A SCHEDULING PROGRAM FOR A
MULTIPROGRAMMED COMPUTER

by
HERBERT JAY BOROCK

B.S., Columbia University
(1963)

SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF
SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June, 1967

Signature of Author

Alfred P. Sloan School of Management, May 19, 1967

Certified by.... Thesis Supervisor

Accepted by.... Chairman, Departmental Committee on Graduate Students
This thesis has been given limited reproduction with the permission of the Alfred P. Sloan School of Management. However, this copy is issued with the understanding that none of the data resulting from this investigation will be used for advertising or publicity purposes, and that the thesis is solely for the confidential use of the organization or individual to whom it is addressed.
A SCHEDULING PROGRAM FOR A MULTIPROGRAMMED COMPUTER

by

HERBERT JAY BOROCK

Submitted to the Alfred P. Sloan School of Management on May 19, 1967 in partial fulfillment of the requirements for the degree of master of science.

ABSTRACT

This thesis describes a scheduling program for an IBM System/360 computer operating in a multiprogramming environment provided by the IBM System/360 Disk Operating System. In this environment up to three programs compete for the use of the computer's resources.

A scheduling rule is developed for computer center operations managers in the form of a computer program. The program saves substantial amounts of computer time since it provides more efficient schedules for the multiprogramming environment than could be provided by the operations manager without using the program. Input to the program consists of statements describing the computer, the installation, and the programs to be scheduled. Output from the program consists of a suggested schedule, showing program sequence and clock time for the scheduled period, which is a week.

The problem of scheduling the computer is solved by dividing the problem into two smaller scheduling problems—a job shop scheduling problem and a flow shop scheduling problem. In the job shop scheduling problem, three programs compete for the use of the computer's central processing unit and input/output channels. Given each program's characteristics, the time required to complete each program is calculated by simulating the operation of the computer.

In the flow shop scheduling problem, each program can use up to three areas of computer storage for program execution. This is an n-job, 3-machine sequencing problem, where each program is a job and each area of computer storage is a machine. Previous algorithms for the flow shop scheduling problem are used to sequence the programs for the class of problems in which two machines cannot operate on the same job in parallel. New branch and bound algorithms are developed for the class of three-machine problems in which two of the machines can operate on the same job in parallel.

Thesis Advisor: Donald C. Carroll

Title: Associate Professor of Management
May 19, 1967

Professor Edward N. Hartley
Secretary of the Faculty
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Dear Professor Hartley:

In accordance with the requirements for graduation, I herewith submit a thesis entitled "A Scheduling Program for a Multiprogrammed Computer."

I want to take this opportunity to thank the many people who assisted me in my research.

The National Dairy Products Corporation provided the financial assistance for this work. Paul A. Strassmann, Director of Systems and Procedures, suggested the topic. Robert I. Benjamin, Manager of Installation Support offered many helpful suggestions.

My thesis committee, consisting of Professors Donald C. Carroll and John F. Pierce, were a constant source of encouragement and were always available for advice.

Computer time was supplied by the M.I.T. Computation Center and the Computer Facility of the Alfred P. Sloan School of Management.

Mrs. Anita Ciminero typed the final draft.

Sincerely,

Herbert Jay Borock
# TABLE OF CONTENTS

Chapter 1. INTRODUCTION ......................................................... 9  
Previous Research ........................................................................ 10  
The Problem ............................................................................... 12  
The Solution ............................................................................. 13  
The Scheduling Program as a Management Control Device ................. 13  

Chapter 2. SCHEDULING ENVIRONMENT ........................................... 15  
The Computer Components .......................................................... 15  
The Programs ............................................................................. 18  
The Weekly Schedule .................................................................. 19  
The Job Shop Scheduling Problem ............................................... 20  
The Flow Shop Scheduling Problem .............................................. 22  

Chapter 3. SIMULATION METHOD .................................................. 24  
Two Component Example ............................................................. 24  
Use with the Flow Shop Scheduling Problem ................................... 30  
General Case ............................................................................. 31  
Example .................................................................................... 32  

Chapter 4. PREVIOUS ALGORITHMS .............................................. 39  
The N-Job, 2-Machine Sequencing Problem .................................... 40  
The N-Job, 3-Machine Sequencing Problem .................................... 41  
Use of the Algorithms in Scheduling the Computer ......................... 44
Chapter 5. BRANCH AND BOUND ALGORITHMS
Parallel Operations on Input
Parallel Operations on Output
Case 1: Input, Different Machines
Case 2: Output, Different Machines
Case 3: Input, Identical Machines
Case 4: Output, Identical Machines
Case 5: Input, Same Operation, Different Machines
Case 6: Output, Same Operation, Different Machines

Chapter 6. SCHEDULING RULE

Chapter 7. CONCLUSIONS AND SUGGESTIONS FOR FUTURE RESEARCH

Appendix A. PROGRAM USER MANUAL
Input Formats
Output Formats

Appendix B. PROGRAM LOGIC MANUAL
Program Organization
Storage Requirements
Flowcharts
FORTRAN Source Statements

BIBLIOGRAPHY
LIST OF ILLUSTRATIONS

Figure 1. Component use as a fraction of total program time....25
Figure 2. Component use as a fraction of one time unit.........26
Figure 3. Total computer use by program 2 during one time unit.27
Figure 4. Component use by program 2 during one time unit.....27
Figure 5. Component use by programs 1 and 2 during one time
unit..................................................28
Figure 6. Total computer use by program 3 during one time unit.29
Figure 7. Component use by program 3 during one time unit.....29
Figure 8. Component use as a fraction of total program time....33
Figure 9. Component use as a fraction of one time unit.........34
Figure 10. Total computer use by program 2 during one time unit.35
Figure 11. Component use by program 2 during one unit of time...36
Figure 12. Component use by programs 1 and 2 during one time
unit..................................................36
Figure 13. Total computer use by program 3 during one time unit.38
Figure 14. Component use by program 3 during one time unit.....38
Figure 15. Tree structure for 4-job, 3-machine problem........43
Figure 16. Tree structure for 4-job, 3-machine problem, with
parallel operations on different machines, input...48
Figure 17. Tree structure for 4-job, 3-machine problem, with
parallel operations on different machines, output.50
Figure 18. Tree structure for 4-job, 2-machine problem, with
two identical machines.............................52
Figure 19. Tree structure for 4-job, 3-machine problem, with
two identical machines, input.......................54

Figure 20. Tree structure for 4-job, 3-machine problem, with
two identical machines, output.....................55

Figure 21. Tree structure for 4-job, 2-machine problem, with
identical operations on different machines...........57

Figure 22. Tree structure for 4-job, 3-machine problem, with
identical operations on different machines, input.58

Figure 23. Tree structure for 4-job, 3-machine problem, with
identical operations on different machines, output.59

Figure 24. Overall program organization......................80

Figure 25. Flowchart symbols ...................................88
Chapter 1

INTRODUCTION

Scheduling a multiprogrammed computer has been recognized for some time as analogous to scheduling the work in a factory. However, little practical work has been done to assist computer center operations managers in scheduling programs run on a multiprogrammed computer. Part of the reason for this is that until recently multiprogrammed computers were used in only large computer installations. Further, when researchers tried to attack the problem they dealt with hypothetical computers. With the introduction of the IBM System/360, multiprogramming is being extended to smaller sized installations. The typical business computer installation is now faced with the problem of scheduling work on a multiprogrammed computer.

In most cases the installation chooses a manufacturer-supplied programming system which provides no scheduling assistance, except for job-to-job transitions. All the programming system does is free the operator from some routine work so that he can devote more time to other routine work. No assistance is given in actually scheduling the jobs, i.e., deciding which job sequence results in the most efficient use of the computer. The purpose of this research is to provide the computer center operations manager with guidance in this area. Using the scheduling program developed here, the manager will be able to obtain good or "near" optimal schedules.
Previous Research

Lynch\(^1\) described the operation of a university computing center. In that environment, the majority of the programs use less than 5 minutes of computing time. Less than 5\% of the programs use any input/output devices other than for card input and printer output. All of the programs are limited by computing time, rather than input/output time. Use of the computer is allocated to different groups of individuals on a priority basis.

Hutchinson\(^2\) described the simulation of an entire computing center, including accounting-in and accounting-out operations, and the use of a tape library. The scope of his study is limited to large aerospace-manufacturing computing centers. He varied manpower levels and equipment to determine the most efficient center. His measure of service was the amount of time to process jobs.

McKenney\(^3\) developed a system which defined computer jobs so that they could be recursively segmented into independent subjobs. An independent subjob is defined by the source of its inputs, the definition

---


of its outputs, and a relation between its inputs and outputs. He concluded that computer programs should not be restricted by arbitrary sequence constraints, so that they could be segmented into independent subjobs, which could be processed by different computer components in parallel.

Heller\(^4\) used a linear graph technique to indicate the use of each computer component and the precedence relationships among subprograms. He formulated the problem; he did not solve it.

Schwartz\(^5\) studied the feasibility of restructuring programs to take advantage of multiple facility computers. He attempted to develop a procedure by which the computer could restructure the programs and then schedule them. He was concerned with the network properties of computer programs and the derivation of a precedence matrix.

Codd\(^6,7\) developed an algorithm for a multiple facility system. He considered two kinds of facilities--those that were time-shared, and those that were space-shared. The purpose of his algorithm was to schedule a set of programs so that they could be executed in a minimum amount of time.


The Problem

The computer we are concerned with is used by business concerns for useful work on a continuing basis. The work in these installations consists of high volume input and output. It is not uncommon for a single program to include hours of card reading and printing. Since these input/output operations are slow relative to other processing, systems programs have been developed to provide a computer environment where one job's processing can be overlapped with other jobs' input/output operations. In this way the work load can be balanced so that one job is not delayed by excessive input, say, and another is not delayed by excessive processing.

The computer environment we are concerned with consists of three programs being multiprogrammed together. One program is a processing program, and the other two programs are input/output programs. The majority of the jobs on the computer use the processing program area of storage and at least one of the input/output programs.

Most jobs enter the computer installation at the beginning of one day and must be completed at the end of that day. The remainder must be completed within the same week they enter the installation. Other than having different due dates, there is no priority among jobs.

This environment poses two problems. First, how much time does each program take to run on the computer? The time is easy to calculate if there is only one program using the computer, since the input for each program consists of a known number of transactions, and the processing time per transaction and input/output time per transaction are known. However, in this environment three programs are sharing
the computer, thus complicating the timing problem.

Second, given that we know how long each program takes, how do we schedule the programs to make efficient use of the computer? The scheduling problem is easy if each program uses the computer only once. However, in this environment each program may use the computer three times—once for processing, and twice for input/output.

The Solution

The two problems are solved separately. The timing problem is treated as a job shop scheduling problem where three programs compete for the use of the computer's central processing unit and input/output channels. Given each program's characteristics (e.g., number of machine instructions per input record), the time required to complete each program is calculated by simulating the operation of the computer.

The scheduling problem itself is treated as a flow shop scheduling problem where each of three areas of computer storage is analogous to a machine. Each time a program uses the computer it uses a different machine. This is the familiar n-job, 3-machine sequencing problem. However, there is a substantial difference. Both input/output areas may perform the same function, i.e., they may be identical machines, or at the very least, a single program may use them in parallel. New branch and bound algorithms are developed for the parallel operation cases.

The Scheduling Program as a Management Control Device

The scheduling rule developed here is provided to computer center operations managers in the form of a computer program. The program is
written for the IBM System/360, the same computer that is in the computer center. The program saves substantial amounts of computer time since it provides more efficient schedules for a multiprogramming environment than could be provided by the operations manager without using the program.

Input to the program consists of statements describing the computer, the installation, and the programs to be scheduled. The program descriptions, which are provided by the programmers who design the programs, include such things as the usage of various input/output devices.

Output from the program consists of a suggested schedule, showing program sequence and clock time for the scheduled period, which is a week. The operations manager has the opportunity to revise the schedule after which the schedule becomes his plan for processing the work in the computer center. The actual performance of the computer center and its manager can then be compared to the planned performance, so that more efficient and effective performance can be produced.
Chapter 2
SCHEDULING ENVIRONMENT

A scheduling environment consists of resources, jobs, and a scheduling rule. The resources and jobs in this scheduling environment are the components of a computer and the programs run on the computer, respectively. They are described in this chapter. The scheduling rule is described in Chapter 6.

The Computer Components

The computer for which this scheduling program is written is the IBM System/360. A System/360 consists of main storage, a central processing unit (CPU), up to six selector channels, one multiplexor channel, and input/output devices attached to the channels. Programs and data reside in main storage. The CPU is used to execute a program's instructions and to transfer data between the input/output units and main storage via the channels.

The channels are used to relieve the CPU of the task of communicating directly with the input/output units and permit instruction execution to occur concurrently with data transfer. A selector channel transfers data to or from a single input/output device at a time and is capable of handling high speed input/output devices. The multiplexor channel can operate in either of two modes: multiplex mode for low-speed devices, and burst mode for high speed devices.

In the multiplex mode, a large number of low-speed input/output devices time-share the channel's single data path, and the channel simultaneously transfers data between all the devices and main storage. In the burst mode, a single high-speed device uses the channel's data path, and that device does not relinquish the channel until its data transfer operation is complete.

The CPU can concurrently execute a machine instruction, transfer data to or from a single input/output device on each of the selector channels, and transfer data to or from either a single high-speed device on the multiplexor channel (burst mode) or multiple low-speed devices on the multiplexor channel (multiplex mode).

System/360 is designed to be multiprogrammed. A multiprogrammed System/360 is one in which two or more programs reside in main storage and compete for the use of the CPU, the selector channels, and possibly the multiplexor channel. They compete for the use of the multiplexor channel if at least one program wants to use that channel in the burst mode; otherwise, all programs can use the multiplexor channel at the same time.

A programming system is used to control the operation of the computer and to allocate the use of the CPU and the channels among the programs. The programming system for which this scheduling program is written is the IBM System/360 Disk Operating System (DOS/360).

DOS/360 provides a multiprogramming environment where main storage is divided into three areas, each of which can contain a single program. Programs are loaded into one of the areas of main storage.

2 For a more detailed discussion see IBM System/360 Disk Operating System: System Control and Service Programs (Endicott: IBM Corporation, 1966).
storage (called the background area) by DOS/360 as they are needed, without operator intervention. Programs are loaded into the other two areas of main storage (called foreground area one and foreground area two) by the operator.

When they compete for the CPU and the channels, the program in foreground area one has the highest priority, the program in foreground area two has the next highest priority, and the program in the background area has the lowest priority. Control of the CPU is given to the highest priority program which is not waiting for the completion of an input/output operation. That program can then issue machine instructions and initiate input/output operations.

If the program issues a multiplex mode input/output operation, and the multiplexor channel is operating in the multiplex mode, the input/output operation is executed immediately. If the program issues a burst mode input/output operation and the multiplexor channel is in use, or a multiplex mode input/output operation and the multiplexor channel is operating in burst mode, or an input/output operation to a selector channel and the channel is in use, then the input/output operation is executed when the channel finishes executing the previous input/output operation issued to it.

If the program cannot execute any more machine instructions until a particular input/output operation is complete, the program relinquishes control of the CPU to a lower priority program which is not waiting for the completion of an input/output operation. When the input/output operation is complete, the higher priority program seizes control of the CPU from the lower priority program.
The Programs

The programs run on the computer are either foreground programs or background programs, depending on the area of main storage in which they are executed.

The foreground programs are used to transfer data between a low-speed device on the multiplexor channel and a high-speed device. Since the operation of the low-speed device takes much longer than the instructions used to initiate the input/output operations on both devices, only a small portion of the total program time is used to execute instructions on the CPU. A somewhat longer time is used to transmit data over the channel to which the high-speed device is attached. Most of the time is used to transmit data over the multiplexor channel in the multiplex mode. This permits the background program to use the CPU and the channels while the foreground programs are waiting for the completion of input/output operations on the multiplexor channel.

Foreground programs may be used as input or output transcription programs in conjunction with background programs, or they may operate independently of the background programs. Both foreground programs may be used for the same function (e.g., output transcription) at the same time.

For example, a typical background program includes many low-speed input/output operations on the multiplexor channel for input transcription (e.g., card reading) and output transcription (e.g., printing). If the background program actually performed these operations, a substantial amount of the program time would consist of waiting for the completion of these operations as is the case with the foreground programs.
Instead of performing card reading and printing directly with the background program, a foreground program can transfer the input data to a high-speed device, the background program can then read the data from that device and put its output data on another high-speed device, which another foreground program can then transfer to the printer.

In this example, while one foreground program is reading input for a subsequent background program, another foreground program is printing the output for a previous background program, and a background program is being executed using data previously read by the first foreground program and supplying data which will subsequently be printed by the second foreground program. Depending on the relative volumes of input and output, both foreground programs may be used for either input or output, one foreground program may be used for either input or output while the other is doing independent work, or both foreground programs may be doing independent work.

Some sets of background programs represent a single accounting unit of processing in terms of DOS/360. The programs in each set must be executed in sequence, with no other programs intervening. These background program sets may operate in conjunction with foreground programs as described above.

**The Weekly Schedule**

In the installations we are concerned with, most of the programs enter the computer installation at the beginning of one day and must be completed by the end of that day. Other programs have different turnaround times, but all programs must be completed during the same calendar week that they enter the computer installation. Thus, each
week a static schedule must be produced for a group of foreground programs and background programs. Given descriptions of the computer (CPU, channel, and input/output device performance characteristics), the installation (computer configuration, number and normal use of foreground areas, and work week), and the programs (execution times, input/output volumes, device usage, and possible foreground program usage), the object is to produce an optimal or near optimal schedule for the week. During the week, conditions in the computer center may change, such that the original schedule is no longer the best schedule: new programs are added to the schedule; the operators make errors and programs have to be rerun; an input/output device goes down. In that case, a new schedule would have to be produced, given the descriptions of the computer, the installation, and the programs when the change occurs.

The problem of scheduling programs in this environment can be divided into two smaller scheduling problems—a job shop scheduling problem and a flow shop scheduling problem. The scheduling rule is in the form of a computer program which is run on the same computer for which the schedule is produced. As long as the computer is capable of performing useful work, then no matter what kinds of changes occur during the week, a new schedule can be produced.

The Job Shop Scheduling Problem

Each of the three programs being multiprogrammed can require the use of up to nine computer components: CPU for instruction execution, up to six selector channels for data transfer, multiplexor
channel for data transfer in burst mode, and multiplexor channel for data transfer in multiplex mode. At any one instant in time, only one program can use the CPU for executing instructions, only one program can use each of the selector channels to transfer data between main storage and an input/output device, and either only one program can use the multiplexor channel to transfer data in burst mode or all programs can use the multiplexor channel to transfer data in multiplex mode.

Given a program's description, it is easy to calculate the time to complete the program, if it is the only program on the computer. All one has to do is calculate its time on the CPU and on each of the channels, and then choose the longest of these times as the total program time, since the operations on these computer components are overlapped.

However, when more than one program is being run at the same time, the scheduling problem is more complex. More than one program is assigned to the same resource (multiplexor channel in multiplex mode) at the same time. More than one resource (e.g., CPU and a channel) is assigned to the same program at the same time. A program uses a resource (CPU or any channel) more than once.

It would be unwise to keep track of all the changes in resource assignment in order to calculate the time for each program in the multiprogramming environment. To do so would require simulating each machine instruction and input/output operation. Such a simulation would require more time to execute on a computer than would the schedule itself.
The approach taken here is first to calculate the total CPU time and channel time for each of the programs, assuming that each program is the only one on the computer. Then, using the DOS/360 program priorities, and the fact that the CPU time of one program can be overlapped with the channel time of that program and the other two programs, new program times are calculated. The simulation method used to calculate these times is described in Chapter 3.

The Flow Shop Scheduling Problem

Given the program times calculated in solution to the job shop scheduling problem, the problem of scheduling the week's work on the computer reduces to a flow shop scheduling program.

In the flow shop scheduling problem, each area of main storage represents a machine. The program times represent operation times on these machines. Jobs consist of from one to three operations, with no more than one operation on each machine. A three operation job consists of a background program run in conjunction with two foreground programs which read the background program's input and/or print its output.

If both foreground programs are used in conjunction with the background programs, then we have an n-job, 3-machine sequencing problem. If only one foreground program is used in conjunction with the background programs, then we have an n-job, 2-machine sequencing problem, since the other foreground program can always operate in parallel with both the first foreground program and the background program.
There are four distinct sequencing problems:

1. The n-job, 2-machine sequencing problem (background programs plus a foreground program for either input or output).
2. The n-job, 3-machine sequencing problem (background programs plus one foreground program each for input and output).
3. The n-job, 3-machine sequencing problem (background programs plus two foreground programs for input).
4. The n-job, 3-machine sequencing problem (background programs plus two foreground programs for output).

Algorithms which provide optimal solutions to problems 1 and 2 have been developed previously. They are described in Chapter 4. As far as I know, algorithms which provide optimal solutions to problems 3 and 4 have not been developed previously. Branch and bound algorithms for problems 3 and 4 are described in Chapter 5.
Chapter 3

SIMULATION METHOD FOR THE JOB SHOP SCHEDULING PROBLEM

Up to three programs may be multiprogrammed in the environment being simulated. For each program, the total CPU time and the total channel time for each channel are calculated, assuming that each program is the only one on the computer. The maximum of these times for each program is the total program time for that program in the non-multiprogrammed environment, since the operations on these computer components are overlapped. The times on these computer components are then expressed as fractions of the total program time for each program.

Two Component Example

Suppose there are three programs—program 1, program 2, and program 3, each of which uses only two computer components—the CPU and the multiplexor channel in the multiplex mode. When it is the only program using the computer, program 1 uses the CPU for 10 minutes and the multiplexor channel for 100 minutes; program 2 uses the CPU for 10 minutes and the multiplexor channel for 50 minutes; and program 3 uses the CPU for 45 minutes and the multiplexor channel for 150 minutes. Then in the non-multiprogrammed environment, the total program times for programs 1, 2, and 3 are 100 minutes, 50 minutes, and

---

This is true for most of the programs we are concerned with. Some programs, however, do have non-overlapped operations. The scheduling rule developed here provides for such programs by including the percentage of non-overlapped operations in the input to the scheduling program.
150 minutes, respectively, since the time on the multiplexor channel is the maximum time for each program.

Program 1 uses the CPU 10/100 or .1 of the total time it uses the computer, and program 1 uses the multiplexor channel 100/100 or 1.0 of the total time it uses the computer. Similarly, program 2 uses the CPU .2 of the time, and the multiplexor channel 1.0 of time; and program 3 uses the CPU .3 of the time, and the multiplexor channel 1.0 of the time. These times are shown in Figure 1.

<table>
<thead>
<tr>
<th>Programs</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU/Total time</td>
<td>.1</td>
<td>.2</td>
<td>.3</td>
</tr>
<tr>
<td>Multiplexor Channel/Total time</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Figure 1. Component use as a fraction of total program time

We now know how long each program uses each computer component during one time unit, when the programs are not multiprogrammed together. We have to calculate how long each program uses each computer component during one time unit, when the programs are multiprogrammed together. During one time unit, however, a program uses each computer component in the same proportion that it uses the component over the total program time. Therefore, we have to calculate how long each program uses the computer during each time unit.

Let CPU/MPX(i) represent the fraction of the total program time that program i uses both the CPU and the multiplexor channel. Let MPX(i) represent the fraction of the total program time that program i
uses only the multiplexor channel. Then CPU/MPX(1) = .1 and MPX(1) = .9, since program 1's total program time is 100 minutes, and during 10 minutes of this time it uses both the CPU and the multiplexor channel; and during the remaining 90 minutes it uses only the multiplexor channel. Similarly, CPU/MPX(2) = .2 and MPX(2) = .8; and CPU/MPX(3) = .3 and MPX(3) = .7. These fractions are shown in Figure 2.

<table>
<thead>
<tr>
<th>CPU/MPX(1) = .1</th>
<th>MPX(1) = .9</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU/MPX(2) = .2</td>
<td>MPX(2) = .8</td>
</tr>
<tr>
<td>CPU/MPX(3) = .3</td>
<td>MPX(3) = .7</td>
</tr>
</tbody>
</table>

Figure 2. Component use as a fraction of one time unit

Now suppose that the three programs are multiprogrammed together, and that program 1 has the highest priority, program 2 has the next highest priority, and program 3 has the lowest priority.

During one time unit, program 1 uses the computer for the entire time unit, since it can seize control of the CPU whenever it needs it. Program 2 can use the CPU only when program 1 is not using it, and program 3 can use the CPU only when programs 1 and 2 are not using it.

When program 1 is using both the CPU and the multiplexor channel, program 2 can also use the multiplexor channel, since multiplex mode operations on the multiplexor channel can be overlapped with each other. During any time period, program 2 uses the multiplexor channel alone for a fraction MPX(2) = .8 of the time period. During the time period
CPU/MPX(1)=.1 time unit, program 2 uses the multiplexor channel for 
(.8)*(.1)=.08 time unit.

During the time period MPX(1)=.9 time unit, program 2 can use 
either the multiplexor channel alone, or both the multiplexor channel 
and the CPU. In any case, program 2 uses the computer during the 
entire .9 time unit. The total time that program 2 uses the computer 
during one time unit is .08+.9=.98 time unit, as shown in Figure 3.

Figure 3. Total computer use by program 2 during one time unit

However, during any time period, program 2 uses each computer 
component in the same proportion that it uses the component over the 
total program time. Specifically, CPU/MPX(2)=.2 and MPX(2)=.8. 
But program 2 is active for only .98 time unit, so in one time unit, 
CPU/MPX(2)=(.2)*(.98)=.196 time unit, and MPX(2)=(.8)*(.98)=.784 
time unit. These times are shown in Figure 4. Note that .704+.08= 
.784=MPX(2).

Figure 4. Component use by program 2 during one time unit.
Let CPU/MPX(1,2) represent the fraction of a time unit when both the CPU and the multiplexor channel are being used by either program 1, program 2, or both programs 1 and 2. Let MPX(1,2) represent the fraction of a time unit when only the multiplexor channel is being used by either program 1, program 2, or both programs 1 and 2. Figure 5 shows these fractions as derived from Figure 4. CPU/MPX(1,2) = .296 and MPX(1,2) = .704.

<table>
<thead>
<tr>
<th>CPU/MPX(1)</th>
<th>MPX(1)</th>
<th>MPX(2)</th>
<th>CPU/MPX(2)</th>
<th>MPX(2)</th>
<th>MPX(1,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1</td>
<td>.9</td>
<td>.08</td>
<td>.196</td>
<td>.704</td>
<td>.704</td>
</tr>
</tbody>
</table>

Figure 5. Component use by programs 1 and 2 during one time unit

When programs 1 and 2 are using both the CPU and the multiplexor channel, program 3 can also use the multiplexor channel, since multiplex mode operations can be overlapped with each other. During any time period, program 3 uses the multiplexor channel alone for a fraction MPX(3) = .7 of the time period. During the time period CPU/MPX(1,2) = .296 time unit, program 2 uses the multiplexor channel for (.7) * (.296) = .2072 time unit.

During the time period MPX(1,2) = .704 time unit, program 3 can use either the multiplexor channel alone, or both the multiplexor channel and the CPU. In any case, program 3 uses the computer during the entire .704 time unit. The total time that program 3 uses the computer during one time unit is .2072 + .704 = .9112 time unit, as shown in Figure 6.
CPU/MPX(1,2) = .296 | MPX(1,2) = .704

MPX(3) = .2072 | MPX(3) + CPU/MPX(3) = .704

Figure 6. Total computer use by program 3 during one time unit

However, during any time period, program 3 uses each computer component in the same proportion that it uses the component over the total program time. Specifically, CPU/MPX(3) = .3 and MPX(3) = .7. But program 3 is active for only .9112 time unit, so in one time unit, CPU/MPX(3) = (.3) * (.9112) = .27336 time unit and MPX(3) = (.7) * (.9112) = .63784 time unit. These times are shown in Figure 7. Note that .43064 + .2072 = .63784 = MPX(3).

CPU/MPX(1,2) = .296 | MPX(1,2) = .704

MPX(3) = .2072 | MPX(3) + CPU/MPX(3) = .704

MPX(3) = .2072 | CPU/MPX(3) = .27336 | MPX(3) = .43064

Figure 7. Component use by program 3 during one time unit

To summarize, during 1 time unit, program 1 uses the computer for 1.0 time unit, program 2 uses the computer for .98 time unit, and program 3 uses the computer for .9112 time unit. These fractions are divided into the total program times calculated for the non-multiprogramming environment to determine how long each program takes in the multiprogramming environment. In this example, the total time for
program 1 is 100/1.0=100 minutes; the total time for program 2 is 
50/.98=51 minutes; and the total time for program 3 is 150/.9112=
165 minutes.

**Use with the Flow Shop Scheduling Problem**

The program with the minimum total time (in this case program 
2) is the first program to complete processing. Normally it will be 
replaced by a new program, and the process of calculating multipro-
gramming times will continue. However, if the foreground programs 
are being used in conjunction with the background programs, the same 
foreground programs will always be using the computer, and the only 
programs that will replace one another are the background programs. 
Since the flow shop scheduling problem exists only when at least one 
of the foreground programs is used in conjunction with the background 
programs, we only have to consider that case now. (We still have to 
simulate in the other cases, but only when we have found a place in the 
schedule for the program in question. However, we need to simulate the 
programs in the flow shop scheduling problem before we place them in 
the schedule.)

When the foreground programs are being used in conjunction with 
the background programs, the fractions calculated for each foreground 
program will always be the same, since the programs do not change. The 
only time fractions change is when a new background program is loaded 
into storage. Given a set of programs which constitute a job, with 
one program to be executed in each of the three areas of storage, the 
foreground program times can be calculated by dividing the non-multi-
programmed program times by the relevant fractions, which had previously
been computed. The background program time can be calculated only after a new fraction, unique to this background program, is calculated.

**General Case**

Programs usually use more than two computer components. For example, a foreground transcription program uses the CPU, the multiplexor channel, and a selector channel. The method for calculating multiprogrammed program times is extended in this section to cover the general case where each program uses more than 2 computer components.

Let program 1, program 2, and program 3 be foreground program 1, foreground program 2, and a background program, respectively. Let \( c_1/c_2/.../c_n(i) \) represent the fraction of the total program time that program \( i \) uses computer components \( c_1, c_2, ..., c_n \) at the same time.

Let \( c_1/c_2/.../c_n(1,2) \) represent the fraction of a time unit when all the computer components \( c_1, c_2, ..., c_n \) are being used by either program 1, program 2, or both programs 1 and 2.

Then the total time that program 1 uses the computer during one time unit is 1.0 time unit, since program 1 can seize control of the CPU and the selector channels whenever it needs them. Program 2 can use the CPU and the selector channels only when program 1 is not using them, but it can use some of them while program 1 is using the others. Program 3 can use the CPU and the selector channels only when programs 1 and 2 are not using them, but it can use some of them while programs 1 and 2 are using the others.

The total time that program 2 uses the computer is calculated as follows:
1. Consider each time period \(c_1/c_2/.../c_n\) (1), the fraction of the total time that program 1 uses components \(c_1, c_2, ..., c_n\) at the same time. Calculate the time that program 2 uses the computer during each of these time periods.

2. Sum the times calculated for program 2.

The total time that program 3 uses the computer is calculated as follows:

1. Consider each time period \(c_1/c_2/.../c_n\) (1,2), the fraction of the total time that either program 1, program 2, or both programs 1 and 2 use components \(c_1, c_2, ..., c_n\) at the same time. Calculate the time that program 3 uses the computer during each of the time periods.

2. Sum the times calculated for program 3.

Example

For example, in the non-multiprogramming environment, suppose program 1 uses the CPU for 10 minutes, selector channel 1 for 30 minutes, and the multiplexor channel in the multiplex mode for 100 minutes; program 2 uses the CPU for 10 minutes, selector channel 2 for 30 minutes, and the multiplexor channel in the multiplex mode for 100 minutes; and program 3 uses the CPU for 20 minutes, selector channel 1 for 50 minutes, and selector channel 2 for 100 minutes. Then in the non-multiprogramming environment, the total program times for programs 1, 2, and 3 are all 100 minutes, since the time on the multiplexor channel is the maximum time for programs 1 and 2, and the time on selector channel 2 is the maximum time for program 3.
Program 1 uses the CPU 10/100 or .1 of the total time it uses the computer, program 1 uses selector channel 1 30/100 or .3 of the total time it uses the computer, and program 1 uses the multiplexor channel 100/100 or 1.0 of the total time it uses the computer. Similarly, program 2 uses the CPU .1 of the time, selector channel 2 .3 of the time, and the multiplexor channel 1.0 of the time; and program 3 uses the CPU .2 of the time, selector channel 1 .5 of the time, and selector channel 2 1.0 of the time. These times are shown in Figure 8.

<table>
<thead>
<tr>
<th></th>
<th>Programs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>CPU/Total Time</td>
<td>.1</td>
</tr>
<tr>
<td>Selector Channel 1/Total Time</td>
<td>.3</td>
</tr>
<tr>
<td>Selector Channel 2/Total Time</td>
<td>--</td>
</tr>
<tr>
<td>Multiplexor Channel/Total Time</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Figure 8. Component use as a fraction of total program time

Let CPU, S1, S2, and MPX represent the CPU, selector channel 1, selector channel 2, and the multiplexor channel, respectively. Then, CPU/S1/MPX(1)=.1, S1/MPX(1)=.2, and MPX(1)=.7, since program 1's total program time is 100 minutes, and during 10 minutes of this time it uses the CPU, selector channel 1, and the multiplexor channel together; during 20 minutes of this time it uses both selector channel 1 and the multiplexor channel; and during 70 minutes of this time it uses only the multiplexor channel. Similarly, CPU/S2/MPX(2)=.1, S2/
MPX(2) = .2, and MPX(2) = .7; and CPU/S1/S2(3) = .2, S1/S2(3) = .3, and S2(3) = .5. These fractions are shown in Figure 9.

<table>
<thead>
<tr>
<th>CPU/S1/MPX(1) = .1</th>
<th>S1/MPX(1) = .2</th>
<th>MPX(1) = .7</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU/S2/MPX(2) = .1</td>
<td>S2/MPX(2) = .2</td>
<td>MPX(2) = .7</td>
</tr>
<tr>
<td>CPU/S1/S2(3) = .2</td>
<td>S1/S2(3) = .3</td>
<td>S2(3) = .5</td>
</tr>
</tbody>
</table>

Figure 9. Component use as a fraction of one time unit

When program 1 is using the CPU, selector channel 1, and the multiplexor channel together, program 2 can use either the multiplexor channel alone or both the multiplexor channel and selector channel 2. During any time period, program 2 uses the multiplexor channel alone for a fraction MPX(2) = .7 of the time period. During the time period CPU/S1/MPX(1) = .1 time unit, program 2 uses the multiplexor channel for (.7) * (.1) = .07 time unit.

During any time period, program 2 uses both the multiplexor channel and selector channel 2 for a fraction S2/MPX(2) = .2 of the time period. During the time period CPU/S1/MPX(1) = .1 time unit, program 2 uses both the multiplexor channel and selector channel 2 for (.2) * (.1) = .02 time unit.

During the time periods S1/MPX(1) = .2 time unit and MPX(1) = .7 time unit, program 2 can use either the multiplexor channel alone, both the multiplexor channel and selector channel 2, or the multiplexor channel, selector channel 2, and the CPU together. In any case, program 2 uses
the computer during the entire .2+.7=.9 time unit. The total time that program 2 uses the computer during one time unit is .07+.02+.9=.99 time unit, as shown in Figure 10.

<table>
<thead>
<tr>
<th>CPU/S1/MPX(1)=.1</th>
<th>S1/MPX(1)=.2</th>
<th>MPX(1)=.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPX(2)=.07</td>
<td>S2/MPX(2)=.02</td>
<td>CPU/S2/MPX(2)=.9</td>
</tr>
</tbody>
</table>

Figure 10. Total computer use by program 2 during one time unit

However, during any time period, program 2 uses each computer component in the same proportion that it uses the component over the total program time. Specifically, CPU/S2/MPX(2)=.1, S2/MPX(2)=.2, and MPX(2)=.7. But program 2 is active for only .99 time unit, so in one time unit, CPU/S2/MPX(2)=(.1)*(.99)=.099 time unit, S2/MPX(2)=(.2)*(.99)=.198 time unit, and MPX(2)=(.7)*(.99)=.693 time unit. These times are shown in Figure 11. Note that .02+.04+.138=.198=S2/MPX(2), .07+.138+.485=.693=MPX(2), and .022+.077=.099=CPU/S1/MPX(2).

Now CPU/S1/MPX(1,2)=.08, CPU/S1/S2/MPX(1,2)=.042, S1/S2/MPX(1,2)=.04, S1/MPX(1,2)=.138, CPU/S2/MPX(1,2)=.077, S2/MPX(1,2)=.138, and MPX(1,2)=.485. Figure 12 shows these times as derived from Figure 11.

When programs 1 and 2 are using both the multiplexor channel and selector channel 1, program 3 can use selector channel 2. During
<table>
<thead>
<tr>
<th>CPU/S1/MPX(1) = .1</th>
<th>S1/MPX(1) = .2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPX(2) = .07</td>
<td>S2/MPX(2) = .02</td>
</tr>
<tr>
<td></td>
<td>MPX(2) + S2/MPX(2) + CPU/S2/MPX(2) = .2</td>
</tr>
<tr>
<td></td>
<td>CPU/S2/MPX(2) = .022</td>
</tr>
<tr>
<td></td>
<td>MPX(2) = .138</td>
</tr>
</tbody>
</table>

\[
\begin{aligned}
\text{MPX(1)} &= .7 \\
\text{MPX(2) + S2/MPX(2) + CPU/S2/MPX(2)} &= .7 \\
\text{CPU/S2/MPX(2) = .077} &\quad \text{S2/MPX(2) = .138} &\quad \text{MPX(2) = .485}
\end{aligned}
\]

Figure 11. Component use by program 2 during one unit of time.

<table>
<thead>
<tr>
<th>CPU/S1/MPX(1) = .1</th>
<th>S1/MPX(1) = .2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU/S1/MPX(1,2) = .08</td>
</tr>
<tr>
<td></td>
<td>CPU/S1/S2/MPX(1,2) = .042</td>
</tr>
<tr>
<td></td>
<td>S1/MPX(1,2) = .138</td>
</tr>
</tbody>
</table>

\[
\begin{aligned}
\text{MPX(1)} &= .7 \\
\text{CPU/S2/MPX(2)} &= .077 &\quad \text{S2/MPX(2)} &= .138 &\quad \text{MPX(2)} &= .485 \\
\text{CPU/S2/MPX(1,2)} &= .077 &\quad \text{S2/MPX(1,2)} &= .138 &\quad \text{MPX(1,2)} &= .485
\end{aligned}
\]

Figure 12. Component use by program 1 and 2 during one time unit.
any time period, program 3 uses selector channel 2 alone for a fraction $S_2(3) = .5$ of the time period. During the time period $S_1/MPX(1,2) = .138$ time unit, program 3 uses selector channel 2 for $(.5)*(1.138) = .069$ time unit. Similarly, during the time period $CPU/S1/MPX(1,2) = .08$ time unit, program 3 uses selector channel 2 for $(.5)*(1.08) = .04$ time unit.

During the time period $MPX(1,2) = .485$ time unit, program 3 can use either selector channel 2 alone, both selector channel 2 and selector channel 1, or selector channel 2, selector channel 1, and the CPU together. In any case, program 3 can use the computer during the entire .485 time unit. Program 3 cannot use the computer during any of the remaining time periods that programs 1 and 2 are using the computer, since during those time periods selector channel 2 is in use, and whenever program 3 uses the computer it must use selector channel 2. Therefore, the total time that program 3 uses the computer during one time unit is $0.069 + 0.04 + 0.485 = 0.594$ time unit, as shown in Figure 13.

However, during any time period, program 3 uses each computer component in the same proportion that it uses the component over the total program time. Specifically, $CPU/S1/S2(3) = .2$, $S1/S2(3) = .3$, and $S2(3) = .5$. But program 3 is active for only .594 time unit, so in one time unit, $CPU/S1/S2(3) = (.2)*(0.594) = 0.1188$ time unit, $S1/S2(3) = (.3)*(0.594) = 0.1782$ time unit, and $S2(3) = (.5)*(0.594) = 0.297$ time unit. These times are shown in Figure 14. Note that $0.069 + 0.04 + 0.188 + 0.297 = 0.594 = S2(3)$. 
### Figure 13. Total computer use by program 3 during one time unit.

<table>
<thead>
<tr>
<th>CPU/S1/MPX(1,2) = .08</th>
<th>CPU/S1/S2/MPX(1,2) = .042</th>
<th>S1/S2/MPX(1,2) = .04</th>
<th>S1/MPX(1,2) = .138</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2(3) = .04</td>
<td></td>
<td></td>
<td>S2(3) = .069</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CPU/S2/MPX(1,2) = .077</th>
<th>S2/MPX(1,2) = .138</th>
<th>MPX(1,2) = .485</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2(3) + S1/S2(3) + CPU/S1/S2(3) = .485</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Figure 14. Component use by program 3 during one time unit.

