the actual instructions of a program is considered:

   a. compile time
   b. program development time
   c. execute time
   d. linkedit time

4. Converting an object module into its executable form is considered:

   a. compile time
   b. program development time
   c. execute time
   d. linkedit time

5. A program is loaded into storage for execution:

   a. after it is compiled.
   b. after it has been assembled.
   c. after it has been linkedited.
   d. after it has been coded.

SECTION 1-8 - SERVICE PROGRAMS

IBM provides service programs as a part of an operating system (Figure 1.10 1). Service programs perform those functions required by a majority of operating system users.

Several of the most commonly used service programs are:

- Linkage Editor
  - Used to convert object modules into executable format.
  - Provides for relocation of programs. (For example, a problem program (object module) may be prepared by the linkage editor to execute, starting at address 130K in storage one time, and the next time the program is to be executed, the linkage editor could prepare it to execute starting at address 256K. This feature of relocation is essential in an operating system environment. Remember that the operating system makes optimum use of the computing system's resources, and storage (core) is a resource. If the program that is being discussed (INVENTORY) could only execute at 130K and this area was being used by another program (PAYROLL), Figure 1.11 1 our program (INVENTORY) 2 would have to wait until PAYROLL had finished.

    But suppose there is an unused or free area 3 in storage from 256K to 512K. Should this area remain unused until a program that runs at 256K is ready to execute? NO. The operating system will linkedit (relocate) INVENTORY to run at 256K and then execute the program. Refer to Figure 1.12 and note that the program INVENTORY 2 is placed in core following the program PAYROLL 1 . This is a good example of the resource management capabilities of an operating system.

- Sort
  - The Sort program enables the customer to sort multiple files of randomly ordered records or to merge multiple files of sequenced records into one sequential file. Sequencing is performed by comparing specified control data fields within the records. Records may be sorted or merged in ascending or descending order. This service program is used by most of our customers that operate under an operating system.

- Utilities
  - The utility programs copy data files from one storage medium to the same or another medium. For example, records on a tape can be copied to another tape or to a disk or to a printer. During the copying, the records can be reblocked, unblocked, or the order of the data fields in the records may be rearranged.
  - There are special purpose utilities such as Initialize Disk, Initialize Data Cell, Copy-Restore, EREP, etc.

Answer the following study questions and when you have completed them, input your answers on the terminal.

Study Questions (Section 1-8)

1. The Sort program is a:
(More than one answer is required)

   a. control program
   b. processing program
   c. customer written program
   d. language translator program
   e. service program

2. Match the programs with their function within an operating system.

Programs	Function
___ a. Linkage Editor	1. Provides for relocation of programs.
___ b. Sort	2. Used to merge multiple files of sequential records into one sequential file.
___ c. Utilities	3. Can be used to initialize a disk pack.

Select one of the following (a, b, c, d, or e) when you input your answer for question 2.

   a. 1,2,3
   b. 3,2,1
   c. 2,1,3
   d. 1,3,2
   e. 3,1,2

SECTION 1-9 - SYSTEM RESIDENCE PACK (SYSRES)

Many programs were discussed during the previous sections. Those programs represent a minute portion of an operating system. For the operating system to operate effectively, it must be able to access these programs quickly. If every time an operating system needed a program, it had to notify the operator to place that program deck in a card reader so the program could be loaded into storage for execution, the time required would be prohibitive. For this reason, most operating systems use a Direct Access Storage Device (DASD) to store the software systems (programs). Refer to Figure 1.13 1 and note that both the control and processing programs, reside on disk (SYSRES).

Use of a DASD device for SYSRES facilitates quick access of programs for the operating system. DASD devices, because of their capacity to store large amounts of data (in this case programs), allow accessing of a large number of programs by the operating system. The SYSRES pack must be online (available) to the operating system at all times.

The system residence pack (SYSRES) is separated into several libraries to support various operating system facilities. At this time, only one library will be discussed. This will be the library that programs can be executed from. The name of this library varies between operating systems. In this topic, the library that programs can be executed from will be called the Core Image Library.

Refer to Figure 1.14 and the following text for a description of this library and its function within the operating system.

1 The Core Image Library is normally located on SYSRES, a DASD device. This library is defined by the customer, and is variable in size as we will see later.

2 The Linkage Editor program prepares the object modules (output from language translators) for execution under control of the operating system. The Linkage Editor places its output (loadable program text) in the Core Image Library 1.

3 At execution time, the system loader routine in the supervisor goes to the Core Image Library 1 for the problem program to be executed and places it in storage 4.

4 After the problem program has been placed in core, the operating system transfers control to it, and execution begins.

5 There may or may not be additional operating system libraries on SYSRES.

Core Image Library Summary

- This library is a part of SYSRES and its size is determined by the customer.
- The Linkage Editor places its output (a program) in this library, in a format that is executable, under control of an operating system.
- This library is used to store executable programs.
- The system loader goes to this library when a program is to be executed and places the program in storage.
- Some of the programs found in this library are:
  - Control programs
    1. IPL
    2. SUPERVISOR
    3. JOB CONTROL
  - System Service Programs
    1. Linkage Editor
    2. Sort
    3. Utilities
    4. Librarian
  - Processing Programs
    1. Customer programs
    2. Language translators

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 1-9)

1. The program which places the object modules into the core-image library is the:

   a. Supervisor.
   b. Linkage Editor.
   c. System Loader.

2. The program which locates (finds) a program in SYSRES and loads it into CPU storage is the:

   a. Service Program.
   b. Linkage Editor.
   c. System Loader.

3. Programs found in the core-image library can be classified into:

   a. Control Programs
   b. Service Programs
   c. Processing Programs
   d. All of the above

SECTION 1-10 - CONCEPTS OF SYSTEM CONTROL PROGRAM FUNCTIONS

The function of the control programs, IPL, JOB CONTROL, and SUPERVISOR, will be shown as related to an operating system.

As discussed earlier, the main functions of these programs are:

- IPL
  Only in core once. Initializes the system and loads the supervisor.
- SUPERVISOR
  Always in core after being loaded by the IPL program. Works in conjunction with all other programs.
- JOB CONTROL
  Only in core between job steps. Interprets control cards.

The control programs all function together to accomplish the customer's work (execute processing programs). To place the control programs in proper perspective and show their interrelationship, Figures 1.15 through 1.18 will be used. The following text will sequentially lead you through the operation. Emphasis will be placed on the individual control program functions within the operating system.

Now refer to Figure 1.15 and read the text that follows:

1 To IPL an operating system, the operator sets up the address of the physical unit to be IPLed (SYSRES) in the appropriate CPU switches. Then the operator presses the 'load' key on the CPU. Hardware IPL (microprogram) loads the software IPL program **2**.

2 The software IPL program is contained on the system residence disk pack (SYSRES) in the Core Image Library.

3 The software IPL program is loaded into low core by the hardware IPL (microprogram).

4 After the software IPL program is loaded into storage, it performs general housekeeping (for example, it clears storage to X'00's).

5 IPL then moves (relocates) itself to a higher location in storage in order to provide room for the Supervisor at the low end of storage.

6 The IPL program, when relocated in high core, **6** then goes to SYSRES to locate the Supervisor program. Refer to Figure 1.16 **6**, **7**, **8**.

7 When the Supervisor program **8** is located on SYSRES, it is loaded **9** into core **10**.

10 The Supervisor now resides in low core and will remain there permanently. The prime functions of the Supervisor are:

- schedules I/O requests
- performs error recovery routines
- provides for operator communications
- process interrupts
- loads problem programs (system loader)

11 After the Supervisor has been loaded, the IPL program has completed its main function. The IPL program has now initialized the system and established an operating system environment. The next program required is Job Control. IPL now informs the supervisor (system loader routine) that Job Control is required.

Refer to Figure 1.17 12.

12 The system loader routine in the Supervisor then locates (on SYSRES) the Job Control program 13. The Job Control program is then loaded into storage 14 and control is then transferred to Job Control 15 and it begins execution.

15 Job Control's function, within an operating system, is to minimize operator intervention when running batch jobs. Job Control will enable the system to continue processing without stopping after every job. Job Control will set up conditions pertaining to the job being run.

The object of Job Control is to control the execution of programs. To accomplish this function within the operating system, Job Control has to know:

- Which program must be run.
- Which partition will it be run in (multiprogramming).
- Which options are required by the program.
- Which I/O devices are needed.

Job Control obtains the above information from control statements (cards) submitted by the operator via Job Control's input device 16.

16 Job Control automatically goes to its input device (card reader) to read control statements. Therefore, if control statements for several jobs are stacked in Job Control's input device, Job Control can provide automatic job to job transition. Refer to Figure 1.19.

Refer back to Figure 1.17.

17 This is an example of a control statement that Job Control may encounter from its input device. Job Control would read and analyze the control statement. The control statement shown here is an execute statement (EXEC). This statement instructs Job Control that the next problem program to be executed is PAYROLL.

18 Job Control would then inform the Supervisor (system loader) that the program PAYROLL must be loaded into storage.

Refer to Figure 1.18 19.

19 The Supervisor would then go to the Core Image Library to locate PAYROLL. PAYROLL is then loaded into storage 21 overlaying Job Control 15.

22 PAYROLL will start execution and perform the operations that the programmer coded in the program. For example, printing payroll checks may be one of the functions the programmer coded. When the program has completed execution, it will signal this to the supervisor. The operation from this point is identical to that which occurred when IPL finished execution in Figure 1.16 11. If you follow the sequence of the blocks again, the only significant change will be that step 16, Job Control will process the cards for job number 2.

Summary

During this session we answered the following questions:

What is a System?
- **An Organized Method Used to Accomplish a Task.**

What is an Operating System?
- **A Group of Programs Combining Together to Do Useful Work With a Minimum Amount of Intervention.**

Why Do We Have One?
- **To Optimize the Effectiveness of the Computer Components.**

[FE 111101]

As an optional reading assignment you may desire to refer to the manual "Introduction to DOS/VS" (GC33-5370) and read all of Part 1 - "What is a Disk Operating System," and Part 2 - "The Functions and Facilities of DOS/VS" through but not including "Resource Utilization."

Answer the following study questions and when you have completed them, input your answers on the terminal.

Study Questions (Section 1-10)

1. The IPL program loads the Supervisor into core from:

   a. tape drive
   b. card reader
   c. SYSRDR
   d. SYSIPT
   e. SYSRES

2. (True/False) Job Control interprets control cards to determine the program to execute and then physically loads that program in storage.

3. In an operating system environment, after a problem program has completed execution, the problem program:

   a. goes to the wait state.
   b. goes to SYSRES and loads Job Control into storage.
   c. goes to SYSRES, and loads IPL into storage.
   d. reads control cards to determine the next program to run.
   e. signals the Supervisor it has finished.

SESSION 2 - SYSTEM CONCEPTS

SECTION 2-1 - REFERENCE PUBLICATIONS

To function effectively, an operating system requires the talents of several kinds of people, such as: system analysts, programmers, operators, etc. DOS/VS reference publications are available to each, providing them with information needed to do their specific job.

These publications are also required by you when servicing equipment in a DOS/VS environment.

Several of these publications are included with this course. As you proceed through this course, keep in mind that they are reference publications. In many cases you will be directed to sections of the publications in this course and your major goals should be to:

1. Develop a general understanding of the contents.

2. Develop the ability to locate information.

3. Develop the ability to use the information available to you.

You will not be able to memorize all of the information you are directed to.

Refer to Introduction DOS/VS (GC33-5370).

Read: Part 6 DOS/VS Documentation. Note specifically the figure that describes the available publications.

One manual of special importance to you, as well as to the system operator and system programmer is the DOS/VS Serviceability Aids and Debugging Procedures (GC33-5380). Because of the depth and scope of this manual it is impossible for this course to cover it thoroughly. However, you will be referenced to this manual in several sections of the course, and you should keep the goals previously discussed in mind. This manual will be referred to as DOS/VS SADP throughout the course.

Take a few minutes to familiarize yourself as to how this manual is organized and the general areas with which it deals. Refer to the flowchart on "How to Use This Manual" and briefly scan the Contents page to see what areas of DOS maintenance are covered. Pay particular attention to the subjects covered under the title Types of Malfunctions.

While you are using this manual during the course, this page will usually provide the key to where you want to be in the manual. However, should this not provide a direct reference, refer to the Index in the back of the manual for the subject you wish to reference.

The other manuals you will be referenced to along with the DOS/VS SADP are:

- DOS/VS Operating Procedures (GC33-5378)
- Introduction to DOS/VS (GC33-5370)

SECTION 2-2 - SYMBOLIC REFERENCES

As a CE, you are in close contact with the hardware of a computing system. It is natural for you to associate I/O devices with their address, such as 180, 00E, etc. If, for some reason, you didn't know the address of a device, the system operator could tell you what it was. The CE's only reference to hardware is through its address.

Programmers don't use the physical device address when referring to I/O devices. Instead, they use a SYMBOLIC reference, such as SYSLST when referring to a printer or SYS003, for instance, when referring to a tape drive.

The reason for this can best be illustrated by using an example. Consider the situation shown in Figure 2.1.

1. A problem program is processing and requests input from tape drive 280. As long as tape drive 280 is operable, the program can continue to process.

2. What happens to the problem program if tape drive 280 goes down? It must stop processing until the tape drive is repaired. What about all the other programs the customer has that uses tape drive 280? All programs that require tape drive 280 are delayed until the tape drive is repaired.

Actually, a problem program, as shown in Figure 2.2, requests input from a symbolic unit (SYS006)1, not from tape drive 2802. You can begin to see advantages to this already. Suppose tape drive 2801 became inoperable as in Figure 2.3. Using symbolic references, the program can continue and not have to be delayed until tape drive 280 was repaired. This is accomplished by reassigning the symbolic unit SYS006 to a different tape drive2. Initially, when the program began processing, it requested input from SYS0061 and tape drive 280 actually read the tape. When tape drive 280 went down, the system operator reassigned SYS006 to tape drive 2812. Now, when the problem program requests input from SYS006, tape drive 281 will read tape.

It is not uncommon for the programmers of a large company to never come in contact with the hardware system. In many cases, the programming staff is located in one city, while the programs are used throughout the country at all of the companies offices. The hardware configurations of systems are usually different. One location might be using a 1442 to read and punch cards, and a 1443 to print, while another location may be using a 2540 and 1403. Programmers must be able to write programs that will execute on each system, regardless of the I/O configuration. What is important to the programmer is that each system have the same capabilities, such as: They must all be capable of punching cards or they must all be capable of performing tape operations. The physical characteristics of the devices, such as their addresses, are not important to the programmer. The flexibility of system configuration and the compatibility of programs is provided in DOS through the use of SYMBOLIC UNITS. Let's take a closer look at how input and output operations are performed in DOS.

In DOS, the Supervisor is responsible for performing all input/output operations. When a problem program requests tape input from SYS006, the Supervisor must be able to associate SYS006 with a physical tape drive. The Supervisor makes this association with the use of 2 tables. The tables are Logical Unit Block (LUB) and Physical Unit Block (PUB). Figure 2.4 shows the relationship of the LUB 2 and PUB 4 for an I/O request. The problem program requests input from SYS006 1. Since the Supervisor is responsible for performing all I/O operations, it must associate SYS006 with some I/O device. The Supervisor searches the LUB table for SYS006 2. The LUB table entry for SYS006 points 3 to the PUB Table entry for device 280 4. The PUB Table contains the physical address of the I/O devices and a read operation is initiated on tape drive 280 5.

We know that if tape drive 280 becomes inoperable that the problem program can still continue if SYS006 is assigned to another tape drive. Let's take a closer look at how this is accomplished. We will start out with SYMBOLIC unit SYS006, assigned to tape drive 280 (solid arrows on Figure 2.5). Tape drive 280 becomes inoperable and SYS006 must be reassigned for the problem program to continue. This reassignment is accomplished by Job Control. The system operator submits a job control statement called an ASSGN statement prior to executing the program 1. The ASSGN statement tells job control to change the tables in the Supervisor so that the symbolic reference SYS006 points to the physical address 281 instead of 280 2. When the problem program requests input from SYS006, the Supervisor now reads from tape drive 281 3. Now, let's take a closer look at the LUB and PUB table.

LUB-PUB Table Expansion

The LUB table contains an entry for each of the symbolic units that will be used on a DOS system. The PUB table contains an entry for each physical I/O device attached to the system. The LUB and PUB tables are created at system generation (SYSGEN) time (SYSGEN is when the operating system is created for a specific installation. SYSGEN will be covered later in this session).

The programmer, when he writes a program, chooses the symbolic name of a device from the fixed set of symbolic names in the LUB table. When the program is executed, the operator determines the actual physical device to be assigned to each symbolic name.

An example of a LUB table is shown in Figure 2.6. The entries in the LUB table are grouped into two classes; the SYSTEMS or 00 class and the PROGRAMMERS or 01 class.

1 The symbolic units in the SYSTEMS or 00 class are those used by the various control programs of DOS, such as IPL, Job Control, Linkage-Editor, etc. to run the system.

2 The symbolic units in the PROGRAMMERS or 01 class are those used by the programmers. However, the programmers can also use the symbolic units in the SYSTEMS class.

3 The symbolic units within each class are numbered starting with 0, so that SYSRDR is unit 00 in the Systems class and SYS000 is unit 00 in the Programmers class. SYSIPT and SYS001 would then be unit 01 in their respective class.

4 The problem program refers to symbolic units using a 2 byte constant. The first byte is the class and the second byte is the unit in the class. SYSLST, for example, is referred to by the constant 0003.

What symbolic unit would the constant 0102 refer to? _____

Answer
The constant 0102 identifies the programmers class (01) and unit 02 within the class. This is referring to symbolic unit SYS002.

An example of the PUB table is shown in Figure 2.7. There is one entry in the PUB table for each physical I/O device on the system.

1 The entries in the PUB table (8 bytes) identify the physical characteristics of a device, including the device address. We will be concerned at this time with only the device address (bytes 0 and 1).

2 When the problem program requests an I/O operation from SYSLST, the Supervisor must make the association between the 2 byte constant (0003) used in the problem program and the physical device 00E. This association is made via a PUB pointer in the LUB table.

We already saw how the constant 0003 points to the SYSLST entry in the LUB table.

3 The LUB table entry for SYSLST contains a PUB pointer (02) which points to an entry in the PUB table. In our example, the LUB table is pointing to the third entry in the PUB table. The third PUB entry is for device 000E.

This relationship between the LUB and PUB is how the Supervisor associates a symbolic unit with a physical device.

4 An FF pointer in a LUB entry means that symbolic unit is not assigned to any device. How many symbolic units are not assigned in Figure 2.7?

Answer

3 (SYSREC, SYS000, SYS004)

5 An FF in byte 0 of a PUB entry identifies the end of the PUB table.

Next, we will examine how symbolic units are reassigned to different devices by Job Control.

Reassignment of Symbolic Units

Job Control, through the use of ASSGN statements submitted by the system operator, reassigns a symbolic unit to a different I/O device by changing the pointer in the LUB table. This is illustrated in Figure 2.8.

1. The ASSGN statement that is submitted by the operator contains the symbolic unit to be reassigned and the new physical device address. In this example, we are reassigning SYS002 to device 281.

2. The PUB pointer in the LUB table before reassignment is 06 which assigned SYS002 to device 280.

3. The ASSGN statement, when processed by Job Control, changes the PUB pointer in the LUB to 07. This now assigns SYS002 to device 281.

Through this simple operator function, symbolic units can be reassigned to any device on the system.

Symbolic Unit Requirements

There are specific symbolic units that are required by the various IBM programs supplied with DOS.

These units are defined with the program or function you will be performing and a reference to the explanation and symbolic address assignments will be given in the description.

For example, the Assembler Program requires:

SYS001
SYS002
SYS003
SYSLNK

Which symbolic unit(s) required can be found by looking in the manual for that program. Also, in most cases, a console message will be issued, indicating that 'SYSXXX' must be assigned, if it were not previously done.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 2-2)

1. The table in the Supervisor which contains the device address is the:

   a. LUB table
   b. PUB table

2. Symbolic units are reassigned to other physical devices by:

   a. the Supervisor
   b. the problem program
   c. the IPL program
   d. the job control ASSGN statement

3. Reassignment of a symbolic unit is accomplished by changing a pointer:

   a. in the problem program
   b. in Job Control
   c. in the LUB table
   d. in the PUB table

Use Figure 2.7 to answer the next question.

4. What physical device would the 2 byte constant of 0101 used in the problem program point to?

   a. 0131
   b. not assigned
   c. 000C
   d. 0130

SECTION 2-3 - MULTIPROGRAMMING CONCEPTS

In the first session, it was identified that an operating system makes effective use of all system resources. We just saw how DOS makes effective use of all I/O devices by being able to reassign them when a device goes down. DOS also makes effective use of storage through multiprogramming. System/370s are available in various memory sizes (storage). Even the most complex program doesn't require all of the storage available. It would not be practical to waste the storage left over as in Figure 2.9.

Multiprogramming provides a means of making effective use of all the storage in a CPU. Storage is divided into partitions as shown in Figure 2.10. Each partition contains a problem program. This technique of being able to execute more than one program simultaneously is called MULTIPROGRAMMING.

Most programs are depending on information coming in from an I/O device before they can proceed. Figure 2.11 represents the relationship between the time a problem program is actually executing instructions 1 and the time it is waiting for data from an I/O device 2 . While the problem program is waiting for data from an I/O device, the CPU is standing idle (in the WAIT state). This is the time when another program can be using the CPU resources. We can put this philosophy to use in DOS, using partitions as identified before (see Figure 2.12). As one problem program is waiting for I/O data, another program is using the CPU.

Figure 2.12 shows an ideal situation of 3 partitions interleaving processing. In reality, when a lower priority partition is running and a higher priority partition was ready to run (I/O complete) it would gain control of the CPU resources.

Refer to Introduction to DOS/VS (GC33-5370).

Read: The section titled Resource Utilization through but not including Virtual Storage Support.

Save Areas

The ability to process more then one program at a time brings up the following questions.

(1) With only one set of general purpose registers and floating point registers per CPU, how is the data in them associated with the correct program?
(2) How is the PSW for the specific partition preserved?

These questions are taken care of through the use of partition save areas. Each partition has a save area where the contents of the general purpose registers, floating point registers and the partition current PSW is stored when the partition is not processing. The location of the partition save areas is shown symbolically in Figure 2.13.

The use of the save areas can best be illustrated using Figure 2.14.

Note: Assume a three partition setup.

1. Let's start off with the F1 partition running a payroll program. The payroll program requests data from SYS006. The payroll program cannot continue until the data is in core.

2. While the payroll program is waiting for the I/O operation to complete, it signals the supervisor, via a Supervisor Call Instruction (SVC), that it is waiting for I/O to complete and doesn't need the CPU for a while.

3. The Supervisor takes the contents of the general purpose and floating point registers and the partition current PSW, and stores them in the F1 save area. (For this example, the partition current PSW will actually be the SVC Old PSW.)

4. The Supervisor, through its Task Selection routine, determines which program will run while the payroll program is waiting for its I/O operation to complete. Because of the priority set up in DOS, the program in the F2 partition will run next.

5. Before the inventory program can start executing, the general purpose register, floating point registers, and the current PSW for the F2 partition must be restored. The information is taken from the F2 save area.

6. The inventory program begins processing and eventually requests input from SYS005. The inventory program cannot continue until the data from SYS005 is in core.

7. While the F2 inventory program is waiting for its I/O operation to complete, it signals the Supervisor via an SVC that it is waiting for I/O to complete and does not need the CPU for awhile. The sequence of events repeats itself as before.

The next partition to gain control could be either F1 or BG. If the I/O operation for the payroll program had completed while the inventory program was processing, then F1 would gain control next. If the payroll program was still waiting for the I/O operation to complete, then the BG partition would gain control next and the stock update program would begin to execute.

Customers determine at system generation time if multiprogramming is to be used.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 2-3)

1. (True/False) Multiprogramming is the ability of a CPU to share it's resources between two or more programs.

2. The F2 save area is located:

   a. in the Supervisor
   b. in the F2 partition
   c. in the problem program

3. (True/False) The partition priority can be changed during system operation as well as during system generation (SYSGEN).

4. (True/False) The size of the BG partition can be less than 10K in a multiprogramming system.

SECTION 2-4 - VIRTUAL STORAGE

Before you begin to study virtual storage concepts and their implementation in the IBM System/370, let's take a brief look at the history of computers and their operating systems. The past fifteen years have seen spectacular changes in computing equipment and programs.

Fifteen years ago, central processing units (CPUs) performed instructions in thousandths of a second. Today, System/370 executes instructions in billionths of a second. In a system like the IBM 1401, main storage sizes ranged from 1400 to 16,000 locations. In System/370, main storage sizes range from 96K to over four million bytes in the range of its models. Data transfer rates for some new tape drive models are up to twenty times faster than early tape drives. Storage capacity on the new 3330 disk drive is twenty five times larger than the early models of the 1311 disk drive. Its transfer rate is eight times faster. Improvements just as dramatic have been made for printers, card readers and card punches. This overview does not even consider new devices for applications like teleprocessing that did not exist fifteen years ago.

A large amount of software development has accompanied this hardware evolution. With early computers, programmers used machine language. They had to know machine code to use these systems. Then came assemblers, compilers and input/output control systems. These had the effect of moving the programmer farther from machine code, letting him spend more time solving application problems. Computers still executed programs one at a time. Each program was set up and run independently by the system's operator.

The first operating systems enabled users to batch jobs and run them in a single stream. Jobs executed one at a time and there was a smoother transition from job to job with some assistance from the operating system. This made the operator's job and system operation, easier. As CPU speeds became faster and main storage more abundant - for example, with the IBM System/360 - operating systems expanded in scope. These new operating systems let users run multiple jobs concurrently - multiprogramming - by sharing the CPU, real storage and other system resources among active jobs. They also made possible teleprocessing applications that control remote computing and data entry operations from one central location. DOS on System/360 is an example of such a system.

These technical developments have meant significant benefits for computer users and their people - system analysts, programmers, computer operators, and so forth. System/370 users can now develop complete systems for operational control and management information in contrast to the typical payroll and billing applications of ten to fifteen years ago. System resources are shared among several jobs in multiprogramming systems, and among multiple users in time sharing systems. Programmers use application-oriented languages like PL/I, COBOL, and FORTRAN. As a result, programmers can devote more time to problem solutions. Operating systems like DOS handle much of the job preparation and job-to-job transition that formerly occupied so much of a computer operator's time. Operators now spend less time handling punched cards and tapes and more time directing system activity, keeping the system productive. Thus, the programmers and operators who work with computers, the scope of applications developed and, in general, computer users have benefited a great amount from the past developments in hardware and software.

With System/360, the primary operating systems were OS and DOS. With System/370, you may still use the OS and DOS systems. We have examined how we arrived at the current mode of computer operation. Where can we go from here?

What would your objectives be if you were designing new features and functions for System/370. Take a moment and think about it. We will list a set of objectives; they might be somewhat different than yours, but we will have a common reference point. Our list of objectives follows:

1. Programmers should have the amount of main storage that they need for designing programs without having to use planned overlay or dynamic management techniques. Even though the size of available computer main storage has increased tremendously, users cut their storage into partitions or regions for multiprogramming efficiency. Programmers, then, are restricted to the size of the largest partition or region used in their installation. This often requires breaking a program up into separate steps or using special overlay techniques to make programs "fit" into a region or partition. All of these design requirements add overhead to solving a problem.

2. If a program is too large for main storage size, the operating system, not the programmer, should make the program "fit" into main storage.

3. Programs should use system resources - especially main storage - only as required during execution. For example, a program that needs 82,000 bytes of main storage when fully loaded may reference only 22,000 bytes during one part of processing. During this time, there is no need for the operating system to commit main storage to 60,000 unreferenced bytes. An example of such a situation is a teleprocessing application running at less than its maximum load.

4. The operating system should not allow main storage to become fragmented. Assume that several programs are executing, each in its own contiguous area of main storage. Three 15K byte areas in main storage are idle (none of the executing programs occupy these areas). If the smallest program waiting to begin execution needs 30K bytes of main storage, it must wait until 30K bytes of continguous main storage becomes available. Until then, a total of 45K bytes of main storage are idle because of storage fragmentation. They are wasted.

5. An operating system should control system resources like main storage in such a way that you automatically get a performance improvement by adding more main storage. For example, if you have a program that must use overlays because it won't fit into main storage, adding more main storage won't help at all unless you redesign and recode the program. It would be nice if the operating system could somehow "automatically" overlay programs and "automatically" use added main storage.

6. A computer user should be independent of the size of the main storage in which programs execute and in which the operating system is structured. He should be able to structure a system more according to his needs than to the size of main storage.

7. Main storage should be shared dynamically among the active jobs in the system. Programs should get the main storage that they need when they need it. In other words, the system should be adaptive to the demands of the system's activities.

8. Scheduling and operating a system should be easier. Systems like OS require a large amount of user participation to schedule jobs and control system resources. A new system should require less user participation to achieve good scheduling and operation results.

Most items in this list of objectives relate in some way to how an operating system manages a computer's main storage. These objectives can be fulfilled by an operating system that supports a virtual storage.

The following reading assignment will provide a big picture description of virtual storage. If you would later desire to read/study more about virtual storage, you can obtain the following publication: "Introduction to Virtual Storage in System/370" (GR20-4260). This publication is not included in this course.

Refer to Introduction to DOS/VS (GC33-5370).

Read: Section titled Virtual Storage Support through but not including Power.

Refer to DOS/VS Operating Procedures.

Read: Section titled Virtual Storage.

In Section 2-3 save areas were discussed and the example shown in Figure 2.14 did not show virtual storage. Figure 2.15 depicts the save areas when using virtual storage. Note that the save areas are in real storage.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 2-4)

1. (True/False) A program that is to be executed must have either all its addresses in the real address range or all its addresses in the virtual address range.

2. A page is a block of code of:

   a. 1K.
   b. 2K.
   c. 4K.

3. The Supervisor always works in:

   a. Real mode.
   b. Virtual mode.

4. Job Control is always executed in:

   a. Real mode.
   b. Virtual mode.

5. A page frame is 2K of:

   a. Real storage.
   b. Virtual storage.

6. The page data set, located on a disk storage medium, is equivalent to the size of the designated:

   a. Real storage.
   b. Virtual storage.

7. (True/False) The main page pool is located in real storage and this storage can be used to handle pages from any partition.

SECTION 2-5 - SYSTEM COMMUNICATION

DOS operation centers around a two way communications between the operating system and the operators. System to operator communications is via messages. Operator to system communications is via operator commands and Job Control statements. This section will introduce the basic communications in DOS. The specific messages and commands will be covered, as they are used, throughout the course.

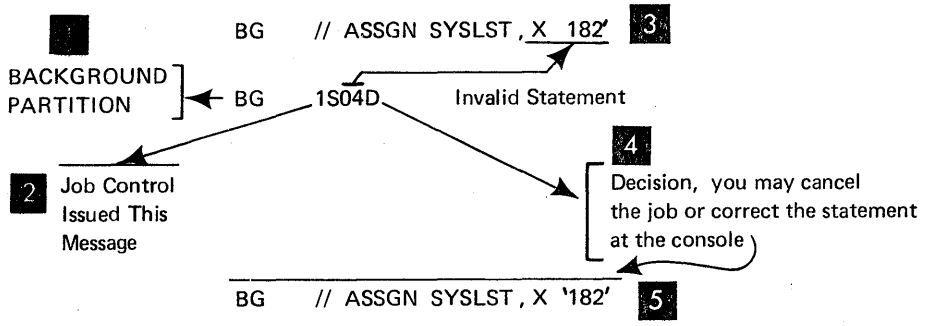
Messages provide the system operator with the following:

(1) The partition issuing the message.
(2) The IBM Program of DOS issuing the message.
(3) The parameter in the command or statement that is in error. (error message only).
(4) The action the operator must take in order for the system to continue.

System communications can best be illustrated using the following example:

(1) The operator attempted to reassign SYSLST to another physical device and made an error while typing in the command.
(2) The system detected the error and responded with an error message.

Let's analyze the error message for the four previously listed items.



1 The partition issuing the message is Background.

2 The control program responsible for the message is Job Control. This is because Job Control is the program which processes job control statements.

3 The fourth parameter in the assign statement is incorrect (the first 'quote' is missing).

4 The action required by the operator is either cancel the job or resubmit the correct assign statement.

5 The correct ASSGN statement can be typed in on the console typewriter.

The DOS/VS messages manual contains all the error messages categorized by the message identifier (the control program issuing the message). Refer to Appendix C in the SCM and glance through the various messages. Appendix C contains selected messages extracted from the DOS/VS messages manual. Note the various identifiers.

Locate the following message in Appendix C.

3M44I

Write the comment that prints with this message.

_____

The complete message with comment would print as:

3M44I PRIVATE CORE IMAGE LIBRARY ASSIGNED ELSEWHERE

What DOS program issued the above message? _____

The Librarian programs issue messages beginning with the digit '3'.

Low Core Messages

Another method of communications between the system and the operator is through the use of Low-Core Messages. The use of low-core messages will be covered next during the IPL procedure.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 2-5)

Use the following message to answer the study questions:

BG3E39I

1. The program that issued the message is:

   a. IPL
   b. Job Control
   c. Supervisor
   d. EREP (Librarian)
   e. Sort

2. The partition that issued the message is:

   a. Background
   b. IPL
   c. Job Control
   d. Supervisor
   e. None of the above

3. The action indicator in the above message indicates:

   a. Action
   b. Information
   c. Decision

SECTION 2-6 - IPL PROCEDURE

IPL is the process of bringing DOS up, and getting the complete system ready to execute programs. It was discussed earlier that the IPL program will clear main storage and load the Supervisor Now we will look at the communications required when a DOS system is IPL'ed.

We will begin the IPL procedure by building a hardware system. A typical system you might see in any account is shown in Figure 2.16. The devices are labeled with their physical device address. It was established in the section on Symbolic References that programs do not use the physical device addresses when performing I/O operations. The IPL program is no exception. There are 3 symbolic units required by the IPL program. These symbolic units are shown in Figure 2.17 as SYSLOG 1 SYSRDR 2 and SYSRES 3 .

The IPL procedure can best be illustrated using a series of figures.

1 Refer to Figure 2.18. IPL begins by mounting the SYSRES disk pack on a drive, setting the disk drive address in the load switches and pressing the LOAD key on the CPU. This brings the IPL program into core.

The IPL program loads the supervisor into core and is ready to establish communications for the first time with the system operator. Only 2 devices can be used for the two way communications between the system and the operator. The 2 devices are SYSRDR or SYSLOG. For this discussion of IPL, the console typewriter will be used for communications.

2 When the system operator depresses the REQUEST Key on the console typewriter, the following system to operator message is printed on SYSLOG:

0I10A GIVE IPL COMMANDS

3 The operator must now provide the operating system with information for it to continue. A SET command must be entered now which provides the operating system with the date, and if the interval timer is supported, the time of day. The SET command must be entered as follows:

SET DATE=11/17/71, CLOCK=10/45/15

3A You must follow the SET command by the DPD command to indicate that IPL is to handle the page data set, which is necessary for the virtual address area. The DPD command is required, with or without operands. If submitted without operands, IPL will use the information specified in the DPD macro during supervisor generation to perform page data set handling. IPL assigns the symbolic name SYSVIS to the page data set.

4 The IPL process continues and if TOD (Time of Day) clock has been supported at system generation time, the system light stays on indicating the need for operating the TOD clock switch on the panel to satisfy this function. This switch is located directly above the LOAD switch operated at IPL time. Following this, Job Control will be loaded into core overlaying IPL. The IPL and Job Control programs will print the following system to operator messages on SYSLOG.

```
BG 0I20I DOX/VS IPL COMPLETE                    (IPL)
BG 1I00A READY FOR COMMUNICATIONS               (Job Control)
BG                                              (Job Control)
```

5 At this time IPL is complete and Job Control has been loaded into core to begin reading in jobs for processing.

The function of Job Control and how jobs are executed is covered in SESSION 5 of this course.

