File No. S360-36
Form C27-6939-0

OS

IBM

Systems Reference Library

IBM System/360 Operating System Planning for Multiprogramming With a Fixed Number of Tasks Version II (MFT II)

This publication provides information concerning Version II of Multiprogramming With a Fixed Number of Tasks (MFT II) for installation personnel who are responsible for selection, evaluation, and implementation of System/360 Operating System configurations. The information is presented in three major categories: Concepts, Considerations, and Characteristics. These sections describe, respectively:

- The principles of operation of MFT II
- How they influence application and operation of the system
- Detailed specifications of storage requirements, system generation, and operation of an MFT II system

The descriptive information is supplemented by examples and illustrations including a sample job scheduling sequence and sample partition configurations for systems with 128K, 256K, and 512K bytes of main storage.

PREFACE

This publication provides the answers to four basic questions:

1. What is MFT II?

2. What does MFT II offer the Operating System user?

3. What should the user do to prepare for effective use of MFT II?

4. What is necessary to specify and operate an MFT II system?

The Concepts section provides information to answer questions 1 and 2, through describing the functional components of MFT II, and how they operate in relation to each other. The Considerations section provides information to answer question 3, by describing how various characteristics of the operation of MFT II may influence installation conventions and machine room procedures. The Characteristics section answers question 4, by providing detailed specifications and descriptions of storage requirements, system generation, and operation of an MFT II system.

This publication is intended for the experienced Operating System user who has a working knowledge of the Primary Control Program (PCP), and at least a familiarity with the characteristics of the existing version of MFT. The user who is not familiar with the existing version of MFT should read the publication IBM System/360 Operating System: Multiprogramming With a Fixed Number of Tasks; Concepts and Considerations, Form C27-6926.

Readers who are not experienced or familiar with PCP or the existing MFT will find maximum benefit from this publication only if they are familiar with the contents of IBM System/360 Operating System: Introduction, Form C28-6534 and IBM System/360 Operating System: Concepts and Facilities, Form C28-6535.

Other publications that the reader will find helpful, and that are referred to within this publication are:

IBM System/360 Operating System: Job Control Language, Form C28-6539
IBM System/360 Operating System: Messages, Completion Codes, and Storage Dumps, Form C28-6631
IBM System/360 Operating System: Operator's Guide, Form C28-6540
IBM System/360 Operating System: Storage Estimates, Form C28-6551
IBM System/360 Operating System: System Generation, Form C28-6554
IBM System/360 Operating System: System Programmer's Guide, Form C28-6550
IBM System/360 Operating System: Supervisor and Data Management Services, Form C28-6646

First Edition (October 1967)

Specifications contained herein are subject to change from time to time. Any such change will be reported in subsequent revisions or in Technical Newsletters.

This publication was prepared for production using an IBM computer to update the text and to control the page and line format. Page impressions for photo-offset printing were obtained from an IBM 1403 Printer using a special print chain.

Copies of this and other IBM publications can be obtained through IBM Branch Offices.

A form for reader's comments appears at the back of this publication. Address any additional comments concerning the contents of this publication to: IBM Corporation, Programming Publications, Department 637, Neighborhood Road, Kingston, New York 12401

CONTENTS (continued)

ILLUSTRATIONS

This publication contains information to aid the potential user of Multiprogramming With a Fixed Number of Tasks, Version II (MFT II) in planning for its use, operation, and installation. This information is presented in the three sections that follow this Introduction.

The Concepts section is intended for data processing executives responsible for selection of a system, and planners and system analysts who must understand its operation to plan for its efficient use, and establish installation procedures. To accomplish this, the Concepts section presents a sample sequence of operation describing the initiation, execution, and termination of a set of jobs of various categories and priorities. This sequence of operation is followed by a description of the principles of operation of the MFT II system, describing the operation of each major component of the system.

The Considerations section contains information of interest to data processing executives, planners, and system analysts. It also describes MFT II considerations of interest to system programmers who will establish programming conventions for their installations, and machine room supervisors and operators who will establish operating procedures.

Information in the Considerations section is presented in seven major topics.

- General Considerations, which apply to all types of jobs to be run under control of MFT II
- Batch consideration, which apply to "batched" production jobs such as compilation, file maintenance, and report generation
- Telecommunications considerations, which apply to telecommunications message processing and message switching jobs under MFT II
- Graphics considerations, which apply to jobs which involve the IBM 2250 or 2260 Graphic Displays
- CPO considerations, which apply to Concurrent Peripheral Operations under MFT II
- Operating Considerations, which outline characteristics or actions of the system of special interest to the machine operator
- Typical storage configurations

The Characteristics section provides detailed descriptions of how to specify generation of an MFT II system, additions to the job control language related to MFT II, and all operator commands and new operator messages associated with MFT II. This information should be read by system programmers, application programmers, and machine operators.

TERMINOLOGY

New terminology associated with MFT II is explained as it is presented. Certain basic definitions, however, are essential to understanding these new terms as they are introduced. These basic definitions are given in the following paragraphs.

MULTIPROGRAMMING WITH A FIXED NUMBER OF TASKS

"Multiprogramming" refers to the concurrent execution of several units of work (tasks), any one of which would, in a single-program environment, occupy the computing system until the task was finished.

Note: Throughout this publication, "job" refers to an externally specified unit of work (a problem program specified by a JOB card), and "task" refers to any unit of work that must be performed by the Central Processing Unit (CPU), under control of a task control block.

The significance of multiprogramming is that it provides increased throughput and better utilization of resources. A typical task makes use of only a small part of the resources available in the system. In a single-program environment, this means that overall application of resources is low. In a multiprogramming environment, however, resource application is markedly improved, because the relatively limited demands of each of several tasks combine to produce a net demand that is more efficient in terms of the system's capabilities.

The phrase "a fixed number of tasks" indicates that the user determines at system generation how many tasks the system is capable of performing at one time. The number of tasks that can be performed at

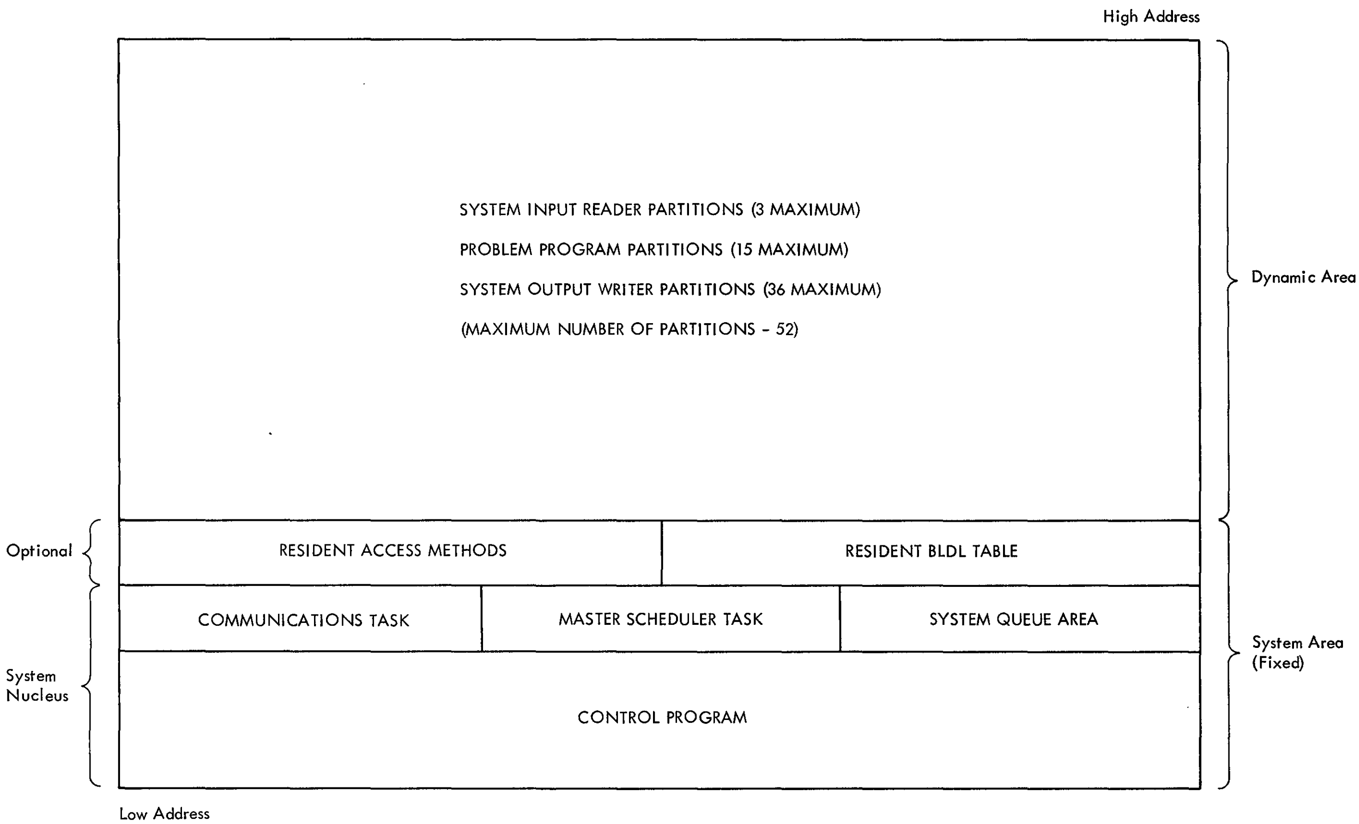

Figure 1. Main Storage Organization

one time can be varied during and after system initialization.

SYSTEM INITIALIZATION

System initialization is the preparation for execution of those elements of the IBM System/360 Operating System that reside in the fixed area of main storage. This preparation is performed by the Nucleus Initialization Program when the system is brought into main storage through the initial program loading (IPL) procedure, and is supplemented by operator action.

PARTITIONS

Multiprogramming is made possible through the establishment of discrete areas of main storage called partitions. Main storage is divided into a system area and a dynamic area (see Figure 1). The dynamic area is further defined, by the user, as consisting of a number of distinct partitions. The number of partitions defined determines the number of tasks that can proceed concurrently, because each task occupies a separate partition.

Partitions can be defined as reader, writer, or problem program partitions. Each partition has a fixed priority within the system (Partition P0 has the highest priority; P51, the lowest). This determines which partition will gain control of the CPU first when a wait condition occurs. Small partitions are problem program partitions that are too small to contain the scheduler. (See "Small Partitions" in the Concepts section.)

CONCURRENT OPERATION

In a multiprogramming system, tasks are performed concurrently. It is important to understand this concept. Execution is not simultaneous, or overlapped, or alternating in a fixed pattern. Each task is contained within a partition. The determination of which task gains control is based on "waits" and "posts". Waiting for an event, such as the completion of an input/output operation, removes the task from contention for control. Posting of the task, which signals that the awaited event has completed, causes the task to be placed in a "ready" status. The task that becomes active is the highest-priority task (highest-priority partition) that is ready. This high-priority task proceeds until another event causes the task to relinquish control. The relinquishing of control by one task, and another task's receipt of control, is called a <u>task switch</u>.

TASK SWITCHING

There are two ways in which a task switch can occur:

1. The active task relinquishes control because it must wait for the completion of an event, such as an input/output operation.
2. Control is seized by the highest priority ready task as a result of an interruption signaling an event for which it was waiting.

The first case illustrates how multiprogramming ensures optimum utilization of resources. Whenever one unit of work cannot proceed, another (highest-priority) is advanced. In a single-program environment, no work can proceed while the single task waits for an event. The second case illustrates how an internal balance between the tasks is achieved. Once a task has control, it retains control only until a task of higher priority becomes ready to proceed.

<u>Note:</u> Throughout this manual, the suffix "K" denotes the value 400(hexadecimal), or 1024(decimal).

MFT II is a System/360 Operating System option that provides extended multiprogramming capabilities and increased flexibility to the Operating System user whose system has 128K bytes or more main storage. The system may reside on any direct-access device except the IBM 2302 Disk Storage Unit. As many as 15 multi-step jobs can be processed concurrently with the operation of up to three system input readers, and 36 system output writers (the maximum number of partitions is 52).

Each job is processed in a discrete area of main storage known as a partition. When a job must wait for completion of an event such as an input/output operation, another job of lower priority is allowed to proceed. When the higher priority job is ready to resume, the lower priority job's processing is suspended and control of the central processing unit (CPU) is returned to the higher priority job. The priority of each job is determined by the partition in which it resides. Jobs are directed to a given partition or group of partitions through the CLASS parameter of the JOB card. This parameter has been added to the job control language (JCL) for use in MFT II.

By using the CLASS parameter to denote different types of jobs, the user can direct jobs to partitions consistent with the jobs' characteristics. Process-limited jobs, for instance, can be directed to low priority partitions so that they do not interfere with efficient processing of jobs that do not require the CPU as often. Telecommunications jobs can be directed to higher priority partitions so that system response time to the terminal user is minimal. Additional applications of the CLASS parameter can be established based on any job characteristics meaningful to the installation.

To use MFT II efficiently, both system and application programmers must understand how it operates. Because other user personnel may be interested in a summary of MFT II operation, without recourse to logic descriptions, this section of the publication contains three major topics:

- "Features and Facilities" lists the functional capabilities of MFT II and describes each briefly.
- "Sequence of Operation" describes the scheduling, initiating, and terminating of a series of jobs.
- "Principles of Operation" describes in detail how each functional component of MFT II operates, and what it does.

FEATURES AND FACILITIES

MFT II makes possible the concurrent execution of up to 15 separate jobs within a single computing system having only one central processor, while continuing to provide all other applicable services of the IBM System/360 Operating System. Other features of MFT II include:

- Independent job scheduling
- Small partitions (smaller than the size of the scheduler)
- Redefinition of partition sizes and characteristics during operation
- Resident and transient system input readers
- Reading of an input stream from IBM 2311 or 2314 disk storage
- Resident system output writers and writers that operate in problem program partitions
- Restarting the system without losing enqueued jobs

Each feature is described briefly in the following paragraphs and explained in detail under the headings "Principles of Operation" and Characteristics.

EXTENDED MULTIPROGRAMMING CAPABILITIES

MFT II extends to 15 the number of jobs that may operate concurrently in the system. Jobs are scheduled into partitions through use of the CLASS parameter on the JOB card in conjunction with the PRTY parameter. The CLASS parameter has been added for MFT II. With storage protection, each of these jobs is protected from damage by other jobs, and the system areas are protected from all problem programs.

INDEPENDENT JOB SCHEDULING

All partitions are independent with respect to job scheduling and initiation. The operator intervention and job sequencing requirements imposed by the WAITR macro instruction and the SHIFT command in the existing MFT, are eliminated. In MFT II, WAITR is treated as a WAIT macro instruction. Jobs are scheduled into the first available problem program partition that services the corresponding job class, according to the PRTY parameter on their JOB cards.

SMALL PARTITIONS

Small problem program partitions can vary in size from 8K bytes to the size of the scheduler selected for the system (26K or 44K). Small partition scheduling permits jobs to be scheduled into partitions smaller than the size of the scheduler by a partition of scheduler size whenever the large partition is available for scheduling purposes. This can take place at initiation and termination of jobs.

The user may define as many as 14 small problem program partitions. Efficient use of small partitions is achieved through associating one or more job classes with them. Thus, jobs whose main storage requirements are small can be directed to the small partitions for execution. Small high-priority jobs can be accomodated by defining a small partition in the upper portion of main storage. This partition might then be assigned a primary job class reserved by the installation for critical jobs. By assigning secondary and tertiary job classes, the partition need not remain idle when there is no critical work. By defining a small partition, only the necessary amount of main storage is set aside for such work.

DYNAMIC PARTITION DEFINITION

Dynamic partition definition allows the user to reconfigure main storage during operation, provided that the partitions to be redefined are contiguous and quiescent. That is, jobs in the affected partitions have been terminated either by a CANCEL command or by the system depending on the type of job. The number of partitions in the system can be decreased or increased within the limits established at system generation (SYSGEN). Job classes assigned to the partitions and partition sizes can be changed also.

SYSTEM INPUT READERS

MFT II provides up to three system input readers which can operate in problem program partitions of scheduler size as <u>transient</u> readers, or in reader partitions as <u>resident</u> readers. Readers accept job control statements and data in the input stream, including multiple data sets for the same job step, and transcribe that data onto a direct-access device for retrieval by the problem program. When a resident reader is specified, input records can be blocked.

INPUT STREAM FROM DISK

MFT II allows the user to establish a disk storage drive (either the IBM 2311 Disk Storage unit or the IBM 2314 Direct Access Storage Facility) as a system input device. Data in the input stream is permitted, including multiple data sets for the same job step, providing the facility for:

1. Reading input from sequentially organized data sets

2. Deblocking of blocked input records (resident readers only)

3. Automatically switching volumes if end-of-volume is detected on a data set extending across volumes, or concatenated data sets are being processed

4. Starting more than one reader for the same disk storage unit

SYSTEM OUTPUT WRITERS

System output writers are IBM-supplied or user-written programs that retrieve problem program output from temporary direct-access storage, and transcribe it onto the device specified by the problem programmer. The device is specified by the SYSOUT parameter on the DD statement describing the output data set. The temporary data sets may be written on any direct access storage device except the IBM 2302 Disk Storage unit; i.e., the valid devices are: IBM 2311 Disk Storage drive, IBM 2314 Direct Access Storage Facility, and IBM 2301 and 2303 Drum Storage units.