<table>
<thead>
<tr>
<th>CPU/S1/MPX(1,2) = .08</th>
<th>CPU/S1/S2/MPX(1,2) = .042</th>
<th>S1/S2/MPX(1,2) = .04</th>
<th>S1/MPX(1,2) = .138</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2(3) = .04</td>
<td></td>
<td></td>
<td>S2(3) = .069</td>
</tr>
<tr>
<td>S2(3) = .04</td>
<td></td>
<td></td>
<td>S2(3) = .069</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CPU/S2/MPX(1,2) = .077</th>
<th>S2/MPX(1,2) = .138</th>
<th>MPX(1,2) = .485</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2(3) + S1/S2(3) + CPU/S1/S2(3) = .485</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| S2(3) = .188 | S1/S2(3) = .1782 | CPU/S1/S2(3) = .1188 |
Chapter 4

PREVIOUS ALGORITHMS FOR THE FLOW SHOP SCHEDULING PROBLEM

1 Johnson was the first to try to develop optimal sequencing rules for the two- and three-machine scheduling problems. He was able to develop a rule for the two-machine problem, but he was unable to develop one for the three-machine problem, except in special cases.

The three-machine problem remained unsolved, except for enumeration, until two separate studies produced branch and bound algorithms which lead to optimal schedules. The branch and bound technique was first used by Little, Murty, Sweeney, and Karel2 for solving the traveling salesman problem. The branch and bound technique consists of breaking a problem up into smaller and smaller subsets and calculating a lower bound for the best alternative therein. The bounds guide the partitioning of the subsets and eventually identify an optimum. The subsets are represented as nodes of a tree and the process of partitioning as a branching of the tree.3

---


3 Ibid., p. 974.
Ignall and Schrage\textsuperscript{4} and Lomnicki\textsuperscript{5} showed how the branch and bound technique could be used to solve the three-machine flow shop scheduling problem. They developed identical branch and bound techniques. Ignall and Schrage's notation is used here.

The N-Job, 2-Machine Sequencing Problem

Johnson's algorithm for the 2-machine problem is as follows:\textsuperscript{6}

1. Let $a_i$ be the operation time for the $i$th job on the first machine, and $b_i$ the time on the second machine. List the $a$'s and $b$'s in two vertical columns.

2. Scan all the operation times for the shortest one.

3. If it is for the first machine, place the corresponding job first.

4. If it is for the second machine, place the corresponding job last.

5. Remove both operation times for the job from the list.

6. Repeat steps 1-5 until all jobs are sequenced.


\textsuperscript{6}Johnson, pp. 16-17
For example, consider the following:

<table>
<thead>
<tr>
<th>i</th>
<th>a_i</th>
<th>b_i</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>13</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>26</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

Then the optimal sequence is (4, 2, 3, 1), and the total operation time is 51.

The N-Job, 3-Machine Sequencing Problem

Ignall and Schrage's algorithm for the 3-machine problem is as follows:

1. Let J_r be a sequence of size r out of a total of n jobs.

Let \( \text{TIMEA}(J_r) \), \( \text{TIMEB}(J_r) \), and \( \text{TIMEC}(J_r) \) be the times when machines A, B, and C, respectively, complete processing the last of the r jobs. Let \( a_i \), \( b_i \), and \( c_i \) be the processing times of the ith job on machines A, B, and C, respectively.

Let \( J_r \) be the set of n-r jobs that are not in the sequence \( J_r \). Then a lower bound for the sequence \( J_r \) is:

\[
\text{LB}(J_r) = \max \left[ \text{TIMEA}(J_r) + \sum_{J_r} a_i + \min_{J_r} (b_i + c_i) \right]
\]

\[
\text{TIMEB}(J_r) + \sum_{J_r} b_i + \min_{J_r} c_i
\]

\[
\text{TIMEC}(J_r) + \sum_{J_r} c_i
\]

---

Ignall and Schrage, pp. 401-403.
2. Assign the lower bound $\text{LB}(J_r)$ to the node in a tree which corresponds to the sequence $J_r$. Keep the lower bounds in a list.

3. Start the tree with a single node which represents a null sequence.

4. Remove the first node from the list.

5. Create a new node for every job that the just removed node has not yet scheduled.

6. Compute the lower bounds for the new nodes and insert them ranked on the list.

7. If a node that has scheduled all jobs is first on the list, that node's sequence is an optimal one.

8. If an optimal sequence has not been found, go to step 4.

Consider the example shown in Figure 15. The lower bound for node 1 is 55, since that is the maximum of 55 (13+35+7), 31 (3+27+1), and 36 (12+24). Nodes 1 and 2 are at the top of the list initially, since the only other nodes are 3 and 4, which have higher lower bounds. Suppose 1 is ahead of 2 on the list. Then node 1 is removed from the list, and nodes 12, 13, and 14 are created, and their lower bounds are computed.

All of the newly created nodes have higher lower bounds than node 2, so node 2 is the next node removed from the list. Nodes 21, 23, and 24 are created, and their lower bounds are computed. Node 21 goes to the top of the list, since it has the smallest lower bound. Node 21 is then removed from the list and nodes 213 and 214 are created.
An optimal sequence is 2-3-1-4

Figure 15. Tree structure for 4-job, 3-machine problem.
Their lower bounds are greater than that of node 12, which is removed from the list. Nodes 123 and 124 are created and are put into the list behind node 23. Node 23 is removed from the list and nodes 231 and 234 are created. Node 231 has a lower bound than any other node in the list. Since node 231 has scheduled all jobs and is first on the list, that node's sequence is an optimal one.

Use of the Algorithms in Scheduling the Computer

Johnson's algorithm and Ignall and Schrage's algorithm are used to produce optimal daily schedules for the following main storage configurations:

1. Johnson's algorithm is used when one foreground program is used in conjunction with the background programs, and the other foreground program does independent work. If the foreground program is used to read the background program's input, then the foreground program is machine A and the background program is machine B. If the foreground program is used to produce the background program's output, then the background program is machine A and the foreground program is machine B.

2. Ignall and Schrage's algorithm is used when one foreground program is used in conjunction with the background program for input, and the other foreground program is used in conjunction with the background program for output. The input foreground program is machine A, the background program is machine B, and the output foreground program is machine C.
Chapter 5

BRANCH AND BOUND ALGORITHMS FOR THE
N-JOB, 3-MACHINE, 2-PARALLEL-OPERATION
FLOW SHOP SCHEDULING PROBLEM

The algorithms in Chapter 4 can be used to produce optimal daily schedules for many of the storage configurations we are concerned with. However, up to now no algorithm has been developed to produce optimal sequences for the class of three-machine problems in which two of the machines can operate on the same job in parallel. This is a very common problem in the context of scheduling the computer described here. Often both foreground programs are used for background program output, or both are used for background program input.

There are three separate cases for both the input and output problems. The foreground programs may be using different input/output devices (e.g., a card reader and a paper tape reader on input, or a printer and a plotter on output). Both foreground programs may be using identical devices (e.g., two 1000 card per minute card readers on input, or two 1100 line per minute printers on output). The foreground programs may be performing the same operation, but on different input/output device models (e.g., one 1000 card per minute card reader and one 500 card per minute card reader, or one 1100 line per minute printer and one 600 line per minute printer).

Using Ignall and Schrage's notation, the problems can be stated as follows:
Parallel Operations On Input

Let \( J_r \) be a sequence of size \( r \) out of a total of \( n \) jobs. Let \( \text{TIMEA}(J_r), \text{TIMEA}^*(J_r), \) and \( \text{TIMEB}(J_r) \) be the times when machines A, A*, and B, respectively, complete processing the last of the \( r \) jobs. Let \( a_i, a_i^* \) and \( b_i \) be the processing times of the \( i \)th job on machines A, A*, and B, respectively. Let \( J_{r'} \) be the set of \( n-r \) jobs that are not in the sequence \( J_r \). Then a lower bound for the sequence \( J_r \) is:

\[
\text{LB}(J_r) = \max \left[ \text{TIMEA}(J_r) + \sum_{i \in J_r} a_i + \min_{i \in J_r} b_i, \text{TIMEA}^*(J_r) + \sum_{i \in J_r} a_i^* + \min_{i \in J_r} b_i, \text{TIMEB}(J_r) + \sum_{i \in J_r} b_i \right]
\]

Since \( \text{TIMEA}(J_r) + \sum_{i \in J_r} a_i = \sum a_i \) and \( \text{TIMEA}^*(J_r) + \sum_{i \in J_r} a_i^* = \sum a_i^* \), the problem can be expressed in the following form:

\[
\text{LB}(J_r) = \max \left[ \sum_{i \in J_r} a_i + \min_{i \in J_r} b_i, \sum_{i \in J_r} a_i^* + \min_{i \in J_r} b_i, \text{TIMEB}(J_r) + \sum_{i \in J_r} b_i \right]
\]
Parallel Operations on Output

Let $J_r$ be a sequence of size $r$ out of a total of $n$ jobs. Let $T_{MEA}(J_r)$, $T_{MEB}(J_r)$, and $T_{MEB^*}(J_r)$ be the times when machines A, B, and $B^*$, respectively, complete processing the last of the $r$ jobs.

Let $a_i$, $b_i$, and $b^*_i$ be the processing times of the $i$th job on machines A, B, and $B^*$, respectively. Let $J_{n-r}$ be the set of $n-r$ jobs that are not in the sequence $J_r$. Then a lower bound for the sequence $J_r$ is:

$$\text{LB}(J_r) = \max_{r} \left[ T_{MEA}(J_r) + \sum_{i} a_i + \min_{i} \left[ \begin{array}{c} \min_{r} b_i, b_i > b^*_i \\ \min_{r} b^*_i, b_i > b^*_i \end{array} \right] \right]$$

As in the previous problem, $T_{MEA}(J_r) + \sum_{i} a_i = \sum_{i} a_i$

Case 1: Parallel Operations on Input, Different Operations

Sum the $a_i$'s and the $a^*_i$'s and choose the larger sum. Choose lower bounds as described above. Follow the algorithm in Chapter 4 for assigning lower bounds to nodes, discarding nodes, and creating new nodes. When a node which has scheduled all jobs is at the top of the list, an optimal solution has been found. Figure 16 illustrates the use of this algorithm.
An optimal sequence is 2 - 1 - 3 - 4

Figure 16. Tree structure for 4-job, 3-machine problem, where 2 different operations are performed on different machines in parallel before the third operation.
Case 2: Parallel Operations on Output, Different Operations

Sum the a_i's. From the operation times for the remaining jobs, add to the sum the minimum of the following: the minimum b_i, where b_i > b*_i, or the minimum b*_i, where b*_i > b_i. Choose lower bounds as described above. Follow the algorithm in Chapter 4 for finding an optimal solution. Figure 17 illustrates the use of this algorithm.

Case 3: Parallel Operations on Input, Identical Machines

In this case we have two parallel operations for each job, where each operation can be put on either machine, and the machines have identical processing characteristics (i.e., each operation will take the same amount of time no matter which machine processes which operation.)

Before we can include the b_i operations in the sequencing problem, we have to solve the parallel machine sequencing problem. Let a_i and a'_i represent the processing times for the two parallel operations for the ith job. We want to assign the a_i's and the a'_i's to machines A and A*, such that we minimize total processing time for the operations. Once we have a minimum, we know which operations for each job should appear on which machines. The sequencing of the jobs will not affect this minimum, since a machine can begin processing one job as soon as it finishes processing the previous job. (This is so because for any job, no operation precedes operations a_i and a'_i.) A lower bound for the sequence J is:
An optimal sequence is 2-1-4-3

Figure 17. Tree structure for 4-job 3-machine problem where 2 different operations are performed on different machines in parallel after the third operation.
Figure 18 illustrates the use of this algorithm. Choose any job to start. In Figure 18, job 1 was chosen as the first job.

1. Compute the lower bounds as described above. Keep a separate tree for each machine.

2. Extend the trees by adding both possible operations to both trees. Compute the lower bounds for the new nodes.

3. Group the new nodes in pairs, each pair representing a feasible schedule (e.g., 12 for machine A, and 1'2' for machine A*; 12' for machine A, and 1'2 for machine A*).

4. Compute the lower bound for each pair of nodes as described above.

5. Keep the minimum of the two lower bounds, and compare it with the previous lower bound.

6. If the previous lower bound is smaller, its nodes define an optimal sequencing for the two machines.

7. If the new lower bound is equal to or less than the previous lower bound, than the new lower bound is used to continue
An optimal sequence is 1-2'-3'-4' & 1'-2-3-4.

Figure 18. Tree structure for 4-job, 2-machine problem, where 2 identical operations are performed on identical machines in parallel.
branching with step 2, unless the new lower bound has scheduled n-1 jobs, in which case it defines an optimal sequencing for the two machines.

Now the \( b_i \) operations can be included in the sequencing problem. Compute a lower bound as described under "Parallel Operations on Input." Follow the algorithm in Chapter 4 for finding an optimal solution. Figure 19 illustrates the use of this algorithm.

**Case 4: Parallel Operations on Output, Identical Machines**

As in case 3, we have two parallel operations for each job, where each operation can be put on either machine, and the machines have identical processing characteristics.

We want to use the lower bound formulas listed under "Parallel Operations on Output." To do so, compute \( \sum_{J} b_i \) and \( \sum_{J} b^*_i \) in the way that \( a_i \) and \( a^*_i \) sums were computed for case 3. After computing the lower bounds using the algorithm in case 3 and formula under "Parallel Operations on Output," we can follow the algorithm in Chapter 4 for finding an optimal solution. Figure 20 illustrates case 4.

**Case 5: Parallel Operations on Input, Same Operation, Different Machines**

In this case we have two parallel operations for each job, where each operation can be put on either machine, but the machines have different operating characteristics (i.e., each operation takes longer to complete on one machine than it does on the other).
An optimal sequence is 2-1-3-4.

Figure 19. Tree structure for 4-job, 3-machine problem, where 2 identical operations are performed on identical machines in parallel before the third operation.
An optimal sequence is 2-1'-4-3 & 2'-1-4'-3'

Figure 20. Tree structure for 4-job, 3-machine problem, where 2 identical operations are performed on identical machines in parallel after the third operation.
The technique used is similar to that for case 3, except that two pairs of trees are maintained instead of one pair. As in case 3, choose any job to start. As shown in Figure 21, assign one operation to machine A in one tree pair, and to machine A* in the other tree pair. For each tree pair follow the algorithm in case 3 to obtain an optimal sequence for that tree pair. Compare the operation times for the two sequences, and choose the smaller as the optimal sequence.

Now include the $b_i$ operations in the sequencing problem, as shown in Figure 22. Compute a lower bound as described under "Parallel Operations on Input." Follow the algorithm in Chapter 4 for finding an optimal sequence.

**Case 6: Parallel Operations on Output, Same Operation Different Machines**

As in case 3, we have two parallel operations for each job, where each operation can be put on either machine, but the machines have different operating characteristics.

We want to use the lower bound formulas listed under "Parallel Operations on Output." To do so, compute $\sum_{j \in J_{r}} b_i$ and $\sum_{j \in J_{r}} b^{*}$ in the way the $a_i$ and $a^{*}$ sums were computed for case 5. After the lower bounds are computed in this manner, we can follow the algorithm in Chapter 4 for finding an optimal sequence, as shown in Figure 23.
Figure 21. Tree structure for 4-job, 2-machine problem, where 2 identical operations are performed on different machines in parallel.

An optimal sequence is 1'-2-3-4' on machine a & 1-2'-3'-4 on machine a*

Jobset

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>13</td>
<td>7</td>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td>a*</td>
<td>15</td>
<td>9</td>
<td>28</td>
<td>4</td>
</tr>
<tr>
<td>a'</td>
<td>3</td>
<td>12</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>a'*</td>
<td>5</td>
<td>14</td>
<td>11</td>
<td>8</td>
</tr>
</tbody>
</table>

b

12 16 7 1
An optimal sequence is 2-1-3-4

Figure 22. Tree structure for 4-job, 3-machine problem, where 2 identical operations are performed on different machines in parallel before the third operation.
Figure 23. Tree structure for 4-job, 3-machine problem, where 2 identical operations are performed on different machines in parallel after the third operation.

An optimal sequence is $2' - 4' - 1 - 3$ on machine $b$ & $2 - 4 - 1' - 3'$ on machine $b^*$. 

Jobset

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>13 7 26 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>3 12 9 6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b*</td>
<td>5 14 11 8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b'</td>
<td>12 16 7 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b'*</td>
<td>14 18 9 3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
We can now combine the tools developed in the preceding three chapters to schedule the programs run on the computer. The schedule period is a week. Most jobs enter the computer center at the beginning of one day and must be completed by the end of that day. Other jobs have different turnaround times up to a maximum of one week, but all jobs must be completed during the week they enter the computer center. The foreground areas are assigned a normal use for the week (e.g., one area for background program input, the other for background program output), but there may be other foreground programs to schedule. Some background programs form a program set which must be executed in sequence with no other programs intervening. Some background programs need devices which one or both of the foreground programs use. Such background programs cannot be multiprogrammed with other programs, if it results in a conflict of this sort.

1. Remove from the list of programs to be scheduled those background programs which cause conflicts, those foreground programs which are not assigned to a foreground area as the normal program for that area, and those programs whose entrance and completion times are not the beginning and end of the same day.

2. Combine background program sets into a single conceptual program by summing the processing times and foreground program times across programs.
3. For the remaining programs (the normal foreground programs; background programs which do not cause conflicts and whose entrance and completion times are the beginning and end of the same day) compute optimal daily schedules as follows:
   a. Compute the individual program times using the simulation method in Chapter 3.
   b. Sequence the programs in each day using the appropriate algorithm in Chapter 4 or 5.

4. For each day's schedule, starting with the first day, and proceeding day by day until the end of the week, do the following:
   a. Adjust the part of the schedule which includes the background program sets. This is required, since we can now overlap one program's processing with other programs' input or output transcription operations.
   b. Insert the daily programs which have been removed from the list because of device conflicts, foreground area usage, or entrance and completion times.
   c. Insert as many programs as possible that have longer than one day turnaround times, starting with those with the nearest due date.

This scheduling rule is in the form of a computer program. Appendix A describes the program's input and output formats. Appendix B contains the program flow charts and FORTRAN coding. Appendix C contains examples of program input and output.
Chapter 7
CONCLUSIONS AND SUGGESTIONS FOR FUTURE STUDY

A scheduling program has been developed which provides good or "near" optimal schedules for a set of programs run on a multiprogrammed computer. The program provides more efficient schedules for the computer than could be provided by a computer operator without using the program.

The scheduling rule developed here consists of a simulation method which is used to calculate the running times for programs in the multiprogramming environment, and sequencing algorithms which are used to schedule the programs after the running times have been calculated. New branch and bound algorithms have been developed for the n-job, 3-machine sequencing problem, where 2 of the machines can operate on the same job in parallel.

Future research in computer scheduling should be directed towards more complex environments than the one studied here, and towards providing accurate data for use with the scheduling program.

The environment studied here is a static one. That is, all the programs to be scheduled during a period are known before the schedule is made. Within the context of the IBM System/360 Disk Operating System, research should be directed towards developing a dynamic scheduling rule. Such a scheduling rule would consist of a computer program which is always in the computer's main storage, ready to be executed whenever a new program enters the computer center. This scheduling program would
relieve the operator of the burden of producing a new schedule whenever the scheduling environment changed. Instead, the scheduling program would keep track of the then current schedule, dispatch programs when an area of main storage is free, and modify the schedule when there are changes in the environment.

The ability of the scheduling program to produce good schedules is dependent upon the accuracy of the input data which describes program characteristics. Before more complex scheduling rules are produced, research should be directed towards developing accurate data on program characteristics.
Appendix A

PROGRAM USER MANUAL

This appendix contains the information needed to use the scheduling program. The scheduling program is an efficient means for producing a weekly schedule for programs run on the IBM System/360 under control of the IBM System/360 Disk Operating System (DOS/360). DOS/360 provides a multiprogramming environment where main storage is divided into three areas (a background area and two foreground areas). Input to the scheduling program consists of statements describing the computer, the installation (including normal foreground area assignments), and the programs to be scheduled. Output from the program consists of a suggested schedule, showing program sequence and clock time for the scheduled week.

The programs to be scheduled are divided into three categories: weekly programs, partial working day programs, and full working day programs. Each program category may consist of background programs, foreground programs, or both background and foreground programs. Weekly programs are ready for processing on one day of the week, but do not have to be completed until some day later in the week. Partial working day programs must be completed the same day they are ready for processing, but in a shorter time span than full working day programs. Full working day programs are ready for processing at the start of a day and must be completed by the end of the same day.

The scheduling program checks all input statements for proper syntax. If there are any errors in a statement, the program prints
an error message, prints the statement in error, and does not produce a schedule. The statements in error can then be corrected and the scheduling program can be rerun.

**Input Formats**

The input to the scheduling program consists of a dictionary statement, hardware statements, installation statements, and job statements.

**Dictionary Statement:** The dictionary statement is used to define the syntactic characters used by the scheduling program. It must be the first statement in the input stream. The format of this statement is:

<table>
<thead>
<tr>
<th>Columns</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-24</td>
<td>ABCDFHIJOPRSTUXZ01234567</td>
</tr>
<tr>
<td>25-36</td>
<td>Must be blank</td>
</tr>
<tr>
<td>37-80</td>
<td>Should be blank</td>
</tr>
</tbody>
</table>

**Hardware Statements:** The hardware statements describe the physical characteristics of the computer. They consist of an HC statement, one or more HD statements, and an HZ statement.

The HC statement describes the physical characteristics of the central processing unit. This statement must immediately follow the dictionary statement. If the HC statement is in error, the run is terminated immediately, since the error may have been caused by the absence of the dictionary statement. The format of the HC statement is:
The HD statements describe the physical characteristics of the input/output devices. These statements must immediately follow the HC statement. There must be at least one HD statement. The maximum number of HD statements is determined when the scheduling program is compiled. Initially, the maximum number is ten. The format of these statements is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>H</td>
</tr>
<tr>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>3-6</td>
<td>Central processing unit model number (e.g., 2030)</td>
</tr>
<tr>
<td>7-9</td>
<td>Average instruction time in tenths of a microsecond</td>
</tr>
<tr>
<td>10-12</td>
<td>Interference time per byte on selector channels in hundredths of a microsecond</td>
</tr>
<tr>
<td>13-15</td>
<td>Interference time per byte on multiplexor channel in tenths of a microsecond. This time may be overridden for individual devices by use of the HD statement.</td>
</tr>
<tr>
<td>16-80</td>
<td>Should be blank</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>H</td>
</tr>
<tr>
<td>2</td>
<td>D</td>
</tr>
<tr>
<td>3-6</td>
<td>Device type (e.g., 2311)</td>
</tr>
<tr>
<td>7-8</td>
<td>Model number; used to distinguish devices with the same device type, but with different physical characteristics. The field must be used to distinguish (1) the punch and read units of a card read punch, (2) 7-track and 9-track tape drives, and (3) devices with the same device type that have different speeds.</td>
</tr>
<tr>
<td>9</td>
<td>Device class--D for disk, T for tape, U for unit record</td>
</tr>
<tr>
<td>10-16 (D or T in 9)</td>
<td>Device speed in bytes per second</td>
</tr>
<tr>
<td>10-12 (U in 9)</td>
<td>Record size in bytes</td>
</tr>
</tbody>
</table>
13-16 (U in 9)  Device speed in records per minute. This speed may be overridden for individual files by use of JBIO and JFIO statements.

17-18  Device set-up time in tenths of a minute.

19-21 (D in 9)  Average latency time for device in milliseconds. This time may be overridden for individual files by use of the IF, JBIO, and JFIO statements.

19-21 (U in 9)  Start-stop time for device in tenths of a millisecond.

19-21 (T in 9)  Should be blank

22-24 (D in 9)  Average seek time for device in milliseconds. This time may be overridden for individual files by use of the IF, JBIO, and JFIO statements.

22-24 (T in 9)  Rewind time for device in tenths of a minute.

22-24 (U in 9)  Should be blank

25-27 (T in 9)  Device density in bytes per inch.

25-27 (D or U in 9)  Should be blank

28-30  Interference time per byte on multiplexor channel in tenths of a microsecond, if desired to override time on HC statement. Otherwise should be blank.

31-80  Should be blank

The HZ statement indicates the end of the hardware statements. It must immediately follow the last HD statement. The format of this statement is:

<table>
<thead>
<tr>
<th>Column</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>H</td>
</tr>
<tr>
<td>2</td>
<td>Z</td>
</tr>
</tbody>
</table>

Installation Statements: The installation statements describe the input/output and main storage configurations, and the work week. They consist of one or more ID statements, two IF statements, an IS statement, and an IZ statement.
The ID statements relate input/output unit numbers to device descriptions. These statements must immediately follow the JZ statement. There must be at least one ID statement. The maximum number of unit numbers is determined when the scheduling program is compiled. Initially the maximum number is 20. The format of these statements is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I</td>
</tr>
<tr>
<td>2</td>
<td>D</td>
</tr>
<tr>
<td>3-6</td>
<td>Device type</td>
</tr>
<tr>
<td>7-8</td>
<td>Model number. The device type and model number must be the same as a device type and model number that appears on a valid HD statement.</td>
</tr>
<tr>
<td>9-10</td>
<td>Number of unit numbers in this statement, an integer from 1 to 23</td>
</tr>
<tr>
<td>11-79</td>
<td>Unit numbers in hexadecimal, each unit number taking up three columns</td>
</tr>
<tr>
<td>80</td>
<td>Any non-blank character, if there are more ID statements, otherwise, blank.</td>
</tr>
</tbody>
</table>

The IF statements describe the normal usage of the foreground areas. These statements must immediately follow the last ID statement. There must be two IF statements, one for each foreground area. The format of these statements is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I</td>
</tr>
<tr>
<td>2</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>1 for the first statement (foreground area one) and 2 for the second statement (foreground area two).</td>
</tr>
<tr>
<td>4-5</td>
<td>R for card reader transcription program, P for printer transcription program, TI for telecommunications input transcription program, TO for telecommunications output transcription program, or blank if no program assigned to this area.</td>
</tr>
<tr>
<td>6-13 (4-5 non-blank)</td>
<td>Program name</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>14-16 (4-5 non-blank)</td>
<td>Unit number for peripheral operation. The device class of this unit number must be U.</td>
</tr>
<tr>
<td>17-19 (4-5 non-blank)</td>
<td>Unit number for intermediate storage. The device class of this unit number must be D or T.</td>
</tr>
<tr>
<td>20-22 (4-5 non-blank)</td>
<td>Number of instructions executed per record.</td>
</tr>
<tr>
<td>23-24 (4-5 non-blank)</td>
<td>Blocking factor on intermediate storage device.</td>
</tr>
<tr>
<td>25-27 (4-5 non-blank)</td>
<td>Average latency time for intermediate storage device (device class D), if desired to override time on HD statement. Otherwise, blank.</td>
</tr>
<tr>
<td>28-30 (4-5 non-blank)</td>
<td>Average seek time for intermediate storage device (device class D), if desired to override time on HD statement. Otherwise, blank.</td>
</tr>
<tr>
<td>6-30 (4-5 blank)</td>
<td>Should be blank</td>
</tr>
<tr>
<td>31-80</td>
<td>Should be blank</td>
</tr>
</tbody>
</table>

The IS statement describes the work week in terms of the time the installation is active during each day. This statement must immediately follow the second IF statement. The format of this statement is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>Number of days in week, from 1 to 7</td>
</tr>
<tr>
<td>4-59</td>
<td>Begin time and end time for each day in the form HHMMHHMM, where HH is an hour from 00 to 23 and MM is a minute from 00 to 59.</td>
</tr>
<tr>
<td>60-80</td>
<td>Should be blank</td>
</tr>
</tbody>
</table>

The IZ statement indicates the end of the installation statements. It must immediately follow the IS statement. The format of this statement is:
Job Statements: The job statements describe the programs to be scheduled. Each background program description consists of a JB statement, one or more JBD statements, two JBF statements, and one or more JBIO statements. Each foreground program description consists of a JF statement, one or more JFD statements and one or more JFIO statements. A JZ statement follows the last program description.

The descriptions for weekly programs must precede the descriptions for daily programs. The daily program descriptions must be ordered by day. Within each day, the partial working day program descriptions must precede the full working day program descriptions. Within each program category (weekly, partial working day for each day, and full working day for each day), background program descriptions must precede foreground program descriptions. If a background job consists of two or more job steps, the program descriptions for the steps must appear in sequence with no other program descriptions intervening, and the job name and begin- and end-times for each step must be the same.

The maximum number of programs in each category is determined when the scheduling program is compiled. Initially, the maximum number of weekly programs is 10, the maximum number of partial working day programs is 5 for each day, and the maximum number of full working day programs is 5 for each day.
The JB and JF statements contain the names of the programs to be scheduled and describe the scheduling constraints for each program. The first of these statements must immediately follow the IZ statement. Additional statements of this type must immediately follow the last JBIO and JFIO statement for the previous program. The format of these statements is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>J</td>
</tr>
<tr>
<td>2</td>
<td>Blank for weekly programs or an integer from 1 to 7 corresponding to a day on the IS statement for daily programs.</td>
</tr>
<tr>
<td>3</td>
<td>B for background programs or F for foreground programs</td>
</tr>
<tr>
<td>4-5</td>
<td>Should be blank</td>
</tr>
<tr>
<td>6-13 (B in 3)</td>
<td>Job name</td>
</tr>
<tr>
<td>6-13 (F in 3)</td>
<td>Program name</td>
</tr>
<tr>
<td>14-21 (B in 3)</td>
<td>Step name</td>
</tr>
<tr>
<td>14-21 (F in 3)</td>
<td>Should be blank</td>
</tr>
<tr>
<td>22-26</td>
<td>Time when program is ready for processing, in the form DHHMM, where D is a day from 1 to 7, HH is an hour from 00 to 23, and MM is a minute from 00 to 59.</td>
</tr>
<tr>
<td>27-31</td>
<td>Time by which program must be completed, in the form DHHMM, where D is a day from 1 to 7, HH is an hour from 00 to 23, and MM is a minute from 00 to 59. For weekly programs, column 22 does not equal column 27. For daily programs those columns are the same. For full working day programs, columns 23-26 and 28-31 must be the same as the begin-time and end-time for day D. For partial working day programs at least one of those times must be different than the corresponding times on the IS statement.</td>
</tr>
<tr>
<td>32-36</td>
<td>Fixed number of instructions executed by program regardless of the volumes of its files.</td>
</tr>
</tbody>
</table>
The JBD and JFD statements relate file names to unit numbers. These statements must immediately follow the JB and JF statements. There must be at least one of these statements for each program description. The maximum number of files for each program is determined when the scheduling program is compiled. Initially this number is 20. The format of these statements is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>J</td>
</tr>
<tr>
<td>2</td>
<td>Same as first statement for this program description</td>
</tr>
<tr>
<td>3</td>
<td>Same as first statement for this program description</td>
</tr>
<tr>
<td>4</td>
<td>D</td>
</tr>
<tr>
<td>5</td>
<td>Should be blank</td>
</tr>
<tr>
<td>6</td>
<td>Number of file names on this statement, an integer from 1 to 7</td>
</tr>
<tr>
<td>7-76</td>
<td>One or more relations between a file name and a unit, each consisting of a 7 character file name and a 3 hexadecimal digit unit number. The unit number must be the same as one that appears on a valid ID statement.</td>
</tr>
<tr>
<td>77-79</td>
<td>Should be blank</td>
</tr>
<tr>
<td>80</td>
<td>Any non-blank character, if there are more of these statements for this program description. Otherwise, blank.</td>
</tr>
</tbody>
</table>
The JBF statements define the unit record files which can be processed by the foreground transcription programs described by the IF statements. These statements must immediately follow the last JBD statement for each background program description. There must be two JBF statements for each background program description. The format of the statements is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>J</td>
</tr>
<tr>
<td>2</td>
<td>Same as first statement for this program description</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
</tr>
<tr>
<td>4</td>
<td>F</td>
</tr>
<tr>
<td>5</td>
<td>Should be blank</td>
</tr>
<tr>
<td>6-7</td>
<td>R if file can be processed by card reader transcription program, P</td>
</tr>
<tr>
<td></td>
<td>if file can be processed by printer transcription program, T if file</td>
</tr>
<tr>
<td></td>
<td>can be processed by telecommunications input transcription program, TO</td>
</tr>
<tr>
<td></td>
<td>if file can be processed by telecommunications output transcription program, or blank if there is no file name on this statement.</td>
</tr>
<tr>
<td>8-14 (6-7 non-blank)</td>
<td>File name. Must be the same as a file name that appears on a valid JBD statement for this program description. The device class of the unit number for this file must be U.</td>
</tr>
<tr>
<td>8-14 (6-7 blank)</td>
<td>Should be blank</td>
</tr>
<tr>
<td>15-80</td>
<td>Should be blank</td>
</tr>
</tbody>
</table>

The JBIO and JFIO statements describe the characteristics of each file processed by the program. The JBIO statements must immediately follow the second JBF statement for each program description. The JFIO statements must immediately follow the last JFD statement for each program. There must be one JBIO statement for each file name that appears on a valid JBD statement. Similarly, there must be one JFIO
statement for each file name that appears on a valid JFD statement.

The format of these statements is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>J</td>
</tr>
<tr>
<td>2</td>
<td>Same as first statement for this program description</td>
</tr>
<tr>
<td>3</td>
<td>Same as first statement for this program description</td>
</tr>
<tr>
<td>4</td>
<td>I</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>6-12</td>
<td>File name</td>
</tr>
<tr>
<td>13-20</td>
<td>Number of records in file</td>
</tr>
<tr>
<td>21-25</td>
<td>Record size in bytes. If the device class of the unit assigned to this file is U, then this field is ignored and the device's record size is used.</td>
</tr>
<tr>
<td>26-27</td>
<td>Blocking factor. If the device class of the unit assigned to this file is U, then this field is ignored, and the blocking factor is set equal to 1.</td>
</tr>
<tr>
<td>28</td>
<td>X if the number of instructions executed by this program is a function of this file's size. Otherwise, blank. If X, then the number of records in this file is multiplied by the number in columns 37-41 of the JB or JF statement, and the product is added to the fixed number of instructions.</td>
</tr>
<tr>
<td>29</td>
<td>A character to indicate valid device classes for the unit assigned to this file. A if device class can be D, T, U; D if device class can be D; H if device class can be D or T; T if device class can be T; or U if device class can be U.</td>
</tr>
<tr>
<td>30-32</td>
<td>Average latency time for device (device class D), if desired to override time on HD statement. Otherwise, blank.</td>
</tr>
<tr>
<td>33-35</td>
<td>Average seek time for device (device class D), if desired to override time on HD statement. Otherwise, blank.</td>
</tr>
</tbody>
</table>
36-39  Factor in thousandths (device class U), if desired to modify device speed on HD statement. Otherwise, blank. If a number appears here, the device speed is multiplied by it.

40-79  Should be blank

80     Any non-blank character, if more statements of this type. Otherwise, blank.

The JZ statement indicates the end of the job statements. It must immediately follow the last statement for the last program description. The format of this statement is:

<table>
<thead>
<tr>
<th>Column</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>J</td>
</tr>
<tr>
<td>2</td>
<td>Z</td>
</tr>
</tbody>
</table>

Output Formats

Output from the scheduling program consists of a summary of program restrictions, a summary of the input statements, error messages, and the schedules for each main storage area for each day.

Program Restriction Summary: The first page of output contains the program maximums for number of device types, unit numbers, files, full working day programs per day, partial working day programs per day, and weekly programs.

Input Summary: Summaries are printed for the central processing unit, each device type and model number, both foreground area assignments, the weekly schedule, and each program.

The central processing unit summary consists of a line for each of the fields on the HC statement.
Each device and model number summary consists of a line for each field on the HD statement followed by the unit numbers related to the device type and model number via the ID statements.

The foreground area summaries consist of a line for each field on the IF statements.

The weekly schedule summary contains the begin and end times for each day in the week.

Each program summary consists of three parts. The first part contains a line for each field on the JB or JF statement. The second part contains a line for each file assigned to a unit number as shown on the JBD or JFD statement, except for those background program files that can be processed by the foreground transcription programs. If the file cannot be processed by the foreground program because of device conflicts, the unit number printed is the one that appeared on the JBD statement. If the file can be processed by only one foreground program, the unit number printed is the intermediate storage device used by the foreground program. If the file can be processed by both foreground programs, no unit number is printed here. Instead the unit number assigned is printed just before the schedule is printed. The third part of the program summary contains a header line for each file, followed by a line for each field on the JBIO or JFIO statements.

Error Messages: Error messages consist of a heading, followed by a message, (usually) followed by the input statement in error, as shown in the following example. Some errors do not refer to a specific
statement and, therefore, no input statement follows the message.

*******ERROR*******
UNIT NUMBER 28I NOT DEFINED BY AN ID STATEMENT
J1BD 1 MASTER 28I

Schedules: A separate schedule is printed for each main storage area for each day. These schedules are followed by a list of programs that could not be included in the schedule due to time constraints.

The format of the background area schedule is:

<table>
<thead>
<tr>
<th>Columns</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-2</td>
<td>Hour on</td>
</tr>
<tr>
<td>3</td>
<td>Blank</td>
</tr>
<tr>
<td>4-5</td>
<td>Minute on</td>
</tr>
<tr>
<td>6</td>
<td>Blank</td>
</tr>
<tr>
<td>7-8</td>
<td>Hour off</td>
</tr>
<tr>
<td>9</td>
<td>Blank</td>
</tr>
<tr>
<td>10-11</td>
<td>Minute off</td>
</tr>
<tr>
<td>12</td>
<td>Blank</td>
</tr>
<tr>
<td>13-20</td>
<td>Job name</td>
</tr>
<tr>
<td>21</td>
<td>Blank</td>
</tr>
<tr>
<td>22-29</td>
<td>Step name</td>
</tr>
<tr>
<td>30</td>
<td>Blank</td>
</tr>
<tr>
<td>31-32</td>
<td>Set-up time in minutes</td>
</tr>
<tr>
<td>33</td>
<td>Blank</td>
</tr>
<tr>
<td>.34</td>
<td>Rewind time in minutes</td>
</tr>
<tr>
<td>35</td>
<td>Blank</td>
</tr>
<tr>
<td>36-39</td>
<td>Central processing unit work time, in the form HHMM, where HH is hours and MM is minutes</td>
</tr>
<tr>
<td>40</td>
<td>Blank</td>
</tr>
<tr>
<td>41-44</td>
<td>Central processing unit wait time, in the form HHMM, where HH is hours MM is minutes</td>
</tr>
<tr>
<td>45</td>
<td>Blank</td>
</tr>
<tr>
<td>46-48</td>
<td>Unit address in hexadecimal</td>
</tr>
<tr>
<td>49</td>
<td>Blank</td>
</tr>
<tr>
<td>50-53</td>
<td>Unit work time, in the form HHMM, where HH is hours and MM is minutes</td>
</tr>
<tr>
<td>54</td>
<td>Blank</td>
</tr>
<tr>
<td>55-58</td>
<td>Unit wait time, in the form HHMM, where HH is hours and MM is minutes</td>
</tr>
<tr>
<td>59</td>
<td>Blank</td>
</tr>
<tr>
<td>60-128</td>
<td>Additional unit addresses, work times, and wait times. If additional lines are needed, they are printed starting with Column 46.</td>
</tr>
</tbody>
</table>
The format of each foreground area schedule is:

<table>
<thead>
<tr>
<th>Column(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-2</td>
<td>Hour on</td>
</tr>
<tr>
<td>3</td>
<td>Blank</td>
</tr>
<tr>
<td>4-5</td>
<td>Minute on</td>
</tr>
<tr>
<td>6</td>
<td>Blank</td>
</tr>
<tr>
<td>7-8</td>
<td>Hour off</td>
</tr>
<tr>
<td>9</td>
<td>Blank</td>
</tr>
<tr>
<td>10-11</td>
<td>Minute off</td>
</tr>
<tr>
<td>12</td>
<td>Blank</td>
</tr>
<tr>
<td>13-20</td>
<td>Program name</td>
</tr>
<tr>
<td>21</td>
<td>Blank</td>
</tr>
<tr>
<td>22-29</td>
<td>Job name, if program is a transcription program. Otherwise, blank.</td>
</tr>
<tr>
<td>30</td>
<td>Blank</td>
</tr>
<tr>
<td>31-38</td>
<td>Step name, if program is a transcription program. Otherwise, blank.</td>
</tr>
<tr>
<td>39</td>
<td>Blank</td>
</tr>
<tr>
<td>40-41</td>
<td>Set-up time in minutes</td>
</tr>
<tr>
<td>42</td>
<td>Blank</td>
</tr>
<tr>
<td>43</td>
<td>Rewind time in minutes</td>
</tr>
<tr>
<td>44</td>
<td>Blank</td>
</tr>
<tr>
<td>45-48</td>
<td>Central processing unit work time, in the form HHMM, where HH is hours and MM is minutes</td>
</tr>
<tr>
<td>49</td>
<td>Blank</td>
</tr>
<tr>
<td>50-53</td>
<td>Central processing unit wait time, in the form HHMM, where HH is hours and MM is minutes</td>
</tr>
<tr>
<td>54</td>
<td>Blank</td>
</tr>
<tr>
<td>55-57</td>
<td>Unit address in hexadecimal</td>
</tr>
<tr>
<td>58</td>
<td>Blank</td>
</tr>
<tr>
<td>59-62</td>
<td>Unit work time, in the form HHMM, where HH is hours and MM is minutes</td>
</tr>
<tr>
<td>63</td>
<td>Blank</td>
</tr>
<tr>
<td>64-67</td>
<td>Unit wait time, in the form HHMM, where HH is hours and MM is minutes</td>
</tr>
<tr>
<td>68</td>
<td>Blank</td>
</tr>
<tr>
<td>69-123</td>
<td>Additional unit addresses, work times, and wait times. If additional lines are needed, they are printed starting with Column 55.</td>
</tr>
</tbody>
</table>
Appendix B

PROGRAM LOGIC MANUAL

This appendix is a guide to the logic of the scheduling program. It is intended for persons who want to modify the program or diagnose program malfunctions. Included are descriptions of the program's major components and their storage requirements, the program flow charts, and the FORTRAN source statements.