ADD and DEL Commands

You just went through an IPL of DOS. There are some variations to the IPL procedure that we will see now. In the section on symbolic references, we discussed the PUB table and how it contained the addresses of the physical devices. It also contains some of the characteristics of the device, such as whether or not a 1403 printer has the UCS feature. Consider the following situation:

Your operating system is generated for a specific system hardware configuration which has a 1403 printer with the UCS feature. Let's call this system A. System A is not available because of PM. Another hardware system (system B) is available to you but it doesn't have a UCS printer. The 1403 on system B is a standard printer. Since each DOS system is generated for a specific system hardware configuration, your DOS system must be modified in order to run on system B. This type of modification is made during the IPL procedure through the use of ADD and DEL (delete) commands. Refer to Figure 2.19.

Steps 1 and 2 of this IPL procedure are the same as in Figure 2.18.

3 The DEL and ADD commands must be used before the SET command. For our example, we will modify the PUB table in the Supervisor so it will be compatible with the system B configuration. The DEL and ADD commands must be entered as follows:

```
DEL     X'00E'
ADD     X'00E',1403
```

The DEL command removes the device, whose address is 00E, from the PUB table. Since the PUB table contains the physical characteristics of each device, the entry for the UCS printer has been removed.

The ADD command adds an entry to the PUB table for a 1403 with the address of 00E. The difference in the ADD command for adding a UCS printer would be:

```
ADD X'00E', 1403U
```

The second parameter in the command determines the device type (1403U).

4 The operator now provides the operating system with the SET command and the DPD command as before.

5 The IPL process continues as before and when completed, the IPL and Job Control programs will print the following system to operator messages:

```
BG 0I20I DOS/VS IPL COMPLETE                (IPL)
BG 1I00A READY FOR COMMUNICATIONS           (Job Control)
BG                                          (Job Control)
```

At this time, the DOS system which was generated for the hardware configuration of system A can run properly on the hardware configuration of system B.

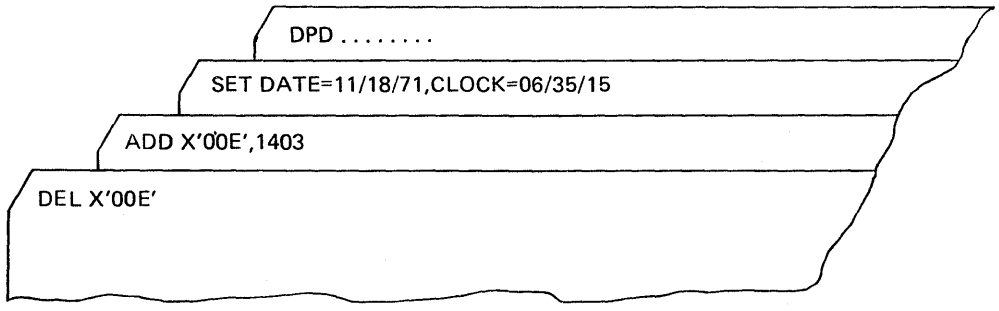
NOTE: When a device is removed from the PUB table with a DEL all symbolic references from the LUB are removed. Therefore, even though an ADD statement with the same device address is added, the operator must submit ASSIGN statements for all symbolic units that reference this device.

IPL Using Card READER

The communications required to IPL DOS, using the card reader, are the same as for using the console typewriter, except that the operator to system commands are punched into cards. Figure 2.20 shows the IPL procedure using the card reader as the communications device.

1 This step remains the same as before. That is, you mount the SYSRES pack on a disk drive, set the disk address in the address switches and depress the LOAD Key. The system will go into the WAIT state.

2 The IPL commands are punched into cards as shown:



These cards are placed in the card reader and the reader made ready.
NOTE: The cards must be punched starting in column 1.

3 The cards will be read from SYSRDR and the IPL process will continue. When it is completed, the following system to operator messages will be printed on SYSLOG by IPL and Job Control.

```
BG 0I20I DOS/VS IPL COMPLETE
BG 1I00A READY FOR COMMUNICATIONS
BG
```

At this time, DOS is ready to execute programs.

If ADD and DEL commands are not required, only the SET and DPD commands are required in the card reader.

Read: In DOS/VS Operating Procedures (GC33-5378), Operator Command (ADD) Section.

Code the ADD statement to add a 2495 Tape Cartridge Reader with a device address of 01A to the PUB table.

The device type is 2495TC. The ADD Statement would look like:

```
ADD X'01A',2495TC
```

IPL Error Messages

Error messages during IPL can appear in one of two formats.

(1) They can print on the console, or
(2) The system will enter the WAIT state with a low-core message. (A message code stored in bytes 0 through 4).

In either case, the action on the part of the operator is the same.

Refer to the DOS/VS SADP Manual (GC33-5380).

Read: Section 2 - Serviceability Aids, Wait State Messages.

Refer to Appendix C - Supplementary Course Material.

Read: Messages
0I10A
0I16A
0I18A
0I20I

What action would you perform as a result of the following low core message?

F0C9F1F5C9

_____

_____

This error message (0I15I) is the result of submitting an ADD statement for a device that is already in the PUB table. You can either correct the command if there is an error in it, or don't use the command because the device already exists.

Summary

We have just completed the operating procedure required for IPLing a system using DOS. The next three sessions will deal with the layout of SYSRES, operating procedures of the System/370, and details of executing jobs using DOS. At the conclusion of the next three topics, you will be able to perform most of the system operator functions.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 2-6)

1. Write the SET command, using January 12, 1972 as the date, and 11:35:07 as the time, that would be submitted at IPL time.

   _____

2. (True/False) The DEL and ADD commands, if used, must precede the SET command.

3. Write the DEL command used to delete device 190 from the PUB table.

   _____

4. Write the ADD command used to add a 1017 model 2 with an address of 01D to the PUB table.

   _____

SECTION 2-7 - SYSTEM GENERATION (SYSGEN)

The term system generation was used in the previous section when we talked about the LUB and PUB tables and about multiprogramming. System generation time (SYSGEN) is that time when the customer builds his operating system tailored to his specific needs. The customer decides such items as:

(1) Will multiprogramming be used?
(2) How many symbolic units will there be in the LUB?
(3) How many entries in the PUB will there be?
(4) Which language compilers will be used?

IBM provides the customer with all the necessary components to do a SYSGEN. These components are provided, on tape or disk, from the Program Information Department (PID). An example of the basic components supplied in a PID system are:

(1) A Basic Supervisor - This is the Supervisor that is initially used to begin the SYSGEN operation. The customer will, as one of the steps in SYSGEN process, assemble a supervisor to fit his needs. The SYSGEN will then be completed using this new Supervisor
(2) Language translators - all the language translators such as COBOL, PL/I, FORTRAN, etc. are provided. The customer will include the translators used by his programmers in his operating system.
(3) IBM utility programs - Utility programs are those programs that a customer uses to maintain his operating system. Some of these utilities will be introduced later in this course.

Summary

An operating system is built, at SYSGEN time, from IBM supplied components distributed from PID. No two operating systems are alike. It is even very likely that two operating systems in the same customer account will be different.

SESSION 3 - SYSTEM RESIDENCE ORGANIZATION

SECTION 3-1 - DOS SYSRES

LIBRARIES

Each DOS SYSRES pack usually contains four libraries: core image, relocatable, source, and procedure library. The size of each library depends on the individual customer's needs. Figure 3.1 presents a pictorial view of a typical DOS SYSRES pack.

1 The IPL bootstrap routine resides on records 1 and 2 of cylinder 0.

2 The core image library residing in cylinders 1 through 50 can contain any number of executable programs. A typical example of programs that can be found in the core image library are:

- The Supervisor
- Job Control
- Language Translators
  - COBOL
  - ASSEMBLER
  - FORTRAN
  - PL/I
- Utility programs
- Customer's problem programs
  - payroll
  - inventory control
  - accounts payable
  - accounts receivable

As can be seen, the size of the core image library depends on the customer's applications and is tailored strictly to his needs.

3 The relocatable library residing in cylinders 51 through 60 can contain any number of object modules. A module is the output from a language translator. For example, a customer writes a program using the COBOL language and compiles it with the COBOL compiler. (The term compiler and translator mean the same thing when referring to languages such as COBOL, PL/I, etc.) The output from this compile is an object module in relocatable format.

The purpose of the relocatable library is to allow the user to maintain frequently used routines in residence and combine them with other modules without requiring recompilation. Routines from the relocatable library can be 'linked' together by the linkage editor and then placed into the core-image library. The size of the relocatable library depends on how many modules the customer wants to store. The relocatable library closely resembles a card file where a customer can store programs. The advantage of the relocatable library is that it is usually on-line with the operating system versus a card file which is a manual operation.

4 The source statement library residing in cylinders 61 through 70 can contain any number of books. A book is a sequence of source language statements used by the translators. There are books (macros) of source statements for the ASSEMBLER and other language translators. These source statements are used while compiling a customer program. The number of different language translators will determine the size of the source statement library.

5 The procedure library is used to catalog frequently used sets of job control and linkage editor statements.

6 SYSRES ends with cylinder 70. The area from cylinder 71 through the end of the pack can be used by the customer as work space. Think of this work space as a scratch pad used by a problem program. When the program is finished, the data in the work space is no longer needed and the space is used by another program.

Refer to Introduction to DOS/VS (GC33-5370).

Read: Libraries through but not including Linkage Editor and Relocating Loader.

DIRECTORIES

(See Figure 3.2.) A directory is a series of records, each record containing information about a specific program, module, or book within the associated library. Each record is considered a directory entry and contains the name of the program, the location of the program in the library, the length of the program, etc. The system loader searches the directory, starting at the beginning, looking for the entry of the program that has been requested. From the directory entry, the system loader can locate the program in the library.

1 The Sub-Directory, located on cylinder 0 of the SYSRES volume, contains entries for the most commonly used system programs in the core-image library. These entries are categorized into groups. The system loader, when searching for a directory entry, will search only the specific group of records instead of the entire directory. This grouping of the records in the Sub-Directory provides faster system operation.

2 3 4 Each of the other directories are located within their respective library. The directory starts with the first record on the first cylinder of the library. The size of the directory is determined by the number of programs or modules in the library. The larger the library, the larger the directory will be.

DIRECTORY CONTENT

The information contained in the directory entry and its use is discussed, using Figure 3.3.

1 A request to execute a program (PAYROLL) is submitted by the system operator.

2 The system loader, a portion of the supervisor, is responsible for locating the program in the SYSRES pack and loading it into the correct location in main storage.

The system loader will first search the Sub-Directory for the PAYROLL program entry. The Sub-Directory, remember, contains the entries for the most commonly used programs. If the entry is not found in the Sub-Directory, the system loader will search the Core Image Directory for the PAYROLL program entry **3**. If no entry is found in the Core Image Directory, a message is issued to the system operator stating PROGRAM NOT FOUND.

4 The directory entry contains the following:

1. The program name
2. The location of the program in the library (library address)
3. The main storage address where the program is to be loaded (load address).
4. The address of the first instruction in the program to be executed (entry address).

5 The system loader uses the data in the directory entry to locate the program in the library, load it into main storage, starting at location HEX '3200', and transfer control to the instruction located at HEX '3400'.

PRIVATE LIBRARIES

So far, the discussion of the SYSRES libraries (called system libraries) was confined to one disk pack. The size of the 4 libraries is restricted to how much data can be stored on one disk volume.

Many customers desire a large core image library (see Figure 3.4), which, in turn, would restrict the size of the relocatable, source statement, and procedure libraries. The use of Private Libraries allows for large libraries without restricting the size of the remaining library.

A private library (see Figure 3.5) resides on a disk pack other than SYSRES. The use of private libraries, as opposed to system libraries, provides the following flexibility to the customer's system configuration.

1. The customer need not be concerned with the size of the system libraries on SYSRES.

2. The customer may have as many private libraries as he desires, each serving a specific function.

3. System libraries (located on SYSRES) may be used in combination with private core-image, private relocatable, and private source statement libraries.

4. A restriction of private libraries is that only one private libary of a type may be online at any one time. (eg, only one private relocatable library can be online with the system libraries at any one time.)

5. Private libraries are useful in a testing environment because the user can keep his working programs on a system library and his modified versions of these programs on private libraries. The modified versions can thereby be tested without destroying the current working programs that are on the systems residence file or another private library.

6. When private libraries are used, the system loader will search the directory of the private library first before searching the directory in the system library.

7. The minimum SYSRES configuration must contain a core image library. The core image library need not take up all of the space on the disk pack. SYSRES does not have to contain a system relocatable library, a system source statement library or a procedure library.

8. The procedure library is available as a system library only.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 3-1)

1. (True/False) The SYSRES pack can contain 4 libraries.

2. The _____ is the only library required on SYSRES.

   a. relocatable library
   b. source statement library
   c. core-image library

3. The library which contains programs in their executable form is the:

   a. relocatable library
   b. source statement library
   c. core-image library

4. (True/False) The relocatable directory is located on the first cylinder of the relocatable library.

5. (True/False) When private libraries are used in combination with system libraries, the system library directory is searched first for an entry.

SECTION 3-2 - DOS VOLUME IDENTIFICATION

VOLUME CONFIGURATION

The organization of SYSRES can become complex with the use of private libraries and system libraries. The operating system must continually be aware of the SYSRES configuration for correct operation. The organization of the disk volume is identified by labels. A label is a record which contains identifying information about a specific portion of the disk volume called an EXTENT. An extent is referred to as an area of storage space on a volume whereas a data file refers to the actual data contained within the extent limits.

Figure 3.6 identifies three extents. 1 The SYSRES extent from cylinder 0 through cylinder 70. 2 A work file 1 extent from cylinder 71 through cylinder 100 and 3 a work file 2 extent from cylinder 101 through cylinder 198. The labels required for identifying these data files are called FORMAT 1 labels and are grouped together and stored in a specific area on the disk volume. This group of FORMAT 1 labels is essentially a directory of the data files on the disk volume and is appropriately called a Volume Table of Contents or VTOC pronounced 'V-Tock', as in clock.

VTOC

Preformatting the VTOC: The VTOC is preformatted by the initialize disk or initialize data cell program. The customer specifies its location and length when initially preparing the volume for use. It can be placed anywhere within the volume with the following restrictions:

1. For the 2311, 2314, or 2319 it must be within cylinders 0-199 (cylinders 200-202 are used as the alternate track area). For the 2321, it must be within subcell 0, strip 0, cylinder 0 and subcell 19, strip 5, cylinder 4 (strips 6-9 are used as the alternate track area). For a 3330, it must be within cylinders 0-403 (cylinders 404-409 are used for alternate track area).

2. If it is on a system residence disk pack (SYSRES), it must be outside of the residence area.

3. It must be one or more full tracks, with the single exception noted in item 5.

4. It must be contained within one cylinder. It cannot overflow onto another cylinder.

5. If it is on a nonresident disk pack, the standard location for the VTOC is cylinder 0, track 0, immediately following the volume label.

Refer to Figure 3.7 while reading the following:

1 The Initialize Disk Utility program creates a volume label. This label is always the third record on cylinder 0. Records 1 and 2 are IPL records. The customer must specify via control cards in the initialize disk program, the volume serial number 2 , the location of the VTOC 3 , and the length of the VTOC. The initialize disk program then builds the volume label and segments the VTOC into fixed records of 140 bytes each. The first record in the VTOC is the label for the VTOC and contains the length of the VTOC 4 . This label is required because the VTOC is considered a data file in itself. The second record in the VTOC is reserved. The third and succeeding records will be used for FORMAT 1 labels.

The volume label is used to maintain volume integrity. Each disk volume in a customers account will have a unique volume serial number. This allows it to be identified from any other disk volume.

FORMAT 1 LABEL

When a data file is created, such as for a PAYROLL application (see Figure 3.8 1) a set of label cards 2 must be provided which identifies some unique characteristics of the data file. They are:

1. The name of the data file
2. The beginning disk address of the extent
3. The ending disk address of the extent
4. The expiration date of the data file. (When the data in the file can be destroyed).

This information is put into the FORMAT 1 label in the VTOC for that data file 3 .

Each time an extent on a disk volume is to be used, the VTOC is checked to determine if a FORMAT 1 label exists with the name of the file that is to be used. This provides data protection and integrity. A data file cannot be used unless the system operator provides the file name and extent limits exactly as they appear in the FORMAT 1 label in the VTOC.

SUMMARY

Because of the many different disk volumes that are used in a customer's account, a method of providing identification to the operating system is required. The volume label provides that identification via the volume serial number. Each disk volume can contain several data files. Protection must be provided to prevent inadvertently destroying data in a data file. This protection is provided by the FORMAT 1 label in the VTOC.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 3-2)

1. The volume label contains: (more than one answer is required)

   a. the volume serial number.
   b. the location of the SYSRES extent limits.
   c. the location of the VTOC.

2. The FORMAT 1 label contains: (more than one answer is required)

   a. the name of a data file.
   b. the location of the VTOC.
   c. the expiration date of a data file.

3. (True/False) The volume label is always record three in the VTOC.

SESSION 4 - SYSTEM/370 CONSOLE OPERATIONS

This portion of the course will deal with some of the specific operations that can be performed on the System/370 Models 135, 145, and 125 consoles to gather data relating to machine failures. It is a problem determination aid and will introduce you to the section in DOS/VS SADP manual (GC33-5380) that describes these functions. All sections of this topic will use this manual as a reference.

Study questions follow each part of the session and the answers to all the questions will be found at the end of the session.

SECTION 4-1 - ALTER/DISPLAY

Read: DOS/VS SADP manual - Section 2 Aids provided by the Operator's Console Alter/Display - Models 135, 145, and 125.

Answer the following study questions relating to the alter/display operation.

1. (True/False) The alter/display key can be used to terminate alter/display mode on the System/370 Models 135 and 145 console.

2. Which mnemonic should be used to display the current PSW on the System/370 Model 145?

   a. DM
   b. DP
   c. DG
   d. DF

3. (True/False) To alter control storage from the keyboard, you would press alter/display, type in AS and type in the address range desired to alter on the System/370 Models 135 and 145.

4. To display the control registers on the System/370 Model 125, some of the following steps must be performed in sequence. This sequence is:

   1. Type A and depress ENTER
   2. Depress mode select key
   3. Depress cancel
   4. Type G and depress ENTER
   5. Type C and depress ENTER

   a. 2, 1, 5
   b. 2, 1, 4
   c. 2, 4, 5

5. (True/False) On the System/370 Model 125, to exit from an alter/display mode to another mode requires that MODE SELECT be depressed two times.

6. (True/False) On a System/370 Model 125, to alter any information displayed, you must move the cursor to the desired digit to be changed and type in the new data.

7. (True/False) To enable any alter/display operation on a System/370 Models 135 and 145, the MANUAL indicator lamp must be on.

SECTION 4-2 - INSTRUCTION STEPPING

Read: DOS/VS SADP - Aids provided by the Operator's Console Instruction Stepping - Models 135, 145, and 155

Answer the following study questions when you have completed this reading assignment.

Study Questions (Section 4-2)

1. (True/False) Instruction stepping is most helpful when used as a problem determination aid to check portions of a program suspected of causing machine malfunctions.

2. On the System/370 Models 135 and 145, what three conditions must be present to allow instruction stepping?

   a. WAIT lamp on
   b. Manual indicator off
   c. Manual indicator on
   d. RATE switch set to "Instruction Step"
   e. PROCEED lamp on keyboard on

3. (True/False) To exit from instruction stepping on the System/370 Models 135 and 145, reset the RATE switch to PROCESS and press the START key.

4. On a System/370 Model 125, while in instruction stepping mode, the address and content appear on line _____ of the display.

   a. 1
   b. 10
   c. 14
   d. 5

5. (True/False) To end the operation of instruction stepping on the System/370 Model 125, press MODE SELECT.

SECTION 4-3 - CONSOLE DUMP OPERATION (SYSTEM/370 MODEL 125 ONLY)

Read: DOS/VS SADP - Aids provided by the Operator's Console Dump Operation - Model 125 only.

Study Questions (Section 4-3)

1. (True/False) A console dump on the System/370 Model 125 is an aid to display low address storage before executing the stand-alone dump program.

2. The method used to terminate a 64K dump print, if desired, is:

   a. Press CANCEL and START
   b. Press MODE SELECT, CANCEL or START
   c. Press MODE SELECT, CANCEL and START
   d. Press CANCEL

SECTION 4-4 - STORE STATUS - MODELS 135, 145, AND 125 SAVE USAGE COUNTERS - MODEL 125 ONLY

Read: DOS/VS SADP - Aids provided by the Operator's Console Store Status and Save Usage Counters.

Additional Information Regarding Usage Counter Save

The usage counters saved on the System/370 Model 125 by the MODE SELECT command "U" contain the following information:

- Number of seeks.
- Dismount records (similar to EREP-RDE)
- All error recording information that is normally saved by EREP on the System/370 Models 135 and 145.

The method used to save this data from storage is by the command "U" on the MODE SELECT option. Upon pressing the ENTER key, microprogram accesses this "status" information in storage and writes it on the diskette for later retrieval by the CE.

To retrieve this information, there are two methods available. Log analysis display will present this information on the display console or a special deck of the SEREP program developed for the System/370 Model 125 will edit and print this information on the system printer. In the SEREP printout, the format of error information is compatible with that received from EREP on other System/370 models.

Remember this information only pertains to those device errors as stated in the DOS/VS SADP manual. This amplifies that this operation must precede a "power-down" situation to prevent loss of pertinent error data that are contained in storage.

Study Questions (Section 4-4)

1. (True/False) The status stored on an ST operation on the System/370 Models 135 and 145 only, takes data in storage and writes it on the recording file (SYSREC) of the DOS/VS system.

2. The System/370 Model 125 employs an error recording system that functions under control of:

   a. Microprogram.
   b. ERP routines.
   c. Log Analysis routines.

3. The data stored on a STORE STATUS operation (System/370 Models 145 and 135) contains the following information:

   a. Contents of Control Registers
   b. Contents of General Registers
   c. Current PSW
   d. CPU Timer
   e. All of the above

Summary

The most significant aids for the operator and you, the CE, have been presented to assist you in problem determination of a system problem through use of the console. Although there are many other functions a programmer might use from the console, the major aids have been highlighted to assist you in gathering error data, low storage data and tracing short program loops in determining whether an I/O device is the cause of the error.

Answers for Section 4-1 Study Questions

1. False
2. b
3. False (control storage cannot be altered)
4. a
5. True
6. True
7. True

Answers for Section 4-2 Study Questions

1. True
2. c, d, e
3. True
4. c
5. False

Answers for Section 4-3 Study Questions

1. True
2. b

Answers for Section 4-4 Study Questions

1. True (this is not true for the System/370 Model 125)
2. a
3. e

SESSION 5 - DOS CONCEPTS AND OPERATING PROCEDURES

SECTION 5-1 - JOB CONTROL

Some of the manual procedures of System/370 operation have been previously introduced along with how to IPL the DOS system. This section will concern itself with the execution of jobs and the operator communications required for system operation.

Communication Considerations

Refer to DOS/VS Operating Procedures (GC33-5378)

Read: In the Communication with the System Section, Operator-to-System Communication.

JOB CONTROL STATEMENTS - OVERVIEW

The following is a more detailed presentation of the optional reading assignment given in Section 1-10. You may desire to only briefly read the following.

Before we get into the job control statements, there are three definitions that must be understood:

- job stream is a series of jobs.
- job is one or more related programs that are executed usually in a specific sequence.
- job step is the execution of one program.

Job Control statements are the system operator's way of identifying jobs and job steps to the Job Control program. They can best be illustrated by using the following series of figures.

Three types of job control statements are used in Figure 5.1.

1 // JOB statement identifies the beginning of a job and contains the job name. The first job name is SORTPAY.

2 // EXEC CALCPAY statement identifies the program to be executed.

3 /& statement identifies the end of a job. Within this job there is only one job step (// EXEC CALCPAY).

4 The // JOB statement indicates the start of a new job named LIST.

5 This job has three job steps because there are three // EXEC statements. The first program to be executed is CHKREG, the second is YRTDPAY, and the third is PAYCHECK. Each // EXEC statement indicates a job step.

6 This statement (/&) terminates the job LIST.

There is an additional job control statement used to identify the end of data in a job stream. For example, if the program CHKREG (Figure 5.2) required some data cards to execute correctly, the data cards must follow immediately after the // EXEC CHKREG statement. The end of the data cards must be identified somehow. This is done with a /* statement (see Figure 5.2 1).

It was discussed earlier in this course that DOS provides the ability to assign symbolic addresses to physical devices. This is accomplished through the use of the ASSGN statement. Review the following statement.

```
// ASSGN SYS006,X'181'
```

This statement assigns the symbolic address of SYS006 to device 181. Figures 5.3 and 5.4 show how the two formats of the ASSGN statement are used.

The // ASSGN statement 1 in Figure 5.3 is a temporary assignment. That is: symbolic unit SYS006 will be assigned to device 181 for the duration of job SORTPAY only. When the // JOB LIST statement 2 is encountered by Job Control, the original assignment of SYS006 reverts back to what it originally was before the // ASSGN statement was encountered.

The ASSGN statement 1 (NO //) used in Figure 5.4 will permanently assign symbolic unit SYS006 to device 181. This assignment will also be in effect for the job LIST 2.

The difference between this ASSGN statement and the // ASSGN statement used in Figure 5.3 is the absence of the //. The permanent assignment will remain in effect until another permanent ASSGN statement to X'181' is encountered or until the system is IPL ed. Whereas, the temporary assignment (//) remains in effect only for one job. (Until Job Control encounters a /& or another // JOB statement.)

When a temporary assignment is made, the LUB pointer is changed as we saw in SESSION 2, but the Supervisor stores away the original PUB pointer, so that the LUB can be restored at the end of the job.

JOB CONTROL STATEMENTS - SPECIFICS

The following reading assignment consists of introductory paragraphs, a list of job control statements, and a detailed description of each statement. It is not necessary to learn the details of the statement. Rather read the introduction paragraphs, look over the list of statements and then note the operand field of the statements listed under the reading assignment.

Refer to DOS/VS Operating Procedures (GC33-5378)

Read: In the Reference Information Section, Job Control Statements.

// ASSGN
// EXEC
// JOB
// LISTIO
// OPTION
// PAUSE
/*
/&
*

OPERATOR COMMANDS

The following reading assignment is structured similar to the previous one. Do not try to learn all the details. Read the introduction paragraphs. Look through the list of commands, noting the type; ie, IPL, job control or attention.

Then note the information in the operand field of the commands listed under the reading assignment.

Refer to DOS/VS Operating Procedures (GC33-5378)

Read: In the Reference Information Section, Operator Commands.

ADD
ALLOC
ASSGN
CANCEL
DEL
DPD
DVCDN
DVCUP
EXEC
LISTIO
MAP
PAUSE
ROD
SET
UCS

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 5-1)

1. Code the job control statement required to execute a job called INPUT.

   _____

2. Code the job control statement that is used to indicate the end-of-data.

   _____

3. Code the job control statement to permanently assign SYSLST to tape drive 184.

   _____

4. (True/False) A JOB must be terminated by a /& card.

5. Which component of DOS cannot accept the ASSGN statement?

   a. JCS
   b. SPI
   c. AR
   d. JCC

   Alloc F1=64K, F2=65K, F3=40K

6. The above alloc command is invalid because:

   a. A partition must be allocated to an even number.
   b. Minimum storage allocation for a virtual partition is 64K.
   c. Both a and b.

SECTION 5-2 - MISCELLANEOUS JOB CONTROL STATEMENTS

The previous section introduced the job control statements required for job execution. Some of the more commonly used statements will now be discussed.

OPTION STATEMENT

The customer at SYSGEN time establishes some options that are to be in effect during system operation such as:

(1) The option of punching or not punching an object module each time a program is compiled.
(2) The option of dumping or not dumping the contents of the general registers and core storage when a program comes to an abnormal end.

Options selected at SYSGEN time can be over-ridden for a specific job by the use of the OPTION statement. Some of the IBM Control programs, such as the Linkage Editor, require the OPTION statement for correct operation. Uses of the OPTION statement are shown in Figure 5.5.

Refer to DOS/VS Operating Procedures (GC33-5378), Section: Operator Commands - //OPTION

When do the options in the OPTION statement revert back to what they were at SYSGEN time?

_____

_____

Answer

The options in an OPTION statement remain in effect for the job in which they are found. When the next JOB statement or /& statement is read, the options revert back to those originally set at SYSGEN time.

PAUSE STATEMENT

The PAUSE statement causes job control processing to pause and unlocks the keyboard. Processing will continue when the operator responds with the EOB/END Key. If the PAUSE statement contains comments, they will print on SYSLOG. This provides the operator with instructions and stops processing until the operator is ready to continue. An example of using the PAUSE statement is shown in Figure 5.6. The comment in the PAUSE will inform the operator to mount a tape.

Refer to DOS/VS Operating Procedures, Section: Operator Commands
- PAUSE statement

The //PAUSE statement and associated comments, if any, will be listed on _____.

Answer

SYSLOG (the console printer or display).

LISTIO STATEMENT

The LISTIO command is used by the operator to check the symbolic assignments of the system. This command has various operands that can provide the symbolic assignments in a variety of formats. Figure 5.7 is a partial SYSLST printout as a result of using the following statement:

// LISTIO ALL

The printout lists the Background system logical units (class 00) 1 , and the Background programmers logical units (class 01) 2 with their assignments 3 . The assignments are then listed for each remaining partition. This is actually a printout of the LUB table. UA-indicates a device is unassigned.

Refer to DOS/VS Operating Procedures, Section: Operator Commands
- LISTIO

The // (slashes) has added meaning in the LISTIO command. If the // (slashes) are present in the command, the printout will be received on SYSLST, but if the // (slashes) are omitted (LISTIO ALL) the printout will be an SYSLOG.

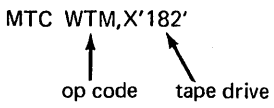
Code the LISTIO statement to print all the assignments for the background partition on SYSLST.

Answer

The statement would be coded as follows: // LISTIO BG

MAGNETIC TAPE CONTROL COMMAND

The Magnetic Tape Control (MTC) command is used to issue control commands to tape drives. One advantage of the MTC command is that it can be used while the system is running. For example, the operator can write a tape mark on drive 182 by inputting the following command on the console typewriter:



The tape drive doesn't have to be taken off-line as is done when the TM is written with the CE panel. The other uses of the MTC command will be covered later in a reading assignment.

Refer to DOS/VS Operating Procedures (GC33-5378), Section: Operator Commands - MTC (command)

Code the MTC command to write a tape mark on tape drive 281.

Answer

MTC WTM, X'281'

CANCEL COMMAND

The CANCEL command can be issued either as an ATTENTION command or JOB CONTROL command. How it is used will determine which partition is cancelled. Examples:

1. Issuing cancel with no operand.

   a. In the attention routine it cancels the BG (Background) partition.
   b. As a job control command it will cancel the partition in which it is issued.

2. Issuing cancel with an operand will cancel the specified partition.

Refer to DOS/VS Operating Procedures, Section: Operator Commands -Cancel

Issuing the cancel command will cause the job running in the partition to:

a. Cancel immediately.
b. Cancel immediately after all outstanding interrupts have been handled.

Answer

b. Outstanding interrupts will be handled prior to cancelling the job.

CANCEL RESPONSE

Another cancel command, the C/CANCEL, is used by the operator to cancel a line of information that the operator is in the process of typing on the console. For example, the operator was typing the following:

MTC WTM,X 281

and realized he made an error. The CANCEL Key on the console typewriter, when depressed, will cancel the line and the operator can re-enter the command.

Summary

The statements and commands just covered are the most commonly used by the system operator. There are many other statements and commands. If you have a need for them, refer to the DOS/VS Operating Guide in your customer's library.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 5-2)

1. Submit the command to printout on SYSLST, the logical units assigned to 00C only.

   _____

2. Submit the command to backspace 3 records on tape drive 180.

3. The options in the OPTION statement remain in effect:

   a. until the end of a JOB.
   b. for the specific job step only.
   c. until the next /*.

SECTION 5-3 - SYSTEM OPERATION

This section deals with the function of Job Control and how it relates to program execution by means of job control statements.

Two job streams will be used to discuss system operation.

During the session on IPL procedure, a hardware system was introduced along with the symbolic units that were required for an IPL. A similar hardware system will be used in this session.

Our first job stream will be an assembly of a source program.

Assembler Job Stream (Initial Setup) Figure 5.8

This section will show various job streams and the interaction with the various system components. The first job stream will be an assembly of a source program. Session 2 of this course covered IPL procedures and we will start the operation from that point. Figure 5.8 is used in conjunction with the following text to explain this system operation.

1. The system has been IPL'ed, as shown by the messages on the console typewriter (SYSLOG).

2. The job that will be used for the first job stream consists of an assembly of a source program. Observe the job stream closely and note it will be read from the card reader.

3. In the LUB table, the symbolic units required have arrows pointing to the physical units in the PUB table. The device type is indicated in the PUB. Locate these symbolic units on Figure 5.8. (Note: Remember, the Assembler manual would list the symbolic units required.)

4. The symbolic units SYS001, SYS002, and SYS003 are work files for the assembler. These symbolic units can be assigned to the same physical device as SYSRES or any other tape or disk device.

5. Job Control is in storage and has control of the system. Job Control's main function is to read Job Control cards to set up a job to execute.

6. The operator enters an EOB/END on SYSLOG. This directs Job Control to read Job Control statements from SYSRDR.

7. The first Job Control card read is // JOB SAMPLE. This informs Job Control that the job being started is SAMPLE.

Assembler Job Stream (Job Control) Figure 5.9

1. Job Control prints the name of the job on SYSLOG. Also, the start time for the job is printed on SYSLOG.

2. Job Control, having processed the // JOB card, now goes to SYSRDR to obtain the next control statement. This statement is a // EXEC statement. Job Control, when it encounters an EXEC statement, checks to see if there is an operand. This execute card has an operand of ASSEMBLY. Therefore, job control recognizes this as the next problem program to execute.

3. Job Control signals the Supervisor (system loader) that it is to load the ASSEMBLY program into storage.

Assembler Job Stream (Assembly Time) Figure 5.10

1 Place the name of the program that is now executing in the blank provided in storage and draw an arrow from storage to where this program is located in the core image library.

Answer

The name of the program in storage is ASSEMBLY and there should be an arrow from this problem program to the last program (ASSEMBLY) in the core image library on SYSRES. The Assembly program overlayed Job Control.

2 The assembler reads its input from the symbolic unit SYSIPT. The input to the assembler is a source program.

3 The assembler will read cards from SYSIPT until a /* card is encountered. A /* is an end-of-data indicator.

4 The assembler will now use the work files SYS001, SYS002, and SYS003 as work files to create an object module and a source listing.

5 The assembler outputs an object module on SYSPCH. The customer at SYSGEN selects certain system options. DECK is one of the options he can specify. For this job, assume the customer specified DECK, which will cause an object module to be punched when a language translator (assembly) is executed.

6 The assembler outputs an assembly listing of the source program on SYSLST.

7 The assembly program has now completed execution. It signals the supervisor that it has finished and the Supervisor (system loader) now loads the next program.

Assembler Job Stream (End-of-Job) Figure 5.11

1 Place the name of the program that is now executing in the blank provided in storage and draw an arrow from storage to where this program is located in the core image library.

Answer

The name of the program now executing in storage is Job Control. There should be an arrow from storage to the Job Control program in the Core Image library on SYSRES.

2 Job Control's main purpose is to read and process Job Control statements. Job Control reads the next control statement from SYSRDR. The next control statement is a /& (end-of-job) command.

3 Job Control now indicates on SYSLOG that the job SAMPLE has ended. It also logs the time the job ended and the time duration for the job.

4 Job Control now goes to SYSRDR for the next control statement and the reader indicates a 'unit exception' (ran out of cards). Job Control then informs the operator of this condition with a 1C00A message and unlocks the keyboard on SYSLOG and waits for more operator instruction.

