AN ABSTRACT OF THE THESIS OF

Seouk Joo Lee for the degree of Master of Science in Industrial Engineering presented on October 16, 1986.

Title: A Flexible Manufacturing System Simulator

Abstract approved: Redacted for Privacy

Eugene Fichter

Most flexible manufacturing systems (FMS) presently in use have generated only modest productivity increases in proportion to capital invested in the process, or they have not been favorably reviewed when compared to the investment costs of other improved manufacturing systems.

This thesis presents a simulator program which will assist manufacturers using flexible manufacturing systems to discover productivity problems. The simulator, written in the FORTRAN language, is easy to use. It will not require that users write FORTRAN code to operate the system, but may be operated by users with no knowledge of either simulation or computer programming.

The simulator uses a question and response technique to encompass the production parameters of the manufacturing systems it is intended to simulate. It will be of particular utility in evaluating the use of alternative work stations and different types of material handling systems.

A Flexible Manufacturing
System Simulator

by

Seouk Joo Lee

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed October 16, 1986

Commencement June 1987

APPROVED:

Redacted for Privacy

Professor of Industrial Engineering in charge of major

Redacted for Privacy

Head of Department of Industrial Engineering

Redacted for Privacy

Dean of Graduate School

Date thesis is presented October 16, 1986

Typed by B. McMechan for Seouk Joo Lee

ACKNOWLEDGEMENTS

There are a number of people who have offered important contributions in the preparation of this thesis. Foremost among them is my graduate advisor, Dr. Eugene Fichter, whose guidance during the past three years has been invaluable. I would also like to thank Dr. Sabah Randawha and Dr. David Birkes, whose suggestions have been very helpful.

My wife, and both of our families, have been an unfailing source of support and to them I would like to express my enduring gratitude.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

A FLEXIBLE MANUFACTURING SYSTEM SIMULATOR

CHAPTER I

INTRODUCTION

Factory automation is one of the more important issues in the effort to remain competitive among the industrialized nations of the world. One of the most important concepts employed in factory automation is the use of Flexible Manufacturing Systems (FMS). Although FMS have only recently been introduced, many manufacturing organizations have recognized FMS as a promising solution to certain low productivity problems and as a means to adapt to radical fluctuations in market demand. Recently a number of far-sighted industrial engineers have foreseen FMS as a major stepping stone in the creation of unmanned manufacturing systems, or the Computer Integrated Manufacturing Systems (CIMS) (Merchant, 1985).

To date there is no precise or universally accepted definition of FMS. In general, FMS serves to integrate machine modules and material handling devices in an automated workflow system under computer control, producing different products in small-batch production systems. The major components of a typical FMS include: workstations, such as a load/unload stations and machining, inspection, and washing stations; material handling devices, such as conveyors, robots, and automatic guided vehicles (AGV); and a computer control system which utilizes appropriate programs to monitor, control, and schedule the entire operation.

An FMS can produce different parts without machine set-up changes. The parts are fed into the system at a loading station and undergo a specified sequence of operations at work stations before leaving the system at an unloading station. The flexibility of the system allows the choice of one or more stations for each operation. Computer control of FMS is executed by one or more computers which control the transportation system and the scheduling of operations at the work stations. The work stations are equipped with stored program controllers which direct local operations.

FMS offers a number of major business advantages. FMS usually results in shorter lead-time, the reduction of work-in-progress inventories, and the improved utilization of work stations. Besides these advantages the other benefits of FMS are: reduction in labor costs; reduction in the number of necessary machine tools; reduction of work floor space requirements and production time; improved tool utilization; and the lowering of production costs.

On the other hand FMS requires a heavy investment to install, higher set-up costs, greater costs of training, and the cost of the time necessary to adapt to the system after its installation until satisfaction is achieved.

It is difficult to find an optimal configuration for an FMS. Some of the methods developed in operations research (scheduling, inventory control, resource allocation, optimal routing, queuing, etc.) may be useful, but these methods are insufficient in their practical application. They only produce limited solutions for specific aspects of the system. In particular, an FMS incorporates many factors which are closely interrelated. Simulation techniques can be used as an effective tool to evaluate the proposed change or to select the most effective configuration under given conditions.

In recent years many powerful simulation package programs have been developed which include a number of functions, such as graphic animation, various performance reports, and user-written functions. Additionally, a few simulators, each of which has a simulation program for its specific purposes, have also been developed.

Simulator programs may be used as a tool to determine good FMS system configurations, such as the number of possible permutations and combinations of workpieces, tools, and automatic transport vehicles. Additionally, the simulation program is an effective tool for testing any number of parameters which affect the production system, such as the number of workstations, the requirements of different workpieces, processing priority rules, or material handling systems, and for designing actual operating software which will control production and real-time scheduling. On the other hand, simulation is thought to be difficult and costly, normally requiring the help of a specialist.

This thesis describes a simulator program for FMS design and application. This simulator uses a question and answer format to allow users who have no background of computer simulation or computer programs to build input data into the program, i.e. the simulator provides the user with a variety of options to choose from within a number of fields. In addition, the simulator provides for input data either through the console or user-created files. By using these functions the user may easily debug the system and change input values without the necessity of reentering the whole data; they may also be used to compare different systems and to find good configurations by changing parameters. In addition, the simulator provides for specific performance reports on the FMS, reports which may be read by personnel inexperienced in a computer language.

This thesis is structured as follows:

Chapter II presents a survey of current literature in analytic methods, simulation methods, and in other studies in this field.

Chapter III presents a brief tutorial on FMS, with a brief explanation of input information required by the simulation program.

Chapter IV presents a detailed description of the simulation algorithm and programming and detailed instruction for using the simulation.

Chapter V presents a detailed explanation of the input and output procedures and an example of a simulator run.

Chapter VI presents a summary of the work presented in this thesis and provides recommendations for further study.

CHAPTER II

REVIEW OF LITERATURE

2.1 Introduction

The concept of the FMS was developed by Theo Williamson of the Mollins Company in the United Kingdom during the late 1950s (Crite et.al., 1984). During the 1960s a few companies in the United States installed FMS in their factories. As a result of this experience similar systems were built in the 1970s in Japan and in Europe, until by the beginning of the present decade there were approximately 40 systems in use worldwide. Since that time additional systems have been installed at an increasing rate. According to the journals ("FMS, A Boom," 1986), the number of FMS will increase from 76 in 1985 to 660 in 1990, an average annual rate of increase of 54 percent.

FMS has been under consistent and thorough study only during the last few years. Before 1980 few articles were devoted to FMS (Goldberg, 1979), but since then large numbers of articles and books on various aspects of FMS have been published. Some of the topics covered are analytical methods, general purpose oriented simulation languages, computer simulations, simulators, simulation and graphics, coordination with other manufacturing systems (CIMS), Artificial Intelligence (AI), and computer-aided design and manufacturing (CAD-CAM).

2.2 Analytical Methods

Since 1970 many analytical methods have been developed to find optimal solutions for FMS problems. In 1970 Coffman developed the combinational technique to solve job shop scheduling and in 1978 Kinemia and Gershwin formulated nonlinear network flows to determine optimal parts routing for alternative operations (cited in Kinemia and Gershwin, 1985). In 1981 Kinemia and Gershwin established the advantages of real control policies when a machine fails and in 1982 Buzacott developed a set of decision rules for alternative operations (Kinemia and Gershwin, 1985).

In 1982 King and Nakornchai developed methods to solve grouping problems of parts and machine loading and in 1983 Stecke formulated and solved machine loading problems by using nonlinear integer programming formulation. In the same year Andrew Kusiak solved the same problem by the formulation of linear integer programming for machine loading problems (cited in Kusiak, 1985). More recently, Kinemia and Gershwin (1985) developed solution techniques for a network flow optimization approach to determine optimal part routing in an FMS modeled by a network of queues.

2.3 FMS Simulation Programs and Simulators

The ideal computer simulation program for FMS has not been written. Such a simulation program or simulator would accurately model any conceivable FMS, as well as run on any computer and be used by persons completely unfamiliar with computers and simulation (Editors of American Machinist, 1983).

Many of the existing computer simulation programs are written in FORTRAN, which has the virtue of modest memory requirements and which may be run on small computers. Even though they have some flexibility in making or modifying models, they are difficult to write, debug, and modify without a computer programming background. On the other hand some of the non-FORTRAN simulation languages (e.g. GPSS or SIMSCRIPT) have no flexibility in working with simulation programs other than those for which they were designed.

Elmaraghy (1982a, 1982b) developed a simulation program, FMSSIM, based on GASP IV, which is capable of various configurations, such as bidirectional trace of material handling systems, route blockage caused by cart interference, and random failures and repair of the various components in the system. Fox (1982) developed VARIABLE MISSION, a program which has a sequential scheduling method for scheduling a flexible system operating in a batch model. Lenz (1983a, 1983b) developed MAST, a simulation-aid program for designing FMS involving machine tools, and which has free-formatted data instructions describing the manufacturing system as well as the flexibility to simulate a wide variety of systems through the use of data alone. In 1984 Crite, Mills and Talavage (1984) designed PATHSIM, a program written in the SLAM simulation language and which may be used to evaluate tool handling systems. Elmaraghy (1985) then followed with TOLSIM, a program based on GASP IV which is used for designing and evaluating automated tooling systems.

Yih-Long Chang, Robert Sullivan, and James Wilson (1986) used SLAM to design the material handling system of a flexible manufacturing system. For the application of graphics to simulations, Bahram Karamati (1983) and Bernard (1984) have developed programs which utilize sophisticated graphics devices to offer engineers a realistic view of systems operations.

For the study of simulators for FMS, Duersch and Layman (1983, 1984) developed a graphic workflow simulator to design or analyze FMS. This simulator incorporates graphics and question-answer interfaces to build a simulation without the user's need to know any programming language. This simulator uses a graphics screen, a graphics tablet, and a "puck" as input devices, and a graphics screen and printer for output. Diersh and Malstrom (1985) devised a physical simulator which has an operational scale model of the actual system. Physical simulators can be used to obtain operational data from the system, data which may be used as an aid in the design, installation, and operation of the actual system.

2.4 Other Related Fields of FMS

Elmaraghy and Ho (1982a, 1982b), Duersch and Layman (1983), and Lenz (1983, 1984) suggested the use of graphic animation during simulation. This method represents the results of simulation in an easily understood fashion. Arbel and Abraham (1984), Ranky (1984), and Hartley (1983) discussed the justification of FMS with the benefits and costs of the problem displayed in an analytic hierarchy process. Raju (1985) has discussed the role of robotics in FMS. Merchant (1985) explained the relation of FMS and CIMS, noting that an FMS implementation is a logical stepping stone toward the development and institution of a CIMS. Additionally, he suggested the use of AI to make FMS an intelligent system. Pun, Doumeingts, and Bourley (1985) developed the GRAI network approach, which is used to improve the utilization of a set of tools and a set of methodologies using AI.

2.5 Scope of This Study

This thesis presents an FMS simulator. This simulator asks of its user, seated at a computer display terminal, questions about the characteristics of the FMS to be simulated. Then, after performing the simulation program, it will immediately display or print the results. After checking the results the user can then change some of the system configurations and in so doing, find good solutions for production problems.

CHAPTER III

FLEXIBLE MANUFACTURING SYSTEM

3.1 Introduction

Hard automation systems have been considered as an economical means for large volume production at a reasonable cost. However, flexible manufacturing systems are a practical means of combining high productivity with the processing of small batch sizes and short lead times. Since Mr. Williams, as director of R & D at Mollins (Deptford, Great Britain), installed "System 24," there has been growing interest in the development and implementation of FMS. To date hundreds of these systems have been implemented around the world, with Japan leading in terms of the number of applications and associated management and organization.

The periodical, Production Engineering (1986, February) noted that the total potential U.S. target market for FMS installations clearly exceeds 10,000 locations and users of small FMS may exceed 4,000 in number.

3.2 Why FMS is Needed

According to the report of Cincinnati's Milacron Research Center (Hartley, 1986), each workpiece only spends about 30 percent of the working time actually on the machines. The rest of the time the machinist is either setting up the machine (loading, unloading, positioning, etc.), going to obtain materials or orders for the next job, or sorting out tools and similar operations. Even worse, another study (Gatelmand, 1983) indicated that each workpiece spends 95 percent of its manufacturing time travelling and waiting and only 5 percent of that time on the machine. A further reason for the need of FMS is that while conventional machinery operates less than 12 hours per day, FMS may be operated 24 hours per day under computer control.

Flexible manufacturing systems produce a wide variety of products on relatively few machines with low manning levels and provide a great deal of flexibility to increase system utilization; FMS reduces direct labor costs because all manufacturing work stations and material handling systems are controlled and efficiently directed by hierarchically structured computers. Concentration of machine stations in a small area enables an automated transport system, faster processing, and therefore shorter lead times. An FMS also reduces work-in-progress inventories by virtue of computer control and offers production flexibility. In addition, totally different products may easily be manufactured to meet the changing demands of the marketplace. When abrupt engineering design changes are required, FMS requires less setup time and lower change-over costs (Hartley, 1984).

3.3 Basic Components of FMS

As previously mentioned, an FMS usually consists of different types of work stations, material handling systems, and a computer control system. In addition, a considerable amount of software is required to control components of the system.

An FMS requires many different kinds of work stations There are load/unload stations, machine stations, inspection stations, and heating and washing stations, among others. Machine stations do the actual work on the part. To maintain versatility they require many different tools handled by an automatic tool changer. The computer maintains records concerning the tools used for each job. After a job is processed on the machine it may require an inspection, which is usually done by people or is only partly automated. However, recent advances in automated measuring systems utilizing computers has enabled their use for inspections.

Most of the material handling systems consist of a number of vehicles and a track. The types of vehicles are those powered by batteries, wire-guided carts, and automatically guided vehicles (AGV). Among these vehicles the AGV is more flexible than the others and among track styles, straight line or loop styles are the most commonly used (Hartley, 1984). Only a few systems have been installed with a complicated network-style of track.

3.3.1 Work Stations

3.3.1.1 Machine Stations

A machine station may be constructed from conventional numerical control (NC) or computerized numerical control (CNC) machine tools, such as lathes, milling, drilling, welding machines, and punch presses. Machine stations may have tool changers, head changers, and workpiece changers as auxiliary equipment. The tool changers select tools automatically under computer control. The computer records the status of each machine tool and assigns the correct tool to each workpiece for processing. On the other hand an FMS may have all of its tools at one location, in this instance requiring a tool carrying system to move tools to the assigned machines. Several software programs, such as parts programs, machine diagnostics, and machine data evaluation, are mounted to control the parts program and various operation programs.

3.3.1.2 Inspection Stations

Inspection is generally executed by a quality test. Quality tests are executed by a common test system or a systematically and well-developed computerized test system. A common test system is cheap, easy to install, and easy to apply. However, this system is simple, lacking flexibility when the addition or deletion of a test system is required. On the other hand, computerized test systems require a substantial setup investment. However, the computerized system has various testing and analytic capabilities: automatic checks at random times; automatic correction of test results. A computer program provides a rapid means of acquiring test parameters, evaluations of test results, calculations of the quality index, and output of the test results by CRT display or on hard copy.

A quality control system in FMS can be viewed as a complex adaptive control loop because all the activities affect one another. With the help of various instruments (sensors, measuring instruments, and computers) all properties and the performance of the product are identified. This quality test is done at many places in a factory, such as receiving docks, load stations, assembly stations, and a final acceptance test at unload stations. During the quality test if a product is found defective, measures for its correction or possible elimination of the defect from the product may be considered. Corrective actions should be executed at the level of the manufacturing process, or they may be adjusted in the product design procedure.

3.3.2 Material Handling and Storage Systems

The systems consist of various conveyors, transport vehicles (guided cars, shuttle cars, towline carts, forklifts), tracks, pallets, warehouse facilities, control devices (sensors, computers, controllers), and a number of software programs (e.g., material flow, inventory, control). The equipment and software used in these systems has been greatly changed. The application of automatic controlled equipment greatly facilitates the movement of the parts from receiving to finished-goods warehousing.

When a part arrives at the system it first must be identified by a visual or automatic inspection. Then a pallet may be used as workpiece delivery equipment between work stations or as actual work tables. In most cases the material handling system contains a work setup area where most often a part is clamped on a pallet and the pallet becomes the actual worktable. The pallet is delivered by computer controlled AGV or conveyor with the identity of the workpiece recorded by the computer. The computer determines the routing of the part, sending it from one machine station to the next station (or to an inspection station). In the machining process the pallet is clamped securely to the worktable of the processing equipment and upon completion of the operation the pallet is released and sent to its next destination. During its travels through the system the computer maintains continual trace of the workpiece and its location through the use of various tracking methods (machine-readable code cards, optical decoding, and machine vision).

Typical flow patterns of a MHS are shown in Figure 3.1.

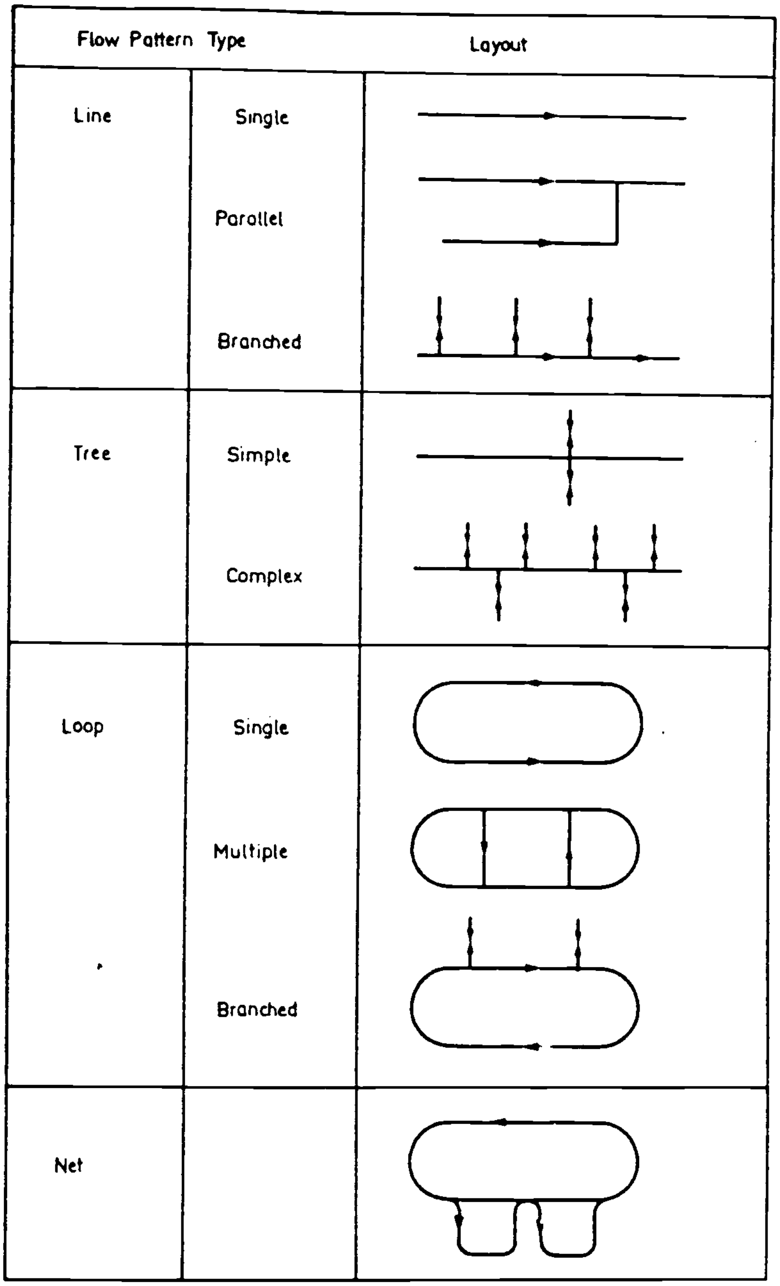


Figure 3.1 Typical Flow Patterns
(Rembold, Blume, & Dillmann, 1985)

3.3.3 Computer Control of FMS

The key element of an FMS is computer control. It integrates the work stations with planning and scheduling and it controls the physical movement and tracking of the job.

The control system typically consists of minicomputers that form a hierarchy of networked controls directing FMS subsystems. At each level of control the computer network assures the efficiency of the process. Operating in real-time mode, the typical FMS control system provides workpiece tracking; material transport, and storage system, and work station control; record maintenance; system performance reports; system simulation capacity; tracking and status for key system components; two-way communication (interactively) with system personnel; data recall and editing capabilities; and production scheduling information. The control system should operate the FMS efficiently in an automatic mode, including redundancy for emergency backup.

This system requires communication terminals for operators and line printers for report generation. Communications hardware is also required for transmitting and receiving signals from the transport and storage systems and work stations.

3.4 FMS Flexibility

Flexibility is one of the keys to FMS utility, since increased flexibility is considered one of the best means to increase system utilization. The FMS offers the following types of flexibility:

a) Mix flexibility: the ability to process various members of a well-defined family of parts without the loss of time for setups.

b) Parts flexibility: the ability to add new parts to the families.

c) Routing flexibility: the ability to reroute a part in process to avoid machines under repair or those with relatively long queues.

d) Design change flexibility: the ability to quickly modify parts with the ease of implementing engineering design changes.

e) Volume flexibility: the ability to respond to volume changes without any increase in unit workpiece cost, loss of productivity, or reduction in equipment utilization.

f) Factory systems flexibility: the ability to accommodate changes in the future factory hardware or information systems.

CHAPTER IV

FMS SIMULATOR ALGORITHM AND PROGRAM

4.1 Introduction

This simulator was implemented on a Tektronix 4170 microcomputer using the CP/M 86 operating system and is written in Fortran 86, which was developed by Intel. This Fortran program has some non-standard features to define numeric types and has no random number generating function. The simulator consists of more than 4,000 lines and is divided into five parts. There are the main program, input subroutines, FMS logic subroutines, the subroutines for random variables generation and FMS statistics, and simulation library subroutines.

4.2 FMS Simulator Modeling

More detailed algorithms are explained in the following section. This section describes the concepts of simulation modeling and the simulator design criteria.

4.2.1 Simulation Modeling

The FMS simulator program uses a discrete event simulation, requiring the concepts of event, activity, and process. Figure 4.2.1 indicates the relationship of these concepts.