Program Organization

The scheduling program consists of a main program and eighteen subprograms. Figure 24 shows how these components are related. In Figure 24, vertical lines connect calling programs with the programs they call. For example, MAIN (the main program) calls subprogram IGNSCH which calls subprograms PARAL1, PARAL2, and ERRPRT.

The following text briefly describes the function of the main program and each subprogram and lists the subprograms that each calls.

Main Program: Reads and checks syntax of all input statements except the JBD, JFD, JBIO, and JFIO statements. Writes all program output except those error messages written by subprograms. Adjusts schedules for inclusion of programs which are not scheduled by use of the algorithms.

Subprograms Called: UNITX, DISKX, TAPEX, FRCTN, TIME23, FILE, FILEIO, TOD, SWITCH, JOHNSN, IGNSCH, ERRPRT.
Subprogram UNITX: Computes device, channel, CPU, and set up times for unit record devices.

Subprogram Called: CHNCUU

Subprogram DISKX: Computes device, channel, CPU, and set up times for disk devices.

Subprogram Called: CHNCUU

Subprogram TAPEX: Computes device, channel, CPU, set up and rewind times for tape devices.

Subprogram Called: CHNCUU

Subprogram CHNCUU: Determines channel and interference rate for a device.

Subprograms Called: None

Subprogram FRCTN: Separates resource uses into multiplexor mode, burst mode, and CPU portions.

Subprogram Called: FRCTBL

Subprogram FRCTBL: Completes computation of resource uses and then sorts resource uses.

Subprogram Called: None

Subprogram TIME2: Calculates multiprogramming time for second priority program.

Subprograms Called: None
Subprogram TIME3: Calculates multiprogramming time for third priority program.

Subprograms Called: None

Subprogram FILE: Reads and checks syntax of JBD and JFD statements.

Subprogram Called: ERRPRT

Subprogram FILEIO: Reads and checks syntax of JBIO and JFIO statements.

Subprogram Called: ERRPRT

Subprogram TOD: Checks validity of begin and end times on JB and JF statements.

Subprogram Called: ERRPRT

Subprogram SWITCH: Moves program description from one storage area to another storage area.

Subprogram Called: None

Subprogram JOHNSN: Solves two-machine two-operation sequencing problem.

Subprograms Called: None

Subprogram IGNSCH: Solves three-machine three-operation sequencing problem.

Subprograms Called: PARAL1, PARAL2, ERRPRT

Subprogram PARAL1: Solves parallel operation sequencing problem for same operations and same machines.

Subprograms Called: None
Subprogram **PARAL2**: Solves parallel operation sequencing problem for same operations and different machines.

Subprograms Called: None

Subprogram **TIME23**: Calculates multiprogramming times for second priority and third priority programs.

Subprograms Called: TIME2, TIME3

Subprogram **ERRPRT**: Prints error heading and increments error count

Subprograms Called: None

**Storage Requirements**

The main storage required for the main program and for several subprograms depends upon the values assigned to certain variables in the main program. The variables are:

- **MDEV**, the maximum number of device types and model numbers in the installation.
- **MCUU**, the maximum number of unit numbers in the installation
- **MFILE**, the maximum number of file names per program description
- **MJOB**, the total number of weekly program descriptions (MWJOB), partial working day program descriptions per day (MDJOB - MWJOB), and full working day program descriptions per day (MJOB - MDJOB).

Values are assigned to these variables when the scheduling program is compiled. The following text lists the names of those arrays whose dimensions should be changed when the values assigned to the
above variables are changed. In addition, the main storage for sub-
program IGNSCH depends upon the value assigned to the variable NLIST,
which is defined in subprogram IGNSCH.

Main Program:

<table>
<thead>
<tr>
<th>MDEV</th>
<th>MCUU</th>
<th>MFILE</th>
<th>MJOB</th>
<th>MJOB- MDJOB</th>
<th>MJOB &amp; MCUU</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEVTP</td>
<td>CUUPNT</td>
<td>FILPNT</td>
<td>DON</td>
<td>TSEL6A</td>
<td>QL</td>
</tr>
<tr>
<td>DEVMOD</td>
<td>CUU</td>
<td>FILNAM</td>
<td>HON</td>
<td>TSEL6W</td>
<td>QN</td>
</tr>
<tr>
<td>DEVCLS</td>
<td>DOFF</td>
<td>TMPXMA</td>
<td>QA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVSPD</td>
<td>HOFF</td>
<td>TMPXMW</td>
<td>QB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVROD</td>
<td>MON</td>
<td>TMPXBA</td>
<td>QC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVSTP</td>
<td>MOFF</td>
<td>TMPXBW</td>
<td>QDX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVLSL</td>
<td>TOTAL</td>
<td>TSETUP</td>
<td>QDP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVSRW</td>
<td>TCPU</td>
<td>TWNDUP</td>
<td>ODUX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVDEN</td>
<td>TNOVL</td>
<td>JOBNM</td>
<td>QD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNITX</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISKX</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TAPEX</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHNCUU</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRCTN</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRCTBL</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME2</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME3</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FILE:

<table>
<thead>
<tr>
<th>MDEV</th>
<th>MCUU</th>
<th>MFILE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEVCLS</td>
<td>CUUPNT</td>
<td>FILPNT</td>
</tr>
<tr>
<td>CUU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FILNAM</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>

FILEIO:

<table>
<thead>
<tr>
<th>MDEV</th>
<th>MCUU</th>
<th>MFILE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEVCLS</td>
<td>CUUPNT</td>
<td>FILPNT</td>
</tr>
<tr>
<td>DEVLSS</td>
<td>FILNAM</td>
<td></td>
</tr>
<tr>
<td>DEVSRW</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TOD: None

SWITCH: None

JOHNSN:

MJOB - MDJOB

<table>
<thead>
<tr>
<th>A</th>
<th>M</th>
<th>B</th>
<th>PA</th>
<th>C</th>
<th>PB</th>
<th>D</th>
<th>PD</th>
<th>DX</th>
<th>PDX</th>
<th>DP</th>
<th>PDP</th>
<th>DPX</th>
<th>PDPX</th>
<th>L</th>
<th>PL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

IGNSCH: The main storage for this subprogram depends upon the value assigned to NLIST, which is the number of nodes created. If N is the number of full working day jobs, then NLIST must be between \((.5 \times N \times (N-1)) + 1\) and \(N!\) The maximum number of full working day jobs that can be scheduled is 9, except for case 4 and case 6, where the maximum is 8.
PARAL1:

MDJOB-MDJOB
A
B
AX
BX
L
LMAX
LMIN

PARAL2:

MDJOB-MDJOB
A
B
C
D
AX
BX
L
L1
LMAX
LMIN

TIME23: None

ERRPT: None

Flowcharts

The following charts can be used to locate the FORTRAN source statements that perform specific program functions. Figure 25 identifies the flowchart symbols. To locate the statements that perform a specific function use the flowcharts to find the flowchart symbol or symbols that describe the function. The FORTRAN statement number of the first source statement used to perform a function appears above the upper left hand corner of the symbol that defines the function.

Flowcharts for the main program have two-digit identifiers. The highest level charts have zero as their first digit. Flowcharts for the subprograms have two-letter identifiers. If there is more than
one chart for a subprogram, all the charts have the same first letter. This letter corresponds to the number which appears in columns 74-75 of the source statements for the subprogram.
Predefined process. No further detail.

Processing

For further detail see chart xx.

Input/Output

For further detail see Chart xx.

Subprogram ABCD

See Chart xx.

Subprogram ABCD

See Chart xx.

Figure 25. Flowchart symbols.
Main Program

BEGIN

PRINT HEADING AND RESTRICTIONS

READ DICTIONARY STATEMENT

10

PROCESS HARDWARE STATEMENTS

20

PROCESS INSTALLATION STATEMENTS

PRINT CPU SUMMARY

1

PRINT FOREGROUND SUMMARY

30

PROCESS JB & JF STATEMENTS

PRINT DEVICE SUMMARY

40

PROCESS JBF STATEMENTS

PRINT PROGRAM SUMMARY

PRINT FILE ASSIGNMENTS

FILE IA

FILE PROCESS JBZ0 & JFD STATEMENTS

50

FILE PROCESS JBF STATEMENTS

2

FOREGROUND OR BACKGROUND

FG

BG

01

02

03
BEGIN

ANY ERRORS

Yes → END

No → ANY FULL-DAY JOBS

No → END

Yes → 3-MACHINE CASE

3-MACHINE CASE

GET NON-CONFLICT BG PM TIMES

IGNSCH NA

FIND OPTIMAL SEQUENCE

SPACE OUT PROGRAMS

PRINT SCHEDULES

END

GET BG PM TIMES

JOHNSON MA

FIND OPTIMAL SEQUENCE
Subprogram UNITX

CALL

CHNCUU 8A
DETERMINE CHANNEL AND INTERFERENCE FOR UNIT

INCREMENT TOTAL SET UP TIME BY THIS UNIT'S SET UP TIME

DOES DEVICE OPERATE AT ITS RATED SPEED? NO

1

CALCULATE DEVICE WORK TIME BASED ON RATED SPEED

2

INCREMENT CHANNEL ACTIVE TIME BY DEVICE WORK TIME

3

INCREMENT CPU OVERLAPPED TIME

INCREASE CPU OVERLAPPED TIME BY DEVICE INTERFERENCE

IS THE NUMBER OF INSTRUCTIONS PROPORTIONAL TO FILES VOLUME?

YES

INCREMENT CPU NON-OVERLAPPED TIME

NO

INCREMENT CPU OVERLAPPED TIME BASED ON RATED SPEED MODIFIED BY FACTOR

CHNCUU 8A
SAVE NEW CHANNEL ACTIVE TIME

RETURN
CALL

CHNCLU DA
DETERMINE CHANNEL AND INTERFERENCE FOR UNIT

INCREMENT TOTAL SETUP TIME BY THE UNIT'S SETUP TIME

CALCULATE DEVICE WORK TIME

INCREMENT CHANNEL ACTIVE TIME BY DEVICE WORK TIME

INCREMENT CPU OVERLAPPED TIME BY DEVICE INTERFERENCE

1

IS THE NUMBER OF EXECUTIONS PROPORTIONAL TO FILE LENGTH?

YES

INCREMENT CPU OVERLAPPED TIME

CALCULATE DEVICE REWIND TIME

IS DEVICE REWIND TIME EQUAL TO DEVICE REWIND TIME?

NO

INCREMENT CPU NON-OVERLAPPED TIME

YES

CHNCLU DA
SAVE NEW CHANNEL ACTIVE TIME

RETURN
CALL

WHICH USE OF THIS SUBROUTINE IS THIS

DETERMINE CHANNEL FROM SAVED CHANNEL NUMBER

GET OLD CHANNEL TIME

SAVE CHANNEL NUMBER

GET INTERFERENCE RATE

RETURN

DETERMINE
CHANNEL
TIME

SAVE NEW
CHANNEL
TIME

RETURN
Subprogram TIME2

Charts GB

1

GB 1

600

REMOVE A RESOURCE FROM PROGRAM 2

IS THE NUMBER OF RESOURCES ZERO?

NO

GA 1

YES

GB 2

700

REMOVE A RESOURCE FROM PROGRAM 1

IS THE NUMBER OF RESOURCES ZERO?

NO

GA 2

YES

MARK END OF TABLE

GB 3

INCREMENT, PROGRAM 2's TIME, SAVE NUMBER OF RESOURCES

SAVE POINTER TO PROGRAM 1 OVERLAPPED CPU TIME

INCREMENT PROGRAM 2's TIMES BY .7X

INCREMENT PROGRAM 2's TIMES BY .7X NON-OVERLAPPED CPU TIME

SAVE POINTER TO PROGRAM 2 OVERLAPPED CPU TIME

INCREMENT, PROGRAM 2's TIME x .7X NON-OVERLAPPED CPU TIME

INCREMENT PROGRAM 2's TIME x .7X

INCREMENT PROGRAM 1's WAIT TIME

RETURN
Subprogram TIME3

Chart HA

CALL

HA 2

IS THERE
A NO CONNECT
FROM 1-PROGRAM
COMB.?

YES

HC 3

FIND THE
NUMBER OF
RESOURCES USED
BY THE THIRD
PROGRAM

2

FIND THE
NUMBER OF
RESOURCES USED
BY ONE OF THE
OTHER PROGRAMS

IS IT
PROGRAM
1 OR 2

PGM 1

PGM 2

HB 1

HB 2

300

CONFLICT
WITH PROGRAM
1

YES

HB 2

NO

2.10

IS THIS
PROGRAM 3
RESOURCE
PARTIAL
OK?

YES

HB 2

NO

220

IS THIS
PROGRAM 3
RESOURCES
BEST.
MODE
OK?

YES

HB 2

NO

230

BURST MODE
OK, OVERLAP
WITI JOINT.
MODE?

YES

HB 2

NO

2.40

IS THIS
PROGRAM 3
RESOURCE
CPU

YES

HB 2

NO

500

CONFLICT
WITH PROGRAM
1

YES

HB 2

NO

2.50

CONFLICT
WITH PROGRAM
1

YES

HB 2

NO

ANYMORE
PROGRAM 3
RESOURCES

YES

HB 2

NO

SAVE NUMBER
OF RESOURCES
FOR PROGRAM
3

1

2
Subprogram TOD

Chart KA

CALL

SET PROGRAM TYPE TO FULL WORKING DAY

IS THIS A WEEKLY OR A DAILY PROGRAM?

DAILY

IS DAY ON A DAY OF SAME AS DAY IN COLUMN 2?

ARE DAY ON A DAY OF SAME AS DAY IN COLUMN 2?

YES

KB 2

NO

KB 2

ARE DAY ON A DAY IN ALLOWABLE RANGE?

YES

KB 2

NO

KB 2

MAX0 IF DAILY PROGRAM, SET PROGRAM TYPE TO PARTIAL WORKING DAY

ON>OFF

ON=BEGIN

ON<END

TIME ON

KB 2

TIME OFF

KB 2

DAY END

KB 2

DAY BEGIN

KB 2

KB 2

KB 2

KB 2

KB 2

KB 1

KB 1

KB 1
Subprogram TOD

Chart KB

KB 1

TIME OFF END
DAY END TIME
OFF>END

TIME OFF BEGIN
DAY BEGIN TIME
OFF>BEGIN

MAX0
IF DAILY PROGRAM
SET PROGRAM TYPE TO PARTIAL WORKING DAY

RETURN

ERRPET RA
PRINT ERROR MESSAGE & INCREMENT COUNT

PRINT ERROR MESSAGE

SET ERROR SWITCH ON

RETURN
Subprogram SWITCH

Chart LA

CALL

MOVE JOBNAME AND STEPPING TO NEW PLACE IN TABLE

MOVE BEGIN AND END TIMES TO NEW PLACE IN TABLE

RETURN
Subprogram PARAL2

Chart PA

CALL

CREATE POSITIONS TO MAXIMUM MINIMUM OPERATION FOR EACH JOB

WHICH TREE FIRST SECOND

CREATE A NODE FOR EACH OF TWO TREES WITH FIRST OPERATION ON MACHINE A

CREATE A NODE FOR EACH OF TWO TREES WITH SECOND OPERATION ON MACHINE B

WHICH MACHINE HAS LARGER SMALLER OPERATIONS

MACH. A MACH. B

ASSIGN SMALLER OPERATIONS OF REMAINING JOBS TO "A"

ASSIGN SMALLER OPERATIONS OF REMAINING JOBS TO "B"

AMAX1 Compute CURRENT LOWER Bound

EXTEND EACH TREE BY ADDING BOTH POSSIBLE OPERATIONS TO IT

AMAX1 Compute NEW LOWER Bounds

WHICH NEW LOWER BOUND IS SMALLER

FIRST SECOND

10 13

NEW LB. CURRENT LB.

NEW LB. CURRENT LB.

N>C N>C

PB 1 14

PB 1

SET CURRENT LB. EQUAL TO NEW LB.

SET CURRENT LB. EQUAL TO NEW LB.

16

ANY SEQUENCES NOT TESTED

PB 1

YES

NO

1
Subprogram PARAL2  
Chart PB

PB 1

17

WHICH TREE PAIR 25 FIRST

SECOND

19

WHICH A, B 25 SMALLER

FIRST

SECOND

20

GET FIRST TREE PAIR'S OPTIMAL SEQUENCE

22

SAVE OPTIMAL SEQUENCE

RETURN

SAVE OPTIMAL SEQUENCE

PA 1
Subprogram TIME23

CALL

SET PROGRAMS TIME TO ZERO

To await program 1

TIME2 GA
CALCULATE TIME FOR BURST PORTION, BOTH PROGRAMS

TIME2 GA
CALCULATE TIME FOR CPU PORTION: PGM 1 & PGM 2

TIME2 GA
CALCULATE TIME FOR BURST PORTION: PGM 1 & PGM 2

TIME2 GA
CALCULATE TIME FOR CPU PORTION: PGM 1 & MPX PORTION PGM 2

TIME2 GA
CALCULATE TIME FOR CPU PORTION: PGM 2 & MPX PORTION PGM 1
Subprogram ERRPT

```
CALL

ADD ONE TO ERROR COUNT

PRINT ERROR HEADING

RETURN
```
FORTRAN Source Statements

The following pages contain the source statements for the scheduling program. The program is written in the IBM System/360 Basic FORTRAN IV Language and has been tested on a System/360 Model 165 using the Operating System/360 (Release 11) FORTRAN G compiler. The input and output data set reference numbers are 5 and 6, respectively.
DISCLAIMER OF QUALITY

Due to the condition of the original material, there are unavoidable flaws in this reproduction. We have made every effort possible to provide you with the best copy available. If you are dissatisfied with this product and find it unusable, please contact Document Services as soon as possible.

Thank you.

The following pages contain text that are illegible.
FORTRAN IV LEVEL 0, MOD 0

MAIN

DATE = 67256  21/30/55  PAGE 1202

0044       TSELW[I],J1,K1]=0,
0045       TSELW[I],J1,K1]=0.
0046       TSELW[I],J1,K1]=0.
0047       TSELW[I],J1,K1]=0.
0048       TSELW[I],J1,K1]=0.
0049       TSELW[I],J1,K1]=0.
0050       TSELW[I],J1,K1]=0.
0051       TSELW[I],J1,K1]=0.
0052       TSELW[I],J1,K1]=0.
0053       TOTAL[J1,K1]=0.
0054       DD 0998 I1=1.
0055       DD 0998 J1=1, MJO
0056       DD 0998 K1=1, MCI
0057       TOTAL[I]=1, K1=1.
0058       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0059       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0060       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0061       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0062       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0063       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0064       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0065       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0066       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0067       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0068       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0069       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0070       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)
0071       DIMENSION FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2), FCMF(12,2)

C

C PRINT HEADING AND PROGRAM RESTRICTIONS

C

WRITE (6,4004)

4004 FORMAT(1H4,46X,46H SYSTEM/360 DOS/TOS SCHEDULING PROGRAM)

WRITE (6,4005)

4005 FORMAT(1H4,56X,20H PROGRAM RESTRICTIONS)

WRITE (6,4006) 4, 0EV

4006 FORMAT(1H3X,75H MAXIMUM NUMBER OF DEVICE TYPES (DEFINED BY HD STMC000114)

I ATMENTS) IN INSTALLATION = 1, 13)

WRITE (6,4007) 4, MCI

4007 FORMAT(1H3X,75H MAXIMUM NUMBER OF UNIT NUMBERS (DEFINED BY TD STMC000111)

I ATMENTS) IN INSTALLATION = 1, 13)

WRITE (6,4008) 4, MCI

4008 FORMAT(1H3X,75H MAXIMUM NUMBER OF FILES (DEFINED BY JSR-JEN AND

I JBO-JEIO STATEMENTS) PER PROGRAM = 2, 13)

WRITE (6,4009) J1

4009 FORMAT(1H3X,16H MAXIMUM NUMBER OF FULL WORKING DAY PROGRAMS PER

I DAY = 1, 13)

WRITE (6,4009) J1
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0112</td>
<td>10</td>
<td>DEVSPD(1) = 110000*DEVSPD(1)+DEVSPD(1)</td>
<td>MOD 71600</td>
</tr>
<tr>
<td>0113</td>
<td>11</td>
<td>IF DEVINT(1) = 011010, 100, 110</td>
<td>MOD 71650</td>
</tr>
<tr>
<td>0114</td>
<td>100</td>
<td>DEVINT(1) = AVMPX</td>
<td>MOD 71700</td>
</tr>
<tr>
<td>0115</td>
<td>110</td>
<td>I0EV=2</td>
<td>MOD 71750</td>
</tr>
<tr>
<td>0116</td>
<td>127</td>
<td>DO 21, 11 = I0EV*DEV</td>
<td>MOD 71800</td>
</tr>
<tr>
<td>0117</td>
<td>READ</td>
<td>5, 15, A1, A2, DEVTYPE(1), DEVMODD(1), DEVLS(1), DEVAP(1), DEVEN(1), DEVINT(1)</td>
<td>MOD 71900</td>
</tr>
<tr>
<td>0118</td>
<td>IF(A1-H) = 1012, 13, 1012</td>
<td>MOD 71950</td>
<td></td>
</tr>
<tr>
<td>0119</td>
<td>13</td>
<td>IF(A2 = 0114, 15, 14)</td>
<td>MOD 72000</td>
</tr>
<tr>
<td>0120</td>
<td>14</td>
<td>IF(A2 = 711014, 27, 1014)</td>
<td>MOD 72050</td>
</tr>
<tr>
<td>0121</td>
<td>15</td>
<td>IF (DEVLS(1) = 0114, 15, 16)</td>
<td>MOD 72100</td>
</tr>
<tr>
<td>0122</td>
<td>16</td>
<td>IF (DEVLS(1) = 17, 18, 17)</td>
<td>MOD 72150</td>
</tr>
<tr>
<td>0123</td>
<td>17</td>
<td>IF (DEVLS(1) = 0117, 19, 19, 17)</td>
<td>MOD 72200</td>
</tr>
<tr>
<td>0124</td>
<td>18</td>
<td>DEVSPD(1) = (10000*DEVSPD(1)+DEVSPD(1))</td>
<td>MOD 72250</td>
</tr>
<tr>
<td>0125</td>
<td>19</td>
<td>IF (DEVINT(1) = 011019, 189, 190)</td>
<td>MOD 72300</td>
</tr>
<tr>
<td>0126</td>
<td>190</td>
<td>DEVINT(1) = AVMPX</td>
<td>MOD 72350</td>
</tr>
<tr>
<td>0127</td>
<td>19C</td>
<td>K = 11 - 1</td>
<td>MOD 72400</td>
</tr>
<tr>
<td>0128</td>
<td>00</td>
<td>DO 21, J = 1 - K</td>
<td>MOD 72450</td>
</tr>
<tr>
<td>0129</td>
<td>IF DEVTYPE(11) = DEVTYPE(11)</td>
<td>MOD 72500</td>
<td></td>
</tr>
<tr>
<td>0130</td>
<td>20</td>
<td>IF DEVMODD(11) = DEVMODD(11)</td>
<td>MOD 72550</td>
</tr>
<tr>
<td>0131</td>
<td>21</td>
<td>CONTINUE</td>
<td>MOD 72600</td>
</tr>
<tr>
<td>0132</td>
<td>11</td>
<td>I = I + 1</td>
<td>MOD 72650</td>
</tr>
<tr>
<td>0133</td>
<td>21O</td>
<td>READ (21, 21) A1, A2</td>
<td>MOD 72700</td>
</tr>
</tbody>
</table>
0134   711 FORMAT(24I1)
0135   IF(A1-N1)121,127,121)
0136   212 IF(A2-2)121,22,121)
C
C SAVE NUMBER OF DEVICES
C
0137   22 MOEY=11-1
0138   11=1
0139   J1=M10(23,MGU)
C
C READ AN ID STATEMENT
C
0140   23 READ (5,24) A1,A2,A3,A4,N1,(PRCTUHK),K=1,23,A5
0141   24 FORMAT(24I1,A4,A2),A1
0142   IF(A1-111024),240,1024
0143   240 IF(A2-111240),241,1240
C
C CHECK FOR VALID NUMBER OF UNIT DEFINITIONS IN STATEMENT
C
0144   241 IF((MCU1=1)-N1=111241,25,25
0145   25 IF(23-N111025,26,26
0146   26 IF(N1-111026,260,260)
0147   260 GO TO 261 K=11,J1
0148   261 G(U)K=PRCTUHK-K+1
C
C RELATE UNIT TO DEVICE TYPE AND MODEL NUMBER
C
0149   265 DO 29 L=1,MOEY
0150   IF(A3-DEVTPY(L))129,27,28
0151   27 IF(A4-DFVMOD(L))128,280,280
0152   28 CONTINUE
0153   290 GO TO 1024
0154   290 M=11+N1-1
C
C CHECK FOR DUPLICATE UNIT NUMBER
C
0155   29  GO TO 1024 K=11,M
0156   300 GO TO (302,29);T1
0157   300 NN=1,N
0158   300 IF(ABS(CUH)-CUH(NN))300,1029,300
0159   300 CONTINUE
C
C CHECK FOR VALID CHANNEL NUMBER
C
0160   307 IF(A1(CUH)-A1)1300,1305,307
0162   301 IF(A1(07)-A1(CUH))1300,1306,306
C
C SAVE PRINTER TO DEVICE TYPE AND MODEL NUMBER

C CHECK FOR ADDITIONAL IF STATEMENTS

0164  IF(A5.BLANK4)31,32,31

0165  31  I=I+1  N1

0166  IF(I1-1000,1000,1000)

0167  310 J=MIN(J+22,1000)

0168  GO TO 23

C SAVE NUMBER OF UNITS

0169  27  MGUIK

0170  12=1

0171  370  00  93  IFG=12+2

C READ AN IF STATEMENT

0172  READ (5,33) A1,A2,M1,A3,A4,PROCNI(IEGI),EGGUP(IEGI),FGOUI(IEGI),

       IFNSTY(IEGI),FBLKFC(IEGI),FLNCRG(IEGI),FSFRK(IEGI)

0173  33  FORMAT(2A1,11,2A1,A8,2A3,F3,0,F3,0,F3,0,F3,0)

0174  IF(A1-1)1033,14,1033

0175  34  IF(A2=0)1034,39,1034

0176  35  IF(N1=1)1035,16,1035

C CHECK FOR VALID "FOREGROUND" PROGRAM TYPE

0177  36  IF(A3.BLANK4)38,37,38

0178  37  IF(A4.BLANK4)1037,63,1037

0179  38  IF(A3=0)141,39,51

0180  39  IF(A4.BLANK4)1037,60,1037

C FOREGROUND PROGRAM IS A PRINTER TRANSCRIPTION PROGRAM

0181  40  IFG(IEGI)=P

0182  GO TO 46

0183  41  IF(A3=0)44,42,44

0184  42  IF(A4.BLANK4)1037,43,1037

C FOREGROUND PROGRAM IS A READER TRANSCRIPTION PROGRAM

0185  43  IFG(IEGI)=R

0186  GO TO 49

0187  44  IF(A3=0)11044,45,1044

0188  45  IF(A4=1)144,47,44

0189  46  IF(A4=0)11046,48,1046
FOREGROUND PROGRAM IS A TELECOMMUNICATIONS INPUT TRANSCRIPTION PROGRAM

0190  47 FG(FG)=1
0191  GO TO 49

FOREGROUND PROGRAM IS A TELECOMMUNICATIONS OUTPUT TRANSCRIPTION PROGRAM

0192  48 FG(FG)=0

C CHECK FOR EXISTENCE OF PROGRAM NAME

C

0193  49 IF(PROGIM(FG)=BLANK)=490,1049,490

C CHECK FOR VALID UNIT NUMBER

C

0194  490 DO 51 I=1,MCU
0195    IF(FGUNIT(FG)=CUNIT(I))=51,50,51
0196    50 JI=CHPNT(I)
0197    FGUNIT(FG)=JI
0198    FGUNIT(FG)=JI

C VERIFY THAT PERIPHERAL DEVICE IS A UNIT RECORD DEVICE

C

0199  IF(FDEVCLS(JI))=1050,52,1050
0200  51 CONTINUE
0201  GO TO 1051

C CHECK FOR VALID UNIT NUMBER

C

0202  52 DO 55 I=1,MCU
0203    IF(FGUNIT(FG)=CUNIT(I))=55,53,55
0204    53 JI=CHPNT(I)
0205    FGCENT(FG)=JI
0206    FGUNIT(FG)=JI

C VERIFY THAT INTERMEDIATE STORAGE DEVICE IS A DISK OR TAPE DEVICE

C

0207  IF(FDEVCLS(JI))=0154,56,54
0208  54 IF(FDEVCLS(JI))=11054,73,1054
0209  55 CONTINUE
0210  GO TO 1055
0211  56 IF(FLTYNCY(FG))=0,11056,57,58

C SAVE LATENCY TIME FOR DISK DEVICE

C

0212  57 FLTYNCY(FG)=0<FVLS(I)
0213  58 IF(FSFK(FG))=0,11058,59,73

C
FOURTAN IV C LEVEL 0, MOD 0

C SAVE SEEK TIME FOR DISK DEVICE

C
0214 59 SEFSK(TFG1)=DEVSTXW (ST)
0215  GO TO 73
C
C THERE IS NO PROGRAM ASSIGNED TO THIS FOREGROUND AREA

C
0216 63 FG(TFG1)=BLANK4
0217 71 CONTINUE

C CHECK FOR DEVICE CONFLICT

C
0218 IF(FG(11)=BLANK4) 64, 680, 684
0219 64 IF(FG(12)=BLANK4) 64, 680, 684
0220 65 IF(FGCUUUP(11)=FGCUUP(12)) 64, 680, 67
0221 66 A0=FGCUUP(11)
0222  GO TO 1066
0223 67 IF(FGCUUUP(11)=FGCUU(2)) 64, 63, 680
0224 68 A0=FGCUU(11)
0225 71 J1=FGCUP(11)
0226 IF(DEVCLS(J1)=T1680, 1064, 680
C COMPUTE RESOURCE USE FRACTIONS FOR EACH FOREGROUND ASSIGNMENT

C
0227 680, 680, 11=1, 2
C IS THERE A PROGRAM ASSIGNED TO THIS FOREGROUND AREA

C
0228 IF(FG(11)=BLANK4) 681, 689, 684
C YES, CALCULATE DEVICE, CHANNEL, CPU, AND SET UP TIMES FOR UNIT REQUESTED

C
0229 J1=FGCUP(11)
0230 CALL UNITX (TDEV(1,1), TDMXMA(1,1,1), TSEL(1,1), TSEL24(1,1,1),
0231 TSEL(1,1,1), TSEL4(1,1,1), TSEL5(1,1,1),
0232 TSEL6A(1,1,1), TSEL7(1,1,1), TSEL8P(1,1), TDEVJ(1,1))
0233 3DEVSP(J1,J1), X1, FINSY(1,1), AVINST, AVSEL, DEVT(1,1)
0234 IF(GCUUP(11)=01, 07, 03, 04, 05, 06, X)
0235 K1=FGCUP(11)
C IS THE INTERMEDIATE STORAGE DEVICE DISK OR TAPE

C
0236 IF(DEVCLS(K1)=T1680, 683, 692
C TAPE, CALCULATE DEVICE, CHANNEL, CPU, SET UP, AND "REMINO TIMES"

C
0237 682 CALL TAPEX (TDEV(1,1,1)), TDMXMA(1,1,1), TSEL(1,1,1), TSEL24(1,1,1),
0238 TSEL(1,1,1), TSEL4(1,1,1), TSEL5(1,1,1), TSEL6A(1,1,1), TSEL7(1,1,1),
0239 TSEL8P(1,1,1), TDEVJ(1,1,1)
0240 CALL TAPEX (TDEV(1,1,1), TDMXMA(1,1,1), TSEL(1,1,1), TSEL24(1,1,1),
0241 TSEL(1,1,1), TSEL4(1,1,1), TSEL5(1,1,1), TSEL6A(1,1,1), TSEL7(1,1,1),
0242 TSEL8P(1,1,1), TDEVJ(1,1,1)
0243 CALL TAPEX (TDEV(1,1,1), TDMXMA(1,1,1), TSEL(1,1,1), TSEL24(1,1,1),
0244 TSEL(1,1,1), TSEL4(1,1,1), TSEL5(1,1,1), TSEL6A(1,1,1), TSEL7(1,1,1),
0245 TSEL8P(1,1,1), TDEVJ(1,1,1)
C
C DISK, CALCULATE DEVICE, CHANNEL, CPU, AND SETUP TIMES
C
0235 CALL DISKX (TOEV[1,1],[1]), TMOXA[1,1], TSFLAT[1,1], TSFLAT[1,1], MOD 32975
1 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], TSCNI[1,1], MOD 32975
2 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], TSCNI[1,1], MOD 32975
3 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], TSCNI[1,1], MOD 32975
4 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], TSCNI[1,1], MOD 32975
5 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], TSCNI[1,1], MOD 32975
C WHICH FOREGROUND PROGRAMS THIS
C
0236 GO TO (685, 686, 11) 
C FIRST, COMPUTE RESOURCE USE FRACTIONS
C
0237 CALL FACN (TOTAL[1,1], TMOXI[1,1], TSFLAT[1,1], TSFLAT[1,1], MOD 14975
1 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], MOD 14975
2 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], MOD 14975
3 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], MOD 14975
4 TMOXML[1,1], TMOXS[1,1], TMOX[1,1], CPU, MOD 14975
5 TSFL[1,1], MOD 14975
0238 GO TO 687
C SECOND, COMPUTE RESOURCE USE FRACTIONS
C
0237 CALL FACN (TOTAL[1,1], TMOXI[1,1], TSFLAT[1,1], TSFLAT[1,1], MOD 14975
1 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], MOD 14975
2 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], MOD 14975
3 TSFLA[1,1], TSFL2[1,1], TSFL3[1,1], TSFL3[1,1], MOD 14975
4 TMOXML[1,1], TMOXS[1,1], TMOX[1,1], CPU, MOD 14975
5 TSFL[1,1], MOD 14975
0240 TOTAL[1,1]=0.
0241 TOEVI[1,1]=0.
0242 TOEVI[1,2]=0.
0243 TMOXI[1,1]=0.
0244 TSFLAT[1,1]=0.
0245 TSFLAT[1,1]=0.
0246 TSFLAT[1,1]=0.
0247 TSFL[1,1]=0.
0248 TSFL[1,1]=0.
<table>
<thead>
<tr>
<th>FORTRAN IV G LEVEL 0, MOD 0</th>
<th>MAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE = 6/15/65</td>
<td>9/1/1955</td>
</tr>
</tbody>
</table>

```
0318     75 11=1
0319     760 IF(ITIMES(111-231)776,770,176)  
0320     770 IF(ITIMES(111)-011770,77,77  
0321     77 IF(ITIMES(111)-231780,780,177  
0322     780 IF(ITIMES(111)-1780,78,78  
0323     78 IF(ITIMES(111)-501790,790,178  
0324     790 IF(ITIMES(111)-1790,79,79  
0325     79 IF(ITIMES(111)-691800,800,179  
0326     800 IF(ITIMES(111)-01800,80,80  
0327     80 IF(ITIMES(111)-231810,810,182  
0328     81 IF(ITIMES(111)-501820,820,183  
0329     82 IF(ITIMES(111)-1830,83,83  
0330     83 IF(ITIMES(111)-691840,840,185  
0331     840 IF(ITIMES(111)-01840,85,85  
0332     850 IF(ITIMES(111)-231850,850,186  
0333     860 IF(ITIMES(111)-501860,860,187  
0334     870 IF(ITIMES(111)-1870,87,87  
0335     880 IF(ITIMES(111)-691880,880,189  
0336     890 IF(ITIMES(111)-01890,89,89  
0337     89 IF(ITIMES(111)-231890,890,190  
0338     90 IF(ITIMES(111)-501900,900,191  
0339     91 IF(ITIMES(111)-1910,91,91  
0340     92 IF(ITIMES(111)-691920,920,193  
0341     93 IF(ITIMES(111)-01930,93,93  
0342     93 IF(ITIMES(111)-231930,930,194  
0343     94 IF(ITIMES(111)-501940,940,195  
0344     95 IF(ITIMES(111)-1950,95,95  
0345     95 IF(ITIMES(111)-691960,960,196  
0346     96 IF(ITIMES(111)-01960,96,96  
0347     96 IF(ITIMES(111)-231960,960,197  
0348     97 IF(ITIMES(111)-501970,970,198  
0349     98 IF(ITIMES(111)-1980,98,98  
0350     98 IF(ITIMES(111)-691990,990,199  
0351     99 IF(ITIMES(111)-01990,99,99  
0352     99 IF(ITIMES(111)-231990,990,200  
0353     200 IF(ITIMES(111)-50200,200,200  
0354     200 IF(ITIMES(111)-1200,200,200  
```

C READ Ij STATEMENT
C
0355     850 READ (5,85) A1,A2  
0356     86 FORMAT(2AI)  
0357     85 IF(AI=11),85,A40  
0358     86 IF(A2=7),1040,5015,1996  

C PRINT CPU INPUT SUMMARY
C
0359     500 WRITE (6,4015)  
0360     4015 FORMAT(1L5X,21HCENTRAL PROCESSING UNIT SUMMARY/)  
0361     501 WRITE (6,4016) CPU  
0362     4016 FORMAT(/1A8 CPU MODEL NUMBER=,20X,I1)  
0363     501 WRITE (6,4017) VINST  
0364     4017 FORMAT(1HMAXIMUM INSTRUCTION TIME =,20X,F5.1,IX,12HMICROSECONDS)  
0365     501 WRITE (6,4018) AVSF  
0366     4018 FORMAT(1HMAXIMUM SELECTOR CHANNEL INTERFERENCE TIMEx,3X,F5.2,IX,12HMICROSECONDS PER BYTE)  
0367     501 WRITE (6,4019) AVFM  
0368     4019 FORMAT(1HMAXIMUM MULTIPLE CHANNEL INTERFERENCE TIMEx,3X,F5.2,IX,12HMICROSECONDS PER BYTE)  