How many job steps were there in the job stream just covered?

Answer

There was only one job step: // EXEC ASSEMBLY

Assemble, Linkedit and Execute Job Stream (Initial Setup) Figure 5.12

The next job stream that will be discussed will be for the following job steps:

- Assembly (compile time)
- Linkage Editor (linkedit time)
- Execute Problem Program (execution time)

1 Assume that IPL has been completed and the operator enters an EOB/END command on SYSLOG.

2 The (B) informs Job Control to read control cards from SYSRDR.

3 Observe this job stream closely. The first card read by Job Control will be the // JOB SAMPLE card.

4 This control card names the job and will be printed on SYSLOG along with the time the job started.

Question

Since this job stream will linkedit a program, an additional symbolic unit of _____ is required.

Answer

SYSLNK 6

5 The next card processed is a temporary assign for SYSLNK.

6 Note the assignment of SYSLNK in the LUB table to the physical device X'130' in the PUB table.

7 Symbolically, SYSLNK is shown as an area on the same disk pack with SYSRES.

8 The next job control statement read by Job Control is an OPTION card. The OPTION card specifies options that will be in effect for this job.

Review in the DOS/VS Operating Procedures (GC33-5378), Job Control Statements Section: - OPTION (pay close attention to the LINK and DECK parameters)

Job Control processing of the OPTION card consists of setting or resetting switches in an area of the Supervisor. This area in the Supervisor is called the Communications Region, and the information within it is available to all programs for intra-program communications. For this OPTION card, the NODECK and LINK switches will be set on. It will be shown later in this session how these switches are used.

9 The next card processed will be the // EXEC ASSEMBLY statement.

10 Job Control will signal the system loader portion of the Supervisor to load the ASSEMBLY program into core and overlay Job Control.

Assembly, Linkedit and Execute Job Stream (Assembler) Figure 5.13

1 The program ASSEMBLY is loaded into storage from the core image library.

2 The assembler goes to SYSIPT to read the source deck.

3 The /* indicates the end of the source cards.

4 The assembler uses work files SYS001, SYS002, and SYS003 to compile the source program into an object program.

5 The ASSEMBLY program tests the LINK switch in the supervisor communications region. Finding this switch on, it will write the object module on SYSLNK. The ASSEMBLY program then tests the NODECK switch and finding it on, the assembler will not output an object module on SYSPCH.

6 The assembler program produces a program listing on SYSLST.

7 The ASSEMBLY program has now completed its function and signals the system loader in the Supervisor that it is finished.

Assemble, Linkedit, and Execute Job Stream (Linkedit) Figure 5.14

1 The Job Control program is loaded into storage overlaying the ASSEMBLY program.

2 Job Control goes to SYSRDR to read Job Control statements. Job Control reads the // EXEC LNKEDT card.

3 Job Control recognizes the operand LNKEDT as the DOS name of the Linkage Editor and signals the system loader portion of the Supervisor to load this program into storage.

Assembly, Linkedit, and Execute Job Stream (Linkedit Time) Figure 5.15

1 The Linkage Editor program is loaded into storage from the core image library.

The function of the Linkage Editor in the DOS System is: _____

_____

Answer

Converts the output of a language translator (Object module) to a format that is executable using DOS.

All programs in DOS are loaded into storage from the _____ _____ library.

Answer

Core Image

As the questions indicate, the object module produced by the Assembler must be converted to an executable format and be placed in the core image library. Let's see how this is accomplished

2 The Linkage Editor goes to SYSLNK for its input. The input to the Linkage Editor will be the object module that was written there by the assembler program.

3 The Linkage Editor uses SYS001 as a work file when converting the object module to an executable format.

4 The output of the Linkage Editor must be placed in the core image library. The output will be placed in the unused portion of the core image library. The unused portion (temporary portion) of the library starts immediately following the last program catalogued in the library.

5 This is the problem program that the Linkage Editor converted to an executable format and placed in the temporary portion of the core image library.

6 The Linkage Editor has now completed its processing and signals the system loader in the Supervisor that it is finished.

Assemble, Linkedit, and Execute Job Stream (Job Control) Figure 5.16

1 Job Control is loaded into storage from the core image library and overlays the Linkage Editor.

2 Job Control goes to SYSRDR to read the next control card. The next control card is a // EXEC command with no operand. The omission of the operand in the EXEC card signals Job Control that the next program to be executed is the program that the Linkage Editor just placed in the temporary portion of the core image library.

3 Job Control signals the system loader in the Supervisor to load the problem program into storage from the temporary portion of the core image library.

Assemble, Linkedit, and Execute Job Stream (Execution Time) Figure 5.17

1 The problem program is loaded into storage from the temporary portion of the core image library.

2 The problem program for this example may read data cards from SYSIPT, print a report on SYSLST, etc.

3 When the problem program has completed execution, it signals this to the system loader in the Supervisor.

Assemble, Linkedit, and Execute Job Stream (End-of-Job) Figure 5.18

1 Job Control is loaded into storage from the core image library.

2 Job Control reads a job control card from SYSRDR. The card read is a /& which indicates end-of-job.

3 Job Control prints on SYSLOG an end-of-job message, the time the job was completed, and the duration of the job.

The Job Control Program must do some clean-up work on the system due to various actions performed directly related to running this job. List below the three areas/items that you think Job Control must restore to the original status. (before the job SAMPLE was run).

a. _____

b. _____

c. _____

Answer

a. Since the ASSGN for SYSLNK was temporary, the SYSLNK assignment in the LUB must be reset to its original value.
b. The OPTION card caused the LINK and NODECK switches to be set in the communications region for this job. These switches must be reset.
c. Remove all reference to the problem program stored in the temporary area of the core-image library. The reference to this program is only valid for the duration of this job stream.

4 The assignment of SYSLNK in the LUB is restored (SYSLNK is now unassigned)

5 Reference to the problem program in the temporary area of the core-image library is removed.

6 The LINK and NODECK switches are reset in the supervisor.

7 Job Control goes to SYSRDR and gets a unit exception (ran out of cards). A 1C00A message is issued to the system operator, the typewriter keyboard is unlocked, and Job Control waits for more instructions.

Summary

System operation will be reinforced in the follow-on lab course, where you will be required to solve trouble analysis problems in the job streams just covered. As a review of these job streams, do the following work project.

Work Project

Number the items in Figure 5.19 in the correct sequence (1-18) as they would occur to assemble, linkedit and execute a problem program.

The first item (Job Control input) is numbered as a starting reference.

The correct sequence can be found in Figure 5.20. You may use Figure 5.20 for hints if you get confused, but try to do as much as you can without constantly referring to this completed figure and its associated explanation.

Answer the following study questions, and when you have completed them, input your answers via the terminal.

Study Questions (Section 5-3)

1. From which of the following symbolic units does JOB CONTROL read Job Control cards?

   a. SYSIPT
   b. SYSRDR
   c. 00C

2. The ASSEMBLER program will write an object module on SYSLNK:

   a. When a // EXEC LNKEDT card is read.
   b. When a // EXEC card is read (no program name specified).
   c. If a // OPTION LINK card has been read.

3. Which symbolic unit does the LINKAGE EDITOR go to for input?

   a. SYSRDR
   b. SYSIPT
   c. SYSLNK
   d. 190

4. How many job-steps are there in the job-stream shown in Figure 5.20?

   a. 3
   b. 1
   c. 20
   d.

5. Why is the /* card required in Figure 5.20?

   a. To recognize the end of job-step 1.
   b. To recognize the end of data cards from SYSIPT.
   c. To recognize the end of data cards from SYSRDR.
   d. To recognize the end of the job.

6. As a result of reading a // OPTION statement, JOB CONTROL will:

   a. Set the appropriate switches in the COMREG of the problem program.
   b. Put the object module on SYSLNK.
   c. Set the appropriate switches in the COMREG of the Supervisor.
   d. Signal the system loader to load the Linkage Editor.

The next three questions consist of error messages that printed as a result of an incorrect procedure. Choose the correct action that the operator should take in each situation. Refer to Appendix C in the SCM for messages and their meaning.

7. Choose the correct operator action for the following situation:

```
//ASSGN SYSLNK, X'280'
1A04D INVALID I/O ASSIGNMENT
```

   a. cancel the job.
   b. ignore the message and continue processing.
   c. reassign SYSLNK to a DASD device.
   d. obtain a system dump and call the CE.

8. Choose the correct operator action for the following situation:

```
// OPTION LNKEDT, NODECK
1S03D INVALID STATEMENT
```

   a. ignore the message and continue processing.
   b. resubmit the statement as // OPTION LINK, NODECK
   c. resubmit the statement as // OPTION LNKEDT
   d. resubmit the statement as // OPTION LNKEDT, DECK

9. Choose the correct operator action for the following situation:

```
// EXEC ASSEMBLE
1C33A PROGRAM NOT FOUND
```

   a. type in CANCEL to cancel the job.
   b. mount the correct SYSRES pack on the system and resubmit the job.
   c. take a dump and call the CE.
   d. resubmit the statement as // EXEC ASSEMBLY

SECTION 5-4 - CHECKPOINT/RESTART FACILITY

Purpose of Checkpoint

Suppose that during a two hour sort operation, a job is aborted because of a power failure. There are three things a customer could do to recover from that situation. First would be to start the job over at the beginning, which would be, from IBM's and the customer's point of view, the worst solution. Second would be to restart the job at the point of the power failure. This would be the best solution. Third would be to restart the job at some point near the time the power failed. This would be a real-life solution.

In order to restart the program, the program must save (checkpoint) all the information about the program. Checkpoint then, can be defined as a program facility to record the status of the system on tape or disk. Using this information, the job can be restarted with a minimal amount of lost time.

A checkpoint record contains:

- A snapshot of the partition
- The General Registers
- Special Restart Information

This information is required so that Job Control can re-establish the partition.

The responsibility of taking checkpoints is up to the customer. His problem program controls when a checkpoint will be taken. This can be at End of Volume, on Record Count, Timer Interval, Operator Interrupt, etc.

Checkpoint Concepts

Refer to Figure 5.21. This is an example of a program that is set up to take checkpoints every thirty minutes, using the interval timer 1. As the program is processing, it keeps checking to see if the 30 minute period has elapsed 2. The end of the period will cause a timer interrupt which will branch to the routine that writes the checkpoint record 3 on the checkpoint file 4. After the record is written, the program will notify the operator 5 with a message on the documentary console 6 and will set up a new 30 minute interval 1. If the job should be aborted for any reason, the operator could restart it using the last checkpoint record.

All the pertinent information necessary to restart will have been saved in the checkpoint record. It will contain the PSW, the general registers, the entire problem program area of storage, and information concerning where the I/O devices are physically positioned. (Determines the number of records already read on tape, for example.)

Refer to Figure 5.22. This figure represents the job control statement required to restart a program from the checkpoint record. 1 This is the job control command. 2 This is the symbolic address of the checkpoint file. 3 This is the number of the checkpoint that will be used to restart the program. That information is printed out on the documentary console each time a checkpoint record is written. 4 This is the filename of the checkpoint file.

Answer the following questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 5-4)

1. (True/False) The checkpoint file must be on either a tape or disk device.

2. (True/False) It is the responsibility of the CE to take checkpoints.

SESSION 6 - MAPS, STORAGE PRINTS, AND SYSTEM MESSAGES

SECTION 6-1 - ASSEMBLER LISTING AND LINKEDIT MAP

The first page of this section will consist of a review of the Assembler Listing. Refer to Figure 6.1 (part 1). The first page of an Assembler Listing is the External Symbol Dictionary. This information is only meaningful to a programmer and will not be discussed in this course.

SOURCE AND OBJECT PROGRAM

Refer to Figure 6.1 (parts 2 and 3).

This section of the listing documents the source statements and the resulting object program.

1. This is the deck identification. It is the symbol that appears in the name field of the first TITLE statement.

2. This is the information taken from the operand field of a TITLE statement.

3. Listing page number.

4. This column contains the assembled address (hexadecimal notation) of the object code.

5. This column contains the object code produced by the source statement. The entries are always left-justified. The notation is hexadecimal. Entries are machine instructions or assembled constants. Machine instructions are printed in full with a blank inserted after every four digits (two bytes). Constants may be only partially printed.

6. These two columns contain effective addresses (the result of adding together a base register value and displacement value):

   1. The column headed ADDR1 contains the effective address for the first operand of an SS or an SI instruction.

   2. The column headed ADDR2 contains the effective address of the second operand of any instruction referencing storage.

   Both address fields contain six digits; however, if the high order digit is a zero, it is not printed.

7. This column contains the statement number. A plus sign (+) to the right of the number indicates that the statement was generated as the result of macro-instruction processing. Macro will be discussed later in the course. The maximum statement number is 65,535. If there are more than 65,535 statements, the statement number wraps-around (begins at 1 again).

8 This column contains the source program statement. The following items apply to this section of the listing:

a. Source statements are listed (including those brought into the program by the COPY assembler instruction, and macro-definitions submitted with the main program for assembly).

b. The statements generated as the result of a macro-instruction follow the macro-instruction in the listing.

c. Diagnostic messages are not listed in-line in the source and object program section. An error indicator, ***ERROR***, 9 appears following the statement in error. The message appears in the diagnostic section of the listing.

d. Literals will appear in the listing following a LTORG or END statement, or both. Literals are identified by the equals sign (=) preceding them. 10 (Figure 6.1, part 3)

e. If the END statement 11 (part 3 of 6) contains an operand, the transfer address will appear in the location column (LOC).

f. For a USING statement 12 (Part 2 of 6), the location column contains the value of the first operand.

13 This field indicates the assembler level and version number (eg, DOS/VS-V28.0) reads as DOS/VS assembler, version 28.0.

14 Current time and date obtained from SET card and timer.

Refer to Figure 6.1 part 4. This section of the listing contains the relocation dictionary information passed to the linkage editor. The entries describe the address constants in the assembled program that are affected by relocation. The information is only meaningful to a programmer and will not be addressed in this course.

CROSS REFERENCE

Refer to Figure 6.1 part 5.

This section of the listing information concerns symbols -- where they are defined and used in the program.

15 This column contains the symbols.

16 This column states the length (decimal notation), in bytes, of the field occupied by the symbol value.

17 This column contains either the address the symbol represents, or a value to which the symbol is equated.

18 This column contains the statement number of the statement in which the symbol was defined.

19 This column contains the statement number in which the symbol appears as an operand. (As an example "again" was defined in statement 5 and referenced in statement 13.)

The following notes apply to the cross-referencing section:

- Symbols appearing in V-type address constants (external references) do not appear in the cross-reference listing.
- A PRINT OFF listing control instruction does not affect the production of the cross-reference section of the listing.
- Undefined symbols appear in the cross-reference section. However, only the symbol column and the reference column have entries.

Following the Cross Reference is the diagnostic section. Refer to Figure 6.1 (part 2). Statement 17 has caused an error. The assembler program will analyze the error and will print a message in the diagnostic section.

DIAGNOSTICS

Refer to Figure 6.1 (part 6).

This section contains the diagnostic messages issued as a result of error conditions encountered in the program.

20 This column contains the number of the statement in error.

21 This column contains the message identifier.

22 This column contains the message.

The following note applies to the diagnostics section:

- A message identifier (ERROR CODE) consists of six characters and is of the form:

IPKxxx

IPK
identifies the issuing agent as DOS/VS assembler.

xxx
is a unique number assigned to the message.

Explanatory notes for each message are contained in DOS/VS Assembler Programmer Guide (not issued in this course).

23 Statistical messages may appear in the listing.

The preceding information will be printed out each time an assembly operation is performed.

LINKEDIT MAP

A Linkedit Map is normally printed every time a // EXEC LNKEDT job control card is processed.

Refer to Figure 6.2.

1 PHASE - This column contains the name of the program. This is the CI (Core Image) library name.

2 XFR-AD - (Transfer Address) This is the address (in core) that control will be passed to after the program is loaded.

3 LOCORE - Contains the lowest core address taken up by this program. In our example, the transfer address and low core address are the same. This will not always be true, as there could be a section in front of the main body of the program.

4 HICORE - This is the highest core address taken up by this program.

5 DSK-AD - This is the disk address of where this program is located in the core image library. It is in the format of cylinder, track, and record. Our example shows cylinder 31, track 04, record 2.

6 ESD TYPE - This item is of value to programmers only, and will not be discussed.

7 LABEL - This is the name of the first label in the program.

8 LOADED - The core address where this program is loaded.

9 REL-FR - Relocation factor. This is the hexadecimal difference of the address the program was assembled at and where it is located in core. The relocation factor can be either plus or minus.

- Plus, if the assembled address is lower than the load address.
- Minus, if the assembled address is higher than the load address.

For our example, the relocation factor is the same as the low core address. This means the assembled address of this program was 0000. If the assembled address of the program was hex 1000, the relocation factor would then be 0026A8. It is necessary to know the relocation factor of a program when cross-referencing between the program listing and a storage print or dump. Storage prints will be covered in the next section.

Study Questions (Section 6-1)

1. (True/False) The Cross-Reference section of an assembler listing contains information concerning symbols, where they are defined and used in the program.

2. Select the correct message identifier for a DOS/VS assembler.

   a. IPK
   b. IJY
   c. IJX
   d. IJZ

3. (True/False) A linkedit map is normally printed everytime a // EXEC LNKEDT card is processed.

4. Select the item that gives the hex difference between the address the program was assembled at and where it is located in the core.

   a. LOADED
   b. XFR-AD
   c. LOCORE
   d. REL-FR
   e. LABEL

SECTION 6-2 - STORAGE PRINTS

Two types of storage prints will be introduced in this section. First will be a DOS/VS stand-alone storage dump and then the DOS/VS system dump.

Stand-Alone Dump Program

Introduction

Your customer will have a program generated for the purpose of obtaining a dump of storage. It is a program that must be IPLed from either the card reader or a tape drive. It is used for problem determination when, for example, the system goes into a wait state.

Depending on the option used in generation of the stand-alone dump program, either a conventional or formatted dump will be generated. The formatted dump will provide a more readable form of printout of the DOS tables, such as PUB, LUB, Error Recovery Blocks, etc.

After the following reading assignment, you will be given an opportunity to examine the contents of a stand-alone storage dump. Therefore, during the reading assignment, do not attempt to learn all the details; rather concentrate only on the overall concept.

Refer to the DOS/VS SADP Manual (GC33-5380).

Read: Section 2-A DUMPGEN and Stand-alone Dump.
- DUMPGEN
- Stand-alone Dump Program

From the above reading assignment you should have noted the following major points:
- The customer used the DUMPGEN program to generate the stand-alone storage dump program.
- The stand-alone storage dump program is loaded via standard IPL procedure.
- The program is loaded into a noncritical area of storage - the program check routine area.
- The IPL card contains the above address, therefore the correct information must be in this card for each Supervisor.
- Certain bytes of low storage must be displayed prior to running the stand-alone storage dump program as these locations are destroyed by the program.
- The STORE STATUS function must be performed prior to executing the stand-alone dump if the control register contents are required. (For a review of the store status function, refer to Session 4 of this course.)

Let's now look at a stand-alone storage printout.

Refer to DOS/VS SADP Manual (GC33-5380), Appendix G. (Example of a stand-alone dump output.)

Read the text preceding the printout and then thumb though the printout noting the various items contained therein. Try to form an opinion as to which items you think may be of importance to you as a hardware CE. When you finish with the above, read the following text.

Storage Print

The first items on the storage printout are the contents of the general purpose registers. The next item includes the PSWs and are in Extended Control (EC) mode.

Notice that the last interrupt code is shown for each PSW.

Following the PSWs are the CSW and CAW.

The above items are normally well worth knowing when starting problem determination.

The next item is the present status of the timer.

Refer to the above items. You may need the System/370 Reference Summary Card (GX20-1850) (yellow card).

1. What is the contents of GP Register D? _____

2. What is the address of the device that caused the last I/O interrupt? _____

3. What caused the program interrupt? _____

Answers

1. 00000029
2. 00C
3. Addressing exception 0005

The next item on the printout is labeled Block 246. This is the last 2K block of real storage. The reason it is printed first is that some portion of the stand-alone dump program will be loaded into this high storage location and therefore the contents of this area is printed prior to loading the program.

As noted previously in this session, certain bytes of information in low storage must be displayed prior to running the stand-alone dump. The note on the printout following Block 246 explains this. The information in the first several bytes of low storage is invalid.

Most of the remaining information in the printout has not yet been discussed. However, you should be familiar with the LUB and PUB tables. These can be found in the printout. Thumb through the printout until you find these two items and then use this information to answer the following. (Use the formatted LUB/PUB.)

1. What is the physical address assigned to:

BG SYSTEM LUBS
SYSRDR _____
SYSPCH _____
SYSLST _____
SYSLOG _____
SYSVIS _____

Answer

BG SYSTEM LUBS

00 SYSRDR 01 FF 00C
02 SYSPCH 02 FF 00D
03 SYSLST 03 FF 00E
04 SYSLOG 00 FF 009
0C SYSVIS 07 FF 131

If you had difficulty with the last question, refer back to Session 2-2 for a review.

One last item should be noted on the printout. There are many areas on the printout that will look as follows:

003C40 00000000---SAME---
003CC0 F1F161F1, etc

The comment ---same--- indicates that the program has encountered the same information in storage for more than one line. It skips those positions and puts one line showing what was contained in those positions.

More areas of the stand-alone dump will be discussed later.

DOS System Dump

This dump is the result of an abnormal condition or operator request. This dump is outputted by the DOS/VS system and is an option of the DOS/VS Supervisor. If option dump has been specified, a dump will occur for each abnormal job termination. If option no dump has been specified in the Supervisor, no dump will occur.

Refer to IBM System/370 DOS/VS SADP Manual (GC33-5380).

Read: Section 2-1
// OPTION DUMP through but not including DUMPGEN.

Refer to Figure 6.3 (part 1).

1 This item contains the name of the program that was running when the dump occurred.

2 This field contains the date.

3 This field contains the page number of the dump.

4 These two lines contain the general purpose registers.

5 This line contains the floating point registers.

6 These two lines contain the control registers.

7 This line is a pointer to the communication region of the DOS supervisor. The Comreg will be covered in a later session.

8 This is the main body of the dump starting from storage location 0000. Each line of print contains 32 bytes of data (20 hex bytes). On this dump, the PSWs, CSW, and CAW are not printed separately as they were on the stand-alone dump. To find the I/O old PSW, you would have to look at hex 38 to 3F on the dump. 9

Refer to Figure 6.3 (part 2).

This is a continuation of the DOS System dump. When storage contains the same information for more than one line, the program prints SAME, 10 until different information is encountered.

11 This is an interpretation of the data in the dump. Unprintable characters appear as a period (.).

You will now be shown how to use the assembly listing, linkedit map and system dump to locate a problem.

Refer to Figure 6.4 (parts 1 to 11). This is a sample program that has been assembled and linkedited. When the program began to execute, it caused a Program Check, canceled the job, and gave a system dump.

The first thing that should be done is to look at the message that was printed as a result of the Program Check.

Refer to Figure 6.4 (part 4).

System cancel messages will be covered in detail in the next section. For now, all we will be concerned with is, 1 this tells us that a program check has occurred. 2 This tells us that the failure occurred at hex 0036C2. In order to find the instruction in the assembler listing that caused the program check, you have to first determine the relocation factor.

Refer to Figure 6.4 (part 3).
By using the linkedit map, the relocation factor (REL-FR) can be determined to be 0036A8. 3 The program check was at the true address 0036C2. By subtracting the relocation factor from the true address, we arrive at the assembler listing address 00001A. The next step is to locate this instruction in the assembler listing.

Refer again to Figure 6.4 (part 1).
The instructions at 00001A is a move character instruction 4 . The B1 register being used is 0 5 . This caused the program check because the instruction is attempting to move data into the Supervisor area which is storage protected.

Figure 6.4 parts 5 to 11 represent the dump taken when the job cancelled.

Refer to Figure 6.4 (part 5).
The Program Old PSW is located at hex 28 to 2F 5 . The interruption code 6 is 0004 (Protection). The next instruction address is 0036C8 7 . This is the address of the instruction following the MVC instruction that caused the Program Check.

Refer to Figure 6.4 (part 11).

8 This item represents the failing instruction in the dump.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 6-2)

Use Figure 6.4 (parts 1 through 11) to answer the following questions.

1. The TM instruction at location 000096 in the Assembly listing is located at what address in main storage when this program is executed.

   a. 000092
   b. 003740
   c. 003730
   d. 00373E

2. What information is contained in the I/O New PSW?

   a. FF15000E 9000373A
   b. 00040000 00000324
   c. 00040000 00001BC0
   d. 5B5BC2C5 D6D1F340

3. What information is contained in Main Storage location 003790 to 003797?

   a. READCCW
   b. I/O Area for both the Reader and Printer
   c. PRINTCCW

4. What status is contained in the Channel Status Word?

   a. Channel End - Incorrect Length
   b. Channel End - Program Check
   c. Channel End - Protection Check
   d. Channel End
   e. Device End

5. What address is in the Channel Address Word?

   a. 0000
   b. 29C0
   c. 29D0
   d. 373A

SECTION 6-3 - SYSTEM MESSAGES

This section will discuss system messages.

Portions of the following text have been extracted from the DOS/VS Messages (GC33-5379).

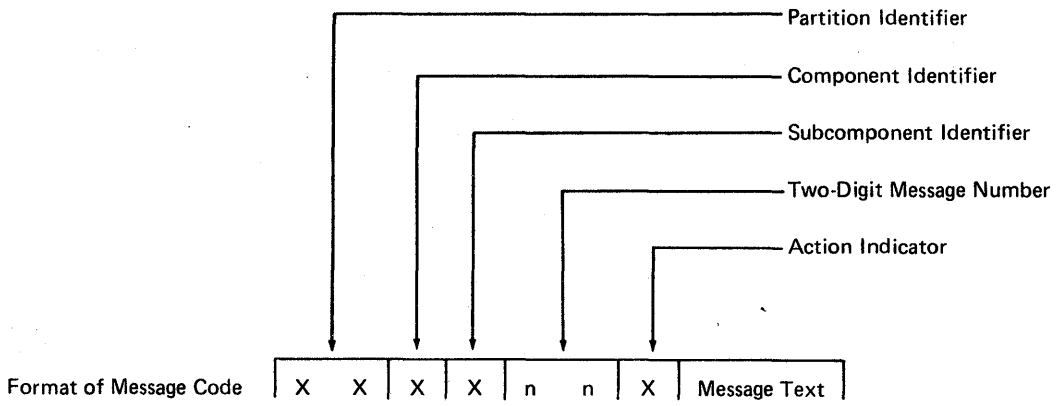
Note: Appendix C of the Supplementary Course Material contains selected messages extracted from the above manual.

The Message Code

Each DOS/VS message is preceded by a string of characters, the so-called message code. This message code tells you (1) which system component or program product issued the message and (2) what type of action you should take.

The message code primarily serves as a pointer to the location in this manual where the message and the corresponding explanation is given in detail.

The format of the message code is shown and described in detail below.



These various identifiers can have the following values:

Partition ID	Component ID	Subcomponent ID	Message Number	Action Indicator
BG F4 F3 F2 F1 AR	0 I Z	0 I Z	00 I 99	A = Action D = Decision I = Information E = Eventual Act W = Wait

Format of Message Code

Partition Identifiers

The first two characters indicate the partition from which the message was issued. The following identifiers are used:

BG	program executing in background
F4	program executing in foreground 4
F3	program executing in foreground 3
F2	program executing in foreground 2
F1	program executing in foreground 1
AR	Attention Routine

When a Supervisor routine such as OPEN or device error recovery is operating on behalf of a user program, any message issued contains the identifier for the partition in which the user program resides.

	COMPONENTS								
	0 Supervisor and IPL Messages	1 Job Control, Buffer Load Attention Routine and POWER Messages	2 Linkage Editor Messages	3 Librarian and EREP Messages	4 LIOCS, Telecom Debug Aids and ESTV Messages	8 System Utilities Messages	9 Access Method Services Messages	A Assembler Messages	E OLTEP and Emulator Messages
SUBCOMPONENTS	C Checkpoint D DOC I IPL P PIOCS R Restart S EOJ T MCAR/CCH V VSS	A Assgn Routine B Buffer Load Program C Job Initiation and Termination I Attention Routine L Label Error M/N Cataloged Procedures P Attention Routine Q/R POWER S JCL		C CORGZ E EREP M MAINT U MAINTUP	1 Tape Handling n Access Methods B BTAM C PDAIDS SDAIDS E ESTV M MICR/OCR Control P Data Check Q QTAM V VTOC Display/Dump	0 Copy and Restore Disk or Data Cell 1 Initialize Disk or Data Cell 2 Assign Alternate Track - Disk 3 Assign Alternate Track - Data Cell 6 Initialize Tape F Fast Copy V VTOC Display			

Component and Subcomponent Identifiers

The first character after the partition identifier indicates the message origin with the system, such as

0 Supervisor, IPL, DOC
1 Job Control or POWER
2 Linkage Editor
3 Librarian, EREP
4 LIOCS, BTAM, QTAM, PDAIDS or ESTV
8 System Utilities
9 Access Method Services
A Assembler
E OLTEP, Emulators

The messages pertaining to most of these components are subdivided again into smaller groups under the next digit of the message code.

Message Numbers

The next two characters are the message number and have no other meaning.

Action Indicators

The action indicator (I, A, D, W, or E) following the message number specifies the type of action required.

Action Indicator	Meaning
A-Action:	The operator must perform a specific manual action before continuing; for example, mounting a magnetic tape, or readying an I/O device.
D-Decision:	The operator must make a choice between different courses of action.
I-Information:	The message does not require communication with the system. For example, this type of message can be used to indicate the successful termination of a problem program.
E-Eventual Action:	The operator need not do anything immediately, but will have to eventually.
W-System Wait:	Operator action is required because a hardware break down has occurred. It may be necessary to set hardware switches and/or run error recovery programs before restarting the system via IPL.

When operator action or decision is necessary, the program responsible for issuing the message usually waits until the operator enters an acceptable reply from the keyboard, or performs an action such as mounting a disk pack, readying a device, or placing cards in the card reader.

The information given in this book for each message defines in detail what decision should be made or what action should be taken in a specific case. If the reply is not one of the prescribed alternatives, the message number will be repeated but this time with the text: INVALID RESPONSE.

Example: BG 1C10A PLEASE ASSIGN SYSRDR

The characters BG indicate that this message was issued for a program executing in background.

The character 1 indicates that Job Control or POWER issued the message.

The character C indicates that either job initiation or termination issued the message.

10 is the message number.

The character A indicates that operator action is required. The operator would respond by typing the assignment for SYSRDR on the keyboard.

PLEASE ASSIGN SYSRDR is the text of the message.

When You Get a Message

Unless you know the response to a message very well, you should always look up the explanation in full, for there may be circumstances accompanying it in this case you have not met before and which need a different treatment.

If the explanation itself does not seem complete, look at the beginning of the (sub) component group to which the message belongs. Some groups of messages follow specific rules according to the peculiarities of the component they deal with. For example the messages of component 1 can have a variable digit n in the fourth character position which indicates the error field in the job control card. On the other hand, the n as a subcomponent identifier of component 4 shows what type of disk file is being processed. Likewise some messages under OPxx have additional information attached to them, which is explained at the beginning of the section covering subcomponent P of component 0.

For any problem determination procedures, if a message keeps appearing, see DOS/VS SADP, GC33-5380. For instance ROD command output is explained in Section 2,F,7; EREP output in Section 2,F,5+7-8; Dumps in Section 2,A.

When You Get a Wait State

Whenever an unexpected wait state occurs, check bytes 0 through 4 of real storage for a message. These messages are listed in DOS/VS SADP (GC33-5380), and not in this manual.

Example

0P70I [JOB or SUB] xxxxxxxx CANCELED DUE TO UNDEFINED LOGICAL UNIT

Cause: Program issued an I/O request for a logical unit for which there is no Logical Unit Block (LUB), or
A CCB or DTF table was not initialized with the proper logical unit, or
A null ISAM file has been accessed. See DOS/VS Supervisor and I/O Macros, (GC33-5373), for additional information on this subject.

System Action: The job or subtask is canceled.

Programmer Action: Obtain a system dump and the supervisor listing. Locate the IOTAB macro in the supervisor listing and determine the values of BGPGR, F1PGR to F4PGR. Locate the CCB in the dump (pointed to by register 1), and check the value of byte 6. If byte 6 is X'00', byte 7 should be less than X'0C'. If byte 6 is X'01', byte 7 should be less than the PGR value for the failing partition. If more logical units are needed, reassemble the supervisor with larger PGR values, or
Check to see that the CCB or DTF table is initialized with the proper logical unit and not overlaid during execution.

If the problem recurs, have

- job stream
- program listing
- stand-alone dump
- supervisor listing

available for problem determination.

Operator Action: To obtain a meaningful storage dump, use DOS/VS SADP (GC33-5380) Section 2,A.

1 This is the message number.

2 This is printed to indicate Job or Sub Task.

3 This will be the name of the Job or Sub Task.

4 This is a brief message to indicate the reason the job has been canceled.

These system messages will be printed out on both SYSLOG and SYSLST. Refer to Figure 6.5 for an example of a system message.

5 This is the message that was printed on SYSLST.

6 This is the message for the entire job that was printed on SYSLOG.

5A This is the message number (0S03I).

5B The message tells us that a program check interruption has occurred.

5C This is the hex location of the instruction that failed.

5D Contains the condition code from the PSW.

5E The interruption code.

Using this information, you can determine what caused the error by examining the instruction that caused the failure.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 6-3)

1. Message 1C33A would be issued by which component and subcomponent?

   a. Supervisor/PIOCS
   b. Supervisor/Checkpoint
   c. Job Control/Attention Routine
   d. Job Control/Job Initiation and Termination
   e. OLTEP

2. (True/False) If message 1C33A were issued, an operator response would be required before processing continued.

SESSION 7 - PHYSICAL INPUT OUTPUT CONTROL SYSTEM (PIOCS) MACROS

SECTION 7-1 - PHYSICAL INPUT OUTPUT CONTROL SYSTEMS (PIOCS) CONCEPTS

The Supervisor is a control program that operates with problem programs. Part of the Supervisor resides in main storage at all times. Certain other supervisory routines are kept in the core image library and are called into transient areas when required. The main functions performed by the supervisor are:

- Handles I/O requests
- Provides Error Recovery Procedures
- Operator Communications
- Interrupt Processing
- Load Problem Programs

The area that is now going to be discussed is the Supervisor's ability to handle I/O requests.

Prior to explaining exactly how this facility is provided, an understanding of how a supervisor interfaces with the problem program, channel, and I/O devices is required.

Supervisor/Problem Program Relationship

The Supervisor, as stated previously, is used as an interface between one or more resident programs and System/370 hardware. Basically, the DOS Supervisor can respond only to interrupts. The Supervisor can gain control of the system via an interrupt and can return control to a problem program via a LPSW instruction. Refer to Figure 7.1 and the following text for an explanation of this concept.

1 If the problem program is executing in storage, it will maintain control of the system until an interrupt occurs. This interrupt could be initiated, for example, by the problem program (SVC) or hardware (interval timer).

2 When one of the 5 classes of interrupts occurs:

- External
- Input/Output
- Machine Check
- Program Check
- Supervisor Call

control will be transferred (by hardware) to the Supervisor. The Supervisor control program consists of many separate routines **3** .

3 Therefore, depending upon the type of interrupt that occurred, one of the routines within the Supervisor will gain control. This routine will handle the interrupt function (program check, I/O interrupts, etc), required. After the function has been completed, control must be returned to the problem program.

4 The Supervisor, in a general exit routine, will issue a load PSW instruction (usually the old PSW of the interrupt that caused the Supervisor to gain control) to return control **5** to the problem program **1** .