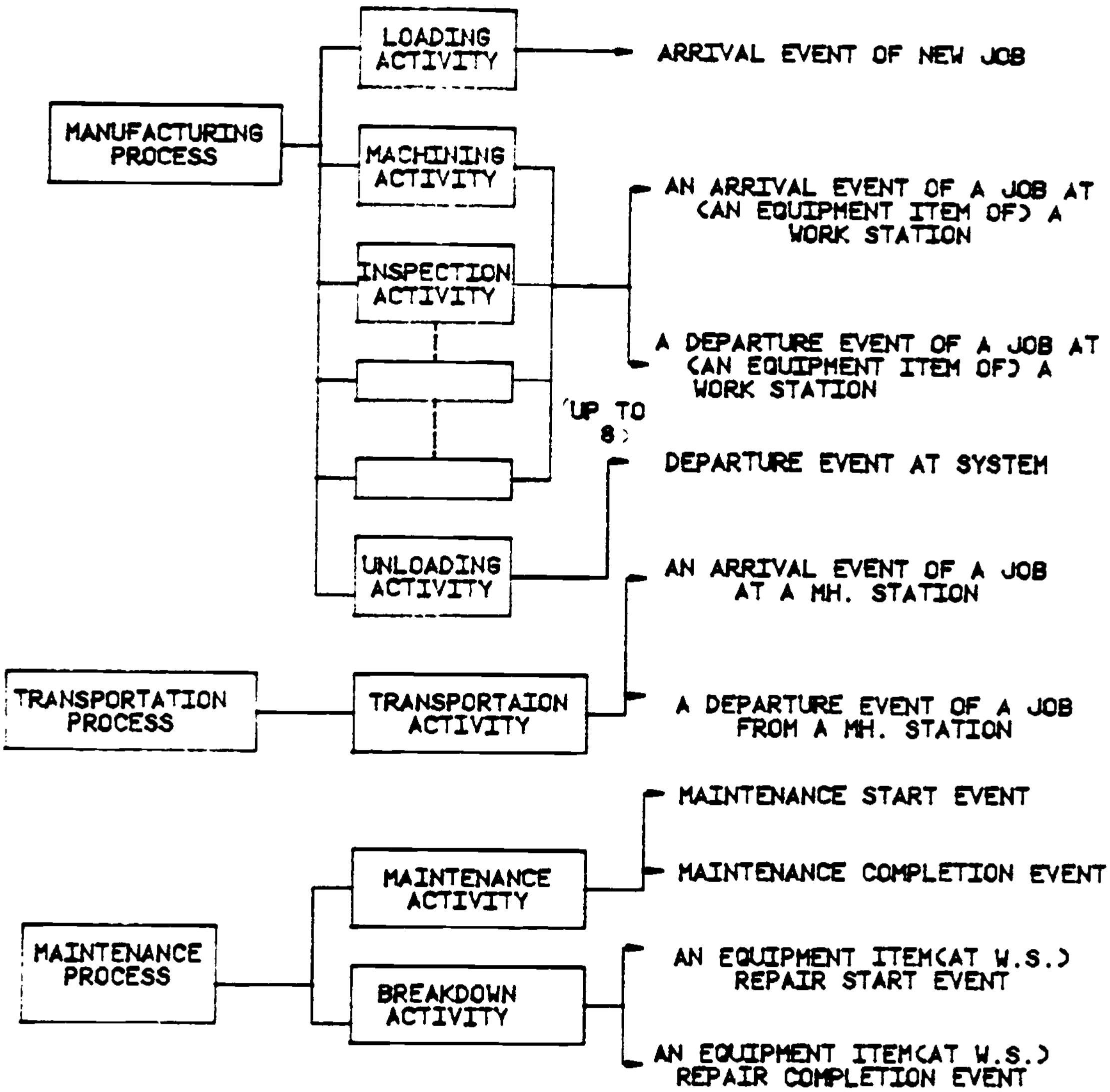


Figure 4.2.1 The Relationship of Event, Activity, and Process

4.2.2 Simulator Design Criteria

The simulator in this study is a program capable of simulating the most common configurations of an FMS. Dialogue-style programs for various data input are used, detailed explanations of which are provided in the next section. In addition, the following design criteria were established to define the scope of the simulator program:

1) The ability to represent two types of material handing system tracks, including either straight line or closed loop bidirectional track and closed loop unidirectional track.

2) The ability to handle up to 10 different kinds of work stations (e.g., load/unload, machine, inspection, washing, heating).

3) The ability to determine the size of buffers for work stations and material handling systems.

4) The ability to avoid collisions between moving AGV vehicles.

5) The ability to use three different kinds of random variable generations for job arrival time, work time, work station repair or maintenance time, or other similar events.

6) The ability to handle tool changes in the tool magazines.

7) The ability to substitute equipment items at a work station because of work station breakdown or the length of the queue (user-defined).

8) The ability to run three types of simulation completions.

9) The ability to simulate both random and predetermined arrival event generations for new jobs.

10) The ability to handle rework or to scrap parts when they are defective.

4.3 FMS Simulator Program

This simulator consists of one main program and 54 subroutines. The subroutines are divided into four parts: input subroutines, FMS logic subroutines, subroutines for random variables generation and FMS statistics, and simulation library subroutines.

4.3.1 Main Program.

The main program consists of a number of subroutines and FORTRAN variables. They are called to implement the simulation process, to enter FMS data into simulation, to execute the FMS simulation, and to generate various results, which may be CRT displayed and/or printed by hard copy machines.

4.3.2 Subroutine Programs

4.3.2.1 Input Subroutines

This group consists of nine subroutines, most of which are used to enter FMS data and several variables for simulation. Table 4.3.1 shows the input subroutines, including the input items.

Table 4.3.1: Input Subroutines

Subroutines	Input Items
HEAD	Project name, user name, date.
INPUT	Number of job types, number of work station types, job mean arrival time, simulation completion time, MH system moving direction, upper or lower limit of random variables, option number (work station breakdown, maintenance schedule, or both).
INPUTJOB	Number of operations in a job, work station type and work station processing time of a job, distribution type of work stations processing times.
INPUTMCH	Number of tools at a tool magazine, tool sequence numbers in tool magazine.
INPUTASK	The tool numbers to use for machine processing.
INPUTAVA	1) Queue sizes at work stations to determine substitution of other equipment items. 2) Work station breakdown rate, repair time, and its distribution type. 3) Work station maintenance interval, maintenance time, and distribution type. 4) Input available work station lists (machine, inspection).
INPUTMHS	MH cart velocity, loading time from work station to MH system and its distribution type.
MAKESIS	Input location (x, y coordinates) of work stations and distance between them. If MH system type = 1 or 3, nearest station number with MH moving direction from assigned station.

4.3.3 FMS Logic Subroutines

This group consists of 20 subroutines which are shown in Table 4.3.2.

Table 4.3.2: FMS Logic Subroutines.

Subroutines	Input Items
ARRIVE	To schedule an arrival event of a new job and a departure event of a job from an equipment item at a work station.
DEPART	To execute a departure event of a job from an equipment item at the work station and schedule an arrival event at the MH station to carry the job to the next work station.
MHSARR & DEFINE	To find the MH station number and the next MH station number when a job is set to this subroutine.
MHSDEP	To execute a departure of a job at the MH station and schedule an arrival event of the job at the next MH station.
LOADST	To schedule an arrival event of the next new job and schedule an arrival event of a job at the MH station (load station), carrying it to the assigned work station.
MACHINE	Select an equipment item at a machine station, then check the status of the equipment item and the tool in use and calculate the statistics of the machine station.
INSPECT	Select an equipment item at an inspection station, then check the status of the equipment item and calculate the statistics of the inspection station.
OTHERS	Check the status of the work station (except machine and inspection stations) and calculate its statistics.
PLAN & AVAIL	When the substitution of an equipment item at a work station is considered, an available equipment item is selected.
PASS	Checks for block conditions on the track and then schedules the departure event of the job for the next MH station.

Table 4.3.2 (continued):.

Subroutines	Input Items
INDEX	To reassign station numbers for calculating FMS statistics.
CHEKTOL	To find an appropriate tool in the tool magazine when machining operation is processed.
CHEKWS	To determine whether the assigned equipment item has been changed because the item is busy or has failed.
CHANGE	To schedule a departure event of a job from one equipment item to another at a work station, changing the destination of the job when the first equipment item selected is occupied.
CHECK	A MH vehicle determines the direction of traffic on its track to avoid collisions.
MIN	To find the next MH station number in a clockwise direction or to the right.
MAX	To find the next MH station number in a counter-clockwise direction or to the left.
FINSERV	To find the work station type for the next operation.

4.3.4 Subroutines for Random Variables Generation and FMS Statistics

This group consists of eight subroutines, five of which are used to generate various kinds of random numbers, including integer, exponential, and uniform numbers. The remaining subroutines are used to calculate FMS statistics. Table 4.3.3 shows the subroutines for random variables generation and FMS statistics.

Table 4.3.3: Random Variables Generation and FMS Statistics Subroutines.

Subroutine	Description
	Statistics
STATMH	To calculate statistics of material handling systems.
STATWS	To calculate statistics of work stations.
	Probabilities
RANDOM	To generate random real numbers between 0 and 1.
DISTRI	To generate random variable numbers.
RANDI	To generate integer random numbers.
EXPON	To generate exponential random numbers.
TRUNEX	To generate truncate exponential random numbers.
UNIFRM	To generate uniform random numbers.

4.3.5 Simulation Library Subroutines

The program has a simulation library which consists of twelve subroutines (see Table 4.3.4), based on the concept of linked storage allocation (Law & Kelton, 1982). The library makes it easy to file a record in a list, to remove a record from a list, to process the event list, and to compute sample statistics variables of interest. In the linked storage allocation approach, each record in a list contains its normal attributes.

Table 4.3.4: Subroutines in the Simulation Library

Subroutine	Description
INITLK	The subroutine initializes the successor and predecessor links, the head and tail points, and the statistics variables for each list.
FILE	The subroutine takes a record which consists of attributes and files it in the list in accordance with the options.
REMOVE	The subroutine removes a record from the list in accordance with the various options.
TIMING	The subroutine determines the event type of the next event to occur and updates the simulation clock.
CANCEL	The subroutine removes the first event from the event list.
SAMPST	The subroutine computes the sample mean, and the maximum and minimum value of a number of observations of the statistics variables.
TIMEST	The subroutine computes the time average (mean), and the maximum and minimum values of a number of observations of the statistics variables.
FILEST	The subroutine computes the time-average number of records, the maximum and minimum number of records in the list.
TRNCOPY	Transfer all the attributes of an event from the simulation library to a subroutine.
QREMOVE	Transfer all the attributes of an event which is placed in the queue to a subroutine.
GOTOQU	Transfer all the attributes of an event which is placed in the queue to the simulation library.
SCHEDUL	Transfer all the attributes of an event which is scheduled to the simulation library.
ERR	To print the error number following the occurrence of an error.

4.4 The Flowchart of the FMS Simulator

The subroutines ARRIVE, DEPART, MHSARF, and MHSDEP play important parts in the simulation program and the subroutines PLAN, AVAIL, CHANGE, PASS, and CHECK are needed to select the appropriate equipment for work processing when an option is selected. The rest of the subroutines (called the general purpose subroutines) are needed to calculating statistics variables, for generating information, for selection of a work station, and for operating three types of material handling systems.

4.4.1 Main Program

The flow of the main program is as follows:

1) The main program, the flowchart for which is shown in Figure 4.4.1, begins with initialization of the FMS model variables (dependent global variables) by calling the subroutine INITCO. The simulation variables are then initialized by calling the subroutine INITLK.

2) Input procedures are executed with the subroutine INPUTF, which provides a "question/answer" format drawing the user step-by-step through all the subroutines. The user can then save the input data to a user file for later use. If the user has already saved the input data file, input procedures can easily be completed after entering the file name. Additionally, if the user wants to modify, add, or delete input data, the procedure may be implemented by calling the subroutine INPUTCHC. At the end of the input procedures the subroutine INPUTF asks for two random number seeds (less than 9 digits) for random number generation by calling the subroutines DISTRI and RANDI. The subroutine DISTRI is for real random number generation and the subroutine RANDI is for integer random number generation. Real random number generation is used to generate

various event times, such as work processing times and maintenance interval times. Integer random number generation is required for job type generation.

3) Simulation is initialized with scheduling of the first job arrival event. However, if manufacturing scheduling has already been assigned, the user can enter the arrival events of all jobs by calling the subroutine GENEVT.

4) The event of the simulation completion is scheduled.

5) The subroutine TIMING in the simulation library is called to determine the event-type of the next event to occur, and the simulation clock is updated.

6) After the next event is determined the appropriate subroutine for that event-type is called: Either ARRIVE, DEPART, MHSARR, or MHSDEP is selected. The subroutine ARRIVE is principally responsible for execution of the schedule of a new job arrival and a departure of a job from an equipment item at a work station. The subroutine DEPART executes the departure of a job and then schedules an arrival event of a job at the material handling (MH) station to carry the job to the next assigned work station. The subroutine MHSARR schedules a departure event of a job from the MH station to the next MH station. The subroutine MHSDEP executes a departure event of a job and schedules an arrival event of a job at the next MH station; or it schedules an arrival event at a work station equipment item if the MH vehicles have arrived at the MH station, which is located at an equipment item in the work station.

7) Steps 5 and 6 are repeated until the simulation completion event is entered as previously set.

8) When the simulation completion event is set to the main program, the program generates an output report by calling the subroutine OUTPUT.

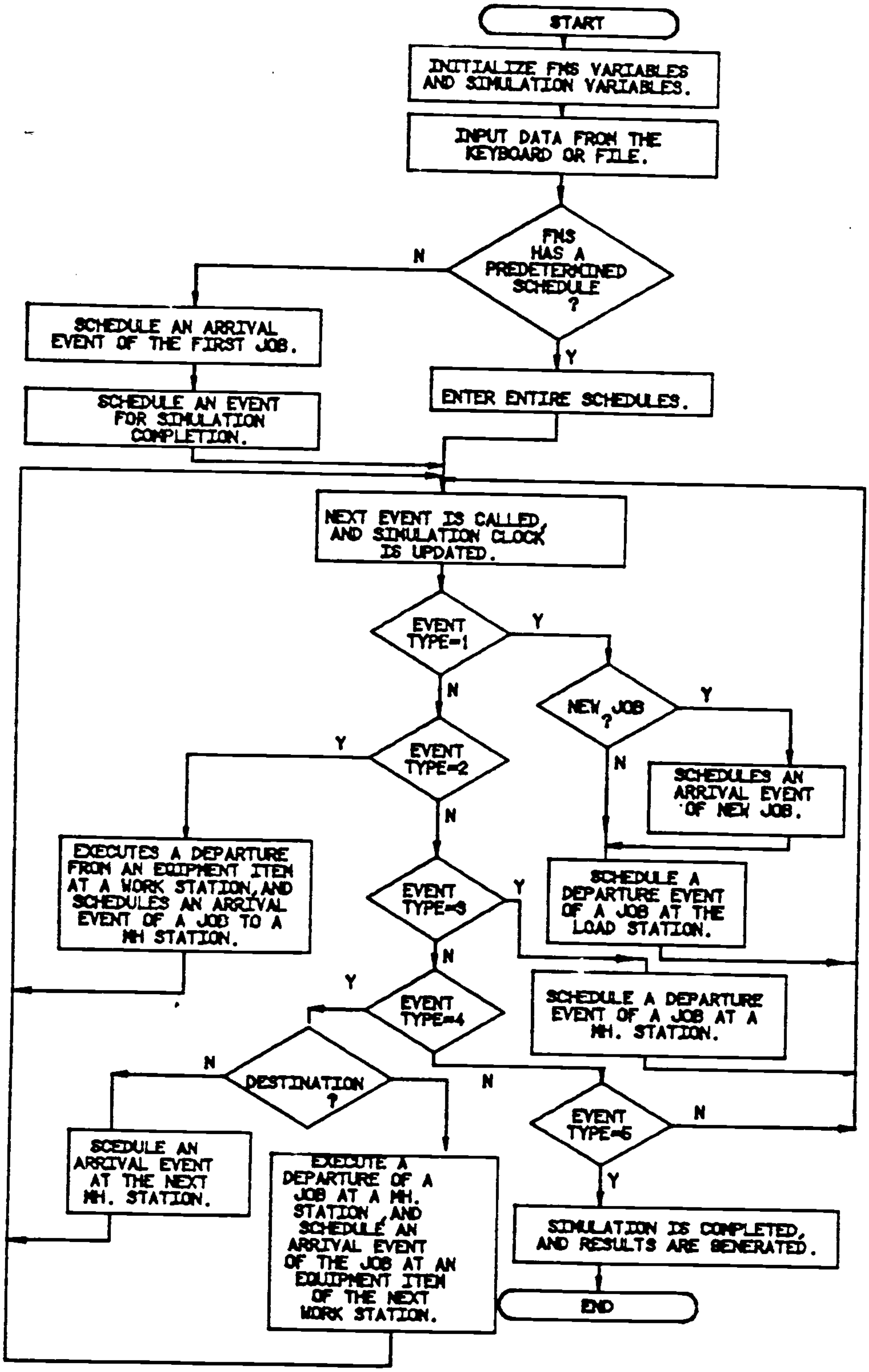


Figure 4.4.1: Main Program Flowchart.

4.4.2 Major Subroutines

4.4.2.1 Subroutine ARRIVE

When the arrival event of a job is set to this subroutine, the subroutine checks whether or nor the job is new. If the job is new the subroutine LOADST is called, which first schedules an arrival event of the next new job and then schedules an arrival event of a MH station to carry the job to its assigned work stations.

If the job is not new, the subroutines MACHINE, INSPECT, or OTHERS are called to schedule a departure event of a job from an equipment item at a work station in accordance with the work station type. If option 2, 3, or 4 is selected in the subroutine INPUTF, a work station breakdown event, or a maintenance event, is scheduled. The subroutines MACHINE and INSPECT call the subroutines PLAN and AVAIL to control the option. The subroutine PLAN first checks which item of equipment at a work station is available for the job. If an equipment item of a work station is busy or it has either failed or is on the maintenance schedule, the use of an alternative equipment item can be considered. In this case the subroutine AVAIL is called to find another item of equipment at the work station. If the option is not considered, or all the equipment is busy, the job generally goes to the queue and waits for processing until the previous job is completed and the next equipment item is available. On the other hand, the subroutine OTHERS does not consider the use of alternative equipment.

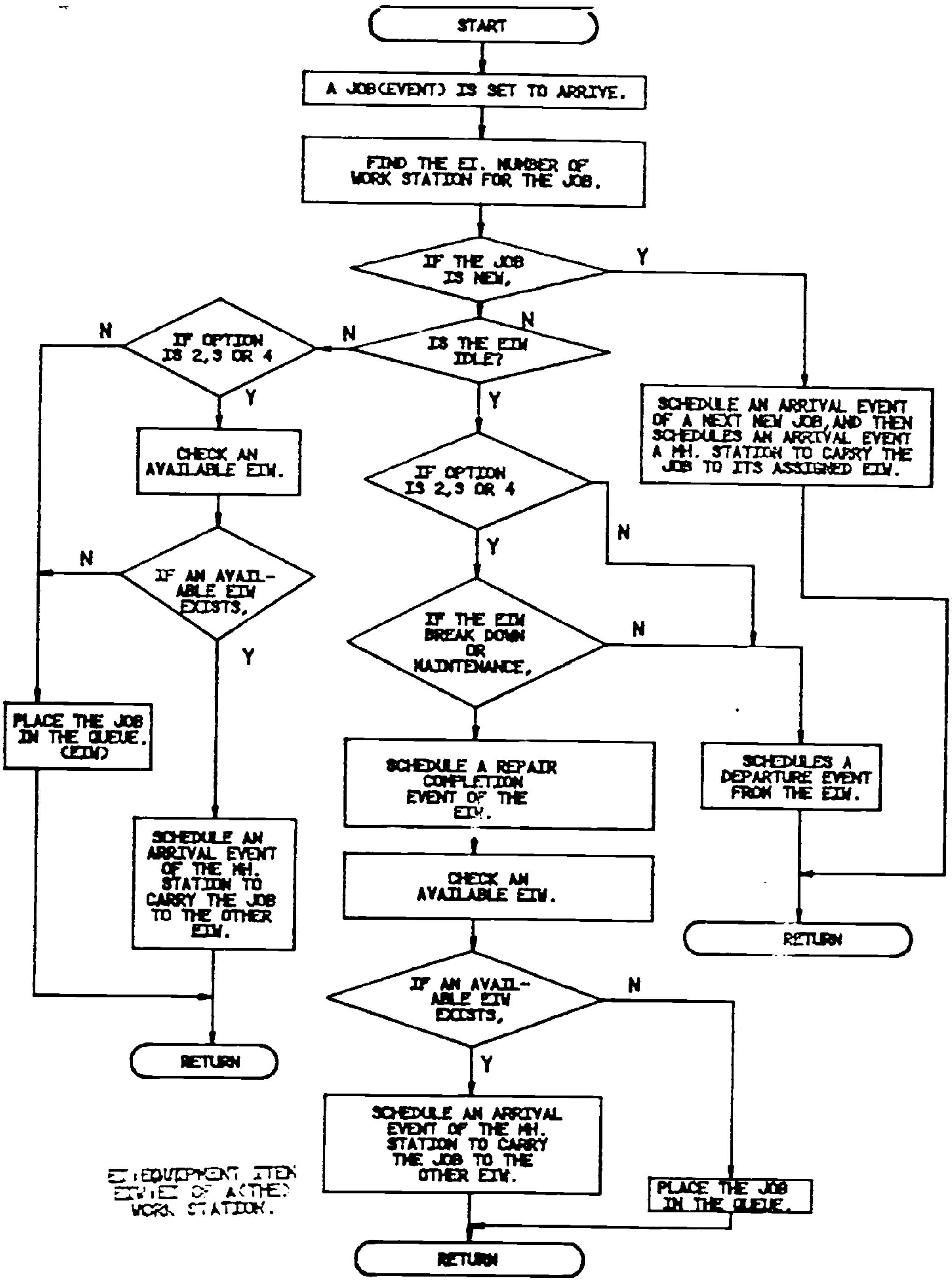


Figure 4.4.2: Subroutine ARRIVE Flowchart

4.4.2.2 Subroutine DEPART

When a job (event) is set to this subroutine it executes the departure of the job from an equipment item or a work station and then schedules the arrival event of an MH station to carry the job to its next destination. The subroutine checks the size of the queue and if a job is waiting, the subroutine then schedules a departure event of the job from the work station. If the job is processed through an inspection station, it is tested by an inspection team or by machine. When the job shows defects the team decides whether the job requires rework or if it is to be discarded. If the job requires reworking, the subroutine schedules an arrival event of an MH station to carry the job to the previous work station. When the job reaches the last processing operation this subroutine schedules an arrival event of an MH station to carry the job to the unloading station.