Up to 36 system output writers can be established by the user. Each operates independently in its own partition concurrent with the execution of other system tasks and of problem programs. MFT II provides the same output writer services as MVT as described in <u>IBM System/360 Operating System: System Programmer's Guide</u>, Form C28-6550. Each system output writer can handle as many as eight different output classes; more than one writer can service the same output class. The user may start as many as 15 non-resident writers if the total number of resident and non-resident writers does not exceed 36. Non-resident writers operate in the same way as resident

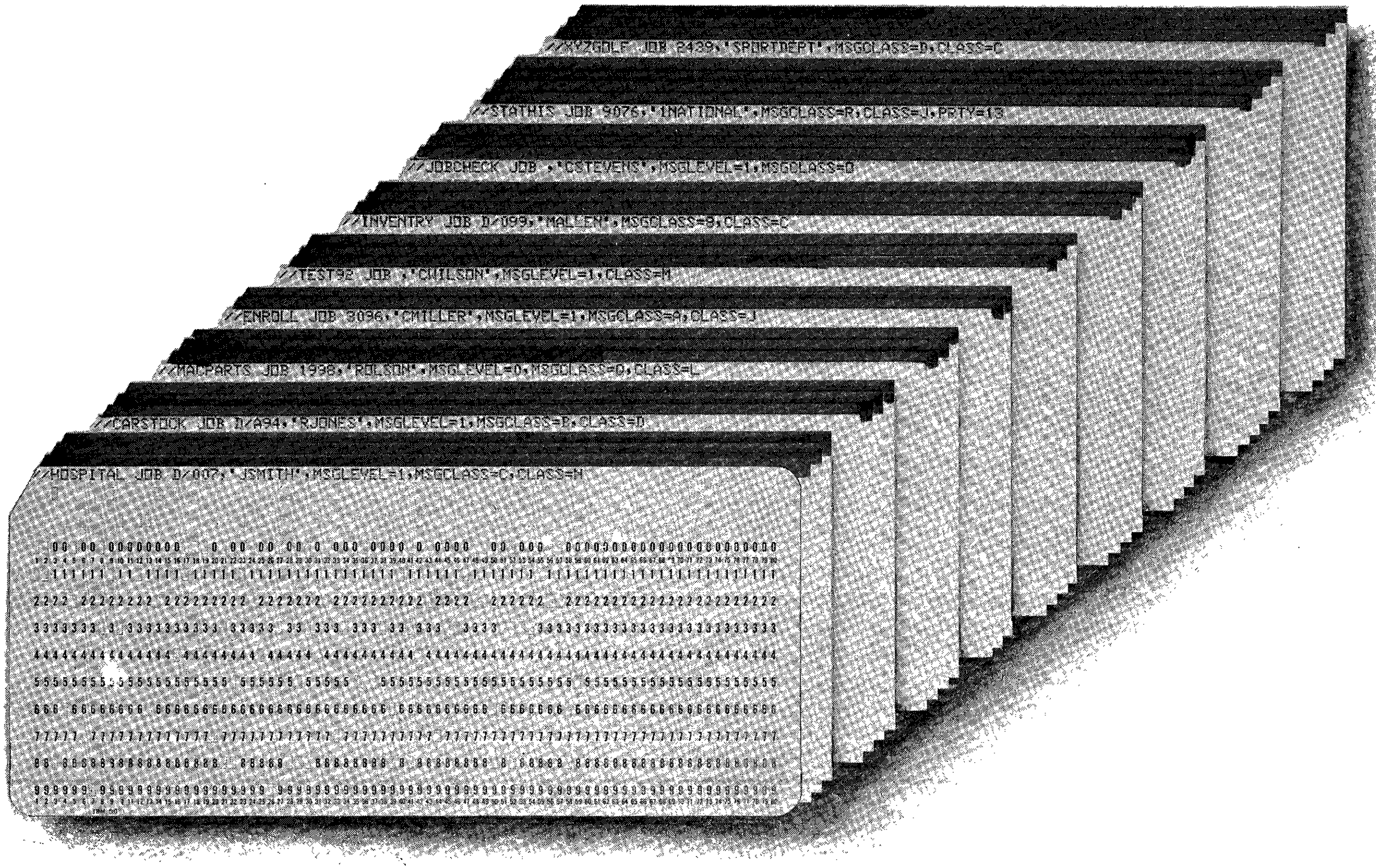

Figure 2. Input Job Stream

Figure 3. Contents of Main Storage After Nucleus Initialization

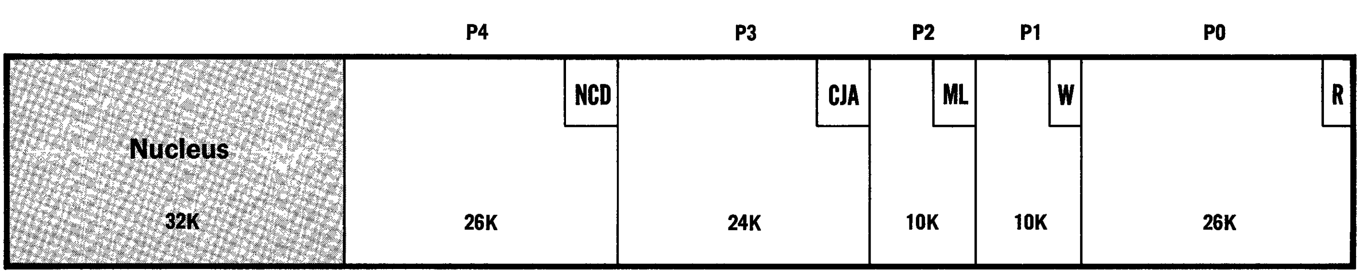

Figure 4. Contents of Main Storage After System Initialization

writers, except that they are executed in problem program partitions.

SYSTEM RESTART

It is sometimes necessary to shut down the system while needed information still exists on the job queue data set. Such occasions might be end-of-shift, end-of-day, scheduled maintenance, system malfunction, or power failure. System restart permits all enqueued input and output jobs which had been entered in the job queue to remain there for subsequent retrieval by the system. Jobs that were being processed at the time of shutdown must be restarted.

SEQUENCE OF OPERATION

To illustrate the concepts of MFT II, a sample sequence of operation is described below. In the job stream shown in Figure 2, the following CLASS parameters appear on the JOB cards:

1. CLASS=N
2. CLASS=D
3. CLASS=L
4. CLASS=J
5. CLASS=M
6. CLASS=C
7. None
8. CLASS=J,PRTY=13
9. CLASS=C

The system is loaded by use of the normal IPL procedure and initializes itself by use of the nucleus initialization program (Figure 3). After system initialization, the contents of main storage are as shown in Figure 4. Figure 5 illustrates the input work queues for the seven classes

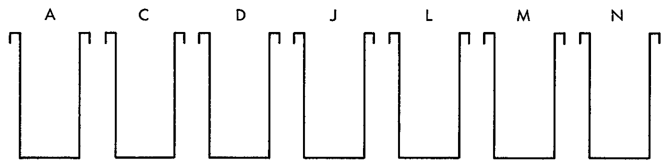

Figure 5. Input Work Queues After System Initialization

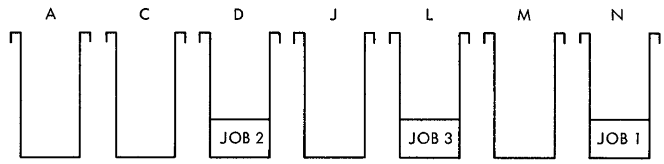

Figure 6. Input Work Queues After First Three Jobs Have Been Entered

used in this example. When a START command is entered for P0, the reader begins reading the job stream and entering jobs into the input work queues for each CLASS (Figure 6). The START commands for the initiator and writer are also entered (Figure 7).

The scheduler in P4 now schedules the first job. Because the scheduler searches the CLASS=N queue first (and job 1 has been placed on the queue by this time), job 1 is initiated and given control (Figure 8). The reader continues reading the input stream, placing jobs in their appropriate queues. Because job 7 has no CLASS parameter, it is placed on input work queue A (the default job class). Job 8, with a PRTY parameter is placed on input queue J ahead of job 4, unless job 4 has already been scheduled. At this point jobs 1 through 9 have been read and placed on their appropriate queues (Figure 9).

When job 1 has finished processing, a scheduler is brought into P4 to terminate the job, and initiate jobs in small partitions P2 and P3. Job 5 is scheduled into P2 (because M is the primary class) and job 6 into P3. The scheduler now searches input work queue N for a job for P4. Because the CLASS=N queue is empty, the

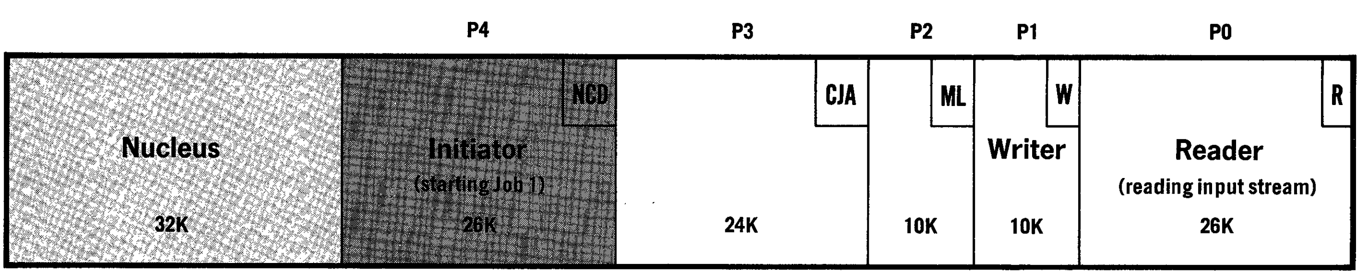

Figure 7. Contents of Main Storage After START Commands

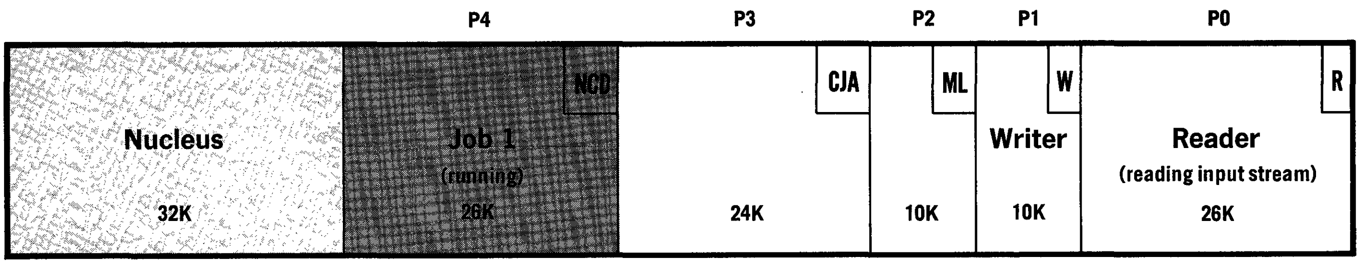

Figure 8. Contents of Main Storage After First Job Has Been Scheduled

CLASS=C queue is searched. Job 9 is waiting on the queue, so it is scheduled into P4. At this point, the contents of main storage are as shown in Figure 10.

A scheduler continues to enter P4 each time a job terminates in that partition. The scheduler first checks whether jobs in P2 or P3 need to be terminated. If so, they are terminated by the scheduler in P4. New jobs are then scheduled into P2 and P3 depending on which one has terminated first, P2 has higher-priority. This sequence continues until all jobs have completed their processing, or until the system is shut down.

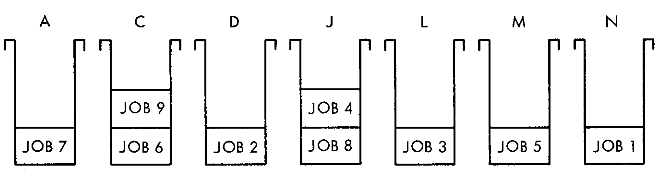

Figure 9. Input Work Queues After All Nine Jobs Have Been Entered

PRINCIPLES OF OPERATION

This topic describes the principles of operation of MFT II. Included are:

- Partition job class facility
- Priority within job class
- Main storage organization
- Partition definition
- System input readers
- Scheduling process
- System output writers
- System restart

Figure 11 illustrates a configuration referred to throughout the remaining discussion of "Principles of Operation". The alphabetic characters in the upper right corner of each partition in Figure 11 are the job classes assigned to that partition. P/P denotes problem program partition.

PARTITION JOB CLASS FACILITY

The partition job class facility allows the user to assign one or more partitions to selected jobs. During system generation, the user must assign a partition to service each job class he intends to use. These assignments may be modified later. (See "Dynamic Partition Definition" in this section.) Each problem program partition may be assigned as many as three job classes designated A through O. These job class designations have no inherent meaning. Therefore they can be used to denote any job characteristic meaningful to the user, which would influence the choice of partitions for the job. More than one partition may be assigned to the same job class(es). In Figure 11, P3 is assigned to job classes C, J, and A; P4 is assigned to N, C, and D. These partition job class identifiers are used by the system to determine which input queue is searched first. (See "Job Initiation" in this section.) The sequence in which jobs are selected from each input work queue is determined by the PRTY parameter. (See "Enqueuing Jobs by CLASS and PRTY" in this section.)

The user controls the partition(s) in which a job executes by using the CLASS parameter on the JOB card. The format of this keyword parameter is:

    CLASS=job class

where job class is the identifier (A-O) assigned to the job. If this parameter is omitted from the JOB card, a job class of A is assigned by the system. All 15 job classes may be assigned by the user, providing that at least one partition has been assigned to each of the classes specified. When the user designates a job class for a particular job (CLASS parameter), the job is executed only in a partition that has been assigned to its class. If more than one partition is assigned to that job class, the job is executed in the first

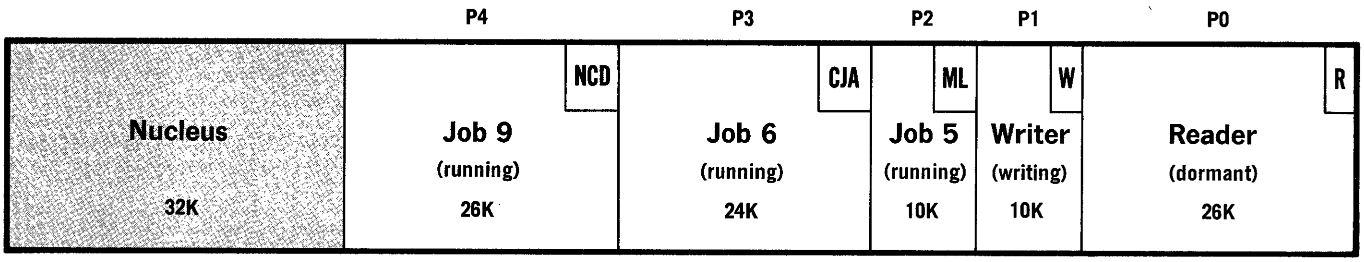

Figure 10. Contents of Main Storage With All Partitions Active

available problem program partition. A typical JOB card may be specified as follows:

```
//JOBPAY JOB 661,'JDOE',CLASS=C,PRTY=13
```

In the configuration illustrated in Figure 11, this JOB card causes the job to execute in either P3 or P4, whichever is available first.

MAIN STORAGE ORGANIZATION

MFT II permits from 1 to 15 separately scheduled jobs to reside in their own predefined sections of main storage. Main storage is composed of a fixed area called the nucleus, and a dynamic area which contains the problem program partitions, the system output writer partitions, and the system input reader partitions. Any configuration of reader (up to 3), writer (up to 36) and problem program partitions (up to 15) may be used as long as the total number of partitions does not exceed 52.

Nucleus

The MFT II system nucleus occupies a fixed area in main storage containing at least 30K bytes. As shown in Figure 12, it contains the resident portion of the control program that performs control functions during the execution of a processing program. The nucleus also contains:

- Resident QSAM in support of system input readers and output writers; (These resident QSAM routines are available as a resident access method to problem programs.)
- The communications task
- The master scheduler task
- The system queue area (SQA)

Control program routines are of three functional types: job management, task management, and data management. Job management provides communication between the user and the operating system by analyzing the input stream and collecting information needed to prepare a job for execution, and by analyzing operator commands. The resident portions of job management are the communications task, and the master scheduler task (described below).

Resident task management routines are: all routines that perform interruption handling, main storage supervision, and time supervision, and some routines that perform task supervision, contents supervision, and overlay supervision.

Data management controls all operations associated with input/output devices, such as allocation of space on volumes, storing, naming, and cataloging data sets, and movement of data between main and auxiliary storage. The resident portion of data management is the input/output supervisor and QSAM for readers and writers. Optionally, other access method routines may be made resident. The 30K MFT II nucleus also includes space for the optional storage protection routines. Although the storage

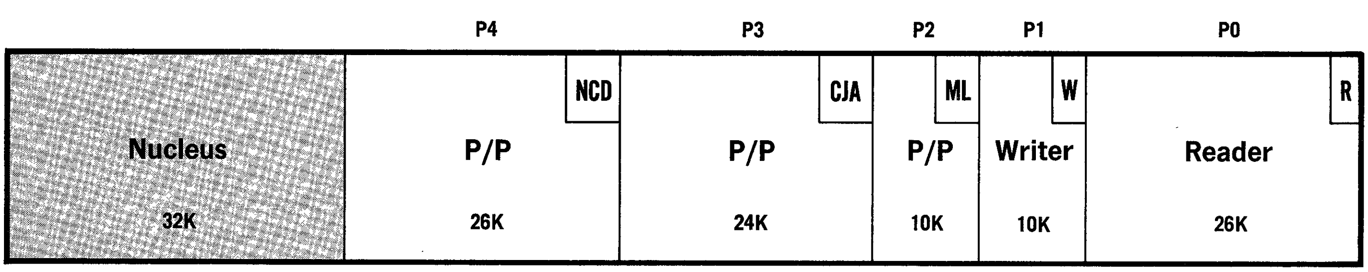

Figure 11. Sample Five-Partition Configuration

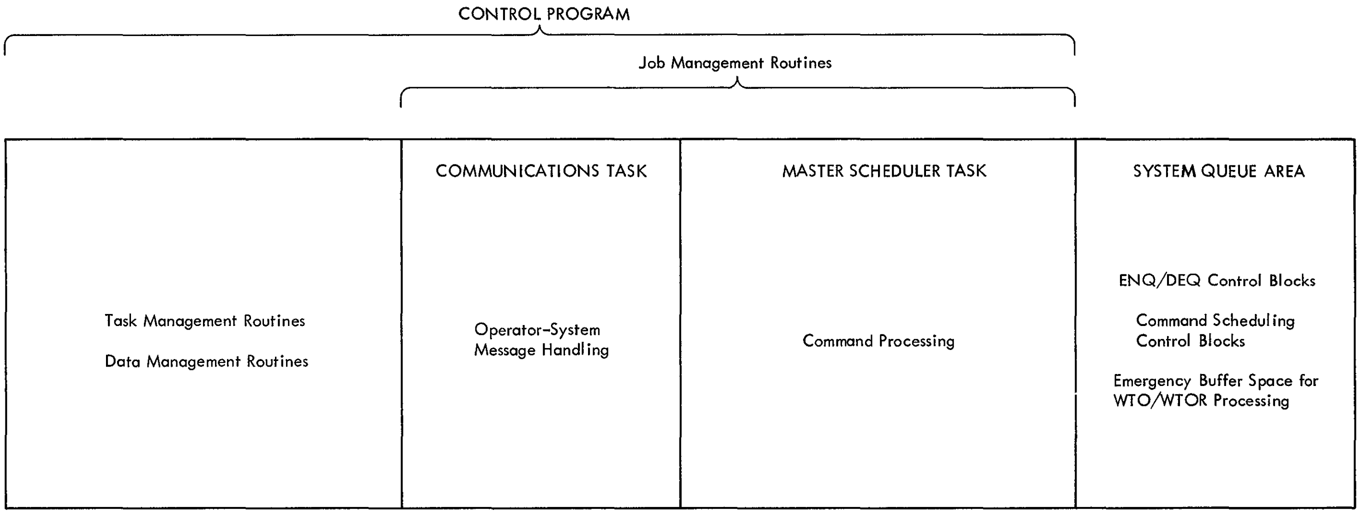

Figure 12. Components of the Nucleus

protection feature is not required for MFT II, it should be included. Storage protection prevents the contents of each partition from being destroyed or changed by another task.

Communications Task: The communications task handles the following types of communication between the operator and the system:

- Operator commands, issued through the console
- Write-to-operator (WTO) and write-to-operator with reply (WTOR) macro instructions

Master Scheduler Task: The master scheduler task handles job queue manipulation commands and dynamic partition definition. For example, a DEFINE or HOLD command is processed by the master scheduler task.