To summarize the problem program/Supervisor relationship, it can be stated:

- Entry to the Supervisor is via an interrupt.
- The Supervisor contains multiple routines to provide interrupt handling facilities.
- The Supervisor transfers control to the problem program via a LPSW instruction.

The supervisor performs many of the functions required by the problem programs. These functions are accomplished by routines within the Supervisor. Refer to Figure 7.2. Shown here are some of the common routines in most DOS Supervisors. The Supervisor, of course, operates in Supervisor mode, and therefore, can issue privileged instructions (Start I/O,LPSW, Test I/O, etc.) , that the problem program cannot issue.

These Supervisor services must be requested by the problem program. This concept is essential to understanding how a problem program operates in a DOS environment. To explain this concept, you must understand the Supervisor Call (SVC) instruction.

Basically, this instruction, when issued by a problem program, creates a Supervisor call interrupt. The Supervisor call interrupt old PSW (bytes 2 and 3), will contain the interruption code of the SVC. For example, if the problem program issues a SVC 1, the SVC old PSW will contain a 0001 in the interrupt code.

SVC old PSW = 0010 0001 7000CEF6
interruption code

Another example would be if a Supervisor call of 14 (X'0E') was issued, the SVC old PSW would be 0010 000E 7000CEF6
interruption code

The supervisor, by analyzing the SVC interruption code, can determine the supervisory function requested.

Figure 7.3, in conjunction with the following text, will explain how supervisory functions (routines) are requested by the problem programmer.

1. To request a supervisory function, the problem program issues a SVC instruction. This causes an SVC interrupt and control goes to the SVC interrupt handler 2 in the DOS Supervisor.

2. The SVC handler routine in the Supervisor will now analyze the SVC old PSW 3 that was stored in a fixed location of storage by the CPU hardware.

3. The SVC old PSW will contain an interruption code that corresponds to the Supervisor call issued in the problem program 1.

4. Depending upon the interruption code, the SVC interrupt handler will branch to the associated routines in the Supervisor.

5 For example, if the SVC interruption code 4 was 00, the channel scheduler routine would be branched to. The channel scheduler handles I/O requests as a result of a SVC 0 issued by the problem program. It is a request for the Supervisor to perform problem program I/O.

6 If the interruption code 4 was X'0E' (14), the SVC handler would branch to the End-of-Job Handler routine 6. Whatever the case, when the Supervisor has completed the requested function, it will go the general exit routine 7 where a LPSW instruction will be issued to transfer control 8 back to the problem program.

Let's quickly review the hardware interface required for an I/O operation. This interface consists of the following:

- CAW - channel address word
- CSW - channel status word
- SIO instruction - Start I/O instruction
- I/O interrupt facilities - (I/O PSW)
- CCW chain

Figure 7.4, in conjunction with the following text, will review the Supervisor/hardware interface.

1 The problem program issues a request for an I/O operation (SVC), and control is eventually given to the Channel Scheduler 2 in the Supervisor. Assume that the problem program has passed information regarding the device address on which the I/O is to be performed to the Supervisor, and also passed the address of the CCW(s) to be executed. We will look at this area in more depth later.

2 The routine named 'Channel Scheduler' in the Supervisor sets up the Channel Address Word (CAW) 3 by using the address of the CCW(s) passed by the problem program and sets up the SIO instruction 3 to address the device (00C in this example).

4 The SIO instruction is issued and the proper device is selected 5.

6 After the proper device is selected, the channel scheduler exits (returns control) to the problem program 1 so that it may continue processing while the I/O operation is taking place.

7 At some later point in time, a channel end interrupt will occur. Several things happen at this time, all being hardware functions:

- Device and Channel Status are stored in the Channel Status Word.
- An I/O old PSW is stored with the address of the device that caused the interrupt in bytes 2 and 3.
- The problem program is interrupted and the I/O new PSW becomes current, and control is transferred 8 to the channel scheduler interrupt handler 9.

9 The interrupt handler will use the information stored in step 7 to determine the device that caused the interrupt and check to see if any errors occurred during the operation. If no errors occurred, control will be transferred back to the problem program 1. If there were errors, the interrupt handler will schedule the device for error recovery procedures.

In Figure 7.4, item 2 is the function that performs the I/O operations for the problem program (Channel Scheduler).

The primary function of the channel scheduler (a part of the Physical Input Output Control System) is to:

- Schedule I/O requests on each channel.
- Start input/output operations
- Handles I/O interrupts - normal completion of data transfer, error detection, end-of-file detection, etc.
- Performs error detection and error correction.
- Monitor DASD channel programs for file protection and address continuity for disk system input/output.

I/O devices in the System 370 are attached to channels rather than directly to the CPU. A channel provides a path for data transfer between the CPU and the I/O device, and allows I/O operations to be overlapped with the CPU processing the I/O operations on other channels. That is, instructions can be executed simultaneously with data movement in one or more input/output channels. For instance, at a given point in time, one channel may be reading data from a Direct Access Storage Device (DASD), another channel may be writing data on a printer, while a previously read record is being processed. This is referred to as read/write/compute overlap.

The two types of channel in this system are: selector (optional feature of block multiplexor) channels and the multiplexor channel. The selector channels allow I/O operations for devices on these channels to overlap with CPU processing and I/O operations on other channels. On the multiplexor channel, tape and DASD I/O operations cannot overlap with other I/O operations on the same channel. On some System/370 Models, tape and DASD devices operating on the multiplexor channel must not overlap with processing. Card, printer and other low-speed (byte interleave mode) I/O devices on the multiplexor channel can overlap with each other, with CPU processing, and with I/O operations on other channels. Thus, greater throughput can be achieved if high-speed devices (tape and DASD) are attached to selector channels, and low-speed devices (card and printer) are attached to the multiplexor channel.

Overlapping I/O operations with CPU processing is inherent in the design of the machine and the Channel Scheduler. However, achieving maximum overlapping also partially depends on the problem program. For instance, if overlap is desired in tape or DASD operations, the problem program should provide for two I/O areas (or buffers). This allows data to be read into, or written from, one I/O area while records are being processed in the other area. Certain devices, however, have buffers built into the device (1403) and require only one I/O area in main storage to achieve overlap. The use of multiple I/O areas and separate work areas is beyond the intent of this course.

All requests for I/O operations are handled by the Channel Scheduler. When a request is received and the affected channel and device are not busy, the requested operation starts and control passes back to the problem program. If the channel or device is busy, the request is placed at the end of a list (or queue) of I/O requests, and the operation is performed as soon as all previous requests have been handled. (Separate queues are maintained for each device.)

The Channel Scheduler also handles all I/O interrupts. If the interrupt indicates the normal end of an I/O operation (channel end and no errors), the Channel Scheduler posts completion, and removes the request from the queue. It then examines the queues for the affected channel or subchannel. If the queues are empty, control returns to the problem program at the point of interrupt. If instead a request is pending, the Channel Scheduler starts the I/O operation and then returns to the problem program. Requests for devices for which device-end interrupts are outstanding cannot be serviced until the device end is received. These requests are bypassed when the Channel Scheduler is selecting an I/O operation to be started. As an example, for a 1403 Model N1, channel end is received as soon as the buffer is completely loaded (about 2ms), but device end is not received until completion of the print operation (55ms).

The Channel Scheduler detects the following specific status conditions:

1. Wrong Length Record (WLR)

2. Unit Exception

3. Channel and device errors.

Wrong-length record and unit exception are treated as normal conditions. They are posted to the user along with the other Channel Status Word information (residual count, status bytes, and CCW address). The physical IOCS user is responsible for checking and handling these conditions.

If an error is detected, the Channel Scheduler passes control to the appropriate device error recovery routine, which takes appropriate action: retry, operator intervention, notify problem program, or terminate job.

For certain devices (1052), the operator can initiate an I/O operation. To do so, he presses the request key on the device. When the Channel Scheduler detects an attention status condition, it passes control to a message processing routine.

A problem program can perform I/O operations in two ways:

1. The problem program can issue physical I/O macro instructions directly.

2. The problem program can use logical IOCS, which in turn issues the physical I/O macro instructions.

The first method of performing I/O will be covered in this course. This method uses Physical Input Output Control System macros to accomplish I/O.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 7-1)

1. The DOS Supervisor can respond (gain control) through a(n):

   a. LPSW instruction
   b. interrupts
   c. branch instruction
   d. SIO instruction

2. The Supervisor normally returns control to the problem program by loading the _____ PSW of the interrupt type that caused the interrupt.

   a. old
   b. new

3. The problem program requests Supervisor services by using a _____ instruction.

   a. LPSW
   b. branch
   c. Program Check
   d. SVC
   e. SIO

4. (True/False) The channel scheduler sets up the hardware interface (SIO instruction and CAW) to accomplish an I/O operation.

5. If an I/O request cannot be started because a device or channel is busy, the channel scheduler will:

   a. ignore the I/O request.
   b. place the I/O request in a queue of I/O requests.
   c. issue an error message to the operator.
   d. enter the wait state.
   e. issue a halt I/O instruction to the busy device or channel.

6. The programmer using PIOCS macros is responsible for the checking and handling of _____ condition(s) in his problem program: (More than one answer is required.)

   a. channel errors
   b. device errors (unit checks)
   c. unit exception
   d. wrong length record (WLR)
   e. machine checks

SECTION 7-2 - PHYSICAL INPUT OUTPUT CONTROL SYSTEM (PIOCS)

The Physical Input Output Control System (PIOCS) is a part of the supervisor. Refer to Figure 7.5 1 . The PIOCS routines in the Supervisor control the transfer of data to or from the external devices. These routines are:

- Channel Scheduler
- I/O Interrupt
- Start I/O
- Device Error

The programmer can request (use) these routines in the problem program by using PIOCS Macros. Figure 7.5 2 . There are only three macros required for a programmer to request an I/O operation. The three macros are:

- EXCP - Execute Channel Program
- WAIT - Wait for I/O completion
  CCB - Command Control Block

For now, consider a macro as an assembler instruction that generates many instructions. Later in the course it will be explained how a macro is generated and how to code a macro.

These PIOCS macros 2 cause linkage 3 to the PIOCS routines in the supervisor. Note that item 3 on Figure 7.5 shows there is communications from PIOCS macros to the PIOCS routines in the Supervisor and vice versa.

Whenever PIOCS macro instructions are used, the programmer must construct the channel command word(s) (CCW) for his input/output operations 4 . The programmer uses the assembler-instruction CCW statement to do this. The programmer must provide any of the functions that are required for a problem program, such as blocking or deblocking records, performing programmed wrong length checks, testing (the CCB) for certain device conditions, switching I/O areas when two areas are used.

The three PIOCS macros available to the programmer are as follows: (Refer to Figure 7.6)

1 EXCP (Execute Channel Program.) This macro instruction communicates directly with the Channel Scheduler to request that an I/O operation be started. When the EXCP macro instruction is used, the problem program must supply the appropriate channel program consisting of channel command words (CCWs).

2 WAIT This macro instruction suspends program operation until an I/O operation (referenced in the WAIT macro instruction) is complete. The problem program must use this macro instruction at the point where processing cannot proceed until the I/O operation is complete. For instance, a problem program may issue the EXCP macro instruction to read a DASD block. At the point where the program needs the block for processing, a WAIT macro instruction must be issued. The instructions generated from this macro test **4** a program switch (in the CCB **3**) to determine if the operation has been completed, and give control to the Supervisor if it has not been completed. The Supervisor places the program in the wait state until the operation is completed, and gives control to a ready lower priority program, if one exists. The completion of the operation causes an I/O interrupt **5** to the Channel Scheduler. The program is taken out of the wait state, the switch is set to show the completion **6** , and control returns to the problem program.

3 CCB (Command Control Block.) This macro instruction generates a command control block for a channel program to be executed. The command control block contains information required by the Channel Scheduler **7** to execute the EXCP and WAIT macro instructions. The block is used to pass information between the problem program and the Channel Scheduler, such as symbolic I/O unit address, channel program address, status of the operation, action to be taken in the event of an error, etc.

Note on Figure 7.6 that the operand of the EXCP macro and WAIT macro is the name RD. This name (RD) corresponds to the name of the CCB macro.

PIOCS Macro/Supervisor Concepts

As previously stated, the programmer codes PIOCS macros in a program and also supplies other pertinent data required for an I/O operation. With five source statements, the programmer can supply enough information for an I/O request.

Figure 7.7 shows the five basic statements required by the programmer. The following text is to be used with Figure 7.7 as each source statement is explained.

A This is the EXCP (execute channel program) macro that initiates the I/O request. The operand of the EXCP macro, READCCB, is the name on the CCB at item **C** .

B This is the WAIT macro which the programmer coded to insure that the input data requested in statement **A** , is transferred into storage before any processing is done on the data. The operand of the WAIT is the name of the CCB used for the I/O operation.

C This is the Command Control Block (CCB) coded by the programmer to specify the I/O operation required. The CCB parameters cause constants to be generated that inform the channel scheduler of what it is to do. There are two operands in this example (SYSIPT and RDCCW) of a CCB.

- SYSIPT = is the name of symbolic unit the programmer wants the I/O operation performed on.
- RDCCW = this is the name of the channel program CCW(s) **D**. the programmer wants executed on SYSIPT. This, in reality, generates an address constant that becomes CAW.

D This is the CCW the programmer wants executed for the I/O request. The name of the CCW is RDCCW and corresponds to the second operand in the CCB **C**.

E This is the I/O area the data is to be read into. Refer to the CCW at item **D**.

Use Figure 7.7 and the following text for an explanation of the PIOCS macros and the Supervisor interrelationship.

1 The EXCP is the request for an I/O operation. It causes control to be passed to the Supervisor via a supervisor call.

2 The EXCP references a CCB containing the [symbolic] device address and the CCW (channel command word) location.

3 The request (the CCB address) is stored in an I/O request table.

4 The device on which the operation is desired is checked to see if it is busy.

5 If the device is not busy, the Supervisor issues the start I/O instruction to the channel. The dotted line indicates a reference to the I/O request table for information necessary to the SIO (Start I/O). Note that if the device had been busy in step 4, this step would have been skipped.

6 Control now returns to the problem program.

7 The problem program is now executing. An I/O interrupt is initiated by the channel upon completion of the previously started I/O operation.

8 Control passes from the problem program to the I/O interrupt routine as a result of the I/O interrupt.

9 The I/O interrupt routine posts the fact of the I/O completion to the CCB.

10 Control passes back to the point of interrupt in the problem program.

11 The WAIT macro performs a test on the CCB to see if I/O completion has been posted. If it has, problem program execution continues. If it has not, the WAIT macro suspends execution of the problem program until the I/O operation is complete.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 7-2)

1. (True/False) The PIOCS macros provide the facility to block and deblock records.

2. The PIOCS macro that requests that an I/O operation be started is the _____ macro.

   a. EXCP
   b. WAIT
   c. CCB
   d. CCW
   e. I/O area

3. The WAIT macro will suspend processing normally until _____ is received.

   a. unit exception
   b. status-in
   c. channel end
   d. device end
   e. unit check

4. Information related to a I/O operation is posted in the _____ by the supervisor.

   a. EXCP
   b. WAIT
   c. CCB
   d. CCW
   e. The Supervisor does not post any information to the problem programmer indicating the status of the I/O operation.

5. The WAIT macro tests the _____ to determine if an I/O operation has been completed.

   a. CCB
   b. I/O area
   c. CCW
   d. CSW
   e. CAW

SESSION 8 - PHYSICAL INPUT OUTPUT CONTROL SYSTEM (PIOCS) MACRO USAGE

SECTION 8-1 - PHYSICAL INPUT OUTPUT CONTROL SYSTEM (PIOCS) MACRO USAGE

General Information on Macros

Prior to actually discussing how to code PIOCS macros themselves, a general knowledge of macros will be required. During preceding sessions there has been various references to macros. The relationship between the assembler and assembler macros will be covered. The rules covering coding of macros, will be covered.

Basically, a macro is an easy way to code commonly used routines without having to supply all the mnemonic instructions when writing a program in assembler language.

For example, let's suppose that throughout a program that was being coded, there was a constant need to add two data fields together and place the sum into a third data field.

An example of how this could be coded in assembler language is shown by the following instructions (Figure 8.1) Assume the fields DATA 1 and DATA 2 are to be summed and placed in an area labeled DATA 3.

The programmer, in this case, may have to code 6 statements to add these fields together and also be careful that the contents of general registers 2 and 3 are saved and not destroyed. Of course, if the registers contained no valid information, they could be used without saving the information.

Now suppose the programmer had a macro available, that would generate the instructions necessary to perform the previous function. The format could possibly be as shown in Figure 8.2.

Code below the ADD macro (use fields with the same names as in Figure 8.1) to perform an add function.

operation	operand

Answer

operation	operand
ADD	DATA1, DATA2, DATA3, WORK

or

operation	operand
ADD	DATA2, DATA1, DATA3, WORK

Note how the programmer needs to only supply the variables of:

- DATA 1
- DATA 2
- DATA 3
- WORK

The assembler, when it encounters the macro 'ADD' as a source statement, will substitute in the above values and generate the instructions shown in Figure 8.1. Refer to Figure 8.3 for an example of the generated code.

Let's do another problem. Assume that later on in the program you wanted to add fields named CAT and DOG together and have the answer placed in a field called ANIMAL. Code the ADD macro to do this. (assume the field WORK is still available as a save area)

operation	operand

Answer

operation	operand
ADD	CAT, DOG, ANIMAL, WORK

or

operation	operand
ADD	DOG, CAT, ANIMAL, WORK

If the ADD statement had been coded:

ADD CAT, DOG, ANIMAL, WORK

what would the assembler instructions look like that were generated for the above ADD macro? Fill in the blanks below.

```
STM     2, 3, _____
L       2, _____
L       3, _____
AR      2, 3
ST      2, _____
LM      2, 3, _____
```

Answer

```
STM     2, 3, WORK
L       2, CAT
L       3, DOG
AR      2, 3
ST      2, ANIMAL
LM      2, 3, WORK
```

The two major advantages of macros are:

- An easy method to code commonly used routines without having to supply all the mnemonic instructions.
- Programmer needs to supply only the variable factors and the assembler will substitute these variables into the routine (generated instructions).

Assembling a Macro

The procedure for using a macro in an assembler program is simple. Reference Figure 8.4 item **A**. This shows the typical coding of the assembler mnemonic instruction LOAD, which we are all familiar with. Item **B** shows our favorite ADD macro coded in the source program.

The coding sheet would be keypunched to produce a source deck 1 . This source deck will be input to the Assembler program 2 . When the Assembler encounters item A , it will look at the operation code (L) and determine that this is a mnemonic LOAD instruction and generate the corresponding machine instruction in the object deck 4 and print the mnemonic instruction C on the assembler listing 4 . The next instruction the assembler would encounter would be the macro ADD B . The Assembler would analyze the operation code and determine it wasn't a mnemonic operation code.

The Assembler will now go 3 to the source library (system or private) to determine if the operation code ADD in the source statement B is a macro. In this example, the macro is in the source library. The assembler will access these macro statements and substitute the operands (X, Y, Z, SAVE) from the ADD macro source statement B .

The assembler will then generate the required machine instructions in the object deck 4 and will print on the assembler listing the generated mnemonic instructions D for the ADD macro. Observe in D the Assembler has substituted the operands from the ADD source statement B . It is very easy to determine from an assembler listing the instructions generated for a macro because they are preceded by a plus (+) sign D .

Answer the following study questions and when you have answered them, input your answers via the terminal.

Study Questions (Section 8-1)

1. The advantage to the programmer in regard to using macros is: (More than one answer is required)

   a. the number of source statements the programmer must code is reduced.
   b. the macro code executes faster than if the instructions were coded in assembler language.
   c. the generated instructions for the macro use less core.
   d. the programmer need supply only variables which will be substituted in the generated instruction by the assembler, thus reducing the time to code a program.

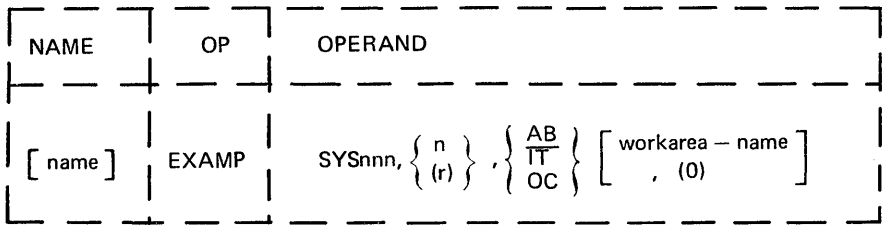
2. (True/False) The following mnemonic instruction has been generated as a result of the programmer coding a macro.

```
+       LM      0,15,WORKAREA
```

SECTION 8-2 - CODING MACROS

Just as there are rules for coding assembler mnemonic instructions, there are rules or conventions for coding macros. It is not the purpose of this course to provide in-depth coding knowledge, but, if you are to understand PIOCS, some knowledge in this area is required.

The programmer who desires to use a macro will reference an SRL for the function and coding rules for the macro. When the programmer locates the macro in the SRL, he may encounter the following:



If a programmer knows the rules of notation conventions, he can code this statement. Let's examine this statement and apply some of the notation conventions. You will be supplied with a complete list later. Study Figures 8.5, 8.6, and 8.7 to see how notation conventions can be applied.

The previous examples show how the rules (notation conventions) apply to coding a macro. Using these rules, here are some of the ways the macro EXAMP could be coded by a programmer.

Name	OP	Operand
CODES	EXAMP	SYS005, 14, IT, WORK

This example shows all fields were used.

The following example shows the minimum parameters the programmer could code.

```
EXAMP SYS005,14
        or
EXAMP SYS005,(14)
```

Even though the third parameter { AB / IT / OC } is omitted, the assembler will assume that AB was specified because it is the default.

Answer the following study questions and when you have completed them, input your answer via the terminal.

Study Questions (Section 8-2)

NOTE: You will be required to use Figures 8.8.and 8.9 during these study questions. Please study these Figures now.

1. The notation convention that indicates a parameter is required is:

   a. [ ] brackets
   b. . . . ellipsis
   c. { } braces
   d. ___ underlined

2. What is the assumed value for the following macro parameter?

$$\left\{\begin{matrix}\text{ABC}\\\text{DEF}\\\text{GHI}\\\text{JKL}\end{matrix}\right\}$$

   a. ABC
   b. DEF
   c. GHI
   d. JKL
   e. None of the above.

3. Refer to Figure 8.8, the COMPUTE macro. Select the correct way to code this macro to add data fields COW and RABBIT, place the answer in the data field PET, and use a working area named REGISTER.

	NAME	OPERATION	OPERAND
a.	ADD	COMPUTE	A, COW, RABBIT, PET, REGISTER
b.		COMPUTE	A, COW, RABBIT, REGISTER, PET
c.	ADD	COMPUTE	S, COW, RABBIT, REGISTER, PET
d.	ADD	COMPUTE	, , COW, RABBIT, REGISTER
e.	ADD	COMPUTE	A, COW, RABBIT, REGISTER, PET

4. Refer to Figure 8.8, the COMPUTE macro. Select the correct way to code this macro to subtract the data field NUTS from the data field BOLTS, have the answer placed in the field NUTS, and use a work area named REGISTER.

	NAME	OPERATION	OPERAND
a.	SUBTRACT	COMPUTE	, , BOLTS, NUTS, REGISTER
b.	SUBTRACT	COMPUTE	, , BOLTS, NUTS, REGISTER, NUTS
c.	SUBTRACT	COMPUTE	S, NUTS, BOLTS, REGISTER, BOLTS
d.	SUBTRACT	COMPUTE	S, BOLTS, NUTS, REGISTER
e.	SUBTRACT	COMPUTE	S, BOLTS, NUTS, BOLTS, REGISTER

SECTION 8-3 - REGISTER USAGE

REGISTERS FOR PROGRAMMER USE

Registers 2-12 are available for programmer usage and should be used to avoid the possibility of errors.

REGISTERS FOR SPECIAL USE

General registers 0, 1, 13, 14, and 15 have special uses, and are available to the problem program only under certain restrictions. The following paragraphs describe the general uses of registers 0, 1, 13, 14, and 15 by IOCS, but the descriptions are not meant to be all inclusive.

Registers 0 and 1

Physical IOCS macros, the Supervisor macros, and other IBM-supplied macros use these registers to pass parameters. Therefore, these registers may be used without restriction only for immediate computations, where the contents of the register are no longer needed after the computation. If the programmer uses them, he must either save their contents himself (and reload them later) or finish with them before IOCS uses them.

Register 13

Control program subroutines, including logical IOCS, use this register.

Registers 14 and 15

Logical IOCS uses these two registers for linkage. Register 14 contains the return address (to the problem program) from IOCS routines (called programs), and user's subroutines. Register 15 contains the entry point into these routines, and is used as a base register by certain macros. If the programmer uses them, he must either save their contents himself (and reload them later) or finish with them before IOCS uses them.

SECTION 8-4 - PIOCS MACROS

Physical IOCS macro instructions provide the user with the capability of obtaining data and performing nondata operations on I/O devices, with the exact CCWs that he requests. For example, if he is handling only physical records, he does not need the logical IOCS routines for blocking and deblocking logical records.

Three macro instructions are available to the programmer for direct communication with physical IOCS: CCB (command control block), EXCP (execute channel program), and WAIT. Whenever physical IOCS macro instructions are used, the programmer must construct the channel command words (CCW) for input/output operations. He uses the assembler instruction CCW statement to do this. If you do not remember how to code this statement, you can review CCW source statements in the DOS Assembler Language SRL (GC24-3414) under the topic name CCW - Define Channel Command Word.

CCB MACRO

A CCB (command control block) macro instruction must be specified in the problem program for each I/O device that is controlled by physical IOCS macro instructions. This block is necessary to communicate information to physical IOCS so that it can perform desired operations (for example, notify the problem program of printer channel 9). The command control block also receives status information after an operation and makes this available to the problem program. The user should ensure proper boundary alignment of the CCB if this is necessary for his program.

1. Blockname: The CCB instruction must be labeled (blockname) with a symbolic name. This name is used as the operand in the EXCP and WAIT macro instructions that refer to the command control block.

2. SYSnnn: Two operands are required in this CCB macro instruction. The first operand specifies the symbolic unit (SYSnnn) for the actual I/O unit with which this control block is associated. A list of symbolic units applying to CCB can be any of the system or programmer logical units that were previously covered. (SYSRDR, SYSLST, SYS000, SYS001, etc.) The actual I/O unit can be assigned to the symbolic unit by a job control ASSGN statement.

3. Command-list-name: The second operand (command-list-name) specifies the symbolic name of the first CCW used with a CCB. This name must be the same as the name specified in the assembler CCW statement that constructs the channel command word.

Refer to Figure 8.10.

4 X'nnnn': A hexadecimal value sets the CCB user option bits. Column 5 of Figure 8.15 ('On Values for Third Operand in CCB Macro') gives the value used to set a user option bit ON. If more than one bit must be set, the sum of the values is used. For example, to set user option bits 3, 5, and 6 of byte 2 ON, X'1600' is used.

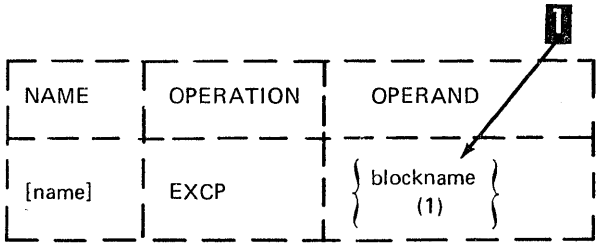
(X'1600'=X'1000' + X'0400' + X'0200')

It is possible for the macro to set ON any of the bits in bytes 2 or 3, but normally, the user need not be concerned with setting the remaining bits ON. This operand will be discussed later in this session.

5 Sense Address: This operand, when supplied, indicates user error recovery and generates a CCW for reading sense information as the last field of the CCB. The name field (sense address) of the area that the user supplies must have a length attribute assigned of at least one byte. Physical IOCS uses this length attribute in the CCW to determine the number of bytes of sense information the user desires.

Figures 8.11 and 8.12 show some simple examples of CCB's using the first 2 operands (SYSnnn and command-list-name). The last 2 operands in the CCB will be covered later.

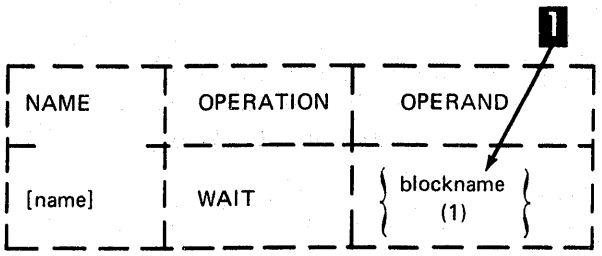
EXCP MACRO



The EXCP (execute channel program) macro instruction requests physical IOCS to start an input/output operation for a particular I/O device. The blockname of the CCB established for the device is the only operand required in this instruction. 'Blockname' can be specified as a symbol or in register notation.

Physical IOCS determines the device from the command control block (CCB) specified by 1 blockname, and places the command control block (CCB) in a queue of such CCBs for this device. If the channel and device are available, the channel program is started and program control is returned to the problem program. I/O interruptions are used to process I/O completion and to start I/O for requests if the channel or device was busy at EXCP time.

WAIT MACRO



This macro instruction is issued whenever the program requires that an I/O operation (started by an EXCP instruction) be completed before execution of the problem program continues. For example, transferring data (a physical record) to virtual storage must be completed before data can be added or moved to another area of virtual storage, or otherwise processed. When the WAIT instruction is executed in a batched job environment, processing is suspended until the traffic bit (byte 2, bit 0) of the related CCB is turned ON. Then, programming automatically continue and the data can be processed. In a multiprogramming environment, the Supervisor gives control to another program until the traffic bit is set ON.

The blockname (as a symbol or in register notation) of the CCB established for the I/O device is the only operand required in this instruction. This is the same name as that specified in the EXCP instruction for the device.

Figure 8.13 shows the EXCP and WAIT macros that would be used with the CCB example just shown to read a card. Use Figure 8.13 and the following text for an explanation.

1 The EXCP macro initiates an I/O operation via an SVC call (not shown) to the Supervisor. During this supervisor call, the Supervisor is informed where the CCB for 'CARD' 2 is located in the problem program. Note that the operand 'CARD' in the EXCP macro corresponds to the name of the CCB 3 .

3 Operand 1 of the 'CARD' CCB indicates to the Supervisor the logical unit (SYSIPT) on which the I/O is to be performed. The CCB also contains the name of the first CCW 4 (READCCW) to be used for the device.

4 Note that operand 2 of the CCW points to the I/O area into which the record is to be read. 5 An 80 byte field named IOAREA.

6 Immediately after the EXCP macro 1 is executed in the problem program, the WAIT macro 6, in our example, is executed. Note that the operand in the WAIT macro corresponds to the name of the CCB 7. The WAIT macro tests a bit in the CCB 3 called the 'traffic bit'. The traffic bit will be turned on by the Supervisor when the data from the device, has filled the area named IOAREA 5. If this operation was on a card reader, the data transfer would be completed at channel end time. When the channel end interrupt is processed,by the Supervisor, the traffic bit will be turned on. The WAIT macro 6 will generate instructions to test this bit and if it is off, it will place the CPU in the wait state. When the channel end interrupt is received, the Supervisor will turn the traffic bit on and the WAIT macro will again test this bit. If the traffic bit is on, the instructions following the WAIT macro are executed 8, which possibly will use the data just read into IOAREA.

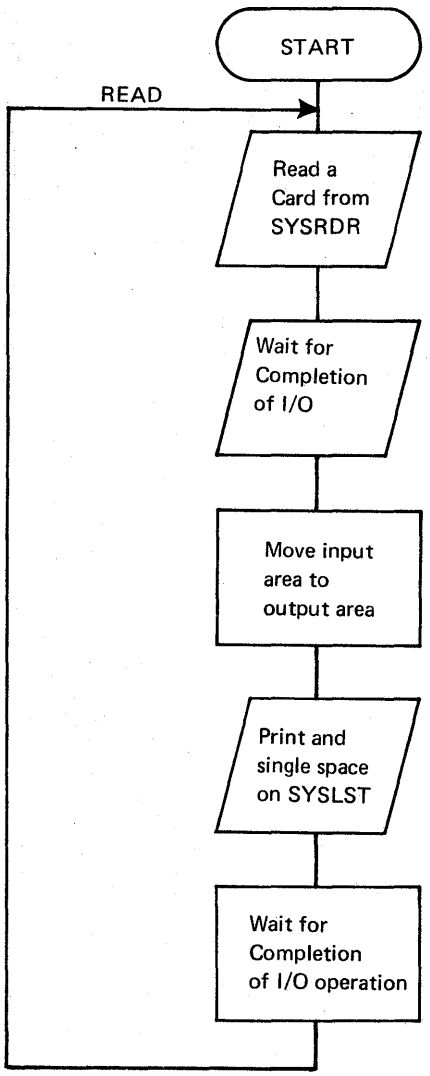
8 Imagine, if you can, that the EXCP macro was immediately followed by the processing instructions 8 and the WAIT macro was omitted. The EXEC macro would tell the Supervisor to read a card, and before the data was read into IOAREA 5, the processing instructions would try to use the data in IOAREA 5. The results would be unpredictable. The WAIT macro must be used to insure the data has been read into an input IO area before the data is used.

The same concept is also true for data in an output I/O area, except that before data is moved into an I/O area for printing, the WAIT macro must be issued to determine if the data now in the IO area has to be transferred to the device. If the data has been transferred to the device (device buffer), then new data can be moved into the IO area.

9 Note that a branch to the label READCARD will cause another I/O request to occur. Something important to remember is that the CCB 3 is not in the executed instruction stream. This is because the CCB generates a group of constants (not instructions).

Study Project

This project will provide an opportunity to use the previously mentioned macros. Following is a flowchart for a program that will read cards and print them on a printer. Observe the logic and comments.



Following is a source program to do the function shown in the flowchart. Fill in the 10 blanks left in the program.

PROJECT QUESTION

```
BEGIN    START    0
         BALR     2, 0
         USING    *,2
READER   EXCP     _______
         WAIT     _______
         MOVE     OUTPUT, INPUT   MOVE CARD RECORD TO PRINT AREA
         EXCP     _______
         WAIT     _______
         BC       15,READER
PRINT    CCB      _________,_________
CARD     CCB      _________,_________
CDCCW    CCW      2,_________,X'00',80
PTCCW    CCW      9,_________,X'20',80
OUTPUT   DS       CL80
INPUT    DS       CL80
         END      BEGIN
```

Answers to project on next page:

PROJECT ANSWERS

```
BEGIN    START   0
         BALR    2,0
         USING   *,2
READER   EXCP    CARD    (read a card)
         WAIT    CARD    (wait for card record)
         MOVE    OUTPUT,INPUT        (move input record to print area)
         EXCP    PRINT       (print a line)
         WAIT    PRINT       (wait for printer operation)
         BC      15,READER       (read another card)
PRINT    CCB     SYSLST,PTCCW        (reader CCB)
CARD     CCB     SYSRDR,CDCCW        (Printer CCB)
CDCCW    CCW     2,INPUT,X'00',80    (card CCW)
PTCCW    CCW     9,OUTPUT,X'20,80    (print CCW)
OUTPUT   DS      CL80                (Print I/O area)
INPUT    DS      CL80                (Card I/O area)
         END     BEGIN
```

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 8-4)

1. The CCB, associated with the EXCP RD macro, that will cause a record to be read from SYSRDR using a command-list-name of GO is _____.

   a. `CCB SYSRDR,GO`
   b. `CCB SYSRDR,GO,RD`
   c. `GO CCB SYSRDR,RD`
   d. `RD CCB SYSRDR,RD`
   e. None of the above.

2. When the WAIT macro is issued in a batch job environment, processing is suspended until the _____ bit is turned on.

   a. CCW
   b. CCW
   c. traffic
   d. PIOCS
   e. WAIT

3. (True/False) The CCB cannot be in the stream of executed mnemonic instructions because it contains constants (no mnemonic instructions).