If the work station equipment item repair completion event or maintenance completion event is set to the subroutine DEPART, this subroutine executes the equipment repair completion of the equipment at the work station. When maintenance is required, an event is scheduled for the next maintenance time. Then, the subroutine checks the size of the queue and repeats the above procedure for any jobs waiting in the queue.

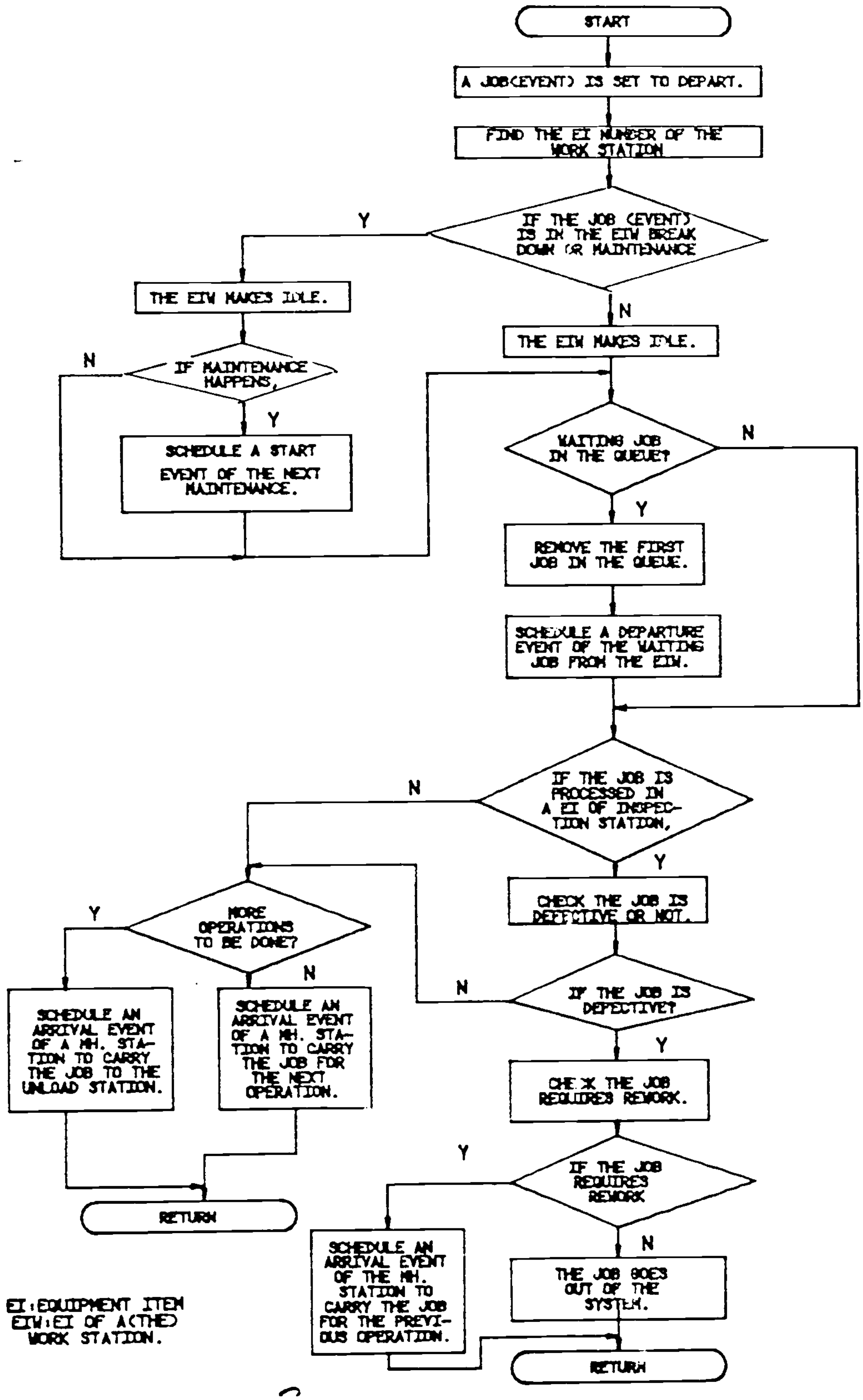


Figure 4.4.3: Subroutine DEPART Flowchart.

4.4.2.3 Subroutines MHSARR and PASS

These subroutines execute two types of departure events of the next MH station to carry a job. First, when a job at an item of equipment of a work station arrives at the MH station, it has no information regarding the MH station. The subroutine MHSARR calls the subroutine DEFINE to find the MH station number and the station number of the next MH station. Second, when a job has already departed for its next MH station, the subroutine MHSARR updates the MH station number from the previous MH station number to the MH station at which it has arrived. Then the subroutine PASS is called to find the next MH station number.

If the MH vehicles are unidirectional and operate in a closed loop, the subroutine PASS schedules a departure event without consideration of other options. However, when the MH vehicles are bidirectional, operating either on a straight line or closed loop system, they check the main program to determine the status of oncoming traffic prior to proceeding. A block condition exists when both stations are unable to proceed because of oncoming vehicles. In this case the MH vehicle which was first blocked has the priority right-of-way. When a job is blocked because of oncoming vehicles the job waits in the queue until the priority vehicle passes. Figures 4.4.4 and 4.4.5 show the flowcharts of the subroutines MHSARR and PASS.

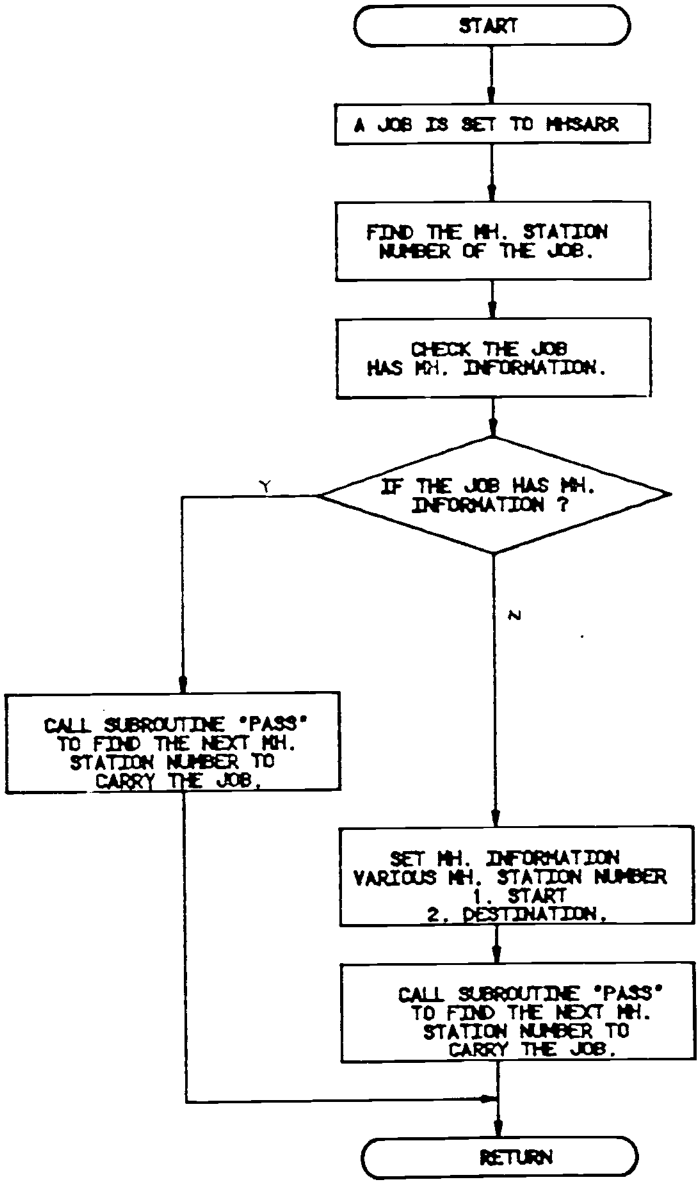


Figure 4.4.4: Subroutine MHSARR Flowchart.

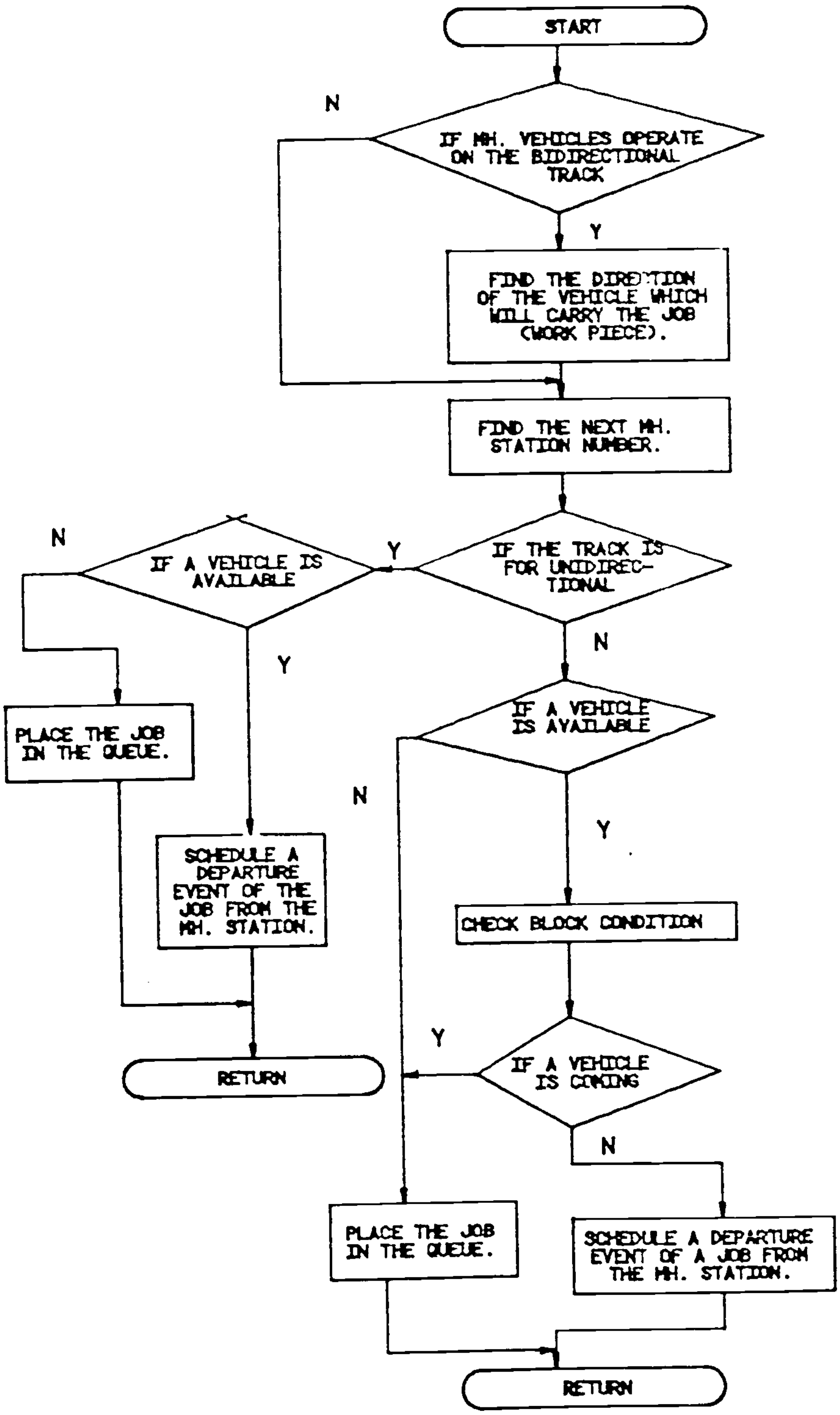


Figure 4.4.5: Subroutine PASS Flowchart.

4.4.2.4 Subroutine MHSDEP

When a job is set to the subroutine MHSDEP, the subroutine executes a departure of the MH station, and then checks the MH station to determine that it has information on its destination. If the MH station at which the vehicle arrives is the destination, an arrival event of a job at an equipment item at the next work station is scheduled. However, if the station is not the destination, an arrival event of the next MH station is scheduled. Additionally, when the job has completed all processing operations, the subroutine schedules an arrival event of the job at an unloading station instead of scheduling an arrival event of the job at an equipment item at the next work station.

The subroutine MHSDEP also checks the job status of the workpiece (i.e. good, rework, or scrap). It the job requires rework, the subroutine schedules an arrival event of the job at the previous work station.

Whenever a job arrives at a MH station, this subroutine checks for the availability of MH vehicles. When a job is waiting in the queue and a vehicle becomes available, this subroutine usually schedules a departure event of the waiting job rom the MH station. However, if MH vehicles are moving on bidirectional track, the subroutine checks for a block condition on the track prior to scheduling a departure event of the MH vehicle for the job waiting in the queue.

The flowchart for this subroutine is shown in Figure 4.4.6

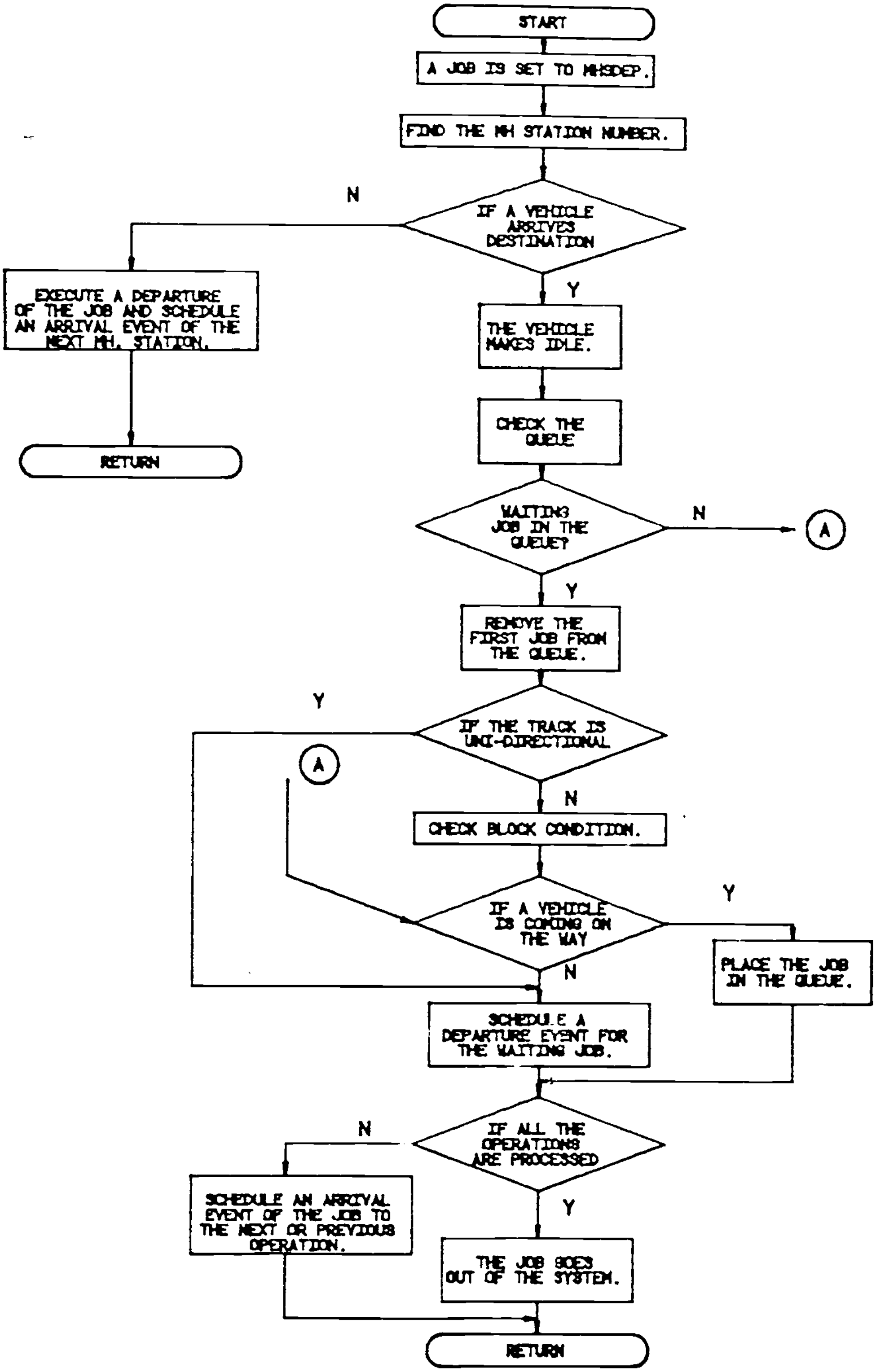


Figure 4.4.64: Subroutine MHSDEP Flowchart.

CHAPTER V

RUNNING THE FMS SIMULATOR

5.1 Introduction

This chapter gives detailed information about the input procedure. Most of the input subroutines ask the user to enter answers to a series of questions to obtain work data, work station data, material handling system data, and other appropriate information.

A number of assumptions and program limitations are explained. Most of them relate to the work stations and material handling system. The form of output results will be discussed and an example will be provided to show how to use this simulator, how to enter FMS data, and how to analyze the results of the FMS simulation.

5.2 Assumptions.

The assumptions on which the FMS simulation is based are listed below and defined.

1) Load and unload stations.
    a) There is one load and unload station.
    b) Every job arrives at the load station and leaves the unload station after completion of processing operations.
    c) When a job arrives at the load station, if a MH vehicle is available it will at once leave the load station to carry the job to its assigned work station.

2) Work stations.

a) All work processing time has the same distribution type.

b) Though alternative work stations are considered, the operation processing times are already entered as input data and are not changed.

c) Work stations, except machine and inspection stations, do not consider alternative work stations.

d) When a job requires rework, the job is returned to the previous work station.

e) When a job is discarded, the vehicle to carry the job returns to the load station.

f) When a job is being processed at work station equipment scheduled for maintenance, the job is completed before maintenance can begin.

3) Material handling vehicles.

a) When the simulation begins, all the vehicles are at the load station.

b) Vehicles carry only one job.

c) The vehicles have the same velocities during simulation time.

d) Vehicle breakdowns are not considered.

5.3 Program Limitations

The program has the following limitations.

1) Jobs.

a) This program can handle up to 10 different job types.

b) This program can handle up to 25 operations within a single job.

2) Work stations.

a) This program can handle up to 10 different kinds of work stations.

b) Each work station consists of up to 10 different items of equipment.

c) The number of separate items of equipment cannot exceed the total of 25 for all work stations.

d) A machine can handle up to 25 tools.

e) In case of breakdown or maintenance, the number of items of equipment which can be replaced by an alternate work station may not exceed 9.

5.4 Input Procedures.

5.4.1 Start the simulator

As mentioned before, input procedures are executed in the subroutine INPUTF. The input procedure consists of thirteen question groups. After the simulation runs, the program first asks the user to choose one of the three input formats (see Figure 5.4.1.1).

Select one of the following options for entering
data values:

1. Enter data through the keyboard.
2. Enter an existing data file.
3. Change the data values of an existing data
   file prior to entering the file.

Figure 5.4.1.1 Input Type

When the user enters "1", a series of questions ask for FMS input data. Details on the series of questions are explained in section 5.4.2.

When the user enters "2", the program asks for the entry of a file name which has already been stored as input data.

When the user enters "3", the program asks for entry of a file name which has already been stored as input data. After entering the file name the program shows the user the following messages.

The values of the variables listed below may be changed in order to correct entry errors or to test different alternatives. If changes are required, enter the project name and date, and make the changes selected from the following type numbers. After making changes save them to the user file and the simulation will run.

TYPE NUMBER	DESCRIPTION
1.	Job arrival time
2.	Queue capacity of work stations
3.	Velocity of MH vehicles
4.	Simulation completion type and time, or number of products to simulate
5.	Number of MH vehicles
6.	Option number
7.	Location of MH stations
8.	Job generation type

How many type numbers do you want to change---?

Figure 5.4.1.2: Change Menu.

5.4.2 Input

Input data are entered in free format data entry style, which is useful for entering multiple data entries in a single line. The question guide lines call for continuous data entry, with each entry delimited by either a space, a comma, or a tab. Input procedures are executed in the following sequence.

1) Input of project information. The subroutine HEAD is called to enter project name (up to 40 characters), user name (up to 20 characters), and date (e.g. mm/dd/yy or mmm/dd/yy, up to 10 characters).

2) Selection of job generation type. The subroutine INPUTF asks the user to enter one of the two job generation types (see Figure 5.4.2.1).

```
This program has two types of job generation. If you
want random event generation, enter the number "1"; if
you are using a predetermined event generation, enter
the number "2".

Enter number ----------?
```

Figure 5.4.2.1 Selection of Event Generation Type for New Job.

When the user selects type "1", the subroutine asks the user to enter the number of job types and different kinds of work station types, job distribution type, and job arrival mean time. For example, three job types, three work station types, job arrival mean time of nine minutes, and exponential distribution type would be entered as 3,3,1,9 (see Figure 5.4.2.2).

This subroutine is used to input system variables. When making multiple entries, the values should be inserted at the positions indicated below (in free format data entry style).

```
Now enter the following data and press RETURN

Number of job type (max = 10)
    :   Number of work station type (min=3, max=10)
    :   :   Job arrival (type, time)
    :   :   :   Type 1)exponential
    :   :   :        2)truncate exponential
    :   :   :        3)uniform
    :   :   :        4)constant
    :   :   :  Mean time
    :   :   :   :
    :   :   :   :
    ?   ?   ?   ?

Enter-----?
```

Figure 5.4.2.2 Input of Simulation Variables.

When the user selects type "2", the subroutine asks the user to enter only the number of the job types and different kinds of work stations. In this case, after entering all of the input data, the subroutine GENEVT is called to enter the entire predetermined schedule.

3) Input of work stations data. The subroutine INPUTF has already been assigned three types of work station types (see Figure 5.4.3.1). When the user wants to use more than three types of work stations, the user must enter the name of the work stations (up to 15 characters). Then, the subroutine asks the user to enter the number of items of equipment per work station.