C PRINT FOREGROUND AREA INPUT SUMMARY
C
0369     4020 FORMAT(///1A14X,H1NORMAL FOREGROUND AREA ASSIGNMENTS/)  
0370     5000 DO 5020 II=1,2  
0371     5020 WRITE (6,4021) II  
0372     4021 FORMAT(///1A16 FOREGROUND AREA,II)  
0373     5000 IF(II=1),5001,5027,5011  
0374     5001 WRITE (6,4022) II  
0375     4022 FORMAT(///1A14X,PROGRAM ASSIGNMENT TO THIS AREA)  
0376     5000 DO 5020 II=1,2  
0377     5001 IF(II=1),5012,5023,5012  
```
0355      5012 IF(ISG(11)-1)5013,5024,5013
0356      5013 TF(ISG(11)=1)5026,5025,.326
0357      5023 WRITE (6,4023)
0358      4023 FORMAT(16H0CARD READER, TRANSCRIPTION PROGRAM ASSIGNED TO THIS AREA)
0359      GO TO 5027
0360      5024 WRITE (6,4024)
0361      4024 FORMAT(15H0INTER TRANSCRIPTION PROGRAM ASSIGNED TO THIS AREA)
0362      GO TO 5027
0363      5025 WRITE (6,4025)
0364      4025 FORMAT(16H0TELECOMMUNICATIONS INPUT TRANSCRIPTION PROGRAM ASSIGNED TO THIS AREA)
0365      GO TO 5027
0366      5026 WRITE (6,4026)
0367      4026 FORMAT(16H0TELECOMMUNICATIONS OUTPUT TRANSCRIPTION PROGRAM ASSIGNED TO THIS AREA)
0368      GO TO 5027
0369      5027 WRITE (6,4027) PROGM(11)
0370      4027 FORMAT(16H0PROGRAM NAME'TS',5X,'AP')
0371      IFIXT=FINSTX(11)
0372      WRITE (6,4028) IFIXT
0373      4028 FORMAT(14H0NUMBER OF INSTRUCTIONS EXECUTED PER RECORD=,5X,'X')
0374      WRITE (6,4029) FGCUP(11)
0375      4029 FORMAT(16H0UNIT NUMBER FOR PERIPHERAL OPERATION IS',5X,'A')
0376      WRITE (6,4030) FGCUP(11)
0377      4030 FORMAT(17H0UNIT NUMBER FOR INTERMEDIATE STORAGE IS',5X,'A')
0378      IFIXT=FILEFC(11)
0379      WRITE (6,4031) IFIXT
0380      4031 FORMAT(17H0REJECTION FACTOR=',29X,'F')
0381      JI=FGCIP(11)
0382      5032 WRITE (6,4032) IFIXT
0383      4032 FORMAT(22H0AVERAGE LATENCY TIME=',23X,'X')
0384      IFIXT=FSFKN(11)
0385      4033 FORMAT(22H0AVERAGE SEEK TIME=',23X,'X')
0386      WRITE (6,4034) IFIXT
0387      5020 CONTINUE

C
C PRINT DEVICE INPUT SUMMARY
C
0389      DO 5030 II=1,4DEV
0390      GO TO 5034,5031,II
0391      5033 IF(ISG(11)=1)11,5000,5035,5001
0392      5034 WRITE (6,4034)
0393      4034 FORMAT(11H1,5X,16DEVICE SUMMARIES/)
0394      GO TO 5001
0395      5035 WRITE (6,4035)
0396      4035 FORMAT(11H1,5X,28DEVICE SUMMARIES CONTINUED/)


0397 5001 IF(DIAG(11))=.015003,5003,5002
0398 5002 IF(DIAG(11)=0,.5003,.5004)
0399 5003 IF(DIAG(11)=BLANK,0,.5004)
0400 5004 IF(DIAG(11)=BLANK)=.5005,.5007
0401 5005 IF(DIAG(11)=BLANK)=.5006,.5004
0402 5006 WRITE (6,4036) DEVTYP(I1)
0403 4036 FORMAT(/SH TYPE,IX,A4,IX,SHSTART)
0404 GO TO 5006
0405 5007 WRITE (6,4037) DEVTYP(I1),DEVMOD(I1)
0406 4037 FORMAT(/SH TYPE,IX,A4,IX,SHMOD,IX,A4,IX,SHSTART)
0407 GO TO 5006
0408 5038 WRITE (6,4038) DEVTYP(I1)
0409 4038 FORMAT(/SH TYPE,IX,A4,IX,SHAPL)
0410 GO TO 5006
0411 5039 WRITE (6,4039) DEVTYP(I1),DEVMOD(I1)
0412 4039 FORMAT(/SH TYPE,IX,A4,IX,SHMOD,IX,A4,IX,SHAPL)
0413 GO TO 5006
0414 5040 WRITE (6,4040) DEVTYP(I1)
0415 4040 FORMAT(/SH TYPE,IX,A4,IX,SHUNIT RECORD DEVICE)
0416 GO TO 5006
0417 5041 WRITE (6,4041) DEVTYP(I1),DEVMOD(I1)
0418 4041 FORMAT(/SH TYPE,IX,A4,IX,SHMOD,IX,A4,IX,SHUNIT RECORD DEVICE)
0419 5046 IFIXIT=DEVSPT(I1)
0420 IF(DIAG(11)=H)5042,5043,5042
0421 WRITE (6,4042) IFIXIT
0422 4042 FORMAT(/SH SPEED,48X,17X,SH币ENTES PER SECOND)
0423 GO TO 5006
0424 5043 WRITE (6,4043) IFIXIT
0425 4043 FORMAT(/SH SPEED,41X,17X,SHRECPE PER MINUTE)
0426 4043 IFIXIT=DEVSPT(I1)
0427 WRITE (6,4044) IFIXIT
0428 4044 FORMAT(/SHRECORD SIZE=,48X,17X,SHRECORD SIZE)
0429 5045 WRITE (6,4045) DEVSPT(I1)
0430 4045 FORMAT(/SHRECORD SIZE=,35X,17X,SHRECORD SIZE)
0431 IF(DIAG(11)=D)15007,5046,5047
0432 5047 IF(DIAG(11)=D)5061,5048,5049
0433 5049 IFIXIT=DEVSTAT(I1)
0434 WRITE (6,4046) IFIXIT
0435 4046 FORMAT(/SHRECORD SECTOR TIME=,35X,17X,SHRECORD SECTOR TIME)
0436 4046 IFIXIT=DEVSPT(I1)
0437 WRITE (6,4047) IFIXIT
0438 4047 FORMAT(/SHRECORD SECTOR TIME=,35X,17X,SHRECORD SECTOR TIME)
0439 GO TO 5061
0440 5048 IFIXIT=DEVSTAT(I1)/10
0441 WRITE (6,4048) IFIXIT
0442 4048 FORMAT(/SHRECORD SECTOR TIME=,35X,17X,SHRECORD SECTOR TIME)
0443 4048 IFIXIT=DEVSPT(I1)/10
0444 WRITE (6,4049) IFIXIT
4049 FORMAT(13HORIEND+20 4X,FS,1,IX,7HMINTES)    MOD 1719115
4046 IFIX=DEVDEM(I)               MOD 1719117
4047 WRITE (6,0460) IFIX           MOD 1719118
4048 4060 FORMAT('13HORIEND+20 4X,13,IX,14BYTES,PEF,TNH') MOD 1719119
4049 5061 WRITE (6,0461) DEVH(I)   MOD 1719120
4050 4061 FORMAT('13HORIEND+20 4X,13,IX,1MHITS,PER,BYTE') MOD 1719121
4051 K1=1
4052 DO 5029 J1=1,4CH;
4053 IF(CUJN(J1) .LT.5529,5049,5029)
4054 5008 PRCWU(J1)=CUJN(J1)
4055 K1=K1+1
4056 IF(K1-30)5029,5029,5062
4057 5062 K1=1
4058 WRITE (6,0462) (PRCWU(J1),1=1,30)
4059 4062 FORMAT(13HORIEND+20 4X,13,IX,43,IX)
4060 5029 CONTINUE
4061 IF(K1-11)5030,5030,5063
4062 5063 K1=K1-1
4063 WRITE (6,0463) (PRCWU(J1),1=1,K1)
4064 5030 CONTINUE
4065 11=1
4066 12=1
4067 13=1
4068 14=1
4069 15=1
4070 16=1
4071 17=1
4072 DAY=BLANK4
4073 WMJOB=WMJOB
4074 C
4075 C  PROCESS JB, JF, JBD, JBF, JAF, JB10, AND JB10 STATEMENTS
4076 870 DO 170 TJ=11,WMJOB
4077 C
4078 C  DETERMINE WHICH STATEMENT IS BEING PROCCSSED
4079 C
4080 875 GO TO (88,871,872,873,874,875,876,877,878,879,880,17) MOD 1719261
4081 871 17=1
4082 872 GO TO 95 MOD 1719262
4083 873 17=1
4084 874 GO TO 999 MOD 1719263
4085 875 17=1
4086 876 GO TO 999 MOD 1719264
4087 877 GO TO 999
C READ A JB, JM, OR JJ STATEMENT
C
88 READ 15,89) A1,A2,A3,JOBNAME(I),STATEMENT(F),CTRLM(I),
88 JOBNAME(I),DOEF(I),HOF(I),MARI(I),VARS(1),PREDEF,NEWLP
C
89 FORMAT(341,2X,2AB,11,7210,11,72,12,245,0,23,21)
89 IFIA1-J1000,,00,1099
49 IFIA2-Z191,196,91

C IS THIS STATEMENT FOR THE SAME DAY AS THE PREVIOUS STATEMENT
C
91 IF(AABS(A21)-ABS(A20))115011501-155159

C YES, CHECK BEGIN AND END TIMES FOR PROGRAM
C
92 CALL TOD(WEEK,MON(I),TOD(I),FREQ(I),DOEF(I),HOF(I),MARI(I),
92 MON(I),DOEF(I),HOF(I),MARI(I),VARS(1),PREDEF,NEWLP)

C WERE THERE ANY ERRORS IN THE STATEMENT
C
IF(ERROPP1-11920,1092,1092)

C NO ERRORS IN STATEMENT, IS THIS A DAILY OR WEEKLY PROGRAM
C
920 IFABS(A21)-ABS(A20))115011501-155159

C WEEKLY PROGRAM, DO BEGIN AND END TIMES AGREE WITH PROGRAM TYPE
C
95 IFITCASE=31003,95,1093

C DAILY PROGRAM, WAS TURNAROUND TIME OF PREVIOUS PROGRAM A FULL WORKING DAY
C
94 GO TO 1440,94,14

C YES, DO BEGIN AND END TIMES AGREE WITH PROGRAM TYPE
C
940 IFITCASE=111940,95,1940
C PARTIAL WORKING DAY. IS THE TURNAROUND TIME OF THIS PROGRAM A PARTIAL DAY

0507 941 IF(IGE=21942,95,1093) M0018700
C
C NO. IS THE TURNAROUND TIME OF THIS PROGRAM A FULL WORKING DAY

0508 942 IF(IGE=11093,943,1093) M0018713
C
C WAS THE PREVIOUS PROGRAM A BACKGROUND PROGRAM OR A FOREGROUND PROGRAM

0509 95 GO TO (97,961,12) M0018720
C
C FOREGROUND. IS THIS PROGRAM A FOREGROUND PROGRAM

0510 96 IF(A3-F11096,980,1096) M0018732
C
C PREVIOUS PROGRAM BACKGROUND. IS THIS PROGRAM A BACKGROUND PROGRAM ALSO

0511 97 IF(A3-B11981,980,981) M0018743
C
C NO. IS THIS PROGRAM A FOREGROUND PROGRAM

0512 981 IF(A3-F11097,982,1097) M0018753
C
C YES. SET SWITCH TO FOREGROUND

0513 982 I=2
0514 GO TO 980 M0018760
C
C PREVIOUS PROGRAM TURNAROUND TIME WAS PARTIAL DAY, CURRENT PROGRAM FULL DAY
C SAVE POINTER TO END OF PARTIAL WORKING DAY TABLE
C HAVE PROGRAM DESCRIPTION TO FIST FULL DAY PLACE IN TABLE
C SET SWITCHES TO BACKGROUND, FULL WORKING DAY, AND PROGRAM TYPE CHANGE

0515 943 MXJ09=IJJ-1 M0018780
0516 MMJ3M=J00 M0018791
0517 JJ=MXJQ9+1 M0018800
0518 CALL SWITCH(JORNAM(JJ),STPMXJ(JJ),H0PMT(JJ),MONT(JJ),MONT(JJ),
1DNFF(JJ),H0PFF(JJ),J0RMAM(JJ),STPNXJJ(JJ),H0PMT(JJ),MONT(JJ),
2MONT(JJJ),DNFF(JJJ),H0PFF(JJJ),J0RNM(JJJ)
0519 IJJ=JJ+1 M0018840
0520 JJ=JJ+1 M0018850
0521 I=I+1 M0018860
0522 I<2 M0018870
0523 GO TO 95 M0018880
C
C THIS PROGRAM IS FOR A DIFFERENT DAY THAN THE PREVIOUS PROGRAM
<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0525</td>
<td>910</td>
<td>GO TO 912, 913, 914, 915, 916, 917, 918, 919, 920</td>
</tr>
<tr>
<td>0526</td>
<td>911</td>
<td>IF (ABS(A2-ABS(4))) 911, 912, 913, 914, 915, 916, 917, 918, 919</td>
</tr>
<tr>
<td>0527</td>
<td>912</td>
<td>IF (ABS(A2-ABS(2))) 911, 912, 913, 914, 915, 916, 917, 918, 919</td>
</tr>
<tr>
<td>0528</td>
<td>913</td>
<td>IF (ABS(A2-ABS(1))) 911, 912, 913, 914, 915, 916, 917, 918, 919</td>
</tr>
<tr>
<td>0529</td>
<td>914</td>
<td>IF (ABS(A2-ABS(0))) 911, 912, 913, 914, 915, 916, 917, 918, 919</td>
</tr>
<tr>
<td>0530</td>
<td>915</td>
<td>IF (ABS(A2-ABS(5))) 911, 912, 913, 914, 915, 916, 917, 918, 919</td>
</tr>
<tr>
<td>0531</td>
<td>916</td>
<td>IF (ABS(A2-ABS(0))) 911, 912, 913, 914, 915, 916, 917, 918, 919</td>
</tr>
<tr>
<td>0532</td>
<td>917</td>
<td>IF (ABS(A2-ABS(7))) 911, 912, 913, 914, 915, 916, 917, 918, 919</td>
</tr>
<tr>
<td>0533</td>
<td>921</td>
<td>15=2</td>
</tr>
<tr>
<td>0534</td>
<td>922</td>
<td>GO TO 912</td>
</tr>
<tr>
<td>0535</td>
<td>923</td>
<td>15=3</td>
</tr>
<tr>
<td>0536</td>
<td>924</td>
<td>GO TO 913</td>
</tr>
<tr>
<td>0537</td>
<td>925</td>
<td>15=4</td>
</tr>
<tr>
<td>0538</td>
<td>926</td>
<td>GO TO 914</td>
</tr>
<tr>
<td>0539</td>
<td>927</td>
<td>15=5</td>
</tr>
<tr>
<td>0540</td>
<td>928</td>
<td>GO TO 915</td>
</tr>
<tr>
<td>0541</td>
<td>929</td>
<td>15=6</td>
</tr>
<tr>
<td>0542</td>
<td>930</td>
<td>GO TO 916</td>
</tr>
<tr>
<td>0543</td>
<td>931</td>
<td>15=7</td>
</tr>
<tr>
<td>0544</td>
<td>932</td>
<td>GO TO 917</td>
</tr>
</tbody>
</table>

C IS IT A VALID DAY

0545 IF (IS-WEK) 917, 919, 911

C YES, SAVE POINTERS TO THE BEGINNING AND END OF PARTIAL WORKING DAY TABLE

0546 IF (MM=JOB=MM) JJ=MM=JOB+1

C WHAT WAS THE JOB TYPE OF THE PREVIOUS PROGRAM

0548 GO TO 927, 928, 929, 930

C PREVIOUS PROGRAM WAS FULL WORKING DAY, SAVE POINTER TO END OF THAT TABLE

0549 MXJOB=1J-1

C PREVIOUS PROGRAM WAS PARTIAL WORKING DAY, SAVE POINTER TO THE END OF THAT TABLE AND INDICATE THAT FULL WORKING DAY TABLE IS EMPTY

0550 GO TO 930

C SCHEDULE JOBS FOR THE PREVIOUS DAY

0551 MXJOB=1J-1

C END OF CODE
C 0550 '927 MWJOB=J-1'         MAP19319
C MOV PROGRAM DESCRIPTION TO FIRST PARTIAL DAY PLACE IN TABLE
C 0557 '931 CALL SWITCH (JONAM(JJ),STEPH(JJ),DON(JJ),HON(JJ),(JON(JJ),
   10OFF(JJ),HOFF(JJ),MFF(JJ),JONAM(JJ),STEPH(JJ),DON(JJ),HON(JJ),
   2*DON(JJ),10OFF(JJ),HOFF(JJ),MFF(JJ))'      MAP19337
C ANY DAYS SKIPPED BETWEEN CURRENT PROGRAM'S DAY AND PREVIOUS PROGRAM'S DAY
C 0558 'IF(KI-1)*1934,933,934'        MAP19357
C YES, INDICATE NULL TABLES AND SCHEDULE THOSE DAYS
C 0559 '937 MWJOB=JWJOB'             MAP19547
0560 MWJOB=MWJOB          MAP19547
0561 L1=1F-1
0562 KI=13
0563 '933 17=0'             MAP19597
0564 'GO TO 6000'           MAP19597
0565 '935 KI=KI+1'          MAP19597
0566 'IF(KI-1)*1934,933,934' MAP19597
C SET SWITCHES TO BACKGROUND, PARTIAL WORKING DAY, AND PROGRAM TYPE CHANGE
C SET DAY EQUAL TO THIS PROGRAM'S DAY. UPDATE NEXT DAY SWITCH
C 0567 '934 JJ=JJ'          MAP19607
0568 JJ=JJ+1
0569 'DAY=2'              MAP19617
0570 'I=15+1'             MAP19627
0571 'I3=15'              MAP19627
0572 'I2=1'               MAP19627
0573 'I4=2'               MAP19627
0574 'I6=2'               MAP19627
0575 'GO TO 92'           MAP19627
C CALCULATE OVERLAPPED AND NON-OVERLAPPED TIME ON CPU
C 0576 TCPUI(3,II,J)=FIXINS*AVVNST*(1.-NOVLP)
0577 TNOVLP(3,II,J)=FIXINS*AVVNST*NOVLP
C
C IS THIS PROGRAM A FOREGROUND PROGRAM OR A BACKGROUND PROGRAM

C GO TO (987,986,985,14)
C
C BACKGROUND PROGRAM. IS THIS PROGRAM FOR THE SAME JOB AS THE PREVIOUS PROGRAM

C

C SAME JOB. VERIFY STEP NAMES AND BEGIN AND END TIMES

C

C DIFFERENT JOBS. CHECK FOR DUPLICATE JOB NAMES

C

C READ JOB OR JFD STATEMENTS

C

C IS THIS PROGRAM A BACKGROUND PROGRAM OR A FOREGROUND PROGRAM

C GO TO (981,986,14)
C
C BACKGROUND PROGRAM

C

C

C
C READ A JBF STATEMENT
0604 READ (5, 99) A0, A5, IA, A7, A8, FILEG3(IJ, IFC)  M0020000
0605 99 (FILEG3)  M0020000
0606 IF(A0=JI)109, 999, 1090  M0020000
0607 993 (FILEG3)  M0020000
0608 101 (FILEG3)  M0020000
0609 102 (FILEG3)  M0020000

C VERIFY FOREGROUND PROGRAM TYPE
0610 IF(A7=BLANK0) 105, 104, 105  M0020000
0611 IF(A8=BLANK0) 104, 113, 1104  M0020000
0612 IF(A7=P107, 106, 107  M0020000
0613 IF(A7=BLANK0) 1104, 115, 1104  M0020000
0614 IF(A7=P108, 109, 109  M0020000
0615 IF(A7=BLANK0) 1104, 116, 1104  M0020000
0616 IF(A7=T109, 111, 1109  M0020000
0617 IF(A8=1117, 117, 117  M0020000
0618 IF(A8=U1112, 118, 1112  M0020000

C THERE IS NO FOREGROUND PROGRAM REFERRED TO BY THIS STATEMENT
0619 113 FILEG3(IJ, IFC)=O  M0021000
0620 GO TO 114  M0021600

C FOREGROUND PROGRAM IS A PRINTER TRANSCRIPTION PROGRAM
0621 115 FILEG3(IJ, IFC)=P  M0021700
0622 GO TO 119  M0021800

C FOREGROUND PROGRAM IS A READER TRANSCRIPTION PROGRAM
0623 116 FILEG3(IJ, IFC)=R  M0021900
0624 GO TO 119  M0022000

C FOREGROUND PROGRAM IS A TELECOMMUNICATIONS INPUT PROGRAM
0625 117 FILEG3(IJ, IFC)=T  M0022100
0626 GO TO 119  M0022200

C FOREGROUND PROGRAM IS A TELECOMMUNICATIONS OUTPUT PROGRAM
0627 118 FILEG3(IJ, IFC)=O  M0022300

C WHICH FOREGROUND AREA HAS THIS FOREGROUND PROGRAM TYPE
0628 119 IF(FILEG3(IJ, IFC)=FG(I11121, 120, 121  M0022400
C

C PRINT PROGRAM SUMMARY

C

0651 130 TCPH(3,1,1)=TCPH(3,1,1)
0652 TNOVL0(3,1,2)=TNOVL0(3,1,1)

C

C PRINT FILE ASSIGNMENTS
C
C WRITE (6,4078)
C 4078 FORMAT(///39H INPUT/OUTPUT UNIT ASSIGNMENTS///)
C D0 5074 IFIXIT=1,FILE
C IF(FILNAMETIXIT),15079,15074,15079
C 15079 1PI=FILENT(IFIXIT)
C GO TO 5074
C 5074 WRITE (5,4079) FILNam(IFIXIT),CHU(IFIXI)
C 4079 FORMAT(5H FILE,IX,A7,IX,24H ASSIGNED TO UNIT NUMBER,IX,A3)
C GO TO 5074
C 5078 DD 5077 IK=1,2
C IF(FILG(IK)-BLANK4)5076,5077,5078
C 5076 IF(FILGIF(IK),1K1-FILNAM(IFIXIT)1)15077,5074,5077
C 5077 CONTINUE
C GO TO 5079
C 5079 CONTINUE
C GO TO (15070,15075),12
C 15070 DD 5075 IK=1,2
C IF(FILG(IK)-BLANK4)5076,5078,5080
C 5080 WRITE (5,5080) FILEG(IJ,IK)
C 5080 FORMAT(5H FILE,IX,A7,IX,24H ASSIGNED TO UNIT NUMBER,IX)
C 5075 CONTINUE
C 15075 1PI=0
C C READ A JAI0 OR JEU STATEMENT
C 131 CALL FILEIO (A,0,II,0,II,01,02,PLANK1,1A2,13,4C,FILE,FLNM,
C IFLPNT,CUPNT,DEVLCS,DEVLSS,DEVPW,VOLUME,FPST,1,VECTOR,K1,
C 21,TCY,SEK,FACTOR,10PNT,NAME,80HTY,ERROR,ERRORX,CLASS,DEVICE,34,
C 304,051
C C WERE THERE ANY ERRORS IN THE STATEMENT?
C 1 IFERRORX=1132,1310,1310
C C ERRORS IN STATEMENT, BYPASS TIMING SUBROUTINES
C 014 1310 ERRORX=0
C C0 TO 153
C 1310 ERRORX=0
C C0 TO 153
C C NO ERRORS IN STATEMENT, PRINT FILE DESCRIPTION SUMMARY
C C 016 132 1PI=1PI+1
C 017 1F14(1PI/3)+1-1PI15082,5091,50P2
C 018 5081 WRITE (6,5081)
C 019 5081 FORMAT(1H1,52X,27H PROGRAM SUMMARY (CONTINUED))
C 020 5082 WRITE (6,5082) NAME
C 021 4082 FORMAT(///5H FILE,IX,A7)
C
C IS THIS A FOREGROUND PROGRAM OR A BACKGROUND PROGRAM

C 1320 GO TO (133,138),12
C BACKGROUND PROGRAM. CHECK FOREGROUND FILES AGAINST CURRENT FILE NAME

C 133 DO 137 IFG=1,2
C 134 IFNAME=FILEG(IFG)
C 135 IF(CLASS=U)1135,1146,1138
C CURRENT FILE IS A FOREGROUND FILE. IS ITS DEVICE CLASS UNIT RECORD
C YES. SAVE INFORMATION FROM INTO STATEMENT AND BYPASS TIMING SUBROUTINES

C 136 FILECTR(IFG)=FACTOR
C 137 FLOOPY(IFG)=100NT

C YES. SAVE INFORMATION FROM INTO STATEMENT AND BYPASS TIMING SUBROUTINES
i

i
FI~~~~~~~~~

i

i

i

t
I

C. I

II

r

~~

~

I

II

i

,

i

CI', i -

I

,i c

.(

c

I

·.

i

i

I I tI

f

I

~~~~~~~~I

F

<

L

CrC C

c

·

I-

'LL~~~~~~~~~L

7-

I

I

:,~~~v

re-J

·

L~

I

L,

'

c,cc. .,c

LLI~~~~~~~I~~~~(
I

L
)

LLr

~~Ii.:!

!
.

I-.-.I*t

<!,'-I-I ~~

-(l

~

r'~
'P

!~ 0.

I

'

~~~r
4 -zlZi

l..u,

.

C
0'

-".

U-,

U CC~
F

U)-

r
~: u~~
-.,. '--,I...
-C--2LC

-

~i

~ r
p
U.
~~~~~~~~~~~~~1,,c

i ,L

0.I

Z~~~~~~~~~~~~i

--rr~,

:-:
-C'

-

0..

O_
·...
, .¢ f'¢. !U
> 1_*L'L->

C>p--.
i" L'·

-C

*-:.--.

C

'

C

,.

I

~~~'

~~
.Z

.

" ]:~ t.,.
'.CCri~
h .
LU

-

C

r~LP

~~~~~~
C

C'
C
L~~~~
...
.

i ~~~~'
c =i " ,. 1r
~~~~~C
!--2

U

C-a

C

)/'

l

C'> r~Li

.C
nC.C)

C

-"--

:1

-

',

I

-Uc

'C

j~~~~/

~~~

15
a. CZ

j

Lr3~7
L..: LU-

'
I

m r

=:

it
C"

,-

LU

c~ _N

o

Ur.r"

1"-i.
~ . ~

.· ,

i

I
i

I

i

I

i

.

I

~~~~~~-- Ur,
-

.~

F

CC

C

CC

4C

C-L

6- LU-.'",
:1 I

u~O
cj
I"~

I- I~"
'I

"~
' V.

-

-

'

> i-.-

1±
O

Cj

-.T - -C

-C

Ul~

e

F

-

'

:.:..

....~

:-

-

i:

C
c:

,

C

--

~C

,{.~

I--*

- ""L .l
i, "
.l l-

'~ iF

i a i 7 ci("

:

j

Z

u.Ivt

r,

r\;~~~~C·
~.'",

i

r

UJ>

'~

-C)LLU>

·:

II
i
I

i

.Z

r

U)
L

' ~.
i 0'.I%~~
rc~`~-·
'-- CI
J CT
.CC!' ~U

ll/
,l~
~ l'-'i

i

I

~-:.-".
L I--f-L

'-f

tl.r;4

L

,

-,,',,I

sC" 0.! M
'CIUIZ IM
0
-- - -.-

'

Y(
·.
>

r:~~~~~~~I
>-l
-

,

l.

:ZU-U~.2~~~t
,,~ U-u -U) i f',, ' ,.~
C'

I

.

ICj'-

<
~C

i
t

'

I i~~~~~~~~~~~~~~~~~~~~~~~~~~
U-

i

i

~·

.

,--

~~c c:
74 ~.
rc- cc~i

l

F

~.
ji

ci

i
I
i

I
I

O.L cc 1V

r (*-.C .
· II~j

Li.-.

--

<C
',l~

II

II

LC'

30'

I'

i

I

i
i
I
I
iI

1I

i

e
j>~r
h)7LL -'-JLt.C

C

'

-C-I=

'L1!

iI
I

c

C,
U(-U
·, er',
U.

.,,,.,C'
C

-c'l-

--

I
I

t- ; e,,. ,,~~>

-- ~

i

i

73c

Liu~~~~~~~
C:L~~~~~
C~
L~i
.'

7

i

I

I~

i

LI e

.

nU

i

t
I

Ii

~->i

Li It. 'Z
C'~~~~~~~~~~~~~=
' !

F

~I'

I

I ' :5

~ .

i
L'

~

i

i

I

c

U

I

cc.z

i

i

I

i ·C

-t

-L1

iI

~-

~

I.-.
'i

CLt~
> L.:
.
i.,-.
i,;
· <;
'lLL;Cr · ~L

~

.

F.

I

<~'o

c c

:~, 11'~~~~~~~~~~~r
'
:
t
I

WU
">

I

i

e
C-vC,'

I--1
1,
M C,,L

--

i
I

It~

I

Il

-C'j e'\>

i

I

L

i
IT

I~~~~~~~~

i
I.
I

i

r.c.,C

CC,

I

i

i

~~~I
~ I·

I

I

r"h
r
I

m

:'

I

~ L.J
'
C:

'

L

C
· a;' I· ~4
' ,..9I
imI·
~-r
A ch ~ t%

1

C.C

))
F

-

'-C, ,-'-

'

I'-2

i·


C CHECK FOR ADDITIONAL "JTO" OR "JEDA" STATEMENTS

C
0781 153 IFFIGHT=BLANK4113111541,131
0782 1540 IF=1
0783 1544 TEIFA-MFILE11541,1541,1550
C CHECK THAT ALL FILES HAVE BEEN ASSIGNED TO A UNIT

C
0784 1541 ON 155 JIF=1,MFILE
0785 1FFI NAM IJI-BLANK811154,155,1154
0786 155 CONTINUE
C
0787 IS THIS A BACKGROUND PROGRAM OR A FOREGROUND PROGRAM

C
0788 1551 JCASE(J1)=3
0789 GO TO 270
C
C BACKGROUND, CALCULATE DEVICE TIMES FOR FOREGROUND FILES
C
0790 156 DO 277 IFG=1,2
C FACTOR=FLACK(IFG)
0791 10PN=FLMPT(IFG)
C
0792 DEVICE=FILEDVF(IFG)
0793 MOLT=FLMULT(IFG)
0794 VOLUME=FLVOL(IFG)
C
C WHICH FOREGROUND PROGRAM CAN PROCESS THIS FILE

C
0796 1IFELEG$T(J,IFG)=1,77,158,157
C
0797 157 IF(FLG$T(J,IFG)=11)159,160,277
C
C FOREGROUND PROGRAM ONE
C
0798 159 K1=1
0799 K2=1
0800 K3=1
0801 GO TO 161
C
C FOREGROUND PROGRAM TWO
C
0802 159 K1=2
C DISK DEVICE, COMPUTE DEVICE, CHANNEL, CPU, AND SET UP TIMES

C

0834 166 CALL DISK (TDEV1(FGX), IJ, CH1), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ)

0835 CALL DISK (TDEV1; iJ, CH1), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ)

0836 GO TO 168

C TAPE DEVICE, COMPUTE DEVICE, CHANNEL, CPU, SET UP, AND REWIND TIMES

C

0837 167 CALL TAPE (TDEV1(FGX), IJ, CH1), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ)

0838 CALL TAPE (TDEV1; IJ, CH1), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ)

0839 CALL TAPE (TDEV1; IJ, CH1), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ)

C COMPUTE DEVICE, CHANNEL, CPU, AND SET UP TIMES FOR UNIT RECORD DEVICE

C

C

0840 168 CALL UNIT (TDEV1(FGX), IJ, CH1), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ), TMRX(N(TMGX), IJ, IJ)

Note: The text appears to be a FORTRAN IV source code listing for a computer program, containing calls to various functions and operations involving disk and tape devices, as well as compute times.
GO TO 177

C BOTH FOREGROUND PROGRAMS CAN PROCESS FILE. SELECT PROPER "RUNNING" SUBROUTINES

GO TO 170, 171, 174

C IS THE "INTERMEDIATE STORAGE DEVICE" DISK OR TAPE?

GO TO 169, 165, 167, 168

C DISK DEVICE, "COMPUTE DEVICE," CHANNEL, CPU, AND SET UP TIMES

CALL DISKX (SDEVI, FGCX, IJ, JIJ, TMPX1, TMPX2, FGCX, JJ, JJJ2, 1)

CALL DISKX (SDEVI, FGCX, IJ, JIJ, TMPX1, TMPX2, FGCX, JJ, JJJ2, 2)

C TAPE DEVICE, "COMPUTE DEVICE," CHANNEL, CPU, SET UP, AND DEWING TIMES

CALL TAPEF (SDEVI, FGCX, IJ, JIJ, TMPX1, TMPX2, FGCX, JJ, JJJ2, 1)

CALL TAPEF (SDEVI, FGCX, IJ, JIJ, TMPX1, TMPX2, FGCX, JJ, JJJ2, 2)

C DISK DEVICE, "COMPUTE DEVICE," CHANNEL, CPU, AND SET UP TIMES

CALL DISKX (SDEVI, FGCX, IJ, JIJ, TMPX1, TMPX2, FGCX, JJ, JJJ2, 1)

CALL DISKX (SDEVI, FGCX, IJ, JIJ, TMPX1, TMPX2, FGCX, JJ, JJJ2, 2)

C TAPE DEVICE, "COMPUTE DEVICE," CHANNEL, CPU, SET UP, AND DEWING TIMES

CALL TAPEF (SDEVI, FGCX, IJ, JIJ, TMPX1, TMPX2, FGCX, JJ, JJJ2, 1)

CALL TAPEF (SDEVI, FGCX, IJ, JIJ, TMPX1, TMPX2, FGCX, JJ, JJJ2, 2)
C IS THIS A NON-CONFLICT BACKGROUND PROGRAM

C GO TO I(72,178,178),11
C YES, CAN BOTH FOREGROUND PROGRAMS PROCESS "THE SAME" FILE

C YES, CALCULATE RESOURCES USE FRACTIONS

C CALL FRCN (TOTAL, 3, 1, 3, 3, 1, 3, 1, 2), 11, 1, 2, 1, 3, 2, 1

C 178 GO TO I(1179,1971,16)
C NO, "READ" STATEMENTS FOR "NEXT" PROGRAM

C 179 CONTINUE
C NO MORE ROOM IN TABLE FOR CURRENT PROGRAM TYPE, "DETERMINE" PROGRAM TYPE

C WEEKLY, SAVE POINTER TO END OF TABLE AND SET DAY TO 1

C PARTIAL WORKING DAY, SAVE POINTER TO THE END OF THAT TABLE
C SET LOOP LIMITS TO BEGINNING AND END OF FULL WORKING DAY TABLE
C SET SWITCHES TO BACKGROUND AND FULL WORKING DAY
C FULL WORKING DAY. SAVE POINTER TO THE END OF THAT TABLE
C
C SCHEDULE JOBS FOR THE CURRENT DAY AND UPDATE CURRENT DAY SWITCH
C
0941
17=9
0942
K1=13-1
0943
GO TO 6000
0944
IX3=13-1
0945
GO TO (185,186,187,188,189,191,192),IX3
0946
185 DAY=02
0947
GO TO 196
0948
186 DAY=03
0949
GO TO 196
0950
187 DAY=04
0951
GO TO 196
0952
188 DAY=05
0953
GO TO 196
0954
189 DAY=06
0955
GO TO 196
0956
190 DAY=07
0957
GO TO 196
C
C READ J2 STATEMENT
C
0958
192 READ (5,194) A1,A2
0959
193 FORMAT (I11,A1)
0960
194 IF (A1=J1) DAY194,1193
0961
195 IF (A2=21194,1195)
C
C END OF PROGRAM
C
0962
196 CALL EXIT
C
C ANY MORE DAYS IN WEEK
C
0963
197 IF (I5=WEKN4,184,192)
C
C YES, SET DO LOOP LIMITS TO BEGINNING AND END OF PARTIAL WORKING DAY TABLE
C
C SET SWITCHES TO BACKGROUND AND PARTIAL WORKING DAY, UPDATE NEXT DAY SWITCHES
C
FORTRAN IV G LEVEL 0, MOD 2  

0014  184 MMJOB=MMJOB  
0015  11=MMJOB+1  
0016  I2=1  
0017  13=12+1  
0018  14=2  
0019  15=15+1  
0020  GO TO 870  

C PROGRAM TYPE CHANGED. SET NO LOOP LIMITS TO BEGINNING AND END OF CURRENT  
C TABLE AND RESET "PROGRAM CHANGE" SWITCH  
C  
0021  197 MMJOB=MMJOB  
0022  11=1J  
0023  16=1  
0024  GO TO 870  

C JZ STATEMENT READ. DETERMINE PROGRAM TYPE OF PREVIOUS PROGRAM  
C  
0025  198 GO TO 1201,700,791,14  

C WEEKLY. SAVE POINTER TO END OF THAT TABLE  
C  
0026  199 MXWJOB=IJ-1  
0027  GO TO 293  

C PARTIAL WORKING DAY. SAVE POINTER TO "END OF THAT TABLE" AND INDICATE THAT  
C FULL WORKING DAY TABLE IS EMPTY.  
C  
0028  200 MXWJOB=IJ-1  
0029  MXWJOB=MXWJOB  
0030  GO TO 202  

C FULL WORKING DAY. SAVE POINTER TO THE END OF THAT TABLE  
C  
0031  201 MXJOB=IJ-1  
C SCHEDULE JOBS FOR THE PREVIOUS DAY  
C  
0032  202 17=10  
0033  K1=13-1  
0034  GO TO 6000  

C ARE THERE ANY MORE DAYS IN THE WEEK  
C  
0035  206 IF(WEEKEND=203,203,205)  
C YES. INDICATE NULL TABLES AND SCHEDULE THOSE DAYS  
C
FORTRAN IV G LEVEL 0, MOD 0

0015 203 MWJOB=MWJOB
0017  WJOB+MWJOB
0020    Y1=13
0024    104 IT=IT
0026    GO TO 6000
0031  207 K1=K1+1
0033  IF(K1=1)204,204,205
0035  END OF PROGRAM
0039  CALL EXIT
0043  SCHEDULE THE JOBS FOR THE CURRENT DAY
0047  SCHEDULE THE JOBS
0049  IFERROR=116999,9999,9999
0053  NO. SCHEDULE THE JOBS
0055  ISIJ=MWJOB+1
0057  BEGIN=0000000,*(60.\*HON(K1)+HON(K1))
0060  END=0000000,*(60.\*HON(K1)+HON(K1))
0063  LS=-1
0065  ARE THERE ANY FULL WORKING-DAY JOBS FOR THIS DAY
0069  IF(MWJOB-MWJOB+119999,6001,6001
0073  YES, FIND OPTIMAL SEQUENCE FOR THE FULL WORKING-DAY JOBS
0077  WHICH SEQUENCING CASE IS THIS
0081  WHICH SEQUENCING CASE IS THIS
0085  GO TO 16007,6007,6007,6007,6007,6007,6007,6007,6007,6007,6007
0089  THREE-MACHINE SEQUENCING PROBLEM
0093  6007 DO 6070 IS=ISIJ,MWJOB
0095  THIS IS A NON-CONFLICT BACKGROUND PROGRAM
0099  TFJCASE(IS)=216007,6070,6070
C    YES. CALCULATE MULTIPROGRAMMING TIME
C
0060 DO 6020 JS=1,12
       6020 DO 6020 KS=1,2
       6047 BGERCMJJS,KS)=FRACM1JS,KS)
       6049 BGERCMJJS,KS)=FRACM1JS,KS)
       6050 DO 6020 BGERCMJJS,KS)=FRACM1JS,KS)
       0960 CALL TIME23(FRCM1,FRCM2,FRCM3,FRCM4,FRCM5,FRCM6,FRCM7,FRCM8,FRCM9,FRCM10)
       0961 IGERCMJ,FRCM1,FRCM2,FRCM3,FRCM4,FRCM5,FRCM6,FRCM7,FRCM8,FRCM9,FRCM10)
       0962 CALL TIME23(FRCM1,FRCM2,FRCM3,FRCM4,FRCM5,FRCM6,FRCM7,FRCM8,FRCM9,FRCM10)
       0963 IGERCMJ,FRCM1,FRCM2,FRCM3,FRCM4,FRCM5,FRCM6,FRCM7,FRCM8,FRCM9,FRCM10)
C
C    ARE THERE PARALLEL OPERATIONS ON EITHER INPUT OR OUTPUT
C
0961 GO TO (6004,6005,6006,6007,6008,6009,6010) CASE
C
C    YES. CALCULATE ALTERNATE MULTIPROGRAMMING TIME
C
0962 DO 6010 JS=1,12
       6010 DO 6010 KS=1,2
       0964 RGERCMJJS,KS)=FRACM1JS,KS)
       0966 RGERCMJJS,KS)=FRACM1JS,KS)
0966 6030 RGERCMJJS,KS)=FRACM1JS,KS)
       0967 CALL TIME23(FRCM1,FRCM2,FRCM3,FRCM4,FRCM5,FRCM6,FRCM7,FRCM8,FRCM9,FRCM10)
       0968 IGERCMJ,FRCM1,FRCM2,FRCM3,FRCM4,FRCM5,FRCM6,FRCM7,FRCM8,FRCM9,FRCM10)
       0969 CALL TIME23(FRCM1,FRCM2,FRCM3,FRCM4,FRCM5,FRCM6,FRCM7,FRCM8,FRCM9,FRCM10)
       0970 IGERCMJ,FRCM1,FRCM2,FRCM3,FRCM4,FRCM5,FRCM6,FRCM7,FRCM8,FRCM9,FRCM10)
C
C    SAVE LARGER TIME FOR USE IN SEQUENCING
C
0968 IFL(TOTAL13,TS,11/TIMEC)-TOTAL13,15,21/TIMEC16037,6037,6037
0970 6031 DO 6039 JS=1,12
       6039 DO 6039 KS=1,2
       0971 BGERCMJJS,KS)=FRACM1JS,KS)
       0972 BGERCMJJS,KS)=FRACM1JS,KS)
0973 6039 BGERCMJJS,KS)=FRACM1JS,KS)
       0974 TIMEC=TIMEC
       0975 SCHED(13,11)=TIMEC
0976 IFL(S1S1J=16039,6037,6037
C
C    IS THIS PROGRAM FOR THE SAME JOB AS THE PREVIOUS PROGRAM
C
0977 6037 IFL(JOBNAM(1S)=JOBNAM(1S)=116038,6139,6039
C
C    NO. WHICH PARALLEL OPERATION CASE IS THIS
C
0978 GO TO (6033,6034,6035,6036,6037,6038,6039) CASE
C
C    INPUT: SAME OPERATION, SAME MACHINES
0970  6033 QCTES1=TOTAL(3,TS,2)+T1MEX+TSETUP(3,TS,2)+TWNDUP(3,TS,2)
0980  GO TO 6144
0981  C OUTPUT. SAME OPERATION, SAME MACHINES
0982  6034 QAIL(S)=TOTAL(3,IS,2)+T1MEX+TSETUP(3,IS,2)+TWNDUP(3,IS,2)
0983  GO TO 6144
0984  C INPUT. SAME OPERATION, DIFFERENT MACHINES
0985  6035 QCNS1=TOTAL(3,IS,2)+T1MEX+TSETUP(3,IS,2)+TWNDUP(3,IS,2)
0986  GO TO 6144
0987  C OUTPUT. SAME OPERATION, DIFFERENT MACHINES
0988  6036 QAINS1=TOTAL(3,IS,2)+T1MEX+TSETUP(3,IS,2)+TWNDUP(3,IS,2)
0989  GO TO 6144
0990  C THIS PROGRAM IS FOR THE SAME JOB AS THE PREVIOUS PROGRAM
0991  6133 NS=LS-1
0992  GO TO 6133,6134,6135,6136,6137,
0993  C WHICH PARALLEL OPERATION CASE IS THIS
0994  6133 QCNS1=QCNS1+TOTAL(3,IS,2)+T1MEX+TSETUP(3,TS,2)+TWNDUP(3,TS,2)
0995  GO TO 6144
0996  C NO PARALLEL OPERATIONS OR DIFFERENT OPERATIONS IN PARALLEL.
FORTRAN IV G LEVEL 0, MOD O