SECTION 8-5 - GENERATED CODE FOR THE CCB MACRO

As shown in the previous project, PIOCS macros are easy to use when writing a problem program to do I/O operations. There are some serious drawbacks to the program in the previous project. The most obvious drawback is:

How does the program terminate?

We can see that the program is a continuous loop that will read and print cards with no facilities to handle end-of-file. The facilities are available in PIOCS for a programmer to test for any unusual device condition or error.

To understand how this is done, the constants generated for a CCB must be discussed.

A CCB macro will generate a 16 or 24 byte group of constants. Figure 8.14 shows the format of a CCB. At execution time the CCB contains information from two sources. When the programmer coded the macro, some of the fields in the CCB were generated. Remember also that the CCB is used for two way communication with the Supervisor. The information coded by the programmer is passed to the Supervisor and during execution time, the Supervisor will pass information to the problem program informing it about status of an I/O operation. When we take a close look at the information passed from the Supervisor to the problem program, it will be seen that most of the information passed is Channel Status Word (CSW) information.

Now use Figure 8.14 in conjunction with the following description as each field is examined.

	Bytes	Contents
1	0-1	After a record is transferred, PIOCS places the residual count from the CSW in these two bytes. By subtracting the residual count from the original count in the CCW, the problem program can determine the length of the record that was transferred. This residual count is set at zero for negative values.
2	2-3	The next two bytes are for transmission of information between physical IOCS and the problem program. The problem program can test any bit in bytes 2 and 3, using the mask given in the rightmost column of Figure 8.15. More than one bit can be tested by the hexadecimal sum of the test values. All bits are set to zero (OFF) when the problem program is assembled, unless the third parameter in the CCB X'nnnn' is specified. If the third parameter is specified, it is assembled into these two bytes. The user may turn on bits 5 and 7 in byte 3, and bits 3 through 7 in byte 2. During execution, each bit may be set to one (ON) by the problem program, or by a condition detected by physical IOCS. Any bits that can be turned ON by physical IOCS during program execution are reset to zero by PIOCS the next time an EXCP macro using the same CCB is executed.

3 4-5 These two bytes are the status bytes of the CSW. If device end posting is requested (byte 2, bit 5), device end status is ORed in. Byte 4 is set to X'00' at EXCP time.

Note: For nonteleprocessing devices, a program-controlled interruption (PCI) is ignored by the Channel Scheduler.

This field will be posted by the Supervisor with the channel and device status from the CSW.

4 6 Byte 6 indicates the type of CCB, ie "original", "translated", etc.

Byte 7 is the hexadecimal representation of the symbolic unit for the I/O device, as specified in the first operand of this CCB.

5 8-11 These four bytes contain the address of the CCW (or first address of a chain of CCWs) associated with this CCB and specified symbolically in the second operand of the CCB. Used by the Supervisor to set up CAW for an I/O request. (Note: Contains switches in byte 8.)

6 12 This byte must not be modified by the user.

7 13-15 These bytes contain the virtual address of the CCW pointed to by the CSW at channel-end interrupt for this I/O operation. (Note: Bytes 13-15 contain the address of the channel appendage routine when X'40' is set in byte 12.

8 16-23 These bytes are allotted only when the sense address operand is supplied in the CCB macro. They contain the CCW for returning sense information to the problem program. The CCW generated will read the sense data into the field specified in the fourth operand when an error occurs on a device.

After having observed the contents of the CCB, it is easy to see that the answer to our original question:

How does the program terminate?

This can be solved by the programmer, by testing in the CCB (Figure 8.14) byte 2-bit 1 or byte 4-bit 7 (CSW bit 39) for an end-of-file, depending on the logical unit. An example of this will be shown later.

A This legend indicates who posts each bit. 'PIOCS' indicates that the bit is reset (0) by the supervisor prior to executing an I/O request and will be turned on (1) by the Supervisor when any of the indicated conditions occur. The legend 'PR. PR.' indicates which bits the problem programmer can turn on (1) with the third operand of the CCB. These bits will not be reset by the Supervisor.

> Note: The problem programmer can turn the PIOCS bits on in the CCB when the CCB is assembled, but the Supervisor will reset them during the execution of the first EXCP macro that uses the CCB.

All the bits in this field **2** are explained in detail because most bits are self-explanatory. Read the comments and notes on Figure 8.14 and 8.15 for each bit with the label PIOCS. This information is device status, exception or error related information. The programmer controls the information returned with byte 2 (bit 3 **B**, bit 4 **C**, bit 6 **D**, and bit 7 **E**). When these bits are on, it indicates that the problem program will handle this error if the Supervisor cannot. Normally, if the Supervisor can't handle a device error, it will abort the job, but if **B**, **C**, **D**, or **E** are on, it tells the Supervisor not to cancel the job and return control to the problem program.

E This bit, when on, tells the Supervisor not to attempt any error recovery but return the control to the problem program. This bit is normally used by the programmer when a customer is using an unsupported device on DOS (DOS doesn't have error recovery procedures for the device).

These bytes can be helpful to the CE. If you are working on a hardware problem and a CCB printout is available, always check these bytes for the device that is failing.

Figure 8.15 shows the 'Condition Indicated' by the setting of each bit.

Figure 8.16 shows examples of generated code for two CCB's. Observe the generated instructions for the CCB named READ **A** at statement 25. Statement 26 is the first generated statement (+). This is only a comment statement and appears as the first statement for all macros. The information in this comment is for Program Support personnel.

Statements 27 through 35 are the generated source statements for the CCB at statement 25. The generated machine constants are to the left, from location counter value X'282E' through X'283D'. These constants correspond to the information shown in Figure 8.13.

Generated Code for the EXCP Macro

Refer to item C in Figure 8.16, an example of a EXCP macro. The actual EXCP macro is at statement 4. Statements 6 and 7 are the generated instructions for the EXCP macro. The first instruction is a load (L) instruction that loads into register 1 the address of the CCB. Observe the address constant being loaded is located at statement 51. The assembler listing ADDR2 for the load instruction E can be used to locate the adcon. Note that the ADDR2 field E contains X'28B0', which is the location counter address of the address constant. The address constant at this location contains X'282E' which is the address of the CCB named READ.

The second source instruction generated is a Supervisor Call Zero (SVC 0). This causes a SVC interrupt and control passes to the Supervisor. General Register 1 contains the address of the CCB, which is all the Supervisor requires to do the I/O request.

To the left of C are the machine instructions for the EXCP at location counter value X'2802' through X'2807'.

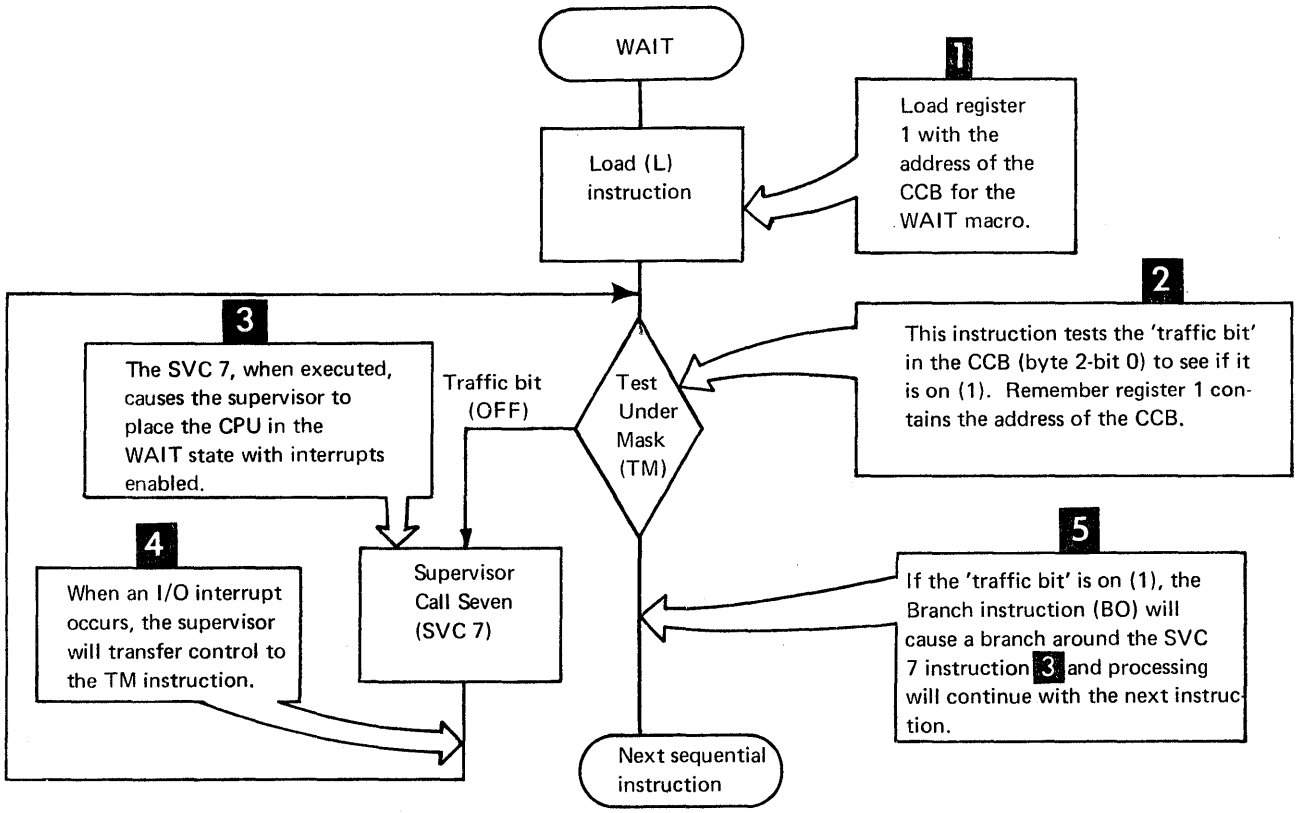
A very important troubleshooting hint that can be gained from analyzing the generated code for this EXCP macro, is:

GENERAL REGISTER 1 WILL ALWAYS POINT TO THE CCB BEING PROCESSED OR TO THE LAST CCB THAT WAS PROCESSED.

If a storage print is being used to analyze a problem the contents of general register 1 can be a very helpful aid.

Generated Code for the WAIT Macro

Refer to Figure 8.16 item **D**, the source statements generated for the WAIT macro. To the left are the generated machine instruction. The following flow chart shows the function of the WAIT macro.



Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 8-5)

1. The CCB's length is _____ bytes.

   a. 8
   b. 8 or 16
   c. 16
   d. 16 or 24
   e. 24

2. In the first byte of the transmission information in the CCB (byte 2), bits _____ will be reset by the Supervisor prior to scheduling the I/O request.

a. 0, 1, 2
b. 3, 4, 5, 6, 7
c. 0
d. 0, 2
e. None of the above.

3. Identify the source of the CCB information in the CCB fields listed below;

	CCB Fields	Source
_____ a.	Count (bytes 0 and 1)	1. CSW
_____ b.	Transmission information (bytes 2 and 3)	2. First operand in CCB
_____ c.	CSW Status Bits (bytes 4 and 5)	3. Second operand in CCB
_____ d.	Symbolic Unit Address (bytes 6 and 7)	4. Third operand in CCB
_____ e.	CCW address (bytes 9-11)	5. Fourth operand in CCB
_____ f.	CCW address in CSW (bytes 13-15)	
_____ g.	Optional Sense CCW (bytes 16-32)	

4. The following is an example of a CCB that may appear in a storage print. What symbolic unit is this CCB for?

01068000 A0000006 00002800 00002820

a. SYS006
b. SYSREC
c. SYS016
d. SYS106
e. None of the above

5. Use the CCB in question 4. The storage address of the last CCW executed is:

a. X'2800'
b. X'27F8'
c. X'2820'
d. X'2810'
e. X'2818'

SECTION 8-6 - PIOCS PROGRAM CONSIDERATIONS

This section will explain, using a sample PIOCS program, some of the common programmer tests that appear in most PIOCS programs. An awareness of these simple techniques may be of assistance if you ever have a need to write your own routine to test a device in the field.

The three areas that will be addressed are:

- Program termination
- End-of-file tests
- Accessing device end status information

Figure 8.18 will be used to address these three areas. Figure 8.17 is a flowchart of the program. Study this flowchart and then items **1**, **2**, and **3** will be discussed as related to Figure 8.18.

Program Termination

Program termination is implemented in a program with an End-of-Job Step (EOJ) macro.

EOJ--END-OF-JOB STEP

NAME	OPERATION	OPERAND
[name]	EOJ	

The EOJ macro is issued in the main task, or only program within a partition, to inform the system that the job step is finished. The operand field is ignored.

Figure 8.18, at statement 40, is an example of how this macro is coded in a problem program.

1 This is an EOJ macro. This macro generates a SVC 14. (Statement 42). When the Supervisor receives the SVC 14 interrupt, it indicates the job step (problem program) has completed. The Supervisor then loads Job Control into the partition to initialize the next job or job step. The next section will discuss how the problem programmer determines when to issue the EOJ macro.

End-of-File Test

The normal end-of-job in a problem program usually occurs when the problem program has no more work to do. This is usually a result of processing all the input data. 2 on Figure 8.17.

This is not to say that all end-of-file conditions indicate an end of job step. Usually, though, some program action must occur at end-of-file on an input device (end job step, print totals, etc.)

Our program example in Figure 8.18 will go to end-of-job. (EOJ macro 1).

Let's look at how an end-of-file condition is handled for a card reader. Refer to Figure 8.18 item 2 .

2 The end-of-file test requires only two instructions. On a 2540 card reader, unit exception will be stored in the CSW device status when the card reader runs out of cards and the EOF indicator is on. The CSW will be stored (indicating unit exception) during 'initial selection' for the following (next) SIO to the 2540. This CSW information is posted to the problem program CCB in bytes 4 and 5 (Figure 8.14). Therefore, a problem programmer can test byte 4 - bit 7 of the CCB to determine if a unit exception occurred.

Statement 24 in Figure 8.18 shows this test.

```
TM CARD + 4,X'01'
```

CARD is the name of the CCB for the card reader and the program is displacing to byte 4 (actually the fifth byte) in the CCB to determine if bit 7 (X'01') is on.

Statement 25 in Figure 8.18 is the condition test for the Test Under Mask (TM) instruction.

```
BC 1,EOJ
```

If the condition code is three (X'11'), the BC mask bit test of 1 will be satisfied and a branch will occur to the label EOJ at statement 40. The job will end because of the EOJ macro.

Accessing Device End Status Information

If the programmer has a need to test or analyze device end information (status), it requires some additional coding in the program. Basically, a CSW can be stored at one of three times.

- Initial Selection
- Channel End
- Device End

For the majority of devices (from a problem programmer's standpoint) the status stored at initial selection and channel end is sufficient for most programming decisions. Therefore, DOS doesn't normally pass this information to the problem program (CCB).

There are several output devices (not input devices) that store information at device end that a programmer requires. The facility does exist through program control to inform the Supervisor that device end status is required by the problem program.

For this example, a 1403 printer will be used to show the need for device end status. Figure 8.19 shows that unit exception is sensed on the 1403 when forms motion occurs. The programmer must test the device end status for unit exception and then take the appropriate action (skip to channel 1 on the carriage tape).

Program Requirements

The problem programmer uses the third operand in the CCB macro (X'nnnn') to inform the supervisor that device end information is required (Refer to Figure 8.10 item **4**).

With this third operand, the programmer can turn on the 'Post at Device End' bit in Figure 8.14 item **F**. Read the note associated with this bit in Figure 8.14. This bit, when on, causes the Supervisor to post the CSW status for channel and device end. The traffic bit (byte 2-bit 0 in Figure 8.14) won't be posted on until the device end status is received. Therefore, a WAIT macro for a CCB with the 'Post at Device End' bit **F** on will Wait until device end time before continuing with processing. Figure 8.15, under the column with the heading 'ON values for Third Operand in CCB Macro' shows the value that the programmer must code in the CCB to wait for device end (X'0400').

In Figure 8.18, at item **3**, there is an example of a CCB requesting device end posting. Note in the generated code for this macro at statement 46, that the 'Post at Device End' bit is on in the communications bytes.

Item **4** at statement 37 shows the programmer test used to determine if an overflow (channel 12 punch) has occurred on the printer.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 8-6)

1. (True/False) The problem program terminates normal processing by loading the Job Control program deck into storage from SYSRDR.

2. The CCB below that is coded correctly to print on SYSLST, using a CCW chain with the name of PRTCCW, wait for device end and accept unrecoverable I/O errors is:

```
a.    PRINT   CCB   SYSLST,PRTCCW,X'1004'
b.    PRINT   CCB   SYSLST,PRTCCW,X'4010'
c.    PRINT   CCB   SYSLST,PRTCCW,X'1400'
d.    PRINT   CCB   SYSLST,PRTCCW,X'4100'
e.    PRINT   CCB   SYSLST,PRTCCW,X'4400'
```

3. Assume that a printer CCB named PRINT has been coded correctly to wait for device end. The TM (Test Under Mask) instruction that could be used to determine if a channel 9 in the carriage tape of a 1403 printer has occurred is:

```
a.  TM  PRINT + 3,X'02'
b.  TM  PRINT + 3,X'01'
c.  TM  PRINT + 4,X'02'
d.  TM  PRINT + 4,X'01'
e.  TM  PRINT + 2,X'02'
```

SESSION 9 - I/O MESSAGES

SECTION 9-1 - PROBLEM DETERMINATION GENERAL

Problem determination is a phrase that has been used with increasing frequency within the last couple of years in our data processing accounts.

This session will cover problem determination actions as related to DOS programmers and operators. Problem determination for operators and programmers is defined as: a procedure or process (provided by IBM) that the user can follow after an error condition to determine the cause of that error.

With more sophisticated hardware and software systems, it is sometimes difficult to determine the cause of a system error due to the intergration of software and hardware facilities. The failures that can occur on a computing system basically fall into the following categories:

- Machine goes into the wait state.
- Program goes into non-ending loop.
- Error Message.
- Incorrect output for a job run.

These failures could be a result of the following:

- Hardware failure (CPU or I/O device)
- Programming problem (IBM or Customer program)
- Media failure (disk pack/tape reel/etc.)
- Operator Problem

IBM has provided a problem determination procedure or process that the customer can follow after an error message, incorrect output, wait state, or loop, to determine the cause of the error.

First, the problem determination procedures associated with error messages will be covered with the main emphasis on hardware errors that create wait states or messages. The second area of problem determination that will be covered will be Problem Determination Aids (PDAID) which are programs supplied by IBM to assist customers and CE's in problem determination. These programs, PDAID's, will be covered in a separate session later in this course.

Problem Determination for DOS Messages

Problem determination procedures for DOS error messages have been integrated in the DOS/VS SADP Manual. The format of DOS messages has been covered previously, but there has been no mention of the problem determination procedures that are included in the message. Several examples of selected messages will be shown with their associated problem determination procedures. These problem determination procedures can be extremely helpful to the CE when trying to localize a problem.

Figure 9.1 is the first example of a message. Study this figure and the following items within the figure:

1 Cause of error.

2 System action as a result of the error.

3 Problem determination procedure for the programmer.

4 Problem determination procedure for the operator.

The previous example of problem determination procedures was very simple. Note that item 1 implies the problem was probably a user error. If the problem persists, a Program Support Representative probably would be called.

In some of the messages, the problem determination procedure is more involved. The more complex problem determination procedures for programmer's and operator's are contained in the DOS/VS SADP manual, Section 3.

The message OP72I in the DOS/VS Message manual indicates under Operator Action to obtain a more meaningful dump to refer to DOS/VS SADP (GC33-5380), Section 2.A. Refer to this message in Appendix C to verify this message.

If the operator and programmer observe the problem determination procedures, there will be meaningful data for the IBM CE or PS Rep. when the error message results in a service call.

0P Error Messages

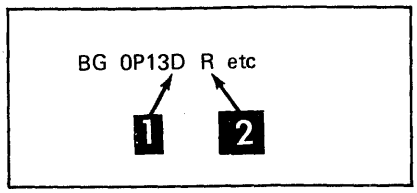
There is a group of messages 0P08-0P88 (Error Recovery Messages) that are essential to the hardware CE. These messages are issued as a result of errors (hardware or software) related to I/O devices. The following information was taken from the DOS/VS Messages manual and pertains to messages 0P08 through 0P88.

Some 0Pxx messages have after the usual message code, another one or two characters (reply code) which, together with the action code (A, I, D, W, or E), inform the operator either of what the system has done (with action code I) or what he can or must do (with action code D).

The messages 0P08-0P88 result in different combinations of action codes (items 1 and 2 in Figure 9.2), depending upon the particular device responsible for issuing the message. For this reason, no (y) or (z) entry appears for these messages in the manual. Six combinations are possible for action codes y and z (refer to Figure 9.3) if the system communication device is a printer-keyboard. If the console keyboard is inoperable, limited operations may continue under some circumstances by displaying messages in low real storage and entering the proper reply in low real storage using the store option of the CPU. The operator action for each case is determined in accordance with this action code.

In Figure 9.3, the x indicates the acceptable responses the operator can submit.

This information can be useful to the CE. For example, suppose you have an intermittent problem with an I/O device and the message prints as follows:



Since the action code 1 is D and the operator code 2 is R, it can be determined from Figure 9.3 that the operator response can be cancel 3 (Note: the command, CANCEL, is typed on the console typewriter and the job will be aborted) or the operator can type in RETRY 4 or hit the EOB/END key 5 and cause the I/O operation to be retried.

Depending on the type of problem and how critical the problem is, the response of RETRY 4 or EOB/END can be submitted to cause the operation to be retried. This would permit the CE to visually observe the failure to gather more information about the problem. (Observe error indicator, use I/O tester, etc.) The information from the 0P message could be used to determine:

- Symbolic unit and device address of the unit that failed.
- CCW operation code the failure occurred on.
- CSW information - From the CSW information, the storage address of the failing CCW can be determined. The CCW address in the CSW is the address of the failing CCW plus eight.
- Sense data
- Storage address of the CCB - The first CCW in a chain of CCW's can be determined by using the CCB address plus nine bytes (Figure 8.14).
- Disk seek address if the failing device is DASD.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 9-1)

You will be required to reference the following to answer some study questions. The figures that may be helpful are indicated below.

Reference: Figure 9.4: Error Recovery Procedures
Figure 9.5: Status and Sense Bytes
Figure 9.6: Command Codes

The following 0P message will be used for questions 1-7.

```
BG 0P18I C COMM REJCT SYSRES=190
          CCSW=1F1000204802000001 CCB=002018 SK=000000000000
          SNS=801000C80000
```

1. The message description indicates that this is probably a _____ error.

   a. hardware
   b. user
   c. disk pack

2. The operator response(s) to this message is (are) _____.

   a. CANCEL
   b. IGNORE
   c. RETRY
   d. EOB/END Key
   e. None required

3. The hardware address of the hardware device that the failure occurred on is:

   a. 190
   b. 191
   c. RES
   d. 18I
   e. 1F

4. The CCW operation code that failed was:

   a. 01
   b. 80
   c. 1F
   d. 18
   e. 90

5. The failing CCW could be located in storage at location_____.

   a. X'0020'
   b. X'0018'
   c. X'204802'
   d. X'2048'
   e. X'2040'

6. The sense data indicates: (More than one answer is required)

   a. Command Reject
   b. Seek Check
   c. Invalid Sequence
   d. File Protected
   e. Serializer Check

7. The problem that caused this error message probably is _____.

   a. invalid seek address.
   b. invalid operation code.
   c. two set file mask commands in the same command chain.
   d. less than six seek address bytes were sent.
   e. command was issued contrary to the file mask.

8. (True/False) The problem determination procedure for the programmer when he receives this message is to check the command sequence in the source program and if it is correct, he should obtain a storage print and check the CCW operation codes pointed to by the CCB.

SECTION 9-2 - PROBLEM DETERMINATION FOR LOOPS AND WAITS

When an error occurs, DOS will always attempt to print the message on the typewriter console. If the error is severe enough to impact the operating system and the possibility exists that the message cannot be typed on the console typewriter, DOS will place a message in low storage. Also, low storage messages will be placed in storage, starting at X'0000', if no console typewriter is available or SYSLOG isn't assigned (at IPL time, for example).

Low storage messages are an important part of DOS to operator communications. The problem determination procedures for DOS operator and programmers are in the DOS/VS SADP manual (GC33-5380) in the form of flowcharts.

The problem determination procedures, when followed, will facilitate problem determination by the operator and programmer.

When a programmer or operator uses problem determination procedures to identify a problem the CE must understand the problem determination procedures used. If problem determination procedures haven't been used for a problem, then the CE should be able to apply the appropriate problem determination procedures to localize the problem.

The problem determination procedures for the various error messages will be constantly referenced and used in the remainder of this course and follow-on lab.

Refer to the DOS/VS SADP manual (GC33-5380) and read all of Section 1 and pay particular attention to the following:

- Loops
- Wait States
- Indicators for Loops and Waits
- Gathering information

Refer to DOS/VS SADP manual (GC33-5380).

Read: In Section 2.E 'Wait State Messages' through but not including 'The Linkage Editor Map'.

Read: In Section 3 - Debugging Procedures for the Operator
- How to Use
- Flowcharts for System in WAIT State and Unintended Loop
  (Read any associated notes and observe the overall concept of the flowcharts.)

Read: In Section 4 - Debugging Procedures for the Programmer
- How to Use
- Flowcharts for Hard Wait State
- Flowcharts for Soft Wait State

The above reading assignments provide you with the basis upon which to approach most problem determination when servicing in a DOS/VS environment. It is hoped that your customer's operators and programmers will also use these approaches.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 9-2)

1. On an IPL the system goes into a WAIT state with the following low storage message - 07E60190. What chart in the DOS/VS SADP manual should you start at and what would be the correct action taken for this problem?

   Assume:

   a. SYSLOG is communication device.
   b. No cards in reader.
   c. Correct SYSRES mounted.
   d. No error lights.

      a. Section 4, chart 5 - check for loop
      b. Section 3, chart 01 - ignore the message
      c. Section 3, chart 01 - Re-IPL the system
      d. Section 2, table E3 - insert X'01' in byte 4 and press the Interrupt key.

2. A 'soft' wait on S/370 will respond to the _____ key on the typewriter console.

   a. EOB/END
   b. REQUEST
   c. IPL
   d. CANCEL
   e. RETRY

3. If the CPU enters a soft wait state after IPL time and there is no low core or operator messages, the problem could be a:

   a. unit check on SYSLOG.
   b. unit check on SYSRES.
   c. unit check on SYSRDR.
   d. missing interrupt.
   e. program check in supervisor.

4. The system is in a hard wait and a low core message of X'07E60130000800000C80000' is displayed. The problem is:

   a. a command reject occurred on disk drive 130 during a program fetch.
   b. a data check occurred on disk drive 130 during a program fetch.
   c. the I/O Error Queue overflowed during a program fetch to disk drive 130.
   d. channel failure on SYSLOG.
   e. IPL control statement error.

5. The system is in the wait state and a low core message of X'11C1000C' is displayed. The problem is:

   a. command reject on 00C.
   b. data check on 0C1.
   c. data check on 00C.
   d. data check on SYS012.
   e. equipment check on 00C.

6. The system is in a wait state and the low core message X'C8E2C9' is displayed. The problem is:

   a. Channel failure, reset/retry coded in ECSW invalid and error recording on SYSREC was partially successful.
   b. Channel failure, channel address invalid and error recording on SYSREC was partially successful.
   c. Channel failure, channel address invalid and error recording on SYSREC was successfully recorded.
   d. Unrecoverable machine check that was recorded on SYSREC successfully.
   e. Unrecoverable machine check and error recording on SYSREC was partially successful.

SESSION 10 - SUPERVISOR CONCEPTS

SECTION 10-1 - SUPERVISOR CONCEPTS

One of the prime functions of the Supervisor is to handle I/O requests (EXCP macros from the problem program). The previous sessions showed the problem program requirements (PIOCS macros) to issue an I/O request. This session will conceptually cover the Supervisor action taken upon receipt of an I/O request. The Supervisor is the interface between the problem program and the hardware channels. This interface is implemented in the Supervisor by a routine named the Physical Input Output Control System (channel scheduler, I/O interrupt handler, etc.) using I/O tables and control blocks.

Let's define the terms 'table' and 'control (information) blocks'.

Table - a collection of data in which each item is uniquely identified by its position relative to other items in the table or by a label. An example of this is the LUB and PUB tables (I/O tables).

Control blocks - a storage area used by an operating system to hold control information. The CCB is an example of this in a problem program. The communications region, a control block in the Supervisor, will be discussed later in this session.

Using I/O tables the supervisor keeps track of:

- Physical device type and actual channel address.
- Physical device ownership (BG, F1, ...F4).
- Logical device name and current physical device assignment.
- Logical device name and original physical device assignment.
- Specific error counts on each tape unit.

Figure 10.1 shows the various tables and blocks that will be used in this session.

1 Logical Unit Table (LUB)
- Located in the Supervisor.
- For every logical unit name (SYSxxx) used, there must be a LUB table entry.
- Contains a pointer to the PUB table.
- Size is determined by the customer at system generation time.

2 Physical Unit Table (PUB)
- Located in the Supervisor
- Contains an entry for each physical I/O device on the system.
- Information contained in an entry is the device address (CUU), device type code, mode set information for tape drives, and miscellaneous switches and indicators related to the I/O device and its current status in the operating system.
- Size is determined by the customer at system generation time.

3 Partitions Communications Region (COMREG)
- An area of the Supervisor set aside for interprogram and intraprogram communications.
- Contains information useful to both the Supervisor and the problem program.
- Used by the Supervisor to locate tables in the Supervisor (PUB, LUB, etc).
- Address of the COMREG is always in low core at bytes hex 14-17.

4 System Communication Region (SYSCOM)
- Located in Supervisor.
- Contains Partition Independent Pointers.
- Address of SYSCOM is in low storage at bytes hex 80-83.

5 Channel Queue (CHANQ)
- Located in the Supervisor.
- Used by the Supervisor to schedule and remember requests for I/O operations.
- The address of the CCB to be used for an I/O request is stored in the CHANQ.
- Size is determined by the customer at system generation time.

6 Command Control Block (CCB)
- Located in the problem program or Supervisor depending upon the component issuing an I/O request.
- The address of the CCB for an I/O request is located in general register 1, when control is given to the channel scheduler in the Supervisor.
- CCB contains information required by the supervisor to schedule an I/O request (address of CCW chain, symbolic unit, transmission information etc.)
- The CCB is either 16,or 24 bytes in size, depending on the operands specified.

Refer to the DOS/VS SADP manual (GC33-5380).

Read: Section 4, chapter 3 - LUB table
Section 4, chapter 3 - PUB table
Section 4, chapter 3 - CHANQ (Channel Queue Table)

NOTE 1: The COMREG and SYSCOM will be covered in more depth in the next session. However if you care to read about these now, you can locate them in the DOS/VS SADP manual Section 4, chapter 2.

NOTE 2: Most supervisor tables discussed in this and the next session can be found in Appendix A of the SCM. They are placed there for your quick reference.

Answer the following study questions.

Study Questions I (Section 10-1)

1. Match the following control blocks and tables with their location within DOS.

____ a.	Communication Region (COMREG)	1. Problem Program
____ b.	PUB table	2. Supervisor
____ c.	LUB table	
____ d.	CCB	
____ e.	Channel Queue (CHANQ)	

2. (True/False) The size of the LUB, PUB, and channel queue (CHANQ) tables are determined by the customer at SYSGEN time.

3. The CCB is an example of a _____.

   a. control block
   b. I/O table

4. By examination of which two tables could you determine whether a particular I/O device is waiting for an I/O operation to complete.

   a. LUB and PUB
   b. LUB and CHANQ
   c. PUB and CHANQ
   d. CCB and PUB

Answers

1. a. 2
   b. 2
   c. 2
   d. 1 and 2
   e. 2

2. True

3. a

4. c

SVC Interrupt and Return

A SVC is executed to pass control from the problem program to the Supervisor. There are many different Supervisor calls, each SVC is unique such that it requests a different service of the supervisor. This session will cover the SVC 0 that is used to request an I/O operation.

Refer to Figure 10.2 and the following text.

1. SVC 0 is issued by the problem program to request an I/O operation from the Supervisor. This causes a PSW swap (performed by microprogramming) where the current PSW is replaced by the SVC new PSW and the current PSW is stored in the SVC old PSW.

2. The Supervisor 'general entry' routine saves the SVC old PSW and the contents of the general registers.

3. The Supervisor examines the SVC old PSW to determine the Supervisor call type.

4. If the interrupt code in the SVC old PSW is 0, it indicates to the Supervisor that the problem program is requesting a Start I/O (SIO) operation.

5. The Supervisor transfers control to the channel scheduler routine which will perform the SIO operation.

6. After the I/O request is scheduled, the Supervisor's general exit routine will restore the general registers with the data previously saved. Then the LPSW instruction is issued, using the SVC old PSW and control is returned to the problem program at the instruction following the SVC instruction.

Supervisor I/O Request Handling (Linkage) Figure 10.3

The Supervisor operation in regard to an I/O request, will be shown using graphic flowcharts. The first sequence will start with the I/O request (EXCP macro) in the problem program. Use the text and associated figure to study this operation.

Reference Figure 10.3 Supervisor I/O Request Handling (Linkage).

1. The problem programmer codes an EXCP macro to request an I/O operation from the Supervisor.

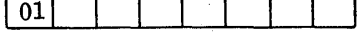
2. The assembler (at assembly time) generated the machine instructions (+) to load register 1 with the address of the CCB 3 associated with this request, and also a SVC 0, which will cause a SVC interrupt and control to be transferred to the Supervisor.

3 Prior to the explanation of what happens in the Supervisor, review the contents of the CCB at item 3. Note the symbolic unit for the I/O request is SYS004 and the name of the CCW chain is RDCCW. Observe the other CCB fields.

The CCB is a two way communications block used by both the problem program and Supervisor. If you need a review of the CCB fields, refer to Figure A.1 in Appendix A of the SCM or review Section 8.5.

4 Control is transferred to the Supervisor via a SVC interrupt. The Supervisor, upon determining that the interrupt code in the SVC old PSW is zero, will go to the channel scheduler routine.

5 One of the first actions of the channel scheduler is to store this request in a table called the channel queue (CHANQ).

The Supervisor locates the first available entry in the CHANQ and stores the contents of register 1 in this entry. Register 1 contains the CCB address associated with this request. The entry shown in Figure 10.3 is after steps 5 and 6 have been completed. The CHANQ entry being discussed was 

upon entry to the Supervisor.

6 The first byte of the channel queue is a chain pointer, which is used to chain requests if there are several requests for the same device. This is the first and only request, so the Supervisor makes this chain pointer equal to FF, designating this as the last request for an I/O operation for this device.

Supervisor I/O Request Handling (Symbolic Unit) Figure 10.4

1 After the Supervisor has stored the I/O request (CCB address) in the CHANQ the next action is to schedule the I/O request on the hardware channel. The supervisor must now determine what device the problem program wants to start.

2 The Supervisor will now analyze the contents of bytes 6 and 7 in the CCB to determine the symbolic unit.

Bytes 6 and 7 in the CCB contain X'0104' the generated code for SYS004.

3 By analyzing this constant (X'0104'), the Supervisor can determine the symbolic unit. To do this, the Supervisor analyzes the first byte X'01' and determines the request is for a programmer's logical unit (SYS000 - SYSmax). Then, the Supervisor will locate the unit in the class by analyzing the second byte X'04', This byte indicates that the request is for the symbolic unit SYS004.

4 Note that the communications region 4 contains the LUB table address. Refer to LUB TABLE (Appendix A: Figure A.2).

A In Appendix A: Figure A.2, there is a complete layout of the LUB table for a multiprogramming system. Note the layout of the LUB's for the BG, F4,....F1 partitions.