There are many different kinds of work stations, each of which may have up to 10 different (or similar) items of equipment. In this simulator the load/unload station, machine stations, and inspections stations are already assigned to types 1, 2 and 3, respectively. Therefore, if you require more than 3 type of stations, enter the name of the additional work station types.

Figure 5.4.3.1 Input of Work Station Data.

4) Input of work station options. The subroutine INPUTF asks the user to enter one of the four options.

This subroutine is used to select options for scheduled maintenance and unscheduled breakdowns. Select one of the following options:
1. Neither will be considered.
2. Only unscheduled breakdowns will be considered.
3. Only scheduled maintenance will be considered.
4. Both will be considered.

Figure 5.4.4.1 Input of Work Station Options.

When the user selects option type 2, 3, or 4 the program asks the user to enter available equipment item list of the work station, then the breakdown rate, repair time and its distribution type, regular maintenance schedules and its distribution type.

In this subroutine the breakdown rates and scheduled maintenance plan for work stations are entered. Breakdown rates are defined by real values (0.nnn). Maintenance plans are defined by entering the interval between scheduled maintenance time, the repair time, and its distribution type.

Figure 5.4.4.2 Input of Breakdown Rate and Maintenance Time.

5) Input of material handling type. The subroutine INPUTF asks the user to enter the direction and the type of material handling devices.

Material handling (MH) devices are used to carry a single workpiece between work stations. The moving direction of MH devices on the track can be selected from one of the following:

1. Unidirectional track in a closed loop
2. Bidirectional track in a straight line
3. Bidirectional track in a closed loop

Figure 5.4.5.1 Input of Material Handling Type.

When the user selects type "1" the subroutine MAKESIS is called to enter the number of the nearest station from a work station and its distance. When the user selects type "2" the subroutine MAKESIS is called to enter X and Y coordinates of each station. When the user selects type "3" the subroutine MAKESIS is called to enter the nearest stations in either direction from a work station and the distances between work stations. Finally, the number of material handling vehicles is entered.

6) Input of random variable limits. The subroutine INPUTF asks the user to enter lower limits, upper limits, or both, for specific random variable generation.

> This subroutine is used to define the values of the limits used to generate truncated exponential distributions or uniform distributions. When truncated exponential distribution is used, enter the upper limit of the mean value. When uniform distribution is used, enter its upper and lower limits, which in this program are defined as a fraction of mean value.

Figure 5.4.6.1 Input of Random Variable Limits.

7) Input of simulation completion types. There are three types of simulator completions. When the user selects type "1", the simulation is finished when the simulation clock reaches the completion time. When the user selects type "2", the user enters the number of total jobs instead of entering completion time. In this case, the simulation is finished after all the jobs are processed. When the user selects type "3", the simulation is finished after processing the jobs which have arrived at the system prior to the completion time.

8) Input of work data. The subroutine INPUTF calls the subroutine INPUTJOB to enter the following work data: number of operations in a job, work station type of a job with its operation number, the equipment

number used at the work station, the work time and its distribution type, and the job allocation rate. First the subroutine asks for the entry of the number of operations per job. Then the work station types with the operation number of job which is in process are entered in groups of five numerical entries. (When no entry is required in a category, a zero ["0"] must be entered.) Finally, the user enters the equipment item number used at the work station and work time with the operation number.

After entering the entire job input, the distribution type of work time and job allocation rate are entered.

9) Input of work stations data. The subroutine INPUTMCH is used to enter the following work stations data: the number of tools at each machine station and the tool list in each machine's tool magazine. First, the subroutine asks for entry of the number of tools of the machine station equipment item. Then, the user enters the tool list. Finally, the tool loading time and its type are entered. (All entries also made in groups of five, with a zero ["0"] entered for each category which does not require an entry.)

10) Input tool or equipment number. The subroutine INPUTASK is called to enter tool or equipment numbers with operations numbers. If no tools (or items of equipment) are used, the user enters "0".

11) Input of queue sizes for selection of alternate equipment items at a work station. the subroutine INPUTAVA is called to enter queue sizes at work stations (load/unload, machine, inspect). First, the subroutine asks for entry of the queue size of a load/unload station, then for the machine stations and the inspection stations with equipment item numbers.

12) Input of machine stations defective rate and rework rate. The subroutine INPUTQC is called to enter breakdown rate and rework rate with the machine numbers.

13) Input of MH vehicles. The subroutine INDUTMHS is called to enter the velocity (m/min) of vehicles, loading time for each vehicle, and its distribution type.

5.5 Output

The simulator program displays results through a CRT, USERFILE, and hard copy.

First, output reports copy the following information: simulation variable (simulation completion time and distribution type); the job data (number of operations, job allocation rate); the work station data (machine station tool information, number of work stations); the material handling data (velocities, number of MH vehicles, MH direction type); and the various selections (options, simulation completion type, job generation type) (see Figure 5.5.1).

FMS SIMULATION REPORT

PROJECT:

DATE:
BY:

SYSTEM INPUT

1. Job.

   Job Arrival Time:
   Number of Job Types:
   Number of Work Stations:
   Number of Operations:

   JOB 1:
   JOB 2:
   JOB 3:

   Distribution type of arrival time:
   Distribution type of work time:
   Job allocation rate:

   JOB 1:
   JOB 2:
   JOB 3:

2. Work Station.

   ** Machine stations **

   Number of Item of Equipment:

Machine Number	Number of tools	Loaded tool number

   ** Inspection station **

   Number of item of equipment:

3. Material handling devices.

   MH Direction Type:
   Velocity:
   Number of MH devices:

4. Others.

   Option type:
   Simulation completion type:
   Job generation type:

Figure 5.5.1 Copy of Input Data.

Then, the simulation program shows various summary reports based on the following statistics collected from the simulation library: work station utilization; average time in the queue and average number of queues; and maximum and minimum number of queues. Additionally, a number of blocks, work station transfers, throughput times, and average job delays are computed to analyze the FMS model.

The first report is the queue summary report. This report shows the following information on queues made for work stations and MH stations: average number in the queue; maximum and minimum number of the queue; and average time in the queue (see Figure 5.5.2).

1. QUEUE STATISTICS REPORT

TYPE	AVERAGE NUMBER	MAXIMUM NUMBER	MINIMUM NUMBER	AVERAGE TIME IN THE QUEUE

Figure 5.5.2 Queue Statistics Report.

The second report is the work station statistics report. This report shows work station utilization, number of blocks at the work station, average delays, and number of observations (see Figure 5.5.2

2. WORK STATION STATISTICS REPORT

NO	AVERAGE UTILIZATION	NUMBER OF BLOCKS	AVERAGE DELAY	NUMBER OF OBSERVATIONS

Figure 5.5.3 Work Station Statistics Report.

The third report is the job statistics report. This report shows the following: average job delays; job average throughput times; job maximum and minimum throughput times; the number of generation jobs; rework; and scrap (see Figure 5.5.3)

3. JOB STATISTICS REPORT

AVERAGE	THROUGHPUT TIME			NUMBER OF			
JOB DELAY	AVERAGE	MAXIMUM	MINIMUM	GEN	SCR	REW	FIN

Figure 5.5.4 Job Statistics Report.

5.6 Simulation Run Example

In this section an example of an FMS is simulated. This FMS simultaneously produces four different types of parts, cylindrical heads, water pump housings, brake mountings, and crank cases. The FMS consists of five work stations, including load/unload stations, machine stations for machine processing, inspections stations for measuring processes, washing stations, and surface treatment stations. This system is based loosely on the example provided in Purdom (1983).

The machine stations consist of three "5-axis" machines, each with its own tool magazine. The inspection station has two inspection tables, consisting of manual labor and robotic measuring systems. A number of robots are used as the pick-and-place device between work stations and the material handling system. Each work station has two buffers where jobs can await processing or can await movement to the next work station after work station processing. Five AGVs operating at the same speed are used to transport jobs between work stations. For detailed information, Figures 5.6.1 and 5.6.2 are provided.

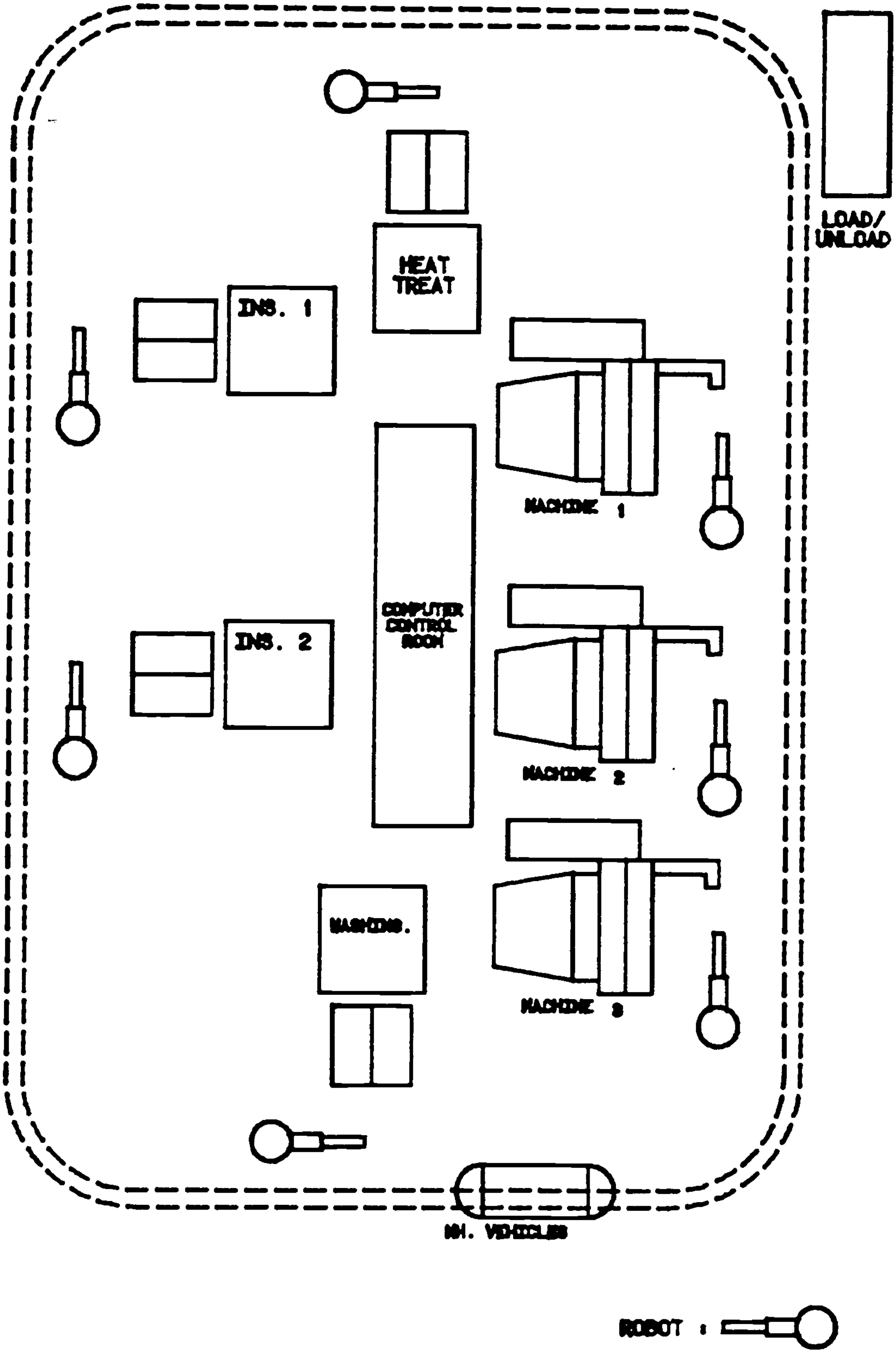


Figure 5.6.1 FMS Layout (Example).

1. SYSTEM INFORMATION.

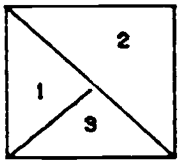
1) JOB TYPE: 4
2) SIMULATION TIME: 1000 MIN
3) SIMULATION COMPLETION TYPE: 1
4) OPTION TYPE: 4
5) MACHINE BREAK DOWN RATE: 1 %
6) DEFECTIVE RATE: 5%
7) REWORK RATE: 50%

2. WORK DATA.

1: WORKSTATION TYPE(M:MACHINE, I=INSPECTION, S:SURFACE TREAT, W:WASHING)

2 WORK TIME.

3. TOOL NUMBER(IF MACHINE STATION).



OP. NO. / JOB TYPE	1	2	3	4	5	ALLOC. RATE.
CYLINDER HEAD.	M1 / 2 / 1	I1 / 3	S1 / 2			.3
WATER PUMP HOUSING.	M1 / 3 / 8	M2 / 3 / 9	I2 / 1.5	S1 / 1	W1 / 1	.3
BRAKE MOUNTING.	M2 / 2 / 5	I1 / 1.5	M3 / 2 / 10	I2 / 1.8		.1
CRANK CASE.	M1 / 2 / 6	W1 / 2	I1 / 2	S1 / 2		.3

3. MACINE DATA.

MACHINE.	TOOL NUMBER IN THE TOOL MAGAZINE.
MACHINE1	1,8,6,7,9,20,10,3,5
MACHINE 2	2,4,8,9,15,18,17,13,5,7
MACHINE 3	1,2,4,8,6,7,9,15,18,17,20,10,3,5

Figure 5.6.2 FMS Input Data Information.

After 1,000 minutes of simulated operation, the results are summarized in the output report (see Figure 5.6.3 and 5.6.4). In summary, a total of 194 jobs completed processing. The statistics variables of the load/unload station show an average of 5.793 jobs awaiting processing and an average of 28.393 job waiting time. The statistics variables for the machine stations reveal that the utilization of machine 1 is higher than other machines. Machine 3 executed 16 jobs as a work station transfer. The statistics variables for the inspection stations reveal that the utilization of inspection table 1 is higher than inspection table 2. Transfers between the two inspection tables occurred once during the simulation.

FMS SIMULATION REPORT

PROJECT:FACTORY

DATE:10/4/86
BY: LEE

SYSTEM INPUT

1. Job.

New job mean arrival time: 5.000
Number of job types: 4
Number of work station types: 5
Number of operations for each job:

JOB 1: 3
JOB 2: 5
JOB 3: 4
JOB 4: 4

Distribution type of arrival time: 2
Distribution type of work time: 2
Job allocation rate:

JOB 1:.300
JOB 2:.300
JOB 3:.100
JOB 4:.300

2. WORK STATION.

** Machine station **

Number of equipment items at the station : 3

Machine Number	Number of tools	Loaded tool number
1	9	1
2	10	2
3	14	1

Figure 5.6.3 Output Report--Part I

```
** Inspection      station **

     Number of equipment items at the station : 2


** SURFACE TREAT  station **

     Number of equipment items at the station : 1


** WASHING           station **

     Number of equipment items at the station : 1
```

3. Material Handling Devices

```
     MH Direction Type : 3
     Velocity            :  50.000
     Number of MH devices: 5
```

4. Others.

```
     Option type : 4
     Simulation completion type: 1
     Job Arrival Event Generation Type: 1
```

Figure 5.6.4 Output Report--Part I (continued)

SUMMARY RESULTS

Current time : 1000.000

1. QUEUE STATISTICS REPORT

QUEUE TYPE	AVERAGE NUMBER	MAXIMUM NUMBER	MINIMUM NUMBER	AVERAGE TIME IN THE QUEUE
MH-Load/Unload	5.793	11	1	28.393
MH-Machine	0.000	0	0	0.000
MH-Machine	0.000	0	0	0.000
MH-Machine	0.000	0	0	0.000
MH-Inspection	0.000	0	0	0.000
MH-Inspection	0.000	0	0	0.000
MH-SURFACE TREAT	0.000	0	0	0.000
MH-WASHING	0.000	0	0	0.000
WS-Machine	0.252	4	0	1.597
WS-Machine	0.011	2	0	0.133
WS-Machine	0.001	1	0	0.022
WS-Inspection	0.323	5	0	2.297
WS-Inspection	0.397	4	0	5.025
WS-SURFACE TREAT	0.053	2	0	0.303
WS-WASHING	0.017	2	0	0.139

2. WORK STATION STATISTICS REPORT

MATERIAL HANDLING STATIONS

NO	AVERAGE UTILIZATION	NUM. OF BLOCK	AVERAGE DELAY	NUMBER OF OBSERVATIONS
1	0.797	0	28.393	201
2	0.446	0	0.000	184
3	0.719	0	0.000	92
4	0.869	0	0.000	35
5	0.557	0	0.000	143
6	0.675	0	0.000	80
7	0.613	0	0.000	176
8	0.757	0	0.000	125

Figure 5.6.4 Output Report--Part II

Machine STATION

E.I. NO	AVERAGE UTILIZATION	NUM. OF ST. CHANGE	AVERAGE DELAY	NUMBER OF OBSERVATION
1	0.308	2	1.597	158
2	0.167	19	0.133	85
3	0.070	16	0.022	34

Inspection STATION

E.I. NO	AVERAGE UTILIZATION	NUM. OF ST. CHANGE	AVERAGE DELAY	NUMBER OF OBSERVATION
1	0.263	1	2.297	139
2	0.105	5	5.025	79

SURFACE TREAT STATION

E.I. NO	AVERAGE UTILIZATION	NUM. OF ST. CHANGE	AVERAGE DELAY	NUMBER OF OBSERVATION
1	0.209	0	0.303	176

WASHING STATION

E.I. NO	AVERAGE UTILIZATION	NUM. OF ST. CHANGE	AVERAGE DELAY	NUMBER OF OBSERVATION
1	0.146	0	0.139	125

3. JOB STATISTICS REPORT

JOB TYPE	AVERAGE DELAY.	THROUGHPUT TIME AVERAGE.	MAXIMUM.	MINIMUM.	NUMBER OF GEN.	SCR.	REW.	FIN.
1	14.485	51.415	125.352	26.959	56	0	1	53
2	17.071	58.670	130.376	33.266	57	2	0	54
3	14.683	50.316	98.943	19.679	20	0	0	19
4	13.112	50.580	96.323	15.361	74	0	0	68

Figure 5.6.4 Output Report--Part II (continued)

5.7 Analysis of the Output Data

A simulator usually reflects the dynamic behavior of a system over time. A simulation model is built to provide results which resemble the output from the real system. A simulation model usually runs once, and then treats the results as the true answers for the model. Most simulation models use random variable numbers for various simulation variables. If the random variable numbers have a large variance, the output generates different results from the corresponding true answers. Many simulation studies have been developed in modeling and programming, but simulation output data analysis methods have not been equally well-developed.

To obtain more useful results, the user may perform a number of simulations or use a longer simulation time period. In either case the user may use average values and variances of output data as better estimated results. In addition, the user may perform sensitivity analyses and construct plots and tables for output data. Sensitivity analysis may be used as a tool to verify the output; it is executed by systematically changing input data. Plots and tables are used as an analysis tool, checking the proper function of a number of simulation variables.

The FMS simulator developed in this study has the following abilities to support output data: 1) It can run a number of simulations by sequentially changing random number seeds; 2) It can run a longer period of simulation time without changing input parameters; and 3) It can execute sensitivity analysis by calling the subroutine INPUTCHC.

CHAPTER VI

SUMMARY AND CONCLUSION

Flexible manufacturing systems have brought about great advances in manufacturing technology. Particularly, FMS provide an efficient solution for productivity problems which have been considered insoluble for low and medium-volume production. However, an FMS requires substantial set-up costs since an FMS generally consists of a number of NC or CNC machines, and a computerized material handling systems. However, to increase productivity by other means may be equally expensive and the FMS will in the long run easily repay the investment.

The simulator program can be used as a simulation tool to design and operate an FMS. The simulator in this study uses a question and answer format to allow performance of manufacturing simulation by personnel with limited simulation and/or programming backgrounds.

The simulator allows the designer to evaluate various alternative systems by testing the different parameters which affect the design, scheduling, and control of flexible manufacturing: the location of work stations, job routing and combinations, and material handling systems. The simulator can be used to predict a system's productive capacity and the utilization effectiveness of FMS production components. This simulator, because of the size of the program and its relative ease of application, is particularly suited to small-scale production systems.

As subjects for further study in conjunction with the simulator, various graphic input devices, such as the mouse, the digitizer, and the tablet, could effectively be used to easily enter much of the input data. Graphic techniques with graphic animation would provide realistic and dynamic views of FMS processing through use of high resolution CRTs. In addition, further research should be devoted to the simulation of real processing time rather than the probability distribution of machine processing times. The results would provide more accurate work processing times.

Finally, other simulator programs should be prepared to integrate parts storage configurations and more complicated material handling systems (network routing, the use of various vehicles, and multiple parts carrying systems), in order to provide an integrated FMS simulation of the entire manufacturing process.

BIBLIOGRAPHY

Arbel, Ami and Seidmann, Abraham. "Performance Evaluation of FMS." IEEE Transactions on System, Man and Cybernetics (SMC-14), Vol. 1, 4, Jul/Aug 1984.

Bahram, Keramiti and Kelly-Sacks, Christine M. "Simulation and Animation of an Assembly System." Proceedings of the 1983 Winter Simulation Conference, ACM, SIG SIM, IEE,IEEE, Arlington, VA, December, 1983, pp. 659-661.

Bernard, John David and Robinson, Pamela Dee. Flexible Manufacturing Simulation. Paper presented at the AUTOFACT 6 Conference, SME (MS 84-726), Anaheim, CA, October, 1984, pp. 1-17.

Carrier, A. S. and E. Adami. "Introducing FMS by Simulation." Proceedings of the 2nd International Conference on FMS (pp. 229-238). London: IFS (Publications) Ltd. and North Holland Publishing Company, 1983.

Chang, Yih-Long, Sullivan, Robert S., and Wilson, James R. "Using SLAM to Design the Material Handling System of a Flexible Manufacturing System." International Journal of Production Research, Vol. 24, 1, 1986, pp. 15-26.

Crite, G. D., Mills, R. I. and Talavage, J. J. "PATHSIM, A Modular Simulator for an Automatic Tool Handling System Evaluation in FMS." Journal of Manufacturing Systems, Vol. 4, 1, 1984, pp. 15-22.

Cutcosky, Mark R., Fussell, Paul S., and Milligan, Robert. "The Design of a Flexible Machining Cell for Small Batch Production." Journal of Manufacturing Systems, Vol. 3, 1, 1984, pp. 39-59.

Diersh, Kurt H. and Malstrom, Eric M. "Physical Simulator "Analyzes Performance of FMS." Industrial Engineering, June 1985, pp. 66-77.