0997 6004 'SCHFD(3,IS,1),TMEF
1000 IF(IS-ISJ-1),6005,6006

C IS THIS PROGRAM FOR THE SAME JOB AS THE PREVIOUS PROGRAM?
1001 6005 IF (JOURNAL(1S)-JOURNAL(1S-1),6004,6006,6006)

C NO. WHICH THREE MACHINE CASE IS THIS?
1002 6040 GO TO (6041,6042,6043,6044,6045,6046,6047),CASE

C PARALLEL OPERATIONS ON INPUT. DIFFERENT OPERATIONS.
1003 6041 QA(I$)=TOTAL(1,IS,1)+TSEQ(I$)+TREU(I$)
1004 6042 OA(I$)=TOTAL(2,IS,1)+TSEQ(I$)+TREU(I$)
1005 6043 OC(I$)=TOTAL(3,IS,1)+TSEQ(I$)+TREU(I$)

C PARALLEL OPERATIONS ON OUTPUT. DIFFERENT OPERATIONS.
1006 6044 QA(I$)=TOTAL(1,IS,1)+TSEQ(I$)+TREU(I$)
1007 OC(I$)=TOTAL(2,IS,1)+TSEQ(I$)+TREU(I$)

C PARALLEL OPERATIONS ON INPUT. SAME OPERATION. SAME MACHINES.
1008 6045 OC(I$)=OC(I$)+TOTAL(3,IS,1)+TSEQ(I$)+TREU(I$)

C PARALLEL OPERATIONS ON OUTPUT. SAME MACHINES.
1009 6046 QA(I$)=TOTAL(1,IS,1)+TSEQ(I$)+TREU(I$)
1010 6047 QA(I$)=TOTAL(2,IS,1)+TSEQ(I$)+TREU(I$)

C PARALLEL OPERATIONS ON INPUT. SAME OPERATION. DIFFERENT MACHINES.
1011 6048 QA(I$)=TOTAL(1,IS,1)+TSEQ(I$)+TREU(I$)
1012 6049 QA(I$)=TOTAL(2,IS,1)+TSEQ(I$)+TREU(I$)

C PARALLEL OPERATIONS ON OUTPUT. SAME OPERATION. DIFFERENT MACHINES.
1013 6044 QA(I$)=TOTAL(1,IS,1)+TSEQ(I$)+TREU(I$)
1014 6045 QA(I$)=TOTAL(2,IS,1)+TSEQ(I$)+TREU(I$)

C PARALLEL OPERATIONS ON INPUT. SAME OPERATION. DIFFERENT MACHINES.
1015 6046 QA(I$)=TOTAL(1,IS,1)+TSEQ(I$)+TREU(I$)
1016 6047 QA(I$)=TOTAL(2,IS,1)+TSEQ(I$)+TREU(I$)

C PARALLEL OPERATIONS ON OUTPUT. SAME OPERATION. DIFFERENT MACHINES.
1017 6048 QA(I$)=TOTAL(1,IS,1)+TSEQ(I$)+TREU(I$)
1018 6049 QA(I$)=TOTAL(2,IS,1)+TSEQ(I$)+TREU(I$)

DATE = 67254 21/30/59 DATE 02/30
ORIRAN IVG LEVEL 0, MOD 0

MAIN DATE = 67/56 21/10/56 PAGE 0056

1020 ODP(LS)=TOTAL(1,IS,1)+TSETUP(1,IS,1)+TWNDUP(1,IS,1)
1021 ODPX(1S)=TOTAL(2,IS,1)+TIME+TSETUP(2,IS,1)+TWNDUP(2,IS,1)
1022 GO TO 6049

C NO PARALLEL OPERATIONS WHICH FOREGROUND AREA IS USED FOR INPUT

1023 6047 IF(FG11)=-916147,6049,6147
1024 6147 IF(FG11)=-1,6049,6049

C FOREGROUND AREA ONE USBD FOR INPUT

1025 OAL(IS)=TOTAL(1,IS,1)+TSETUP(1,IS,1)+TWNDUP(1,IS,1)
1026 OC(IS)=TOTAL(2,IS,1)+TIME+TSETUP(2,IS,1)+TWNDUP(2,IS,1)
1027 GO TO 6149

C FOREGROUND AREA TWO USBD FOR INPUT

1028 OAL(IS)=TOTAL(2,IS,1)+TIME+TSETUP(2,IS,1)+TWNDUP(2,IS,1)
1029 OC(IS)=TOTAL(1,IS,1)+TSETUP(1,IS,1)+TWNDUP(1,IS,1)
1030 6149 OAL(IS)=TOTAL(3,IS,1)+TIME+TSETUP(3,IS,1)+TWNDUP(3,IS,1)

C THIS PROGRAM IS FOR THE SAME JOB AS THE PREVIOUS PROGRAM

1031 6050 NS-LS-I

C WHICH THREE-MACHINE CASE IS THIS

1032 GO TO (6051,6052,6053,6054,6055,6056,6057),CASE

C PARALLEL OPERATIONS ON INPUT DIFFERENT OPERATIONS

1033 QA(NS)=QAM(NS)+TOTAL(IS,1)+TSETUP(IS,1)+TWNDUP(IS,1)
1034 QA(NS)=QAM(NS)+TOTAL(2,IS,1)+TIME+TSETUP(2,IS,1)+TWNDUP(2,IS,1)
1035 QC(NS)=QCN(NS)+TOTAL(3,IS,1)+TIME+TSETUP(3,IS,1)+TWNDUP(3,IS,1)
1036 6070 GO TO 6070

C PARALLEL OPERATIONS ON OUTPUT DIFFERENT OPERATIONS

1037 QA(NS)=QAM(NS)+TOTAL(IS,1)+TIME+TSETUP(IS,1)+TWNDUP(IS,1)
1038 QA(NS)=QAM(NS)+TOTAL(2,IS,1)+TIME+TSETUP(2,IS,1)+TWNDUP(2,IS,1)
1039 QC(IS)=QCN(IS)+TOTAL(3,IS,1)+TIME+TSETUP(3,IS,1)+TWNDUP(3,IS,1)
1040 6070 GO TO 6070

C PARALLEL OPERATIONS ON INPUT SAME OPERATION SAME MACHINES

1041 QC(NS)=QC(IS)+TOTAL(3,IS,1)+TIME+TSETUP(3,IS,1)+TWNDUP(3,IS,1)
1042 6154 GO TO 6154
C PARALLEL OPERATIONS ON OUTPUT, SAME OPERATION, SAME MACHINES

1043 6054 OINS-\(OINS + \text{\textsc{\textit{TOTAL}} (3,15,1)}/\text{\textsc{\textit{TIMEF}}+\text{\textsc{\textit{SETUP}}} (3,15,1)+\text{\textsc{\textit{TWNDUP}}} (2,15,1)}\) 1059
1044 6154 OINS-\(OINS + \text{\textsc{\textit{TOTAL}} (3,15,1)} + \text{\textsc{\textit{SETUP}}} (3,15,1) + \text{\textsc{\textit{TWNDUP}}} (1,15,1)\) 1060
1045 QDINS-\(QDINS + \text{\textsc{\textit{TOTAL}} (2,15,1)}/\text{\textsc{\textit{TIMEF}}+\text{\textsc{\textit{SETUP}}} (2,15,1)+\text{\textsc{\textit{TWNDUP}}} (2,15,1)}\) 1061
1046 GO TO 6070 1062

C PARALLEL OPERATIONS ON INPUT, SAME OPERATION, DIFFERENT MACHINES

1047 6055 QCINS-\(QCINS + \text{\textsc{\textit{TOTAL}} (3,15,1)}/\text{\textsc{\textit{TIMEF}}+\text{\textsc{\textit{SETUP}}} (3,15,1)+\text{\textsc{\textit{TWNDUP}}} (3,15,1)}\) 1063
1049 GO TO 6156 1064

C PARALLEL OPERATIONS ON OUTPUT, SAME OPERATION, DIFFERENT MACHINES

1049 6056 OAINS+OAINS + \text{\textsc{\textit{TOTAL}} (3,15,1)}/\text{\textsc{\textit{TIMEF}}+\text{\textsc{\textit{SETUP}}} (3,15,1)+\text{\textsc{\textit{TWNDUP}}} (3,15,1)}\) 1065
1050 6156 OAINS+OAINS + \text{\textsc{\textit{TOTAL}} (1,15,1)} + \text{\textsc{\textit{SETUP}}} (1,15,1) + \text{\textsc{\textit{TWNDUP}}} (1,15,1)\) 1066
1051 QDINS+QDINS + \text{\textsc{\textit{TOTAL}} (1,15,1)+\text{\textsc{\textit{SETUP}}} (1,15,2)+\text{\textsc{\textit{TWNDUP}}} (1,15,2)\} 1067
1052 QDINS+QDINS + \text{\textsc{\textit{TOTAL}} (2,15,2)} + \text{\textsc{\textit{SETUP}}} (2,15,2)+\text{\textsc{\textit{TWNDUP}}} (2,15,2)\) 1068
1053 QDINS+QDINS + \text{\textsc{\textit{TOTAL}} (2,15,1)} + \text{\textsc{\textit{TWNDUP}}} (2,15,1)\) 1069
1054 GO TO 6070 1070

C NO PARALLEL OPERATIONS, WHICH FOREGROUND AREA IS USED FOR INPUT

1055 6057 IF (EG (11) 15, 6057) 1071
1056 6157 IF (EG (11) 15, 6057) 1072

C FOREGROUND AREA ONE USED FOR INPUT

1057 6058 OAINS+OAINS + \text{\textsc{\textit{TOTAL}} (1,15,1) + \text{\textsc{\textit{SETUP}}} (1,15,1)+\text{\textsc{\textit{TWNDUP}}} (1,15,1)\) 1073
1059 QCINS+QCINS + \text{\textsc{\textit{TOTAL}} (2,15,1) + \text{\textsc{\textit{TIMEF}}}+\text{\textsc{\textit{SETUP}}} (2,15,1)+\text{\textsc{\textit{TWNDUP}}} (2,15,1)\) 1074
1059 GO TO 6159 1075

C FOREGROUND AREA TWO USED FOR INPUT

1060 6059 QAINS+QAINS + \text{\textsc{\textit{TOTAL}} (2,15,1) + \text{\textsc{\textit{TIMEF}}}+\text{\textsc{\textit{SETUP}}} (2,15,1)+\text{\textsc{\textit{TWNDUP}}} (2,15,1)\) 1076
1061 QCINS+QCINS + \text{\textsc{\textit{TOTAL}} (1,15,1) + \text{\textsc{\textit{TIMEF}}}+\text{\textsc{\textit{SETUP}}} (1,15,1)+\text{\textsc{\textit{TWNDUP}}} (1,15,1)\) 1077
1062 QAINS+QAINS + \text{\textsc{\textit{TOTAL}} (3,15,1) + \text{\textsc{\textit{TIMEF}}}+\text{\textsc{\textit{SETUP}}} (3,15,1)+\text{\textsc{\textit{TWNDUP}}} (3,15,1)\) 1078
1063 GO TO 6070 1079
1064 6069 ISCHEMS (2) = 15 1080
1065 LS=LS+1 1081

C ARE THERE ANY MORE FULL WORKING DAY PROGRAMS

1066 6070 CONTINUE 1082
1067 LS=LS-1 1083

C NO. WERE THERE ANY NON-CONFLICT FULL WORKING DAY BACKGROUND PROGRAMS?
I \[TILS-19999,6173,9774\]

C YES, FIND OPTIMAL SEQUENCE

1068 DO 73 CALL 'GNSCH (OA;Q;OB;QD;ODP;ODQ;OL;Q;LE;CASE;ERROR;TRPDB) MOD 82705

C COULD THE ALGORITHM FIND AN OPTIMAL SEQUENCE

1070 IFERRORX=116074,9999,9999

C YES, SAVE SEQUENCING INFORMATION

1071 MOD 62000

1072 MOD 61000

1073 MOD 6074=30 6076 15=1,LS

1074 MOD 6076=INT(15)

1075 ISCHED1=ISCHED(JS;2)

1076 KS=ISCHED(JS;2)

1077 NS=INT(15)

C WHICH THREE MACHINE CASE IS THIS

1078 GO TO 1699, 6082, 6093, 6046, 6045, 6047, CASE

C PARALLEL OPERATIONS ON INPUT. DIFFERENT OPERATIONS

1079 MOD 6081 SCHED1;J;S, 2=OA1(S)

1080 SCHED2;J;S, 2=OA1(S)

1081 SCHED3;J;S, 2=OA1(S)

1082 GO TO 6078

C PARALLEL OPERATIONS ON OUTPUT. DIFFERENT OPERATIONS

1083 MOD 6082 SCHED3;J;S, 2=OA1(S)

1084 SCHED1;J;S, 2=OA1(S)

1085 SCHED2;J;S, 2=OA1(S)

1086 GO TO 6078

C PARALLEL OPERATIONS ON INPUT. SAME OPERATIONS, SAME MACHINES

1087 MOD 6083 SCHED3;J;S, 2=OA1(S)

1088 GO TO (6094, 6194), NS

C PARALLEL OPERATIONS ON OUTPUT. SAME OPERATIONS, SAME MACHINES

1089 MOD 6084 SCHED3;J;S, 2=OA1(S)

1090 GO TO (6094, 6194), NS
C PARALLEL OPERATIONS ON INPUT, SAME OPERATION, DIFFERENT MACHINES

1094 SCHED(1,KS,2) = DO(1,15)
1095 GO TO 6079
1096 SCHED(2,KS,2) = DO(1,15)
1097 GO TO 6079

C PARALLEL OPERATIONS ON OUTPUT, SAME OPERATION, DIFFERENT MACHINES

1098 SCHED(1,KS,2) = DO(1,15)
1099 GO TO 6096, 6106, 6116
1100 SCHED(2,KS,2) = DO(1,15)
1101 GO TO 6079
1102 SCHED(2,KS,2) = DO(1,15)
1103 GO TO 6079
1104 SCHED(2,KS,2) = DO(1,15)
1105 GO TO 6079
1106 SCHED(2,KS,2) = DO(1,15)
1107 SCHED(2,KS,2) = DO(1,15)
1108 SCHED(2,KS,2) = DO(1,15)
1109 SCHED(2,KS,2) = DO(1,15)
1110 SCHED(2,KS,2) = DO(1,15)
1111 SCHED(2,KS,2) = DO(1,15)
1112 SCHED(2,KS,2) = DO(1,15)
1113 SCHED(2,KS,2) = DO(1,15)
1114 SCHED(2,KS,2) = DO(1,15)
1115 SCHED(2,KS,2) = DO(1,15)
1116 SCHED(2,KS,2) = DO(1,15)
1117 SCHED(2,KS,2) = DO(1,15)
1118 SCHED(2,KS,2) = DO(1,15)
1119 SCHED(2,KS,2) = DO(1,15)
1120 SCHED(2,KS,2) = DO(1,15)
1121 SCHED(2,KS,2) = DO(1,15)
1122 SCHED(2,KS,2) = DO(1,15)
1123 SCHED(2,KS,2) = DO(1,15)
C CASE(IXS)=CASE(IXS)+JS
6175 CASE(IXS)=CASE(IXS)+JS
6175 CONTINUE
6075 CONTINUE
6076 CONTINUE
GO TO 6200

C TWO-MACHINE SEQUENCING PROBLEM, WHICH FOREGROUND AREA IS BEING USED?
C
6004 IF(FC11)=BCMK6111,6111,6111
C FOREGROUND AREA ONE IS BEING USED
C
6111 NS=1
GO TO 6000
C FOREGROUND AREA TWO IS BEING USED
C
6112 NS=2
6009 DO 6090 IS=TSIJ,MXJOIN
C IS THIS A NON-CONFLICT BACKGROUND PROGRAM?
C
6133 IF(JCASE(1S1)=216111,6090,6090)
C YES; CALCULATE MULTIPROGRAMMING TIME
C
6134 6114 DO 6114 JS=1,12
6114 DO 6114 KS=1,7
6114 6090 IS=1,7,12
6090 NS=1,7,12
6114 BFRAC(JS,KS)+FRAC(IS,JS,KS)
6090 BFRAC(JS,KS)+FRAC(IS,JS,KS)
6114 BFRAC(JS,KS)+FRAC(IS,JS,KS)
GO TO 6115,6116,NS
6115 CALL TIME23(JRCAF1,Fracfi,FracCAN,FRACX,FRACX,FRACX,FRACX,FRACX,FRACX,
FRACX,FRACX,TIMEI,TIMEI,TIMEI)
6116 CALL TIME23(JRCAF1,Fracfi,FracCAN,FRACX,FRACX,FRACX,FRACX,FRACX,FRACX,
FRACX,FRACX,FRACX,FRACX,TIMEI,TIMEI,TIMEI)
6117 SCHED(1,IS,11)=TIMEX
6117 SCHED(1,IS,11)=TIMEX
6117 SCHED(1,IS,11)=TIMEX
6117 SCHED(1,IS,11)=TIMEX
6120 IS=1,11
1615 IS=1,11
1616 IS=1,11
1617 IS=1,11
1618 IS=1,11

C IS THIS PROGRAM FOR THE SAME JOB AS THE PREVIOUS PROGRAM
C
6118 JF(JROUND(IS)=JROUND(1S1)=116111,6119,6119
6119 1P1=CASE(1S1)-7
C YES, WHICH TWO MACHINE CASE IS THIS
I             C        F
1.             C   L
I             C   I
I             C   U X
I             C   V

C   PROGRAM USED FOR INPUT
C
C   OAKLS=TOTAL(IS,15,1)+TSETUP(IS,15,1)+TNDUP(IS,15,1)

C   GO TO 6129

C   PROGRAM USED FOR OUTPUT
C
C   OAKLS=TOTAL(IS,15,1)+TSETUP(IS,15,1)+TNDUP(IS,15,1)

C   GO TO 6129

C   THIS PROGRAM IS FOR THE SAME JOB AS THE PREVIOUS PROGRAM
C
C   6123   TDI=CASE(IS)-7

C   KS=LS+1

C   WHICH MACHINE CASE IS THIS
C
C   GO TO (6124,6125),TDI

C   PROGRAM USED FOR INPUT
C
C   OAKKS=OAK(KS)+TOTAL(IS,15,1)+TSETUP(IS,15,1)+TNDUP(IS,15,1)

C   GO TO 6090

C   PROGRAM USED FOR OUTPUT
C
C   OAKKS=OAK(KS)+TOTAL(IS,15,1)+TSETUP(IS,15,1)+TNDUP(IS,15,1)

C   GO TO 6090

C   ARE THERE ANY MORE FULL WORKING DAY PROGRAMS
C
C   6090 CONTINUE

C   LS=LS+1

C   NO, WERE THERE ANY NON-CONFLICT FULL WORKING DAY BACKGROUND PROGRAMS
C
C   IF(LS-119999,6127,6125

C   YES, FIND OPTIMAL SEQUENCE
C
A126 CALL JOHNSTON(0,0,ON,1,1)
GO TO 6128
C SAVE SEQUENCING INFORMATION
C
6127 ON(1)=1
6128 ON 4160 IS=1,IS
JS=QA(1S)
TSCHED(IS,1):TSCHED(JS,2)
KS=TSCHED(JS,2)
IP1=JCEAS(IS),
C WHICH TWO MACHINE CASE IS THIS
C
GO TO (6161,6164),IP1
6161 GO TO (6147,6161),NS
C FOREGROUND AREA ONE USED FOR INPUT
C
6162 SCHED(1,KS,2)=0A(IS)
SCHED(2,KS,2)=0
GO TO 6066
C FOREGROUND AREA TWO USED FOR INPUT
C
6163 SCHED(1,KS,2)=0.
SCHED(2,KS,2)=0A(IS)
6063 SCHED(3,KS,2)=0A(IS)
GO TO 6067
6164 GO TO (6165,6166),NS
C FOREGROUND AREA ONE USED FOR OUTPUT
C
6165 SCHED(1,KS,2)=0A(IS)
SCHED(2,KS,2)=0.
GO TO 6066
C FOREGROUND AREA TWO USED FOR OUTPUT
C
6166 SCHED(1,KS,2)=0.
SCHED(2,KS,2)=0A(IS)
6066 SCHED(3,KS,2)=0A(IS)
6067 JCEAS(KS)=0
GO 6169 JS=1,IS
SCHED(1,KS,2)=0
6194 JS=1,JS
6195 IF SCHED(IS,KS,2)1,6169,6169,6167
6196 JCASE(KS)=JCASE(KS)+JS
6197 6169 CONTINUE
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1198</td>
<td>6160 CONTINUE</td>
</tr>
<tr>
<td>1199</td>
<td>GO TO 6200</td>
</tr>
<tr>
<td>1200</td>
<td>C NO SEQUENCING PROBLEM</td>
</tr>
<tr>
<td>1201</td>
<td>6010 IF 6110 TS=STJ, MXJOB</td>
</tr>
<tr>
<td>1202</td>
<td>IP=JCASE(IS)</td>
</tr>
<tr>
<td>1203</td>
<td>GO TO (6011,6011,6110,6011,10)</td>
</tr>
<tr>
<td>1204</td>
<td>6011 ISCHED(IS),1)=1S</td>
</tr>
<tr>
<td>1205</td>
<td>SCHED(3,IS,1)+TOTAL(3,IS,1)+TSETUP(3,IS,1)+TWDUP(3,IS,1)</td>
</tr>
<tr>
<td>1206</td>
<td>SCHED(2,IS,2)=0</td>
</tr>
<tr>
<td>1207</td>
<td>SCHED(1,IS,3)=0</td>
</tr>
<tr>
<td>1208</td>
<td>DO 6210 KS=1</td>
</tr>
<tr>
<td>1209</td>
<td>6019 IF SCHED(KS,IS,3)=1</td>
</tr>
<tr>
<td>1210</td>
<td>JCASE(IS)=3</td>
</tr>
<tr>
<td>1211</td>
<td>LS=LS+1</td>
</tr>
<tr>
<td>1212</td>
<td>6110 CONTINUE</td>
</tr>
<tr>
<td>1213</td>
<td>GO TO 6210</td>
</tr>
<tr>
<td>1214</td>
<td>C ADJUST SEQUENCE FOR MULTISTEP JOBS</td>
</tr>
<tr>
<td>1215</td>
<td>6200 IF MXJOB</td>
</tr>
<tr>
<td>1216</td>
<td>DO 6209 IS=IS JMXJOB</td>
</tr>
<tr>
<td>1217</td>
<td>C ARE THERE ANY MORE PROGRAMS SEQUENCED</td>
</tr>
<tr>
<td>1218</td>
<td>IF (SCHED(IS),1)=116210,6201,6201</td>
</tr>
<tr>
<td>1219</td>
<td>6201 JS=SCHED(IS),11</td>
</tr>
<tr>
<td>1220</td>
<td>C YES, IS THIS PROGRAM FOR THE SAME JOB AS THE NEXT PROGRAM</td>
</tr>
<tr>
<td>1221</td>
<td>IF (JOBNAME(ISJ)=JOBNAM(ISJ+1))6209,6202,6209</td>
</tr>
<tr>
<td>1222</td>
<td>6202 ISJ=ISJ+1</td>
</tr>
<tr>
<td>1223</td>
<td>C YES, COMPUTE INDIVIDUAL PROGRAM MULTIPROGRAMMING TIMES</td>
</tr>
<tr>
<td>1224</td>
<td>DO 6208 KS=JS, MXJOB</td>
</tr>
<tr>
<td>1225</td>
<td>MS=SCHED(3,KS,3)</td>
</tr>
<tr>
<td>1226</td>
<td>SCHED(3,KS,2)+TOTAL(3,KS,M5)+SCHED(2,KS,1)+TSETUP(3,KS,M5)+</td>
</tr>
<tr>
<td>1227</td>
<td>TWDUP(3,KS,M5)</td>
</tr>
<tr>
<td>1228</td>
<td>IF (JOBNAME(KSJ)=JOBNAM(KSJ+1))6207,6203,6207</td>
</tr>
<tr>
<td>1229</td>
<td>6203 NS=IS+2</td>
</tr>
<tr>
<td>1230</td>
<td>NS=IS+1</td>
</tr>
<tr>
<td>1231</td>
<td>DO 6204 IP=MS, MXJOB</td>
</tr>
<tr>
<td>1232</td>
<td>6204 ISCHED(MXJOB,IP1+MS,1)=ISCHED(MXJOB,IP1+MS,1)</td>
</tr>
<tr>
<td>1233</td>
<td>ISCHED(IS+1,1)=JS+1</td>
</tr>
<tr>
<td>1234</td>
<td>ISJ=IS+1</td>
</tr>
</tbody>
</table>
GO TO 0000
6229 SCHED(3,JS,2)=SCHED(3,JS,21)-SCHED(3,JS,11)
SCHED(3,JS,11)=0
SCHED(3,JS,11)=2
MS=15+1
DO 6228 JS=MS,LS
SCHED(3,JS,11)=0
SCHED(3,JS,11)=0
SCHED(3,JS,1)=0
GO TO 0000

C FOREGROUND AREAS USED FOR OUTPUT
C
6243 IS=1
JS=ISCHED(15,11)
SCHED(3,JS,1)=BEGIN
SCHED(3,JS,21)=BEGIN+SCHED(3,JS,2)
IF(SCHED(3,JS,21)-FND16241,6241,6243)
DO 6242 IS=2,15
JS=SCHED(15,11)
KS=ISCHED(15-1,11)
SCHED(3,JS,11)=SCHED(3,JS,21)+SCHED(3,JS,21)
IF(SCHED(3,JS,21)-FND16247,6247,6243)
SCHED(3,JS,1)=2
GO TO 6245
6243 SCHED(3,JS,21)=SCHED(3,JS,21)-SCHED(3,JS,11)
SCHED(3,JS,1)=0
SCHED(3,JS,1)=2
MS=15+1
DO 6244 IS=MS,15
SCHED(3,JS,1)=0
DO 6244 SCHED(3,JS,1)=2
DO 6245 DO 6239 NS=1,7
IS=1
JS=SCHED(15,11)
SCHED(NS,JS,1)=SCHED(3,JS,21)
SCHED(NS,JS,2)=SCHED(NS,JS,1)+SCHED(NS,JS,21)
IF(SCHED(NS,JS,21)-FND16244,6244,6243)
6246 SCHED(NS,JS,4)=1
DO 6247 IS=2,15
JS=SCHED(15,11)
KS=ISCHED(15-1,11)
SCHED(NS,JS,1)=AMAX1(SCHED(3,JS,21),SCHED(NS,JS,21)
SCHED(NS,JS,2)=SCHED(NS,JS,1)+SCHED(NS,JS,21)
IF(SCHED(NS,JS,21)-FND16247,6247,6243)
SCHED(NS,JS,4)=1
GO TO 0000
FORTRAN IV G LEVEL 0, MOD 0

1317 6248 SCHEDINS,JS,21=SCHEDINS,JS,21-SCHEDINS,JS,11
1318 SCHEDINS,JS,11=0
1319 SCHEDINS,JS,41=2
1320 NS=JS+1
1321 GO 6240 NS=MS,LS
1322 JS=ISCHED(15,1)
1323 SCHEDINS,JS,11=0
1324 6249 SCHEDINS,JS,41=2
1325 6239 CONTINUE
1326 GO TO 9000
1327 C
1328 C FOREGROUND AREAS NOT USED
1329 C
1330 6230 DO 6235 NS=1,3
1331 JS=1
1332 JS=ISCHED(15,1)
1333 SCHEDINS,JS,11=BEGIN
1334 SCHEDINS,JS,21=BEGIN+SCHEDINS,JS,21
1335 IF SCHEDINS,JS,21=END16231,6231,6233: 6233
1336 SCHEDINS,JS,41=1
1337 DO 6232 IS=1,5
1338 JS=ISCHED(15,1)
1339 SCHEDINS,JS,11=SCHEDINS,KS,71
1340 SCHEDINS,JS,21=SCHEDINS,JS,11+SCHEDINS,JS,21
1341 IF SCHEDINS,JS,21=END16232,6232,6233: 6233
1342 SCHEDINS,JS,41=1
1343 GO TO 6235
1344 6233 SCHEDINS,JS,21=SCHEDINS,JS,21-SCHEDINS,JS,11
1345 SCHEDINS,JS,11=0
1346 SCHEDINS,JS,41=2
1347 "S=JS+1
1348 GO 6236 IS=MS,LS
1349 JS=ISCHED(15,1)
1350 SCHEDINS,JS,11=0
1351 6234 SCHEDINS,JS,41=2
1352 6235 CONTINUE
1353 GO TO 9000
1354 C
1355 C FOREGROUND AREAS USED FOR BOTH INPUT AND OUTPUT. WHICH AREA IS USED FOR INPUT
1356 C
1357 6250 IF (FGE11)=16236,6237,6236
1358 6236 IF (FGE11)=16238,6237,6238
1359 C
1360 C FOREGROUND AREA ONE USED FOR INPUT
1361 C
1362 6237 NS=1
1363 GO TO 6255
1364 C
1365 C
C FOREGROUND AREA TWO USED FOR INPUT