System Queue Area: The system queue area (SQA) is a protected area in the nucleus that must contain at least 1600 bytes for a minimum two-partition system. An additional 800 bytes must be added for each additional partition specified at system generation. SQA contains ENQ/DEQ control blocks and command scheduling control blocks (CSCBs). In addition, if the communications task cannot obtain WTO buffer space, SQA is used.

System Input Reader Partitions

MFT II allows the user to specify as many as three resident reader partitions. Two readers are available: a 26K reader, and a 44K reader. The reader used in the system is determined by the size of the job scheduler specified at system generation. (See "Choosing the Size of the Scheduler" in the Considerations section.)

A resident reader is established by assigning job class R to a partition. No other task can operate in this partition unless the partition's job class is changed. (See "Dynamic Partition Definition" in this section.) It is unnecessary to establish any resident reader partitions if the user plans to use transient readers. (See "Input Readers" in this section.)

Problem Program Partitions

MFT II permits the user to specify up to 15 partitions for problem programs. Each partition may have up to three job class identifiers; more than one partition may be assigned the same job class(es). Problem program partitions must be at least 8K in size. At least one of the partitions must be the size of the scheduler (26K or 44K). This partition should not be used for processing a long-duration or unending job such as telecommunications if it is the only large partition in the system.

Large problem program partitions may also be used by the scheduler (if it has work to do), by a transient input reader, or by a system-assigned output writer.

Problem programs run concurrently with system readers and writers. When a problem program is terminated in a large partition, an initiator is brought into the partition to initiate or terminate a small partition, to start a reader or writer in the partition, or to retrieve another job from the input work queue for the appropriate job class(es). Control is then given to the appropriate task; e.g., if a problem program is retrieved from an input queue, control is given to the program for execution.

For example, in Figure 11, P4 is assigned job classes N, C, and D. If a job of class N has just been terminated, and no small partition has requested scheduling, the initiator first searches the job class N input work queue. If no class N jobs exist, the initiator searches for job class C jobs; if no class C jobs exist, the job class D input queue is searched. If all three queues are empty, the partition remains dormant until it is requested by a transient reader, until it is used by the initiator to schedule a small partition, or until another job with class N, C, or D is read into the system and scheduled.

When the job in a small partition has finished executing, the small partition remains dormant until a large partition is free to terminate it.

System Output Writer Partitions

A resident writer partition may be as small as 10K plus the size of the input buffers (see "Output Writer Partition Size Requirement" in the Characteristics section). The size of the writer depends upon the output buffer space necessary; i.e., if blocked output is used, the size of the partition must be increased by the logical record length times the blocking factor. MFT II provides the capability of running as many as 36 system output writers concurrently with problem programs and system input readers.

A resident writer partition is established by assigning job class W to it. This identifies the partition as a writer partition. As with the resident reader, no other task can operate in this partition unless the partition's job class is changed through dynamic partition definition. Each writer can accomodate as many as eight output (SYSOUT) classes and can share output classes with other writers. (See "Output Writers" in this section.)

DYNAMIC PARTITION DEFINITION

Dynamic partition definition allows the operator to change the number of partitions, their size, and their job classes at any time after initial program loading (IPL). Adjacent partitions may be combined to accomodate jobs with large storage requirements; these partitions may be reestablished subsequently (within SYSGEN limits) when the need for a large partition has passed. Job classes assigned to a partition may be changed also, to accomodate changes in the work load for one or more job classes. Reader and writer partitions may be respecified as problem program partitions and assigned to service jobs from as many as three job classes. Problem program partitions may be respecified as reader or writer partitions if needed.

Dynamic partition definition is invoked in either of two ways, depending on whether it is invoked during or after system initialization. At system initialization, the partition configuration may be changed by replying 'YES' to the message 'IEE601D CHANGE PARTITIONS?'. Alternatively, dynamic partition definition may be invoked after system initialization, by entering a new operator command, DEFINE. The format of this command is:

OPERATION	OPERAND
{DEFINE / N}	[LIST]

The optional operand LIST specifies that the current partition definitions are to be listed.

Note: The DEFINE command is not allowed in the input stream.

Partition Combination

Adjacent partitions may be combined as soon as their jobs have been terminated. If an unending job is being executed in a partition that the user wishes to combine with an adjacent partition, the unending job must first be terminated with a CANCEL command. All other partitions that are to be combined, including readers and writers, are made quiescent by the system as soon as their current tasks are completed. Any number of adjacent partitions may be combined. For example, in Figure 11, if the user wants to combine P2 and P3 into one larger partition of 34K, he may do so. However, he may not combine P2 and P4, P1 and P3, etc. When P2 and P3 are combined, the new configuration is as shown in Figure 13. P2 or P3 may be made the inactive partition. When combining P2 and P3, the user must decide which job class(es) to assign to the new partition (P2). (See "Identity Change" below.) The task control block (TCB) for the inactive partition (P3) is made nondispatchable. When the inactive partition is recovered, the TCB for P3 is made dispatchable.

With the storage protection feature attached to the system, a unique protection key is available for each problem program partition. A list is kept of each available key for subsequent reassignment to combined or recovered partitions. When

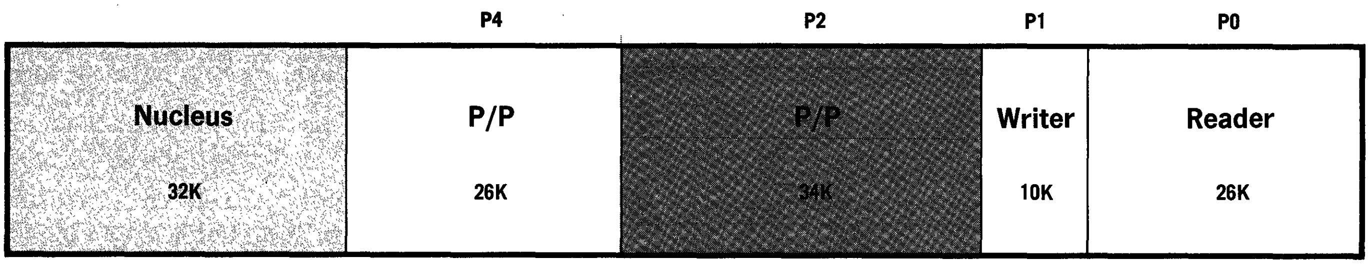

Figure 13. Partition Configuration After Combination

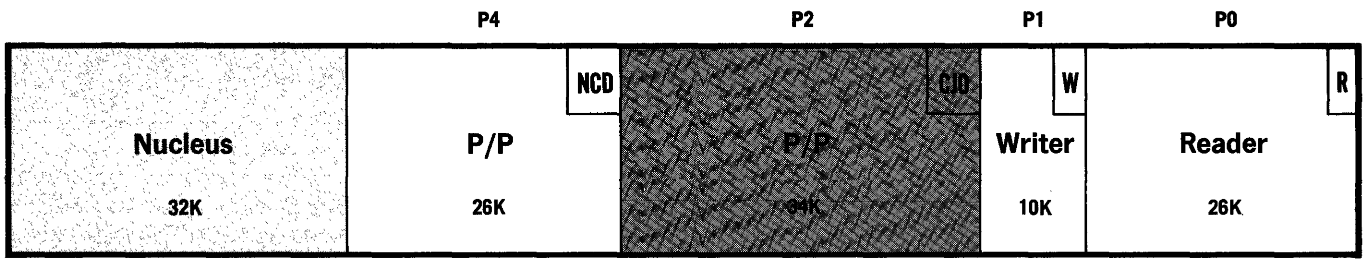

Figure 14. Partition Identification After Combination

partitions are combined or recovered, the first available protection key on the list is assigned to them.

Note: With systems including the protection feature, storage assignment increases through dynamic partition definition should be made in increments of 2K bytes. If they are not, the system rounds the value to the next 2K increment.

Identity Change

Dynamic partition definition also permits the user to change job classes specified at system generation or at system initialization. Problem program partitions may be redefined either as readers (by entering RDR) or as writers (by entering WTR). Reader and writer partitions may be changed to problem program partitions with as many as three problem program job class identifiers. When partitions are combined or recovered, the user must determine which job class(es) to assign to the resulting partitions. In Figure 13, P2 and P3 were combined into the larger partition P2. However, the original partitions each had three job classes. Therefore, the user must decide whether to choose new job classes or some combination of the six old classes. For example, P2 could be assigned job classes N, C, and J, or a new job class could be specified, such as O. A new configuration is illustrated in Figure 14.

Partition Recovery

Partitions that were combined may be reestablished, or recovered. In Figure 14, P3 is now inactive, and the user wishes to recover it. Once again he has to decide which job classes to assign to both P2 and P3. P2 and P3 need not retain their original size, nor their previous job classes. With P3 recovered, Figure 15 shows a possible new configuration. The user could have made P3 a reader partition

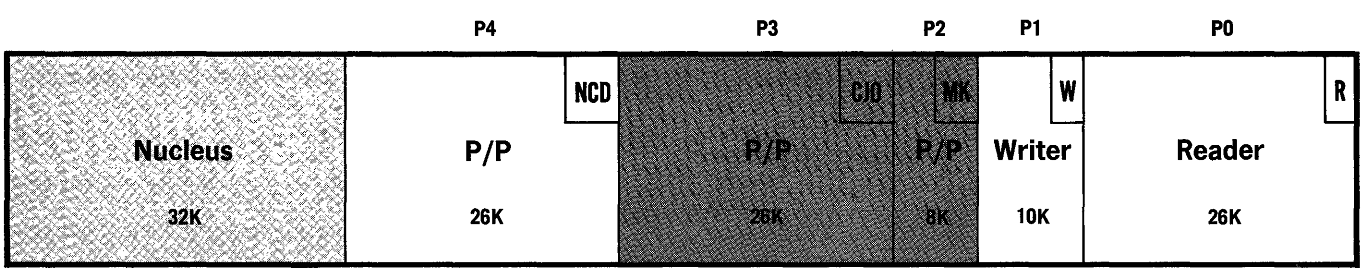

Figure 15. Partition Configuration After Recovery

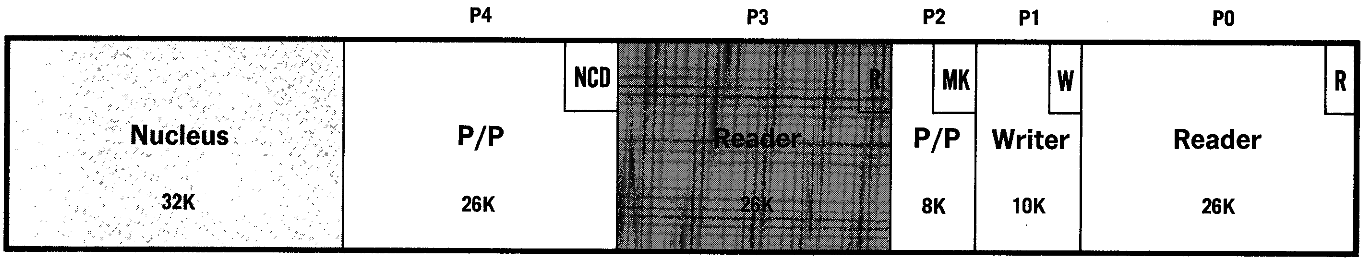

Figure 16. Partition Recovery With Identity Change

instead of a problem program partition. This configuration would be as shown in Figure 16.

When the operator enters either DEFINE or the reply 'YES' to the CHANGE PARTITIONS? message, the system requests that the definitions be entered. If 'LIST' was specified, the system lists the new partition configuration. (The operator must remember to CANCEL all affected unending jobs before redefining the system.) After definitions are entered, the system checks their validity. The system inhibits scheduling subsequent jobs into the affected partitions. When the current jobs have been terminated, the new definitions are then made. (See Figure 17.) The Considerations section contains further examples and discussion of dynamic partition definition; the Characteristics section details related operating procedures.

INPUT READERS

MFT II allows as many as three input readers to execute concurrently in any problem program partition or previously defined reader partition large enough to accommodate them. As they read and analyze input streams, readers operate concurrently with problem programs and writers. Input stream data for the step being read is transcribed onto direct-access storage where it is held until execution of the associated job begins. The problem program retrieves the data directly from the storage device. Multiple input stream data sets for the same job step are permitted. A card reader, magnetic tape unit, or disk storage unit may be specified as the input device.

A START command is entered to activate the reader. As the input stream is read, statements are analyzed as shown in Figure 18. If the statement is Job Control Language (JCL), it is scanned, and keyword and positional parameters are determined. If errors are encountered (e.g., no programmer's name, invalid continuation card, invalid parameters), control is

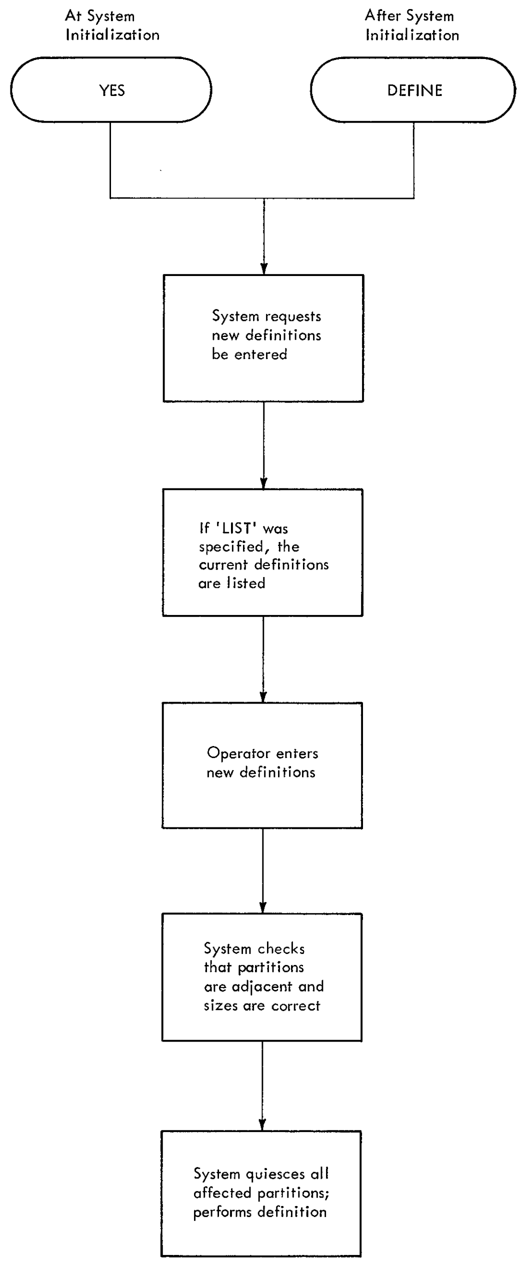

Figure 17. Partition Definition Processing

passed to a routine which prints the error messages on a SYSOUT device, and fails the job. If no error is encountered, JCL statements for the entire job are converted to control tables and entered into the input work queue corresponding to the CLASS and PRTY parameters specified on the JOB card. (See "Enqueuing Jobs by CLASS and PRTY" in this section.)

If the statement is data, it is transcribed onto direct-access storage for the problem program to retrieve. Command statements in the input stream are passed to the master scheduler for processing. When end-of-file is reached, or a STOP command is issued, control is passed to the communications task to issue the READER CLOSED message.

Resident Readers

A resident reader partition is established by assigning job class R to a partition. This identifies the partition as a reader partition; no other work can be performed in the partition.

The resident reader in Figure 11 would be started by the following START command:

```
START READER(P0),devicename
```

This reader can be stopped only when the job stream input device reaches end-of-file, or by the STOP command:

```
STOP READER,P0
```

If a resident reader is used, blocked input is permitted.

Note: "READER" and "WRITER" are used throughout the text as names of standard reader and writer cataloged procedures. Cataloged procedures are explained in IBM System/360 Operating System: Concepts and Facilities, Form C28-6535, IBM System/360 Operating System: Job Control Language, Form C28-6539, and IBM System/360 Operating System: System Programmer's Guide, Form C28-6550.

Transient Readers

A transient reader operates in a problem program partition until it reads a job for that partition's or a small partition's job class(es). At that time, the reader's work areas are saved on direct-access storage, and control is passed to the initiator to schedule the job into the partition. When the job has been terminated, and no further jobs require the partition, the reader's work areas are restored, and the reader resumes execution.

If a transient reader is started in a specific partition by including the partition assignment in the START command, it always resumes operation in that same partition when the partition becomes free. This type of transient reader is referred to as user-assigned and is started in the same manner as the resident reader.

The user may substitute 'S' for the partition assignment in the START command, allowing the system to place the reader into the first available scheduler-size problem program partition. This type of transient reader is referred to as system-assigned and is started with the following command:

```
START READER(S),devicename
```

Resident and transient readers may operate in the same system, provided no more than one system-assigned transient reader is specified, and the total number of readers does not exceed three.

Note: Blocked input is not permitted with a transient reader.

Enqueuing Jobs by CLASS and PRTY

Each job read by system input readers is converted into tables that are placed in the queue specified by the CLASS parameter. Jobs are entered into the input work queues for each class according to the PRTY parameter (PRTY values range from a low of 0 to a high of 14). One input work queue exists for each of the 15 job classes. Jobs having the same class and priority are placed in the queue first-in/first-out (FIFO). When the input work queue for a job class contains one or more jobs, termination of a job in any partition assigned to service work for that job class is followed by selection of the next highest-priority job from the input queue. Selection and initiation of the new job requires no operator intervention.

For example, if the user specifies CLASS=D,PRTY=14, the job is placed into the input work queue for job class D, behind any previously enqueued PRTY=14 jobs, but ahead of all other jobs of lower priority.

The PRTY parameter applies only to initiation priority, not to dispatching priority. Dispatching priority determines which job should be given control of the CPU. In MFT II, dispatching priority is derived from the relative position of the partitions: P0 is highest priority, P51 lowest.