Duersch, Ralph D. and Layman, M. A. "A Graphic Workflow Simulator." Proceedings of the Summer Computer Simulation Conference, Society of Computer Simulation, Vancouver, Canada, July, 1983.

Duersch, Ralph D. and Layman, M. A. A Graphic Workflow Simulator. Paper presented at the 17th Annual Simulation Symposium, IEEE, Tampa, FL, 1984, pp. 37-48.

Editors of American Machinist. Computers in Manufacturing. New York: McGraw-Hill, 1983.

Elmaraghy, Hoda A. "Simulation and Graphical Animation of Advanced Manufacturing Systems." Journal of Manufacturing Systems, Vol 1, 1, 1982, pp. 53-63.

Elmaraghy, Hoda A. and Ho, N. C. "Automated Tool Management in FMS." Journal of Manufacturing Systems, Vol. 4, 1, 1984, pp. 1-13.

Elmaraghy, Hoda A. and Ho, N. C. "Simulation of Flexible Manufacturing Systems." Computers in Mechanical Engineering, Aug 1982, pp. 16-23.

"FMS, A Boom You'll Guide." Production Engineering, Feb 1986, pp. 38-60.

Fox, Kenneth. Simulation for Design and Scheduling of Flexible Manufacturing Systems. Paper presented at the AUTOFACT 4 Conference, SME (MS 82-419), Philadelphia, PA, November, 1982.

Gatelman, Catherine Dupont. "A Survey of Flexible Manufacturing Systems." Journal of Manufacturing Systems, Vol. 1, 1, 1983, pp. 1-16.

Ham, Inyong, Hitomi, Katsundo, and Yoshida, Teruhiko. Group Technology. Boston: Kluwer-Nijhoff Publishing, 1985.

Hartley, John. FMS at Work. London: IFS (Publications) Ltd., 1984.

Ingersoll Engineer. The FMS Report. London: IFS (Publications) Ltd., 1982.

Kinemia, Joseph and Gershwin, Stanley B. "Flow Optimization in Flexible Manufacturing Systems." International Journal of Production Research, Vol. 23, 1, 1985, pp. 81-96.

Kusiak, Andrew. "Conveyor Systems for Flexible Assembly Operation." Material Handling, Jul 1985, pp. 62-70.

Kusiak, Andrew. "Flexible Manufacturing Systems: A Structural Approach." International Journal of Production Research, Vol. 23, 6, 1985, pp. 1059-1077.

Law, Averill M. and Kelton, W. David. Simulation Modeling and Analysis. New York: McGraw-Hill Book Company, 1982.

Lenz, John E. FMS Design Using Microcomputer Graphics. Paper presented at the AUTOFACT 5 Conference, SME (MS 83-743), Detroit, MI, November, 1983.

Lenz, John E. "MAST: A Simulation Tool for Designing Computerized Metalworking Factories." Simulation, Feb 1983, pp. 51-58.

Maramatsu, Rintaro, Ishii, Kazuyashi, and Takahashi, Katsuhito. "Some Ways to Increase Flexibility in Manufacturing Systems." International Journal of Production Research, Vol. 23, 4, 1985, pp. 691-703.

Merchant, M. Eugene. "The FMS--A Stepping Stone to CIM." CIM Review, Spring, 1985, pp. 3-7.

Pritsker, A. A. B. The GASP IV Simulation Language. New York: John Wiley & Sons, 1974.

Pritsker, A. A. B. and Pegden, Claude Dennis. Introduction to Simulation and SLAM. New York: John Wiley & Sons, 1979.

Pun, L., Doumeingts, G., and Bourely, A. "The GRAI Approach to the Structural Design of Flexible Manufacturing Systems." International Journal of Production Research, Vol. 23, 6, 1985, 1197-1215.

Purdom, P. B. "The Citroen Flexible Manufacturing Cell." Proceedings of the 2nd International Conference on FMS (pp. 93-103). London: IFS (Publications) Ltd. and North Holland Publishing Company, 1983.

Raju, Venkitaswamy. "The Role of Robotics in FMS." Robotics Age, July 1984, pp. 31-34.

Ranky, Paul. The Design and Operation of FMS. London: IFS (Publications), 1983.

Raouf, A. and Ahmad, S. I. Flexible Manufacturing. New York: Elsevier, 1985.

Rembold, Ulrich, Blume, Christian, and Dillmann, Ruediger. Computer Integrated Manufacturing Technology and Systems. New York and Basel: Marcel Dekker, Inc., 1985.

Suri, Rajan and Hildebrant, Richard R. "Modeling Flexible Manufacturing Systems Using Mean Value Analysis." Journal of Manufacturing Systems, Vol. 3, 1, 1984, pp. 27-38.

Wilhelm, W. E. and Shin, Hyun Myung. "Effectiveness of Alternate Operations in a Flexible Manufacturing System." International Journal of Production Research, Vol. 23, 1, 1985, pp. 65-79.

Zisk, Burton I. "The Appeal of Flexible Manufacturing." CIM Review, Fall 1984, pp. 67-71.