B In previous sessions, there have been references to symbolic units that have not appeared in Figures of the LUB tables used for instructional purposes. Item B shows all the symbolic units and their relative location in the LUB table.

C This item shows the various values that can be in byte 0 of the LUB table. Byte 1 of the LUB table will not be referenced in this course.

Supervisor I/O Request Handling (Physical Device Selection) Figure 10.5

1 Having located the SYS004 entry in the LUB table, the Supervisor now analyzes the first byte in the entry (PUB pointer) to determine the physical device. Because the PUB pointer is X'05', the Supervisor can determine that the physical device the I/O request is for is the sixth entry in the PUB table.

2 The sixth entry of the PUB table bytes 0 and 1 contain the address of the physical device for this I/O request or 180. In our example, the device address for 180 is that of a tape drive.

3 The Supervisor now knows what physical device address (180) the I/O request is for.

Supervisor I/O Request Handling (PUB to CHANQ Tie-in) Figure 10.6

Now that the Supervisor has determined the physical device the I/O request is for, the request (CCB address) must be tied directly to the actual physical device (Pub table entry for this device).

1 The CCB address associated with this I/O request was previously stored in the CHANQ at entry 0. The CHANQ entry number X'00' is now placed in byte 2 of the PUB entry for device 180.

The question you may be asking yourself is why must the PUB be tied to the CHANQ (CCB). There are three primary reasons for this.

- Suppose the device cannot be started now because the channel was busy handling a previous I/O request. It is not desirable to wait in the Supervisor for the channel to become available because this would waste time that could be used by the other partitions in a multiprogramming environment. When the previous requested I/O operation has completed, the Supervisor can schedule this device and can locate the CCB it requires by using byte 2 (channel queue pointer) of the PUB entry.

- The Supervisor needs to know where the CCB is in the problem program so that it can post the CSW information and turn on the traffic bit indicating the I/O request is complete.

- Used by the Supervisor to retry an I/O operation after device error recovery.

2 Byte 2 of a PUB entry will contain one of two values. If the value is X'FF' (null) it indicates there are no I/O requests for this device in the CHANQ.

If the value is other than 'FF', it is a pointer to an entry in the CHANQ and indicates there is an I/O operation pending or in process.

The Supervisor, by looking at this byte, locates the CCB associated with an I/O request and can determine if there are more I/O requests to be scheduled for a device. There will be more information on this byte later in the course.

Supervisor I/O Request Handling (CAW) Figure 10.7

1 The next action of the Supervisor is to set up the CAW. The Supervisor goes to the CCB (bytes 9-11) to determine the address of the CCW chain to be used for the I/O request.

2 The Supervisor moves the address from the CCB to the CAW. Note that the CCW will read 80 bytes into INAREA.

3 The Supervisor must now set up the storage protect key in the CAW. To do this it determines the partition the I/O request came from and moves in the corresponding storage protect key.

- Supervisor = X'00'
- Background = X'10'
- Foreground 4 = X'20'
- Foreground 3 = X'30'
- Foreground 2 = X'40'
- Foreground 1 = X'50'

Supervisor I/O Handling (Device Availability Test) Figure 10.8

1 The next action of the Supervisor is to determine if the I/O device is available for an I/O operation. If the device is not available, as we will see later, the Supervisor will not issue a SIO to save time.

2 The Supervisor can easily determine if a device is available for a SIO by testing byte 6-bit 0 (busy bit) in the PUB table.

3 If this bit is not on (0), it indicates that the device is not busy and is available to do an I/O operation. This bit is turned on (1) whenever a successful I/O operation is started on a device and will be turned off (0) by the supervisor when ending status is received (device end).

For this reason, it is important that a device end is signaled by an I/O device. If there is an outstanding device end status, device end gets lost, or the device fails to send device end to the CPU this bit will not be reset (0) and the I/O device will never be scheduled for an I/O operation.

When device end is missing, the system ends up in a 'soft' wait state, waiting for the device end interrupt. The problem of missing device end interrupts from devices has occurred occasionally in the field.

In our example, the busy bit is off (0) so the I/O device is available for an I/O operation.

Supervisor I/O Request Handling (SIO) Figure 10.9

1 Since the device is not busy, the Supervisor can issue a SIO instruction for the device 180.

2 If the SIO instruction was successful, the Supervisor will turn on (1) the busy bit in the PUB table (byte 6-bit 0).

3 The Supervisor will now restore the problem programs general registers and load the SVC old PSW and return control to the instruction following the SVC 0.

The problem program can continue processing while the tape drive is transferring data into the I/O area (INAREA).

Supervisor I/O Request Handling (Device Busy) Figure 10.10

Let's back up a bit and consider what would have happened if the Supervisor could not have started the device because the device was servicing a prior I/O request.

1 The Supervisor tests the busy bit in the PUB (byte 6-bit 0) and finds it on (1).

2 The Supervisor will not issue a SIO to the device now. When the Supervisor handles the interrupt, it will remember this I/O request and start this I/O operation. (This condition will be covered during the discussion on device end status).

3 The Supervisor will restore the general registers and load the SVC old PSW and return control to the problem program.

Answers follow the study questions.

Study Questions II (Section 10-1)

1. The Supervisor stores _____ in the CHANQ for an I/O request.

   a. CCB contents
   b. address of the CCB
   c. CCW contents
   d. CCW address
   e. address of EXCP macro

2. If bytes 6 and 7 in a CCB contain X'0006', the CCB is referencing symbolic unit _____.

   a. 190
   b. 130
   c. SYS006
   d. SYSLNK
   e. SYSRES

3. If byte 0 of a LUB entry contains X'FE', the symbolic unit is:

   a. unassigned
   b. assigned to ignore
   c. contained in the 254 entry in the PUB table
   d. non of the above

4. Byte(s) _____ in a PUB table entry contain the channel and device address.

   a. 0 and 2
   b. 1
   c. 0 and 1
   d. 2
   e. 2 and 3

5. Byte 2 of the PUB table contains a pointer to the _____.

   a. I/O device
   b. LUB table
   c. CCB
   d. CHANQ
   e. COMREG

6. The Supervisor operates with a storage protect key of _____.

   a. 0
   b. 1
   c. 2
   d. 3
   e. 4

7. The Supervisor builds the CAW (CCW address portion) from bytes_____of the CCB.

   a. 0-1
   b. 6-7
   c. 9-11
   d. 13-15
   e. none of the above.

8. The Supervisor determines a device's availability to handle an I/O request by checking _____ before issuing a SIO, instruction to the device.

   a. byte 2-bit 0 of the CCB
   b. byte 2-bit 5 of the CCB
   c. byte 4-bit 5 of the CCB
   d. byte 6-bit 0 of the CCB
   e. byte 6-bit 0 of the PUB entry

9. (True/False) If a device is busy and a SIO instruction can't be issued, the programmer must re-issue the EXCP macro again, when the device is available.

Answers

1. b - a CHANQ entry is eight bytes in length and the Supervisor stores the CCB address from register 1. Review Figure 10.3.

2. e - Review in Appendix A - Figure A.1 and A.2 item B.

3. b - Refer to Figure A.2 item C.

4. c - Review Figure 10.5 item 2.

5. d - This byte contains a pointer to the entry in the channel queue that contains the CCB address associated with the I/O request. Review Figure 10.6 item 1.

6. a - Review the text description of item 3 on Figure 10.7.

7. c - Review Figure 10.7 item 2 and Appendix A: Figure A.1.

8. e - Review Figure 10.8.

9. False - Review Figure 10.10. The Supervisor will automatically start the device when device end interrupt for the device is handled and there is a pending I/O request (byte 2 is not 'FF').

Supervisor I/O Request Handling (WAIT Logic) Figure 10.11

There are times that the problem program may not want to continue with the program until the information requested from an I/O device is in core. To accomplish this, the problem program issues a WAIT macro.

1 When the programmer decided that the data in the I/O area was to be processed, he coded a WAIT macro prior to this processing to insure the data was in storage.

2 At assemble time, the assembler generated the instruction to load the address of the CCB into general register 1. Instructions were also generated to test the traffic bit in the CCB to see if the data transfer portion of the I/O operation is complete.

3 The test is made in the CCB (byte 2-bit 0) to determine if the data transfer is complete. If the traffic bit is off (0), it indicates the data transfer is incomplete. If the traffic bit is on (1), it indicates the data transfer is complete and the problem program can continue processing. In our example, the traffic bit is not on (0), so the program will take the no leg out of the decision block at item 2.

4 If the traffic bit isn't on, the problem program will execute a SVC 7 instruction. This will cause an SVC interrupt and control will be transferred to the Supervisor.

5 The Supervisor will analyze the SVC interrupt code in the SVC old PSW and determine that a WAIT macro has been issued (SVC old PSW interrupt code=07) and branch to the wait routine in the Supervisor. This routine will load a PSW with all interrupts enabled to wait for interrupts.

(NOTE: in a multiprogramming system, control would have been transferred to any partition ready to run.)

6 When an I/O interrupt occurs and has been processed by the Supervisor, control will be transferred back to the problem program so that the test to determine if the traffic bit is on 2 can be made again.

Supervisor I/O Request Handling (Channel End Interrupt) Figure 10.12

1 When the data transfer has been completed, the I/O device generates a channel end interrupt. (For instructional purposes, we will assume that channel end and device end do not occur at the same time for the tape device we started) The I/O interrupt will cause the I/O new PSW to become current and the I/O old PSW will have the address of the device that caused the I/O interrupt in the interrupt code.

(I/O old PSW = FF150180 80nnnnnn)

Address of I/O device (cuu)

The I/O new PSW will transfer control to the routine in the supervisor that will process the interrupt. This routine will save the general registers and I/O old PSW. The next action is to determine the device that caused the interrupt and this is done by interrogating the interrupt code in the I/O old PSW.

2 The next action is to locate the PUB entry for this device in the PUB table. This is done by comparing the address of the device address stored in the I/O old PSW with byte 0 and 1 of the PUB table.

3 Now that the PUB entry for this device has been found, the CCB associated with this I/O operation can be located by using byte 2 of the PUB entry.

4 The channel queue pointer points to where in the CHANQ the address of the CCB associated with this I/O operation is located. The channel queue pointer is X'00' which points to the first entry in the CHANQ. This entry in the CHANQ contains the address of the CCB named TAPE.

5 After having located the CCB in the problem program, the Supervisor will post the following CSW information to the CCB:

- Count
- CSW status (device and channel)
- CCW address from the CSW

Supervisor I/O Request Handling (Dequeue I/O Request) Figure 10.13

1 The I/O operation is complete as far as the problem program and Supervisor are concerned when channel end status is received with no errors indicated. Therefore, the CCB address in the channel queue is no longer needed.

The hardware operation in this example will not be completed until device end.

NOTE: If the user had indicated that device end posting was required, the channel queue entry would not be dequeued until device end.

2 The I/O request is dequeued by moving the chain pointer (X'FF') from the CHANQ to the channel queue pointer in the PUB entry. The channel queue pointer in the PUB now being X'FF' indicates there are no more I/O requests for this device.

Supervisor I/O Request Handling (Completion of Interrupt Processing) Figure 10.14

1 The Supervisor must do some clean up operations with the channel queue. For example, the chain pointer in the CHANQ is changed from the residual X'FF' back to X'01'.

2 The next action the Supervisor takes is to post the traffic bit (byte 2-bit 0) in the CCB that indicates the data transfer portion of the I/O operation is complete (channel end).

3 The Supervisor will then restore the general register, and load a PSW to transfer control back to the problem program at either item **4** or **5** .

4 If the problem program had been processing instructions, control will be transferred back to the instruction following the instruction the interrupt occurred at.

5 If the WAIT macro (SVC 7) had been issued, control will be returned to the test for the traffic bit.

6 In both cases: where the WAIT macro was issued **5** and the case where processing had not reached the macro **4** , the test of the traffic bit will find it on (1) indicating the data transfer portion of the I/O operation has been completed.

7 Processing will now continue with the next sequential instruction following the WAIT macro.

Supervisor I/O Request Handling (Device End Interrupt 1 of 2) Figure 10.15

1 The device will generate a device end after channel end. The device end will cause an I/O interrupt. The Supervisor will gain control and save the general registers. The Supervisor obtains the address of the device that caused the I/O interrupt from the I/O old PSW (180).

2 The Supervisor then will locate the PUB entry for this device (180) in the PUB table.

3 Since this is a device end interrupt, the Supervisor will reset the busy bit (byte 6-bit 0) in PUB table entry for this device indicating the device is available for another I/O request.

Supervisor I/O Request Handling (Device End Interrupt 2 of 2) Figure 10.16

1 The Supervisor then checks the channel queue pointer (byte 2) of each PUB entry to see if there are any more I/O requests for the channel. If this byte is not X'FF' and the device busy bit is off (0) the I/O device would be scheduled for an I/O operation starting at Figure 10.4 item **1**. This condition could have occurred if an I/O request was received while the device was busy. If the channel queue pointer (byte 2) in the PUB is all X'FF', the Supervisor will return control to the problem program.

Answers follow the study questions.

Study Questions III (Section 10-1)

1. The Supervisor locates the CCB associated with an I/O request at channel end by using the _____.

   a. CHANQ
   b. LUB table
   c. COMREG
   d. CAW
   e. CSW

2. Match the item and condition with the corresponding time the Action occurs for the following CCB. (Assume there are no pending I/O requests for the device).

   TAPE CCB SYS004,RDCCW

   ____ a. traffic bit turned on (1) in CCB
   ____ b. device busy bit turned on (1) in PUB
   ____ c. device busy bit reset (0) in PUB
   ____ d. Channel Queue Pointer reset to X'FF' in PUB
   ____ e. CSW information posted to CCB

   1. device end
   2. channel end
   3. successful SIO instruction to device

3. After the Supervisor handles a device end interrupt, it checks byte _____ in the PUB table to determine if there is a queued (pending) I/O request.

   a. 0
   b. 1
   c. 2
   d. 3
   e. 6

Answers

1. a - Figure 10.13 item 2 shows the dequeuing of an I/O request from the CHANQ at channel end time.

2. a. 2
   b. 3
   c. 1
   d. 2
   e. 2

3. c - Figure 10.16 shows this concept.

Pub Table Contents

During this topic we have discussed some of the contents in the PUB table. There is other information in the PUB table that can be useful to the CE.

Refer to Appendix A: Figure A.3 and the following description of these bytes.

0 1 2 - Have been previously discussed.

3 This byte is used as a counter for errors by device error recovery routines for non-tape devices. For tape devices, it is used as a Tape Error Block (TEB) pointer.

4 This byte can be helpful because it indicates the device type the PUB entry is for. The device type is specified at SYSGEN or at IPL time with ADD cards. Refer to Figure A.5 item A. This hexadecimal one byte code is used to represent the device types in column B.

5 This byte contains the standard MODE set for a tape drive. This value is entered at SYSGEN or when the IPL ADD statement is used to add a tape device to the PUB table.

If the customer specified at SYSGEN time that 7 track tape drive support was desired in the Supervisor, then the DOS Supervisor will use this byte to issue a mode set command automatically to the tape drive for the customer. When item 7 is discussed, it will be explained how this byte is affected by the ASSGN statements.

Additional uses are shown in Figure A.3.

6

Bit 0 = This is the device busy bit. This bit is turned on (1) when a device is started and will be turned off when a CSW is stored with device end indicated.

Bit 1 = This indicates a device is switchable for two channels. Device is in the PUB table only on the lowest channel. If a SIO finds the channel busy, the Supervisor will add one to the channel number and attempt to start the device on this channel.

Bit 2 = This bit is turned on when a /& card is encountered on SYSRDR or SYSIPT. There should not be any more I/O requests from the problem program for this symbolic unit after this bit is turned on (1) or the Supervisor will abort the job.

Bit 3 = This bit is turned on when an I/O error is detected for this device. The device error recovery procedures (ERP) are then scheduled by the Supervisor. The ERP routines check this bit in each PUB entry to determine which device needs recovery.

Bit 4 = Indicates that a device had a SIO but was not ready. The operator is informed of this condition.

Bit 5 = This bit is turned on when an I/O request is handled and the problem program has asked for device end posting (Appendix A: Figure A.1, byte 2-bit 5). When this bit is on, the Supervisor will not dequeue the I/O request (CCB address) from the CHANQ until device end.

Bit 6 = This bit is used by the Supervisor to schedule burst devices (disk, tape, 1442 reader, etc.) on the multiplex channel. This bit is turned on when the device is placed in the PUB table (SYSGEN or ADD statement at IPL) depending on the device type.

NOTE: DOS categorizes some unbuffered devices as burst devices.

The Supervisor, if the customer selected 'burst devices on multiplex channel' support at SYSGEN, will schedule these devices so there will be no overruns. For example, if an unbuffered device (1442 card reader) is running on the channel, the Supervisor would not schedule a burst device like a 2311 disk until channel end is received from the 1442.

Bit 7 = This bit is on when a device in the PUB table is a seven track tape drive. If the customer specified 7 track tape support for the Supervisor at SYSGEN time, the Supervisor will chain a mode set CCW to the first CCW of the I/O request. The operation code of the CCW for the mode set CCW will be byte 5 of the PUB table.

7 Bits 0-4 = This is the standard mode set for a tape device. This value is created in the PUB when the Supervisor is generated or added at IPL time. These bits are the same as bits 0-4 in byte 5 immediately after IPL. Refer to Appendix A: Figure A.4 for SS code.

Bytes 5 and 7 are controlled by ASSGN statement (X'ss') parameter after IPL.

Temporary assignments for a job will alter byte 5 in the PUB table. For example, the following underlined parameter will alter byte 5 bits 0-4 to X'68'

// ASSGN SYS005, X'181', X'68'

At end of job the mode 5 will be set back to the standard mode (byte 7, bits 0-4) in the PUB entry.

Permanent assignments will alter both byte 5 bits 0-4, and byte 7 bits 0-4 to the X'ss' mode set in the ASSGN statement.

Bits 6-7 Will not be discussed.

Answers follow the study questions.

Study Questions IV (Section 10-1)

Following is a printout of a PUB entry. Use this printout to answer all questions.

0120030150934192

1. (True/False) There is an I/O request queued to this device.

2. This PUB entry is for a:

   a. 2495 tape cartridge reader
   b. 9-track 2400 Magnetic Tape Unit
   c. 7-track 2400 Magnetic Tape Unit
   d. 2321 DASD
   e. 2703 Transmission Control

3. The mode set for this device indicates:

   a. Density=200, Parity=odd, Convert feature=on, Translate=off
   b. Density=800, Parity=odd, Convert feature=off, Translate=off
   c. Density 800, Parity=odd, Convert feature=off, Translate=off
   d. Density 800, Parity=odd, Convert feature=on, Translate=off
   e. None of the above (9 track)

4. (True/False) This is a switchable device.

Answers

1. True - Byte 2 of the PUB is not X'FF'. The request is queued in the 03 entry in the channel queue. Refer to Appendix A: Figure A.3.

2. c - Byte 4 contains X'50' indicating a 2400 tape drive and byte 6-bit 7 indicates it is a 7-track tape unit. Refer to Appendix A: Figures A.3 and A.5.

3. d - Bytes 5 and 7 - bits 0-4 contain 10010xxx hex 90. Refer to Figure A.4 in Appendix A for density data.

4. True - Refer to Appendix A: Figure A.3 byte 6-bit 1. This bit on (1) indicates a switchable device.

SECTION 10-2 - CHANNEL PROGRAM TRANSLATION

Section 10-1 was a discussion on I/O operation assuming that the DOS/VS system was operating in REAL mode rather than VIRTUAL mode. Channel program operation in VIRTUAL mode is different and will be shown in this section.

It should be pointed out, that, based on your background and knowledge, you may not understand all that will be covered. Therefore, keep in mind that the intent of this section is to try to provide you with a conversational knowledge of the subject matter. This may be helpful to you when discussing problems in an account or with a PSR.

In a previous session you read about virtual storage and specifically that virtual addresses were translated (via the DAT - Dynamic Address Translation feature).

The channel(s) do not have the DAT feature and, therefore, channel programs must translate virtual addresses by means of software.

The channel program's CCWs and CCB will be placed (copied) into copy blocks.

If you refer to the DOS/VS SADP manual (GC33-5380) Appendix G, you will find near the end of the formatted dump, these copy blocks.

Refer now to the DOS/VS SADP manual Section 4, chapter 13 - Channel Program Translation and read this section.

There are no study questions for this section.

SESSION 11 - SUPERVISOR TABLES

SECTION 11-1 - SUPERVISOR TABLES

There are many areas in the supervisor that can be extremely helpful to the hardware CE. This session will show how to locate these areas and use the contents contained within these areas. These areas are extremely useful when the system enters the wait state, CPU just stops, or the CPU goes into a microprogram loop on an I/O operation. This is not to say that these areas will always give an answer to a problem, but in many instances they can be helpful.

Many times a Programming System Representative (PS Rep) has been called in to assist on a problem and after scanning a couple of core dumps says, 'This probably is a reader problem', and lo and behold, the problem turns out to be a reader failure.

This portion of the course will show you the area of a storage print that a PS Rep analyzes to determine the status of I/O devices on the system.

This session will explain the following areas and how they can be used:

- PUB (contents discussed in previous session)
- LUB (contents discussed in previous session)
- Partition Communications Region (COMREG)
- Error Recovery Block (Error Queue)
- Program Save Areas (Partitions)
- Channel Bucket
- System Communication Region (SYSCOM)

The reason that a storage print is so important to the PS Rep is that he has no one place to go to gather error information about a program failure. The hardware CE has the CPU console error indicators that may show where the problem is located. For a PS Rep, the storage print must supply additional information beyond what the hardware CE gains from the CPU indicators for an error.

The PS Rep can use the storage print to determine the status of programs and I/O devices within the computing system. This session will show some of the areas in a storage print that can be used to determine the status of I/O devices on the system.

Partition Communications Region (COMREG)

The Communications Region (COMREG) is located in the supervisor. The COMREG is an index to the various components and tables within the supervisor. It contains many switches that are used for interprogram communications. For example, during Session 2, when the OPTION Job Control Statement was discussed, it was indicated that the LINK switch was set on in the COMREG by Job Control. The Assembler program subsequently tested this switch.

There will be from one to five COMREGs in the supervisor, depending what the customer specified at SYSGEN. If the customer did not specify multiprogramming support, there will be only one COMREG.

If the customer specifies that his supervisor must support multiprogramming, there is a COMREG generated for each partition.

COMREG Location

The COMREG is easy to locate in a storage print produced by the DOS system because the DOS dump program prints the contents of low storage. The address of the COMREG is in bytes X'14-17' in storage.

For a supervisor supporting multiprogramming, the address in bytes X'14-17' will be the address of the COMREG for the partition in control of the system. The contents of the COMREGs are basically identical.

Using a stand-alone dump presents a problem in regard to locating the COMREG address. The IPL bootstrap routine (first IPL record) of a stand-alone dump program is 24 bytes in length. This IPL record destroys the address of the COMREG in bytes X'14-17' (Decimal 20-23) in storage.

For this reason, the COMREG address must be manually displayed before the stand-alone dump program is IPLed. The address of the COMREG varies from supervisor to supervisor.

COMREG Layout

It is not the intent of this course to go through every byte in the COMREG. The items that the CE can use will be shown and certain other fields that may be of interest. The COMREG is laid out clearly in the DOS/VS SADP manual (GC33-5380). Locate the Partition Communication Region in Section 4, chapter 2. Read the introductory paragraphs.

The displacement value for items/fields in the COMREG have been provided in both hexadecimal and decimal. Note the legends to the left side indicating 'Displacement hexadecimal' and 'Displacement decimal'.

The length of each field in the COMREG is shown by the X's at the bottom of the entry. Each X represents a byte. The job name field is 8 bytes long because there are eight X's.

The decimal displacements on this figure will also serve as a 'Key' which will be used to give more detailed information about the specific field in the COMREG. The decimal displacement for the Job Name field is 24. Refer to the key of 24 on the page following the layout of the COMREG.

This key indicates that this field, Job Name, is built from the operand in a // JOB job control statement.

COMREG Contents

Several areas of the COMREG will be observed. Locate these fields and read the associated key.

Highest Storage Address of the Partition (DEC Displacement 32)

PIK - Partition Identification Key (DEC Displacement 46)

These bytes can be very helpful when trying to determine the partition in control of the CPU when the system enters a wait state. Our discussion will be for a multiprogramming system without asynchronous processing support, because this will be the most common type of supervisor the CE will encounter in the field.

Let's discuss each of the PIK values (X'nnn') that could be encountered.

The PIK is a halfword, consisting of a zero value in the high-order byte and the key value in the low-order byte. The key value is the key of the partition that was last enabled for interrupts (has control of the CPU resources and is processing).

When an interrupt occurs the value in PIK indicates to the supervisor which partition was interrupted (was processing).

Refer to the DOS/VS SADP manual Appendix B - PIK. Read through but not including SYSLOG ID.

Question: What is the PIK value for F2 if NPARTS (number of partitions) = 4? _____

Answer: PIK = X'30'.

Refer back to the layout of the COMREG in Section 4, chapter 2.

ADDRESS of PUB table in the Supervisor (decimal displacement 64).

ADDRESS of LUB table in the Supervisor (decimal displacement 76).

ADDRESS of PIB table in the Supervisor (decimal displacement 90).

NOTE: An explanation of the PIB table (Program Information Block) can be found in Section 4, chapter 7 of the DOS/VS SADP manual.

System Communication Region (SYSCOM)

Another area of the supervisor to be aware of is the SYSCOM. This table is similar to the partition COMREGs and is located immediately following the background COMREG. The address of the SYSCOM can be found at low storage address X'80-83'. The SYSCOM contains partition independent pointers and addresses.

Refer to the DOS/VS SADP manual (GC33-5380) Section 4, chapter 2 - System Communication Region (SYSCOM).

Briefly scan through the table and note the various fields.

Summary

The COMREGs and SYSCOM, located in the supervisor, contain address and pointers to various tables, information blocks, etc. Most of the items shown in the fields of the COMREGs and SYSCOM are defined and explained in the DOS/VS SADP manual (GC33-5380). For example, if you wanted to know what the JIB (Job Information Block) was, you could look in the index under Job Information Block and this would lead you to a description and layout of the JIB table and would also explain how to find it in the storage dump.

The address of the active COMREG can be found by looking at low storage address X'14-17'.

The address of the SYSCOM can be found by looking at low storage address X'80-83'.

Answer the following study questions and when you have completed them, input your answers on the terminal.

NOTE 1: Appendix B in the Student Guide contains a DOS storage dump (only a portion of the dump has been reproduced).

NOTE 2: Appendix A and B will be used for this study session and should be removed from the Student Guide to reduce the amount of page flipping required.

NOTE 3: Feel free to mark up or label the various tables and control blocks in the storage dump as you access them.

NOTE 4: The address of the start of the COMREG, as taken from low storage bytes X'14-17' is X'4A0'.

NOTE 5: Appendix A of the student supplementary course material contains the same information that can be found in the manuals referenced in this course. They are placed in the supplementary course material for quick reference.

Study Questions (Section 11-1)

1. The name of the job that was running in background partition when the Appendix B storage print was taken is:

   a. SAMPLE
   b. ASSEMBLY
   c. TA
   d. 01/11/72
   e. NO NAME

2. Use the Machine Configuration byte in the COMREG (hex displacement 34) and select the feature(s) that are supported for this CPU. (Multiple answers may be required.)

   1. Storage protect feature
   2. Decimal feature
   3. Floating point feature
   4. Physical transient overlap option
   5. Timer feature
   6. MCH/CCH
   7. Burst mode on multiplex channel support

   Select one of the following (a, b, c, d or e) when you input your answer to question 2.

   a. 1, 2, 3, 4, 5, 6, 7
   b. 1, 2, 4, 5, 6, 7
   c. 1, 2, 3, 4, 6, 7
   d. 1, 3, 4, 7
   e. 1, 4, 5, 6, 7

3. The LUB table starts at location _____ in the storage print. (Use COMREG)

   a. X'0000'
   b. X'224E'
   c. X'233C'
   d. X'2F50'
   e. X'081C'

4. The address of the PUB entry for device 00D in the PUB table is: (Use COMREG)

   a. X'04E0'
   b. X'1A83'
   c. X'3088'
   d. X'224E'
   e. X'3090'

5. If the PIK value is X'0010', the _____ program/task has control of the system (assume NPARTS=4).

   a. Supervisor
   b. Background
   c. Foreground 1
   d. Foreground 2
   e. Foreground 3

SECTION 11-2 - PARTITION SAVE AREA

It was discussed previously that each partition has an area used to save a program's PSW and registers. This is necessary for multiprogramming, because there is only one set of general registers, and when a partition loses control to another partition, the partition that gains control must use the general registers. When control is transferred back to the partition that originally lost control, it must be able to restore the registers and current PSW for the partition.

The information in the save area can be helpful because it shows the contents of general registers when the partition was interrupted (lost control of the system.) The PSW can be used to determine where the program was interrupted (lost control) and in some instances, the reason the partition lost control.

Refer to the DOS/VS SADP manual (GC33-5380) Section 4, chapter 11 - Save Areas.

Read: Partition Save Areas and Label Save Areas through but not including ABSAVE area.

Partition Save Area Contents

The contents of all partition save areas is identical for a DOS system. The only item variable about the save area is that it may vary in size between different DOS systems, depending on whether floating point is supported.

Let's examine the contents of the partition save area.

Program Name - This is the name taken from the // JOB card. If no // JOB card is used, the name in this field will be: NO NAME.

Return PSW - This is the PSW the supervisor will load (LPSW instruction) when this partition gains control of the system. Normally, this PSW is the current PSW being used when the partition lost control of the system. For this reason, the interruption code (bits 16-31) and instruction address (bits 40-63) of the PSW can be helpful when trying to determine the status of a problem program.

General Registers - Contents of general registers 9 through 8 [Reg 9, Reg 10 . . . Reg 15, Reg 0, Reg 1 . . . Reg 8] at the time the partition was interrupted.

Length of Label Area - Within a partition the system may reserve a label area between the end of the partition save area and the beginning of the problem program in the partition.

Floating Point Registers - This area is used to save the contents of the floating point registers for a partition. This area is optional and will only be used if floating point is supported by DOS. The customer specifies if floating point support is required at SYSGEN.

Partition Save Area Length

The partition save area will be one of two sizes. For systems without floating point, the partition will be decimal 88 (X'58') bytes in length.

For systems with floating point support, the partition save area will be decimal 120 (X'78') bytes in length.

The easiest way to determine if a DOS system has floating point support is to ask the customer personnel. If this isn't possible, the CE can check in the COMREG byte 52 (decimal) - bit 2. If this bit is off (0), the save area is 88 decimal bytes in length. If this bit is on (1) the save area is 120 decimal bytes in length.

Locating the Partition Save Areas

The partition save area for any partition is contained within that partition.

If you are looking at a formatted dump, the save areas are formatted and can be located near the end of the dump. The save areas can be seen in the DOS/VS SADP manual (GC33-538) Appendix G.

If you are looking at a system dump, the partition save areas can be found by scanning the dump and looking for the section as shown in the DOS/VS SADP manual Section 4, chapter 11 - Save Areas.

Answer the following study questions and when you have completed them, input your answers on the terminal.

The following figure has been extracted from a system dump and should be used to answer the questions:

```
LBLTYP  HEX LENGTH IS 0000
--BG--
03C000  D7C8C1E2 C55C5C5C 071D0000 0003C084   0A16180C 00000000 182F07F1 0003C078
03C020  D7C8C1E2 C55C5C5C 00092800 0003C078   0003C0A4 4003C07A 0000C5B8 0000C710
03C040  80000015 80000015 000927FF 0003E0C8   000083C1 CDB84F81 00000000 00000000
03C060  00000000 00000000 00000000 00000000   00000000 00000000 05205810 218E0A00
03C080  5810218E 91801002 47102014 0A075810   21920A00 58102192 91801002 47102028
03C0A0  0A070A0E 0000A000 00000006 0003C0B8   00000000 00000000 0703C0E8 40000006
03C0C0  3103C0EA 40000005 0803C0C0 00000000   0603C0EF 40000050 0603C13F 40000050
03C0E0  0603C18F 00000050 0000B400 00090100   00000000 00000000 00000000 00000000
03C100  00000000 --SAME--
03C1E0  00000000 00000300 03C1F000 00000000   0903C0EF 60000050 0903C13F 60000050
03C200  0903C18F 20000050 0003C0A4 0003C1DF   00000000 00000000 00000000 00000000
03C220  00000000 --SAME--
0927E0  00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000
```

Study Questions (Section 11-2)

1. What is the contents of GP Register 9?

   a. D7C8C1E2
   b. 071D0000
   c. 0A16180C
   d. 0003C0A4

2. The return PSW is:

   a. D7C8C1E2C55C5C5C
   b. 071D00000003C084
   c. 0003C0840A16180C
   d. C55C5C5C071D0000

3. What is the length of the label area?

   a. 0000
   b. 0008
   c. 0004

SECTION 11-3 - ERROR RECOVERY BLOCKS

Error Recovery Block is a table within the supervisor that is used to store information pertinent to an I/O device error. If the system fails (enters wait state) during the attempt to recover from an I/O device error, this information can be helpful.

Error Recovery Block Usage

Before we examine what is in an Error Queue Entry, let's look at how the Error Queue Entries are created and used by DOS.

Refer to Figure 11.1 and the following description of DOS device error recovery.

1. Assume that an I/O interrupt for a device occurred and unit check was indicated in the CSW.

2. One of the first tests made in the I/O interrupt routine is to determine if a unit check occurred on the I/O operation.

3. If a unit check has been detected, the supervisor enters all the pertinent information about the error in a 44 byte error queue entry. The supervisor also issues a sense command to the device to obtain the sense data and places the sense data in the error queue entry.

4. At this point, the supervisor tests to determine if the error is a DASD error (2311, 2314, etc).

5. If the error is a DASD error, control will be transferred to the supervisor resident error recovery procedures for disk.

A Item A on Figure 11.2 shows the resident disk routines in the supervisor. The reason the supervisor checks to determine if the unit check is for DASD is because the error recovery routines for all the other devices are in the core image library B .

If the supervisor has to handle a DASD unit check and the error recovery routines for DASD were in the core image library, the system probably couldn't load the error recovery procedures into storage to recover from the disk error. This would be a catastrophic error condition. Therefore, the disk device error recovery routines for the SYSRES device type are resident in the supervisor.

B The transients for device error recovery are resident in the SYSRES core image library and are loaded by the system loader (in the supervisor) when required. The transients for device error recovery are small routines (approximately 600 bytes in length).

C The transients are loaded into the supervisor in a specific area called the transient area.

6 Refer to Figure 11.1 again. If the unit check is for a device other than DASD, the supervisor loads an error recovery monitor into the transient area and passes control to it.

7 The error recovery monitor will interrogate the error queue entry 8 for the device that failed and locate the PUB entry for the failing device 9 . The monitor then tests byte 4 of the PUB table 10 to determine the device type that failed. The monitor will then load the appropriate error recovery procedure transient 11 .

8 The device error recovery transient will use the information in the error queue entry for the device to effect device recovery.

Error Queue Entry Contents

Refer to the DOS/VS SADP manual (GC33-5380). Read Section 4, chapter 3 - Error Recovery Block and Error Queue. Then read the following description of the error queue entry contents.

The error queue entry for a device 44 bytes is in length.

Bytes 0-7 contain the CSW that was stored as a result of the error.

Bytes 8 and 9 contain the address of the PUB entry for the device that failed.

Byte 10 is a flag byte and the meaning of the bits is shown below the byte.

Byte 11 is the message code (in hexadecimal) and corresponds to the 0P message with the same number. For example, if the value in byte 11 was X'12' the 0P12 VERIFY CHK message would be issued.

Bytes 12-15 contain the disk seek address for a DASD device.

Bytes 16-19 contain the address of the associated CCB.

Bytes 20-43 contain sense data concerning the device and failure.