6238 NS=2
6255 IS=1
6256 JS=1
SCHED(IS,JS,1)=BEGIN
SCHED(IS,JS,2)=BEGIN+SCHED(IS,JS,2)
IF(SCHED(IS,JS,2)=END)+IS=1,6251,6256,6252,6253
6251 SCHED(IS,JS,4)=1
DO 6252 IS=2,JS
JS=IS=SCHED(IS,JS,1)
KS=SCHED(IS,JS,1)+SCHED(IS,JS,2)
IF(SCHED(IS,JS,2)=END)+IS=1,6252,6252,6253
6252 SCHED(IS,JS,4)=1
GO TO 6256
6253 SCHED(IS,JS,2)=SCHED(IS,JS,2)+SCHED(IS,JS,1)
SCHED(IS,JS,1)=0
SCHED(IS,JS,4)=1
MS=IS+1
GO 6254 IS=MS,JS
SCHED(IS,JS,1)=0
SCHED(IS,JS,4)=2
SCHED(IS,JS,2)=3
GO 6250 NN=1,1,3,NS
GO TO 6261,6261,6262,6262,NN
6261 IK=NS
GO 6233
6262 IK=3-NS
6263 IS=1
JS=IS=SCHED(IS,1)
SCHED(IS,JS,1)+SCHED(IS,JS,2)
SCHED(IS,JS,2)=SCHED(IS,JS,1)+SCHED(IS,JS,2)
IF(SCHED(IS,JS,2)=END)+IS=1,6257,6257,6257
6257 DO 6258 IS=2,JS
JS=IS=SCHED(IS,1)
KS=SCHED(IS,1)+SCHED(IS,JS,2)
SCHED(IS,JS,1)=4*X(I,SCHED(IS,JS,2)),SCHED(IS,KS,2)
SCHED(IS,JS,2)=END)+IS=1,6258,6258,6258
6258 SCHED(IS,JS,4)=1
GO TO 6000
6259 SCHED(IS,JS,2)=SCHED(IS,JS,2)+SCHED(IS,JS,1)
SCHED(IS,JS,1)=0
SCHED(IS,JS,4)=2
MS=IS+1
GO 6264 IS=MS,JS
1567  CALL EXIT
1568  WRITE (6,2000) C,D
1570  GO TO 2331
1571  FORMAT (6,4,2330)
1572  2330 FORMAT (20H CPU MODEL NUMBER IN COLUMNS 7-8 MUST BE '2065', '2075')
1573  GO TO 2002
1574  1069 CALL ERRPT (ERROR)
1575  WRITE (6,2001) 4,01
1576  2005 IFIXIT=DEVSPD(11)
1577  JFIXIT=DEVSPT(11)
1578  KFIXIT=10*(DEVSP(11)+.05)
1579  LFIXIT=DEVLSS(11)
1580  MFIXIT=DEVSUP(11)
1581  NFIXIT=DEVEK(11)
1582  OFIXIT=10*(DEVINT(11)+.05)
1583  WRITE (6,2005) A1,A2,DEVTP(11),DEVMD(11),DEVCLS(11),TFIXIT,EFIXIT
1584  3005 FORMAT (H,4A1,14A2,A1,13A14,17,14,17,14)
1585  GO TO 4
1586  1006 CALL ERRPT (ERROR)
1587  WRITE (6,2000) 0,02
1588  1007 CALL ERRPT (ERROR)
1589  WRITE (6,2009)
1590  2009 FORMAT (4H DEVICE CLASS IN COLUMN 9 MUST BE D, T, OR U)
1591  GO TO 2005
1592  2010 CALL ERRPT (ERROR)
1593  WRITE (6,2010)
1594  2020 FORMAT (5H INFEERENCE TIME IN COLUMNS 25-27 MUST BE NON-NEGATIVE)
1595  GO TO 2005
1596  2012 CALL ERRPT (ERROR)
1597  WRITE (6,2000) 4,01
1598  2012 IFIXIT=DEVSPD(11)
1599  JFIXIT=DEVSPT(11)
1600  KFIXIT=10*(DEVSP(11)+.05)
1601  LFIXIT=DEVLSS(11)
1602  MFIXIT=DEVSUP(11)
1603  NFIXIT=DEVEK(11)
1604  OFIXIT=10*(DEVINT(11)+.05)
1605  WRITE (6,2005) A1,A2,DEVTP(11),DEVMD(11),DEVCLS(11),TFIXIT,EFIXIT
1606  JFIXIT,KFIXIT,IFIXIT,MFIXIT,OFIXIT
1607  [DE1]
1668  GO TO 12
1669  1014 CALL ERRPT (ERROR)
1670  WRITE (6,20011) N,7,92
1671  GO TO 2012
1672  1017 CALL ERRPT (ERROR)
1673  WRITE (6,20069)
1674  GO TO 2012
1675  1019 CALL ERRPT (ERROR)
1676  WRITE (6,2010)
1677  GO TO 2012
1678  1020 CALL ERRPT (ERROR)
1679  WRITE (6,2020) DEVYYP(T1),DEVMOD(T1)
1680  2020 FORMAT (1H DUPLICATE DEVICE TYPE AND MODEL NUMBER ,41,1X,A2)
1681  GO TO 2012
1682  1211 CALL ERRPT (ERROR)
1683  WRITE (6,2000) 4,01
1684  2211 WRITE (6,3211) A1,A2
1685  3211 FORMAT (1H A1)
1686  GO TO 210
1687  1212 CALL ERRPT (ERROR)
1688  WRITE (6,2000) 7,02
1689  GO TO 2212
1690  1024 CALL ERRPT (ERROR)
1691  WRITE (6,2000) 1,01
1692  2024 WRITE (6,1023) A1,A2,A3,A4,M1,IPJUH(N),F=1,231,A5
1693  3024 FORMAT (1H,2A1,4A,12,23A3,A1)
1694  N=0
1695  K=11-1
1696  GO TO 303
1697  1240 CALL ERRPT (ERROR)
1698  WRITE (6,20001) N,02
1699  GO TO 2204
1700  1241 CALL ERRPT (ERROR)
1701  IP1=11-1
1702  WRITE (6,2241) N1,IP1,M1
1703  2241 FORMAT (1H NUMBER OF UNITS IN STATEMENT (/12,43H) PLUS NUMBER OF UNITS PREVIOUSLY DEFINED (/12,23H) EXCEEDS PROGRAM MAXIMUM (/12,11H)0106500)
1704  GO TO 2024
1705  1025 CALL ERRPT (ERROR)
1706  WRITE (6,2025) N1
1707  2025 FORMAT (1H NUMBER OF UNITS (/12,39H) IN COLUMN 0-10 EXCEEDS MAXIMUM (/12,39H) SUM OF 21)
1708  GO TO 2024
1709  1026 CALL ERRPT (ERROR)
1710  WRITE (6,2026)
1711  2026 FORMAT (1H NUMBER OF UNITS IS LESS THAN 1)
1712  GO TO 2024
<table>
<thead>
<tr>
<th>FORTRAN IV G LEVEL 0, 800 0</th>
<th>MAIN</th>
<th>DATE = A7256</th>
<th>21/20/56</th>
<th>PAGE 000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1653 1028 CALL ERRPRIT (ERROR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1654 WRITE (13,2026) A3,A4</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1655 2029 FORMTION DEVICE TYPE AND MODEL NUMBER A5,1X,12,1X,3 NOT OK DEFINE</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1656 &quot;BY AN HD STATEMENT.&quot;</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1657 GO TO 2024</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1658 1029 CALL ERRPRIT (ERROR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1659 WRITE (6,2027) C04</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1660 &quot;2029 FORMAT(2EH DUPLICATE UNIT NUMBER A5)&quot;</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1661 GO TO 2024</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1662 1300 CALL ERRPRIT (ERROR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1663 PRINT 2300,C04</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1664 2300 FORMAT(2EH INVALID CHANNEL NUMBER A5)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1665 GO TO 2024</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1666 1033 CALL ERRPRIT (ERROR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1667 PRINT 2000,1,01</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1668 2033 IFIXIT=FINSTK(IFG)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1669 JFIXIT=FLKFC(IFG)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1670 KFIXIT=FMTFCY(IFG)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1671 LFIXIT=FSPPK(IFG)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1672 WRITE (6,3033) A1,A2,A5,A6,A7,A8,A9,A10,TG,17,2,131</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1673 12=IFG</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1674 GO TO 129</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1675 1034 CALL ERRPRIT (ERROR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1676 WRITE (6,2000) F,02</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1677 GO TO 2033</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1678 1035 CALL ERRPRIT (ERROR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1679 GO TO (2034,2056,1,IFG)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1680 WRITE (6,2000) D1,D3</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1681 GO TO 2033</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1682 2034 WRITE (6,2000) D1,D3</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1683 GO TO 2033</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1684 1037 CALL ERRPRIT (ERROR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1685 WRITE (6,2037)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1686 2037 FORMAT(35H PROGRAM EXPECTED BLANK IN COLUMN 5)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1687 2038 IFIXIT=FINSTK(IFG)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1688 JFIXIT=FLKFC(IFG)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1689 KFIXIT=FMTFCY(IFG)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1690 LFIXIT=FSPPK(IFG)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1691 WRITE (6,3033) A1,A2,A5,A6,A7,A8,A9,A10,TG,17,2,131</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1692 12=IFG</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1693 GO TO (2039,680,IFG)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1694 2039 I2=2</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1695 GO TO 320</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1696 1044 CALL ERRPRIT (ERROR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1697 WRITE (6,2044)</td>
<td>M0179700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1698</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1643 2044 FORMAT(47H PROGRAM EXPECTED P, 9, T, OR BLANK IN COLUMN 4) M0111900
1649 GO TO 2038
1700 1046 CALL ERRPT (ERROR)
1701 WRITE (6,2001) IA,IB,IC
1702 GO TO 2038
1703 1049 CALL ERRPT (ERROR)
1704 WRITE (6,2049)
1705 2049 FORMAT(40H NO FOREGROUND PROGRAM NAME IN COLUMNS 5-13)
1706 GO TO 2038
1707 1050 CALL ERRPT (ERROR)
1708 WRITE (6,2050)
1709 2050 FORMAT(40H DEVICE CLASS OF UNIT IN COLUMNS 14-15 MUST BE 1)
1710 GO TO 2038
1711 1051 CALL ERRPT (ERROR)
1712 WRITE (6,2051) FGCU1(IFG)
1713 2051 FORMAT(2H UNIT NUMBER 'AS, 3TH NOT DEFINED BY AN IN STATEMENT)
1714 GO TO 2038
1715 1054 CALL ERRPT (ERROR)
1716 WRITE (6,2054)
1717 2054 FORMAT(40H DEVICE CLASS OF UNIT IN COLUMNS 17-18 MUST BE 3 OR 11)
1718 GO TO 2038
1719 1055 CALL ERRPT (ERROR)
1720 WRITE (6,2055) FGCU1(IFG)
1721 GO TO 2034
1722 1056 CALL ERRPT (ERROR)
1723 WRITE (6,2056)
1724 2056 FORMAT(40H LATENCY TIME IN COLUMNS 25-27 MUST BE NON-NEGATIVE)
1725 GO TO 2038
1726 1058 CALL ERRPT (ERROR)
1727 WRITE (6,2058)
1728 2058 FORMAT(40H SEEK TIME IN COLUMNS 28-30 MUST BE NON-NEGATIVE)
1729 GO TO 2038
1730 1068 CALL ERRPT (ERROR)
1731 WRITE (6,2068) IA
1732 2066 FORMAT(40H BOTH FOREGROUND PROGRAMS ASSIGNED UNIT 'A')
1733 FG(1)=BLANK4
1734 FG(2)=BLANK4
1735 GO TO 6A0
1736 1074 CALL ERRPT (ERROR)
1737 WRITE (6,2074) IA
1738 2074 FORMAT(40H MON(1), MON(2), HHDF(1), HHDF(2), TH=1, 2)
1739 TH=1,2
1740 3074 FORMAT(2H AI, A1, 7(4I2))
1741 GO TO 730
1742 1075 CALL ERRPT (ERROR)
1743 WRITE (6,2075) IA, IB
1744 GO TO 2074
1745 1750 CALL ERRPT (ERROR)
<table>
<thead>
<tr>
<th>FORTRAN IV G LEVEL</th>
<th>MOD</th>
<th>MAIN</th>
<th>DATE = 6/27/70</th>
<th>PAGE 0700</th>
</tr>
</thead>
<tbody>
<tr>
<td>1745</td>
<td></td>
<td>WRITE (6,2750)</td>
<td></td>
<td>01176800</td>
</tr>
<tr>
<td>1746</td>
<td>2750</td>
<td>F0MAT(14H NUMBER OF DAYS IN WEEK IS GREATER THAN 7)</td>
<td>2130/55</td>
<td>01176900</td>
</tr>
<tr>
<td>1747</td>
<td></td>
<td>WEEK=7</td>
<td></td>
<td>01177000</td>
</tr>
<tr>
<td>1748</td>
<td>2752</td>
<td>WRITE (6,3074)</td>
<td>AL,2,3,AT(HMONIT11),MMONIT11,HHOFT11,MMOFF11,</td>
<td>01177100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>I1J=1,7)</td>
<td></td>
<td>01177200</td>
</tr>
<tr>
<td>1749</td>
<td></td>
<td>GO TO 76</td>
<td></td>
<td>01177300</td>
</tr>
<tr>
<td>1750</td>
<td>1751</td>
<td>CALL ERRPT (ERROR)</td>
<td></td>
<td>01177400</td>
</tr>
<tr>
<td>1751</td>
<td></td>
<td>WRITE (6,2751)</td>
<td></td>
<td>01177500</td>
</tr>
<tr>
<td>1752</td>
<td>2751</td>
<td>F0MAT(14H NUMBER OF DAYS IN WEEK IS LESS THAN 1)</td>
<td>2130/55</td>
<td>01177600</td>
</tr>
<tr>
<td>1753</td>
<td></td>
<td>WEEK=1</td>
<td></td>
<td>01177700</td>
</tr>
<tr>
<td>1754</td>
<td></td>
<td>GO TO 2752</td>
<td></td>
<td>01177800</td>
</tr>
<tr>
<td>1755</td>
<td>1760</td>
<td>CALL ERRPT (ERROR)</td>
<td></td>
<td>01177900</td>
</tr>
<tr>
<td>1756</td>
<td></td>
<td>WRITE (6,2766)</td>
<td>11</td>
<td>01178000</td>
</tr>
<tr>
<td>1757</td>
<td>2763</td>
<td>F0MAT(14H HOUR ON FOR DAY 11,15H IS GREATER THAN 23)</td>
<td>2130/55</td>
<td>01178100</td>
</tr>
<tr>
<td>1758</td>
<td></td>
<td>HMONIT11=23</td>
<td></td>
<td>01178200</td>
</tr>
<tr>
<td>1759</td>
<td>3760</td>
<td>WRITE (6,3074)</td>
<td>AL,2,3,1T(HMONIT11),MMONIT11,HHOFT11,MMOFF11,</td>
<td>01178300</td>
</tr>
<tr>
<td></td>
<td></td>
<td>I1J=1,7)</td>
<td></td>
<td>01178400</td>
</tr>
<tr>
<td>1760</td>
<td></td>
<td>GO TO 77</td>
<td></td>
<td>01178500</td>
</tr>
<tr>
<td>1761</td>
<td>1770</td>
<td>CALL ERRPT (ERROR)</td>
<td></td>
<td>01178600</td>
</tr>
<tr>
<td>1762</td>
<td></td>
<td>WRITE (6,2770)</td>
<td>11</td>
<td>01178700</td>
</tr>
<tr>
<td>1763</td>
<td>2770</td>
<td>F0MAT(14H HOUR ON FOR DAY 11,15H IS LESS THAN 0)</td>
<td>2130/55</td>
<td>01178800</td>
</tr>
<tr>
<td>1764</td>
<td></td>
<td>HMONIT11=0</td>
<td></td>
<td>01178900</td>
</tr>
<tr>
<td>1765</td>
<td></td>
<td>GO TO 3760</td>
<td></td>
<td>01179000</td>
</tr>
<tr>
<td>1766</td>
<td>1777</td>
<td>CALL ERRPT (ERROR)</td>
<td></td>
<td>01179100</td>
</tr>
<tr>
<td>1767</td>
<td></td>
<td>WRITE (6,2077)</td>
<td>11</td>
<td>01179200</td>
</tr>
<tr>
<td>1768</td>
<td>2077</td>
<td>F0MAT(14H HOUR OFF FOR DAY 11,15H IS GREATER THAN 23)</td>
<td>2130/55</td>
<td>01179300</td>
</tr>
<tr>
<td>1769</td>
<td></td>
<td>HHOFT11=23</td>
<td></td>
<td>01179400</td>
</tr>
<tr>
<td>1770</td>
<td>3077</td>
<td>WRITE (6,3074)</td>
<td>AL,2,3,1T(HMONIT11),MMONIT11,HHOFT11,MMOFF11,</td>
<td>01179500</td>
</tr>
<tr>
<td></td>
<td></td>
<td>I1J=1,7)</td>
<td></td>
<td>01179600</td>
</tr>
<tr>
<td>1771</td>
<td></td>
<td>GO TO 78</td>
<td></td>
<td>01179700</td>
</tr>
<tr>
<td>1772</td>
<td>1780</td>
<td>CALL ERRPT (ERROR)</td>
<td></td>
<td>01179800</td>
</tr>
<tr>
<td>1773</td>
<td></td>
<td>WRITE (6,2780)</td>
<td>11</td>
<td>01179900</td>
</tr>
<tr>
<td>1774</td>
<td>2780</td>
<td>F0MAT(14H HOUR OFF FOR DAY 11,15H IS LESS THAN 0)</td>
<td>2130/55</td>
<td>01180000</td>
</tr>
<tr>
<td>1775</td>
<td></td>
<td>HHOFT11=0</td>
<td></td>
<td>01180100</td>
</tr>
<tr>
<td>1776</td>
<td></td>
<td>GO TO 3780</td>
<td></td>
<td>01180200</td>
</tr>
<tr>
<td>1777</td>
<td>1787</td>
<td>CALL ERRPT (ERROR)</td>
<td></td>
<td>01180300</td>
</tr>
<tr>
<td>1778</td>
<td></td>
<td>WRITE (6,2078)</td>
<td>11</td>
<td>01180400</td>
</tr>
<tr>
<td>1779</td>
<td>2078</td>
<td>F0MAT(14H MINUTE ON FOR DAY 11,15H IS GREATER THAN 50)</td>
<td>2130/55</td>
<td>01180500</td>
</tr>
<tr>
<td>1780</td>
<td></td>
<td>MMONIT11=50</td>
<td></td>
<td>01180600</td>
</tr>
<tr>
<td>1781</td>
<td>3078</td>
<td>WRITE (6,3074)</td>
<td>AL,2,3,1T(HMONIT11),MMONIT11,HHOFT11,MMOFF11,</td>
<td>01180700</td>
</tr>
<tr>
<td></td>
<td></td>
<td>I1J=1,7)</td>
<td></td>
<td>01180800</td>
</tr>
<tr>
<td>1782</td>
<td></td>
<td>GO TO 79</td>
<td></td>
<td>01180900</td>
</tr>
<tr>
<td>1783</td>
<td>1790</td>
<td>CALL ERRPT (ERROR)</td>
<td></td>
<td>01181000</td>
</tr>
<tr>
<td>1784</td>
<td></td>
<td>WRITE (6,2790)</td>
<td>11</td>
<td>01181100</td>
</tr>
<tr>
<td>1785</td>
<td>2790</td>
<td>F0MAT(14H MINUTE ON FOR DAY 11,15H IS LESS THAN 0)</td>
<td>2130/55</td>
<td>01181200</td>
</tr>
<tr>
<td>1786</td>
<td></td>
<td>MMONIT11=0</td>
<td></td>
<td>01181300</td>
</tr>
<tr>
<td>1787</td>
<td></td>
<td>GO TO 3078</td>
<td></td>
<td>01181400</td>
</tr>
<tr>
<td>1788</td>
<td>1079</td>
<td>CALL ERRPT (ERROR)</td>
<td></td>
<td>01181500</td>
</tr>
</tbody>
</table>
1789 WRITE(6,2079) II
1790 2079 FORMAT(20H"MINUTE OFF FOR DAY II: HH:HH IS GREATER THAN 0")
1791 MMFF(I1)=59
1792 3079 WRITE(6,3074) A1,A2,NI,(HHON(J1)),MMFF(I1),HOFF(J1),MMFF(I1),
1793 J1=1,7
1794 GO TO 950
1795 1400 CALL ERRPT (ERROR)
1796 WRITE(6,2000) II
1797 2800 FORMAT(20H"MINUTE OFF FOR DAY II: HH:HH IS LESS THAN 0")
1798 MMFF(I1)=0
1799 GO TO 979
1800 1081 CALL ERRPT (ERROR)
1801 WRITE(6,2081) II
1802 2081 FORMAT(1H" TIME OFF FOR DAY II: HH:HH IS LESS THAN OR EQUAL TO TIME")
1803 WRITE(6,3074) A1,A2,NI,(HHON(J1)),MMFF(I1),HOFF(J1),MMFF(I1),
1804 J1=1,7
1805 GO TO 82
1806 1085 CALL ERRPT (ERROR)
1807 WRITE(6,2085) II
1808 3085 FORMAT(1H";II")
1809 GO TO 950
1810 1085 CALL ERRPT (ERROR)
1811 WRITE(6,2085) II
1812 2085 II=1,7
1813 1089 CALL ERRPT (ERROR)
1814 WRITE(6,2089) II
1815 IXFIX=IXINS
1816 IXFIX=IXINS
1817 IXFIX=IXINS
1818 IXFIX=IXINS
1819 IXFIX=IXINS
1820 WRITE(6,3089) A1,A2,A3,A4,JOHNAM(J1),STEPNH(J1),DDN(J1),HON(J1),
1821 JOH(J1),HOFF(J1),HOFF(J1),IXFIX,IXFIX,PRECED,IXFIX,
1822 3089 FORMAT(1H;3A1,2X,2A,R,11,212,11,217,215,AB,12)
1823 2090 II=1,7
1824 GO TO 870
1825 1091 CALL ERRPT (ERROR)
1826 WRITE(6,2091) II
1827 GO TO 289
1828 1092 ERRPT=0
1829 1092 ERRPT=0
1830 1093 CALL ERRPT (ERROR)
1831 WRITE(6,2093) II
1832 2093 FORMAT(1H" TIME ON AND TIME OFF DO NOT AGREE WITH JOB TYPE")
1833 GO TO 2399
1834 1040 CALL ERRPT (ERROR)
1835 WRITE(6,2399) II
1836 2399 GO TO 2399
1837 2940 FORMAT(5H TIME ON AND TIME OFF INDICATE STATEMENT OUT OF SEQUENCE")
<table>
<thead>
<tr>
<th>FORTRAN IV</th>
<th>LEVEL</th>
<th>MOD 0</th>
<th>MAIN</th>
<th>DATE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRITE (4,2104)</td>
<td>1378</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1379</td>
<td>2109</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1380</td>
<td>2112</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1381</td>
<td>2113</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1382</td>
<td>2114</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1383</td>
<td>2115</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1384</td>
<td>2116</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1385</td>
<td>2117</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1386</td>
<td>2118</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1387</td>
<td>2119</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1388</td>
<td>2120</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1389</td>
<td>2121</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1390</td>
<td>2122</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1391</td>
<td>2123</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1392</td>
<td>2124</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1393</td>
<td>2125</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1394</td>
<td>2126</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1395</td>
<td>2127</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1396</td>
<td>2128</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1397</td>
<td>2129</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1398</td>
<td>2130</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1399</td>
<td>2131</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1400</td>
<td>2132</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1401</td>
<td>2133</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1402</td>
<td>2134</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1403</td>
<td>2135</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1404</td>
<td>2136</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1405</td>
<td>2137</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1406</td>
<td>2138</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1407</td>
<td>2139</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1408</td>
<td>2140</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1409</td>
<td>2141</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1410</td>
<td>2142</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1411</td>
<td>2143</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1412</td>
<td>2144</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1413</td>
<td>2145</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1414</td>
<td>2146</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1415</td>
<td>2147</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1416</td>
<td>2148</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1417</td>
<td>2149</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1418</td>
<td>2150</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1419</td>
<td>2151</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1420</td>
<td>2152</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1421</td>
<td>2153</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE (4,2109)</td>
<td>1422</td>
<td>2154</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FORTRAN IV</th>
<th>LEVEL</th>
<th>MOD 0</th>
<th>MAIN</th>
<th>DATE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRITE (4,2109)</td>
<td>1423</td>
<td>2155</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYMBOL</td>
<td>LOCATION</td>
<td>SCALAR MAP</td>
<td>SYMBOL</td>
<td>LOCATION</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>----------</td>
<td>------------</td>
<td>--------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>BLANKR</td>
<td>180C</td>
<td>NAME</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>MELE</td>
<td>180E</td>
<td>MDO</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>ERRORX</td>
<td>180L</td>
<td>MVAR</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>1924</td>
<td>T</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>1938</td>
<td>T</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>1940</td>
<td>T</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>1974</td>
<td>CPU</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>TDEY</td>
<td>1988</td>
<td>K</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>A5</td>
<td>1989</td>
<td>L</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>IFG</td>
<td>1980</td>
<td>IFG</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>TIMEB</td>
<td>1964</td>
<td>WEEK</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>I3</td>
<td>1908</td>
<td>T</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>DAY</td>
<td>1960</td>
<td>T</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>NZH</td>
<td>1980</td>
<td>CASE</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>MKJOB</td>
<td>1A14</td>
<td>MXWJOB</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>IN</td>
<td>1A28</td>
<td>A6</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>VOLUME</td>
<td>1A3C</td>
<td>DCD</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>SEEK</td>
<td>1A50</td>
<td>FACTOR</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>DEVICE</td>
<td>1A64</td>
<td>K</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>CHUP</td>
<td>1A78</td>
<td>CIPT</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>TSIJ</td>
<td>1A8C</td>
<td>SIJ</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>IS</td>
<td>1A40</td>
<td>JS</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>TIMEY</td>
<td>1A84</td>
<td>NS</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>JNORKH</td>
<td>1A8C</td>
<td>JNORKM</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>JFIXIT</td>
<td>1A8C</td>
<td>JWAIT</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
<tr>
<td>OFIXIT</td>
<td>1A8C</td>
<td>OFIXIT</td>
<td>TBP</td>
<td>180F</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>ARRAY MAP</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>1AF4</td>
<td>QA</td>
<td>F1</td>
<td>1B4C</td>
</tr>
<tr>
<td>POPT</td>
<td>1B7C</td>
<td>FSC</td>
<td>F1P</td>
<td>1B4C</td>
</tr>
<tr>
<td>D1</td>
<td>1C14</td>
<td>D1</td>
<td>F2P</td>
<td>1B4C</td>
</tr>
<tr>
<td>DM1</td>
<td>1D27</td>
<td>DEVCLS</td>
<td>F3P</td>
<td>1B4C</td>
</tr>
<tr>
<td>DEVLS</td>
<td>1E44</td>
<td>DEVLS</td>
<td>F4P</td>
<td>1B4C</td>
</tr>
<tr>
<td>FGCP</td>
<td>1F41</td>
<td>FGCP</td>
<td>F5P</td>
<td>1B4C</td>
</tr>
<tr>
<td>WAT1</td>
<td>1G4C</td>
<td>WAT1</td>
<td>F6P</td>
<td>1B4C</td>
</tr>
<tr>
<td>ISCH1</td>
<td>1H20</td>
<td>ISCH1</td>
<td>F7P</td>
<td>1B4C</td>
</tr>
<tr>
<td>PIC1</td>
<td>1J46</td>
<td>PIC1</td>
<td>F8P</td>
<td>1B4C</td>
</tr>
<tr>
<td>FLCP</td>
<td>1K3C</td>
<td>FLCP</td>
<td>F9P</td>
<td>1B4C</td>
</tr>
<tr>
<td>CLH1</td>
<td>1L3C</td>
<td>CLH1</td>
<td>F10P</td>
<td>1B4C</td>
</tr>
<tr>
<td>TNSX</td>
<td>1M4F</td>
<td>TNSX</td>
<td>F11P</td>
<td>1B4C</td>
</tr>
<tr>
<td>TSLA1</td>
<td>1N84</td>
<td>TSLA1</td>
<td>F12P</td>
<td>1B4C</td>
</tr>
<tr>
<td>TSLB1</td>
<td>1O4F</td>
<td>TSLB1</td>
<td>F13P</td>
<td>1B4C</td>
</tr>
<tr>
<td>TSLA6</td>
<td>1P84</td>
<td>TSLA6</td>
<td>F14P</td>
<td>1B4C</td>
</tr>
<tr>
<td>TSLB6</td>
<td>1Q84</td>
<td>TSLB6</td>
<td>F15P</td>
<td>1B4C</td>
</tr>
</tbody>
</table>
0001 SUBROUTINE UNITXTDEVS,TMPXMA,TSEL1A,TSEL2A,TSEL3A,TSEL4A,TSEL5A, S0100001
    ITSEL6A,TCPU,TSETUP,NOVLPI,DEVSPD,DEVREC,DEVSTP,VOLUME,MULT,FACTOR, S0100002
    2VARINS,AVINST,AVSEL,AVMPX,NOVLPI,CU1,01,D1,D2,D3,D4,D5,D6,X1) S0100003
C ***************************************************************
C COMPUTE DEVICE, CHANNEL, CPU, AND SET UP TIMES FOR UNIT RECORD DEVICES
C ***************************************************************
0002 REAL MULT,NOVLPI S0100005
C DETERMINE CHANNEL AND INTERFERENCE FOR UNIT
C CALICHCNLIIC(CMPXMA,TSEL1A,TSEL2A,TSEL3A,TSEL4A,TSEL5A,TSEL6A, S0100010
    CAVSEL,AVMPX,DI,02,03,04,05,06,CU1,1,CHNA,AVCHN,ICHN)
0004 TSETUP=TSETUP*(DEVSTP*600000000)
C DOES DEVICE OPERATE AT ITS RATED SPEED
C IF(FACTOR<0.11,1,2 S0100040
C YES, COMPUTE DEVICE TIME
C 1 TDEV=(VOLUME*600000000)/DEVSPD S0100050
0007 GO TO 3 S0100060
C NO, MULTIPLY DEVICE SPEED BY FACTOR
C 2 TDEV=(VOLUME*600000000)/(DEVSPD*FACTOR)
0008 TCHNA=TCHNA*TDEV S0100070
0009 TCPU=TCPU*(VOLUME*DEVREC*AVCHN)
0010 C IS THE NUMBER OF INSTRUCTIONS IN THE PROGRAM A FUNCTION OF THIS FILE'S VOLUME
C IF(MULT-X)5,4,5 S0100100
C YES, MULTIPLY VOLUME BY VARIABLE NUMBER OF INSTRUCTIONS PER RECORD
C 4 TCPU=TCPU*{(VOLUME*VARINS*AVINST)/1.-NOVLPI)} S0100110
0012 TNOVLPI=TNOLLP*(VOLUME*VARINS*AVINST*NOVLPI) S0100120
C STORE NEW CHANNEL TIME
C 5 CALL CHNCUU (TMPXMA,TSEL1A,TSEL2A,TSEL3A,TSEL4A,TSEL5A,TSEL6A, S0100125
    CAVSEL,AVMPX,DI,02,03,04,05,06,CU1,2,CHNA,AVCHN,ICHN)
0015 RETURN
0016 END
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMPXMA</td>
<td>BA</td>
<td>TSEL1A</td>
<td>HC</td>
<td>TSEL2A</td>
<td>CO</td>
<td>TSEL3A</td>
<td>C4</td>
<td>TSEL4A</td>
<td>CA</td>
</tr>
<tr>
<td>TSEL5A</td>
<td>CC</td>
<td>TSEL6A</td>
<td>DO</td>
<td>AVSEL</td>
<td>D4</td>
<td>AVMPX</td>
<td>D8</td>
<td>D1</td>
<td>DC</td>
</tr>
<tr>
<td>O2</td>
<td>E0</td>
<td>D3</td>
<td>E4</td>
<td>D4</td>
<td>L8</td>
<td>D5</td>
<td>EC</td>
<td>O6</td>
<td>FO</td>
</tr>
<tr>
<td>CVU</td>
<td>F4</td>
<td>TECHNA</td>
<td>F8</td>
<td>AVCHN</td>
<td>FC</td>
<td>ICHN</td>
<td>100</td>
<td>TSETUP</td>
<td>104</td>
</tr>
<tr>
<td>DEVSTP</td>
<td>108</td>
<td>FACTOR</td>
<td>10C</td>
<td>TDEV</td>
<td>110</td>
<td>VOLUME</td>
<td>114</td>
<td>DEVSPD</td>
<td>118</td>
</tr>
<tr>
<td>TCPU</td>
<td>11C</td>
<td>DEVRCD</td>
<td>120</td>
<td>MULT</td>
<td>124</td>
<td>X</td>
<td>128</td>
<td>VARINS</td>
<td>12C</td>
</tr>
<tr>
<td>AVINST</td>
<td>130</td>
<td>NUVLP</td>
<td>134</td>
<td>TNDVLP</td>
<td>138</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SCALAR MAP**

**SUBPROGRAMS CALLED**

**TOTAL MEMORY REQUIREMENTS 00056C BYTES**
ITSEL1A, ITSEL3W, ITSEL4A, ITSEL5W, ITSEL6A, ITSEL6W, TCPU, S0200002
TSETUP, TNVLPL, TENVSPD, TENVSTP, VOLUME, RCDSIZE, BLKFC, MULT, TNC, SEEK, S0200003
VARINS, AVINST, AVSEL, AVMPX, NOVLPL, CUU, D1, D2, D3, D4, D5, D6, X) S0200004
C ******************************************************************************************
C COMPUTE DEVICE, CHANNEL, CPU, AND SET UP TIMES FOR DISK DEVICES
C ******************************************************************************************
C DETERMINE CHANNEL AND INTERFERENCE FOR UNIT
C
CALL CHNCU (TMPXBA, TSEL1A, TSEL2A, TSEL3A, TSEL4A, TSEL5A, TSEL6A, IAVSEL, AVMPX, D1, D2, D3, D4, D5, D6, CUU, I, TCHNA, AVCHN, ICHN) S0200010
CALL CHNCU (TMPXBW, TSEL1W, TSEL2W, TSEL3W, TSEL4W, TSEL5W, TSEL6W, IAVSEL, AVMPX, D1, D2, D3, D4, D5, D6, CUU, I, TCHNA, AVCHN, ICHN) S0200020
TSETUP = TSETUP + (TENVSTP*60000000) S0200025
TCHNA = (TNC*1000000*RCDSIZE)/TENVSPD S0200030
TCHNW = (TCHNA + TNC) / BLKFC S0200040
TDEV = TDEV + TCHNX + TCHNW S0200050
IF (TCHNA > TCHNW) TCHNA = TCHNW S0200060
1 TCHNW = TCHNW - TCHNA S0200070
GO TO 2 S0200080
2 TCHNW = 0. S0200090
3 IF (TCHNA > TCHNW) TCHNA = TCHNW S0200100
4 TCHNW = TCHNW - TCHNA S0200110
5 TCHNA = TCHNA - TCHNA S0200120
6 TCHNA = TCHNW - TCHNA S0200130
7 TCHNA = TCHNW - TCHNA S0200140
C IS THE NUMBER OF INSTRUCTIONS IN THE PROGRAM A FUNCTION OF THIS FILE'S VOLUME
C
IF (MULT = 17, 6, 7) S0200150
C YES. MULTIPLY VOLUME BY VARIABLE NUMBER OF INSTRUCTIONS PER RECORD
C
6 TCPU = TCPU + (TNC*VARINS*AVINST)*(1 - NOVLPL) S0200160
7 TNVLPL = TNVLPL + (TNC*VARINS*AVINST*NOVLPL) S0200170
C STORE NEW CHANNEL TIME
C
CALL CHNCU (TMPXBA, TSEL1A, TSEL2A, TSEL3A, TSEL4A, TSEL5A, TSEL6A, IAVSEL, AVMPX, D1, D2, D3, D4, D5, D6, CUU, 2, TCHNA, AVCHN, ICHN) S0200173
CALL CHNCU (TMPXBW, TSEL1W, TSEL2W, TSEL3W, TSEL4W, TSEL5W, TSEL6W, IAVSEL, AVMPX, D1, D2, D3, D4, D5, D6, CUU, 2, TCHNA, AVCHN, ICHN) S0200180
RETURN S0200186
END S0200190
SUBROUTINE TAPEX(DEV,TMPXB,A,TSEL1A,TSEL2A,TSEL3A,TSEL4A,TSEL5A,TSEL6A, TSEL6A,TCPU,TSUP,TWINDUP,TNVLPL,DEVSPD,DEVSTP,VOLUME,RCDISIZ, 2BFLKCT,MULT,STRSTP,REWIND,DEVDEVN,VARINS,AVINST,AVSL,AVMPX,NOVLPL, S030001 0001
ITSEL6A,TCPU,TSUP,TWINDUP,TNVLPL,DEVSPD,DEVSTP,VOLUME,RCDISIZ, S030002
2BFLKCT,MULT,STRSTP,REWIND,DEVDEVN,VARINS,AVINST,AVSL,AVMPX,NOVLPL, S030003
3CUU,DL2,02,04,05,06,07) S030004
C ******************************************************************************************
C COMPUTE DEVICE, CHANNEL, CPU, SET UP, ANDREWIND TIMES FOR TAPE DEVICES
C ******************************************************************************************
0002
C DETERMINE CHANNEL AND INTERFERENCE FOR UNIT
C CALL CHNCOU (TMPXB,A,TSEL1A,TSEL2A,TSEL3A,TSEL4A,TSEL5A,TSEL6A, S030001
1AVSEL,AVMPX,DL2,02,04,05,06,CUU,1,TCHNA,AVCHN,ICHN) S030002
C TSEUP=TSUP+(DEVSTP*6000000) S030003
C TDEV=(STRSTP*VOLUME*1000/BLKFCT)+(VOLUME*1000000*RCDISIZ)/DEVSPD) S030005
C TCHNA=TCHNA+TDEV S030006
C TCPU=TCPU+(VOLUME*RCDISIZ*AVCHN) S030080
C C IS THE NUMBER OF INSTRUCTIONS IN THE PROGRAM A FUNCTION OF THIS FILE'S VOLUME
C IF(MULT-X12,1,2 S030090
C C YES, MULTIPLY VOLUME BY VARIABLE NUMBER OF INSTRUCTIONS PER RECORD
C 1 TCPU=TCPU+(VOLUME*VARINS*AVINST)*((1-NOVLPL)) S030010
C TNOVLPL=TNVLPL+(VOLUME*VARINS*AVINST*NOVLPL) S030011
C 2 TWIND=(((VOLUME*RCDISIZ)/DEVDEV)+(.5*(VOLUME/BLKFCT)))*6000000* S030012
C IREWINDI/2400*12) S030012
C IF(TWINDUP=TWIND+1,4,4 S030020
C 3 TWINDUP=TWIND S030015
C C STORE NEW CHANNEL TIME
C CALL CHNCOU (TMPXB,A,TSEL1A,TSEL2A,TSEL3A,TSEL4A,TSEL5A,TSEL6A, S030015
1AVSEL,AVMPX,DL2,02,04,05,06,CUU,2,TCHNA,AVCHN,ICHN) S030016
C RETURN S030015
C END S030017
### SCALAR MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMPXBA</td>
<td>80</td>
<td>TSELF1A</td>
<td>84</td>
<td>TSELF2A</td>
<td>88</td>
<td>TSELF3A</td>
<td>8C</td>
<td>TSELF4A</td>
<td>A0</td>
</tr>
<tr>
<td>TSELF5A</td>
<td>C4</td>
<td>TSELF6A</td>
<td>C8</td>
<td>AVSEL</td>
<td>CC</td>
<td>AVMPX</td>
<td>D0</td>
<td>D1</td>
<td>D4</td>
</tr>
<tr>
<td>02</td>
<td>08</td>
<td>03</td>
<td>0C</td>
<td>D4</td>
<td>E0</td>
<td>D5</td>
<td>E4</td>
<td>D6</td>
<td>E8</td>
</tr>
<tr>
<td>CHUN</td>
<td>EC</td>
<td>TCHNA</td>
<td>F0</td>
<td>AVCHN</td>
<td>F4</td>
<td>ICHN</td>
<td>F8</td>
<td>TSETUP</td>
<td>FC</td>
</tr>
<tr>
<td>DEVSIP</td>
<td>100</td>
<td>TOFV</td>
<td>104</td>
<td>STRSTP</td>
<td>108</td>
<td>VOLUME</td>
<td>10C</td>
<td>PLKFCT</td>
<td>110</td>
</tr>
<tr>
<td>RCDSIZ</td>
<td>114</td>
<td>DEVSPO</td>
<td>118</td>
<td>TCPU</td>
<td>11C</td>
<td>MULT</td>
<td>120</td>
<td>X</td>
<td>124</td>
</tr>
<tr>
<td>VAINST</td>
<td>120</td>
<td>AVINST</td>
<td>12C</td>
<td>NOVLUR</td>
<td>130</td>
<td>TNVLUR</td>
<td>134</td>
<td>TWND</td>
<td>138</td>
</tr>
<tr>
<td>DEVLEN</td>
<td>130</td>
<td>RWIND</td>
<td>140</td>
<td>RWIND</td>
<td>144</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### SUBPROGRAMS CALLED

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHINCU</td>
<td>148</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TOTAL MEMORY REQUIREMENTS**: **00062E BYTES**
SUBROUTINE CHNCUU(CC0, C1, C2, C3, C4, C5, AVSEL, AVMPX, D1, D2, D3, D4, D5, S0400010
  1B6, CUU, I, TCHN, AVCHN, ICHN)

C **********************************************************************

C DETERMINE CHANNEL AND INTERFERENCE FOR UNIT -- OR STORE NEW CHANNEL TIME
C
C **********************************************************************

0002    GO TO (20, 29), I
0003       IF(ABS(CUU) - ABS(D1)) 10, 11, 21
0004       IF(ABS(CUU) - ABS(D2)) 11, 12, 22
0005       IF(ABS(CUU) - ABS(D3)) 12, 13, 23
0006       IF(ABS(CUU) - ABS(D4)) 13, 14, 24
0007       IF(ABS(CUU) - ABS(D5)) 14, 15, 25
0008    25 IF(ABS(CUU) - ABS(D6)) 15, 16, 36
0009    10 TCHN = C0
0010       ICHN = 7
0011    11 AVCHN = AVMPX
0012    RETURN
0013    12 TCHN = C1
0014       ICHN = 1
0015    GO TO 27
0016    13 TCHN = C2
0017       ICHN = 2
0018    GO TO 27
0019    14 TCHN = C3
0020       ICHN = 3
0021    GO TO 27
0022    15 TCHN = C4
0023       ICHN = 4
0024    GO TO 27
0025    16 TCHN = C5
0026       ICHN = 5
0027    GO TO 27
0028    17 TCHN = C6
0029       ICHN = 6
0030    27 AVCHN = AVSEL
0031    RETURN
0032    28 GO TO (31, 32, 33, 34, 35, 36, 30), ICHN
0033    30 CO = TCHN
0034    RETURN
0035    31 C1 = TCHN
0036    RETURN
0037    32 C2 = TCHN
0038    RETURN
0039    33 C3 = TCHN
0040    RETURN
0041    34 C4 = TCHN
0042    RETURN
### SCALAR MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>140</td>
<td>C1U</td>
<td>144</td>
</tr>
<tr>
<td>5</td>
<td>154</td>
<td>D5</td>
<td>158</td>
</tr>
<tr>
<td>ICIN</td>
<td>168</td>
<td>AVCHN</td>
<td>16C</td>
</tr>
<tr>
<td>C3</td>
<td>17C</td>
<td>C4</td>
<td>180</td>
</tr>
<tr>
<td>D1</td>
<td>146</td>
<td>D2</td>
<td>14C</td>
</tr>
<tr>
<td>D4</td>
<td>150</td>
<td>D3</td>
<td>150</td>
</tr>
<tr>
<td>D5</td>
<td>152</td>
<td>TCHN</td>
<td>160</td>
</tr>
<tr>
<td>C1</td>
<td>174</td>
<td>C2</td>
<td>178</td>
</tr>
<tr>
<td>C1</td>
<td>170</td>
<td>C6</td>
<td>188</td>
</tr>
<tr>
<td>C1</td>
<td>174</td>
<td>AVSEL</td>
<td>18C</td>
</tr>
</tbody>
</table>

Total memory requirements: 00005FC bytes
C SEPARATE RESOURCE USES INTO MULTIPLEXOR MODE, BURST MODE, AND CPU PORTIONS
C
C INTEGER CPU
INTEGER CPU
C DIMENSION FRACB(12,2), FRACB(12,2)
DIMENSION FRACB(12,2), FRACB(12,2)
C
C DETERMINE MAXIMUM SELECTOR CHANNEL USAGE
C
C TOTAL = MAX1(TSEL1A+TSEL1W, TSEL2A+TSEL2W, TSEL3A+TSEL3W, TSEL4A+TSEL4W, TSEL5A+TSEL5W, TSEL6A+TSEL6W)
C
C ARE THERE ANY MULTIPLEXOR CHANNEL OPERATIONS FOR THIS PROGRAM
C
C IF (TMPXMA+TMPXMB=0) 17, 7, 10
C C NO. SET MULTIPLEXOR MODE FRACTION TO ONE, AND SET BURST MODE FRACTION TO ZERO
C
7 RATIO1 = 1.
RATIO2 = 0.

GO TO 15
C
C YES. COMPUTE MULTIPLEXOR MODE AND BURST MODE FRACTIONS
C
10 RATIO1 = TMPXMA / (TMPXMA+TMPXBA)
RATIO2 = TMPXBA / (TMPXMA+TMPXBA)
RATIO3 = TMPXMA / (TMPXMA+TMPXBA+TCPU*RATIO2)
RATIO4 = (TCPU*RATIO2) / (TMPXMA+TCPU*RATIO2)

GO TO 25
C
C ARE THERE ANY BURST MODE OPERATIONS FOR THIS PROGRAM
C
C IF (RATIO2 = 0) 15, 15, 20
C C NO. SET NON-OVERLAPPED BURST MODE AND NON-OVERLAPPED CPU FRACTIONS TO ZERO
C
15 RATIO3 = 0.
RATIO4 = 0.