APPENDIX

APPENDIX

COMPUTER SIMULATOR PROGRAM

```
      PROGRAM MAINFMS
c
c ---- This program is made for FMS simulator ------
c -----      Written by SEOUK JOO LEE
c -----      Date:JUN 28 1986                      ------
c -----      update:Oct 3 1986                     ------
c
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25)
      INTEGER*2 NVALUE,JOBT,SERV,TYEVT,NCHWS(45),NBLOCK(25)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      REAL*4 MARRVT,PROBD(10),LTIME,TR,TREM
      CHARACTER PROJNAME*40,USER*20,DATE*10,NAMESERV(10)*15,CONT*1
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/INSP/INSPEC
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/RAND/NVALUE,PROBD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C ---- Initialize common variables.-------
C
 1000 CALL INITCO
      CALL INITLK
      TREM=0
C
C ---- Input FMS data through subroutine INPUTF.----
C
      CALL INPUTF(PROJNAME,USER,DATE,NAMESERV)
C
C ---- Specify value for subroutine RANDI
c
      NVALUE=NTYP
C
C ---- If FMS have a predetermined schedule,the subroutine
c      GENEVT be called.Otherwise the first event is entered
C
      IF(TYEVT.EQ.2) THEN
```

```
            CALL GENEVT
               GO TO 70
        END IF
 C
        CALL DISTRI(2,TYARR,MARRVT,TAVE,TUNI,TR)
        CALL RANDI(2,JOBT)
        CALL FINSERV(JOBT,1,ROUTE,NOSERV,SERV)
        NUTY(JOBT)=1
        NTOT=1
 C
        CALL SCHEDUL(TR,1,JOBT,1,SERV,0,0,0,1,TR)
 C
 C---- If simulation completion type is 2 or 3,a arbitrary completion event
 c      will be needed. Otherwise a completion event is entered.
 C
        IF(FINTYP.GE.2) THEN
            TR=1.0E+10
            CALL SCHEDUL(TR,5,0,0,0,0,0,0,0,0.)
        ELSE
            CALL SCHEDUL(LENGTH,5,0,0,0,0,0,0,0,0.)
        END IF
 C
 C ---- Determining the next event to occur and updating simulation clock.
 C
  70    CALL TIMING
 C
 C ---- Call the appropriate subroutine in accordance with the event type
 C
        GO TO(80,90,100,110,120),NEXT
  80    CALL ARRIVE
        GO TO 70
  90    CALL DEPART
        GO TO 70
  100   CALL MHSARR
        GO TO 70
  110   CALL MHSDEP
c
        IF(FINTYP.EQ.1) GO TO 70
        TREM=TIME
        GO TO 70
  120   CALL REPORT(TREM,PROJNAME,USER,DATE,NAMESERV)
        WRITE(6,211)
  211   FORMAT(//,1X,'If you want to continue this simulator,'/,
      $1x,'press the key[C] and RETURN. ???',$)
        READ(5,1) CONT
  1     FORMAT(A1)
        IF(CONT.EQ.'C'.OR.CONT.EQ.'c') GO TO 1000
        END
C
C
```

```
      SUBROUTINE INITCO
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25)
      INTEGER*2 NCHWS(45),NBLOK(25),TYEVT
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/INSP/INSPEC
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/RAND/NVALUE,PROBD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C *** INITIALIZE COMMON VARIABLES
C
      LENGTH=0.
      LTIME=0.
      VEL=0.
      SMHS=0.
      NVALUE=0
      NTOT=0
      TIREC=0.
      TYEVT=0
      MARRVT=0.
      NTYP=0
      NOSERV=3
      NOSTA=0
      MAXATR=10
      OPTION=1
      FINTYP=0
      MHSDIR=0
      NPRO=0
      TYARR=0
      TYSER=0
      TYMAT=0
      TYLT=0
      NOWS=0
      TAVE=3.
```

```
      LOAD=0.
      MAT1=0.
      MAT2=0.
      TYREC=0.
      TREC=0.
C
      DO 100 I=1,10
      TRNSFR(I)=0.
      NTASKS(I)=0
      NUTY(I)=0
      NREW(I)=0
      NSCR(I)=0
      NSGRO(I)=0
      PROBD(I)=0.
      RDEF(I)=0.
      RREW(I)=0.
      NFIN(I)=0
      NTOL(I)=0
      LTOL(I)=0
C
      DO 200 J=1,10
      AVAMA(I,J)=0
      NIDL(I,J)=0
      INSPEC(I,J)=0
 200  CONTINUE
 100  CONTINUE
C
      DO 300 I=1,10
      DO 400 J=1,25
      MACHT(I,J)=0
      CTASK(I,J)=0
      NBLOK(J)=0
      DO 500 K=1,10
      ROUTE(I,J,K)=0
      MSERVT(I,J,K)=0.
 500  CONTINUE
 400  CONTINUE
 300  CONTINUE
C
      DO 600 I=1,25
      CORDX(I)=0
      CORDY(I)=0
      NOMHS(I)=0
      DO 700 J=1,25
      LDIS(I,J)=0
      IBLOK(I,J)=0
      RFLOW(I,J)=99
 700  CONTINUE
 600  CONTINUE
      DO 800 I=1,45
```

```
      NCHWS(I)=0
      NGWS(I)=0
 800  CONTINUE
      DO 900 I=2,3
      TUNI(I-1)=0.0
      DO 950 J=1,10
      FAILR(I,J)=0.0
      QS(I,J)=0
 950  CONTINUE
 900  CONTINUE
      RETURN
      END
C
      SUBROUTINE GENEVT
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 TYARR,TYSER,TYMAT,TYREC,TYLT
      INTEGER*2 NVALUE,JOBT,SERV,TYEVT
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/RAND/NVALUE,PROBD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
1     FORMAT(6(/))
c
      WRITE(6,1)
      WRITE(6,11)
 11   FORMAT(//,
     $'  ______________________________________________________________ '/,
     $' |                                                              |'/,
     $' |  When an FMS has a predetermined manufacturing schedule     |'/,
     $' |  the following informations ,such as the number of job,     |'/,
     $' |  job type,the start date are required. First,enter the      |'/,
     $' |  number of jobs ,and then enter the job type and its        |'/,
     $' |  start date.                                                |'/,
     $' |______________________________________________________________|')
C
      WRITE(6,10)
 10   FORMAT(/,1X,'- Enter number of jobs ---?',$)
      READ(5,*) NEVT
c
      DO 100 I=1,NEVT
      WRITE(6,21) I
 21   FORMAT(//,
     $2x,'- Enter job type and its start date.'/,
     $2x,'                :              :'/,
     $2x,'                ?              ?'/,
```

```
     $'enter--?',$)
c
      READ(5,*) JOBT,TR
      CALL FINSERV(JOBT,1,ROUTE,NOSERV,SERV)
      CALL SCHEDUL(TR,1,JOBT,1,SERV,0,0,0,1,TR)
 100  CONTINUE
C
      TR=1.0E+10
      CALL SCHEDUL(TR,5,0,0,0,0,0,0,0,0.)
C
      RETURN
      END
c
      SUBROUTINE INPUTF(PROJNAME,USER,DATE,NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25)
      INTEGER*2 NVALUE,JOBT,SERV,TYEVT,TERM,NCHWS(45),NBLOK(25)
      INTEGER*4 IX,IY
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC,TAVE
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      REAL*8 RN
      CHARACTER PROJNAME*40,USER*20,DATE*10,NAMESERV(10)*15,FILENM*15
      CHARACTER COPY*1
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/INSP/INSPEC
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/OUT/NUMMHS
      COMMON/RAND/NVALUE,PROBD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
c *** Define format ****
C
 1    FORMAT(//,1x,A,$)
 2    FORMAT(//,1X,A)
 3    FORMAT(A40,A20,A10,10(A15,1X))
 4    FORMAT('JOB1**',8(I4,2X))
 5    FORMAT('NSGRO*',10(I2,2X))
```

```
 6    FORMAT('JOB3**',5(F8.2,2X))
 7    FORMAT('ROUTE*',10(I2,2X))
 8    FORMAT('MSERVT',10(F6.4,1X))
 9    FORMAT('CTASK*',25(I2,2X))
 10   FORMAT('NTASKS',10(I2,2X))
 11   FORMAT('PROBD*',10(F6.2,1X))
 12   FORMAT('RDEF**',10(F4.3,2X))
 13   FORMAT('RREW**',10(F4.3,2X))
 14   FORMAT('MACHT*',25(I2,2X))
 15   FORMAT('NTOL**',10(I2,2X))
 16   FORMAT('LTOL**',10(I2,2X))
 17   FORMAT('AVAMA*',9(I2,2X))
 18   FORMAT('INSPEC',9(I2,2X))
 19   FORMAT('COORDX',2(I5,2X,I5))
 20   FORMAT('LDIS**',25(I5,2X))
 21   FORMAT('RFLOW*',25(I2,2X))
 22   FORMAT(I2)
 23   FORMAT(6X,8(I4,2X))
 24   FORMAT(6X,6(I2,2X))
 25   FORMAT(6X,5(F8.2,2X))
 26   FORMAT(6X,10(I2,2X))
 27   FORMAT(6X,10(F6.2,1X))
 28   FORMAT(6X,25(I2,2X))
 29   FORMAT(6X,10(F4.3,2X))
 30   FORMAT(6X,9(I2,2X))
 31   FORMAT(6X,2(I5,2X,I5))
 32   FORMAT(6X,25(I5,2X))
 33   FORMAT('FAILR*',10(F6.4,1X))
 34   FORMAT(A15)
 35   FORMAT(//,1X,A,A15,A,$)
 36   FORMAT(I2)
 37   FORMAT(//,1X,A,I2,A,$)
 38   FORMAT(//,1X,A/,' ',$)
 39   FORMAT(//,1X,'Now enter the following data and press RETURN')
 40   FORMAT(/,'??? Your input is mistyped,please try again.')
 41   FORMAT('*QS***',10(I2,2X))
 42   FORMAT('*NIDL*',10(I2,2X))
 43   FORMAT('*QS1**',2(I2,2X))
 44   FORMAT(6X,2(I2,2X))
 45   FORMAT(A1)
 46   FORMAT(10(/))
 47   FORMAT('*LENG*',F16.3,2X,F6.3)
 48   FORMAT(6X,F16.3,2X,F6.3)
 49   FORMAT(6X,10(F6.4,1X))
C
C ** DEFINE INPUT FORMAT **
C
      DO 50 I=1,10
      NAMESERV(I)='
 50   CONTINUE
```

```
C
      WRITE(6,46)
      WRITE(6,101)
 101  FORMAT(/,
     $'________________________________________________________________'/,
     $' |                                                              |'/,
     $' | Select one of the following options for entering data        |'/,
     $' | values:                                                      |'/,
     $' |                                                              |'/,
     $' |         1. Enter data through the keyboard.                  |'/,
     $' |         2. Enter an existing data file.                      |'/,
     $' |         3. Change the data values of an existing             |'/,
     $' |            date file prior to entering the file.             |'/,
     $' |______________________________________________________________|'/,
     $'                                                                 '/,
     $' - Enter number  -----?',$)
      READ(5,22) IT
c
      GO TO(1000,2000,2000) IT
c
 1000 CALL HEAD(PROJNAME,USER,DATE)
c
      WRITE(6,46)
      WRITE(6,112)
 112  FORMAT(///,
     $'______________________________________________________________'/,
     $' |                                                            |'/,
     $' |  This program has the ability to simulate both random      |'/,
     $' |  and predetermined arrival event generations for new       |'/,
     $' |  job. If you want random event generation,enter the        |'/,
     $' |  the number "1"; if you want predetermined event           |'/,
     $' |  generation,enter the number "2".                          |'/,
     $' |____________________________________________________________|')
c
      WRITE(6,1) ' - Enter  number-------?'
      READ(5,*) TYEVT
C
      IF(TYEVT.EQ.1.OR.TYEVT.EQ.2) GO TO 115
      WRITE(6,40)
      READ(5,*) TYEVT
C
 115  IF(TYEVT.EQ.1) GO TO 120
      WRITE(6,145)
 145  FORMAT(//,1X,'Enter number of job types and work stations.'///,
     $6X, 'Number of job types.(max=10)'/,
     $6x, '  :        Number of work station types (min=3,max=10)'/,
     $6x, '  :           :'/,
     $6x, '  ?           ?'/,
     $'--?',$)
      READ(5,*) NTYP,NOSERV
```

```
      GO TO 135
C
C
 120  WRITE(6,161)
 161  FORMAT(//////,
     $'  _____________________________________________________________'/,
     $' |                                                             |'/,
     $' | This subroutine is used to input system variables.          |'/,
     $' | When making multiple entries, the values should be          |'/,
     $' | inserted at the positions indicated below(in free           |'/,
     $' | format data entry style).                                   |'/,
     $' |_____________________________________________________________|')
C
C
      WRITE(6,39)
      WRITE(6,221)
 221  FORMAT(///,
     $6X,' Number of job types. (max=10)'/,
     $6x,'    :   Number of work station types (min=3,max=10)'/,
     $6x,'    :      : New job arrival{type,time}'/,
     $6x,'    :      :      :  Type 1) exponential'/,
     $6x,'    :      :      :        2) truncate exponential'/,
     &6x,'    :      :      :        3) uniform'/,
     $6x,'    :      :      :        4) constant'/,
     $6x,'    :      :      :  Mean time.'/,
     $6X,'    :      :      :      :'/,
     $6X,'    :      :      :      :'/,
     $6X,'    ?      ?      ?      ?'/,
     $'enter-?',$)
C
      READ(5,*) NTYP,NOSERV,TYARR,MARRUT
C
C
 135  NAMESERV(1)='Load/Unload'
      NAMESERV(2)='Machine'
      NAMESERV(3)='Inspection'
C
      WRITE(6,46)
      WRITE(6,231)
 231  FORMAT(/,
     $'  _____________________________________________________________'/,
     $' |                                                             |'/,
     $' |  There are many different kinds of work stations,           |'/,
     $' |  each of which may have up to 10 different(similar)         |'/,
     $' |  items of equipment. In this simulator the load/unload      |'/,
     $' |  station,machine station, and inspection station are        |'/,
     $' |  already assigned to types 1,2, and 3,respectively.         |'/,
     $' |  Therefore, if you require more than 3 types of the         |'/,
     $' |  stations, enter the name of the additional work station    |'/,
     $' |  types.                                                     |'/,
```

```
     $' |_______________________________________________________________|')
C
      J=4
 70   IF(NOSERV.LE.3) GO TO 80
      WRITE(6,37) '-Enter the name of work station type ',J,' ---?'
      READ(5,34) NAMESERV(J)
      J=J+1
      IF(J.GT.NOSERV) GO TO 80
      GO TO 70
C
 80   DO 90 I=2,NOSERV
      WRITE(6,35) '- Enter number of items of equipment of '
     $,NAMESERV(I),'station. '
      READ(5,*) NSGRO(I)
 90   CONTINUE
C
C ---- calculate # of work station and  MH station .-----
c
      N=0
      DO 110 I=1,NOSERV
      N=N+NSGRO(I)
      DO 127 J=1,NSGRO(I)
      NIDL(I,J)=1
 127  CONTINUE
 110  CONTINUE
      NOWS=N
      NOSTA=N+1
C
      WRITE(6,46)
      WRITE(6,51)
 51   FORMAT(///,
     $' _______________________________________________________________'/,
     $' |                                                               |'/,
     $' |  This subroutine is used to select options for scheduled     |'/,
     $' |  maintenance and unscheduled breakdowns. Select one of        |'/,
     $' |  the following options:                                       |'/,
     $' |     1. Neither will be considered.                            |'/,
     $' |     2. Only unscheduled breakdowns will be considered.        |'/,
     $' |     3. Only scheduled maintenance will be considered.         |'/,
     $' |     4. Both  will be considered.                              |'/,
     $' |_______________________________________________________________|'/)
c
      WRITE(6,1) '-Enter number-----?'
      READ(5,*) OPTION
C
      CALL INPUTAVA(1,NAMESERV)
      CALL INPUTOPT(NAMESERV)
c
c
      WRITE(6,46)
```

```
      WRITE(6,61)
 61   FORMAT(///,
     $'  ____________________________________________________________  '/,
     $' |                                                            |'/,
     $' |  Material Handling(MH) devices are used to carry a         |'/,
     $' |  single workpiece between workstations. The moving         |'/,
     $' |  direction of MH devices on the track can be selected      |'/,
     $' |  from one of the following:                                |'/,
     $' |            1. Unidirectional track in a closed loop        |'/,
     $' |            2. Bidirectional track in a straighted line     |'/,
     $' |            3. Bidirectional track in a closed loop         |'/,
     $' |____________________________________________________________|'/)
c
      WRITE(6,1) '-Enter number -------? '
      READ(5,*) MHSDIR
c
      IF(MHSDIR.GE.1.AND.MHSDIR.LE.3) GO TO 160
      WRITE(6,40)
      READ(5,*) MHSDIR
 160  CALL MAKESIS(MHSDIR,NAMESERV)
C
      WRITE(6,46)
      WRITE(6,1) '- Enter number of MH devices -----?'
      READ(5,*) NOMHS(1)
      NUMMHS=NOMHS(1)
c
      WRITE(6,46)
      WRITE(6,71)
 71   FORMAT(//,
     $'  ____________________________________________________________  '/,
     $' |                                                            |'/,
     $' |  This subroutine is used to define  the value of           |'/,
     $' |  the limits generated when truncated exponential           |'/,
     $' |  distribution or uniform distribution are used.            |'/,
     $' |  When truncated exponential ditribution is used, enter     |'/,
     $' |  the upper limit of the mean value. When uniform           |'/,
     $' |  distribution is used,enter its upper and lower limits,    |'/,
     $' |  which in this program are defined as a fraction of        |'/,
     $' |  mean value.                                               |'/
     $' |____________________________________________________________|')
C
      WRITE(6,1) '- Enter the upper limit of Truncated exponential',
     $'distribution--?'
      READ(5,*) TAVE
      WRITE(6,1) '- Enter the lower limit of Uniform distribution--?'
      READ(5,*) TUNI(1)
      WRITE(6,1) '- Enter the upper limit of Uniform distribution--?'
      READ(5,*) TUNI(2)
C
      IF(TYEVT.EQ.2) GO TO 4500
```

```
      WRITE(6,46)
      WRITE(6,146)
 146  FORMAT(//,1X,'- Enter simulation completion type------?'/,
     $1x,     '          1. Simulation is finished when the simulation'/,
     $1x,     '             clock reachs its completion time'/,
     $1x,     '          2. Simulation is finished after all the '/,
     $1x,     '             jobs are processed'/,
     $1x,     '          3. Simulation is finished after processing the job',
     $/1x,     '              which have arrived at the system prior to the'/,
     $1x,     '             system.'/,
     $//,'- Enter number and the completion time,or number of jobs---?'/,
     $  '                   :             :'/,
     $  '                   ?             ?'/,
     $1x,'--------?',$)
c
      READ(5,*) FINTYP,TERM
      IF(FINTYP.NE.2) THEN
          LENGTH=FLOAT(TERM)
          NPRO=0
      ELSE
          NPRO=TERM
      END IF
c
c
 4500 CALL INPUTJOB(NAMESERV)
      CALL INPUTMCH(NAMESERV)
      CALL INPUTASK(NAMESERV)
      CALL INPUTAVA(5,NAMESERV)
      CALL INPUTQC(NAMESERV)
      CALL INPUTMHS
C
C
c
 5000 WRITE(6,2) '-Copy your data to user file.'
      WRITE(6,1) '-Enter file name for your file(b:cccccc.dat)---?'
      READ(5,34) FILENM
c
      OPEN(80,FILE=FILENM,STATUS='NEW')
c
      WRITE(80,3) PROJNAME,USER,DATE,(NAMESERV(I),I=1,10)
      WRITE(80,4) NTYP,NOSERV,FINTYP,MHSDIR,TYARR,TYSER,TYMAT,TYREC
      WRITE(80,43) QS(1,1),TYTOL
      WRITE(80,4) TYLT,OPTION,TYEVT,NOWS,NOSTA,NOMHS(1),NPRO,NTOT
      WRITE(80,6) VEL,SMHS,MAT1,TUNI(1),TUNI(2)
      WRITE(80,6) MAT2,TREC,MARRVT,TIREC,LTIME
      WRITE(80,47) LENGTH,TAVE
c
      WRITE(80,5) (NSGRO(I),I=1,10)
      WRITE(80,10) (NTASKS(I),I=1,10)
      DO 100 I=1,NTYP
```

```
      DO 200 J=1,NTASKS(I)
      WRITE(80,7) (ROUTE(I,J,K),K=1,10)
 200  CONTINUE
 100  CONTINUE
C
      DO 150 I=1,NTYP
      DO 250 J=1,NTASKS(I)
      WRITE(80,8) (MSERVT(I,J,K),K=1,10)
 250  CONTINUE
 150  CONTINUE
C
      DO 300 I=1,NTYP
      WRITE(80,9) (CTASK(I,J),J=1,25)
 300  CONTINUE

C
      WRITE(80,11) (PROBD(I),I=1,10)
C
      DO 330 I=2,3
      WRITE(80,33) (FAILR(I,J),J=1,10)
 330  CONTINUE
C
      DO 340 I=2,3
      WRITE(80,41) (QS(I,J),J=1,10)
 340  CONTINUE
C
      WRITE(80,12) (RDEF(I),I=1,10)
      WRITE(80,13) (RREW(I),I=1,10)
C
      DO 350 I=1,NSGRO(2)
      WRITE(80,14) (MACHT(I,J),J=1,25)
 350  CONTINUE
C
      WRITE(80,15) (NTOL(I),I=1,10)
      WRITE(80,16) (LTOL(I),I=1,10)
C
      DO 400 I=1,NSGRO(2)
      WRITE(80,17) (AVAMA(I,J),J=1,9)
 400  CONTINUE
C
      DO 450 I=1,NSGRO(3)
      WRITE(80,18) (INSPEC(I,J),J=1,9)
 450  CONTINUE
C
      DO 550 I=1,NOSTA
      WRITE(80,20) (LDIS(I,J),J=1,25)
 550  CONTINUE
      DO 600 I=1,NOSTA
      WRITE(80,21) (RFLOW(I,J),J=1,25)
 600  CONTINUE
```

```
C
      DO 620 I=1,NOSERV
      WRITE(80,42) (NIDL(I,J),J=1,10)
 620  CONTINUE
C
      CLOSE(80)
      GO TO 4000
C
 2000 WRITE(6,2) '-Copy your data from user file.'
      WRITE(6,1) 'Enter file name for your file.(b:cccccc.dat)------?'
      READ(5,34) FILENM
c
      CALL INITCO
      CALL INITLK
      OPEN(90,FILE=FILENM,STATUS='OLD')
c
      READ(90,3) PROJNAME,USER,DATE,(NAMESERV(I),I=1,10)
      READ(90,23) NTYP,NOSERV,FINTYP,MHSDIR,TYARR,TYSER,TYMAT,TYREC
      READ(90,44) QS(1,1),TYTOL
      READ(90,23) TYLT,OPTION,TYEVT,NOWS,NOSTA,NOMHS(1),NPRO,NTOT
      READ(90,25) VEL,SMHS,MAT1,TUNI(1),TUNI(2)
      READ(90,25) MAT2,TREC,MARRVT,TIREC,LTIME
      READ(90,48) LENGTH,TAVE
      NUMMHS=NOMHS(1)
c
      READ(90,26) (NSGRO(I),I=1,10)
      READ(90,26) (NTASKS(I),I=1,10)
      DO 650 I=1,NTYP
      DO 700 J=1,NTASKS(I)
      READ(90,26) (ROUTE(I,J,K),K=1,NOSERV)
 700  CONTINUE
 650  CONTINUE
C
      DO 720 I=1,NTYP
      DO 740 J=1,NTASKS(I)
      READ(90,49) (MSERVT(I,J,K),K=1,NOSERV)
 740  CONTINUE
 720  CONTINUE
C
      DO 760 I=1,NTYP
      READ(90,26) (CTASK(I,J),J=1,25)
 760  CONTINUE
C
C
      READ(90,27) (PROBD(I),I=1,10)
C
      DO 770 I=2,3
      READ(90,49) (FAILR(I,J),J=1,10)
 770  CONTINUE
C
```

```
      DO 775 I=2,3
      READ(90,26) (QS(I,J),J=1,10)
 775  CONTINUE
C
      READ(90,29) (RDEF(I),I=1,10)
      READ(90,29) (RREW(I),I=1,10)
C
      DO 780 I=1,NSGRO(2)
      READ(90,28) (MACHT(I,J),J=1,25)
 780  CONTINUE
C
      READ(90,26) (NTOL(I),I=1,10)
      READ(90,26) (LTOL(I),I=1,10)
C
      DO 800 I=1,NSGRO(2)
      READ(90,30) (AVAMA(I,J),J=1,9)
 800  CONTINUE
C
      DO 820 I=1,NSGRO(3)
      READ(90,30) (INSPEC(I,J),J=1,9)
 820  CONTINUE
C
      DO 840 I=1,NOSTA
      READ(90,32) (LDIS(I,J),J=1,25)
 840  CONTINUE
      DO 860 I=1,NOSTA
      READ(90,28) (RFLOW(I,J),J=1,25)
 860  CONTINUE
C
       DO 880 I=1,NOSERV
       READ(90,26) (NIDL(I,J),J=1,10)
 880   CONTINUE
C
      CLOSE(90)
      IF(IT.EQ.3) GO TO 3000
C
C
      GO TO 4000
 3000 CALL INPUTCHC(PROJNAME,USER,DATE,NAMESERV)
C
      WRITE(6,1) 'If you want to save data to your file,press the',
     $' key [S],if not,press the key [N]. ????'
      READ(5,45) COPY
 1111 IF(COPY.EQ.'S'.OR.COPY.EQ.'s') GO TO 5000
      IF(COPY.EQ.'N'.OR.COPY.EQ.'n') GO TO 4000
      WRITE(6,1) ' The input is wrong. please,try again.'
      READ(5,45) COPY
      GO TO 1111
C
 4000 CALL RANDI(1,JOBT)
```

```
      CALL DISTRI(1,TYARR,MARRVT,TAVE,TUNI,ETIME)
C
C
      RETURN
      END
C
C
C
      SUBROUTINE REPORT(TREM,PROJNAME,USER,DATE,NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA,NCHWS(45),NBLOK(25)
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25)
      INTEGER*2 NVALUE,JOBT,SERV,TYEVT,IS(45),IMH(25)
      INTEGER*2 ISS,MINN,IE,IFI,NENT,MAXN
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      REAL*4 MARRVT,PROBD(10),LTIME,TR,TREM
      REAL*4 AVEN,AVTI,UTIL,ADELA,BDELA,AVED,AMETH,AMAXTH,AMINTH
      CHARACTER PROJNAME*40,USER*20,DATE*10,NAMESERV(10)*15
      CHARACTER STNAME(10)*20,FILENM*15,CONT*1
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/INSP/INSPEC
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/OUT/NUMMHS
      COMMON/RAND/NVALUE,PROBD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
      DATA IS/45*0/,IMH/25*0/,STNAME/10*' '/
c
c
 1    FORMAT(20X,A,I2,A,I2)
 2    FORMAT(20X,A,I2,A,F4.3)
 3    FORMAT(10X,A)
 4    FORMAT(3X,70('-'))
 5    FORMAT(10(/))
 6    FORMAT(A1)
 7    FORMAT(1X,'After you read this,press the key[C] to continue.-?',$)
 8    FORMAT(A15)
 9    FORMAT(//,1X,A,$)
c
```

```
      WRITE(6,5)
      WRITE(6,11) DATE,PROJNAME,USER
 11   FORMAT(//,20X,'FMS SIMULATION REPORT'/,