Location of Error Queue Entries

As can be seen in the DOS/VS SADP, the first error queue entry (ERRQ1) is the sixteenth byte in from the start of the Error Recovery Block (ERBLOC). Therefore by adding hex 10 (dec 16) to the address of the Error Block (found in SYSCOM), the first error queue entry can be located. Then each 44 bytes (decimal) will represent one error queue entry. Remember, however, that the number of error queue entries are variable, and the only way to determine this number is to ask the system programmer who generated the supervisor as to how many error queues there are.

Summary

The information contained in the error queues is printed out on the formatted dump. In most cases the information will be printed out as a console message. If, however, the only available tool to be used is the system dump, you will have to use the above procedure to obtain the information contained in the error queue entries.

Answer the following study questions and when you have completed them, input your answers on the terminal.

NOTE 1: Appendix B in the Student Guide contains a partial DOS storage print.

NOTE 2: Appendix A and B will be used for this study session and should be removed from the Student Guide to reduce the amount of page flipping required.

NOTE 3: Feel free to mark up or label the various tables and control blocks in the storage print as they are accessed in the storage print.

NOTE 4: The address of the start of the COMREG, taken from bytes X'14-17' is X'4A0'.

Study Questions (Section 11-3)

1. The address of the error block is:

   a. X'48B8'
   b. X'48A8'
   c. X'0540'
   d. X'2E6C'

2. The first error queue entry in the error queue block is located at _____ in the storage print in Appendix B.

   a. X'48B8'
   b. X'48A8'
   c. X'2E7C'
   d. X'0550'

Following is a sample printout of an error queue entry. Use this printout for questions 3 through 6.

```
1000F920 0E400005 30D00C20 00960004 0000BDC0 08800000
00000000 00000000  00000000  00000000 00000000
```

3. The previous sample error queue printout is for the device with the address _____ in the Appendix B: storage print.

   a. 00C
   b. 130
   c. 131
   d. 01E
   e. 01F

4. The 0P message that would print as a result of the error in the sample error queue printout is:

   a. 0P12
   b. 0P0C
   c. 0P20
   d. 0P32
   e. 0P08

5. The disk address (cylinder and head) the failure occurred at is:

   a. cylinder X'C800' and head 0
   b. cylinder X'00C8' and head 0
   c. cylinder X'9600' and head 0
   d. cylinder X'0096' and head 0
   e. cylinder X'0096' and head 4

6. The first byte of sense data in the same error queue entry is:

   a. 00
   b. 08
   c. 20
   d. 80
   e. 96

SECTION 11-4 - CHANNEL SCHEDULER SIO INSTRUCTION AND CHANNEL CONTROL TABLE/CHANNEL BUCKET

This section will show how the CE can locate the SIO instruction in the supervisor channel scheduler. It will also show how to determine the last device successfully started on a channel.

There is one SIO instruction in the channel scheduler that is used for the majority of Start I/O's issued by the system.

Each DOS customer has an assembler language listing of his supervisor and is available for reference upon request.

Figure 11.3 item 1 is an example of the SIO instruction in a channel scheduler. The start I/O instruction is updated with the device address 2 for each Start I/O. For example; if an I/O operation is for the device with the address of 281, the SIO instruction would be altered to:

9C00 0281
2

The SIO statement number in this supervisor listing is 506 3. To locate this instruction, obtain the supervisor listing from the customer and locate the label SIO in the assembler cross-reference.

NOTE: The instructions in Figure 11.3 are shown logically in the correct sequence for instructional purposes.

Refer to the example of the cross-reference listing for this supervisor in Figure 11.4.

1 Symbol SIO in cross-reference.

2 Statement number symbol is defined at (506).

3 This is the location counter value (storage address) of the label SIO.

In Figure 11.3, it can be seen that SIO is defined at statement 506 3 and that the location counter (storage address) is X'54A' 4.

The information in the SIO instruction can sometimes be useful if the CE needs to know the address of the last device a SIO instruction was issued to.

In Figure 11.3 the instruction following the SIO is a branch on condition (BC) instruction that determines if the I/O operation was started correctly.

If the operation was started correctly, the next instruction executed is at statement 509.

Channel Control Table and Channel Bucket

The channel control table and especially the channel bucket (information block), both located in the supervisor, can in some instances be helpful if you need to know the device addresses of the last devices successfully started on the channels.

Refer to the DOS/VS SADP manual (GC33-5380). Read in Section 4, chapter 3 - Channel Control Table and Channel Bucket. Then read the following text which further explains this subject.

Channel Bucket

There is one 24 byte bucket for each channel (whether attached to the system or not).

Bytes 0-3 - This is the address of the first PUB assigned to the channel. This information is not directly related to the remaining fields of the bucket; and also is not of major importance to you.

Bytes 4-19 - These fields relate to each other and you should understand each of them.

Bytes 4-7 - This field is a pointer to the CCB for the device indicated in bytes 8-11.

Bytes 8-11 - This field contains the device address of the last device successfully started on the channel.

Bytes 12-15 - This field points to the PUB for the device in bytes 8-11.

Bytes 16-19 - This field points to the channel queue for the device in bytes 8-11.

Bytes 20-23 - This field points to the PIB and is of no major significance to you.

Summary

Bytes X'30-33' of the SYSCOM point to the channel buckets and the information contained therein can at times be helpful to you if it becomes necessary to determine which device was the last to be successfully started on any given channel.

Answer the following study questions and when you have completed them, input your answers on the terminal.

NOTE 1: Appendix B in the Student Guide contains a partial DOS storage print.

NOTE 2: Appendix A and B will be used for this study session and should be removed from the Student Guide to reduce the amount of page flipping required.

NOTE 3: Feel free to mark up or label the various tables and control blocks in the storage print as they are accessed in the storage print.

NOTE 4: The address of the start of the COMREG, taken from bytes X'14-17' is X'4A0'.

Study Questions (Section 11-4)

1. Match the supervisor components with the label in the cross-reference listing (Supervisor) that can be used to locate that item.

	Component		Label
____ a.	Start I/O Instruction (channel scheduler)	1.	SIOADR
		2.	SIO
		3.	CHNSIO
		4.	CHNPUB

Refer to Appendix B - Storage Print for the next two questions.

2. What is the address of the channel buckets?

   a. X'570'
   b. X'4EA0'
   c. X'BCFFF'

3. What is the device address of the last device successfully started on channel '0'?

   a. 00C
   b. 01F
   c. 00D

SESSION 12 - STANDALONE AND SYSTEM UTILITY PROGRAMS

SECTION 12-1 - SEREP PURPOSE AND OPERATING PROCEDURES

Purpose

SEREP (System Error Recording Editing Printing) is a self-loading interpretive dump program intended to be used to retrieve and edit core resident hardware logout. It is manually loaded via IPL immediately following a system stop that is due to a machine check, interface control check, channel control check, or certain other hardware failures. The primary purpose of SEREP is to provide a hardcopy listing of available data pertinent to the error. SEREP does not perform any diagnostic function except to provide edited logout data to the customer engineer.

SEREP's function then is to:

- Provide a hard copy listing of data pertinent to an error.
- Record system environment at the time of a failure.
- Provide a limited dump.

Program operation is documented for the CE in the Maintenance Diagnostic Program Manual (ID 3FE3).

Procedure

Program Loading

SEREP does not require any other program in storage while it is running. SEREP is assembled with device address 00E as the output printer. If it is desired to change the output device address, it can be accomplished in one of three ways:

1. Make device 00E 'Not Ready'. When the program is loaded, the Wait State will be entered. Reassignment can be accomplished by causing a ready (or Attention) interrupt from any Printer (or Console Typewriter).

2. Punch the desired address, (CUU), in columns 67, 68, 69 of the last TXT card, which is the next to last card in the deck. If the device is a console typewriter, a T should be punched in column 70. Any other character in column 70 will indicate printer output. Figure 12.1.

3. Reproduce the last TXT card and leave columns 67 thru 70 blank. This will cause SEREP to go to a Wait State after loading. By causing a Not Ready to Ready interrupt (or request on the console printer), the device causing the interrupt will be assigned as the output printer.

No operator action is required after loading, unless an output device address is not specified, or the device is not ready.

Waits

There are three types of Waits involved with SEREP.

Normal Wait - If an output device has not been specified, SEREP enters the WAIT state immediately after loading.

Error Wait - Any unexpected program check that occurs while SEREP is running will cause a message to be printed and the WAIT state to be entered. Retry may be attempted by a PSW RESTART.

An error occurring on the output printer during the edit will cause the Wait State to be entered.

A data transfer error during the loading of the SEREP deck will cause the Wait state to be entered.

Program Termination - When the edited logout is complete, a message is printed and the Wait state is entered.

Program Interface

SEREP interrogates byte 115 decimal (73 hex) for a run request to determine the type of error for which it has been loaded. There are 4 valid run requests.

1. 0F (Channel Error)
2. 1F (Device Error)
3. 2F (Test Channel Failure)
4. 3F (Device Not Operational)

If none of the 4 run requests are found, SEREP must determine if either a Stop After Log (SAL) or a Machine Check has occurred. This is done by checking bytes 3 and 4 of the machine check old PSW for a SAL code. This code is provided by the microprogram when the CPU is in Stop After Log mode. The codes define which channel caused the error or if it was a CPU error.

In SAL mode, no CSW, or I/O PSW is stored. SEREP uses the contents of the identified logout to construct such information for printout purposes whenever required.

If not in SAL mode, the PSW and machine check interrupt code are examined for validity. If a machine check appears to be the trouble, a machine check edit is initiated. If no machine check old PSW has been stored, or if the interrupt code is invalid, a message indicating that no error symptoms can be found by SEREP is printed and the program is terminated.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 12-1)

1. (True/False) The last TXT card in the SEREP deck is used to assign the output device.

2. Byte X'73' of storage contains a 3F. What type of failure does this indicate?

   a. device error
   b. channel error
   c. machine check
   d. device not operational
   e. test channel failure

3. If no run request is found, SEREP must determine if either a Stop After Log or Machine Check has occurred. Select the correct answer as to how SEREP does this.

   a. Checks bytes 3 and 4 of the I/O old PSW
   b. Checks bytes 3 and 4 of the Machine Check old PSW
   c. Checks bytes 4 and 5 of the I/O old PSW
   d. Checks bytes 4 and 5 of the Machine Check old PSW

SECTION 12-2 - PROBLEM DETERMINATION AND PRINTOUT

PROBLEM DETERMINATION

Refer to the DOS/VS SADP manual (GC33-538) Section 3, chart 03 (wait states).

This chart can be used as an aid to determine when SEREP should be run.

Let's take an example: The system enters a hard wait with no error message printed on the printer/kybd (SYSLOG), and only the wait light on. Low storage (bytes 0-4) contains the following:

0	1	2	3	4
X'00'	X'E2'	X'C9'	X'00'	X'00'

Start at chart 03, part 2 and the decision blocks using the above information. On chart 03, part 2, a test is made (Press Request) to determine if hard or soft wait. A decision is required: DOS low storage bytes match any codes in Table 3.1. A correct decision will lead you to chart 03, part 4 and a block that directs you to 'Execute SEREP'.

Printouts

Since the purpose of SEREP is to edit and print data relevant to several types of errors, the printouts will vary accordingly. We will use for an example a Channel Inboard logout. Refer to Figure 12.2 parts 1 through 4.

Part 1 - This page consists of a title line 1 and the SEREP run request 2 followed by a line that contains the CPU model 3, serial number 4, job identity 5 and program identity 6. It follows with information pertinent to the failure. The channel and unit address 7, the CCW 8, the CSW 9, the ECSW 10, the sequence code 11 and a break out of the unit and channel status 12 and 13. Item 10 the ECSW and 11 the seq code will be covered in the actual CPU training.

Part 2 - This page consists of the information in the Machine Check Registers and will depend on the actual system type.

Part 3 - This page contains the Local Storage Register information which pertains to the channel.

Part 4 - This page contains the summary information and the Storage Protect Keys.

Answer the following study questions and when you have finished, input your answer via the terminal.

Study Questions (Section 12.2)

Refer to Figure 12.2

1. Select the correct channel status shown in Figure 12.2 (part 1 of 4):

   a. program check
   b. chan data check
   c. chan ctl check
   d. interface ctl check
   e. chaining check

2. Select the run request from Figure 12.2 (part 1 of 4) that SEREP found when it was IPLed.

   a. channel error (0F)
   b. device error (1F)
   c. test channel error (2F)
   d. device not operational (3F)

SECTION 12-3 DITTO

DOS Interfile Transfer and Manipulator Program for Aid in Testing and Operations (DITTO) is a generalized program which operates in a DOS environment. Thirty-three functions are available to support Unit Record, Tape, and 2311, 2314 Disk. Available in addition to normal tape, disk and card functions are:

Tape and Disk Record Scan and Record Alteration
Disk IO Volume Number Change
Initialize Tape
Deblocking of Tape and Disk Records when printing
User Tape Error Handling

Operating Characteristics are:

1. The program is self-relocating to allow operating in any partition.

2. Operation may be performed entirely from the console or from the card reader as part of a job stream.

3. All console communication uses physical hardware I/O addresses, which eliminates the need for knowledge of current logical assignments. Tape density and mode settings may be entered via the console.

4. DITTO functions in a DASD file protected environment without volume label statements. Printed output reflects the physical and logical addresses and file characteristics (density, etc) of the input file.

Console Operation

All operator communication may be in upper or lower case. The end of message may be denoted by the EOB/END key or by repeatedly depressing the space bar.

Tape address replies are entered in the form 'cuumm' where c = Channel, uu = unit address and mm = density and mode setting.

DITTO may be executed entirely from the console. Refer to Figure 12.3.

A normal DOS IPL has been completed and the '1100A Ready for Communications' message printed. At that time, the DITTO function is initiated by entering // EXEC DITTO 1 . The DITTO program responds with the message DITTO FUNCTION? 2 The operator response with CP 3 , which is card to printer. When the card to print function is completed, the DITTO program again prints the message DITTO FUNCTION? 4 . For this example, there were no other functions required, so the operator responded with EOJ 5 . That ends the DITTO job and Job Control again prints '1100A Ready for Communications'.

Refer to Figure 12.4 for a complete list of the DITTO commands that can be submitted via the Console typewriter.

A complete list of all functions available and/or parameters can be obtained by typing "xxx" as shown in Figure 12.4 (next to last statement).

DITTO Operation Via Control Cards

If the user wishes to remove the decision requirements from the console operator, he can prepare control cards and submit the task in a normal Job Stream environment.

To denote control card operation, the user must submit a // UPSI 1 card between the Job card and the EXEC card. (A UPSI card sets program switches [binary bits] in the Communication Region.) DITTO will test the Communication Region to determine whether control card or console operation is desired.

All required control card information can be contained on one card. Each control card is treated as a new operation. The control card format is as follows:

Card Columns 1-7	$$ DITTO
Card Columns 10-12	Function Code Refer to Figure 12.5 and note the functions codes vary slightly for those function codes submitted via the console typewriter (Figure 12.4).
Card Column 16	Parameter 1, ..., Parameter N (Refer to Figure 12.5)

Parameters are in standard key word format. Each parameter must be separated with a comma with no embedded blanks. A blank stops the card scan. Lower case letters denote user supplied information. Parenthesis ( ) denote option parameters.

Refer to Figure 12.6 for a listing of parameters associated with each function.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 12-3)

1. (True/False) Operation of DITTO may be performed entirely from the console or from the card reader as part of a job stream.

2. Select the correct DITTO function to perform a card to printer in character and Hex format.

   a. CC
   b. CP
   c. CD
   d. CCS
   e. CT

3. Select the correct DITTO function to perform a Tape to Printer Reblocked in Character and Hex format.

   a. TP
   b. TDD
   c. TFA
   d. TC
   e. TPV

4. Select the correct control card necessary to perform the following DITTO function: Tape to Printer Reblocked in Character format.

a. $$DITTO TPV INPUT=SYSnnn (, NBLKS=nnnn)
b. $$DITTO TPD INPUT=SYSnnn, RECSIZE=nnnnn(, NBLKS=nnnn)
c. $$DITTO TFD INPUT=SYSnnn, RECSIZE=nnnnn(, NBLKS=nnnn)
d. $$DITTO TFA INPUT=SYSnnn
e. $$DITTO TC INPUT=SYSnnn

SECTION 12-4 - DEBE AND DASD PRINT PROGRAMS

DEBE II (Stand-alone)

DEBE II is a stand-alone program that functions as a multiple utility program.

DEBE II can be used to exercise I/O devices. It can also be used to reproduce data (card and tape) for testing purposes.

Functions

DEBE II is a standalone program and must be IPLed. This can be from either card or tape.

The user requests the functions he wants via the console.

The program uses a two letter code for each function.

Operation

Refer to Figure 12.7. 1 Place the DEBE card deck in the card reader and perform an IPL. When the system enters the wait state, you must press the interrupt key. DEBE will then type the message 'ENTER PROG ID-xx' 2 . You then reply with the two letter code for the function that is required. CC space is a card to card 80/80 reproduce using the physical units 00C and 00D 3 . The remainder of the codes for the other functions are shown in item 4 . Item 5 explains the procedure for putting DEBE onto a self loading tape. These instructions can be obtained by listing the DEBE deck, that is how this figure was made.

DOS-DEBE

DOS-DEBE is a program written in DOS Assembler Language and is designed to provide card, tape printer and DASD file to file utilities. DOS-DEBE is basically the same as DEBE II with the addition of file to file utilities. DOS-DEBE is not a standalone program and is run under the DOS system.

With the DOS-DEBE program cataloged in the Core Image Library, the program can be executed by supplying an // EXEC DEBE statement either from the card reader (SYSRDR) or the console typewriter (SYSLOG). At that time, the DOS-DEBE program will print the message 'ENT PROG ID-xx'. The operator can then reply with one of the utility identifiers for the function he requires. Refer to Figure 12.8. The identifiers for DOS-DEBE are the same as for DEBE II with the addition of the file identifiers.

DASD Print

The DASD Print Program is a standalone utility program that will allow you to print the contents of a disk pack, which is mounted on a drive and ready for accessing.

Run Procedure

The program is loaded from the card reader through a normal IPL operation. After half the deck is loaded, the CPU will enter the wait state (Wait Light ON). The INTERRUPT key on the CPU must be pressed to continue loading the remaining cards from the card reader. When all the cards are read, the System Light will still be on, indicating a loop. At this time, to continue, the REQUEST key on the console typewriter must be pressed TWICE. The program will then print the message that this is a DASD Print. Refer to Figure 12.9 item 1 and also request the address of the DASD device 2. The user responds with the three digit address 3 and hits the END key on the console typewriter. The next line to print is a request for the area of the disk pack you wish to print 4 and is given in the form: from cylinder (3 digits) head (2 digits) to cylinder (3 digits) head (2 digits). All numbers must be entered in decimal directly beneath the specified dots on the console typewriter 5 and hit the END key. This will cause track 00 of cylinder 199 to print. When it has printed the specified area, it will again request a DASD address 6. When there are no additional areas to print, the program can be terminated by pressing the System Reset key on the CPU.

Refer to Figure 12.10. This is the printout of cylinder 199 head 0 and consists of three records, zero, one and two. 1 This information is taken from the count field and gives the cylinder number, head number and record number. 2 This is the count field for the record. The information is printed in EBCDIC 4 and vertically in hex 5. The count field is broken up in the following manner. CCHHRK D D

where
- CC = two bytes for cylinder
- HH = two bytes for head
- R = one byte for record
- K = one byte for key length
- D D = two bytes for data length

Item 3 represents the data field. 6 This field tells us that this track is not defective and not an alternate track. 7 This field contains the Home Address information.

Answer the following study questions and when you have finished, input your answer via the terminal.

Study Questions (Section 12-4)

1. Select the proper commands required to copy a deck of cards onto a 9-track tape, address 181, using the DEBE II utility.

   a. CC and 00181
   b. CT and 181
   c. CT and 00181
   d. TC and 00181
   e. TC and 181

2. (True/False) DEBE II can be used to dump Disk to Printer.

Refer to Figure 12.10 to answer the following:

3. Select the correct count area length for R1 (Record 1).

   a. 8
   b. 44
   c. 96
   d. 10
   e. 12

4. Select the correct key length for R0 (Record 0).

   a. 44
   b. 96
   c. 8
   d. 0
   e. 10

5. Which of the following is the eighth data byte in R1 (Record 1).

   a. 60
   b. 04
   c. 00
   d. 9E
   e. C8

SECTION 12-5 - OS DUMP RESTORE AND DSERV

OS Dump Restore

OS Dump Restore is a standalone utility program that is used to copy from tape to disk or card; or, disk to tape or card.

For the DUMP Function, the program copies information from disk to tape, disk, or cards. On the RESTORE function, it copies from tape or cards to disk.

The Utility is used frequently to copy the DOS resident disk pack to obtain a back-up system.

Operation

The OS Dump Restore is a standalone program and is IPLed from the reader. Once the program has been loaded, the cards will continue to be read until approximately the last four cards. These will be left in the reader. When the reader stops reading the cards, you must press the REQUEST key TWICE to continue. A message will print out requesting you to enter the input device.

At this point, type INPUT = XXXX␣CUU where XXXX is either 2540 or 1442 (if the input device is a 2501 or 2520, you will still enter 2540). ␣ is a blank space, and CUU is the device address of your card reader. You must then hit the END key and the dump or restore function will proceed.

When the dump or restore is complete, a message will print on the console typewriter stating End of Job and the system will enter the wait state.

Card Deck Set Up

Before you run your job, you must be sure that the NEXT-TO-LAST card in the program deck has the correct information.

Refer to Figure 12.11 for the proper format for this card. DUMP or RESTORE **1** signifies the type of function. The FROMDEV **2** can be a 2400, 2311, 2314, 2319, or 3330. The FROMADDR **3** is the physical address of the disk or tape unit. TODEV **4** can be 2400, 2311, 2314, 2319, or 3330. TOADDR **5** is the physical address of the receiving device. VOLID **6** must be specified for restore functions (going from tape to disk). The VOLID punched in the card must match the VOLID (Volume ID) of the disk pack on which data is to be written. MODE **7** is optional for dumping to tape. If mode is not specified, data will be written at the highest density supported by the tape drive.

MODE = CB for 800 bpi output
MODE = C3 for 1600 bpi output

DSERV

DSERV is a service function that will display the contents of a directory (core image directory, relocatable directory, source statement directory, transient directory, or system directory).

The display function prints the contents of the directories defined for the system. Any number of directories can be displayed within a single run, and the system directory is always displayed.

Control statement input for the display function, read from the properly assigned device, (usually SYSIN), is: Refer to Figure 12.12.

1. The Job Control statement.
2. The EXEC DSERV control statement.
3. The DSPLY control statement.
4. The /* control statement if other job steps are to follow.
5. The /& control statement, which is the last statement of the job.

The DSPLY control statement in the format

DSPLY dir 1,[dir 2,...]

is used to display a specific directory where dir represents the name of the directory to be displayed. If more than one directory is to be displayed, the symbols for the directories must be separated by a comma and can be in any order.

DSPLY ALL can be used if all five directories are to be displayed.

The DOS/VS SADP manual contains all information required to perform library functions (Section 2).

Answer the following study questions and when you have finished, input your answers via the terminal.

Study Questions (Section 12-5)

1. (True/False) To copy the contents of a disk onto tape or card, requires a dump function of the OS Dump Restore program.

2. Select the correct entry for the operand field of the DSPLY control card to disply the source statement library directory.

   a. CD
   b. TD
   c. SD
   d. RD

SESSION 13 - PROBLEM DETERMINATION PROGRAM

SECTION 13-1 - DOS STANDALONE DUMP (DUMPGEN)

If at any time DOS is "hung up" in a hard wait state, a standalone dump could be invaluable in displaying the current information in storage. Because the standalone dump program must be loaded into storage, some current information in storage will be overlayed by the dump program.

Depending upon the conditions and the contents of storage, the dump program may have to fulfill some specific needs of the user. Because of this, there is a need for a variety of dump programs.

Listed below are a number of items that could be considered before using a dump program:

- How much storage is to be displayed?
- What information in storage is least valuable, so it can be overlayed by the dump program?
- Is it necessary to format information in the Supervisor and information contained in low main storage for ease of interpretation?

A facility to generate dump programs that can be tailor made to meet a users specific need is provided. The user is able to generate many different types of dump programs by using the DUMPGEN program. This program reads control cards from SYSIPT then generates dump programs tailor made to specifications coded in these control cards.

The DUMPGEN program outputs to the device assigned to SYSPCH which can be a card punch or tape device. There are two types of statements read by the DUMPGEN program.

1. ASSGN statement - tells DUMPGEN the physical device address of the printer that the generated dump program will display main storage on.
2. OPTN statement - tells DUMPGEN specific options to generate into the dump program, such as:
   a) The contents of real storage (in 2K increments).
   b) The number of self-loading dump decks to produce.
   c) If data is to be formatted.
   d) If the dump program is to be outputted on tape or punched cards.

For the following work project refer to the DOS/VS SADP manual (GC33-5380) Section 2 - DUMPGEN.

Work Project

Using the following user specifications, fill in the blanks in the control statements to generate a "tailor made" Standalone dump.

User Specifications

a. The standalone dump is to output to a printer which has a physical address of X'002'.
b. Generate 3 card decks of the standalone dump.
c. The output is to be a formatted dump.
d. Dump 32K bytes of main storage.

```
// JOB JET
// EXEC _____
   ASGN SYSLST, _____
   OPTN INTR = _____
   OPTN DECKS = _____
   OPTN FORMAT = ____
/*
/&
```

Answers are on the following page.

Work Project Answers

```
// EXEC DUMPGEN
   ASSGN SYSLST, X'002'
   OPTN INTR = NO      - This statement could have been omitted as INTR=NO is
                         assumed.
   OPTN DECKS = 3
   OPTN FORMAT = YES
```

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 13-1)

1. (True/False) The PPOOL option for the DUMPGEN program defaults to NO if it is not included in the job stream.

2. To have DUMPGEN produce a stand-alone dump program on tape, you must:

   a. Code TAPEIPL = YES in an OPTN statement.
   b. Assign SYSLST to a tape drive.
   c. Code INTR = Yes in an OPTN statement, then when the system enters the WAIT state, make the desired tape drive Ready.
   d. Assign SYSPCH to a tape drive.

3. (True/False) At least one ASSGN statement and one OPTN statement must be submitted to DUMPGEN.

4. (True/False) To execute the stand-alone dump the cards are placed in the reader, the STOP key is pressed, SYSTEM RESET, and LOAD keys in that order.

5. (True/False) The original information in Bytes 0 thru 23 are destroyed when a dump is loaded into storage.

SECTION 13-2 - FORMATTED DUMP

One of the operands in the OPTN statement that can be submitted to DUMPGEN is FORMAT=. This session will deal with the advantages of using OPTN FORMAT = YES. In prior sessions, you were given projects that required analysis of nonformatted dump to locate various tables. We will now take a look at a formatted dump and point out the various information readily available.

Refer to the DOS/VS SADP manual (GC-5380) for the following discussion.

Review the introductory paragraphs prior to the figure in Appendix G - Example of a Stand-alone Dump output.

You will remember that the following items were found in the nonformatted system dump and the first part of the formatted stand-alone dump:
- COMREGs for all partitions
- Various table and control block address
- LUB and PUB tables
- Error queue
- Channel queue
- Partition save areas

All of these can be found in the formatted section of the stand-alone dump in Appendix G.

As you thumb through the Appendix G dump you will note several more formatted tables.

Communication Region

The communications regions are formatted as follows:

The first column is the hex displacement into the communication region. The next five columns are the contents of the background and foreground communications regions. The last column is an explanation of that particular field in the communications region.

Partition Save Area

Contains the contents of the PSW and general registers that were in effect when the partition was interrupted (lost control of the system resources). Note that if the foreground partitions were active, they too would have information in this area.

Logical Unit Block Table

The contents of the LUB table is formatted as follows:

1st Column - LUB TAB indicates the displacement into the LUB table for a specific entry in the LUB.
2nd Column - LOGIC UNIT indicates the logical unit name for a specific entry.
3rd Column - PUB PRT indicates the entry in the Physical Unit Block (PUB) that is associated with the LUB entry.
4th Column - JIB PTR - this entry is of use only to the Program Support CE and will not be discussed.
5th Column - CUU indicates the physical unit address that is associated with the LUB entry.

BG SYSTEM LUBs (SYSaaa: where aaa is alphabetic).

Contains the contents of the background system LUB entries.

BG PROGRAMMER LUBs (SYSnnn; where nnn is numeric).

Contains the contents of the background partitions LUB entries.

The contents of the foreground partition LUBs are broken down the same way as the background LUBs, that is, the systems LUBs are displayed first then followed by the programmers LUBs.

Physical Unit Block Table (PUB)

The contents of each entry in the PUB is formatted.

Byte 6 of the PUB entry (channel schedule flags) is further broken down into bits and the meaning of a bit being on is explained by the heading. If any of the bits are on in this byte, the appropriate asterisk (*) will print under the appropriate heading.

For example, note that device 0009 has an asterisk, indicating Dev Busy.

Error Recovery Block

This is formatted such that the ERBLOC (16 bytes) is listed first, followed by the data in each of the error queue entries (44 bytes each).

NOTE 1: The data in the error queue can be misleading if no error occurred as it may contain residual information.

NOTE 2: A layout of the error recovery block can be found in Section 4, chapter 3 of the DOS/VS SADP.

Channel Queue Table

Formats each entry in the channel queue as to position, chain pointer, CCB address and the physical unit address (CUU).

Read the associated notes on the remaining formatted tables.

Use the formatted dump on Appendix G to answer the following questions. The answers follow the questions.

1. What is the address for the F4 partition of the:

   PUB table _____
   LUB table _____
   COMREG _____

2. What is the physical address assigned to the BG SYSRDR? _____

3. (True/False) The contents of register 1 for the F2 partition is '00089D24'.

4. What is the contents of the third entry in the PUB table? _____

5. (True/False) The device in the above question (4) is a 2540 Card Punch.

Answers

1. PUB table 4104
   LUB table 4F0C
   COMREG 4590

   See the formatted section of the dump communication region.

2. 00C - See the formatted section of the dump logical unit block table.

3. True - See the formatted section of the dump partition save area.

4. 000DFF00210000F8 - Each PUB entry is eight bytes long. The PUB table starts at address 4104 and the third entry starts at 4114 if you want to compare this to the formatted PUB table entry.

5. True - The device type is the fifth byte of the PUB entry - 21 - reference Section 4, part 2 - Device Types.

SECTION 13-3 PDAID - I/O TRACE

A program supplied by IBM to trace various events which are extremely helpful in Problem Determination is the PDAID program. When executing PDAID (// EXEC PDAID) the user has the opportunity to select one of four tracing options provided by the program. These four options are:

1. Fetch/Load Trace - Records pertinent data associated with every event that involves fetching or loading of other programs.
2. Input/Output Trace - Records pertinent data associated with all or selected I/O activity.
3. Generalized SVC Trace - Records pertinent data associated with all or selected Supervisor Calls.
4. QTAM Trace - Records pertinent data associated with QTAM events.

The tracing option that is of greatest importance to the hardware CE is the I/O trace option. The other options have greater significance to programmer and will not be discussed in this course.

Refer to the DOS/VS SADP manual (GC33-5380). Read in Section 2B, Part 1.

- PDAIDS to fetch/load trace.

Pay particular attention to the example of the printouts received by these PDAID functions.

NOTE: Disregard any reference to cataloging of programs and system requirements necessary. For our purposes assume these requirements are satisfied.

Note the table "Options and Control Statements for Executing PDAIDS" and the associated operator's flowchart for running the PDAIDS.

- PD area including locating and dumping the PD area. Reference the trace entry table showing the relative locations within the PD area of each type trace function is performed.
- Initiating the PDAID trace routings through but not including operator's flowcharts.

Displaying PD Area during Wait State or Loop

The previously described methods for identifying events via the PDAIDS are useful but, if a system enters a hard wait, a stand-alone dump is required to display the PD area of storage.

In addition, there are operator commands available with which you can use to display the PD area if the system loops or enters a soft wait. If the system is in a soft wait or loop you can use the dump and alter/display commands as described in Session 4 of this course.

Work Project

Fill in the blanks in the following job stream to:

a. Trace all I/O devices during the execution of a user program except for the SYSRES device (X'190').
b. Output the data from the trace on tape drive X'180'.
c. Code the statements necessary to terminate the I/O trace function following the user program.
d. Code the statements to list the data collected on the tape.

NOTE: There may be more blank spaces than required. Use only the blanks needed to meet the above conditions.

```
// JOB EXER
// EXEC _____
   PDAID = _____
   OUTPUT DEVICE = _____
   IGNORE DEVICE = _____
_____
_____
_____
/*
// EXEC USRPRGM
/*
_____
_____
_____
/*
_____
_____
_____
/*
```

Answers

```
// JOB EXER
// EXEC PDAID
   PDAID = IT
   OUTPUT DEVICE = X'180'
   IGNORE DEVICE = X'190'
   GO
/*
// EXEC USRPRGM
/*
// EXEC PDAID
   PDAID = XX
   GO
/*
// ASSGN SYS005,X'180'
// MTC REW,SYS005
// EXEC PDLIST
/*
/&
```

Study Questions (Section 13-3)

1. To identify an I/O event that was a result of an SIO (CSW Stored Condition) the I/O event data contains:

   a. an * in the first position of the PSW.
   b. the next instruction address in the last 3 bytes of the PSW.
   c. CSW information only.
   d. An E240 in the first 2 bytes of the trace entry.
   e. the CCB address in the last 3 bytes of the PSW.

2. The name of the program that is used to print data the PDAID program outputted to tape is:

   a. IOLST
   b. PDAIDLST
   c. TRCLIST
   d. PDLIST
   e. LSTPDID

3. (True/False) The CCB address is placed in the PSW only when the I/O event being recorded is the result of an SIO.

Given: The response to the 4C10D message is
PDAID=IT

Use the following entry from the PD area to answer questions 4 and 5:

E2400000000EFF00861000000000004000000

4. (True/False) The entry was made as a result of a successful SIO operation.

5. The significance of the 8610 in the entry is:

   a. CCW address
   b. CCB address
   c. Device status information
   d. No significance

SESSION 14 - POWER CONCEPTS

Priority Output Writers, Execution Processors and Input Readers

SECTION 14-1 - POWER CONCEPTS

The processing of jobs by the computing system was first introduced in this course as one job following another; that is, one job had to complete before the next job could begin.

Next, multiprogramming was introduced, showing how up to three jobs could be in storage at one time. Here again, in each partition, one job had to complete before the next job could begin. Multiprogramming improved job throughput (more jobs processed during a specific period of time), however, more I/O devices are usually required.

In each of the two situations above, the time required to process the jobs is dependent on the speed of the I/O devices used. Unit record devices (readers, punches and printers) are slow in comparison to disk or tape, consequently, much CPU time is actually wasted when performing unit record I/O operations.

The resources of a computing system represent a considerable investment. It is important, therefore, to use them efficiently. To do this, it is necessary to keep each resource busy doing the kind of work it is intended for, such as keeping the CPU processing and the I/O devices running. The better these are accomplished, the better the performance of the computing system. The performance of a computing system is determined by a combination of three factors, throughput, response time, and availability.

Let's define these terms.

Throughput - Throughput is the total volume of work performed by a computing system for a given period of time.

Response Time - Response time (sometimes referred to as a turn around time) is the interval between the time a user submits a job and the time he receives the results.

Availability - Availability is the degree to which a computing system is ready when needed to process data.

IBM developed POWER for DOS customers to improve system performance primarily in the area of throughput and availability. This is accomplished basically by performing all unit record I/O operations on disk or tape, by redirecting the output for a printer or punch to disk or tape. The output will be printed or punched later. For example: a problem program issues an I/O request to print a line. POWER intercepts the I/O operation and causes the data to be written on disk(this is called SPOOLing). The data will be printed later. The POWER functions are transparent to the problem program, that is, as far as the problem program is concerned, it thinks it is printing a line on the printer.