GO TO 25
C
C YES. COMPUTE NON-OVERLAPPED BURST MODE AND NON-OVERLAPPED CPU FRACTIONS
C
20 RATIO3 = TMPXBA / (TMPXBA+TCPU*RATIO2)
RATIO4 = (TCPU*RATIO2) / (TMPXBA+TCPU*RATIO2)

GO TO 25
C
C CAN BURST MODE OPERATIONS BE OVERLAPPED WITH INSTRUCTION EXECUTION
C
25 IF(IPCPU=204011,1,2)
C
C NO. COMPUTE TOTAL PROGRAM TIME
C
1 TOTAL=AMAX1(TOTAL,AMAX1(TMPXMA,TMPXWB,TCPURAT1O1)+
1 AMAX1(TMPXMA,TMPXWB,TCPURAT1O3)+AMAX1(TCPURAT1O2,TMPXW*RATIO4)+
2 TDWLP
C
C YES. COMPUTE TOTAL PROGRAM TIME
C
2 TOTAL=AMAX1(TOTAL,AMAX1(TMPXMA,TMPXWB,TCPURAT1O1)+
1 AMAX1(TMPXMA,TMPXWB,TCPURAT1O2)+TDWLP
C
C SET UP POINTERS FOR RESOURCES
C
3 DO 4 I=1,12
FRACI(I,1)=I-1
4 FRAC2(I,1)=I-1
C
C ARE THERE ANY MULTIPLEX MODE OPERATIONS FOR THIS PROGRAM
C
IF(RATIO1=0.140,40,45)
C
C NO. SET MULTIPLEX MODE PORTION TO ZERO
C
40 DO 41 I=1,12
41 FRAC(I,2)=0.
C
C GO TO 49
C
C YES. COMPUTE FRACTIONS FOR MULTIPLEX MODE PORTION
C
45 FRACI(I,2)=TMPXMA/TOTAL
0030 FRACR(I,2)=0.
0031 FRAC9(I,2)=(TCPURAT1O1)/TOTAL
C
CALL FRACTLB(FRAC,FRACL,TOTAL,RATIO1,TSEL1A,TSEL2A,TSEL3A,TSEL4A,
ITSEL5A,TSEL6A,TDWLP,RATIO1)
C
C ARE THERE ANY BURST MODE OPERATIONS FOR THIS PROGRAM
C
49 IF(RATIO2=0.150,50,55)
C
C NO. SET BURST MODE PORTION TO ZERO
C
50 DO 51 I=1,12
51 FRACB(I,2)=0.
C CAN BURST MODE OPERATIONS BE OVERLAPPED WITH INSTRUCTION EXECUTION
C
0036 IF(CPU-2040)160,60,59
C YES. RETURN TO MAIN PROGRAM
C
0037 59 RETURN
C
0038 C BURST MODE OPERATIONS EXIST. COMPUTE FRACTIONS FOR BURST MODE PORTION
C
0039 55 FRACB(1,2)=0.
0040 FRACB(18,2)=MPXB/MPX
C
0041 C CAN BURST MODE OPERATIONS BE OVERLAPPED WITH INSTRUCTION EXECUTION
C
0042 IF(CPU-2040)15,5,6
C
0043 C NO. SET CPU FRACTION FOR BURST MODE PORTION EQUAL TO ZERO
C
0044 5 FRACB(9,2)=0.
0045 CALL FRCTBL (FRACB,TOTAL/(RATIO*RATIO),TSSEL,TSEL2A,TSEL3A,
0046 ITSEL4A,TSEL5A,TSEL6A,TNOLP,RATIO*RATIO)
C
0047 C ARE THERE ANY CPU OPERATIONS
C
0048 IF(RATIO-0.160,60,65
C
0049 C NO. SET CPU PORTION TO ZERO
C
0050 DO 61 I=1,12
0051 61 FRACC(I,2)=0.
0052 RETURN
C
0053 C YES. COMPUTE FRACTIONS FOR CPU PORTION
C
0054 63 FRACC(1,2)=0.
0055 FRACC(3,2)=0.
0056 FRACC(6,2)=ICP*(MPX)/(RATIO*RATIO)
0057 CALL FRCTBL (FRACC,TOTAL/(RATIO*RATIO),TSSEL,TSEL2A,TSEL3A,
0058 ITSEL4A,TSEL5A,TSEL6A,TNOLP,RATIO*RATIO)
C
0059 C BURST MODE OPERATIONS AND INSTRUCTION EXECUTION CAN BE OVERLAPPED
C
0060 6 FRACB(9,2)=ICP*(MPX)/TOTAL
0061 CALL FRCTBL (FRACB,TOTAL/(RATIO*RATIO),TSSEL,TSEL2A,TSEL3A,TSEL4A,
0062 ITSEL5A,TSEL6A,TNOLP,RATIO*RATIO)
0063 RETURN
SCALAR MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTAL</td>
<td>130</td>
</tr>
<tr>
<td>TSEL3A</td>
<td>144</td>
</tr>
<tr>
<td>TSEL5W</td>
<td>158</td>
</tr>
<tr>
<td>RATIO1</td>
<td>16C</td>
</tr>
<tr>
<td>CPU</td>
<td>130</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSEL1A</td>
<td>134</td>
</tr>
<tr>
<td>TSEL3W</td>
<td>148</td>
</tr>
<tr>
<td>TSEL6A</td>
<td>15C</td>
</tr>
<tr>
<td>RATIO2</td>
<td>170</td>
</tr>
<tr>
<td>TMPX8W</td>
<td>184</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSEL1W</td>
<td>138</td>
</tr>
<tr>
<td>TSEL4A</td>
<td>14C</td>
</tr>
<tr>
<td>TSEL6W</td>
<td>160</td>
</tr>
<tr>
<td>RATIO3</td>
<td>174</td>
</tr>
<tr>
<td>TMPX4W</td>
<td>188</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSEL2A</td>
<td>13C</td>
</tr>
<tr>
<td>TSEL4W</td>
<td>150</td>
</tr>
<tr>
<td>TMPX4A</td>
<td>164</td>
</tr>
<tr>
<td>RATIO4</td>
<td>178</td>
</tr>
<tr>
<td>TNOVLP</td>
<td>18C</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSEL2W</td>
<td>140</td>
</tr>
<tr>
<td>TSEL5A</td>
<td>154</td>
</tr>
<tr>
<td>TMPX8A</td>
<td>168</td>
</tr>
<tr>
<td>TCPU</td>
<td>17C</td>
</tr>
<tr>
<td></td>
<td>190</td>
</tr>
</tbody>
</table>

ARRAY MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRAACM</td>
<td>194</td>
</tr>
<tr>
<td>FRACC</td>
<td>198</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRACB</td>
<td>198</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSRK1</td>
<td>196</td>
</tr>
</tbody>
</table>

SUBPROGRAMS CALLED

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMAX1</td>
<td>1A4</td>
</tr>
</tbody>
</table>

TOTAL MEMORY REQUIREMENTS 000996 BYTES
SUBROUTINE FRCTBL(FRAC, DENOM, TSEL1A, TSEL2A, TSEL3A, TSEL4A, TSEL5A)  
DIMENSION FRAC(12, 2)  
C COMPLETE COMPUTATION OF FRACTIONS FOR THIS PORTION AND THEN SORT FRACTIONS  
C  
C COMPUTE SELECTOR CHANNEL FRACTIONS  
C  
FRAC(12, 2) = TSEL1A/DENOM  
FRAC(13, 2) = TSEL2A/DENOM  
FRAC(14, 2) = TSEL3A/DENOM  
FRAC(15, 2) = TSEL4A/DENOM  
FRAC(16, 2) = TSEL5A/DENOM  
FRAC(17, 2) = TSEL6A/DENOM  
C COMPUTE NON-OVERLAPPED CPU FRACTION  
C  
FRAC(11, 2) = TNOVLP/DENOM  
C COMPUTE WAIT FRACTION  
C  
FRAC(12, 2) = AMAX1(0, RATIO - FRAC(11, 2)) - AMAX1(FRAC(1, 2), FRAC(2, 2), FRAC(3, 2), FRAC(4, 2), FRAC(5, 2), FRAC(6, 2), FRAC(7, 2), FRAC(8, 2), FRAC(9, 2))  
C SORT FRACTIONS  
C  
DO 3 I = 1, 8  
K = I + 1  
DO 3 J = K, 9  
IF(FRAC(J, 2) > FRAC(I, 2)) GOTO 3  
3 CONTINUE  
C MARK END OF TABLE FOR OVERLAPPED RESOURCE USES  
C  
FRAC(10, 1) = 9.  
FRAC(10, 2) = 0.  
RETURN  
END
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSEL1A</td>
<td>B3</td>
<td>TSEL2A</td>
<td>B6</td>
<td>TSEL3A</td>
<td>B4</td>
<td>TSEL4A</td>
<td>B8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSEL5A</td>
<td>B6</td>
<td>TSEL6A</td>
<td>C0</td>
<td>TNOVLP</td>
<td>C4</td>
<td>RATIO</td>
<td>C8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>K</td>
<td>D0</td>
<td>J</td>
<td>D4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRAC</td>
<td>D8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Array Map**

**Subprograms Called**

**Total Memory Requirements 000320 Bytes**
SUBROUTINE TIME2(FRAC1,FRAC2,FRAC12,CPU,TIMEB)

INTEGER CPU
DIMENSION FRAC1(12,21),FRAC2(12,2),FRAC12(11,2)

DO 2 J=1,11
DO 2 K=1,2
2 FRAC12(J,K)=0.
I=1

C FIND THE NUMBER OF RESOURCES USED BY THE FIRST PROGRAM

IF(FRAC1(I,2)-0.)100,100,1
1 CONTINUE
100 M=M-1

C IS THIS NUMBER ZERO

IF(M-0)800,800,110

C NO. FIND THE NUMBER OF RESOURCES USED BY THE SECOND PROGRAM

110 DO 101 N=1,10
101 CONTINUE
120 N=N-1

C IS THIS NUMBER ZERO

IF(N-0)800,800,210

C NO. DETERMINE IF ALL RESOURCE USES BY BOTH PROGRAMS CAN BE OVERLAPPED

210 DO 201 J=1,N

C IS PROGRAM TWO'S RESOURCE THE MULTIPLEXOR CHANNEL IN MULTIPLEX MODE

IF(FRAC2(J,1)-0.)300,300,220

C NO. IS PROGRAM TWO'S RESOURCE THE MULTIPLEXOR CHANNEL IN BURST MODE

220 IF(FRAC2(J,1)-7.1500,400,230

C NO. CAN BURST MODE MPX OPERATIONS BE OVERLAPPED WITH INSTRUCTION EXECUTION
C 230 IF(CPU=2040,240,500
C NO. IS PROGRAM TWO'S RESOURCE THE CPU
C 240 IF(FRAC2(J,1)=8.1500,250,500
C YES. IS THERE A RESOURCE CONFLICT BETWEEN PROGRAM TWO AND PROGRAM ONE
C 250 DO 251 K=1,M
C IFRAC2(K,1)=7.1251,600,260
C 260 IF(FRAC2(K,1)=8.1251,600,251
C 251 CONTINUE
C NO CONFLICT. CHECK NEXT PROGRAM TWO RESOURCE
C GO TO 201
C PROGRAM TWO USES MPX CHANNEL IN MPX MODE. IS THERE A RESOURCE CONFLICT
C 300 DO 301 K=1,M
C IFRAC1(K,1)=7.1301,600,301
C 301 CONTINUE
C NO CONFLICT. CHECK NEXT PROGRAM TWO RESOURCE
C GO TO 201
C PROGRAM TWO USES MPX CHANNEL IN BURST MODE. IS THERE A RESOURCE CONFLICT
C 400 DO 401 K=1,M
C IFRAC1(K,1)=0.1600,600,410
C 410 IF(FRAC1(K,1)=7.1401,600,420
C 420 IF(CPU=2040)430,430,491
C 430 IFRAC1(K,1)=8.1401,600,401
C 401 CONTINUE
C NO CONFLICT. CHECK NEXT PROGRAM TWO RESOURCE
C GO TO 201
C ARE BOTH PROGRAMS USING THE CPU OR THE SAME SELECTOR CHANNEL
C 500 DO 501 K=1,M
C IFRAC1(K,1)=FRAC2(J,1)501,600,501
C 501 CONTINUE
C NO. CHECK NEXT PROGRAM TWO RESOURCE
0042 201 CONTINUE
0043 C NO RESOURCE CONFLICT, SAVE NUMBER OF RESOURCES FOR EACH PROGRAM AND
0044 C INCREMENT PROGRAM TWO'S TIME BY PRODUCT OF TIMES FOR THE TWO SETS OF RESOURCES
0045 FRAC21(1,1) = M
0046 FRAC21(1,2) = N
0047 TIMEB = TIMEB + (FRAC11, 1) * (FRAC21, 1) + (FRAC21, 2) * (FRAC21, 2)
0048 I = I + 1
0049 GO TO 200
0050 C RESOURCE CONFLICT, REMOVE A RESOURCE FROM PROGRAM TWO
0051 600 M = N - 1
0052 C IS THE NUMBER OF RESOURCES EQUAL TO ZERO
0053 IF (M = 0) GO TO 700, 700, 210
0054 C YES, REMOVE A RESOURCE FROM PROGRAM ONE
0055 700 M = M - 1
0056 C IS THE NUMBER OF RESOURCES EQUAL TO ZERO
0057 IF (M = 0) GO TO 800, 800, 110
0058 C YES, MARK END OF TABLE
0059 800 FRAC121(1,1) = 0.
0060 FRAC121(1,2) = 0.
0061 C SAVE POINTER TO PROGRAM ONE'S OVERLAPPED CPU TIME
0062 DO 900 L = 1, 9
0063 IF (FRAC11, 1) - 0.1900, 901, 900
0064 900 CONTINUE
0065 901 FRAC121(1,1) = L
0066 C INCREMENT PROGRAM TWO'S TIME BY ITS NON-CONFLICT NON-OVERLAPPED CPU TIME
0067 TIMEB = TIMEB + (FRAC211, 2) * (FRAC11, 2) * (FRAC11, 2)
0068 C SAVE POINTER TO PROGRAM TWO'S OVERLAPPED CPU TIME
0069 DO 1000 L = 1, 9
0070 IF (FRAC211, 1) = 8.11000, 1001, 1000
0071 1000 CONTINUE
0061  1000 CONTINUE
0062    1001 FRAC12(11,2)=L  S0700550
                      S0700560
               C
              C INCREMENT TIME FOR PGM TWO BY PGM ONE'S NON-CONFLICT NON-OVERLAPPED CPU TIME
               C
0063    TIMEB=TIMEB+((FRAC1(11,2))*(FRAC2(1,2)-FRAC2(L,2)))  S0700565
               C
              C INCREMENT TIME FOR PROGRAM TWO BY PROGRAM ONE'S WAIT TIME
               C
0064    TIMER=TIMER+FRAC1(12,2)  S0700566
0065    RETURN
0066    END
               S0700570
                      S0700580
SUBROUTINE TIME3(FRAC1,FRAC2,FRAC3,FRAC12,CPU,TIMEB,TIMEC)
C*******************************************************************************
C CALCULATE MULTIPROGRAMMING TIME FOR THIRD PRIORITY PROGRAM
C*******************************************************************************
INTEGER CPU
DIMENSION FRAC11(12,2),FRAC21(12,2),FRAC31(12,2),FRAC1211,2
DIMENSION FRAC231(11,2)
TIME=0.
DO 700 I=1,10
C IS THERE A NO CONFLICT PROGRAM ONE-PROGRAM TWO RESOURCE COMBINATION
IF(FRAC12(1,1)<.1800,800,110)
C YES, CALCULATE PROGRAM THREE MULTIPROGRAMMING TIME FOR THAT COMBINATION
110 DO 101 N=1,10
C FIND THE NUMBER OF RESOURCES USED BY THE THIRD PROGRAM
IF(FRAC3(1,2)<.1200,200,101
101 CONTINUE
200 N=N-1
DO 202 L=1,2
C FIND THE NUMBER OF RESOURCES USED BY ONE OF THE OTHER PROGRAMS
N=FRAC12(1,1)
C IS IT PROGRAM ONE OR PROGRAM TWO
GO TO (210,5210),L
C PROGRAM ONE. DETERMINE IF RESOURCE USES OF PGM 1 AND PGM 3 CAN BE OVERLAPPED
210 DO 201 J=1,N
C IS PROGRAM THREE'S RESOURCE THE MULTIPLEXOR CHANNEL IN MULTIPLEX MODE
IF(FRAC3(1,1)<.1300,300,220
C NO. IS PROGRAM THREE'S RESOURCE THE MULTIPLEXOR CHANNEL IN BURST MODE
220 IF(FRAC3(1,1)<.1500,400,230
C NO. CAN BURST MODE MPX OPERATIONS BE OVERLAPPED WITH INSTRUCTION EXECUTION
230 IF(CPU-2040)1240,240,500

C NO. IS PROGRAM THREE'S RESOURCE THE CPU

240 IF(FRA(1,J,1)-8.1500,250,500

C YES. IS THERE A RESOURCE CONFLICT BETWEEN PROGRAM THREE AND PROGRAM ONE

250 DO 251 K=1,M
251 CONTINUE

260 IF(FRA(1,K,1)-7.1251,600,260
261 CONTINUE

C NO CONFLICT. CHECK NEXT PROGRAM THREE RESOURCE

GO TO 201

C PROGRAM THREE USES THE MPX CHANNEL IN MPX MODE. IS THERE A RESOURCE CONFLICT

300 DO 301 K=1,M
301 CONTINUE

310 IF(FRA(1,K,1)-7.1501,600,301
311 CONTINUE

C NO CONFLICT. CHECK NEXT PROGRAM THREE RESOURCE

GO TO 201

C PROGRAM THREE USES MPX CHANNEL IN BURST MODE. IS THERE A RESOURCE CONFLICT

400 DO 401 K=1,M
401 CONTINUE

410 IF(FRA(1,K,1)-7.1401,600,410
411 CONTINUE

C NO CONFLICT. CHECK NEXT PROGRAM THREE RESOURCE

GO TO 201

C ARE BOTH PROGRAMS USING THE CPU OR THE SAME SELECTOR CHANNEL

500 DO 501 K=1,M
501 CONTINUE

C NO. CHECK NEXT PROGRAM THREE RESOURCE
C
201 CONTINUE
C NO RESOURCE CONFLICT BETWEEN PROGRAMS THREE AND ONE
C SAVE NUMBER OF RESOURCES FOR PROGRAM THREE
C
0040 NX=N
0041 GO TO 202
C RESOURCE CONFLICT. REMOVE A RESOURCE FROM PROGRAM THREE
C
0042 600 N=N-1
C IS THE NUMBER OF RESOURCES EQUAL TO ZERO
C IF(IN=0)1700,700,210
C PROGRAM TWO. DETERMINE IF RESOURCE USES OF PGM 2 AND PGM 3 CAN BE OVERLAPPED
C
0044 5210 DO 5201 J=1,N
C IS PROGRAM THREE'S RESOURCE THE MULTIPLEXOR CHANNEL IN MULTIPLEX MODE
C IF(FRAC3(J,1)<0.15300,5300,5220
C NO. IS PROGRAM THREE'S RESOURCE THE MULTIPLEXOR CHANNEL IN BURST MODE
C 5220 IF(FRAC3(J,1)<7.15500,5400,5230
C NO. CAN BURST MODE MPX OPERATIONS BE OVERLAPPED WITH INSTRUCTION EXECUTION
C 5230 IF(ICPU=204015240,5240,5500
C NO. IS PROGRAM THREE'S RESOURCE THE CPU
C 5240 IF(FRAC3(J,1)<8.15500,5250,5500
C YES. IS THERE A RESOURCE CONFLICT BETWEEN PROGRAM THREE AND PROGRAM TWO
C
0049 5250 DO 5251 K=1,M
0050 IF(FRAC2(K,1)<7.15251,5600,5260
0051 5260 IF(FRAC2(K,1)<8.15251,5600,5251
0052 5251 CONTINUE
C NO CONFLICT. CHECK NEXT PROGRAM THREE RESOURCE
C
0053 GO TO 5201
C PROGRAM THREE USES THE MPX CHANNEL IN MPX MODE. IS THERE A RESOURCE CONFLICT
C
5300 DO 5301 K=1,M
0054 IF(FRAC2(K,1)-7.)5301,5600,5301
0055 5301 CONTINUE
0056
C NO CONFLICT. CHECK NEXT PROGRAM THREE RESOURCE
C
0057 GO TO 5201
C
C PROGRAM THREE USES MPX CHANNEL IN BURST MODE. IS THERE A RESOURCE CONFLICT
C
5400 DO 5401 K=1,M
0058 IF(FRAC2(K,1)-0.,15600,5600,5410
0059 5410 IF(FRAC11(K,1)-7.,15401,5600,5420
0060 5420 IF(CPU-2040)5430,5430,5401
0061 5430 IF(FRAC2(K,1)-8.,15401,5600,5401
0062 5401 CONTINUE
0063
C NO CONFLICT. CHECK NEXT PROGRAM THREE RESOURCE
C
0064 GO TO 5201
C
C ARE BOTH PROGRAMS USING THE CPU OR THE SAME SELECTOR CHANNEL
C
5500 DO 5501 K=1,M
0065 IF(FRAC2(K,1)-FRAC3(J,1)5501,5600,5501
0066 5501 CONTINUE
0067
C NO. CHECK NEXT PROGRAM THREE RESOURCE
C
0068 5201 CONTINUE
0069 GO TO 202
C
C RESOURCE CONFLICT. REMOVE A RESOURCE FROM PROGRAM THREE
C
5600 N=N-1
0070
C IS THE NUMBER OF RESOURCES EQUAL TO ZERO
C
5700 N=N-1
0071 IF(N=0)5700,5700,5210
0072
C YES. REMOVE A RESOURCE FROM PROGRAM TWO
C
0073 5700 N=N-1
0074
C IS THE NUMBER OF RESOURCES EQUAL TO ZERO
C
0073 IFIM=01700,700,5710
0074 C NO. RESET PROGRAM THREE'S RESOURCE COUNT
0075 5710 N=NX
0076 GO TO 5210
0077 202 CONTINUE
0078 C NO RESOURCE CONFLICT AMONG THE THREE PROGRAMS. INCREMENT PROGRAM THREE'S TIME
0079 BY PRODUCT OF TIMES FOR THE THREE SETS OF RESOURCES
0080 J=FRACL(1,1)
0081 T1MEC=TIMEC+1((FRACL(1,2)-(FRACL(J+1,2)))*(FRACL(1,2)-(FRACL(M+1,2)))*
0082 (FRACL(1,2)-(FRACL(N+1,2))))
0083 C ARE THERE ANY MORE NO CONFLICT PROGRAM ONE-PROGRAM TWO RESOURCE COMBINATIONS
0084 700 CONTINUE
0085 C NO. INCREMENT PROGRAM THREE'S TIME BY ITS NON-CONFLICT NON-OVERLAPPED CPU TIME
0086 800 J=FRACL(11,1)
0087 M=FRACL(11,2)
0088 TIMEC=TIMEC+1((FRACL(1,2)-(FRACL(J,2)))*(FRACL(1,2)-(FRACL(M,2)))*
0089 (FRACL(1,2))}
0090 C INCREMENT PROGRAM THREE'S TIME BY PROGRAM ONE'S WAIT TIME
0091 IF(FRACL(12,2))=0.1805,805,804
0092 804 CALL TIME2 (FRACL2,FRACL1,FRACL23,CPUM,TIME)
0093 TIMEC=TIMEC+TIME2+FRACL(12,2))
0094 C INCREMENT PROGRAM THREE'S TIME BY PROGRAM TWO'S WAIT TIME
0095 805 TIMEC=TIMEC+TIME2+FRACL(12,2))
0096 RETURN
0097 END
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME0</td>
<td>234</td>
<td>I</td>
<td>238</td>
<td>N</td>
<td>23C</td>
<td>L</td>
<td>240</td>
<td>M</td>
<td>244</td>
</tr>
<tr>
<td>I</td>
<td>248</td>
<td>CPU</td>
<td>24C</td>
<td>K</td>
<td>250</td>
<td>NX</td>
<td>254</td>
<td>TIMEC</td>
<td>258</td>
</tr>
<tr>
<td>TIMEB</td>
<td>25C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ARRAY MAP**

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRAC1</td>
<td>260</td>
<td>FRAC2</td>
<td>264</td>
<td>FRAC3</td>
<td>268</td>
<td>FRAC12</td>
<td>26C</td>
<td>FRAC23</td>
<td>270</td>
</tr>
</tbody>
</table>

**SUBPROGRAMS CALLED**

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME2</td>
<td>2C8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TOTAL MEMORY REQUIREMENTS 000A44 BYTES**
SUBROUTINE FILE(D1,D2,A0,A1,A2,FILNAM,FILENAME,MCUU,CUU), $0900010
1CUUPNT,DEVCLS,BLANK4,BLANK8,B,ERROR,D3,D4) $0900015
C **************************************************************/
C READ AND PROCESS JBD AND JFD STATEMENTS
C **************************************************************/
0002 INTEGER CUUPNT(20),CUUPNT(20),ERROR
0003 DIMENSION CUU(20),DEVCLS(10),A(2G1),C(7)
0004 DOUBLE PRECISION FILEAM(201),BLANK8,B(7)
0005 1=1
0006 J1=MINO(7,FILNAM)
C READ A JBD OR JFD STATEMENT
C 0007 1 READ (5,21) A3,A4,A5,A6,N2,(B(K),C(K),K=1,7),A7
0008 2 FORMAT(4A1,1X,11,7(A7,A3),3X,A1)
0009 3 IF(A4-AI)1002,3,1002
0010 4 IF(A5-A2)1004,5,1004
0011 5 IF(A6-AI)1005,6,1005
C CHECK FOR VALID NUMBER OF FILE DEFINITIONS ON STATEMENT
C 0013 6 IF(FILAM=I1-N2+1)1006,7,7 $0900140
0014 7 IF(7-N2)1007,0,0 $0900150
0015 8 IF(N2-1)1008,05,85 $0900155
0016 85 U0 81 K=I1,J1
0017 FILAM(K)=B(K-I1+1)
0018 81 A(K)=C(K-I1+1) $0900158
0019 K1=I1+N2-1 $0900160
0020 DO 135 L1=11,K1 $0900170
0021 IF(L1-2)100,9,9 $0900180
0022 9 K2=L1-1 $0900190
C CHECK FOR DUPLICATE FILE NAME
C 0023 DO 10 L2=1,K2
0024 IF(FILAM(L1)-FILAM(L2))10,109,10
0025 10 CONTINUE
C RELATE FILE NAME TO UNIT NUMBER
C 0026 DO 11 L3=1,MCUU $0900230
0027 IF(A3(L1)-CUU(L3))11,12,11 $0900240
0028 11 CONTINUE $0900250
0029 GO TO 1011 $0900260
C
C SAVE POINTER TO UNIT NUMBER

C
0030  12 FILPNT(L1)=13
0031  1FILT-2)135,120,120
0032  120 K3=L1-1
0033  DO 13 L4=1,K3

C IS A FILE ALREADY ASSIGNED TO THIS UNIT NUMBER

C
0034  IF(FILPNT(L1)-FILPNT(L4))13,121,13
0035  121 L5=CUPNT(L3)

C YES. IS THE UNIT A DISK DEVICE

C
0036  IF(DEVCLS(L5)-D)121,135,1121
0037  13 CONTINUE

C YES. ARE THERE ANY MORE FILES ON THIS STATEMENT

C
0038  135 CONTINUE

C NO. CHECK FOR ADDITIONAL STATEMENTS OF THIS TYPE

C
0039  130 IF(A7-BLANK)15,14,15

C NO MORE STATEMENTS OF THIS TYPE. SET REMAINING PLACES IN FILE TABLE TO BLANK

C
0040  14 IF(I1+N2-MFILE)140,140,150
0041  140 J1=I1+N2
0042  GO TO 141 K1=J1,MFILE
0043  141 FILNAM(K1)=BLANK8
0044  150 RETURN
0045  15 I1=I1+N2
0046  IF(I1-MFILE)16,16,150
0047  16 J1=M100(I1+6,MFILE)
0048  GO TO 1

C *********************************************************************************************

C ERROR MESSAGES

C *********************************************************************************************

0049  2000 FORMAT(WH PROGRAM EXPECTED ,A1,11H IN COLUMN ,A1)  S0900270
0050  1002 CALL ERRPT (ERRORK)  S0900271
0051  WRITE (6,2000) A0,D1  S0900272
0052  WRITE (6,3002) A3,A4,A5,A6,N2,(F11K),C(K),K=1,7),A7  S0900273
0053  3002 FORMAT(1H ,A1,1X,11,7{A7,A3},3X,A1)  S0900274
0054  N2=0  S0900275
0055  GO TO 130  S0900276
0956 1003 CALL ERRPR1 (ERROR) 0957 WRITE (6,2000) A1,A2 0958 GO TO 2002 0959 1004 CALL ERRPR1 (ERROR) 0960 WRITE (6,2000) A2,A3 0961 GO TO 2002 0962 1005 CALL ERRPR1 (ERROR) 0963 WRITE (6,2000) D,D4 0964 GO TO 2002 0965 1006 CALL ERRPR1 (ERROR) 0966 IPI=I-1 0967 WRITE (6,2006) N2,IP1,MFILE 0968 2006 FORMAT(31H NUMBER OF FILES IN STATEMENT ,12,43H) PLUS NUMBER OF FILES PREVIOUSLY DEFINED ,12,27H) EXCEEDS PROGRAM MAXIMUM ,12,1H) 0969 2006 FORMAT(31H NUMBER OF FILES ,12,34H IN COLUMN 6 EXCEEDS MAXIMUM 0970 IF 7) 0971 GO TO 2002 0972 1007 CALL ERRPR1 (ERROR) 0973 WRITE (6,2007) N2 0974 1008 CALL ERRPR1 (ERROR) 0975 WRITE (6,2008) 0976 2008 FORMAT(31H NUMBER OF FILES IS LESS THAN 1) 0977 GO TO 2002 0978 1009 CALL ERRPR1 (ERROR) 0979 WRITE (6,2009) F.INAM(11) 0980 2009 FORMAT(21H DUPLICATE FILE NAME ,A7) 0981 GO TO 2002 0982 1011 CALL ERRPR1 (ERROR) 0983 WRITE (6,2011) A111 0984 2011 FORMAT(13H UNIT NUMBER ,A3,31H NOT DEFINED BY AN ID STATEMENT) 0985 GO TO 2002 0986 1121 CALL ERRPR1 (ERROR) 0987 WRITE (6,2121) A111 0988 2121 FORMAT(6H UNIT ,A3,27H ALREADY ASSIGNED TO A FILE) 0989 GO TO 2002 0990 END
### SCALAR MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLANK8</td>
<td>160</td>
<td>I1</td>
<td>168</td>
<td>J1</td>
<td>176</td>
<td>MFILE</td>
<td>178</td>
<td>A3</td>
<td>176</td>
<td>164</td>
<td>168</td>
</tr>
<tr>
<td>A4</td>
<td>168</td>
<td>A5</td>
<td>176</td>
<td>A6</td>
<td>188</td>
<td>N2</td>
<td>188</td>
<td>K</td>
<td>188</td>
<td>176</td>
<td>168</td>
</tr>
<tr>
<td>7</td>
<td>188</td>
<td>A0</td>
<td>196</td>
<td>A1</td>
<td>208</td>
<td>A2</td>
<td>216</td>
<td>0</td>
<td>216</td>
<td>188</td>
<td>176</td>
</tr>
<tr>
<td>K1</td>
<td>208</td>
<td>L1</td>
<td>216</td>
<td>L2</td>
<td>224</td>
<td>L3</td>
<td>224</td>
<td>2</td>
<td>224</td>
<td>208</td>
<td>216</td>
</tr>
<tr>
<td>MCUR</td>
<td>224</td>
<td>K3</td>
<td>232</td>
<td>L4</td>
<td>240</td>
<td>L5</td>
<td>240</td>
<td>2</td>
<td>240</td>
<td>224</td>
<td>232</td>
</tr>
<tr>
<td>ERROR</td>
<td>232</td>
<td>D1</td>
<td>240</td>
<td>D2</td>
<td>248</td>
<td>D3</td>
<td>248</td>
<td>2</td>
<td>248</td>
<td>232</td>
<td>240</td>
</tr>
<tr>
<td>1PI</td>
<td>240</td>
<td>1</td>
<td>248</td>
<td>2</td>
<td>256</td>
<td>2</td>
<td>256</td>
<td>2</td>
<td>256</td>
<td>240</td>
<td>248</td>
</tr>
</tbody>
</table>

### ARRAY MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILPNT</td>
<td>240</td>
<td>GUUPNT</td>
<td>244</td>
<td>CUU</td>
<td>248</td>
<td>DEVCLS</td>
<td>248</td>
<td>B</td>
<td>256</td>
</tr>
<tr>
<td>C</td>
<td>256</td>
<td>FILNAM</td>
<td>264</td>
<td>B</td>
<td>256</td>
<td>256</td>
<td>256</td>
<td>2</td>
<td>256</td>
</tr>
</tbody>
</table>

### SUBPROGRAMS CALLED

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1BCOM</td>
<td>264</td>
<td>ERRPRT</td>
<td>268</td>
<td>MINO</td>
<td>300</td>
<td>200</td>
<td>306</td>
</tr>
</tbody>
</table>

### FORMAT STATEMENT MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
</table>

**TOTAL MEMORY REQUIREMENTS 000052 BYTES**
SUBROUTINE FILE10(A,D,H,1,D,T,U,O,D2,BLANK,AO,A1,A2,NFILE,  
IFILNAM,FILPNT,CUPNTR,DEVCLS,DEVLSS,DEVSRH,VOLUME,RCDSIZ,BLKFCY,  
2MUL,LTNCY,SEEK,FACTOR,DOPNT,NAME,EIGHTY,ERROR,ERRORX,CLASS,  
3DEVICE,D3,D4,D5)  
C ********************************************************************  
C READ AND PROCESS J8IO AND J9IO STATEMENTS  
C ********************************************************************  
0002   REAL I,0,MULT,LTNCY $1000010  
0003   INTEGER FILPNT(20),CUPNTR(20),ERROR,ERRORX,DEVICE $1000020  
0004   DIMENSION DEVCLS(10),DEVLSS(10),DEVSRH(10),IFIXIT(16) $1000030  
0005   DOUBLE PRECISION FILNAM(20),NAME,BLANK,AR $1000035  
C READ AN IFIO, J8IO, OR J9IO STATEMENT  
C 0006   READ (5,1) A3,A4,A5,A6,A7,A8,VOLUME,RCDSIZ,BLKFCY,MULT,A9,LTNCY,  
1SEEK,FACTOR,EIGHTY $1000040  
0007   1 FORMAT(5A1,A7,F8.0,F5.6,F2.0,2A1,2F3.0,F4.3,40X,A1) $1000050  
0008   NAME=1 $1000060  
0009   IF(A3-A011001,2,1001 $1000070  
0010   2 IF(A4-A11002,6,1002 $1000080  
0011   6 IF(A5-A21006,7,1006 $1000090  
0012   7 IF(A6-T11008,10,1008 $1000100  
0013   8 IF(A7-D11008,10,1008 $1000110  
C CHECK FOR VALID FILE NAME  
C 0014   10 DD 11 11=1,NFILE $1000120  
0015   IF(FILNAM(11)-A9)11,12,11 $1000130  
0016   11 CONTINUE $1000140  
0017   GO TO 1011 $1000150  
C SAVE FILE NAME AND POINTERS TO UNIT NUMBER AND DEVICE TYPE AND MODEL NUMBER  
C 0018   12 NAME=2 $1000160  
0019   NAME=FILNAM(11) $1000170  
0020   FILNAM(11)=BLANK8 $1000180  
0021   DOPNT=FILPNT(11) $1000190  
0022   J1=CUPNTR(DOPNT) $1000200  
0023   DEVICE=J1 $1000210  
C CHECK FOR VALID DEVICE CLASS  
C 0024   IF(A9-A13,17,13 $1000220  
0025   13 IF(A9-D14,21,14 $1000230  
0026   21 IF(A9-H15,19,15 $1000240  
0027   15 IF(A9-T16,22,16 $1000250
0064 1006 CALL ERRPR1 (ERROR)
0065      WRITE (6,2000) A2,D3
0066      GO TO 2001
0067 1007 CALL ERRPR1 (ERROR)
0068      WRITE (6,2000) I,D4
0069      GO TO 2001
0070 1008 CALL ERRPR1 (ERROR)
0071      WRITE (6,2000) 0,D5
0072      GO TO 2001
0073 1011 CALL ERRPR1 (ERROR)
0074      WRITE (6,2011) A8,A3,A4,A5,D,A3,A6,A5,I,I0
0075 2011 FORMAT(6X FILE ,A7,1X,16HNOT DEFINED BY A1X,4A1,I1X,33HSTATEMENT OS101240
1R APPEARED IN PREVIOUS,1X,5A1,1X,9HSTATEMENT) S101285
0076      GO TO 2001
0077 1016 CALL ERRPR1 (ERROR)
0078      WRITE (6,2016)
0079 2016 FORMAT(51H DEVICE CLASS IN COLUMN 29 MUST BE A, D, H, T, OR U) S101320
0080      GO TO 2001
0081 1020 CALL ERRPR1 (ERROR)
0082      WRITE (6,2020) S101350
0083 2020 FORMAT(80H DEVICE CLASS OF THIS STATEMENT DOES NOT AGREE WITH DEV1S101369
ICE CLASS OF NO STATEMENT) S101370
0084      GO TO 2001' S101380
0085 1210 CALL ERRPR1 (ERROR)
0086      WRITE (6,2210) S101400
0087 2210 FORMAT(51H LATENCY TIME IN COLUMNS 30-32 MUST BE NON-NEGATIVE) S101410
0088      GO TO 2001
0089 1212 CALL ERRPR1 (ERROR)
0090      WRITE (6,2112) S101440
0091 2112 FORMAT(48H SEEK TIME IN COLUMNS 33-35 MUST BE NON-NEGATIVE) S101450
0092      GO TO 2001 S101460
0093 END S109999
### SCALAR MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>A3</td>
<td>180</td>
</tr>
<tr>
<td>A5</td>
<td>100</td>
</tr>
<tr>
<td>BLKFCI</td>
<td>1E4</td>
</tr>
<tr>
<td>FACTOR</td>
<td>1F8</td>
</tr>
<tr>
<td>A2</td>
<td>20C</td>
</tr>
<tr>
<td>IOPNTI</td>
<td>220</td>
</tr>
<tr>
<td>H</td>
<td>234</td>
</tr>
<tr>
<td>D1</td>
<td>248</td>
</tr>
<tr>
<td>D5</td>
<td>25C</td>
</tr>
</tbody>
</table>

### SYMBOL LOCATION MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>188</td>
</tr>
<tr>
<td>BLANK8</td>
<td>1B8</td>
</tr>
<tr>
<td>A7</td>
<td>1D8</td>
</tr>
<tr>
<td>INAME</td>
<td>200</td>
</tr>
<tr>
<td>0</td>
<td>210</td>
</tr>
<tr>
<td>DEVICE</td>
<td>224</td>
</tr>
<tr>
<td>U</td>
<td>238</td>
</tr>
<tr>
<td>ERRORX</td>
<td>24C</td>
</tr>
</tbody>
</table>

### ARRAY MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILPNT</td>
<td>260</td>
</tr>
<tr>
<td>IFIXIT</td>
<td>274</td>
</tr>
</tbody>
</table>

### SUBPROGRAMS CALLED

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCCM</td>
<td>290</td>
</tr>
<tr>
<td>ERRRT</td>
<td>294</td>
</tr>
</tbody>
</table>

### FORMAT STATEMENT MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2C0</td>
</tr>
<tr>
<td>2020</td>
<td>38C</td>
</tr>
</tbody>
</table>
SUBROUTINE TOD1(WEEK, DON, HON, MON, DOFF, HOFF, HCON, MMON, MHOF, TMOFF, TCASE, DAY, ERROR, ERRORD)

INTEGER DON, HON, DOFF, HOFF, HCON, MMON, MHOF, S1100020
DIMENSION MMON(7), MHOF(7)
TCASE=1 S1100026

C CHECK BEGIN AND END TIMES FOR PROGRAM

C

C CHECK TIMES FOR VALIDITY AND CONSISTENCY AND DETERMINE PROGRAM TYPE

C

C WEKY/L PROGRAM

C
C
C NOT FULL WORKING DAY PROGRAM
C
0028 14 TCase=MAX0(TCase,2)  $1100180
0029 15 IF(MDaFF-MMOFF(DOFF))155,16,1005 $1100190
0030 16 IF(MDaFF-MMOFF(DOFF))155,18,1007 $1100200
0031 155 IF(MDaFF-MMOFF(DOFF))1005,165,17 $1100203
0032 165 IF(MDaFF-MMOFF(DOFF))1007,1007,17 $1100206
C
C NOT FULL WORKING DAY PROGRAM
C
0033 17 TCase=MAX0(TCase,2)  $1100210
0034 18 RETURN  $1100220
C
C ERROR MESSAGES
C
C
0035 1000 CALL ERRPRRT (ERROR)  $11110010
0036 WRITE (6,2000)  $11110020
0037 2000 FORMAT(18H DAY ON IS INVALID)  $11110030
0038 2001 ERDORx=ERRORx+1  $11110040
0039  RETURN  $11110050
0040 1002 CALL ERRPRRT (ERROR)  $11110060
0041 WRITE (6,2002)  $11110070
0042 2002 FORMAT(19H DAY OFF IS INVALID)  $11110080
0043 GO TO 2001  $11110090
0044 1004 CALL ERRPRRT (ERROR)  $11110100
0045 WRITE (6,2004)  $11110110
0046 2004 FORMAT(19H HOUR ON IS INVALID)  $11110120
0047 GO TO 2001  $11110130
0048 1005 CALL ERRPRRT (ERROR)  $11110140
0049 WRITE (6,2005)  $11110150
0050 2005 FORMAT(20H HOUR OFF IS INVALID)  $11110160
0051 GO TO 2001  $11110170
0052 1006 CALL ERRPRRT (ERROR)  $11110180
0053 WRITE (6,2006)  $11110190
0054 2006 FORMAT(21H MINUTE ON IS INVALID)  $11110200
0055 GO TO 2001  $11110210
0056 1007 CALL ERRPRRT (ERROR)  $11110220
0057 WRITE (6,2007)  $11110230
0058 2007 FORMAT(22H MINUTE OFF IS INVALID)  $11110240
0059 GO TO 2001  $11110250
0060 1008 CALL ERRPRRT (ERROR)  $11110260
0061 WRITE (6,2008)  $11110270
0062 2008 FORMAT(23H TIME OFF IS LESS THAN OR EQUAL TO TIME ON)  $11110280
0063 GO TO 2001  $11110290
0064 1050 CALL ERRPRRT (ERROR)  $11110300
0065    WRITE (6,2050)  S1110310
0066    2050 FORMAT(43H DAY ON DOES NOT AGREE WITH DAY IN COLUMN 2) S1110320
0067    GO TO 2001   S1110330
0068    1051 CALL ERPRPT (ERROR) S1110340
0069    WRITE (6,2051) S1110350
0070    2051 FORMAT(44H DAY OFF DOES NOT AGREE WITH DAY IN COLUMN 2) S1110360
0071    GO TO 2001   S1110370
0072    END          S1199999
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCASE</td>
<td>1CB</td>
<td>DAY</td>
<td>1CC</td>
<td>DDN</td>
<td>1DG</td>
<td>DOFF</td>
<td>1DG</td>
<td>WEEK</td>
<td>1DA</td>
</tr>
<tr>
<td>HON</td>
<td>1GC</td>
<td>HOFF</td>
<td>1EO</td>
<td>MON</td>
<td>1E4</td>
<td>MOFF</td>
<td>1E8</td>
<td>ERROR</td>
<td>1EC</td>
</tr>
<tr>
<td>ERRORX</td>
<td>1FD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ARRAY MAP**

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMON</td>
<td>1FC</td>
<td>HHOFF</td>
<td>1F8</td>
<td>MMON</td>
<td>1FC</td>
<td>MMOFF</td>
<td>200</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SUBPROGRAMS CALLED**

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRKPR</td>
<td>204</td>
<td>IBCOM=</td>
<td>208</td>
<td>MAXO</td>
<td>20C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FORMAT STATEMENT MAP**

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007</td>
<td>289</td>
<td>2008</td>
<td>203</td>
<td>2050</td>
<td>301</td>
<td>2051</td>
<td>330</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TOTAL MEMORY REQUIREMENTS 000942 BYTES
SUBROUTINE SWITCH(J0BNM1, J0BNM2, STPNM1, STPNM2, DON1, DON2, HON1, HON2, DOFF1, DOFF2, HOFF1, HOFF2)
1.0BNM2, STPNM2, DON2, HON2, MON2, DOFF2, HOFF2, HOFF2)  S1200010
C******************************************************************************
C MOVE PROGRAM DESCRIPTIONS TO FIRST PLACE IN TABLE FOR CURRENT PROGRAM TYPE
C******************************************************************************
0002 DOUBLE PRECISION J0BNM1, J0BNM2, STPNM1, STPNM2  S1200030
0003 INTEGER DON1, DON2, HON1, HON2, DOFF1, DOFF2, HOFF1, HOFF2  S1200040
0004 J0BNM2=J0BNM1  S1200050
0005 STPNM2=STPNM1  S1200060
0006 DON2=DON1  S1200070
0007 HON2=HON1  S1200080
0008 MON2=MON1  S1200090
0009 DOFF2=DOFF1  S1200100
0010 HOFF2=HOFF1  S1200110
0011 HOFF2=HOFF1  S1200120
0012 RETURN
0013 END
### SCALAR MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOBNM2</td>
<td>0</td>
<td>JOBNM1</td>
<td>96</td>
</tr>
<tr>
<td>DON1</td>
<td>64</td>
<td>HON2</td>
<td>88</td>
</tr>
<tr>
<td>DOFF2</td>
<td>C8</td>
<td>DOFF1</td>
<td>CC</td>
</tr>
<tr>
<td>MOFF1</td>
<td>0C</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TOTAL MEMORY REQUIREMENTS 000286 BYTES**
SUBROUTINE JOHNSN(A,R,N,NI)