     $20X,         '---------------------'/,
     $50X,'DATE:',A10/,2X,'PROJECT:',A40,2X,'BY: ',A20//)
c
      WRITE(6,21) MARRVT,NTYP,NOWS
 21   FORMAT(/,6X,'SYSTEM INPUT'/,
     $6X,'------------'/,
     $10X,'1. Job.'/,
     $15x,'New job mean arrival time: ',F8.3/,
     $15x,'Number of job types: ',I2/,
     $15x,'Number of work station types: ',I2/,
     $15x,'Number of operations for each job:'/)
c
      DO 100 I=1,NTYP
      WRITE(6,1) ' Job',I,':',NTASKS(I)
 100  CONTINUE
c
      WRITE(6,31) TYARR,TYSER
 31   FORMAT(/,15X,'Distribution type of arrival time: ',I2/,
     $15x,'Distribution type of work time: ',I2/,
     $15x,'Job allocation rate:'/)
c
      DO 200 I=1,NTYP
      WRITE(6,2) 'Job',I,':',PROBD(I)
 200  CONTINUE
c
      WRITE(6,41)
 41   FORMAT(///,10x,'2. WORK STATION.'/,10x,'-----------------'/)
C
      WRITE(6,51) NSGRO(2)
 51   FORMAT(15X,'** Machine station **'//,
     $20x,'Number of item of equipment at a machine station: ',I2//,
     $20x,32('-')/,
     $20x,'Machine',2x,'Number of ',2x,'Loaded tool'/,
     $20x,'Number ',2x,'tools     ',2x,'number     '/,20x,32('-')/)
c
      DO 300 I=1,NSGRO(2)
      WRITE(6,61) I,NTOL(I),LTOL(I)
61    FORMAT(23x,I2,9X,I2,10X,I2)
300   CONTINUE
c
      DO 400 I=3,NOSERV
      WRITE(6,71) NAMESERV(I),NSGRO(I)
 71   FORMAT(//,15X,'** ',A15,'station **'//,
     $20x,'Number item of equipment of the station :',I2,/)
 400  CONTINUE
c
      WRITE(6,81) MHSDIR,VEL,NUMMHS
```

```
 81   FORMAT(//,10x,'3. Material Handling Devices'/,
     $10x,          '----------------------------'/,
     $15x,'MH Direction Type :',I2,/,
     $15x,'Velocity          :',F8.3/,
     $15x,'Number of MH devices:',I2)
c
      WRITE(6,91) OPTION,FINTYP,TYEVT
 91   FORMAT(///,10X,'4. Others.'/,
     $10X,           '----------'/,
     $15x,'Option type :',I2,/,
     $15x,'Simulation completion type:',I2,/,
     $15x,'Job Arrival Event Generation Type:',I2)
c
      WRITE(6,5)
c
      ITOT=NOWS+NOSTA
      IS(1)=1
      J=2
      ISS=1+NSGRO(2)
C
      DO 450 I=2,NOSTA
      IF(I.LE.ISS) THEN
         IMH(I)=J
      ELSE
         J=J+1
         ISS=ISS+NSGRO(J)
         IMH(I)=J
      END IF
C
 450  CONTINUE
C
      DO 500 I=2,ITOT
      IF(I.LE.NOSTA) THEN
         IS(I)=IMH(I)
      ELSE
         IS(I)=IMH(I-NOSTA+1)
      END IF
 500  CONTINUE
C
C
      IF(FINTYP.EQ.1) THEN
          WRITE(6,101) TIME
      ELSE
          TIME=TREM
          WRITE(6,101) TREM
      END IF
C
 101  FORMAT(10X,'SUMMARY RESULTS'/,
     $10x,        '---------------'//,
     $50x,'Current time :',F10.3,//,
```

```
     $5X,'1. QUEUE STATISTICS REPORT'/,
     $5X,'-------------------------'//,3X,70('-')/,
     $7X,3X,'QUEUE TYPE',9X,' AVERAGE',2X,' MAXIMUM',1X,
     $' MINIMUM',4X,' AVERAGE TIME  '/,
     $30x,' NUMBER',3X,' NUMBER',2x,' NUMBER',3X,'  IN THE QUEUE'/,
     $3X,70('-'))
C
      DO 550 I=1,ITOT
      CALL FILEST(I)
        AVEN=TRNSFR(1)
        MAXN=TRNSFR(2)
        MINN=TRNSFR(3)
      CALL SAMPST(0.,-I)
        AVTI=TRNSFR(1)
C
      IF(I.LE.NOSTA) THEN
          WRITE(6,105) NAMESERV(IS(I)),AVEN,MAXN,MINN,AVTI
 105      FORMAT(11X,'MH-',A15,1X,F7.3,4X,I2,7X,I2,7X,F7.3)
          GO TO 550
      ELSE
          WRITE(6,107) NAMESERV(IS(I)),AVEN,MAXN,MINN,AVTI
 107      FORMAT(11X,'WS-',A15,1X,F7.3,4X,I2,7X,I2,7X,F7.3)
          GO TO 550
      END IF
C
 550  CONTINUE
C
      WRITE(6,4)
      WRITE(6,7)
      READ(5,6) CONT
      IF(CONT.EQ.'C'.OR.CONT.EQ.'c') GO TO 720
      RETURN
 720  WRITE(6,5)
C
      WRITE(6,111)
 111  FORMAT(5X,'2. WORK STATION STATISTICS REPORT'/,
     $5X,        '---------------------------------')
C
      STNAME(1)='MATERIAL HANDLING'
      DO 580 I=2,NOSERV
      STNAME(I)=NAMESERV(I)
 580  CONTINUE
C
      IK=NOSTA
      DO 600 J=1,NOSERV
      IF(J.EQ.1) THEN
          WRITE(6,113) STNAME(J)
 113      FORMAT(//,25X,'**',A20,'STATIONS','**'//,4X,70('-')/,
     $5X,'NO',9X,'AVERAGE',7X,'NUM. OF',8X,'AVERAGE',8X,'NUMBER OF'/,
     $16X,'UTILIZATION',4X,'BLOCK',10X,'DELAY',9X,'OBSERVATIONS'/,
```

```
     $4X,70('-'))
      ELSE
          WRITE(6,116) STNAME(J)
 116      FORMAT(//,25X,'**',A20,'STATIONS','**'//,4X,70('-')/,
     $5X,'NO',9X,'AVERAGE',7X,'NUM. OF',8X,'AVERAGE',8X,'NUMBER OF'/,
     $16X,'UTILIZATION',4X,'ST. CHANGE',7X,'DELAY',9X,'OBSERVATION'/,
     44X,70('-'))
      END IF
C
      IF(J.EQ.1) THEN
          IB=1
          IFI=NOSTA
      ELSE
          IK=IK+1
          IB=IK
          IFI=IB+NSGRO(J)-1
      END IF
C
      N=1
      DO 620 I=IB,IFI
C
      CALL TIMEST(0.,-I)
         UTIL=TRNSFR(1)
      CALL SAMPST(0.,-I)
         ADELA=TRNSFR(1)
         NENT=TRNSFR(2)
C
      IF(J.EQ.1) THEN
          WRITE(6,115) N,UTIL,NBLOK(I),ADELA,NENT
      ELSE
          WRITE(6,115) N,UTIL,NCHWS(I),ADELA,NENT
      END IF
C        .
 115  FORMAT(5X,I2,9X,F8.3,7X,I3,9X,F8.3,10X,I4)
      N=N+1
 620  CONTINUE
      IK=IFI
      WRITE(6,4)
 600  CONTINUE
      WRITE(6,5)
C
      WRITE(6,7)
      READ(5,6) CONT
      IF(CONT.EQ.'C'.OR.CONT.EQ.'c') GO TO 740
      RETURN
C
 740  WRITE(6,117)
 117  FORMAT(///,5x,'3. JOB STATISTICS REPORT'/,
     $5X,         '------------------------'//,3X,70('-')/,
     $5X,'JOB',3X,'AVERAGE',6X,'THROUGHPUT TIME',14X,'NUMBER OF'/,
```

```
     $22x,27('-'),2x,22('-')/,
     $5X,'TYPE',5X,'DELAY.',2X,'AVERAGE.',2X,'MAXIMUM.',2X,'MINIMUM.',
     $1X,'GEN.',2X,'SCR.',2X,'REW.',2X,'FIN.'/,3X,70('-'))
C
      ITOT=NOWS+NOSTA
      JTOT=ITOT+NTYP
      DO 640 I=1,NTYP
          CALL SAMPST(0.,-(ITOT+I))
          AVED=TRNSFR(1)*NTASKS(I)
C
          CALL SAMPST(0.,-(JTOT+I))
          AMETH=TRNSFR(1)
          AMAXTH=TRNSFR(3)
          AMINTH=TRNSFR(4)
C
          WRITE(6,121) I,AVED,AMETH,AMAXTH,AMINTH,NUTY(I),NSCR(I),
     $    NREW(I),NFIN(I)
 121      FORMAT(6X,I2,3X,F8.3,2X,F8.3,2X,F8.3,2X,F8.3,2X,I4,2X,I4,2X,
     $    I4,2X,I4)
 640  CONTINUE
      WRITE(6,4)
      WRITE(6,5)
C
      WRITE(6,805)
 805  FORMAT(///,1X,'This report is completed.'/,
     $1X,'If you want to save your output,press the key"s".',$)
      READ(5,6) CONT
      IF(CONT.EQ.'S'.OR.CONT.EQ.'s') GO TO 900
      RETURN
C
 900  WRITE(6,9) '- Enter your file name(b:cccccc.dat)--?'
      READ(5,8) FILENM
C
      OPEN(70,FILE=FILENM,STATUS='NEW')
      WRITE(70,5)
      WRITE(70,11) DATE,PROJNAME,USER
      WRITE(70,21) MARRVT,NTYP,NOWS
      DO 810 I=1,NTYP
      WRITE(70,1) ' JOB',I,':',NTASKS(I)
 810  CONTINUE
      WRITE(70,31) TYARR,TYSER
      DO 820 I=1,NTYP
      WRITE(70,2) 'JOB',I,':',PROBD(I)
 820  CONTINUE
      WRITE(70,41)
      WRITE(70,51) NSGRO(2)
      DO 830 I=1,NSGRO(2)
      WRITE(70,61) I,NTOL(I),LTOL(I)
830   CONTINUE
      DO 840 I=3,NOSERV
```

```
      WRITE(70,71) NAMESERV(I),NSGRO(I)
 840  CONTINUE
      WRITE(70,81) MHSDIR,VEL,NUMMHS
      WRITE(70,91) OPTION,FINTYP,TYEVT
      WRITE(70,5)
C
C
      WRITE(70,101) TIME
      DO 855 I=1,ITOT
      CALL FILEST(I)
        AVEN=TRNSFR(1)
        MAXN=TRNSFR(2)
        MINN=TRNSFR(3)
        CALL SAMPST(0.,-I)
        AVTI=TRNSFR(1)
C
      IF(I.LE.NOSTA) THEN
          WRITE(70,105) NAMESERV(IS(I)),AVEN,MAXN,MINN,AVTI
      ELSE
          WRITE(70,107) NAMESERV(IS(I)),AVEN,MAXN,MINN,AVTI
      END IF
 855  CONTINUE
      WRITE(70,4)
      WRITE(70,5)
      WRITE(70,111)
C
      IK=NOSTA
      DO 860 J=1,NOSERV
      IF(J.EQ.1) THEN
           WRITE(70,113) STNAME(J)
      ELSE
           WRITE(70,116) STNAME(J)
      END IF
      IF(J.EQ.1) THEN
          IB=1
          IFI=NOSTA
      ELSE
          IK=IK+1
          IB=IK
          IFI=IB+NSGRO(J)-1
      END IF
C
      N=1
      DO 862 I=IB,IFI
C
      CALL TIMEST(0.,-I)
         UTIL=TRNSFR(1)
      CALL SAMPST(0.,-I)
         ADELA=TRNSFR(1)
         NENT=TRNSFR(2)
```

```
C
      IF(J.EQ.1) THEN
          WRITE(70,115) N,UTIL,NBLOK(I),ADELA,NENT
      ELSE
          WRITE(70,115) N,UTIL,NCHWS(I),ADELA,NENT
      END IF
C
      N=N+1
 862  CONTINUE
      IK=IFI
      WRITE(70,4)
 860  CONTINUE
      WRITE(70,5)
C
      WRITE(70,117)
      DO 864 I=1,NTYP
          CALL SAMPST(0.,-ITOT-I)
          AVED=TRNSFR(1)*NTASKS(I)
C
          CALL SAMPST(0.,-JTOT-I)
          AMETH=TRNSFR(1)
          AMAXTH=TRNSFR(3)
          AMINTH=TRNSFR(4)
C
          WRITE(70,121) I,AVED,AMETH,AMAXTH,AMINTH,NUTY(I),NSCR(I),
     $NREW(I),NFIN(I)
 864  CONTINUE
      WRITE(70,4)
      WRITE(70,5)
      CLOSE(70)
      RETURN
      END
c
c
      SUBROUTINE DISTRI(IT,TYPE,A1,B1,C1,OUT)
      INTEGER*2 TYPE,IT
      INTEGER*4 IX,IY
      REAL*8 RN
      REAL*4 OUT,A1,B1,PROBD(10),C1(2)
      COMMON/RAND/NVALUE,PROBD
C
C ** GENERATE RANDOM NUMBERS AND FIND TYPE YOU NEED **
C
      IF(IT.EQ.1) THEN
           WRITE(6,11)
 11        FORMAT(1x,'- Enter random number seed(less than 9 digits',
     $               ') for random variable generation'/,'----?',$)
           READ(5,*) IX
           CALL RANDOM(IX,RN)
      ELSE
```

```
          CALL RANDOM(IX,RN)
      END IF
C
      GO TO(100,200,300,400,500) TYPE
 100  CALL EXPON(A1,OUT,RN)
      RETURN
 200  CALL TRUNEX(A1,B1,OUT,RN)
      RETURN
 300  D1=A1*C1(1)
      E1=A1*C1(2)
      CALL UNIFRM(D1,E1,OUT,RN)
      RETURN
 400  OUT=A1
      RETURN
 500  OUT=REAL(RN)
      RETURN
      END
C
C
      SUBROUTINE TRUNEX(RMEAN,RNAVE,RESULT,RN)
      INTEGER*4 IX,IY
      REAL*8 RN
      REAL*4 RMEAN,RNAVE,AA,BB,RESULT
C
      BB=RMEAN*RNAVE
 100  AA=-RMEAN*LOG(1-RN)
C
      IF(AA.GT.BB) THEN
          CALL RANDOM(IX,RN)
          GO TO 100
      END IF
      RESULT=AA
      RETURN
      END
C
C
C
      SUBROUTINE EXPON(RMEAN,RESULT,RN)
      INTEGER*4 IX,IY
      REAL*8 RN
      REAL*4 RMEAN,RESULT
C
C--- Generate an exponential random vasiable with RMEAN.-----
C
      RESULT=-RMEAN*LOG(1-RN)
      RETURN
      END
C
C
C
```

```
      SUBROUTINE RANDI(IT,RESULT)
      INTEGER*2 I,NI,RESULT
      INTEGER*4 IX,IY
      REAL*8 RN
      REAL*4 CUM,PROBD(10)
      COMMON/RAND/NVALUE,PROBD
C
C --- Generate a U(0,1) random variable. ----
C
      IF(IT.EQ.1) THEN
              WRITE(6,11)
 11           FORMAT(1X,'- Enter random number seed(less than 9 digits)',
     $                   'for integer random number -----? ',$)
              READ(5,*) IX
              CALL RANDOM(IX,RN)
      ELSE
              CALL RANDOM(IX,RN)
      END IF
C
C --- Generate a random integer between 1 and N. ----
C
      N1=NVALUE-1
      CUM=0.
      DO 10 I=1,N1
      CUM=CUM+PROBD(I)
      IF (RN.LE.CUM) THEN
          RESULT=I
          RETURN
      ENDIF
 10   CONTINUE
      RESULT=NVALUE
      RETURN
      END
C
C
C
      SUBROUTINE UNIFRM(A,B,RESULT,RN)
      INTEGER*4 IX,IY
      REAL*8 RN
      REAL*4 A,B,RESULT
C
C --- Generate a U(A,B) random variable. -----
C
      RESULT=A+(RN*(B-A))
      RETURN
      END
C
C
C
      SUBROUTINE RANDOM(IX,RN)
```

```
      INTEGER*4 A,P,IX,B15,B16,XHI,XALO,LEFTLO,FHI,K
      REAL*8 RN
      SAVE IX
      DATA A/16807/,B15/32768/,B16/65536/,P/2147483647/
C
      XHI=IX/B16
      XALO=(IX-XHI*B16)*A
      LEFTLO=XALO/B16
      FHI=XHI*A+LEFTLO
      K=FHI/B15
      IX=(((XALO-LEFTLO*B16)-P)+(FHI-K*B15)*B16)+K
      IF(IX.LT.0) IX=IX+P
      RN=FLOAT(IX)*4.656612875E-10
      RETURN
      END
```

```
c
c ---- this module is made for FMS data input----
c ---- Update date Jun 28 1986
C ---- Update Oct 4 1986 ---------
c
      SUBROUTINE HEAD(PROJNAME,USER,DATE)
      CHARACTER PROJNAME*40,USER*20,DATE*10
c
c *** Define format ***
c
 1    FORMAT(//,1x,A,$)
 2    FORMAT(A40)
 3    FORMAT(A20)
 4    FORMAT(A10)
C
      WRITE(6,101)
 101  FORMAT(15(/),
     $'_______________________________________________________________'/,
     $' |                                                              |'/,
     $' | This subroutine is used to enter project name (up to 40 chrs), |'/,
     $' | user name (up to 20 chrs.), and input date (mm/dd/yy).        |'/,
     $' |______________________________________________________________|'/)
c
      WRITE(6,1) '-Enter project name --------?'
      READ(5,2) PROJNAME
      WRITE(6,1) '-Enter user name -----------?'
      READ(5,3) USER
      WRITE(6,1) '-Enter date of today ---?'
      READ(5,4) DATE
c
      RETURN
      END
C
C
C
      SUBROUTINE INPUTJOB(NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 S(25),TYEVT
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      CHARACTER NAMESERV(10)*15
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/RAND/NVALUE,PROBD
      DATA S/25*0/
```

```
C
c *** Define format ****
C
 2     FORMAT(//,1x,A,$)
 3     FORMAT(I2)
 4     FORMAT(F8.3)
 6     FORMAT(//,1X,A)
 7     FORMAT(/,6X,':Now enter the following data and Press "RETURN"')
 8     FORMAT(//,1x,A,I2,A,$)
c
c
       WRITE(6,11)
 11    FORMAT(12(/),
     $'________________________________________________________________'/,
     $' |                                                              |'/,
     $' |  This subroutine is used to enter data for each job in       |'/,
     $' |  the following sequence. First the number of operations      |'/,
     $' |  of ajob entered. Then the work station types with the       |'/,
     $' |  operation number of the job are entered in groups of        |'/,
     $' |  five numerical entries. Second, the equipment item          |'/,
     $' |  number used at the work station and its work time with      |'/,
     $' |  the operation numbers are entered. Third,an allocation      |'/,
     $' |  rate of the job is entered.                                 |'/,
     $' |     After the above procedure ,the distribution type of      |'/,
     $' |  work time is the final entry.                               |'/,
     $' |______________________________________________________________|')
c
c
       DO 100 I=1,NTYP
       WRITE(6,8) '- Enter the number of operations of job ',I,'-----?'
       READ(5,3) NTASKS(I)
c
       WRITE(6,6) '- Enter the work station types with the operation numbers.'
       J=1
 300   WRITE(6,16) (K,K=J,J+4)
 16    FORMAT(//,'Operation numbers:',2x,5('    ',I2,'        ')/,
     $          '                  ',2x,5('    :         ')/,
     $          'Enter work station',2x,5('    :         ')/,
     $          'types             ',2x,5('    ?         ')/,
     $          'enter--?',$)
c
       READ(5,*) (S(K),K=J,J+4)
       DO 400 IK=J,J+4
          IF(S(IK).EQ.0) THEN
             S(IK)=1
          END IF
 400   CONTINUE
C
C
       WRITE(6,21)
```

```
 21   FORMAT(2(/),
     $           'Enter item number',/,
     $           ' and work time-? ',5('item  work ')/,
     $           '                 ',5('  #   time ')/,
     $           '                 ',5('  :     :  ')/,
     $           'enter--? ',$)
      READ(5,*) (ROUTE(I,K,S(K)),MSERVT(I,K,S(K)),K=J,J+4)
C
      IF (K.GT.NTASKS(I)) GO TO 200
      J=J+5
      GO TO 300
C
 200  IF(TYEVT.EQ.2) GO TO 100
      WRITE(6,8) '- Enter job ',I,' allocation rate(0.nnn)----?'
      READ(5,4)  PROBD(I)
 100  CONTINUE
C
      WRITE(6,2) '- Enter the distribution type (1,2,3,or 4) of work time--?'
      READ(5,3) TYSER
      RETURN
      END
C
C
C
      SUBROUTINE INPUTMCH(NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      CHARACTER NAMESERV(10)*15
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
C
c *** Defineformat ****
C
 1    FORMAT(//,1x,A,I2,$)
 2    FORMAT(//,1x,A,A)
 3    FORMAT(I2)
 6    FORMAT(//,1X,A)
 7    FORMAT(//,1x,'Now enter the following data and Press "RETURN"')
 8    FORMAT(//,1x,A,I2,A,$)
 9    FORMAT(6(/))
c
c
      WRITE(6,9)
      WRITE(6,11)
 11   FORMAT(//,
     $'  ________________________________________________________________________'/,
```

```
     $' |                                                           |'/,
     $' |  This subroutine is used to enter the number of tools    |'/,
     $' |  and the tool lists for each item of equipment at a      |'/,
     $' |  machine station. Each equipment item has up to 25       |'/,
     $' |  tools in its tool magazine.                             |'/,
     $' |__________________________________________________________|'/)
c
      DO 100 I=1,NSGRO(2)
      WRITE(6,9)
      WRITE(6,8)'- Enter the number of tools of equipment item ',I,
     $' of the machine station --?'
      READ(5,3) NTOL(I)
c
      WRITE(6,1) '- Enter the tool lists of equipment item ',I,
     $' of the machine station --?'
      J=1
 200  WRITE(6,21) (K,K=J,J+4)
 21   FORMAT(//,'Tool list:',5('   ',I2,'    ')/,
     $          '          ',5('   :    ')/,
     $'enter-?',$)
c
      READ(5,*) (MACHT(I,K),K=J,J+4)
c
      IF(K.GT.NTOL(I)) GO TO 300
      J=J+5
      GO TO 200
C
 300  LTOL(I)=MACHT(I,1)
 100  CONTINUE
c
      WRITE(6,31)
 31   FORMAT(//,1X,'- Enter tool loading time and its distribution type.'/,
     $1x,'              :                              : (1 to 4)'/,
     $'enter---?',$)
      READ(5,*) LTIME,TYTOL
      RETURN
      END
C
C
C
      SUBROUTINE INPUTAVA(ITYPE,NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 FINTYP,MHSDIR,NPRO,OPTION
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      REAL*4 MARRVT,PROBD(10),LTIME,TR
```

```
      CHARACTER NAMESERV(10)*15,CHAN*1
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/INSP/INSPEC
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
C
c *** Define format ****
C
 1    FORMAT(//,1x,A,I2)
 2    FORMAT(//,A,$)
 3    FORMAT(A1)
 4    FORMAT(//,1X,A,A15,A)
 5    FORMAT(//,1x,A,A,I2,/,A,A15,A)
 7    FORMAT(/,6x,'The default value(s) is(are) zero(es). When the user ',
     $'want to define '/6x,'new value(s), press the key [C]. Otherwise ',
     $'press RETURN. ',$)
 8    FORMAT(9(/))
C
      IX=2
      GO TO (400,500,600,600,400) ITYPE
C
 400  IF(OPTION.EQ.1) RETURN
      WRITE(6,8)
      WRITE(6,16)
 16   FORMAT(//,
     $' ____________________________________________________________________'/,
     $' |                                                                  |'/,
     $' |  This subroutine is used to enter the lists of available        |'/,
     $' |  equipment item numbers of work stations. When an equipment     |'/,
     $' |  item at a work station is not available because of a          |'/,
     $' |  maintenance or breakdown, the user can define the queue        |'/,
     $' |  capacities of all the equipment items at the work station      |'/,
     $' |  (load/unload,machine, or inspection station).                  |'/,
     $' |__________________________________________________________________|'/)
      IF(ITYPE.EQ.5) GO TO 700
c
 500  WRITE(6,2) '- Enter the queue capacity at the load/unload station---?'
      WRITE(6,7)
      READ(5,3) CHAN
      IF(CHAN.EQ.'C'.OR.CHAN.EQ.'c') THEN
            WRITE(6,2)' Enter the queue capacity--?'
            READ(5,*)  QS(1,1)
      END IF
c
      IF(ITYPE.EQ.2) RETURN
C
 600  IF(ITYPE.EQ.3.OR.ITYPE.EQ.1) THEN
            IX=2
```

```
      ELSE
            IX=3
      END IF
C
      DO 300 I=IX,NOSERV
      WRITE(6,4) '- Enter the queue capacities of equipment items at the ',
     $NAMESERV(I),'station.'
      WRITE(6,7)
      READ(5,3) CHAN
      IF(CHAN.EQ.'C'.OR.CHAN.EQ.'c') GO TO 650
      GO TO 300
 650  WRITE(6,11) NAMESERV(I)
 11   FORMAT(//,20X,A15,' STATION'//,
     $'item number:','*1*   *2*   *3*   *4*   *5*   *6*   *7*   *8*   *9*   *10*'/,
     $'              ',' :     :     :     :     :     :     :     :     :      :'/,
     $'              ',' ?     ?     ?     ?     ?     ?     ?     ?     ?      ?'/,
     $'enter--?',$)
c
      READ(5,*) (QS(I,J),J=1,10)
      IF(ITYPE.EQ.4.OR.ITYPE.EQ.3) RETURN
 300  CONTINUE
c
      IF(ITYPE.EQ.1) RETURN
 700  WRITE(6,8)
      DO 100 J=1,NSGRO(2)
      WRITE(6,5)'- Enter the available equipment item lists for equiment',
     $' item number',J,' at the ', NAMESERV(2),' station.'
      WRITE(6,21)
 21   FORMAT(/,
     $10x,'*1*   *2*   *3*   *4*   *5*   *6*   *7*   *8*   *9*'/,
     $10X,' :     :     :     :     :     :     :     :     :'/,
     $10X,' :     :     :     :     :     :     :     :     :'/,
     $10X,' ?     ?     ?     ?     ?     ?     ?     ?     ?'/,
     $'enter--?',$)
c
      READ(5,*) (AVAMA(J,K),K=1,9)
 100  CONTINUE
C
      WRITE(6,8)
      DO 200 J=1,NSGRO(3)
      WRITE(6,5) ' Enter the available equipment item lists for equipment',
     $' item number ',J,'at the ',NAMESERV(3),' station.'
      WRITE(6,21)
      READ(5,*) (INSPEC(J,K),K=1,9)
 200  CONTINUE
C
      RETURN
      END
C
C
```

```
C
      SUBROUTINE INPUTASK(NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      CHARACTER NAMESERV(10)*15
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
C
c *** Define format ****
C
 1    FORMAT(//,1x,A,A,I2)
 7    FORMAT(/,1x,'Now enter the following data and Press "RETURN"')
 8    FORMAT(9(/))
c
      WRITE(6,8)
      WRITE(6,16)
 16   FORMAT(//,
     $'  ____________________________________________________________ '/,
     $' |                                                            |'/,
     $' |  This subroutine is used to enter the tool number with     |'/,
     $' |  operation numbers for each job.                           |'/,
     $' |____________________________________________________________|')
C
      DO 100 I=1,NTYP
      WRITE(6,8)
      J=1
      WRITE(6,1) '- Enter the tool (or equipment) number with operation'
     $'numbers of job ',I
 200  WRITE(6,11) (K,K=J,J+9)
 11   FORMAT(///,'Operation numbers :',10('*',I2,'* ')/,
     $       '                   ',10('  :  ')/,
     $'enter---?',$)
      READ(5,*) (CTASK(I,K),K=J,J+9)
C
      IF(K.GT.NTASKS(I)) GO TO 100
      J=J+10
      WRITE(6,8)
      GO TO 200
 100  CONTINUE
C
      RETURN
      END
C
C
C
      SUBROUTINE INPUTOPT(NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
```

```
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      INTEGER*2 TYEVT
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      CHARACTER NAMESERV(10)*15
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
C
c *** Define format ****
C
 3    FORMAT(/,1x,A,$)
 4    FORMAT(//,1x,A,$)
 5    FORMAT(//,1x,A)
 6    FORMAT(//,1X,A,A15,A)
 7    FORMAT(/,6X,':Now enter the following data and Press "RETURN"')
 8    FORMAT(9(/))
c
c
      MAT1=0.0
      MAT2=0.0
C
      WRITE(6,8)
      WRITE(6,11)
 11   FORMAT(/,
     $' _______________________________________________________________'/,
     $' |                                                             |'/,
     $' |    In this subroutine the breakdown rates and scheduled     |'/,
     $' |    maintenance plan for work stations are entered.          |'/,
     $' |    Breakdown rates are defined by real values(0.nnn).       |'/,
     $' |    Maintenance plans are defined by entering the            |'/,
     $' |    interval between scheduled maintenances times, the       |'/,