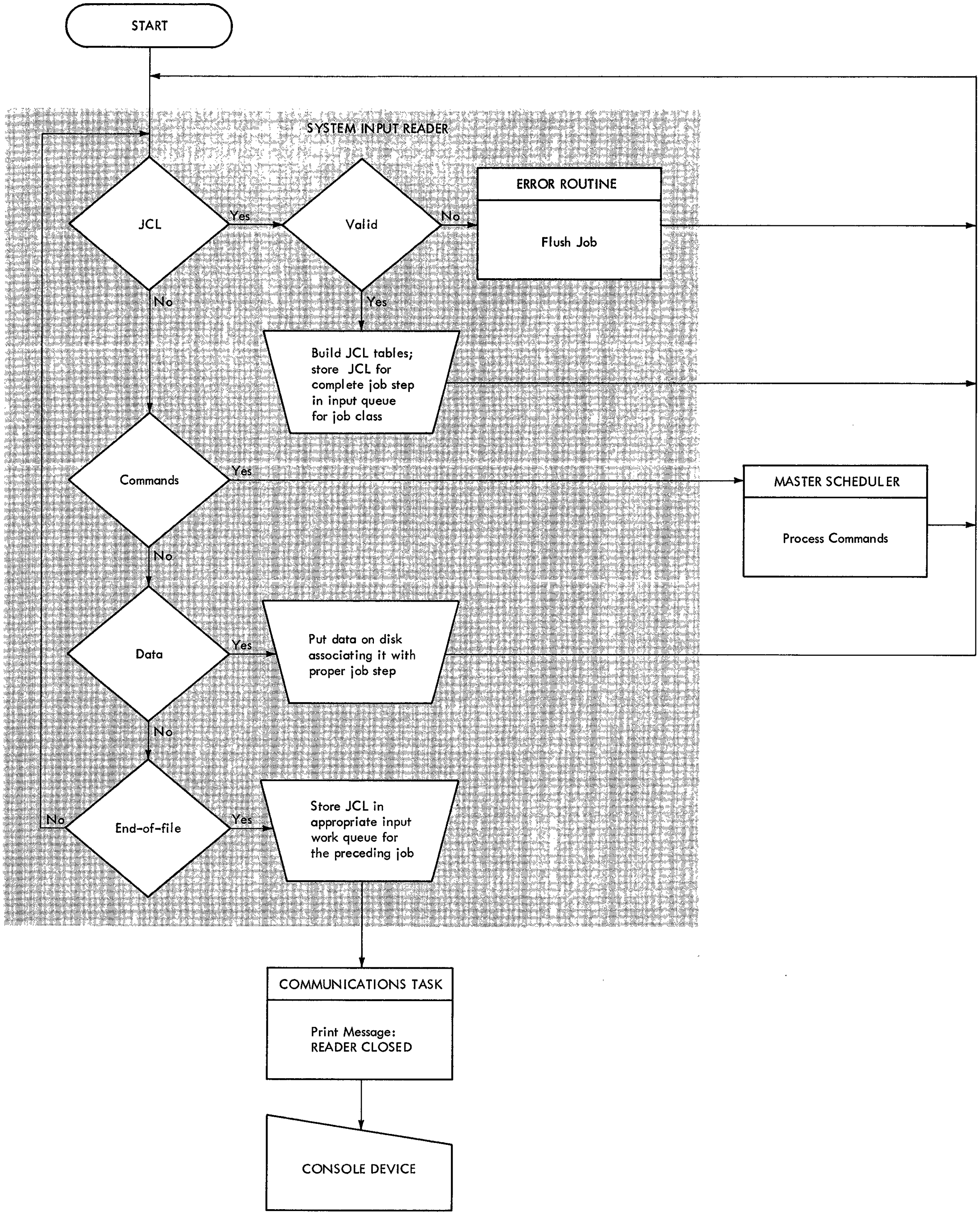

Figure 18. System Input Reader Processing

Note: If no PRTY parameter is specified on the JOB card, the job is assigned the default priority specified in the reader procedure.

JOB INITIATION AND TERMINATION

As illustrated in Figure 19, the job initiation portion of the control program selects jobs from the input work queues. As each problem program is executed, it retrieves its input (SYSIN) data from the direct-access device where it was previously stored by the system input reader. (Note that this retrieval takes place at direct-access speeds, and is much faster than reading input data directly from a card reader or magnetic tape unit). During problem program execution, output data directed to an output class is recorded on a direct-access device.

Jobs are scheduled for execution according to:

1. Their job class identifier
2. Their priority within the job class queue
3. An available partition corresponding to the appropriate job class

When a job is complete, the terminator performs the required termination and informs a system output writer that the data produced by the problem program is ready to be written on the specified device.

Job Initiation

To schedule a job, the system places an initiator into an available partition. The initiator selects a job from the input work queues established by the system input reader, allocates devices for them, and schedules them for execution. The initiator operates in any scheduler-size problem program partition. Small partitions are scheduled by an initiator in a large partition. Initiators obtain jobs for partitions based on the job classes assigned to the partitions and the priority of the jobs within their job classes. The job is then scheduled for execution. An initiator can be given control during system initialization, or after a job has been terminated.

An initiator first checks whether a small partition needs scheduling. If the small partition has requested scheduling, a job requiring that partition is scheduled into it. The initiator then schedules the next available job into its partition and passes control to the first step of that job. As the job is processed, output data sets are placed on direct-access storage for subsequent retrieval by a system output writer. These output data sets are enqueued by output class (described below).

Job Termination

The terminator determines first whether step termination or job termination is to be performed. Step termination includes disposing of data sets, deallocating input/output devices, processing condition codes, and executing the user's accounting routine. If the job contains additional steps, control is returned to the initiator to schedule the next job step.

Job termination is performed after the last step of a job has been terminated. The user's accounting routine is executed, data set disposition and input/output deallocation that could not be done at step termination are completed, and the job's output is entered in the output work queue for processing by a system output writer.

Output work queue members are enqueued FIFO within output class. For example, if a single output class is specified for system messages and all output for a particular problem program, the output work queue for that class includes, at job termination:

1. All system messages produced at job initiation, such as allocation messages
2. All problem program output
3. All system messages produced during job termination

This output is transferred to the specified output device in the order shown. Different types of output, such as system messages and problem program data, are never intermixed. Control is then returned to the initiator to schedule a new job.

Data sets for a job are enqueued on the output work queue according to SYSOUT class, so that they can be written by an output writer. These data sets may include data sets produced during a job step, as well as control program messages. Depending on its characteristics and the way the user wishes it to be processed by the control program, a data set may be assigned to any one of 36 output classes (A-Z, 0-9) defined at an installation. A particular output class may reflect such characteristics as priority of the data, type of device to record it, or location or department to which it is to be sent. (See "Choosing SYSOUT Classes" in the Considerations section.)

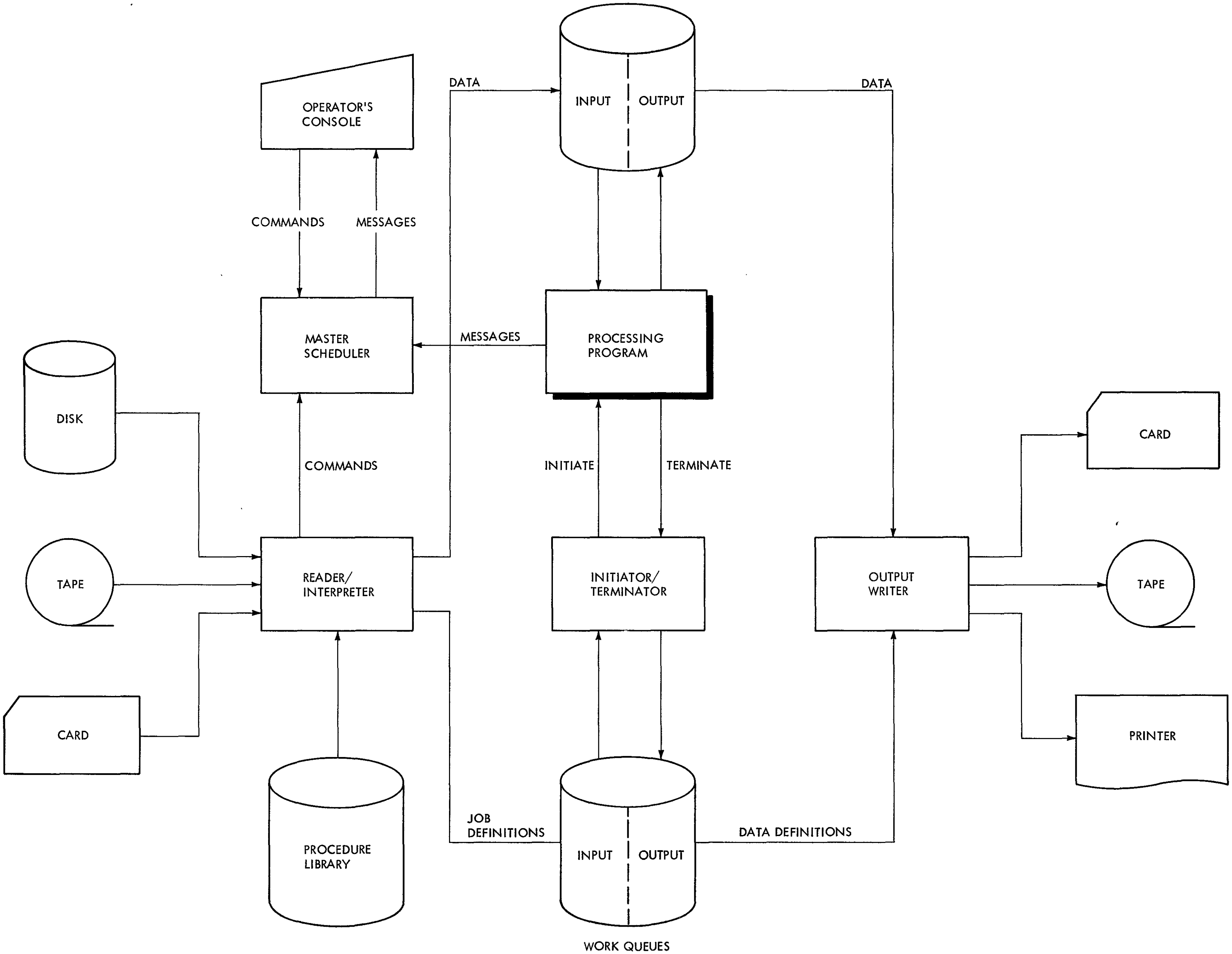

Figure 19. The MFT II System

OUTPUT WRITERS

Output writers write SYSOUT data sets created by problem programs, and system messages produced by the Initiator/Terminator tasks. All SYSOUT data sets must be written on direct-access storage devices by the user's problem programs. These are then used as the input data sets for the output writers. Valid output devices for an output writer are printer, punch, and magnetic tape. Output writers can be resident or can operate in problem program partitions, when required.

When a job is terminated, system messages and output data are enqueued in the appropriate SYSOUT queue. One SYSOUT queue exists for each output class. Queues are serviced in the order specified in the START command. As many as eight SYSOUT classes may be specified. These classes override those in the writer cataloged procedure. The writer dequeues the first entry from the primary queue. If there are no entries in the primary queue, the writer dequeues the first entry in the secondary queue. This continues through the eighth queue or until the writer finds work.

For example, to start a writer in P2 (see Figure 11) to process six output classes, the operator would enter the following command:

```
START WRITER(P2),00E,,(A,B,C,D,E,F)
```

(The comma before output class A replaces the positional parameter, volumeserial.)

The writer first processes all the output from output class A. When the queue for class A is empty, the writer processes class B. When the output class B queue is empty, the writer again searches the class A queue. If no output has been enqueued for class A, the writer now searches the output class C queue. The writer continues processing in this manner until the following STOP command is entered:

```
STOP WRITER,P2
```

If there is no work enqueued for any of the classes assigned to an output writer, the writer is placed in a wait condition until a job is terminated that has system messages and/or SYSOUT data sets for one of the writer's classes. When the last record in a queue entry has been processed, the writer deletes the entry before dequeuing another entry.

Writer tasks are performed concurrently with other writer tasks and with problem programs. A writer task is terminated only when the operator issues a STOP command. The MODIFY command can be used to change a writer's classes. (See "Changing the Output Classes" in the <u>Considerations</u> section.)

If the user wishes a resident writer, he assigns job class W to the partition at system generation, designating it as a writer partition. As with a resident reader, no other work is performed in the partition. Resident writers run concurrently with resident reader and problem program partitions, and are scheduled by a scheduler-size partition, whenever the large partition is available for scheduling duties. (See "Job Initiation and Termination" in this section.)

Writers that operate in problem program partitions are brought into a problem program partition of sufficient size (at least 10K plus the input buffer size) by a START command which specifies that partition or contains an S for a system-assigned writer. A system-assigned writer does not leave its partition, as does a system-assigned reader. The writer operates in a partition until a STOP command is entered. At that time, the scheduler may initiate a problem program in the partition.

The user may also write his own output writer procedure. This procedure may execute a user-written writer program. If a user-written writer procedure is used, it must be placed in SYS1.PROCLIB and named in the START command. For example, if the user wants to start his own output writer called USERWRIT in P3, the following command would be used:

```
START USERWRIT(P3),00E
```

(See "System Output Writers" in the <u>Considerations</u> section for additional information concerning resident and non-resident writers.)

SYSTEM RESTART

Because it is sometimes necessary to shut down the system (end-of-shift, end-of-day, normal maintenance, or system malfunction), system restart allows the system to resume operation without having to reenter jobs that have been enqueued by the system. Information concerning input work queues, output work queues, and jobs in interpretation, initiation, execution, or termination, is preserved for use when the system is reloaded. When the system is restarted, a message is written to the operator describing the status of each job in the system. The message includes the name of each job being performed at the time of shutdown, the name of the job step involved, and whether the job step was being interpreted, initiated, executed, or terminated.

Invoking System Restart

After the system is reloaded, and after nucleus initialization, system restart may be invoked by omitting the "F" suffix from the Q=(unitname,[F]) parameter of the SET command, or by omitting the Q= parameter entirely. This is valid only for the initial SET command, and cannot be done at any other time. This omission indicates to the system that the job queue data set already exists in the proper format and requires only initialization.

Jobs That Were Being Interpreted

When the system must be restarted, jobs that were being interpreted are run-out and must be reentered in the input stream. Any jobs that were being read, but were not yet interpreted, must also be reentered in the input stream.

Enqueued Input and Output Jobs

All input and output jobs on their respective work queues remain there for subsequent processing. A purge queue is constructed consisting of entries representing all input jobs that were being initiated, executed, or terminated, and all SYSOUT jobs that were being written. After the operator receives a list of the status of each job, the input and output queues are reconstructed from the purge queue. Input jobs are placed in their respective work queues at the end of the priority 14 entries for their original job class. SYSOUT jobs are placed at the end of the appropriate SYSOUT queue.

Jobs That Were Dequeued

Input jobs that were being initiated, executed, or terminated, are requeued for a run-out of their subsequent steps and the step that was being processed at the time the system was shut down. These jobs must be reentered in the input stream.

SYSOUT Processing

SYSOUT jobs that were being written are requeued to reprocess any data sets which had not been written completely at the time the system was shut down.

SYSOUT Pertaining to Dequeued Input

SYSOUT work associated with input jobs in the purge queue is enqueued at the end of the appropriate SYSOUT class queue. (See Figure 20.)

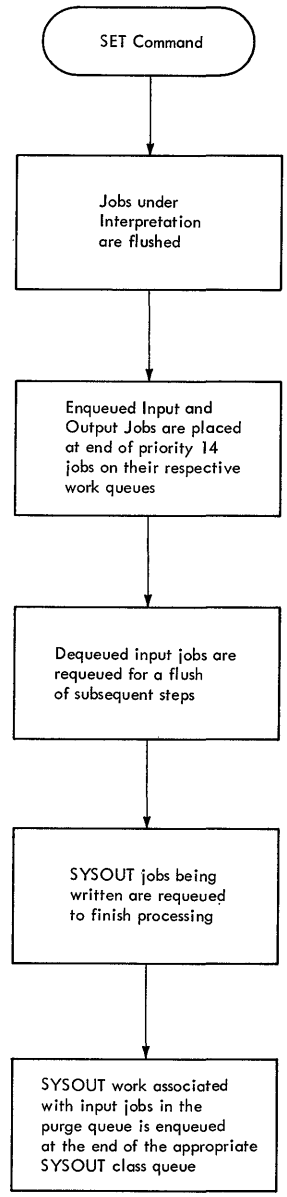

Figure 20. Restarting the System

In preparing for the use of MFT II, data processing planners and system programmers should evaluate not only the characteristics and requirements of the jobs to be processed by the system, but the characteristics and facilities of the system that influence how a job is processed once it has been presented to the system. Some of these characteristics are general and apply equally to all types of jobs. Others are related directly to job type. A third category of characteristics, although related to job types, is exhibited primarily in system operation, and must be considered by machine room supervisors and machine operators.

In this section, the topic "General Considerations" describes items of interest primarily to planning personnel, that should be considered before generation of an MFT II system. The next four topics -- "Batch Processing", "Telecommunications", "Graphics", and "Concurrent Peripheral Operation" -- describe considerations important to the systems programmer and the application programmer. These four topics are organized similarly, in "checklist" fashion, so that the reader interested in a given job type need read only "General Considerations" and the topic corresponding to the job type in which he is interested, to learn all considerations pertinent to that job type. Because partition configurations will depend on the amount of main storage available as well as on the types of jobs to be run, the topic "Typical System Configurations" describes partition arrangements for systems with 128K bytes, 256K bytes, and 512K bytes of main storage. These configurations are general, but should be helpful for planning. The sixth topic, "Operating Considerations" describes briefly characteristics of MFT II that may affect operating procedures.

GENERAL CONSIDERATIONS

Several considerations apply to all phases of the system; the user must know these regardless of the type of job he is running. These include proper placing of system libraries on direct-access devices and choosing the size of the scheduler for the installation. Others include choosing the number and size of partitions, and specifying appropriate job classes.

PLACING SYSTEM LIBRARIES ON DIRECT-ACCESS DEVICES

Several factors must be considered when putting system libraries (SVCLIB, MACLIB, LINKLIB, PROCLIB, and SYSJOBQE) on direct-access storage devices. If all five libraries are on the same device, throughput is decreased because of excessive arm interference. To increase throughput, the user should decide which channels he should use for the affected devices. Libraries should be balanced on devices; devices should be balanced on channels. The ideal condition would be to have each library on a different direct-access device, and each device on a separate channel. In installations with smaller systems, it would be best to have SYSJOBQE and LINKLIB on the same direct-access device on channel 1, and SVCLIB, PROCLIB, and MACLIB on another device on channel 2.

CHOOSING THE SIZE OF THE SCHEDULER

MFT II provides two schedulers, 26K and 44K. The choice of scheduler depends upon several factors: desired throughput, desired partition configuration, and main storage size. Usually, the primary factor is the amount of main storage available. In a 128K system, using the 26K scheduler allows specification of a greater number of partitions because 72K (with the 30K nucleus) would still be available to specify as other problem program partitions, writer partitions, and/or reader partitions.

With the 44K scheduler in a 128K system, the user has 54K to establish either another 44K partition and a 10K small partition, or several small problem program and/or writer partitions. The 44K scheduler increases throughput, but leaves less main storage for other partitions. If jobs are of short duration, it may be advisable to use the 44K scheduler, because scheduling activity will be high. However, if jobs are relatively long in execution time, the scheduler will not be needed as often; therefore, the 26K scheduler could be utilized.

Note: The size of the scheduler chosen at system generation determines which reader will be used by the system when START commands are entered for a reader (see "System Input Reader Partitions").

CHOOSING THE NUMBER AND SIZE OF PARTITIONS

The number of partitions needed at an installation depends primarily on the number of different job categories (i.e., batch, graphics, telecommunications, and CPO) expected to run concurrently. At least one partition must be specified for each category. The user should then establish the number of partitions for each category, based on the number of jobs expected to be run in each. In practice, the user generally should define at system generation the maximum number of partitions for which he has main storage. If fewer partitions are needed during operation, the number of partitions can be reduced by the operator. If necessary, partitions may be reestablished up to the limit specified at SYSGEN.