C *******************************************************************************
C
C SOLVE TWO-MACHINE TWO-OPERATION SEQUENCING PROBLEM
C
C *******************************************************************************

D001

DIMENSION A(5),B(5),N(5)
I=1
K=1
L=IN

C WHICH IS SMALLER -- FIRST OPERATION OR SECOND OPERATION

D006

5 IF(A(I)-B(I))>20,20,10

C SECOND OPERATION TIME. COMPARE THIS TIME WITH THE OTHER JOBS' TIMES

D007

10 J=I
I=I+1

C IS THERE ANOTHER JOB WHICH HAS NOT BEEN SEQUENCED

D009

11 DO 15 IY=I,IN
IY=I
15 CONTINUE

D010

16 CONTINUE

D011

DO 12 IZ=1,IN
12 IF(I-N(IZ))>16,15,16

D012

C YES. COMPARE OPERATION TIMES WITH THAT JOB

D014

GO TO 30

D015

15 CONTINUE

D016

C NO. PLACE THIS JOB IN THE SEQUENCE

D017

GO TO 75

D018

C IS EITHER OF THE SECOND JOB'S TIMES SMALLER THAN THE FIRST JOB'S TIME

D019

30 IF(A(I)-B(I))>35,35,36
35 IF(A(I)-B(I))>60,60,70
36 IF(B(I))>35,35,36

D020

C YES. SECOND OPERATION TIME IS SMALLER. USE THIS TIME TO COMPARE WITH

50 J=I

C ANY MORE JOBS WHICH HAVE NOT BEEN SEQUENCED
C 70 I=I+1
0022 IF(I=IN)11,11,75

C NO. SEQUENCE THIS JOB BY PLACING IT IN THE LAST UNFILLED SPACE
C
0024 75 H(L)=J
0025 L=L-1
0026 GO TO 90

C FIRST OPERATION IS SMALLER. COMPARE THIS TIME WITH THE OTHER JOBS' TIMES
C
0028 20 J=1
0029 I=I+1
0030
C IS THERE ANOTHER JOB WHICH HAS NOT BEEN SEQUENCED
C
0032 21 DO 25 IY=1,IN
0033 I=IY
0034 DO 26 IZ=1,IN
0035 IF(I=IN(IZ))26,25,26

C YES. COMPARE OPERATION TIMES WITH THAT JOB
C
0036 26 CONTINUE
0037 GO TO 40
0038 25 CONTINUE
0039
C NO. PLACE THIS JOB IN THE SEQUENCE
C
0040 85 CONTINUE
0041
C IS EITHER OF THE SECOND JOB'S TIMES SMALLER THAN THE FIRST JOB'S TIME
C
0042 40 IF(A(I)-B(I))45,45,46
0043 45 IF(A(J)-A(I))80,80,60
0044 46 IF(A(J)-B(I))80,80,50

C YES. FIRST OPERATION TIME IS SMALLER. USE THIS TIME TO COMPARE WITH
C
0045 60 J=I
0046
C ANY MORE JOBS WHICH HAVE NOT BEEN SEQUENCED
C
0047 80 I=I+1
0048 IF(I=IN)21,21,85
0049
C NO. SEQUENCE THIS JOB BY PLACING IT IN THE FIRST UNFILLED SPACE
C 85 N(K)=J
  K=K+1
C FIND THE FIRST JOB WHICH HAS NOT BEEN SEQUENCED
C
  90 DO 95 IY=1,IN
    I=IY
  95 CONTINUE
  DO 96 IZ=1,IN
C IF(I-N(IZ))<0,96,95,96
  IF(I-N(IZ))=0,96,95
  96 CONTINUE
  GO TO 98
C
  95 CONTINUE
C IS IT THE ONLY JOB WHICH HAS NOT BEEN SEQUENCED
C
  98 IF(K+1-L)<5,99
C YES. PLACE THAT JOB IN THE ONLY REMAINING SPACE IN THE TABLE
C
  N(K)=I
  RETURN
  END
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>173</td>
</tr>
<tr>
<td>IY</td>
<td>186</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>K</td>
<td>176</td>
</tr>
<tr>
<td>IZ</td>
<td>189</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>180</td>
</tr>
<tr>
<td>IN</td>
<td>184</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>J</td>
<td>188</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>196</td>
</tr>
<tr>
<td>B</td>
<td>198</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>19C</td>
</tr>
</tbody>
</table>

TOTAL MEMORY REQUIREMENTS 0005A6 BYTES
SUBROUTINE IGNSCHIA,B,C,DX,DP,DPX,L,N,IN,CASE,ERROR,EPROMX
C *********************************************************************
C SOLVE THREE-MACHINE THREE-OPERATION SEQUENCING PROBLEM
C *********************************************************************
INTEGER CASE,ERROR,EPROMX,PL(5),POPER(120),POPERR,POPRD
DIMENSION BOUND(120),LIST(120),TIMEA(120),TIMEB(120),TIMEC(120)
NLIST=120

ARE THERE PARALLEL OPERATIONS ON INPUT
GO TO (1,1,103,1,105,1,1,1,CASE)

YES. SAME OPERATION, SAME MACHINE. FIND OPTIMAL INPUT SEQUENCE
103 CALL PARALL(D,DP,A,B,L,IN)
GO TO 1

YES. SAME OPERATION, DIFFERENT MACHINES. FIND OPTIMAL INPUT SEQUENCE
105 CALL PARALL(D,DX,DPX,A,B,L,IN)

INITIALIZE SUBROUTINE
1 JN=IN-1
NEXT=1
LIST(1)=0
BOUND(1)=0.
TIMEA(1)=0.
TIMEB(1)=0.
TIMEC(1)=0.
DO 100 I=1,NLIST
100 POPER(1)=0
POPERR=0
DO 101 I=1,IN
101 N(I)=0

DETERMINE THE JOB SEQUENCE FOR THE FIRST NODE ON THE LIST
2 DO 3 I=1,IN
3 M(I)=1
DO TO (160,40),NEXT
4 DO 4 I=1,IN
4 LIST(I)/10**((JN-1)
J=JN-1
FORTRAN IV G LEVEL 0, MOD 0

IGNSCH

DATE = 67249   06/23/31   PAGE 0002

0029    DO 5 I=1,J
0030     K=JN-I
0031     5 N(K+1)=MOD(N(K+1),10*N(K))

C ARE THERE PARALLEL OPERATIONS ON OUTPUT
C
0032    45 GO TO (60,60,60,50,60,50,60,01,CASE
C
0033    C YES, SAME OPERATION, SAVE OPTIMAL OUTPUT SEQUENCE
C
0034    50 DO 52 I=1,IN
0035    52 L(I)=POPER(I)/10**(IN-I))
0036    K=IN-I
0037    53 L(K+1)=MOD(L(K+1),10*L(K))
0038    GO TO (62,62,62,62),CASE
0039    DO 61 I=1,IN
0040     II=L(I)
0041    GO TO (58,59),II
0042    58 B(II)=D(II)
0043     C(II)=DPX(II)
0044    GO TO 61
0045    59 B(I)=DP(I)
0046     C(I)=DX(I)
0047    61 CONTINUE
0048    GO TO 60
0049    62 DO 65 I=1,IN
0050     II=L(I)
0051    GO TO (63,64),II
0052    63 B(II)=D(II)
0053     C(II)=DP(II)
0054    GO TO 65
0055    64 B(I)=DP(I)
0056     C(I)=DX(I)
0057    65 CONTINUE

C HAS THIS NODE SCHEDULED ALL JOBS
C
0058    60 DO 7 I=1,JN
0059     IF(IN(I)-1110,6,6
0060     6 J=II)
0061     M(J)-0
0062    7 CONTINUE

C YES, OPTIMAL SEQUENCE HAS BEEN FOUND
C
0063    DO 8 I=1,IN
0064     IF(M(I)-1110,9,9

$1400170
$1400180
$1400190
$1400191
$1400194
$1400195
$1400197
$1400198
$1400199
$1400209
$1400210
$1400211
$1400212
$1400213
$1400214
$1400215
$1400216
$1400217
$1400218
$1400219
$1400220
$1400221
$1400222
$1400223
$1400224
$1400225
$1400226
$1400227
$1400228
$1400229
$1400230
$1400231
$1400232
$1400240
$1400250
$1400260
8 CONTINUE
9 N(IN)=1
RETURN
C
C OPTIMAL SEQUENCE HAS NOT BEEN FOUND. REMOVE FIRST NODE FROM LIST
C
10 LIST=LIST(1)
TIMEA=TIMEA(1)
TIMEB=TIMEB(1)
TIMEC=TIMEC(1)
POP=POP(1)
GO TO (13,11),NEXT
11 J=NEXT-2
C
C MOVE ALL NODES UP ONE POSITION IN LIST
C
DO 12 K=1,J
LIST(K)=LIST(K+1)
BOUND(K)=BOUND(K+1)
TIMEA(K)=TIMEA(K+1)
TIMEB(K)=TIMEB(K+1)
TIMEC(K)=TIMEC(K+1)
POP=POP(K+1)
NEXT=NEXT+1
13 DO 21 J=1,N
C
C IS THE LIST FILLED
C
IF(NEXT-1).LT.131,131,22
C
C FIND A JOB WHICH THE JUST REMOVED NODE HAS NOT SCHEDULED
C
131 IF(M(J).LT.121,14,14
C
C CREATE A NODE FOR THE SEQUENCE CONTAINING THE ADDITIONAL JOB
C
14 LIST=LIST+(J+100*(JN-1))
C
C COMPUTE TIMEA, TIMEB, AND TIMEC FOR THE SEQUENCE CONTAINING THE ADDITIONAL JOB
C
TIMEA=TIMEA+A(J)
ASUM=ASUM+A(J)
CSUM=CSUM+A(J)
C
C WHICH THREE-OPERATION CASE IS THIS
C
GO TO (141,142,141,134,141,134,147),CASE
C PARALLEL OPERATIONS ON OUTPUT. SAME OPERATION

```plaintext
0092 134 IF ITIMEB0-TIMEA0)135, 133, 132
0093 132 IF ITIMECU-TIMEA0)137, 148, 148
0094 133 IF ITIMECO-TIMEA0)136, 148, 148
0095 135 IF ITIMECO-TIMEA0)136, 136, 137
0096 136 II=1
0097 GO TO 138
0098 137 PDX(I)=AMAX1(G.,TIMEBO-TIMEA0)
0099 PDX(1)=AMAX1(I.,TIMECO-TIMEA0)
0100 POP(I)=AMAX1(G.,TIMEBO-TIMEA0)
0101 POPX(I)=AMAX1(I.,TIMECO-TIMEA0)
0102 II=2
0103 138 J1=1
0104 IFN=IN-1+1
0105 DO 140 K=1, IN
0106 IF (M(K)=140, 139, 139
0107 139 PDX(I)=D(K)
0108 PDX(I)=DX(K)
0109 POP(I)=DP(K)
0110 POPX(I)=DPX(K)
0111 II=II+1
0112 140 CONTINUE
0113 GO TO (143, 143, 143, 143), CASE
0114 CALL PARAL2 (PD, PD0, PD0, PA, PB, PL, IPN)
0115 GO TO 144
0116 143 CALL PARAL1 (PD, PD0, PA, PB, PL, IPN)
0117 144 DO 146 K=1, IN
0118 IF (M(K)=146, 145, 145
0119 B(K)=PA(J1)
0120 C(K)=PB(J1)
0121 L(K)=PL(J1)
0122 J1=J1+1
0123 146 CONTINUE
0124 POPERR = 0
0125 GO TO 149
0126 149 POPERR=POPERR+L(I1)*(10***(IN-I1))
0127 GO TO 142
0128 148 POPERR=POPAPP
0129 GO TO 142

C PARALLEL OPERATIONS ON INPUT
C
0130 141 TIMEBR=TIMEBO+B(J)
0131 TIMECR=AMAX1(TIMEBR, TIMEBO, TIMECO)+C(J)
0132 GO TO 150
```

C PARALLEL OPERATIONS ON OUTPUT
C
0131     142 TIMEBR=AMAX1(TIMEBR,TIMEBO)+B(J)  $1400520
0134     TIMECR=AMAX1(TIMECR,TIMECO)+C(J)  $1400530
0135     BMIN=10.**74  $1400531
0136     GO TO 150  $1400532
C
C NO PARALLEL OPERATIONS
C
0137     147 TIMEBR=AMAX1(TIMEBR,TIMEBO)+B(J)  $1400533
0138     TIMECR=AMAX1(TIMECR,TIMECO)+C(J)  $1400534
0139     BCNIN=10.**74  $1400535
0140     150 CMIN=10.**74  $1400536
0141     DO 17 K=1,IN  $1400537
0142     15 IF(M(K)-1)17,15,15  $1400538
0143     16 IF(K-J)16,17,16  $1400539
0144     17 ASUM=ASUM+A(K)  $1400540
0145     HSUM=HSM+B(K)  $1400541
0146     CSUM=CSUM+C(K)  $1400542
C
C WHICH THREE-OPERATION CASE IS THIS
C
0147     GO TO (161,162,161,162,161,162,161,167),CASE  $1400543
C
C PARALLEL OPERATIONS ON OUTPUT
C
0148     162 IF(B(K)-C(K))161,163,164  $1400544
0149     163 BMN=AMIN1(BMIN,B(K))  $1400545
0150     GO TO 161  $1400546
0151     164 BMN=AMIN1(BMIN,B(K))  $1400547
0152     GO TO 17  $1400548
C
C NO PARALLEL OPERATIONS
C
0153     167 BCNIN=AMIN1(BCMIN,B(K))+C(K)  $1400549
C
C PARALLEL OPERATIONS ON INPUT
C
0154     161 CMIN=AMIN1(CMIN,C(K))  $1400550
0155     17 CONTINUE  $1400551
0156     TIMECT=TIMECR+CSUM  $1400552
C
C WHICH THREE-OPERATION CASE IS THIS
C
0157     GO TO (171,172,171,172,171,172,177),CASE  $1400553
C
C PARALLEL OPERATIONS ON INPUT
C
171 TIMEA=TIMEA+ASUM+GMIN
   TIMEB=TIMEB+BSUM+GMIN
   GO TO 178
C  PARALLEL OPERATIONS ON OUTPUT
C 172 TIMEA=TIMEA+ASUM+AMIN1(MIN,GMIN)
   TIMEB=TIMEB+BSUM
   GO TO 178
C  NO PARALLEL OPERATIONS
C 177 TIMEA=TIMEA+ASUM+BCMIN
   TIMEB=TIMEB+BSUM+GMIN
C  COMPUTE THE LOWER BOUND FOR THE SEQUENCE CONTAINING THE ADDITIONAL JOB
C 178 BOUNDR=AMAX1(TIMEA,TIMEB,TIMEC)
   GO TO (181,179),179
   179 K=NEXT-1
C  FIND THE PROPER PLACE IN THE LIST FOR THE NEW NODE
C  DD 18 J1=1,K
   IF(BOUNDR-BOUND(J1))19,19,18
   19 CONTINUE
C  THE NEW NODE'S LOWER BOUND IS LARGER THAN ALL OTHER NODES' LOWER BOUNDS.
C  PLACE THE NEW NODE IN THE NEXT FREE POSITION IN THE LIST
C 181 LIST(NEXT)=LISTR
   BOUND(NEXT)=BOUNDR
   TIMEA(NEXT)=TIMEA
   TIMEB(NEXT)=TIMEB
   TIMEC(NEXT)=TIMEC
   POPER(NEXT)=POPER
   NEXT=NEXT+1
   GO TO 21
C  THE NEW NODE'S LOWER BOUND IS EQUAL TO OR LESS THAN SOME OTHER NODE'S LOWER
C  BOUND. MOVE DOWN THE LIST ALL NODES WITH LOWER BOUNDS GREATER THAN OR EQUAL
C  TO THAT OF THE NEW NODE
C 19 K=NEXT-J1
   DO 20 II=1,K
   20 LIST(NEXT+II-1)=LIST(NEXT-1)
   BOUND(NEXT+II-1)=BOUND(NEXT-1)
   TIMEA(NEXT+II-1)=TIMEA(NEXT-1)
0185     TIMEB(NEXT+1-I1)=TIMEB(NEXT-I1)  $1400840
0186     TIMEC(NEXT+1-I1)=TIMEC(NEXT-I1)  $1400850
0187     20 POPERNEXT+1-I1)=POPERNEXT-I1)  $1400855
C
C INSERT THE NEW NODE IN THE LIST
C
0188     LIST(I1)=LISTR  $1400860
0189     BOUND(I1)=BOUNDR  $1400870
0190     TIMEA(I1)=TIMEAR  $1400880
0191     TIMEB(I1)=TIMEBR  $1400890
0192     TIMEC(I1)=TIMECR  $1400900
0193     POPER(I1)=POPCR  $1400910
0194     NEXT=NEXT+1  $1400920
0195     21 CONTINUE  $1400930
0196     GO TO 2  $1400940
C
C LIST FULL. ALGORITHM CANNOT CONTINUE DUE TO LACK OF STORAGE
C
0197     22 DO 30 I=1,NLIST  $1400949
C
C DETERMINE THE JOB SEQUENCE FOR A NODE
C
0198     DO 23 J=I,J1  $1400950
0199     23 N(J)=LIST(I)/10**(JN-J1)  $1400960
0200     J=JN+1  $1400970
0201     DO 24 I1=1,J  $1400980
0202     K=JN-I1  $1400990
0203     24 N(K+1)=MOD(N(K+1),10*N(K))  $1401000
C
C HAS THIS NODE SCHEDULED ALL JOBS
C
0204     DO 26 K=1,JN  $1401010
0205     IF(N(K)-1130,25,25  $1401020
0206     25 J=N(K)  $1401030
0207     M(J)=0  $1401040
0208     26 CONTINUE  $1401050
C
C YES. IS THIS NODE THE FIRST NODE ON THE LIST
C
0209     IX=2  $1401060
0210     GO TO (45,27),I  $1401070
C
C NO. THE SEQUENCE MAY NOT BE OPTIMAL
C
0211     27 GO TO 1038  $1401080
0212     30 CONTINUE  $1401090
C
C NO NODE HAS SCHEDULED ALL THE JOBS. AN OPTIMAL SEQUENCE COULD NOT BE FOUND
C
IX=1
GO TO 1030
C
C ERROR MESSAGES
C
C
1030 CALL EPRPRT (ERROR)
WRITE (6,2030)
WRITE (6,2031)
2031 FORMAT (1H NO NODE HAS SCHEDULED ALL THE DAILY JOBS)
1032 WRITE (6,2032) NLST
2032 FORMAT (40H NUMBER OF NODES AVAILABLE TO ALGORITHM=,16)
2033 FORMAT (24H NUMBER OF JOBS TO BE SCHEDULED=,12)
AI=1.5*IN*IN-1]*1
WRITE (6,2034) I
2034 FORMAT (34H MINIMUM NUMBER OF NODES REQUIRED=,14)
DO 2035 I=2,IN
2035 K=K+1
WRITE (6,2036) K
2036 FORMAT (34H MAXIMUM NUMBER OF NODES REQUIRED=,17)
ERRORX=ERRORX+1
GO TO (1037,45),IX
2037 RETURN
1038 CALL EPRPRT (ERROR)
WRITE (6,2030)
WRITE (6,2038) I
2038 FORMAT (1H NO. IX,16,52H IS THE FIRST NODE THAT HAS SCHEDULED ALL
10DAILY JOBS)
GO TO 1032
END
### SCALAR MAP

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLIST</td>
<td>304</td>
<td>CASE</td>
<td>308</td>
<td>IN</td>
<td>30C</td>
<td>JN</td>
<td>310</td>
<td>NEXT</td>
<td>314</td>
</tr>
<tr>
<td>LISTS</td>
<td>32C</td>
<td>TIMEAD</td>
<td>330</td>
<td>TIMEBD</td>
<td>334</td>
<td>TIMECO</td>
<td>338</td>
<td>POPERO</td>
<td>33C</td>
</tr>
<tr>
<td>LISTR</td>
<td>340</td>
<td>TIMEAR</td>
<td>344</td>
<td>SUM</td>
<td>348</td>
<td>BSUM</td>
<td>34C</td>
<td>CSUM</td>
<td>350</td>
</tr>
<tr>
<td>J1</td>
<td>354</td>
<td>IPN</td>
<td>358</td>
<td>TIMEBR</td>
<td>35C</td>
<td>TIMECR</td>
<td>360</td>
<td>BMIN</td>
<td>364</td>
</tr>
<tr>
<td>MCKIN</td>
<td>368</td>
<td>CMIN</td>
<td>36C</td>
<td>TIMECT</td>
<td>370</td>
<td>TIMEAT</td>
<td>374</td>
<td>TMEBT</td>
<td>378</td>
</tr>
<tr>
<td>BOUNDR</td>
<td>37C</td>
<td>I1</td>
<td>380</td>
<td>ERROR</td>
<td>384</td>
<td>AI</td>
<td>388</td>
<td>ERRORMX</td>
<td>39C</td>
</tr>
</tbody>
</table>

### ARRAY MAP

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>PL</td>
<td>390</td>
<td>POPER</td>
<td>3A4</td>
<td>A</td>
<td>584</td>
<td>B</td>
<td>588</td>
<td>C</td>
<td>5AC</td>
</tr>
<tr>
<td>D</td>
<td>590</td>
<td>DX</td>
<td>594</td>
<td>DP</td>
<td>598</td>
<td>DPX</td>
<td>59C</td>
<td>L</td>
<td>5A0</td>
</tr>
<tr>
<td>TIMEB</td>
<td>65C</td>
<td>TIMEC</td>
<td>D3C</td>
<td>PA</td>
<td>F1C</td>
<td>PB</td>
<td>F30</td>
<td>PD</td>
<td>F44</td>
</tr>
<tr>
<td>PDX</td>
<td>F58</td>
<td>PDP</td>
<td>F6C</td>
<td>PDPX</td>
<td>F80</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### SUBPROGRAMS CALLED

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>F9A</td>
<td>2031</td>
<td>F9A</td>
<td>2032</td>
<td>FRXPI</td>
<td>F80</td>
<td>ERRXPI</td>
<td>FA0</td>
<td>ERRXPR</td>
<td>FA4</td>
</tr>
<tr>
<td>AMAX1</td>
<td>FA8</td>
<td>AMAX1</td>
<td>FAC</td>
<td>ANINI</td>
<td>F00</td>
<td>AMAX1</td>
<td>ANINI</td>
<td>FAC</td>
<td>F00</td>
</tr>
</tbody>
</table>

### FORMAT STATEMENT MAP

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
<th>Symbol</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>2030</td>
<td>1104</td>
<td>2031</td>
<td>1150</td>
<td>2032</td>
<td>118A</td>
<td>2033</td>
<td>1188</td>
<td>2034</td>
<td>11DE</td>
</tr>
<tr>
<td>2036</td>
<td>1206</td>
<td>2038</td>
<td>122E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TOTAL MEMORY REQUIREMENTS 00288A BYTES**
SUBROUTINE PARAL(A,B,AX,BX,L,IN)

C SOLVE PARALLEL OPERATION SEQUENCING PROBLEM FOR SAME OPERATION, SAME MACHINES

C ********************************************************************

0002 DIMENSION A(5),B(5),AX(5),BX(5),L(5),LMAX(5),LMIN(5)

0003 JN=IN-1

0004 DO 3 I=1,IN

0005 IF(A(I)-B(I))1,1,2

0006 1 LMAX(I)=2

0007 LMIN(I)=1

0008 GO TO 3

0009 2 LMAX(I)=1

0010 LMIN(I)=2

0011 3 CONTINUE

C CREATE A NODE FOR EACH TREE

C TIMEA=A(I)

C WHICH MACHINE HAS THE LARGER OPERATION TIME FOR THE FIRST JOB

C GO TO (4,6),I1

C MACHINE A. ASSIGN SMALLER OPERATION TIMES OF REMAINING JOBS TO MACHINE A

C 4 DO 5 I=2,IN

C TIMEA=TIMEA+AMIN(A(I),B(I))

C 5 L(I)=LMIN(I)

C MACHINE B. ASSIGN SMALLER OPERATION TIMES OF REMAINING JOBS TO MACHINE B

C 6 DO 7 I=2,IN

C TIMEB=TIMEB+AMAX(A(I),B(I))

C 7 L(I)=LMAX(I)

C COMPUTE THE LOWER BOUND FOR THE SEQUENCES CREATED

C 8 BOUNDO=AMAX1(TIMEA,TIMEB)
DO 16 I=2,IN
C EXTEND THE TREES BY ADDING BOTH POSSIBLE OPERATIONS TO BOTH TREES
C
0030 TIMEA=TIMEA+AMIN1(A(I),B(I))
0031 TIMEB=TIMEB+AMAX1(A(I),B(I))
0032 TIMEA=TIMEA+AMAX1(A(I),B(I))
0033 TIMEB=TIMEB+AMAX1(A(I),B(I))
0034 TIMEA=TIMEA
0035 TIMEB=TIMEB
0036 TIMEA=TIMEA
0037 TIMEB=TIMEB
0038 IX=IX+1
0039 DO 9 J=IX,IN
0040 TIMEA=TIMEA+AMIN1(A(IJ),B(IJ))
0041 TIMEB=TIMEB+AMAX1(A(IJ),B(IJ))
0042 TIMEA=TIMEA+AMAX1(A(IJ),B(IJ))
0043 TIMEB=TIMEB+AMIN1(A(IJ),B(IJ))
C
C COMPUTE THE LOWER BOUNDS FOR THE NEW SEQUENCES
C
0044 BOUNDR=AMAX1(TIMEA,TIMEB)
0045 BOUNDS=AMAX1(TIMEA,TIMEB)
C
C THE SMALLER OF THESE TWO LOWER BOUNDS IS THE NEW LOWER BOUND
C
0046 IF(BOUNDR-BOUNDS).LT.10,10,13
C
C IS THE CURRENT LOWER BOUND SMALLER THAN THE NEW LOWER BOUND
C
0047 10 IF(BOUNDR-BOUNDS).LT.11,11,17
C
C NO. USE THE NEW LOWER BOUND TO CONTINUE THE ALGORITHM
C
0048 11 BOUND0=BOUNDR
0049 L(I)=LMAX(I)
0050 DO 12 J=IX,IN
0051 12 L(IJ)=LMIN(IJ)
0052 GO TO 16
C
C IS THE CURRENT LOWER BOUND SMALLER THAN THE NEW LOWER BOUND
C
0053 13 IF(BOUNDR-BOUNDS).LT.14,14,17
C
C NO. USE THE NEW LOWER BOUND TO CONTINUE THE ALGORITHM
C
0054 14 BOUND0=BOUNDS
0055 L(I)=LMIN(I)
S1500290
S1500300
S1500310
S1500320
S1500330
S1500340
S1500350
S1500360
S1500370
S1500380
S1500390
S1500400
S1500410
S1500420
S1500430
S1500440
S1500450
S1500460
S1500470
S1500480
S1500490
S1500500
S1500510
S1500520
S1500530
S1500540
S1500550
DO 15 J=1X,IN
15 L(J)=LMAX(J)
TIMEAX=TIMEAY
TIMEBX=TIMEBY
16 CONTINUE
C THE CURRENT LOWER BOUND IS SMALLER THAN ANY OTHER LOWER BOUND
C AN OPTIMAL SEQUENCE HAS BEEN FOUND
C
17 DO 20 I=1,IN
   II=II
   GO TO (18,19),II
18 AX(II)=A(II)
   BX(II)=B(II)
   GO TO 20
19 AX(II)=B(II)
   BX(II)=A(II)
20 CONTINUE
RETURN
END
SUBROUTINE PARAL2(A, B, C, D, AX, BX, L, IN)
C******************************************************************************
C
C SOLVE PARALLEL OPERATION SEQUENCING PROBLEM FOR SAME OPERS. DIFFERENT MACHINES
C******************************************************************************
0003   LMIN(5)
0004   JN=IN-1
0005   DO 19 K=1,2
0006    DO 3 I=1,IN
0007   IF(A(I)-C(I))1,1,2
0008   1 LMAX(I)=2
0009    LMIN(I)=1
0010    GO TO 3
0011   2 LMAX(I)=1
0012    LMIN(I)=2
0013    3 CONTINUE
C
C IS THIS THE FIRST TREE PAIR OR THE SECOND TREE PAIR
C
0014   GO TO (300,400), K
C
C FIRST PAIR. CREATE A NODE FOR EACH TREE WITH FIRST OPERATION ON MACHINE A
C
0015   300 TIMEAX=A(I)
0016   TIMEBX=D(I)
0017   TIMEAX=B(I)
0018   TIMEBR=D(I)
0019   GO TO 500
C
C SECOND PAIR. CREATE A NODE FOR EACH TREE WITH SECOND OPERATION ON MACHINE A
C
0020   400 TIMEAX=C(I)
0021   TIMEBX=B(I)
0022   TIMEAX=C(I)
0023   TIMEBR=B(I)
0024   500 L(I)=K
0025   LL=LMAX(I)
C
C WHICH MACHINE HAS THE LARGER OPERATION TIME FOR THE FIRST JOB
C
0026   GO TO (600,700), K
0027   600 GO TO (4,6), I
0028   700 GO TO (6,4), I
C
C MACHINE A. ASSIGN SMALLER OPERATION TIMES OF REMAINING JOBS TO MACHINE A
C
0029 4 DO 5 I=2,IN          $1600290
0030  TIMEA=TIMEA+MIN1(A(I),C(I)) $1600300
0031  TIMEB=TIMEB+MAX1(B(I),D(I)) $1600310
0032  5 L(I)=LMIN1(I)           $1600320
0033  GO TO 8                   $1600330

C  MACHINE B, ASSIGN SMALLER OPERATION TIMES OF REMAINING JOBS TO MACHINE B
C
0034  6 DO 7 I=2,IN             $1600340
0035  7 TIMEA=TIMEA+MAX1(A(I),C(I)) $1600350
0036  TIMEB=TIMEB+MAX1(B(I),D(I)) $1600360
0037  7 L(I)=LMAX1(I)           $1600370
C
C  COMPUTE THE LOWER BOUND FOR THE SEQUENCES CREATED
C
0038  8 BOUND=AMAX1(TIMEA,TIMEB) $1600380
0039  DO 16 I=2,JN               $1600390
C  EXTEND THE TREES BY ADDING BOTH POSSIBLE OPERATIONS TO BOTH TREES
C
0040  TIMEA=TIMEA+AMIN1(A(I),C(I)) $1600400
0041  TIMEB=TIMEB+AMIN1(B(I),D(I)) $1600410
0042  TIMEA=TIMEA+AMAX1(A(I),C(I)) $1600420
0043  TIMEB=TIMEB+AMAX1(B(I),D(I)) $1600430
0044  TIMEA=TIMEA               $1600440
0045  TIMEB=TIMEB               $1600450
0046  TIMEA=TIMEA               $1600460
0047  TIMEB=TIMEB               $1600470
0048  JX=I+1                    $1600480
0049  DO 9 J=JX,IN               $1600490
0050  TIMEA=TIMEA+AMIN1(A(J),C(J)) $1600500
0051  TIMEB=TIMEB+AMIN1(B(J),D(J)) $1600510
0052  TIMEA=TIMEA+AMAX1(A(J),C(J)) $1600520
0053  TIMEB=TIMEB+AMAX1(B(J),D(J)) $1600530
C  COMPUTE THE LOWER BOUNDS FOR THE NEW SEQUENCES
C
0054  BOUNDR=AMAX1(TIMEA,TIMEB) $1600540
0055  BOUNDS=AMAX1(TIMEA,TIMEB) $1600550
C  THE SMALLER OF THESE TWO LOWER BOUNDS IS THE NEW LOWER BOUND
C
0056  IF(BOUNDR-BOUNDS)10,10,13 $1600560
C  IS THE CURRENT LOWER BOUND SMALLER THAN THE NEW LOWER BOUND
C
0057  10 IF(BOUNDR-BOUNDD)11,11,17 $1600570
C
C USE THE NEW LOWER BOUND TO CONTINUE THE ALGORITHM
C
C 11 BOUNDQ=BOUNDR
C 0059 L(I)=LMAX(I)
C 0060 GQ 12 J=JX,IN
C 0061 12 L(J)=LMIN(J)
C 0062 GO TO 16
C
C IS THE CURRENT LOWER BOUND SMALLER THAN THE NEW LOWER BOUND
C
C 0063 IF(BOUNDQ-BOUNDR)14,14,17
C
C USE THE NEW LOWER BOUND TO CONTINUE THE ALGORITHM
C
C 0064 14 BOUNDQ=BOUNDS
C 0065 L(I)=LMIN(I)
C 0066 GQ 15 J=JX,IN
C 0067 15 L(J)=LMAX(J)
C 0068 TIMEAX=TIMEAY
C 0069 TIMEBX=TIMEBY
C 0070 16 CONTINUE
C
C IS THIS THE FIRST TREE PAIR OR THE SECOND TREE PAIR
C
C 0071 GO TO (18,19),K
C
C FIRST PAIR. THE CURRENT LOWER BOUND IS SMALLER THAN ANY OTHER LOWER BOUND FOR
C THIS TREE PAIR. SAVE THE SEQUENCE AND THE LOWER BOUND
C
C 0072 18 BOUND1=BOUNDQ
C 0073 DO 180 I=1,IN
C 0074 180 L(I)=L(I)
C 0075 19 CONTINUE
C
C SECOND PAIR. THE CURRENT LOWER BOUND IS SMALLER THAN ANY OTHER LOWER BOUND FOR
C THIS TREE PAIR. THE SEQUENCE FOR THE TREE PAIR WHICH HAS THE SMALLER LOWER
C BOUND IS AN OPTIMAL SEQUENCE
C
C 0076 IF(BOUND1-BOUNDQ)20,20,22
C 0077 20 DO 21 I=1,IN
C 0078 21 L(I)=L(I)
C 0079 22 DO 25 I=1,IN
C 0080 25 L(I)=L(I)
C 0081 GO TO (23,24),I1
C 0082 23 AX(I)=A(I)
C 0083 0X(I)=D(I)
C 0084 GO TO 25
C 0085 24 AX(I)=C(I)
C
### SCALAR MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>JN</td>
<td>17C</td>
<td>IN</td>
<td>180</td>
<td>K</td>
<td>183</td>
<td>I</td>
<td>183</td>
</tr>
<tr>
<td>TIMEX</td>
<td>190</td>
<td>TIMEAR</td>
<td>194</td>
<td>TIMEBR</td>
<td>198</td>
<td>IL</td>
<td>190</td>
</tr>
<tr>
<td>TIMEAY</td>
<td>1A4</td>
<td>TIMEBY</td>
<td>1A8</td>
<td>TIMEAS</td>
<td>1AC</td>
<td>TIMEBS</td>
<td>1A0</td>
</tr>
<tr>
<td>J</td>
<td>188</td>
<td>BOUNDR</td>
<td>1B6</td>
<td>BOUNDS</td>
<td>1C0</td>
<td>BOUNDI</td>
<td>1C4</td>
</tr>
</tbody>
</table>

### ARRAY MAP

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1CA</td>
<td>B</td>
<td>1CC</td>
<td>C</td>
<td>100</td>
<td>D</td>
<td>104</td>
</tr>
<tr>
<td>BX</td>
<td>1UC</td>
<td>L</td>
<td>1E0</td>
<td>L1</td>
<td>1E4</td>
<td>LMAX</td>
<td>1F8</td>
</tr>
<tr>
<td>AMIN1</td>
<td>220</td>
<td>AMAX1</td>
<td>224</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### SUBPROGRAMS CALLED

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
<th>SYMBOL</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### TOTAL MEMORY REQUIREMENTS 0000656 BYTES
SUBROUTINE TIME23(FRAC1,FRCB1,FRACC1,FRAC2,FRCB2,FRACC2,
1FRAC3,FRCB3,FRACC3,FRACM3,FRCBM3,FRACM4,FRCBM4,FRACBC,FRCBMBC,
2FRACCM,FRCBCC,FRACC1,1,CPU,TIMEB,TIMEC)
C******************************************************************************
C CALCULATE MULTIPROGRAMMING TIMES FOR SECOND- AND THIRD-PRIORITY PROGRAMS
C******************************************************************************
C
0002 INTEGER CPU
0003 DIMENSION FRACM(12,2),FRCBM(12,2),FRACCM(12,2),FRACM(12,2)
0004 DIMENSION FRACT(12,2),FRACMT(12,2),FRACM(12,2),FRACBT(12,2)
0005 DIMENSION FRACTC(12,2),FRACTMT(12,2),FRACMT(12,2),FRACBC(12,2)
0006 DIMENSION FRACTBC(12,2),FRACTBCM(12,2),FRACMBC(12,2),FRACCM(12,2)
0007 TIMEC=0.
0008 C
C DO WE HAVE TO CALCULATE THE TIME FOR THE SECOND PRIORITY PROGRAM
C
0009 GO TO (1,4,1),1
C
C YES, CALCULATE THE TIME FOR PORTION OF PROGRAM OVERLAPPED WITH MPX CHANNEL OMS
C
0010 1 TIMEB=0.
0011 CALL TIME2 (FRAC1,FRCB1,FRACM1,CPU,TIMEB)
0012 CALL TIME2 (FRAC1,FRCB1,FRCBM1,CPU,TIMEB)
0013 CALL TIME2 (FRAC1,FRCB1,FRACM2,CPU,TIMEB)
0014 CALL TIME2 (FRAC1,FRCB1,FRCBM2,CPU,TIMEB)
C
C ARE BURST MODE MUXPLEXOR OPERATIONS OVERLAPPED WITH INSTRUCTION EXECUTION
C
0015 IF CPU=204012,2,3
C
C NO, CALCULATE TIME FOR PORTION OF PROGRAM NOT OVERLAPPED WITH BURST MODE OMS
C
0016 2 CALL TIME2 (FRAC1,FRCB1,FRACM1,CPU,TIMEB)
0017 CALL TIME2 (FRAC1,FRCB1,FRCBM1,CPU,TIMEB)
0018 CALL TIME2 (FRAC1,FRCB1,FRACM2,CPU,TIMEB)
0019 CALL TIME2 (FRAC1,FRCB1,FRCBM2,CPU,TIMEB)
0020 CALL TIME2 (FRAC1,FRCB1,FRACM3,CPU,TIMEB)
C
C DO WE HAVE TO CALCULATE THE TIME FOR THE THIRD PRIORITY PROGRAM
C
0021 3 GO TO (6,4,1),1
C
C YES, CALCULATE THE TIME FOR THE THIRD PRIORITY PROGRAM
C
0022 4 CALL TIME3 (FRAC1,FRCB1,FRACM1,CPU,TIMEB,TIMEC)
0023 CALL TIME3 (FRAC1,FRCB1,FRCBM1,CPU,TIMEB,TIMEC)
S1700010
S1700020
S1700030
S1700040
S1700050
S1700060
S1700070
S1700080
S1700090
S1700100
S1700110
S1700120
S1700130
S1700140
S1700150
S1700160
S1700170
S1700180
S1700190
S1700200
S1700210
S1700220
S1700230
S1700240
S1700250
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700260
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700270
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700280
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700290
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700300
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700310

C ARE BURST MODE MULTIPLEXOR OPERATIONS OVERLAPPED WITH INSTRUCTION EXECUTION
C
IF(CPU-2040)5,5,6 S1700320
C
C CALCULATE TIME FOR PORTION OF PROGRAM NOT OVERLAPPED WITH BURST MODE OPS
C
5 CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700330
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700340
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700350
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700360
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700370
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700380
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700390
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700400
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700410
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700420
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700430
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700440
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700450
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700460
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700470
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700480
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700490
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700500
CALL TIME3 (FRAC1, FRAC2, FRAC3, FRACM, CPU, TIMEB, TIMEC) S1700510
6 RETURN S1700520
051 END S1700530
SUBROUTINE ERRPRT(ERROR)

C ***********************************************************************
C PRINT ERROR HEADING AND UPDATE ERROR COUNT
C ***********************************************************************

INTEGER ERROR

ERROR=ERROR+1

WRITE (6,1)

1 FORMAT('********** ERROR **********')

RETURN

END
BIBLIOGRAPHY