POWER operation will be explained later in this session. First, let's look at some advantages of using POWER. This can best be illustrated by using the example in Figure 14.1. Each job consists of CPU processing and printing of output.

1 This portion of the figure represents the processing of jobs in any one partition of a DOS system without POWER. The duration of each job includes the combination of CPU processing and printer I/O operations. Each job is considered complete when all the output has printed. The jobs are executed in-line, that is, job 2 cannot begin until job 1 has completed.

2 This portion of the figure represents the processing of the same jobs in any one partition of a DOS system with POWER. The duration of each job is divided into three segments:

- CPU time - CPU processing time for the job. The print records are spooled (written) to disk. When these records are on disk, they are referred to as being on the OUTPUT QUEUE.
- Queue time - Queue time is the time the print records are on the OUTPUT QUEUE waiting to be printed.
- Printer time - printer time is the time required to print the output from the OUTPUT QUEUE.

The duration from 3 to 6 represents the time required to process job 1 without POWER. The print operations were performed directly on the printer.

The duration from 3 to 7 represents the time required to process job 1 with POWER. The print operations were first spooled to disk (output queue) and then the output was printed from the output queue.

You can see from this that the turnaround time for job 1 was longer when POWER was used. So far, we don't see any advantage to using POWER, but let's analyze the job stream further.

The duration from 3 to 4 represents the CPU processing time for job 1. During this time, the printer records are spooled on disk. Because of the speed of the disk as compared to the printer, less CPU time is required to process the job. At 4 job 2 begins to execute, spooling its printer records on disk. Here again, because of the speed of the disk, less CPU time is required. As soon as job 2 completes job 3 begins etc. All of the jobs have completed processing, using the disk for I/O operations, between 3 and 5 .

The duration from 4 to 7 represents the time to print the output for job 1 from the output queue. This printing, however, is overlapped with the processing of other jobs. By the time job 1 output has finished printing, the output queue contains the print records for job 2, 3, 4 and 5.

The duration between 7 and 8 represents the time to print the output for jobs 2 through 5. Note that the time required to print the output for jobs 2 through 5 is considerably less then the time it took to print the output for job 1. This is because the CPU is not processing any jobs. The CPU is available to process more jobs, starting at 5 .

All the jobs are complete by 8. This includes CPU processing time and the time required to print the output from the output queue. The duration from 8 to 9 represents the time that the system is actually not used and is available to process more jobs.

You can now see the advantage of using POWER. The CPU can actually process more jobs in a given period of time. This is increased throughput and availability.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 14-1)

1. (True/False) All the unit record I/O instructions in the problem program are modified to disk I/O instructions when using POWER.

2. A DOS system using POWER:

   a. requires more time to complete a job.
   b. can process more jobs in a given period of time.
   c. can print the output of one job from the output queue while processing another job, putting its output records on the output queue.
   d. all of the above.

Refer to Figure 14.1.

3. (True/False) When using POWER, job 2 begins processing before job 1 output has finished printing.

SECTION 14-2 POWER DATA FLOW

Now that the advantages of POWER have been pointed out, we can take a look at the data flow within a POWER system, using the following series of diagrams.

POWER System Configuration - Figure 14.2

1. Core storage is divided as in a multiprogramming system. POWER will run in either the F1 or F2 partition. With POWER running in F1, it will support both F2 and BG partitions. This is considered a 2 partition POWER system. With POWER running in F2, it will support only the BG partition. This is considered a one partition POWER system.

2. The INPUT QUEUE is an area on disk used for input data from the card reader.

3. The PRINT and PUNCH QUEUE's are an area on disk used for print and punch output records. In section 14-1, we referred to these Queues as the OUTPUT QUEUE.

4. The unit record devices supported by POWER are the printer, reader and punch.

5. The Reader Task is a portion of POWER which reads the input data from SYSRDR and writes it on the INPUT QUEUE.

6. The Print and Punch Tasks are a portion of POWER which reads the output data from the PRINT and PUNCH QUEUES and outputs it on SYSPCH or SYSLST.

POWER Data Flow (Part 1 of 5) Figure 14.3

1. The job stream in SYSRDR is read in by the POWER Reader Task and written on the INPUT QUEUE. Each job is identified by job name and the partition in which it is to run.

2. When the Reader Task has completely read in the first job, it signals the POWER nucleus that there is a job in the INPUT QUEUE ready to execute. For our example, the job is named SAMPLE.

3. The job control statements are read from the INPUT QUEUE and analyzed to determine the program to be executed and the partition it is to run in.

4. As a result of the EXEC statement, the system loader is notified to load the program ASSEMBLY, from the core-image library, into the BG partition. While the program ASSEMBLY is processing, the remainder of the job stream continues to be read from SYSRDR and written on the INPUT QUEUE by the Reader Task.

5 When the second job is completely in the INPUT QUEUE, POWER is again signaled 2 . The job control statements are analyzed for this job to determine the program to execute and the partition in which it is to run.

6 As a result of the EXEC statement, the system loader is notified to load the program PAYDAY into the F2 partition. If this program were to run in the BG partition, it would not be loaded into core until the program ASSEMBLY completed.

At this point in time, the program ASSEMBLY is executing in the BG partition, the program PAYDAY is executing in the F2 partition and the Reader Task continues to read jobs from SYSRDR and writing them on the INPUT QUEUE.

POWER Data Flow (Part 2 of 5) Figure 14.4

1 With the job stream completely in the INPUT QUEUE, the Reader Task remains in core storage but is inactive, waiting for more cards to be put in SYSRDR.

The conditions that exist now are:

(1) The program ASSEMBLY is processing in the BG partition.
(2) The program PAYDAY is processing in the F2 partition.
(3) The job stream is completely in the INPUT QUEUE.

We will now analyze I/O operations under POWER.

2 The program ASSEMBLY requests an I/O operation from the card reader. It does this by issuing a SVC0. Remember all I/O operations are handled by the Supervisor.

3 Because this operating system includes POWER, the Supervisor gives control to POWER to handle all unit record I/O requests.

4 POWER analyzes the I/O request. If the request is from a program in a partition that is supported by POWER, the request will be redirected to a disk I/O routine in the supervisor and the I/O data will be read from disk. If the request is from a program in a partition that is not supported by POWER, the request is given back to the supervisor to handle in the usual way.

5 The data is read from the INPUT QUEUE and presented to the ASSEMBLY program. This I/O data transfer is performed at disk speed instead of at the speed of the reader.

This sequence of events is repeated each time a problem program requests input from the card reader.

POWER Data Flow (Part 3 of 5) Figure 14.5

Print and punch operations are performed much like reader operations.

1. The problem program requests a print or punch operation by issuing a SVC 0 to the Supervisor.

2. Because the operating system includes POWER, the Supervisor gives control to POWER to handle all unit record I/O requests.

3. POWER analyzes the I/O request. If the request is from a program in a partition that is supported by POWER, the request will be redirected to a disk I/O routine in the supervisor. If the request is from a program in a partition that is not supported by POWER, the request is given back to the supervisor to handle in the usual way.

4. The request can be to either punch a card or print a line. In either case, the record is written on the appropriate output QUEUE. The punch records are written on the PUNCH QUEUE and the print records are written on the PRINT QUEUE. Here again the I/O data transfer is performed at disk speed instead of at the speed of the punch or printer.

POWER Data Flow (Part 4 of 5) Figure 14.6

1. When the problem program is finished executing (all the output is in the Print and Punch Queue) it signals POWER.

2. POWER signals the PRINTER Task and the PUNCH Task that there is data in the PRINT and PUNCH QUEUE's.

3. The next set of job control statements are read from the INPUT QUEUE and analyzed to determine the program to execute and the partition it is to run in.

4. As a result of the EXEC statement, the system loader is notified to load the program INVENTORY, from the core-image library into the BG partition and overlay the program ASSEMBLY.

5. The Printer Task begins reading the data from the PRINT QUEUE and printing it. The Punch Task begins reading the data from the PUNCH QUEUE and punching it.

POWER Data Flow (Part 5 of 5) Figure 14.7

The following conditions exist at this time:

(1) The program INVENTORY is executing in the BG partition.
(2) The program PAYDAY is executing in the F2 partition.
(3) The Printer Task is printing the output for the job SAMPLE.
(4) The Punch Task is punching the output for the job SAMPLE.

1. As the program PAYDAY is executing, output records are written on the PRINT QUEUE.

2. When the Printer Task and Punch Task complete printing and punching, the output for job SAMPLE, they signal POWER that they have completed.

3. When all the output for job SAMPLE has completed punching and printing, POWER purges the INPUT QUEUE records for the job and the disk space is available to the Reader Task.

4. The PRINT and PUNCH QUEUE's are also purged of output records for the job SAMPLE and this disk space becomes available for output of succeeding jobs.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 14-2)

1. Writing the job stream on the INPUT QUEUE is a function of:

   a. the Reader Task.
   b. POWER nucleous.
   c. the supervisor.

2. When a problem program running under POWER issues a request for card input:

   a. the supervisor gets control and reads the card from the card reader.
   b. POWER will control the operation and read the card from the card reader.
   c. POWER will control the operation and cause the data to be transferred from the INPUT QUEUE to the problem program.

3. The records for a specific job in the INPUT QUEUE:
   (More than one answer is required.)

   a. may contain job control statements and data records.
   b. are purged when the job has completed executing.
   c. identify the partition that the job is to execute in.

4. (True/False) The PRINTER TASK is responsible for writing the printer records on the Printer Queue.

SECTION 14-3 - SERVICING IN A POWER ENVIRONMENT

Because of more efficient system utilization with POWER, many DOS customers are using it. As a result, you will at some time be required to service a POWER system. System operation is modified somewhat. A typical situation you might experience is best explained using the example in Figure 14.8.

1 The INPUT QUEUE contains 7 jobs. Jobs 1, 2 and 3 have completed processing.

2 The PRINT QUEUE contains output for jobs 1, 2 and 3. The output for job 1 has completed printing. The PUNCH QUEUE contains output for job 3.

3 Job 4 is currently executing in core.

Situation:

You suspect a tape drive is failing and you want to run the tape Inter-Block-Gap On-Line-Test. The INPUT QUEUE still contains 3 jobs yet to process. When you submit your On-Line Test job, it becomes job 8 in the INPUT QUEUE. The time required to process all the jobs that are ahead of yours and print all the output might take several hours. You obviously can't wait that long to run the On-Line-Test, and you don't have to. POWER includes the facility to assign priorities to jobs so that a critical job, although entered last in the INPUT QUEUE, can be the next job to execute.

4 POWER provides the capability of establishing a priority to jobs. Although the On-Line-Test job is the last job in the INPUT QUEUE, it will execute immediately after job 4, if it is assigned the highest priority.

The details of how to prepare job control statements to run jobs in a POWER environment will be covered in the next session.

SESSION 15 - POWER OPERATION

SECTION 15-1 - POWER OPERATION (JOB STATEMENT)

The typical POWER environment that we saw in the previous session is shown in Figure 15.1. Let's review the conditions we had.

1 The INPUT QUEUE contained job control statements for 7 jobs. The first 3 jobs have completed processing with the job control statements for job 1 purged from the Queue. Job 4 is currently executing.

2 The PRINT QUEUE contained output for 3 jobs. The output for job 1 has already completed printing and has been purged. The output for job 2 is currently printing.

3 The PUNCH QUEUE contains output for job 3.

The situation that existed was:

We wanted to run a job such as an On-Line-Test. The last session established that priorities could be assigned to jobs so that the On-Line-Test could execute immediately after job 4. This is accomplished by using POWER JECL (Job Entry Control Language) statements.

Using POWER JECL statements is optional, however added flexibility and control, such as establishing priorities, is provided when they are used. Some of the functions that can be performed using POWER JECL statements are:

(1) assigning priorities to jobs.
(2) Holding a job in the INPUT QUEUE. The job will execute only when the system operator releases it from the Hold status.
(3) Spooling output to tape instead of disk.
(4) Request that print and punch output not be spooled to disk. The output will print or punch as the program is executing.
(5) Specify the partition in which a job is to execute in.

The use of POWER JECL statements can best be illustrated using a sample job stream. Refer to Figure 15.2.

1 This is a typical DOS job stream containing 3 jobs. This job stream can run under POWER, however, since there are no POWER JECL statements in it, all of the jobs will run in the BG partition and be assigned the lowest priority or the priority default, which was established when the POWER program is assembled.

2 This is the format of the POWER JOB statement. The major difference between it and the DOS JOB statement is the * in column 1 and the $ in column 3 and 4.

3 jobname: specifies the name by which the job is known to the POWER system. The jobname may be from 1 to 8 alphameric characters. If jobname is not specified, the POWER system will assign AUTONAME as the jobname.

4 hold: specifies that the job is to be held in the input queue until it is released by the operator. Hold is specified with the character H. If hold is not specified, the job will be placed in the input queue on a first-in, first-out basis within its priority classification. What this means is that the Job with Hold specified will not be brought into core until specified by the operator, regardless of priority.

5 priority: specifies the priority assigned to this job. Priority is specified as a single numeric character from 0 to 9. Nine is the highest priority. If priority is not specified, the POWER system will assign to the job a priority based on the PRIORTY generation option.

6 partition: specifies the partition to which the job is assigned for execution. Partition is specified as either BG or F2. If partition is not specified, BG is assumed. If F2 is specified, the POWER system should have been generated to support the foreground 2 partition.

Flexibility can be added to the job stream by replacing the DOS JOB statements with POWER JOB statements.

7 This is the POWER JOB statement that might replace the first DOS JOB statement. Let's examine it closer.

The first operand - job name - is SAMPLE
The second operand - hold - is omitted
The third operand - priority - is 3
The forth operand - partition - is omitted

The job SAMPLE is assigned the priority of 3 and will execute in the BG partition.

The operand field contains one or more positional parameters of information separated by commas. Positional parameters are described as values for which information must be substituted and must be declared in a specific order.

The operand field has no fixed length. A blank terminates the operand field; therefore, none of its parameters may contain embedded blanks. If trailing parameters are omitted, trailing commas may be omitted also.

8 This is the POWER JOB statement that might replace the second DOS JOB statement. The job name is CHECKREG with a priority of 5 and will execute in the BG partition.

9 This is the POWER JOB statement that might replace the third DOS JOB statement. The job name is LIST with a priority of 9 and will execute in the BG partition.

The processing of this job stream using POWER JOB statements is shown in Figure 15.3.

1 The POWER JOB statements indicate:
(1) All jobs will run in the BG partition.
(2) Job LIST has the highest priority.
(3) Job CHECKREG has the next highest priority.
(4) Job SAMPLE has the lowest priority.

2 Looking at the job entries in the INPUT QUEUE, you might think that job LIST will execute first followed by job CHECKREG and job SAMPLE. However, let's review the operation of the Reader Task as it reads the job stream and builds the INPUT QUEUE.

3 The READER Task, as it reads the job stream, will signal the POWER nucleus that it has completed loading the first job into the INPUT QUEUE.

4 As soon as the entry for the first job is in the INPUT QUEUE, it will be loaded into core and start executing.

5 As the first job is executing, the READER Task continues to read the remainder of the job stream into the INPUT QUEUE.

6 When job SAMPLE completes processing, the job in the INPUT QUEUE with the highest priority, will execute next. In our example, job LIST will execute next with job CHECKREG executing last.

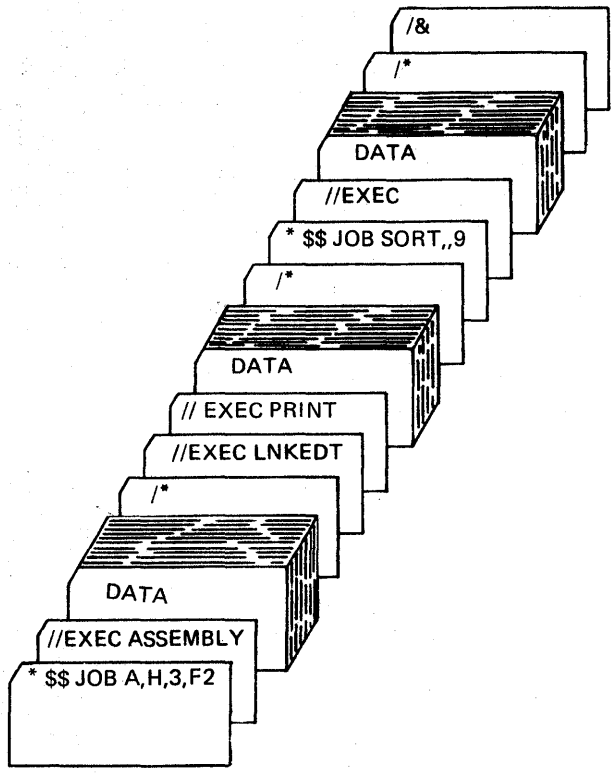
NOTE: If more then one job with the same priority is in the INPUT QUEUE. The first job that was read into the QUEUE will be the first one to execute.

Summary

The POWER JOB statement provides the facility of establishing priorities to jobs in the INPUT QUEUE and identifying the partition in which the job is to execute. Although the POWER JECL statements provide flexibility to the jobs running under POWER, they ate optional. POWER will accept standard DOS job control statements.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Use the following job stream as a reference to answer the study questions.



Study Questions (Section 15-1)

1. Which job will execute first when this job stream is read into the INPUT QUEUE?

   a. JOB A
   b. JOB LIST
   c. JOB SORT

2. What partition will JOB SORT execute in?

   a. F2
   b. BG
   c. F1

SECTION 15-2 - POWER OPERATION (PRT AND PUN STATEMENTS)

The previous section showed how the POWER JOB statement was used to establish priorities between jobs and also indicate the partition in which a job was to execute. This section will introduce two more POWER JECL statements (PRT and PUN) that can be used to control print and punch output.

PRT Statement

The PRT statement is described using Figure 15.4.

1 This is the format of the PRT statement. If it is used, it must follow the JOB statement or the PUN statement. Only one PRT statement is permitted for a job entry and it pertains to all printer output intercepted by the POWER system for that job entry. If the PRT statement is not included in the job entry, the default values of the PRT statement operands are assumed.

2 disposition: specifies the disposition of the printed output during execution and after execution of the job entry. The disposition specification options are listed below:

Option	Meaning
D	SPOOL output to disk intermediate storage.
T	SPOOL output to tape intermediate storage. This specification requires tape output writer capabilities to have been generated in the POWER system; otherwise, it is an invalid operand.
N	Do not intercept print requests. Print output as it is processed.
H	Hold the output until requested by the user.

If the disposition specification is omitted, D is assumed.

3 class: specifies the class of the printed output after execution of the job entry and may be specified as any alphabetic character (A through Z).

Class may be used to group related jobs together so that the output for them will print as a group such as printing the output for all the jobs that are required to process a payroll application.

4 forms-number: specifies the type of forms to be used according to installation requirements. Forms-number is specified by 1 to 4 alphameric characters. If forms-number is not specified, the forms-number is assumed to be blanks. When the print writer task prepares to process output from a job entry, it will determine whether this forms-number agrees with the forms-number of the last job entry. If the numbers agree, the print writer task processes the output. If the numbers do not agree, the print writer task writes a message to the operator to inform him that a change to the forms-number specified is required. The print writer task will wait for the operator to activate the task when the change of forms has been completed.

5 number-of-copies: specifies the desired number of copies of printer output from the job entry. If number-of-copies is not specified, one is assumed. Number of copies is specified by one or two numeric characters. If 0 or 00 is specified, one copy will be produced. Number-of-copies may not be specified if disposition is specified as T. If additional copies are required when disposition is T, the tape writer may be restarted.

devaddr-of-tape: specifies the cuu hardware address of tape drive to be used for tape SPOOLing when the job is being executed. This parameter may be used only when the disposition was specified as T and precludes the use of number-of-copies. 'cuu'is specified as the hexadecimal characters representing the channel and unit address of the tape drive. No default is permitted. If this parameter is omitted, the operator will be required to specify the address at the console at execution time. The devaddr-of-tape may not be one which is used for punch output.

6 number-of-lines-before-msg: specifies the number of print records that are written on the Print Queue before a warning message is printed on the console. This message indicates that the Print Queue is filling up to near capacity.

7 linetab: specifies the carriage control tape format to be used when print requests are intercepted. This parameter is specified as 26 numeric characters and overrides, for this job entry only, the linetab generation parameter. If linetab is not specified, the generated specification is used. Linetab specifies the print lines between the first carriage control tape channel punched and the other punches and the end of the page. Linetab is specified as d d ...d d where d is the number of lines from the channel 12 punch to the first punch; d specifies the number of lines between the first punch and the channel 1 punch; d specifies the number of lines between the first punch and the channel 2 punch; and so forth. If channel 1 is the first punch, the distance is expressed as 00. If a channel is not punched, a 00 should be specified for that channel. Otherwise, d is specified as two numeric characters. If linetab is not generated for the POWER system, the linetab overrride option in the PRT statement is not checked.

8 This is the job stream used in the previous section with PRT statements added to control the printed output for the jobs.

9 This is a PRT statement that might be used in the first job. It indicates that the printer requests for job SAMPLE will not be intercepted by POWER and spooled to disk, but will print directly as the job is executing.

CAUTION: Care should be taken if output is being printed from the PRINT QUEUE at the same time that the output from this job is to print. Unpredictable results will occur.

10 This is a PRT statement that might be used in the second job. It indicates that the print output for job CHECKREG is to be spooled to tape using tape drive 283.

11 This is a PRT statement that might be used in the third job. It indicates that the print output for job LIST will be grouped into class A and spooled to disk.

When the Print Task is ready to print the output for this job, a message will be issued to the operator to mount special forms 32A.

PUN Statement

The PUN statement is similar to the PRT statement. If it is used, it must follow the JOB statement or the PRT statement.

Only one PUN statement is permitted for a job entry and it pertains to all punch output intercepted by the POWER system for that job entry. For example, if the job entry consists of more than one DOS job and if SYSPCH is assigned to a card punch, output from all the jobs is handled in accordance with the PUN statement operands. If the PUN statement is not included in the job entry, the default values of the PUN statement operands are assumed.

The format of the PUN statement is shown in Figure 15.5.

1. disposition: specifies the disposition of the punched output during execution and after execution of the job entry. The disposition specification options are listed below. If the disposition specification is omitted, D is assumed.

Option	Meaning
D	SPOOL output to disk intermediate storage.
T	SPOOL output to tape intermediate storage. This specification requires tape output writer capabilities to have been generated in the POWER system; otherwise, it is an invalid operand.
N	Do not intercept punch requests. This specification allows direct writer capabilities. Care should be exercised, if a punch task has been started, that punch requests are not made to the same device.
H	Hold the output until request by the user.

2. class: specifies the class of the punched output after execution of the job entry and may be specified as any alphabetic character (A through Z). Class may be used in conjunction with card number to group like types of punched output. A punch writer task may be started by class to process this output. Class is ignored unless output is SPOOLed to disk intermediate storage and punched at a local punch.

3. card-number: specifies the type of forms to be used according to installation requirements. Card-number is specified by one to four alphameric characters. If card-number is not specified, the card-number is assumed to be blanks. When the punch writer task prepares to process output from a job entry, it will determine whether this card-number agrees with the card-number of the last job entry. If the numbers agree, the punch writer task processes the output. If the numbers do not agree, the punch writer task writes a message to the operator to inform him that a change to the card-number specified is required. The punch writer task will wait for the operator to activate the task when the change of forms has been completed.

4. number-of-copies: specifies the desired number of copies of punch output from the job entry. If number-of-copies is not specified, one is assumed. Number-of-copies is specified by one or two numeric characters. If 0 or 00 is specified, one copy will be produced. Number-of-copies may not be specified if disposition is specified as T. If additional copies are required when disposition is T, the tape writer may be restarted.

devaddr-of-tape: specifies the cuu hardware address of tape drive to be used for tape SPOOLing when the job is being executed. This parameter may be used only when the disposition was specified as T and precludes the use of number-of-copies. cuu is specified as the hexadecimal characters representing the channel and unit address of the tape drive. No default is permitted. If this parameter is omitted, the operator will be required to specify the address at the console at execution time. The devaddr-of-tape may not be one which is used for print output.

5 number-of-cards-before-msg: specifies the number of records written on the Punch Queue before a warning message is written on the console typewriter. This message indicates that the Punch Queue is filling up to near capacity.

Some examples of the PUN statement follow.

Example 1. Punch output requiring special forms

```
* $$ PUN     ,6509
```

In this job entry, the punch output will be SPOOLed to disk and will require the installation's special cards 6509. One copy is required and the STDCARD value is used.

Example 2. Direct writer

```
* $$ PUN N
```

In this job entry, punch requests will not be intercepted.

Example 3. Punch output SPOOLed to tape

```
* $$ PUN T,,285
```

In this job entry, output is SPOOLed to tape drive 285 during job execution.

Summary

The PRT or PUN statements, if used, must follow the POWER JOB statement in any order. The PRT or PUN statement can be used to spool output to tape instead of disk, request multiple copies of the output or identify a special form to be used for the output. The PRT and PUN statements are optional.

SECTION 15-3 - POWER OPERATION CONSOLE COMMANDS

The two previous sections explained how to prepare POWER JECL statements. This section will show how they might be used to run a job in a typical POWER system configuration.

A situation that you might encounter in the field is shown in Figure 15.6.

1. The INPUT QUEUE contains entries for 6 jobs.

2. Job ASSEMBLY is currently executing under POWER.

3. The PRINT QUEUE contains output records for 3 jobs. The Print Task is currently printing the output for job LIST, while POWER is spooling the output records for job ASSEMBLY to the PRINT QUEUE and PUNCH QUEUE. The output for job SORT is waiting to be printed.

4. On the printer is the complete output for job SAMPLE and a portion of the output for job LIST.

In order to submit a job that will run immediately after the job ASSEMBLY, the operator or the CE must be able to interrogate the POWER system to determine:

(1) The number of jobs in the INPUT QUEUE and their priorities.
(2) The number of jobs in the PRINT and PUNCH QUEUES and their priorities.

With this information available, you can then prepare the POWER JECL statements that will cause a job to execute immediately after job ASSEMBLY.

The POWER system is interrogated using Queue management console commands.

With these console commands, the system operator can:

(1) Display the status of jobs in the Queues.
(2) Alter priorities of jobs in the Queues.
(3) Hold a job in the Queues.
(4) Delete a job from the Queues.
(5) Release a job from the Hold state.

This section will show how to use these commands when executing jobs under POWER. The Request Key on the console typewriter must be depressed before entering any of these commands.

The DISPLAY command is used to display the status of the jobs in the Queues. Some of the formats of the DISPLAY Command are:

Operation	Operand	
D	queue,jobname[,jobnumber]	Job option
D	queue[,ALL]	ALL option
D	queue,HOLD	HOLD option
D	queue,priority	Priority option
D	queue,class	Class option

Queue: specifies the queue to which the display command is directed in the xxyyy format, where

xx = BG
F2

yyy = RDR
PRT
PUN

jobname: specifies for the job option, the name by which the job entry is known to the POWER system. Jobname may be from one to eight alphameric characters. Each job entry name is qualified by a job number assigned to it by the POWER system. Therefore, if the possibility of duplicate jobnames exists, the jobnumber should be specified also. If a duplicate jobname exists and jobnumber is not specified, the first job entry found with the specified jobname is displayed.

jobnumber: specifies, for the job option, the number assigned to the job entry by the POWER system when the job entry is logged in the input queue. This number is also used with output queue entries for this job. Jobnumber may be specified with from one to five numeric characters.

ALL: specifies, for the ALL option, a general request for status information for all job entries in the specified queue.

HOLD: specifies for the HOLD option, a general request for the status of all job entries in the hold state in the specified queue.

priority: specifies, for the priority option, a general request for the status of all job entries in the queue with the specified priority. Priority is specified by Pn, where n is a numeric character from 0 through 9. Nine is the highest priority. The parameter is not dependent upon the PRIORTY generation parameter.

class: specifies for the class option, a general request for the status of all job entries in the queue with the specified class. Class is specified by CLASSn where n is any alphabetic character from A through Z.

The display command provides the operator with a status report on the console typewriter (SYSLOG) of the job entry or job entries required by the operand. If there are no job entries in the queue, the display command will report Q is EMPTY.

Figure 15.7 shows the format of the DISPLAY command that is used to display the status of all the jobs in the INPUT Queue and the report that would print on SYSLOG.

The report is analyzed in the following way.

1. The first item in the report is the job name.
2. The second item is the job number.
3. The third item is the priority for the job.
4. The fourth is the disposition of the job entry. If this item is blank, the job is not in the hold state. If this item is H, the job is in the hold state. If this item is * the job is being processed (printed, punched, executed).
5. The fifth item is the class for the job. The class may be any alphabetic character or blank if omitted.
6. The sixth item is the number of records in the entry.

It can be seen from the report that job ASSEMBLY is currently executing in storage. By looking at the priorities of the jobs we can assume the job LIST has already executed because it has a higher priority then the job that is now executing. We can also assume the job SORT has executed because when more then one job has the same priority the job with the lowest job number will execute first.

To further analyze the POWER system, it is necessary to obtain the status of the job in the PRINT and PUNCH Queues.

Write the command to display the status of the jobs in the PRINT Queue.

Figure 15.8 shows the command and the report that would print on SYSLOG. We can see from this report that job SORT is waiting to print while job LIST is now printing.

The one remaining Queue that we need to analyze is the PUNCH Queue. Figure 15.9 shows the command and report that would print on SYSLOG. Although there are output records in the PUNCH QUEUE, they are for the job that is currently executing. The report includes only entries for jobs that have completed executing.

We now have enough information about the status of the various jobs in the Queues to be able to run a job and have it execute immediately after job ASSEMBLY. The only thing required is the JECL statements.

Job ASSEMBLY could complete processing and the next job could start while you are preparing the JECL statements. This will cause more delay before you could execute your job. A means of preventing jobs from starting to execute is to issue a HOLD command. The hold command is used to prevent the processing of one or more job entries in the specified queue. Following are the various formats of the HOLD command.

Operation	Operand	
H	queue,jobname[,jobnumber]	Job option
H	queue[,ALL]	ALL option
H	queue,priority	priority option

queue: specifies the queue to which the hold command is directed in the xxyyy format, where

xx = BG
F2

yyy = RDR
PRT
PUN

jobname: specifies, for the job option, the name by which the job entry is known to the POWER system. Jobname may be from one to eight alphameric characters. Each job entry name is qualified by a job number assigned to it by the POWER system. Therefore, if the possibility of duplicate jobnames exists, the jobnumber should be specified also. If a duplicate jobname exists and the jobnumber is not specified, the first job entry found with the specified jobname is placed in the hold state.

jobnumber: specifies the number assigned to the job entry by the POWER system when the job entry is logged in the input queue. This number is also used with output queue entries for this job. The display command may be used to inform the operator of the correct specification for jobnumber. Jobnumber may be specified with from one to five numeric characters.

ALL: specifies, for the ALL option, that all job entries in the specified queue are to be placed in the hold state.

priority: specifies for the priority option, that all job entries in the specified queue with the specified priority are to be placed in the hold state. Priority is specified by Pn where n is a numeric character from 0 through 9. Nine is the highest priority. If PRIORITY=NO was generated as a POWER option, jobname is assumed.

The best way to be sure that no more jobs start to execute is to issue the following HOLD command:

H BGRDR,ALL

The results of this command is shown in Figure 15.10.

1. All the jobs in the INPUT Queue are placed in the HOLD state. Job ASSEMBLY continues to execute.

2. You can now prepare your POWER JECL statements and submit them to the POWER system. The job entry will be placed in the INPUT Queue and will execute after job ASSEMBLY completes.

3. Depending on the job you want to run, you may want to place the jobs that are in the PRINT and PUNCH Queues in the HOLD state, also using the following commands:

```
H BGPRT,ALL
H BGPUN,ALL
```

After your job begins to execute, the next thing you must do is to release the jobs that were in the INPUT QUEUE, from the HOLD status.

The release command is used to remove a job entry or job entries from the hold state. If there are no jobs in the queue, the command processor will type the message Q IS EMPTY on the console typewriter (SYSLOG).

The format of the RELEASE Command is:

Operation	Operand	
R	queue,jobname[,jobnumber]	Job option
R	queue[,ALL]	ALL option
R	queue,priority	priority option

queue: specifies the queue to which the release command is directed in the xxyyy format, where

xx = BG
F2

yyy = RDR
PRT
PUN

jobname: specifies, for the job option, the name by which the job entry is known to the POWER system. Jobname may be from one to eight alphameric characters. Each job entry name is qualified by a job number assigned to it by the POWER system. Therefore, if the possibility of duplicate jobnames exists, the jobnumber should be specified also. If a duplicate jobname exists and the jobnumber is not specified, the first job entry found in the hold state with the specified jobname is released.

jobnumber: specifies the number assigned to the job entry by the POWER system when the job entry is logged in the input queue. This number is also used with output queue entries for this job. The display command may be used to inform the operator of the correct specification for job number. Jobnumber may be specified with from one to five numeric characters.

ALL: specifies, for the ALL option, that all job entries in the specified queue are removed from the hold state.

priority: specifies for the priority option that all job entries in the specified queue with the specified priority are to be removed from the hold state. Priority is specified by Pn where n is a numeric character from 0 through 9. Nine is the highest priority. If PRIORTY-NO was generated as a POWER option, jobname is assumed.

Write the command you would issue to release the jobs in the INPUT QUEUE from the HOLD state.

Figure 15.11 shows the correct command and the result of executing it.

1. The job CE is executing in storage and spooling print records to disk.

2. This command will release all the jobs in the INPUT Queue from the HOLD state.

3. The INPUT Queue returns to its original status. Job STOCK will execute after job CE completes (see NOTE).

4. The printer now contains the completed output for job SAMPLE and job LIST.

   NOTE: JOB LIST, JOB SORT, and JOB ASSEMBLY have already been executed (outputs are in the Print and Punch Queues) and therefore JOB STOCK has the next highest priority.

The operation continues as shown in Figure 15.12.

1. When job CE completes executing, job STOCK is loaded into core and starts executing, spooling output to the PRINT Queue and PUNCH Queue.

2. The PRINT Task begins to print the output for job CE from the PRINT Queue.

3. Now that the output for your job is printing, you can release the jobs in the PRINT and PUNCH Queues.

At this point the POWER system is returned to its original status.

Review

Let's briefly review the procedure to submit a job to the POWER system and have it execute immediately.

(1) Determine the status of the jobs in INPUT, PRINT and PUNCH Queues. Review Figures 15.7, 15.8 and 15.9.
(2) Prepare the POWER JECL statements required to execute a job.
(3) Suspend the execution of any more jobs. Review Figure 15.10.
(4) Submit your job to the POWER system. Review Figure 15.10.
(5) When your job starts to execute, release the jobs in the INPUT Queue from the HOLD state. Review Figure 15.11.
(6) When your job output is printing, you must release the jobs in the PRINT and PUNCH Queue. Review Figure 15.12.

During the follow-on-lab, you will have an opportunity to operate a POWER system.

Answer the following study questions and when you have completed them, input your answers via the terminal.

Study Questions (Section 15-3)

1. Which one of the following commands would you use to determine the status of the jobs in the F2 INPUT QUEUE?

   a. DISPLAY F2RDR,ALL
   b. D F2RDR,ALL
   c. D F2RDR,QUEUE
   d. D F2RDR,H

2. Which one of the following commands would you use to prevent any more jobs with a priority of 5 from executing in the BG partition?

   a. H BGRDR,5
   b. P BGRDR,5
   c. H BGPRT,5
   d. H BGPUN,5

3. Which one of the following commands would you use to release all the jobs in the BG print Queue from the hold state?

   a. R BGPRT,5
   b. R BGPUN,ALL
   c. R ALL,BGPRT
   d. R BGPRT,ALL

SR25-5640-3

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
[U.S.A. only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

DOS Maintenance Facilities Student Self-Study Course Printed in U.S.A. SR25-5640-3