Within the limits of the system, the maximum number of partitions that can be specified depends on the size of the selected scheduler in relation to the amount of main storage available. At least one partition must be large enough to accomodate the selected scheduler. If a job is known to exceed the size of its intended partition, a partition can be eliminated, and its storage reassigned to one or more of the other partitions. Reassignment of contiguous partitions can be accomplished without interfering with unaffected partitions.

JOB CLASS CONCEPT

The partition job class concept places a responsibility on the user in setting up his system for maximum efficiency and throughput. Particularly, the processing characteristics of jobs likely to execute concurrently must be examined. Failure to consider job mix can lead to degrading system performance. Multiprogrammed jobs can, under certain circumstances, run slower than they would if processed sequentially. Because previous systems could not recognize processing characteristics of a particular group of jobs, it was necessary for job streams to be balanced so that concurrently operating jobs were complementary rather than conflicting. In MFT II the required balance can be achieved simply by proper use of the CLASS parameter. To use this parameter effectively, the user must:

- Establish the job characteristics to be controlled, based on his typical processing workload.
- Establish a suitable partition structure compatible with the job characteristics to be monitored.
- Establish the convention that jobs having certain characteristics are to be directed, through the CLASS parameter, to the appropriate partitions.

Typical job characteristics are:

- High compute, low input/output time
- Balanced compute and input/output time
- Low compute time, high input/output time
- Use of specific types of input/output equipment, such as 2250 terminals, magnetic tape only, or telecommunications
- Large main storage requirements
- Small main storage requirements

With this type of categorization, job mix can be balanced for improved throughput. For example, one partition can be established for high-input/output jobs and another for high compute-time jobs. Process-limited jobs can then be assigned to the high compute-time partition, and jobs with high input/output requirements (such as sort programs, reading, and writing of data sets) to the input/output partition. Normal job scheduling should then produce a satisfactory job mix. Since jobs are queued by the CLASS parameter, and since each partition is scheduled for its next job immediately after the preceding one is complete, the system as a whole will tend to execute complementary jobs concurrently.

The user must also be sure that he has assigned a partition to each job class specified on a JOB card. If he fails to do this, a job assigned to an unserviced class remains on the input queue for that class indefinitely, or until the operator discovers that the job has not been executed.

Default Job Class

If no CLASS parameter is specified on the JOB card, the system assigns job class A to the job. Therefore, the user should avoid assigning a small partition to job class A unless he is certain that all jobs run at the installation will fit into that partition. It is advisable to make job class A either a secondary or tertiary job class in one or more partitions, to ensure that any jobs that are assigned the default job class will be executed.

The default job class is given to a job only when no CLASS parameter is specified, not when an incorrect job class is given. For example, if P2 is specified as job classes M and L (Figure 11), P3 as C, J, and A, and P4 as N, C, and D, the following JOB card illustrates an invalid job class specification:

```
//MFTII JOB ,'MYJOB',MSGLEVEL=C,CLASS=G
```

Because job class G is an invalid job class for this particular configuration, the job will not be assigned job class A. It is placed on the CLASS=G queue, and is never initiated. It remains there indefinitely until the operator discovers that the job has not been executed. Therefore, extreme caution should be used when choosing a job class for the job to ensure that a partition has been specified for that job class. To prevent delays in processing jobs with "invalid" job class designators, the operator should enter DISPLAY N periodically to obtain a listing of the jobs on the hold and input work queues.

Priority Scheduling Within Job Classes

MFT II enables the user to initiate jobs <u>within</u> job classes according to a priority specified in the PRTY parameter on the JOB card. For example, the user may have several jobs designated job class B. Within this group of jobs some are to be initiated before others. Therefore, he assigns priorities to these jobs with the PRTY parameter. This affects only the way the job is initiated, <u>not</u> dispatched. If no PRTY parameter is specified, jobs are assigned the default priority established in the reader procedure and are initiated FIFO for each job class. Therefore, the user should consider each group of jobs for a certain job class, decide if some are to be initiated before others, and assign these preferred jobs higher priorities.

SYSTEM INPUT READERS

The size of the system input reader is determined by the user when he selects either the 26K or 44K scheduler; i.e., the size of the scheduler determines which reader is used by the system. (See "Choosing the Scheduler" in this section.) The user may choose a combination of resident and transient readers, but may not have more than three readers operating in the system at the same time.

Resident Reader vs. Transient Reader

The decision to use a resident or transient reader depends upon the amount of main storage available, and the quantity of work to be read from any one input device. A resident reader improves performance, but reduces the amount of main storage available for general use. Whether the advantages of a resident reader compensate for its overhead depends upon the size of the system and the type of job most frequently run. The following general considerations apply:

1. The system input reader can be resident in any MFT II system (128K or larger), and would probably be resident in any system larger than 128K.

2. In any system of sufficient size, the reader should be resident if a high-intensity job stream is typical. A high-intensity job stream is one in which a number of relatively short jobs appear; input-stream processing time is an appreciable percentage of overall job time.

With a 128K system, it may not be feasible to use a resident reader because of the small amount of main storage available. Therefore, a transient reader should be considered. A transient reader makes more main storage available for general use, at the possible cost of reduced overall performance. A transient reader should be considered also if the user has input streams on more than one device, and no device has a majority of the work.

Single Reader vs. Multiple Readers

In determining whether to have more than one reader, the user should consider the size of his machine and the number of problem program partitions necessary for the installation. It is not possible to have three resident input readers in 128K bytes of main storage machine, because there would be no main storage available for problem programs. However, if the user has 256K bytes or more of main storage, it might be advisable to specify more than one resident reader. For example, he could specify one reader for cards, one reader for magnetic tape, and a third for disk.

The primary consideration for the user is to analyze his jobs and determine which input device will have the majority of the input in terms of CPU time. If this device is the card reader, then he should probably specify one reader partition to read the input stream from the card reader continuously. If, on the other hand, he has a long input stream on magnetic tape and/or direct-access storage, he should specify a reader partition(s) for these devices, and use a transient reader for the card reader. If the user has only one long input stream, it would be advisable to specify one reader partition for that particular device, and start a transient reader for the other devices.

SYSTEM OUTPUT WRITERS

If possible, records to be written by a system output writer should be blocked. This improves throughput, because less input/output time is required, and disk arm interference is reduced. However, additional main storage must be provided within the problem program partition, where the records are initially blocked, and within the system output writer partition into which the logical records are read. (The additional space required, in each case, is equal to the logical record length times the blocking factor plus the input buffer space.)

Because the user has the choice of specifying partitions as resident writers or having the writers operate in problem program partitions (see "Output Writers" in the Concepts section), there are several considerations that he must be aware of when deciding which type of writer to use.

Resident vs. Non-Resident Writers

With a resident writer, machine size is not as large a factor as with a resident reader because the minimum system writer is only 10K plus the input buffer space and can operate in a small partition. The size difference between the writers varies only with the amount of buffer space specified for the output data sets (i.e., blocked vs. unblocked records).

If a resident writer is not desired and, instead, a writer is started in a small problem program partition, throughput may be decreased since the writer is not started until the problem program is completed in the small partition. Once the problem program has completed execution, the writer must wait until a large partition is free to schedule it. Thus the advantage of a resident writer is that it need not wait until a problem program partition is free but can continue writing until a STOP command is entered. If the system does not contain a resident writer, the user must always start a writer in a problem program partition often enough to prevent the output work queues from being filled. If the output queues become filled, the output data not placed on the queues is lost. Frequent use of DISPLAY Q command will allow the operator to anticipate this condition.

A system-assigned writer may also be started in a problem program partition; i.e., START WRITER(S),00E. However, the system-assigned writer does not operate in the same way as a system-assigned reader. The system-assigned writer does not leave the partition as the reader does, because the only way to stop a writer in a problem program partition is to enter a STOP command. Therefore, when using the system-assigned writer, the user should ensure that the writer will not remain idle, occupying the partition unnecessarily.

Use of Multiple Writers

The use of multiple output writers has several advantages. In general, a unique output writer can be used for each requirement in the system. For example, the following output classes might be assigned:

- An output class for all system messages
- An output class for all high-priority printed output, or for printed output requiring special forms
- An output class for all punched output
- An output class for all output to magnetic tape

By specifying the appropriate output class in his DD statement, the programmer selects the particular device on which his output is to be recorded. Because writers can share output classes, a writer can have a primary and a secondary function. For example, if output class B is assigned to a high-priority printer, and output class C to a "background" printer, the high-priority printer processes only high-priority output (SYSOUT=B).

If no high-priority data is waiting on the output work queue, the output writer performs its secondary function, by taking a job from the SYSOUT=C queue. The advantage in this use of multiple writers is not only that it makes writers available for certain types of unique work, but that it also permits them to perform other work when circumstances permit.

Note: Problem programs that are device-dependent with respect to data sets written by a system output writer may have to be modified to run under MFT II. If the DCB for the system output device was coded for other than a direct-access device, it may not be large enough for conversion to a direct-access device DCB, as required by MFT II.

SYSTEM INTERLOCK

A problem can exist when one task (A) controls a specific resource (Y), while waiting for another task (B) to relinquish control of another resource (Z) that task A needs to continue processing. If task B is waiting for task A to relinquish control of resource Y before task B gives up control of resource Z, a system interlock condition exists. Task A cannot give up resource Y until task B gives up resource Z, and vice versa. Therefore, tasks A and B are virtually in a deadlock; processing can not continue in either partition.

Two ways to avoid this problem are:

1. Request all resources initially; do not begin an irreversible course of action until all required resources have been obtained.

2. If holding a formerly obtained resource which may prevent acquisition of another resource, release the resource before requesting the other resource(s). If the former resource is still required, request it together with the other resource(s). (See the publication IBM System/360 Operating System: Supervisor and Data Management Services, Form C28-6646, for further information on system interlock.)

BATCH PROCESSING

If the installation's work is primarily batch jobs, the user must consider several factors when he initializes and operates his system. First, he must choose the number, size and job class(es) for each partition he wishes to use. Then he must decide which partitions, if any, should be specified as resident reader and writer partitions. He must also choose the proper output classes for the installation.

CHOOSING NUMBER AND SIZE OF PARTITIONS

The number and size of partitions depends upon the size of the installation's system. Naturally, a user with 512K bytes or more main storage has great flexibility in setting up his system, yet even the 128K MFT II user has more flexibility than the PCP user. There are several possible configurations for a 128K system. The best configuration for a particular installation usually depends on the type of job most frequently run. There must be at least one scheduler-size problem program partition. (See "Choosing the Size of the Scheduler" in this section.) Remaining main storage can be assigned to other problem programs, reader, and/or writer partitions. (See "Typical System Configurations" in this section for examples of batch processing system configurations.)

SMALL PARTITIONS

Small partitions are well suited to batch processing. Remember though, that small partitions require a large partition for scheduling. (See "Initiation of Jobs" in the Concepts section), so total throughput may be reduced somewhat. For an installation with only small jobs to be run, however, the configuration shown in Figure 21 would operate efficiently.

With this configuration, P1 should be assigned to a job class that is never used (B in this example). Then, in effect, P1 has a "resident" scheduler; its only function is to schedule small partitions (P2-P5). Initially, the writer (P2) is scheduled first, and is not rescheduled unless a STOP and START command sequence is reentered for that writer. If another writer is necessary, P4 or P5 can be redefined as resident writers, or a non-resident writer started in either partition.

Note: The user must ensure that any job assigned a specific class can run in any partition assigned to that class.

CHOOSING READER PARTITIONS

A reader partition would probably be resident in any system over 128K, but may also be resident in a 128K system. If the user has a 128K system and chooses a resident reader, he should use the 26K reader. This makes more main storage available for problem program and/or writer partitions. When a resident reader is desired,

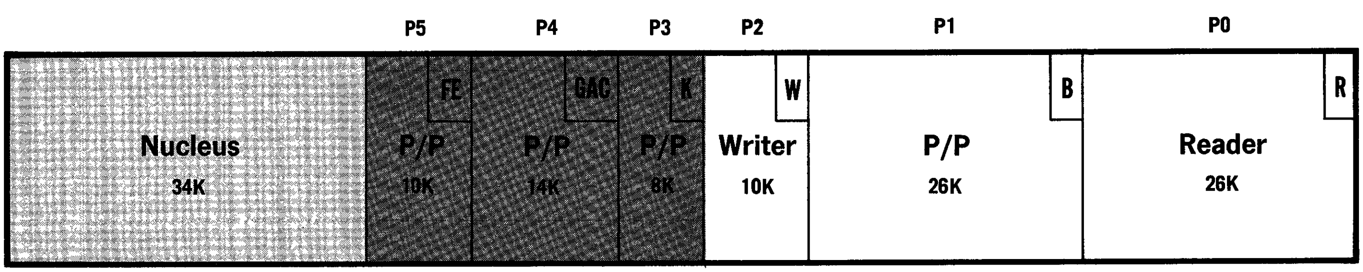

Figure 21. Sample 128K Small Partition Configuration

the best flow of input to the system, and the greatest amount of processing time for problem programs is gained by placing the reader in a high-priority partition. If the user has at least 256K, the 26K reader would probably not be used, since the 44K reader operates more efficiently.

Note: See "Reader/Interpreter Partition Size Requirement" in the Characteristics section.

DATA IN THE INPUT STREAM

Because MFT II permits data in the input stream, the total performance of the system may be slightly reduced because of the extra Reader/Interpreter time necessary to transcribe the data onto a direct-access device. The user can avoid this extra time by writing his data onto a direct-access device or magnetic tape unit prior to running his job. If this is not feasible, the data should be left in the input stream, or processed by a CPO program. (See "Concurrent Peripheral Operation" in this section.)

ASSIGNING JOB CLASSES TO JOBS

The user should assign every job a job class, using the CLASS parameter. For batch processing of input/output-limited jobs, he should assign a job class that corresponds to a high-priority partition. Process-limited jobs should be assigned to a lower-priority partition.

ASSIGNING PARTITIONS TO JOB CLASSES

Once job classes have been assigned to jobs, the user must ensure that appropriate partitions have been assigned to service those jobs. If the partitions do not have the appropriate job classes specified, the user can change these (see "Dynamic Partition Definition" in the Concepts section and "Redefining Partitions" in the Characteristics sections), or change the CLASS parameter on the JOB card.

CHOOSING WRITER PARTITIONS

If the user specifies several partitions as problem program partitions, it is advisable to have at least one resident writer in a higher-priority partition (P1 or P2). A resident writer will ensure that the output is continually being written. If another writer is needed a non-resident writer can be started in a problem program partition, or a problem program partition redefined as a writer partition.

Choosing SYSOUT Classes

At an installation it may be advisable to set up certain output classes for specific duties. For example, SYSOUT class A could be for system messages, and class B for problem program output. Or, class A could be for system messages, class B for problem program output for the accounting department, class C for problem program output for the purchasing department, etc.

Note: System messages are assigned an output class through the MSGCLASS parameter on a JOB card. Problem program output is assigned a class through the SYSOUT parameter on a DD card. (See IBM System/360 Operating System: Job Control Language, Form C28-6539.)

Another approach would be one in which SYSOUT class A represents printer system message output, class B represents punched card output, class C represents magnetic tape output, and class D represents printer problem program output. Up to 36 SYSOUT classes may be specified. When using special forms on the printer, the user should ensure that system messages are not written on the special forms. This possibility can be eliminated by establishing a different output class for output requiring the special forms.

Note: An identification problem may arise if system messages are assigned an output class different from problem program output. Therefore, it may be helpful for the programmer to print, as the first line of output, his name and department, if he chooses to use different classes for message and problem program output. This would also alleviate some operator problems (see "Operating Considerations" in this section).

TELECOMMUNICATIONS

MFT II enables the user to run telecommunications jobs concurrently with other types of jobs such as batch, graphics, and CPO. Several MFT II considerations are of interest to the telecommunications user.

As in the existing MFT, a major source of interference to other jobs in the system is the use of a system console device by the telecommunications job for line status information. When this technique is used, the wait time of the telecommunications job is used for writing the line status information, and processing time available to lower priority problem program partitions is severly limited. This limitation is avoided by using a "local terminal" for operator communication with the telecommunications job, and reserving the console device for communication with other jobs.

Other considerations include placement of the telecommunications partition, its size and job class, and the number of telecommunications partitions required.

CHOOSING NUMBER AND SIZE OF PARTITIONS

Telecommunications jobs are considered unending in that they are scheduled only once, unless a CANCEL command is entered for partition redefinition. (See "Dynamic Partition Definition" in the Concepts section and "Redefining Partitions" in the Characteristics section.) There must be at least one partition for each telecommunications job being run. The size of the partition depends upon the size of the telecommunications control program used by the installation.

To avoid delays in servicing lines, a telecommunications job should have unrestricted access to the resources of the central processor. For this reason, it is best to run telecommunications jobs in high-priority partitions. Because the telecommunications job is not alone in the system, its activities should cause minimum interference with jobs in other partitions, and it should not be susceptible to interference from these other jobs.

SMALL PARTITIONS

If the telecommunications control program needs a small amount of main storage (i.e., less than the size of the chosen scheduler), use of a small partition would be efficient since it will have to be scheduled only once.

CHOOSING READER PARTITIONS

If the user is running primarily telecommunications jobs, it would not be necessary to specify a partition as a resident reader. Initially a transient reader could be brought into a batch problem program partition to read in the telecommunications jobs as shown in Figure 22. Once the telecommunications partitions are activated, and a transient reader has read a job for P4, the reader will give control to the initiator to schedule a job into P4. (See "Input Readers" in the Concepts section.)

ASSIGNING JOB CLASSES TO JOBS

Each telecommunications job should have a unique job class assigned to it. The message control partition (P0) should have a different job class from the message processing partition. The user must also take care not to assign problem programs job classes that correspond to those of the telecommunications partitions. In Figure 22, a job is assigned a CLASS parameter of F if it is a telecommunications message control job. However, if the CLASS parameter is K, the job is placed on input queue K, but never initiated.

ASSIGNING PARTITIONS TO JOB CLASSES

Each telecommunications partition should also have a unique job class so that the appropriate jobs may be directed to that partition. In Figure 22, P0 is assigned job class F, P1 is assigned E, and P2, D. If the user decides to change job classes, he may do so by first entering a CANCEL command to terminate the unending job, and then dynamically repartitioning the system (see "Dynamic Partition Definition" in Concepts). Likewise, if the partition is not assigned to the telecommunications job class, the telecommunications job may never be initiated.

CHOOSING WRITER PARTITIONS

Telecommunications jobs will have no real need for resident writers. However,

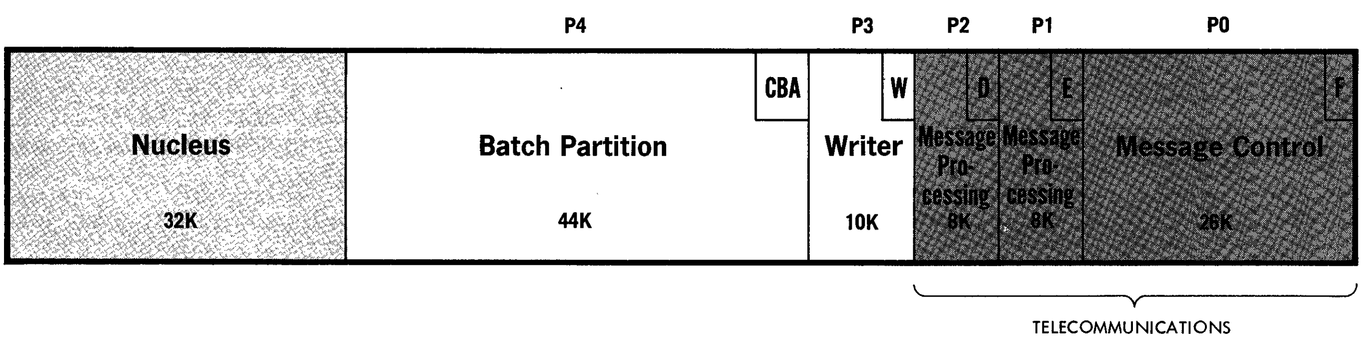

Figure 22. Sample 128K Telecommunications-Oriented Configuration

when batch jobs are run concurrently with telecommunications, a resident writer can be assigned to service the batch partition, as in Figure 22.

GRAPHICS

Graphics jobs in an MFT II environment are subject to the same general considerations as in the existing MFT. A graphics job associated with an unbuffered IBM 2250 Display Unit may operate with reduced performance if high telecommunications activity interferes with its access to the central processor for regenerating the display. In this case the user must determine the relative importance of the graphics and telecommunications jobs, and decide which to run in the higher-priority partition. Additional considerations for MFT II include assigning job classes to jobs, choosing the partition to service graphics jobs, and assigning partitions to job classes.

CHOOSING NUMBER AND SIZE OF PARTITIONS

There must be at least one partition for each graphics job being run. The partition size depends upon the size of the graphics job. Generally, graphics jobs should be run in a high-priority partition to cause minimum interference with other jobs. Figure 23 shows a graphics-oriented system configuration. Graphics jobs are executed in P0. P3 could be used for large compilers, with P1 used as a resident writer to service P2 and P3. If telecommunications and graphics are being run in the same system, the best performance would be gained by placing the telecommunications job in a high-priority problem program partition, and the graphics job in a relatively high-priority partition also.

CHOOSING READER PARTITIONS

In an installation running primarily graphics jobs, a resident reader should not be necessary. The system could utilize a transient reader. When the graphics job has been scheduled, the partition containing the transient reader could process batch jobs.

ASSIGNING JOB CLASSES TO JOBS

Graphics jobs should also have a unique job class assigned to them, to ensure that they are executed in the selected partition. In Figure 23, jobs are assigned a CLASS parameter of C if they are graphics jobs.

ASSIGNING PARTITIONS TO JOB CLASSES

A graphics partition should be assigned a unique job class that corresponds to the job classes assigned to the graphics jobs. This ensures that jobs will be enqueued on the proper input queue, and executed in the appropriate partition. The partition could also be assigned secondary and tertiary job classes to reduce idle time. In Figure 23, P0 is assigned a primary job class of C for graphics jobs, and secondary and tertiary job classes of H and I. If the user decides to change the partition's job class, and a graphics job is being run, he must issue a CANCEL command and then redefine the partition. The CANCEL command is explained under the topic "Operator Commands" in the Characteristics section. "Redefining Partitions" is also in the Characteristics section.

CONCURRENT PERIPHERAL OPERATION

Concurrent Peripheral Operation (CPO) is the capability of performing utility functions such as card-to-tape, tape-to-print, or tape-to-punch while other jobs in the system continue processing. Execution of CPO jobs in MFT II involves the same general considerations for assigning job classes to jobs and partitions as for telecommunications and graphics jobs. CPO jobs should be assigned a class that corresponds to that of the CPO partitions. CPO jobs can be placed anywhere in the system. Figure 24 illustrates a system configuration containing one CPO partition,

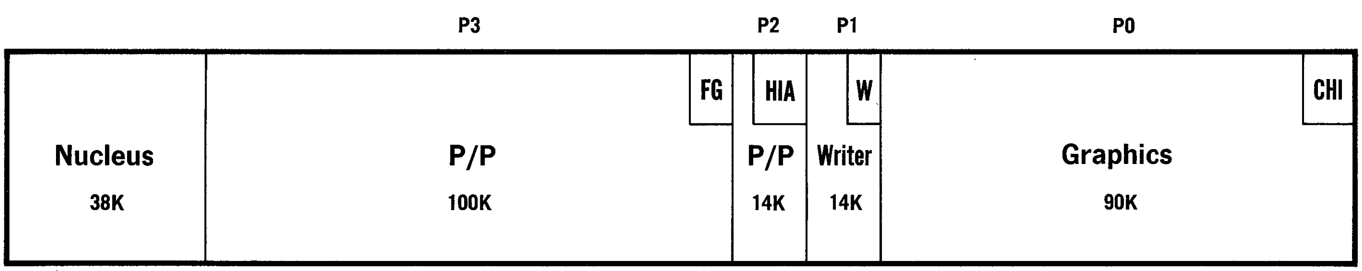

Figure 23. Sample 256K Graphics-Oriented System Configuration

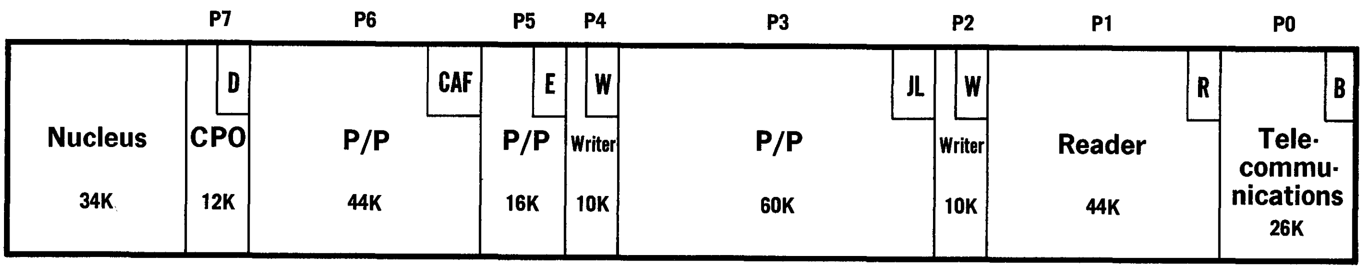

Figure 24. Sample 256K Telecommunications, CPO, and Batch Processing Configuration

one telecommunications partition, one resident reader, two resident writers, and three batch partitions. The CPO partition (P7) is assigned job class D, and no other partition is assigned this class.

TYPICAL SYSTEM CONFIGURATIONS

This topic describes partition configurations for systems with 128K, 256K, and 512K bytes of main storage. These configurations are based on the considerations presented in the preceding four topics. Actual configurations will depend on the individual requirements of each installation.

SYSTEMS WITH 128K MAIN STORAGE

A 128K system can support a variety of configurations. The configuration best for a particular installation usually depends upon the type of job most frequently run. Two examples follow.

Long-Duration Jobs

For applications where the typical job is of relatively long duration, a transient reader may be best. For such an application, where input stream processing requires a small percentage of total operating time, the configuration shown in Figure 25A might be used. This configuration has:

- Two problem program partitions, each handling batched jobs
- Two to three system output writers, servicing the batch partitions
- A transient system input reader

This configuration permits all of the advantages of MFT II to be realized, including independent partition scheduling and concurrent operation of problem programs and system output writers.

It would be advantageous to assign a single job class to each of the two problem program partitions, with one partition to be used for small batch jobs and the other partition for larger jobs. Alternatively, jobs could be directed to one partition or the other on the basis of characteristics other than size such as input/output device requirements.

High-Intensity Job Stream

A second configuration, shown in Figure 25B, has:

- One resident system input reader partition
- One problem program partition, occupying all storage remaining available after reader and writer assignment
- Two or three system output writer partitions, servicing the problem program partition

This configuration should provide extremely efficient processing of a high-intensity job stream, where the typical workload involves a relatively high number of short-duration jobs. With the reader resident, input stream processing and job scheduling is concurrent with job execution and output writer processing. This combination should operate faster than PCP because of concurrent processing.

Because a high-intensity job stream requires more processing time than input/output time, the characteristics of the problem programs and the reader are highly complementary, with a corresponding improvement in throughput.

SYSTEMS WITH 256K MAIN STORAGE

A 256K system makes possible a wider variety of configurations. Depending on the user's requirements, the most likely configurations will include two large (80K to 90K) batch partitions, or three to four medium-size (44K or greater) batch partitions. In either case, several system

	P3	P2	P1	P0
Nucleus	P/P	Writer	P/P	Writer
32K	32K	10K	44K	10K

A. Two Problem Program Partitions - Two Resident Writers

	P3	P2	P1	P0
Nucleus	Writer	P/P	Writer	Reader
32K	10K	50K	10K	26K

B. One Problem Program Partition - One Resident Reader - Two Resident Writers

Figure 25. Sample 128K Batch System Configurations

output writers could be provided to support the batch partitions. Figure 26A illustrates a configuration with one large (88K) partition, one medium size (44K) partition, two small (14K) partitions, a resident reader, and a resident writer. Figure 26B illustrates the second case: three medium-size batch partitions, a resident reader, and three resident writers.

SYSTEMS WITH 512K MAIN STORAGE

The choice of configurations available to the 512K user is so great that no "typical" system can be defined. Such a system can support three or four 80K to 90K partitions, and whatever combination of supporting output writers and input readers is desired. Figure 27 shows this possible 512K configuration: four large (88K and 90K) batch partitions, two resident readers, two resident writers, and one small batch partition.

OPERATING CONSIDERATIONS

The operator of an MFT II System must be aware of several considerations related primarily to program execution, partition definition, and output class reassignment.

	P5	P4	P3	P2	P1	P0
Nucleus	P/P	P/P	P/P	Writer	P/P	Reader
36K	14K	14K	44K	16K	88K	44K

A. One Resident Reader - One Resident Writer - Four Problem Program Partitions

	P6	P5	P4	P3	P2	P1	P0
Nucleus	Writer	P/P	P/P	Writer	Writer	P/P	Reader
34K	10K	44K	60K	10K	10K	44K	44K

B. One Resident Reader - Three Resident Writers - Three Problem Program Partitions

Figure 26. Sample 256K Batch System Configurations

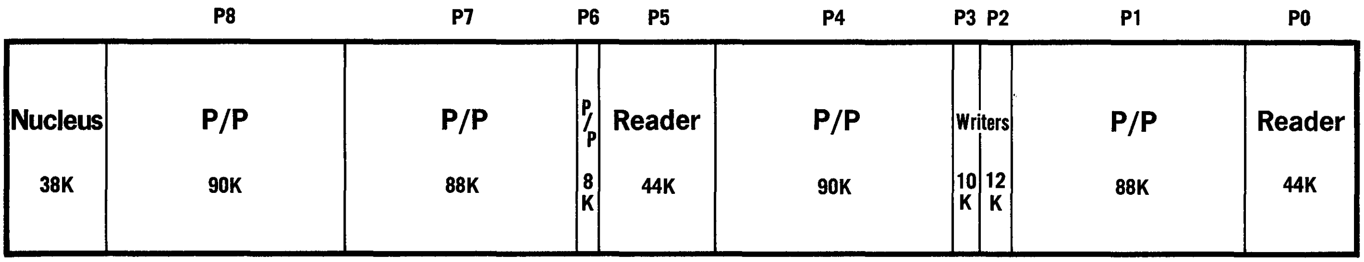

Figure 27. Sample 512K Batch Processing System Configuration

These considerations are explained in the following paragraphs.

PROGRAM EXECUTION

Because 15 problem programs can be executed concurrently, the system places additional responsibility on the MFT II operator. At times he may become busy replying to system messages and problem program messages, placing special forms in the printer, etc. Therefore, whenever possible he should perform as much preparatory work, such as mounting required volumes, ahead of the required time. When responding to problem program messages, the operator should respond to the highest priority task first; i.e., the message from the partition with the lowest number. (See "Problem Program Messages" in the <u>Characteristics</u> section.) The operator must also remember that problem program and system messages may be intermingled with each other on the console device.

In addition, because jobs may not be completed in the same order as they were entered into the system, the operator must ensure that the correct output is returned to each user. The operator may also be required to start system input readers and output writers at certain times during operation. He may be given a specific time each day, or may have to use his judgment based on work load for the system.

PARTITION DEFINITION

Even though the installation may not intend to use the maximum number of partitions at all times, the system must be regenerated if the user decides to increase the number of partitions originally specified. Therefore, the maximum number of partitions that an installation expects to use should be specified at system generation. Partitions can then be redefined to decrease the number actually in use.

Caution must be observed when redefining partitions. Before redefining partitions, (see "Entering the Partition Definitions" in <u>Characteristics</u>) the operator should check the job class(es) of all pending jobs and ensure that the prospective partition definitions have job classes corresponding to the jobs that will be executed. This includes knowing the job classes of jobs which have already been read onto SYSJOBQE, but have not been executed (see "Specifying LIST and CLASS" in the <u>Characteristics</u> section). If possible, he should also check pending jobs for their size requirement and compare this with the size of their partitions. If they are originally assigned a CLASS parameter that corresponds only to a large partition, he should reassign them to a large partition.

CHANGING OUTPUT CLASSES

The output classes with which a writer is associated can be changed at any time, through STOP and START commands. A user with a special forms requirement can obtain exclusive use of a printer by issuing a MODIFY command or a STOP command followed by a START command for the same writer but specifying a unique output class. The STOP command causes the writer to stop at the end of the job it is currently executing. The operator then inserts the required forms and issues the new START command. That command would limit use of the printer to the data set associated with the new output class until another STOP and START command sequence for the printer is issued. The MODIFY command can also be used to change the conditions under which the output writer pauses for servicing of its device.

This section presents detailed descriptions of how to estimate storage requirements for an MFT II system, MFT II changes and additions to the system generation process, and the Job Control Language (JCL), and operator commands and messages. The first four topics -- "Storage Estimates," "System Generation," "Assigning Job Classes," and "The WAITR Macro Instruction in MFT II" -- should be read by planners, systems programmers, and application programmers. The remaining topics -- "Redefining Partitions," "Restarting the System," "Operator Commands," and "Operator Messages" contain information for the MFT II operator, but may also be of interest to systems and application programmers.

The following conventions are used in illustrating the format of macro instructions, replies, and commands:

- Required letters (those shown in upper case) must be entered, but can be entered in either upper or lower case.
- Lower-case letters indicate that an appropriate value must be substituted.
- Ellipses ... (indicating a series of terms), brackets [ ], and braces{ } are not entered.
- Entries within brackets are optional.
- Entries within braces are required - one must be selected.
- Stacked items, enclosed in either brackets or braces, represent alternative items. No more than one of the stacked items should be coded by the programmer.
- If an alternative (stacked) item is underlined, that item is implied; that is, the control program automatically assumes it is the programmer's choice when none of the items is specified.
- Numbers and punctuation marks (other than ellipses, brackets, and braces) must be entered as shown.

STORAGE ESTIMATES

MFT II operates in a system with at least 128K bytes of main storage. The system nucleus requires approximately 30K bytes, exclusive of space for resident access methods or the resident BLDL table (see "System Nucleus" in the Concepts section for the components of the nucleus). If these options are selected, the corresponding main storage requirements should be added to the size of the nucleus. Remember that 800 bytes must be added to the size of the nucleus for each partition specified above two.

CONSTANT REQUIREMENT FOR MFT II

The following formula is used to determine the constant storage requirement for MFT II:

CONSTANT = N + XP + 800(P-2)

where: N = the fixed amount required by the control program, 25090 bytes.

X = the size of the control blocks for each task;

X = 212 bytes if the central processing unit has floating point registers.

X = 168 bytes if these registers are not present.

P = the number of partitions generated.

READER/INTERPRETER PARTITION SIZE REQUIREMENT

The size of a partition for a reader/interpreter depends on the size of the scheduler chosen at system generation, and the size of buffers. The buffer sizes are specified in the cataloged procedure used when a reader is started. If the buffer sizes are overridden, the partition size must be adjusted accordingly. The partition sizes for the 26K and 44K readers are:

$$\begin{Bmatrix} 26K \\ 44K \end{Bmatrix} + IB + OB$$

where: IB = the size of the input buffers
OB = the size of the output buffers.

OUTPUT WRITER PARTITION SIZE REQUIREMENT

The size of a partition for an output writer depends on the size of the buffers and the data set writer used. The buffer sizes are specified in the cataloged procedure used when a writer is started. If the buffer sizes in the procedure are overridden, the partition size must be increased accordingly.

If the standard data set writer is used, the partition size requirement for the writer is:

10K + IB

where: IB = the size of the input buffer.

The preceding requirements allow for 132-character output buffers. If larger output buffers are required, their size must be added to the partition size. If the user provides a non-standard data set writer, the size of the partition must be adjusted accordingly.

MINIMUM SYSTEM

The 30K MFT II nucleus supports the following minimum configuration:

- Two partitions
- One multiplexor channel (required)
- One selector channel (required)
- Three IBM 2311 direct-access storage devices (two required)
- Four tape devices
- One 1052 console (required)
- One card reader or additional device (required)
- One card punch or additional tape device (required)
- One printer or additional tape device (required)
- Storage protection
- RAM (include QSAM only, for system readers and writers)

Note: Items not listed above are not included in the minimum system.

For each additional channel or input/output device, the size of the nucleus must be increased by the amount of main storage required for the corresponding Input/Output Supervisor routines, as listed in the publication: IBM System/360 Operating System: Storage Estimates, Form C28-6551.

The configurations possible at a given installation may be roughly calculated as follows:

1. From the total main storage capacity, subtract the size of the system area (see "Constant Requirement for MFT II," described above).
2. Subtract the size of the scheduler-size problem program partition which is required for the system.
3. If resident writers are used, subtract 10K plus the input buffer size for each writer specified.
4. If it is desired that the system input reader(s) be resident also, subtract the size of the reader for each one specified (26K or 44K plus the buffer sizes, depending on the scheduler chosen).

Remaining main storage may be apportioned through any combination of the following:

1. Inclusion of other optional features of the Operating System.
2. Increasing the size of the problem program partitions already defined.
3. Increasing the number of problem program partitions.
4. Increasing the number of system output writer partitions.
5. Increasing the number of system input reader partitions.

Note: If the system includes the storage protection feature, all storage assignment increases must be made in increments of 2K bytes.

SYSTEM GENERATION

Generation of an MFT II system involves changes to the SCHEDULR and CTRLPROG macro instructions, and a new macro instruction designated PARTITNS. These changes, and the PARTITNS macro instruction are described in this topic.

THE CTRLPROG MACRO INSTRUCTION

MFT II requires two additions to the CTRLPROG macro instruction as it is now defined in IBM System/360 Operating System: System Generation, Form C28-6554. These are shown below.

Name	Operation	Operand
	CTRLPROG	[SYSQUE=size] TYPE= {PCP / MVT / MFT}

SYSQUE=size
: specifies the size of the system queue area. At least 800 bytes must be allocated for each partition specified. The default for this parameter is 1600 bytes.

TYPE= {PCP / MVT / MFT}
: specifies the type of system under which a problem program is to be run. If MVT or MFT is not specified, PCP is assumed. This parameter should agree with the TYPE parameter of the SCHEDULR macro instruction.

Note: The PARTITNS macro instruction eliminates the requirement for the HITASK and LOWTASK parameters of the CTRLPROG macro instruction. The STORAGE=PARTITIONED parameter of the CTRLPROG macro instruction is replaced by the TYPE=MFT parameter.

EXAMPLE: The following example illustrates the use of the CTRLPROG macro instruction to specify that the maximum number of input/output operations that can be processed simultaneously is 20. The basic overlay supervisor and standard fetch are included. The system queue area is assigned the default size of 1600 bytes. Problem programs are to run in a fixed task environment (MFT).

```
CTRLPROG MAXIO=20,TYPE=MFT
```

THE SCHEDULR MACRO INSTRUCTION

The DESIGN parameter of the SCHEDULR macro instruction is changed for MFT II. The TYPE parameter is new for MFT II. These are described below, along with the MVT parameters used by MFT II.

Name	Operation	Operand
	SCHEDULR	[TYPE= {PCP / MVT / MFT}] For TYPE=MFT Only: [DESIGN= {26K / 44K}] For TYPE=MFT and MVT Only: [STARTI= {MANUAL / AUTO}] [JOBQRES=address] [PROCRES=address] [JOBQFMT=value] [JOBQLMT=value] [JOBQTMT=value]

TYPE= {PCP / MVT / MFT}
: specifies whether the PCP, the MVT, or the MFT Scheduler is to be included in the system. If MFT or MVT is not specified, PCP is assumed. This parameter should agree with the TYPE parameter of the CTRLPROG macro instruction.

DESIGN= {26K / 44K}
: specifies the size of the job scheduler in bytes. Only the 26K and 44K schedulers are provided for MFT II. 26K is assumed if neither size is specified.

STARTI= {MANUAL / AUTO}
: specifies whether an initiator is to be started automatically or manually after IPL. If STARTI is not specified, MANUAL is assumed.

JOBQRES=address
: specifies the address of the device upon which the job queue resides. If JOBQRES is not specified, the system residence device is assumed.

PROCRES=address
: specifies the address of the device upon which the procedure library resides. If PROCRES is not specified, the system residence device is assumed.

JOBQFMT=value
: specifies the number of 176-byte data records to be included on a logical track on the Job Queue. Value is greater than or equal to 10, and less

than or equal to 255. If JOBQFMT is not specified, the value is set to 12.

JOBQLMT=value
: specifies the number of 176-byte records to be reserved for each initiator started. Any job will be terminated if more than this number of records is required to initiate the job. This value should be a multiple of the number of records per logical track (JOBQFMT). Value is greater than or equal to 2 times JOBQFMT, and less than or equal to 9,999. If JOBQLMT is not specified, the value is set to 60.

JOBQTMT=value
: specifies the number of 176-byte records to be reserved for the termination of initiated jobs which exceed JOBQLMT. This value should be a multiple of logical tracks (JOBQFMT). Value is greater than or equal to JOBQFMT, and less than or equal to 9,999. If JOBQTMT is not specified, the value is set to 60.

The following SCHEDULR parameters are supported by MFT II as they are documented in the publication IBM System/360 Operating System: System Generation, Form C28-6554: CONSOLE, ALTCONS, STARTR, STARTW, ACCTRTN, WTOBFRS, REPLY, and OPTIONS.

The TSYSIN, TSYSOUT, MINPART, WTLCLSS, WTLBFRS, RESJOBQ, INITQBF, CANCEL, VLMOUNT, and TAVR parameters are not used by MFT II.

EXAMPLE: The following example illustrates the use of the SCHEDULR macro instruction. The size of the job scheduler is 44K bytes. The address of the console device is 009. The address of an alternative console is 01F. The START commands for readers and writers are to be executed after the system is loaded into main storage. The devices to be started are located at 00C and 00E respectively. The START command for initiators is to be issued manually. The user will supply an accounting routine. The job queue and the procedure library reside on the device located at 190. There are 10 data records per logical track on the job queue. An MFT scheduler is to be included in the system. The default value is accepted for all parameters not specified.

```
SCHEDULR   DESIGN=44K,CONSOLE=009,
           ALTCONS=01F,
           STARTR=A-00C,STARTW=A-00E,
           ACCTRTN=SUPPLIED,
           JOBQRES=190,PROCRES=190,
           JOBQFMT=10,TYPE=MFT
```

THE PARTITNS MACRO INSTRUCTION

The PARTITNS macro instruction is used to specify the job class and size of each partition. The format of the PARTITNS macro instruction is shown below.

Name	Operation	Operand
	PARTITNS	Pn(C-class,S-nnK),...

Pn
: n specifies a partition number 0-51 (leading zeroes are not necessary). Partition numbers may be specified in any order so long as every partition in the sequence is included.

C-class
: specifies the partition's function. "C-" appears as shown; One of the following must be substituted for class:

  class
  : specifies one to three alphabetic characters from A through O. Each character identifies a partition job class. Job classes are scheduled into the partition by priority; the first job class specified has the highest priority. One problem program partition must be specified.

  R
  : specifies a resident reader; as many as three may be specified.

  W
  : specifies a resident writer; as many as 36 may be specified.

S-nnK
: specifies the amount of main storage to be allocated to the partition. "S-" appears as shown; "nn" multiplied by 1024 bytes indicates the amount of main storage to be allocated to the partitions and may contain eight digits. "nn" must equal at least 8. "K" appears as shown. One problem program partition must be the size of the selected scheduler. Resident reader partitions must also be scheduler-size. Resident writer partitions must be at least 10K. (See "Storage Estimates" in this section.)

EXAMPLE: This example illustrates the use of the PARTITNS macro instruction to create a four partition system. Partition 0 is to be a 26624 byte problem program partition. Jobs that specify CLASS=B on their job card are to be scheduled into partition 0. If no CLASS=B jobs are pending, CLASS=C jobs are to be scheduled. If neither CLASS=B nor CLASS=C jobs are pending, then CLASS=A jobs are to be scheduled. Partition 2 is to be a 10240 byte resident writer partition. Partition 1 is to be a 26624 byte resident reader partition (Partition 1 need not be specified before partition 2). Partition 3 is to be a 36864 byte problem program partition. Only CLASS=D jobs are to be scheduled into partition 3.

```
PARTITNS    P0(C-BCA,S-26K),P2(C-W,S-10K),
            P1(C-R,S-26K),P3(C-D,S-36K)
```

ASSIGNING JOB CLASSES (CLASS)

Job classes are assigned to jobs through an addition to the JOB card. To assign a job class other than the default job class A, the following keyword parameter must be coded in the operand field of the JOB card:

CLASS=job class

where job class is replaced with an alphabetic character (A through O). (See "Job Class Concept" in the Concepts section.)

THE WAITR MACRO INSTRUCTION IN MFT II

Because MFT II provides independent job scheduling, SHIFT commands and WAITR macro instructions necessary for MFT I are no longer needed. Therefore, programs such as DUMWAIT (see IBM System/360 Operating System: System Programmer's Guide, Form C28-6550) are not necessary for MFT II. Present users' programs coded with the WAITR macro instruction will not have to be recompiled to eliminate the WAITR. The WAITR will be treated as a normal WAIT within MFT II.

REDEFINING PARTITIONS

The number, size, and job classes of partitions may be changed by the operator at any time after IPL. The number of partitions can be equal to or less than that specified at system generation. The total size of all partitions and the nucleus cannot exceed the amount of main storage available.

PARTITION REDEFINITION AT SYSTEM INITIALIZATION

The master scheduler issues a series of messages, some of which require a reply from you. These replies specify whether partitions are to be redefined, the values of the new parameters, and the end of redefinition information.

When initial program loading has been completed, the master scheduler issues the message:

id IEE601D CHANGE PARTITIONS?

If partitions are to be redefined, and if a list of the current partition definitions is desired, enter:

REPLY id,'YES,LIST'

If partitions are not to be redefined, enter NO instead of YES. The characters ,LIST are omitted from the reply if a list is not desired.

If you enter the NO reply, the READY message is issued, and you proceed with system initialization. If you enter the YES reply the system issues the message:

id IEE602A ENTER DEFINITION

(preceded by the current definitions if LIST was specified). At this time you can enter REPLY id,'END', if the partitions are not to be redefined. The system responds to END by issuing READY. Alternatively, you can redefine the partitions. See "Entering Partition Definitions" for an explanation of the redefinition procedure.

PARTITION REDEFINITION AFTER SYSTEM INITIALIZATION

To redefine partitions after system initialization enter the command:

DEFINE [LIST]

A list of current partition definitions is printed if the optional operand LIST is included in the DEFINE command. In response to the DEFINE command, the system issues the message:

id IEE602A ENTER DEFINITION.

You now enter the new definitions.

ENTERING PARTITION DEFINITIONS

When responding to the message ENTER DEFINITION or CONTINUE DEFINITION you may change a partition's size or job class, specify the last partition which is to be active, request a list of all current partition definitions including all definitions which may have just been entered, or request a list of all job classes which are currently being serviced. You may also end all further definition in the present series, and/or cancel all definitions which have been entered since the last ENTER DEFINITION message.

Changing the Size of a Partition

To change the size of a partition, enter:

REPLY id,'Pn=size'

where n is the partition number (0-51) and size is the new partition size (1-8 digits) expressed as an absolute decimal value (e.g., 40960) or as a multiple of 1024 plus the suffix K (e.g., 40K). The minimum acceptable size is 8192, or 8K. Size values that do not fall on a double-word boundary are rounded up to the next double-word boundary. If storage protection is included, sizes that are not multiples of 2K will be increased to the next multiple of 2K. A size of zero causes a partition to remain inactive.

Changing the Job Class of a Partition

To change the job class(es) of a partition enter:

REPLY id,'Pn=job class'

where n is the partition number (0-51) and one of the following is substituted for job class:

xxx specifies 1 to 3 alphabetic characters from A through O. Each character identifies a partition job class. Job classes are scheduled into the partition by priority, the first jobclass specified having the highest priority.

RDR specifies a resident reader partition.

WTR specifies a resident writer partition.

Specifying LAST

To specify the last partition which is to be active, enter:

REPLY id,'Pn=LAST'

where n is the partition number.

All partitions with partition numbers higher than n are considered inactive. If LAST is entered in error, the partition definition may be reentered, omitting LAST. LAST may not be specified for more than one partition.

Note: If the total space specified for all partitions defined is less than the total space available, all excess space is added to the LAST partition.

Multiple Changes to a Partition

To change the size and jobclasses of a partition enter:

REPLY id,'Pn=(size,job class)' or
REPLY id,'Pn=(job class,size)'.

Size, job class, and LAST may be used singly or in any combination of two or three. If used in combination they must be separated by commas and enclosed in parentheses.

Specifying LIST

To request a list of current partition definitions, (size and job class) enter:

REPLY id,'LIST'.

At system initialization, all job classes being serviced by active partitions are also listed.

Specifying CLASS

To request a list of all job classes being serviced by active partitions, enter:

REPLY id,'CLASS'.

At system initialization, this parameter is unnecessary if LIST is specified.

Specifying END

To end all redefinitions in the current series (a series begins with the reply following the message ENTER DEFINITION and ends with the reply in which END is specified) and to allow the system to implement all replies just entered, enter:

REPLY id,'END'.

You may enter definitions for each partition as frequently as necessary. When END is entered, the last entry for each partition number is accepted by the system.

Specifying CANCEL

To cancel all partition definitions in a series, you enter:

REPLY id,'CANCEL'.

The partition definitions remain as they were before the current series of definitions was entered; i.e., the system continues to operate as though no new definitions were entered.

Multiple Entries per REPLY

Replies may specify more than one request. Partitions may be redefined, and LIST, CLASS, END, and CANCEL all may be specified in a single reply. Any combination of these optional parameters may be specified in any order so long as they are separated by commas. The only restriction is that replies cannot occupy more than one line (128 characters).

EXAMPLE:

```
REPLY id,'LIST,P1=(36K,RDR),P2=(ABC,10K),
         CLASS,P0=0,P3=(LAST,WTR),
         P1=26K,END'
```

In this example a list of the new partition definitions for P0 through P3 is requested. Partition 1 is initially redefined as a resident reader partition of 36864 bytes. The last entry before END of this example changes the size of this reader partition to 26624 bytes. Partition 2 is redefined as a 10240 byte problem program partition having a primary job class of A, secondary job class of B, and tertiary job class of C. A list of the job classes currently being serviced by active partitions is requested. Job classes being defined in this reply are included. Partition 0 is to be inactive. Partition 3 is redefined as a resident writer partition. Any partitions numbered higher than 3 are to be inactive. The end of the current series of redefinitions is specified; all redefinitions entered since the last ENTER DEFINITION message can now be implemented by the system.

RESTARTING THE SYSTEM

To restart the system after it has been shut down, the same steps taken in starting the system are followed, except when the SET command is entered, either the "F" suffix from the "Q=unitname" parameter is omitted, or the entire "Q=unitname" parameter is omitted.

The following command illustrates this procedure:

```
SET DATE=yy.ddd,CLOCK=hh.mm.ss
```

By omitting the "Q=unitname" parameter, job queue data set information is saved. When restarting the system to save the information, you must make certain that all auxiliary storage volumes which were in use remain available. This insures that the job queue data set, output data sets, and input data sets accurately reflect the conditions which existed when a restart became necessary.

OPERATOR COMMANDS

This topic describes the commands used to give control information to the operating system. The formats, functions, parameters, and options of the commands are included. For convenience, the commands are presented in alphabetical order.

In systems with MFT II, abbreviations as well as the full command name can be used when keying in the commands. The usable names and abbreviations are:

CANCEL	C	REPLY	R
DEFINE	N	RESET	E
DISPLAY	D	SET	T
HALT	Z	START	S
HOLD	H	STOP	P
MODIFY	F	UNLOAD	U
MOUNT	M	VARY	V
RELEASE	A		

Note: The SHIFT command, used with the existing MFT is not recognized in MFT II.

Command formats are essentially free form, but one or more blanks must follow the operation field. Commands cannot occupy more than one line. For example, if a command is entered through a card reader, it may not be more than 80 characters in length, and if entered on the console, the command may be a maximum of 128 characters. (DEFINE is not allowed in the input stream.)

If comments on commands are necessary, they should appear to the right of the operand field and be separated from it by at least one blank. If the operand field is null, a comma followed by at least one blank indicates that comments will follow.

CANCEL--TERMINATE JOB IMMEDIATELY

The CANCEL command is used to immediately terminate the scheduling or execution of a job. The CANCEL command may also be used to stop a writer procedure. The writer will stop processing the data set it currently is handling, and proceed to the next data set. Optionally, an abnormal-end-of-task storage dump may be requested if the command is received while the job is running.

If the named job is in the input work queue when the CANCEL command is issued, the job is deleted from the queue and a message is issued stating the job has been cancelled. If the job is not on the queue and is not running, you will receive a message stating the command cannot be executed.

Operation	Operand
{CANCEL C}	{jobname taskname} [,DUMP]

jobname
: specifies the name of the job to be terminated. The maximum length of a job name is eight characters.

taskname
: corresponds to the taskname used when starting a writer procedure. (Taskname is valid only when cancelling a writer procedure.)

DUMP
: specifies that an abnormal-end-of-task storage dump is to be taken if a step of the job is being executed when the command is received. If the programmer has put in the SYSABEND data definition statement, a full dump is taken. If this card was omitted, an indicative (partial) dump is taken.

DEFINE--INVOKE DYNAMIC PARTITION DEFINITION

The DEFINE command is issued after nucleus initialization to invoke Dynamic Partition Definition. This will allow the user to change the size and description of any partition while allowing unaffected partitions to continue processing.

Operation	Operand
{DEFINE N}	[LIST]

LIST
: specifies that the current partition definitions will be listed. All job classes currently being serviced are listed also.

DISPLAY--CAUSE CONSOLE DISPLAY

The DISPLAY command is used primarily to cause a console display of the name of each job at the time it is initiated, and at the time it is terminated. The command provides the job name information needed for effective use of the CANCEL command and, together with system messages, keeps the operator informed of which jobs are currently being executed.

In addition, the DISPLAY command is used to cause a console display of the time of day and the date.

Operation	Operand
{DISPLAY D}	{STATUS JOBNAMES T A Q N jobname}

STATUS
: specifies that the data set names and volume serial numbers of data sets with dispositions of KEEP, CATLG, or UNCATLG are to be displayed on the console at step and job termination.

JOBNAMES
: specifies that the name of each job is to be displayed both when the job starts and when it terminates, and that unit record allocation is to be displayed when the step starts.

T
: specifies that the time of day and the date are to be displayed in the following format:

```
hh.mm.ss                 yy.ddd
 |  |  |                  |  |
 |  |  |                  |  |
 |  |  └-Seconds          |  |
 |  └----Minutes          |  └----Day
 └-------Hours (00-23)    └-------Year
```

A
: specifies that the jobname and its partition is to be displayed for each job currently running.

Q
: specifies that a listing of the number of entries on each of the input, hold, and output queues is to be displayed.

N
: specifies that a listing of jobnames on the hold and input queues is to be displayed.

jobname
: specifies that the queue location, priority, and position on the queue is to be displayed for the job specified.

Note: If STATUS, JOBNAMES, T, A, Q, or N is used as a jobname it must be enclosed in quotation marks.

HALT--PREPARE FOR POWER-OFF

The HALT command is used before you turn the power off at the end of the day, or anytime the computer is not to continue under the control of the operating system. You must use this command to ensure that important statistics and data records in main storage are not lost permanently.

Operation	Operand
{HALT / Z}	EOD

EOD
: specifies that end-of-day storing is to be done of internal input/output device error counts. The information is stored in the SYS1.LOGREC data set (see the topic "Hardware Debugging Aids" in Chapter 4 of IBM System/360 Operating System: Operator's Guide, Form C28-6540).

When the storing is done, the system sends you a message, EOD SUCCESSFUL. At this point, you can safely turn the power off.

HOLD--TEMPORARILY SUSPEND JOB SELECTION

The HOLD command, in conjunction with the RELEASE command, temporarily prevents one job, or all jobs in the input work queue from being selected for processing. If the named job has already been selected, or if it is not in the input work queue, a message will be received. Jobs temporarily suspended by HOLD are subject to CANCEL and RESET commands.

Operation	Operand
{HOLD / H}	{jobname / Q}

jobname
: specifies the name of the job whose selection is to be suspended. The maximum length of a job name is eight characters. Although any job name can be in apostrophes, a job with the name Q must have the Q in apostrophes in the command statement.

Q
: specifies that selection of all jobs from the input work queues is to be suspended.

MODIFY--ALTER OUTPUT WRITER

The MODIFY command is used to change the characteristics of an operating output writer. The operator can change the output classes associated with the output writer, change the conditions under which the output writer pauses for servicing of its device, or both.

Whenever the output writer pauses, it writes a message requesting the operator to perform any necessary action on its device. If the pause results from a new form number specification, the operator is given the form number.

Operation	Operand
{MODIFY / F}	{procedurename,taskname / taskname} [,CLASS(classname,...)] [,PAUSE={FORMS / DATASET}]

procedurename
: specifies the name of a cataloged procedure to start a writer.

taskname
: specifies the name of a writer (corresponding to the taskname which was specified in the START command). (See the "taskname" description under the START command.)

CLASS(classname,...)
: specifies one to eight single-character names of the classes to be associated with the output writer. If only one classname is specified, the parentheses may be omitted. If more than one classname is specified, the writer treats the specified classes on a priority basis, where the left-most character indicates the highest-priority output class. Do not use a comma after the final classname.

PAUSE=FORMS
: specifies that the output writer is to pause when a change in forms on its device is necessary.

PAUSE=DATASET
: specifies that the output writer is to pause before starting to process each data set.

MOUNT--ALLOCATE DEVICE

The MOUNT command is used to allow allocation of an input/output device to all job steps that require a particular volume, without intervening dismountings and remountings of that volume.

Operation	Operand
{MOUNT / M}	unitname [{,VOL=(NL,serial) / ,VOL=(SL,serial)}] [,USE={STORAGE / PUBLIC / <u>PRIVATE</u>}]

unitname
: specifies the name of the input/output device to be allocated. In systems with MFT II, you can specify a loaded or an unloaded device. The system will request that the correct volume be loaded if it is not already mounted.

VOL=(NL,serial)
: specifies that the volume does not have labels. The alphameric serial number, up to six characters long, is used for allocation references. This parameter is not used for direct-access volumes.

VOL=(SL,serial)
: specifies that the volume has standard labels. The alphameric serial number, up to six characters long, is used in label checking and for allocation references.

USE=STORAGE or PUBLIC or PRIVATE
: specifies that a direct-access volume will be used as either a storage, private , or public volume. The default for this parameter is PRIVATE. A storage volume is the most freely allocated kind of volume, open to use by the largest variety of data sets, temporary or non-temporary. Slightly restricted is a public volume, which can be allocated freely for temporary data sets, but which must be specified by volume serial number to be allocated to non-temporary data sets. A private volume is the least freely allocated kind of volume; it is allocated only if its volume serial number is specified.

RELEASE--MAKE JOB AVAILABLE FOR SELECTION

The RELEASE command is used to resume job selection that has been suspended by the HOLD command. If the job is in the input queue in a canceled status, or if the job is not found, you will receive a message.

Operation	Operand
{RELEASE / A}	{jobname / Q}

jobname
: specifies the name of the job to be made available for processing. The maximum length of a job name is eight characters. Although any job name can be in apostrophes, a job with the name Q <u>must</u> have the Q in apostrophes in the command statement.

Q
: specifies that all jobs in the input work queues are to be made available for processing.

REPLY--REPLY TO INFORMATION REQUEST

The REPLY command is used to reply to messages from the operating system and from problem programs that request information.

The REPLY command need not directly follow the message requesting the reply. The message id ensures that the message is routed by the system to the correct job.

Operation	Operand
{REPLY / R}	id,'text'

id
: specifies a numeric 2-character message identification field of the message requesting the reply.

text
: specifies the text to be entered in response to a message. The information passed to the program expecting the reply does not include the enclosing apostrophes.

RESET--CHANGE PRIORITY OF JOB

The RESET command is used to change the selection priority of a job in the input work queue. If the job has already been selected for processing when the command is issued, the priority is not changed, and you will receive a message. You will also receive a message if the job is not found in the input queue.

Operation	Operand
{RESET / E}	jobname,value

jobname
: specifies the name of the job whose priority of selection is to be changed. The maximum length of a job name is eight characters.

value
: specifies the value to which the job's priority is to be set. The value is a two-digit numeric field that may range from a low of 00 to a high of 14.

SET--SET DATE, TIME, AND LOCATION

The SET command is used to establish the date, the time of day, the device for the input work queue and whether the queue is to be formatted, the location of the procedure library, or the automatic commands you wish to override. Any combination of these may be specified.

Use of the CLOCK operand in systems with MFT II is suggested if the timer option is included. The system uses the data in that operand when they name system data sets.

Operation	Operand
{SET / T}	DATE=yy.ddd,[CLOCK=hh.mm.ss] [,Q=(unitname[,F])] [,PROC=unitname] [,AUTO=characters]

DATE=yy.ddd
: specifies the date in the following format:

```
yy.ddd
|  |
|  |
|  └----Day (001-366)
|
└--------Year (00-99)
```

CLOCK=hh.mm.ss
: specifies the time of day in the following format:

```
hh.mm.ss
|  |  |
|  |  |
|  |  └--Seconds (00-59)
|  |
|  └-----Minutes (00-59)
|
└--------Hours (00-23)
```

If the new clock setting implies a change of date, the new date must be explicitly stated using the DATE parameter.

Q=(unitname[,F])
: specifies the name of the direct-access device on which the volume containing the input work queues (SYS1.SYSJOBQE) is to reside. Space on this volume for the input work queues must have already been allocated. If this parameter is not given, the system assumes that the queues are on the system residence volume. (This parameter is used only in the initial SET command issued immediately after IPL.) When the input work queues are to be used for the first time, a comma and an F must be added to the 3-character unitname. The F is a request to the system to format the job queue data set prior to the first job initiation; it must be used during the first IPL, but need not be used at subsequent IPLs. If the F is used at subsequent IPLs, it will cause reformatting each time it is used. If the job queue data set is on the system residence volume, the formatting operand can be shortened to Q=[,F]. If the F is omitted or if the entire parameter is omitted, job queue data set information which had been entered during a previous IPL is saved. When restarting the system to save information all auxiliary storage volumes which were in use must remain available.

PROC=unitname
: specifies the name of the direct-access device on which the volume that contains the procedure library resides. If this parameter is not given, the system assumes that the procedure library is on the system residence volume. (This parameter is used only in the initial SET command issued immediately after IPL.)

AUTO=characters
: specifies, in systems with automatic START commands, whether you wish to retain any of those commands. For each automatic command printed out by the system, follow the equals sign by a Y if you want to retain the command, or by an N if you want to override the command. For example, if the system prints out S RDR, S WTR, S INIT, and you want to retain the automatic reader and writer but not the initiator, key in AUTO=YYN. If you want to reject all automatic commands, key in AUTO=NONE.

START--START SYSTEM PROCESS

The START command is used to start readers and writers.

```
┌──────────┬─────────────────────────────────┐
│Operation │Operand                          │
├──────────┼─────────────────────────────────┤
│ {START}  │procedurename(taskname)          │
│ {  S  }  │[,devicename]                    │
│          │[,volumeserial]                  │
│          │                                 │
│          │ [ {,jobname      } ]            │
│          │ [ {,(classnames) } ]            │
│          │ [ {,parmvalue    } ]            │
│          │                                 │
│          │[,keyword=option,...]            │
└──────────┴─────────────────────────────────┘
```

procedurename
: specifies the name of a catalogued procedure which describes an output writer or an input reader.

taskname
: specifies one of the following:

Pn
: where n is a partition number (0 through 51) which specifies the partition in which the reader or writer is to be started.

S
: specifies that the reader or writer is to be system-assigned. Only one <u>system-assigned</u> reader or writer can be active in a system at any one time.

The taskname identifies the name of a reader or writer when multiple tasks are STARTed with the same procedurename and either the devicename is not specified or is the same for more than one task.

devicename
: specifies the name of the input or output device associated with an input reader or output writer. This can be either a unit name (such as 293) or a general name (such as 2400). If specified, the name will override any corresponding unit specification in the cataloged procedure.

volumeserial
: specifies the six-character serial number of a magnetic tape or direct-access volume. If specified, this parameter overrides any corresponding volume serial specification in the cataloged procedure. Do not specify a serial number for a direct-access volume when starting a writer.

jobname
: specifies the name of a job in the input stream. This parameter is used only when starting an input reader. Use this parameter to cause forward spacing through the input stream until the name job is met. The maximum length of a job name is eight characters.

classnames
: specifies one to eight single-character system output classnames. This parameter is used only when starting a writer. If one class name is specified, no parentheses are needed. If more than one is specified, enclose the group in parentheses, separating the names by commas; for example, (A,B,C). When starting an output writer, use this parameter to limit the system output classes the writer will process. The writer treats the specified classes on a priority basis where the leftmost character indicates the highest priority output class. If this operand is used, it overrides <u>all</u> classnames specified in the standard cataloged procedure.

parmvalue
: specifies parameters to be passed to the program receiving control as a result of a START command.

keyword=option
: specifies any appropriate keyword syntax allowable on a DD statement. (For detailed information on these keywords, refer to <u>IBM System/360 Operating System: Job Control Language</u>, Form C28-6539.) If such keyword parameters are specified, they will override the corresponding parameters on the DD statement for the input or output device in the cataloged procedure. If the devicename positional parameter is used, the UNIT keyword may not be used. If the volumeserial positional parameter is used, the VOLUME keyword may not be used.

<u>Note:</u> To make clear to the system which operands you are specifying when you are not specifying all of them, insert commas for the missing operands in the middle of the string. For example,

```
START WRITER(P1),185,,C
```

lets the system know that you are not specifying the volumeserial operand.

START INIT--START INITIATOR

Operation	Operand
{START / S}	INIT {(ALL) / (Pn) [,,,job class]}

INIT
: specifies that an initiator is to be started.

ALL
: specifies that initiators be started in all partitions.

Pn
: specifies the partition in which the initiator is to be started, where n is a partition number (0-51).

job class
: specifies up to 3 job class identifiers (A-O).

STOP--STOP SYSTEM PROCESS

The STOP command is generally used to stop the operation of an input reader, an output writer, or an initiator, or to stop a console display of job names.

Operation	Operand
{STOP / P}	{procedurename,taskname / taskname / INIT,Pn / JOBNAMES / STATUS}

procedurename
: specifies the name of a cataloged procedure defining an input reader or output writer.

taskname
: specifies the name of a reader or writer to be STOPped (corresponding to the taskname specified in the START command). (See the "taskname" description under the START command.)

INIT
: specifies that an initiator is to stop after initiating the current job.

Pn
: specifies the partition number (0-51) of the initiator which is to be stopped; e.g., P4.

JOBNAMES
: specifies that a console display of the names of jobs, initiated by the JOBNAMES parameter of the DISPLAY command, is to be terminated. For more information about JOBNAMES, see the discussion of the DISPLAY command.

STATUS
: specifies that the console display of the data set names (with their volume serial numbers and dispositions of KEEP, CATLG, or UNCATLG), initiated by the STATUS parameter of the DISPLAY command, is to be terminated.

UNLOAD--PREPARE VOLUME FOR DISMOUNTING

This command is normally used to remove a volume previously mounted in response to a MOUNT command.

The UNLOAD command causes a volume on an input/output device to be prepared for dismounting. When the volume is ready to be dismounted, the operator will receive a message. (The message may not be received until the current job is completed.)

Operation	Operand
{UNLOAD / U}	unitname

unitname
: specifies the unit address of the input/output device to be prepared for dismounting.

VARY--VARY STATUS OF DEVICE

The VARY command is used to place an input/output device (other than a communications line) into an online or offline status.

Operation	Operand
{VARY / V}	unitname {,ONLINE / ,OFFLINE}

unitname
: specifies the unit address of the input/output device whose status is to be changed.

ONLINE
: specifies that the device is to be made available for allocation by the job scheduler to problem programs.

OFFLINE
specifies that the device is to be removed from the recognition of the job scheduler, and that any further allocation of the device to problem programs is to be prevented. If the device is in use (allocated to a problem program or to an input reader or output writer), the status is not changed until all the current users have finished with the device. When the status is changed to offline, you will receive a message. A device can be removed from the offline status only by a subsequent VARY command.

OPERATOR MESSAGES

This topic describes both problem program and control program messages as they will appear to the operator of an MFT II system. "Problem Program Message Formats" describes the addition of partition identifiers to messages generated by problem programs. "Control Program Messages" describes system messages associated with facilities or conditions unique to MFT II.

PROBLEM PROGRAM MESSAGE FORMATS

To avoid confusion in MFT II, it is necessary to relate WTO and WTOR messages to the problem program which issued them. In order to do this, a partition identifier will be added to each message issued by a problem program partition. The maximum length of WTO messages is 122 characters. The maximum length of WTOR messages is 119.

Examples: Currently WTO messages appear in the following form:

    PHASE A ENTERED

The new form will be:

    P3 PHASE A ENTERED

Similarly, WTOR messages will be changed from:

    id REPLY 8 CHARACTER NAME

to the following format:

    id P3 REPLY 8 CHARACTER NAME.

CONTROL PROGRAM MESSAGES

Messages associated with facilities or conditions unique to MFT II are described below. Messages associated with other portions of the system such as the nucleus initialization program (NIP), data management, and job management are the same as in other configurations of the Operating System. These messages are described in IBM System/360 Operating System: Messages, Completion Codes, and Storage Dumps, Form C28-6631.

IEE601D CHANGE PARTITIONS?

Explanation: The nucleus initialization program (NIP) has finished processing. This message asks if partitions are to be redefined, or if a listing of partition definitions and job classes being serviced is desired.

Operator Response: If partitions are not to be redefined, enter

    REPLY id,'NO'

If partitions are to be redefined, enter

    REPLY id,'YES'

If a list of current partition definitions is desired, add

    ,LIST

to the reply.

This causes the list to be printed in the form specified in message IEE604I. Message IEE616I is printed also.

IEE602A ENTER DEFINITION

Explanation: In response to this message the operator can redefine partition sizes, request a listing of all partition sizes as altered, indicate the end of partition redefinition, request a listing of all job classes being serviced, or cancel all redefinitions which have been entered in this series.

Operator Response: Enter partition definitions. (See "Entering Partition Definitions" in this section.)

IEE603A CONTINUE DEFINITION

Explanation: The definition just entered has been accepted by the system. Redefinition may continue.

Operator Response: Enter partition definitions. (See "Entering Partition Definitions" in this section.)

IEE604I Pn=(SIZE,CLASS) OR Pn=(INACTIVE)

<u>Explanation</u>: The operator requested a listing of partition definitions. Two partition definitions are printed per message. N is the partition number. Size is the size of the partition expressed in decimal notation (if the system has storage protection, it is a multiple of 2K). Class is the job class(es) associated with each partition, or it specifies a reader partition (R or RDR) or a writer partition (W or WTR). Class may be followed by ,LAST indicating that all higher numbered partitions are inactive. INACTIVE indicates that the partition has been defined as having a size of zero.

<u>Operator Response</u>: None.

IEE605I DEFINITION COMPLETED

<u>Explanation</u>: When redefining partitions, END was specified in the last entry. The message DEFINITION COMPLETED is printed after the redefinitions have been implemented by the system.

<u>Operator Response</u>: Enter a START INIT command for each of the newly defined partitions. START INIT, ALL may be used.

IEE606I TOTAL SIZE OF PARTITIONS IS XXXXX BYTES TO LARGE FOR STORAGE

<u>Explanation</u>: After partition redefinition, the total size of all partitions exceeds total storage capacity. This message is followed by message IEE603A so that the error can be corrected.

<u>Operator Response</u>: See message IEE603A.

IEE607A DEFINITION PARAMETER ERROR, REPLY AGAIN

<u>Explanation</u>: The size or job class of a partition definition has been specified incorrectly, RDR has been specified for a small partition, LAST has been specified more than once in the same statement, or some other syntax error has been entered.

<u>Operator Response</u>: Reenter the partition definition.

IEE608A Pn NOT DEFINABLE -- REPLY AGAIN

<u>Explanation</u>: The partition designated by n was not defined at system generation.

<u>Operator Response</u>: Redefine partitions, omitting the undefinable partition.

IEE609I PARTITIONS DEFINED EXCEED AVAILABLE SPACE

<u>Explanation</u>: When redefining sizes of adjacent partitions, the total new space specified is greater than the original space. This message is followed by message IEE603A which allows corrections to be made.

<u>Operator Response</u>: See message IEE603A.

IEE610A PROBLEM PROGRAM PARTITIONS EXCEED 15, RESPECIFY

<u>Explanation</u>: More than fifteen problem program partitions have been specified.

<u>Operator Response</u>: Redefine partitions.

IEE611A CHANGE PARTITIONS NOT ADJACENT, RESPECIFY

<u>Explanation</u>: The operator attempted to redefine nonadjacent partitions.

<u>Operator Response</u>: Redefine partitions.

IEE612I Pn HAS XXXXX EXCESS BYTES ADDED

<u>Explanation</u>: The total space specified for all partitions defined is less than the total space available.

<u>System Action</u>: All excess space is added to the lowest priority problem program partition which was defined.

<u>Operator Response</u>: None.

IEE614I DEFINITION CANCELLED

<u>Explanation</u>: The operator requested that all partition redefinitions subsequent to an ENTER DEFINITION message be cancelled.

<u>System Action</u>: Partition redefinitions are cancelled. No further redefinitions will be accepted

unless a DEFINE command is entered first.

Operator Response: None

IEE615A DEFINITION DELIMITER ERROR, REPLY AGAIN

Explanation: The operator incorrectly indicated parentheses, commas, blank spaces, or some other delimiter when defining a partition.

Operator Response: Reenter the partition definition.

IEE616I CLASSES= JOB CLASS

Explanation: This message is printed in response to a request for a listing of all job classes currently being serviced. Job class is a listing of the job classes of all active partitions.

Operator Response: None.

Note: Message IEE613 is not used.

Where more than one page reference is given, the first page number indicates the major reference.

C27-6939-0

Printed in U.S.A. C27-6939-0

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

READER'S COMMENTS

IBM System/360 Operating System: Planning for Multiprogramming With a Fixed Number of Tasks; Version II (MFT II)

C27-6939-0

Your comments will help us produce better publications for your use. Please check or fill in the items below, adding explanations and other comments in the space provided. All comments and suggestions become the property of IBM.

Which of the following terms best describes your job?

☐ Programmer	☐ Systems Analyst	☐ Customer Engineer
☐ Manager	☐ Engineer	☐ Systems Engineer
☐ Operator	☐ Mathematician	☐ Sales Representative
☐ Instructor	☐ Student/Trainee	☐ Other (explain) ____________

Does your installation subscribe to the SRL Revision Service? ☐ Yes ☐ No

How did you use this publication?

- ☐ As an introduction
- ☐ As a reference manual
- ☐ As a text (student)
- ☐ As a text (instructor)
- ☐ For another purpose (explain) ________________________________

Did you find the material easy to read and understand? ☐ Yes ☐ No (explain below)

Did you find the material organized for convenient use? ☐ Yes ☐ No (explain below)

Specific criticisms (explain below)

Clarifications on pages ____________________
Additions on pages ____________________
Deletions on pages ____________________
Errors on pages ____________________

Explanations and other comments:

No postage necessary if mailed in U.S.A.

C27-6939-0

FOLD FOLD

FIRST CLASS
PERMIT NO. 116
KINGSTON, N. Y.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

POSTAGE WILL BE PAID BY

IBM CORPORATION
NEIGHBORHOOD ROAD
KINGSTON, N. Y. 12401

ATTN: PROGRAMMING PUBLICATIONS
DEPARTMENT 637

FOLD FOLD

Printed in U.S.A. C27-6939-0

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]