     $' |    repair time and its distribution type.                   |'/,
     $' |_____________________________________________________________|')
c
      GO TO(200,300,400,300) OPTION
 200  RETURN
C
 300  DO 100 I=2,3
      WRITE(6,6)'- Enter the breakdown rates of ',NAMESERV(I),'station.'
      WRITE(6,14)
 14   FORMAT(//,'equipment item number:',
     $'* 1* * 2* * 3* * 4* * 5* * 6* * 7* * 8* * 9* *10*'/,
     $          '                                           ',
     $'  :    :    :    :    :    :    :    :    :    :'/,
     $          '                                           ',
     $'  ?    ?    ?    ?    ?    ?    ?    ?    ?    ?'/,
     $'enter--?',$)
```

```
      READ(5,*) (FAILR(I,J),J=1,10)
      WRITE(6,8)
 100  CONTINUE
C
      WRITE(6,16)
 16   FORMAT(//,10x,'When an equipment item at a machine station breakdowns,',
     $/10x,'the user enters the repair time and its distribution type.')
      WRITE(6,3) '- Enter the repair time and its distribution type.'
      WRITE(6,3) '                        :                   :'
      WRITE(6,3) '                        ?                   ?'
      WRITE(6,4)'enter--?'
      READ(5,*) TREC,TYREC
c
      IF(OPTION.EQ.4) GO TO 400
      RETURN
C
      WRITE(6,8)
 400  WRITE(6,5) '- Enter maintenance plan.'
      WRITE(6,31)
 31   FORMAT(///,
     $10x,'Maintenance time interval.'/,
     $10x,' :        Repair time.'/,
     $10x,' :          :       The distribution type.'/,
     $10x,' :          :              :'/,
     $10x,' ?          ?              ?'/,
     $'enter--?',$)
c
      READ(5,*) MAT1,TIREC,TYMAT
      CALL DISTRI(2,TYMAT,TIREC,TAVE,TUNI,ET)
      MAT2=MAT1+ET
c
      RETURN
      END
C
C
C
      SUBROUTINE INPUTQC(NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      CHARACTER NAMESERV(10)*15
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
C
c *** Define format ****
C
 4    FORMAT(10(/))
 5    FORMAT(//,1x,A,A15,A)
 6    FORMAT(//,1X,A)
```

```
c
      WRITE(6,4)
      WRITE(6,11)
 11   FORMAT(//,
     $'  _______________________________________________________________'/,
     $' |                                                               |'/,
     $' |  This subroutine is used to enter the defective rates        |'/,
     $' |  of equipment items at a machine station. When a job is      |'/,
     $' |  defective , the possibility of re-use after the job is      |'/,
     $' |  repaired is considered .In such a case the user enters      |'/,
     $' |  the rework rate.                                            |'/,
     $' |_______________________________________________________________|')
c
      WRITE(6,5) '- Enter the defective rates at a  ',NAMESERV(2),'station'
      WRITE(6,21)
 21   FORMAT(//,'Equipment item number:',
     $'*1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*'/,
     $
     $' :    :    :    :    :    :    :    :    :     :'/,
     $
     $' ?    ?    ?    ?    ?    ?    ?    ?    ?     ?'/,
     $'enter--?',$)
c
      READ(5,*) (RDEF(I),I=1,10)
c
      WRITE(6,4)
      WRITE(6,5) '- Enter the rework rates at a ',NAMESERV(2),' station.'
      WRITE(6,21)
      READ(5,*) (RREW(I),I=1,10)
      RETURN
      END
c
c
c
      SUBROUTINE INPUTMHS
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA,TYEVT
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25),NCHWS(45),NBLOK(25)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC,TAVE
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
```

```
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
C
c *** Define format ****
C
 6    FORMAT(//,1X,A)
 7    FORMAT(6(/))
c
c
      WRITE(6,7)
      WRITE(6,11)
 11   FORMAT(//,
     $' ______________________________________________________________ '/,
     $' |                                                             |'/,
     $' |  Material handling (MH) devices transport the workpieces    |'/,
     $' |  between MH stations. This subroutine is used to enter the  |'/,
     $' |  velocity of MH vehicles,the loading time for each vehicles,|'/,
     $' |  and its distribution type.                                 |'/,
     $' |_____________________________________________________________|')
c
      WRITE(6,6) '- Enter values for the following variables.'
      WRITE(6,21)
 21   FORMAT(//,
     $10x,'1. Velocity of MH vehicles(m/min)'/,
     $10x,'     :   '/,
     $10x,'     :     2. Loading time(min)'/,
     $10x,'     :          :'/,
     $10x,'     :          :     3. The distribution type '/,
     $10x,'     :          :          :   (1 to 4)'/,
     $10x,'     :          :          :'/,
     $10x,'     ?          ?          ?'/,
     $'enter--?',$)
      READ(5,*) VEL,SMHS,TYLT
C
C
      RETURN
      END
C
C
      SUBROUTINE MAKESIS(TYPE,NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25),NCHWS(45),NBLOK(25)
      INTEGER*2 DISX1,DISX2,DISY1,DISY2,NUM,TYPE,DIST,MAP(25,2)
      INTEGER*2 IMH(45),INN(45),ILDIS(25,25)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),MARRVT,LENGTH
      CHARACTER NAMESERV(10)*15,ANSW*1
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
```

```
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      DATA DISX1,DISY1,DISX2,DISY2,NUM/5*0/,MAP/50*0/
C
 6    FORMAT(//,1x,A)
C
      DO 60 I=1,NOSTA
       DO 70 J=1,NOSTA
         LDIS(I,J)=0
         ILDIS(I,J)=0
         RFLOW(I,J)=99
 70    CONTINUE
 60   CONTINUE
C
C --- MH INDEX ---
      IMH(1)=1
      INN(1)=1
      J=2
      ISS=1+NSGRO(2)
      NN=0
      DO 101 I=2,NOSTA
         IF(I.LE.ISS) THEN
            IMH(I)=J
            NN=NN+1
            INN(I)=NN
         ELSE
            J=J+1
            ISS=ISS+NSGRO(J)
            IMH(I)=J
            NN=1
            INN(I)=NN
         END IF
 101  CONTINUE
C
      WRITE(6,11)
 11   FORMAT(//,
     $'  _______________________________________________________________'/,
     $' |                                                               |'/,
     $' |  This program can handle the three types of MH direction.    |'/,
     $' |  When type "1" is selected, the user enters the distance to  |'/,
     $' |  the next MH station number and its number. When type "2" is |'/,
     $' |  selented, the user enters the x,y coordinates of all the MH |'/,
     $' |  stations. When type "3" is selected, the user enters the    |'/,
     $' |  the distance to and the number of nearest MH stations both  |'/,
     $' |  directions.                                                  |'/,
     $' |_______________________________________________________________|')
C
      ITYPE=TYPE
      GO TO(1000,2000,3000) ITYPE
```

```
C
 1000 DO 1050 I=1,NOSTA
      WRITE(6,21) I,INN(I),NAMESERV(IMH(I))
 21   FORMAT(//,3x,'MH station number: ',I2,' (Equipment item number:',I2,
     $' at the ',A15,'station)'/,
     $10X,'The nearest MH station number from this MH station'/,
     $10x,'      :        The distance between them'/,
     $10x,'      :                          :'/,
     $'enter----?',$)
c
      READ(5,*) JK,DIST
      MAP(I,1)=JK
      LDIS(I,JK)=DIST
      RFLOW(I,JK)=0
 1050 CONTINUE
      IX=1
      GO TO 4000
C
 2000 WRITE(6,6) 'Enter the X and Y coordinates of a MH station'
      DO 2050 I=1,NOSTA
      WRITE(6,31) I,INN(I),NAMESERV(IMH(I))
 31   FORMAT(6x,'MH Station number :',I2,' (Equipment item number:',I2,
     $' at a ',A15,'station)'/,
     $10x,'X-coordinate     Y-coordinate'/,
     $10X,' :                 :'/,
     $10x,' ?                 ?'/,
     $'enter---?',$)
      READ(5,*) CORDX(I),CORDY(I)
 2050 CONTINUE
C
C
      DO 100 I=1,NOSTA
      DISX1=CORDX(I)
      DISY1=CORDY(I)
      DO 200 J=1,NOSTA
C
      IF(I.EQ.J) THEN
          LDIS(I,J)=0
          GO TO 200
      ELSE
          DISX2=CORDX(J)
          DISY2=CORDY(J)
          LDIS(I,J)=(DISX2-DISX1)+(DISY2-DISY1)
      END IF
 200  CONTINUE
 100  CONTINUE
C
      DO 300  I=1,NOSTA
C
      CALL MIN(I,NO,IN)
```

```
      IF(IN.EQ.1) THEN
         CALL MAX(I,NUM,IM)
         RFLOW(I,NUM)=0
      ELSE IF(IN.LT.NOSTA) THEN
         CALL MIN(I,NO,IN)
         RFLOW(I,NO)=0
         CALL MAX(I,NUM,IM)
         RFLOW(I,NUM)=0
      ELSE
         CALL MIN(I,NO,IN)
         RFLOW(I,NO)=0
      END IF
C
 300  CONTINUE
      RETURN
C
 3000 DO 3050 I=1,NOSTA
      WRITE(6,41)I,INN(I),NAMESERV(IMH(I))
 41   FORMAT(//,3X,'MH station number: ',I2,' (Equipment item number:',I2,
     $' at the ',A15,'station)'/,
     $15x,'<To the direction of clock-wise.>'/,
     $10x,'The nearest MH station number from this station.'/,
     $10x,'         :              The distance between them.'/,
     $10x,'         :                         :'/,
     $'enter---?',$)
c
      READ(5,*) JK,DIST
      MAP(I,2)=JK
      IF(JK.EQ.0) GO TO 3050
      RFLOW(I,JK)=0
      LDIS(I,JK)=-1*DIST
      ILDIS(I,JK)=-1*DIST
c
      WRITE(6,51) I,INN(I),NAMESERV(IMH(I))
 51   FORMAT(//,3X,'MH station number: ',I2,' (Equipment item  number:',I2,
     $' at the ',A15,'station)'/,
     $15x,'<To the direction of counter clock-wise.>'/,
     $10x,'The nearest MH station number from this station.'/,
     $10x,'        :               The distance between them.'/,
     $10x,'        :                       :'/,
     $'enter---?',$)
c
      READ(5,*) JK,DIST
      MAP(I,1)=JK
      IF(JK.EQ.0) GO TO 3050
      RFLOW(I,JK)=0
      LDIS(I,JK)=DIST
      ILDIS(I,JK)=DIST
 3050 CONTINUE
      IX=1
```

```
C
 4000 DO 4200 I=1,NOSTA
      IF(MAP(I,IX).EQ.0) GO TO 4200
      DIST=LDIS(I,MAP(I,IX))
      K=I
C
 4300 IK=MAP(K,IX)
      IF(MAP(IK,IX).EQ.0) GO TO 4200
      DIST=DIST+ILDIS(IK,MAP(IK,IX))
      LDIS(I,MAP(IK,IX))=DIST
      IF(MAP(IK,IX).EQ.I) GO TO 4250
      K=IK
      GO TO 4300
 4250 LDIS(I,I)=0
 4200 CONTINUE
C
      IF(ITYPE.EQ.3) THEN
           IX=2
           ITYPE=2
           GO TO 4000
      END IF
C
      RETURN
      END
C
C
C
      SUBROUTINE INPUTCHC(PROJNAME,USER,DATE,NAMESERV)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25)
      INTEGER*2 NVALUE,JOBT,SERV,TYEVT,NCHWS(45),TERM,NBLOCK(25)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      CHARACTER PROJNAME*40,USER*20,DATE*10,NAMESERV(10)*15,CHAN*1
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/INSP/INSPEC
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
```

```
      COMMON/RAND/NVALUE,PROBD
C
C ---- DEFINE  FORMAT -----
C
 2    FORMAT(//,1X,A,$)
 3    FORMAT(A40)
 4    FORMAT(A10)
 5    FORMAT(A1)
C
      WRITE(6,11)
 11   FORMAT(//,
     $'  _____________________________________________________________'/,
     $' |                                                             |'/,
     $' |  The values of the variables listed below may be changed    |'/,
     $' |  in order to correct entry errors or to test different      |'/,
     $' |  alternatives.If changes are required enter the project     |'/,
     $' |  name and date,and make the changes selected from the       |'/,
     $' |  following type numbers. After making changes,save them     |'/,
     $' |  to the user file and the simulation will run.              |'/,
     $' |                                                             |'/,
     $' |      TYPE NUMBER       DESCRIPTION                          |'/,
     $' |         1.             Job mean arrival time.               |'/,
     $' |         2.             Queue capacity of work stations      |'/,
     $' |         3.             Velocity of MH devices.              |'/,
     $' |         4.             Simulation completion type and       |'/,
     $' |                        time,or number of products to        |'/,
     $' |         5.             Number of MH devices.                |'/,
     $' |         6.             Option number for maintenance.       |')
      WRITE(6,16)
 16   FORMAT(' ',
     $'|         7.             Location of MH station               |'/,
     $' |         8.             Job generation type.                 |'/,
     $' |_____________________________________________________________|')
c
      WRITE(6,2) ' How many type numbers you want to change--?'
      READ(5,*) NUMCH
      WRITE(6,2) 'Enter program name --------?'
      READ(5,3) PROJNAME
      WRITE(6,2) 'Enter date ----------------?'
      READ(5,4) DATE
c
      DO 1000 I=1,NUMCH
      WRITE(6,2) '- Enter type number -----?'
      READ(5,*) NUMTY
C
      GO TO(100,150,200,250,300,350,400,450) NUMTY
C
 100  WRITE(6,2) '- Enter job mean arrival time --------?'
      READ(5,*) MARRVT
      GO TO 1000
```

```
C
 150  WRITE(6,21)
 21   FORMAT(//,1X,'- Enter work station type-------?',
     $1x,          '        1. Load/Unload station.'/,
     $1x,          '        2. Machine station.'/,
     $1x,          '        3. Inspection station.'/,
     $1x,'enter----?',$)
      READ(5,*) II
C
      GO TO(160,170,180) II
 160  CALL INPUTAVA(2,NAMESERV)
      GO TO 1000
 170  CALL INPUTAVA(3,NAMESERV)
      GO TO 1000
 180  CALL INPUTAVA(4,NAMESERV)
      GO TO 1000
C
 200  WRITE(6,2) '- Enter the velocity of MH vehicles-----?'
      READ(5,*) VEL
      GO TO 1000
C
 250  WRITE(6,26)
 26   FORMAT(/,1X,'If you want to change simulation completion type,'/,
     $1x,'press the key[C].Otherwise,press any keys. ???',$)
      READ(5,5) CHAN
      IF(CHAN.EQ.'C'.OR.CHAN.EQ.'c') THEN
                WRITE(6,2)'- Enter simulation completion type---?'
                READ(5,*) FINTYP
      END IF
C
      WRITE(6,31)
 31   FORMAT(//,'- Enter the completion time or number of products to',
     $' simulate-------?',$)
      READ(5,*) TERM
C
      IF(FINTYP.NE.2) THEN
           LENGTH=FLOAT(TERM)
           NPRO=0
      ELSE
           NPRO=TERM
           LENGTH=0
      END IF
      GO TO 1000
C
 300  WRITE(6,2)'- Enter number of MH devices-----?'
      READ(5,*) NOMHS(1)
      GO TO 1000
C
350   WRITE(6,2) '- Enter option number -------?'
      READ(5,*) OPTION
```

```
      CALL INPUTOPT(NAMESERV)
      GO TO 1000
 400  WRITE(6,411)
 411  FORMAT(/,'If you want to change MHS direction,press the key[C]',
     $/,5x,'otherwise press any keys. ????',$)
      READ(5,5) CHAN
      IF(CHAN.EQ.'C'.OR.CHAN.EQ.'c') THEN
            WRITE(6,2) '- Enter type of MHS direction-----?'
            READ(5,*) MHSDIR
      END IF
C
      GO TO(410,420,430) MHSDIR
C
 410  CALL MAKESIS(1,NAMESERV)
      GO TO 1000
 420  CALL MAKESIS(2,NAMESERV)
      GO TO 1000
 430  CALL MAKESIS(3,NAMESERV)
      GO TO 1000
C
 450  WRITE(6,2)'- Enter job generation type ----?'
      READ(5,*) TYEVT
 1000 CONTINUE
      RETURN
      END
```

```
c--   this is written for FMS logic programs------
c--        update:oct 2 1986
c
      SUBROUTINE ARRIVE
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 TASK,B1,B2,B3,FLAG,NIND,NO
      INTEGER*2 N1,N2,N3,N4,STANO,WSNO
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25)
      INTEGER*2 NVALUE,JOBT,SERV,TYEVT,NCHWS(45),NBLOCK(25)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/INSP/INSPEC
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/RAND/NVALUE,PROBD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      CALL TRNCOPY(JOBT,TASK,SERV,B1,B2,B3,FLAG,ATIME)
      CALL CHEKWS(JOBT,TASK,SERV,FLAG,NIND,NO)
      CALL INDEX(STANO,WSNO,N3,N4,JOBT,SERV,NIND,FLAG)
c
      IF(FLAG.EQ.1) THEN
         CALL LOADST(STANO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
         RETURN
      ELSE IF(SERV.EQ.2) THEN
         CALL MACHINE(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
         RETURN
      ELSE IF(SERV.EQ.3) THEN
         CALL INSPECT(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
         RETURN
      ELSE
         CALL OTHERS(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      END IF
      RETURN
      END
C
C
```

```
      SUBROUTINE DEPART
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 JOBT,TASK,SERV,B1,B2,B3,FLAG,NIND,NO,WSNO,JOBTQ,SERVQ
      INTEGER*2 N1,N2,N3,N4,TASKQ,FLAQ,NINQ,TYPE,TYEVT,TYSER
      INTEGER*2 NCHWS(45),NBLOK(25),STANO
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,ATIME,QTIME,BTIME,TR,DELAY
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TAVE,RN1,RN2
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/RAND/NVALUE,PROBD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      NEPT=0
      CALL TRNCOPY(JOBT,TASK,SERV,IB1,IB2,IB3,FLAG,ATIME)
C
      IF(JOBT.GT.0) GO TO 50
c
c--- schedule maintenance,breakdown -----------
c
      TYPE=SERV
      NIND=TASK
      IF(TYPE.EQ.2) THEN
          NIDL(IB1,NIND)=NIDL(IB1,NIND)+1
          IF(NIDL(IB1,NIND).NE.1) CALL ERR(001,'DEPART')
          WSNO=IB2
          GO TO 400
      ELSE
          NIDL(IB1,NIND)=NIDL(IB1,NIND)+1
          IF(NIDL(IB1,NIND).NE.1) CALL ERR(002,'DEPART')
          WSNO=IB2
          MAT1=TIME+MAT1
          CALL DISTRI(2,TYMAT,TIREC,TAVE,TUNI,TERM)
          MAT2=MAT1+TERM
          GO TO 400
      END IF
      RETURN
C
 50   CALL CHEKWS(JOBT,TASK,SERV,FLAG,NIND,NO)
      NIDL(SERV,NIND)=NIDL(SERV,NIND)+1
```

```
      IF(NIDL(SERV,NIND).NE.1) CALL ERR(003,'DEPART')
      CALL INDEX(STANO,WSNO,N3,N4,JOBT,SERV,NIND,FLAG)
      NGWS(WSNO)=NGWS(WSNO)-1
C
 400  IF(LSIZE(WSNO).EQ.0) THEN
            BUSY=0.
            CALL TIMEST(BUSY,WSNO)
      ELSE
            CALL QREMOV(1,WSNO,DELAY,QTIME,JOBTQ,TASKQ,SERVQ,B2,B3,NINQ,
     &      B5,FLAQ,BTIME)
c
            IF(FLAQ.LE.3) THEN
                 NINQ=ROUTE(JOBTQ,TASKQ,SERVQ)
                 NEPT=0
            ELSE
                 NEPT=NINQ
            END IF
C
            IF(SERVQ.EQ.2) THEN
                 K=CTASK(JOBTQ,TASKQ)
                 CALL CHEKTOL(K,NINQ,TOC)
            ELSE
                 TOC=0
            ENDIF
C
            A1=MSERVT(JOBTQ,TASKQ,SERVQ)
            CALL DISTRI(2,TYSER,A1,TAVE,TUNI,ETIME)
            TR=TIME+ETIME+TOC
            CALL SCHEDUL(TR,2,JOBTQ,TASKQ,SERVQ,0,NEPT,0,FLAQ,BTIME)
            CALL STATWS(DELAY,TR,2,JOBTQ,TASKQ,SERVQ,NINQ,FLAQ,BTIME)
      END IF
      IF(ATIME.EQ.0.) RETURN
C
C
      IF(SERV.EQ.3) GO TO 200
 300  CALL DISTRI(2,TYLT,SMHS,TAVE,TUNI,ETIME)
      TR=TIME+ETIME
      IF(TASK.EQ.NTASKS(JOBT)) GO TO 100
c
c------ check use of same work station -----
c
      CALL DEFINE(2,NOW,JOBT,TASK,SERV,NIND,FLAG)
      NWS=NOW+NOWS
      IF(NWS.EQ.WSNO) THEN
           TR=TIME
           TASK=TASK+1
           CALL SCHEDUL(TR,1,JOBT,TASK,SERV,0,NIND,0,2,ATIME)
           RETURN
      END IF
c
```

```
      IF(NO.EQ.1) THEN
          FLAG=5
      ELSE
          NIND=0
          FLAG=2
      END IF
C
      CALL SCHEDUL(TR,3,JOBT,TASK,SERV,0,NIND,0,FLAG,ATIME)
      RETURN
 100  IF(NO.EQ.0) THEN
         NIND=0
      END IF
      CALL SCHEDUL(TR,3,JOBT,TASK,SERV,0,NIND,0,10,ATIME)
      RETURN
C
 200  CALL DISTRI(2,5,0.,0.,TUNI,RN1)
c
c ---- check previous work station ----
c
      IPTAS=TASK-1
      CALL FINSERV(JOBT,IPTAS,ROUTE,NOSERV,IPSER)
      IPSER=ROUTE(JOBT,IPTAS,IPSER)
      IF((1.-RN1).GT.RDEF(IPSER)) GO TO 300
      CALL DISTRI(2,5,0.,0.,TUNI,RN2)
c
      IF((1.-RN2).GT.RREW(IPSER)) THEN
          NSUM=NREW(JOBT)
          NSUM=NSUM+1
          NREW(JOBT)=NSUM
          CALL DISTRI(2,TYLT,SMHS,TAVE,TUNI,ETIME)
          TR=TIME+ETIME
C
          IF(NO.EQ.1) THEN
            FLAG=6
          ELSE
            NIND=0
            FLAG=3
          END IF
          CALL SCHEDUL(TR,3,JOBT,TASK,SERV,0,NIND,0,FLAG,ATIME)
      ELSE
          NSUM=NSCR(JOBT)
          NSUM=NSUM+1
          NSCR(JOBT)=NSUM
          CALL SCHEDUL(TIME,3,JOBT,TASK,SERV,STANO,0,1,5,ATIME)
      END IF
      RETURN
      END
C
C
      SUBROUTINE MHSARR
```

```
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25)
      INTEGER*2 JOBT,TASK,SERV,PREV,MID,DEST,FLAG,STANO,START
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 NIND,NEPT,N1,N2,N3,N4,TYSER,TYEVT
      INTEGER*2 NSER,NEST,PSER,PRST
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,ATIME
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,NPRO,OPTION
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      NEPT=0
      NIND=0
      K=0
C
      CALL TRNCOPY(JOBT,TASK,SERV,PREV,MID,DEST,FLAG,ATIME)
      IF(FLAG.EQ.9.AND.MID.EQ.0) GO TO 100
      IF(PREV.EQ.0)  GO TO 700
      IF(PREV.GT.0.AND.MID.GT.0) THEN
          K=2
          GO TO 350
      ELSE
          CALL ERR(402,'MHSARR')
          RETURN
      END IF
C
  700 K=1
      IF(MID.GT.0) THEN
          IF(FLAG.EQ.10) THEN
               NIND=MID
               GO TO 300
          ELSE
               NIND=ROUTE(JOBT,TASK,SERV)
               NEPT=MID
              GO TO 300
          END IF
      ELSE
          NIND=ROUTE(JOBT,TASK,SERV)
          GO TO 300
      END IF
C
  300 CALL INDEX(STANO,N2,N3,N4,JOBT,SERV,NIND,FLAG)
C
      IF(K.EQ.1) GO TO 100
```

```
 350  IF(K.EQ.2) THEN
           CALL PASS(MHSDIR,PREV,MID,DEST,JOBT,TASK,SERV,FLAG,ATIME)
           RETURN
      ELSE
           CALL ERR(403,'MHSARR')
           RETURN
      END IF
C
 100  IF(FLAG.EQ.1)THEN
         START=1
         MID=0
         NKK=FLAG+1
         CALL DEFINE(1,DEST,JOBT,TASK,SERV,NIND,NKK)
      ELSE IF(FLAG.EQ.2) THEN
         START=STANO
         MID=0
         CALL DEFINE(2,DEST,JOBT,TASK,SERV,NIND,FLAG)
      ELSE IF(FLAG.EQ.3) THEN
         START=STANO
         MID=0
         CALL DEFINE(3,DEST,JOBT,TASK,SERV,NIND,FLAG)
      ELSE IF(FLAG.EQ.9) THEN
         START=PREV
         MID=0
         DEST=1
      ELSE IF(FLAG.EQ.10) THEN
         START=STANO
         MID=0
         DEST=1
      ELSE IF(FLAG.EQ.4) THEN
         START=STANO
         MID=0
         CALL DEFINE(1,DEST,JOBT,TASK,SERV,NEPT,FLAG)
      ELSE IF(FLAG.EQ.5) THEN
         CALL DEFINE(1,START,JOBT,TASK,SERV,NEPT,FLAG)
         MID=0
         CALL DEFINE(2,DEST,JOBT,TASK,SERV,0,FLAG)
      ELSE IF(FLAG.EQ.6) THEN
         CALL DEFINE(1,START,JOBT,TASK,SERV,NEPT,FLAG)
         MID=0
         CALL DEFINE(3,DEST,JOBT,TASK,SERV,0,FLAG)
      ELSE
         CALL ERR(901,'ASSIGN')
         RETURN
      END IF
C
      CALL PASS(MHSDIR,START,MID,DEST,JOBT,TASK,SERV,FLAG,ATIME)
      RETURN
      END
C
```

```
C
      SUBROUTINE MHSDEP
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25),NBLOK(25)
      INTEGER*2 JOBT,TASK,SERV,MID,DEST,FLAG,JOBTQ,TASKQ,SERVQ,MIDQ
      INTEGER*2 B2,DESTQ,FLAQ,NINQ,NIND,PREV,PREVQ,STANO,NCHWS(45)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 TYSER,TYEVT
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,ATIME,DELAY,QTIME,BTIME,TR
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      K=0
c
      CALL TRNCOPY(JOBT,TASK,SERV,PREV,MID,DEST,FLAG,ATIME)
      RFLOW(PREV,MID)=RFLOW(PREV,MID)-1
c
      IF(MID.EQ.DEST) GO TO 100
      K=1
      TR=TIME
      CALL SCHEDUL(TR,3,JOBT,TASK,SERV,PREV,MID,DEST,FLAG,ATIME)
c
      IF(NOMHS(MID).GT.0) THEN
           IS=NOMHS(MID)
           GO TO 110
      END IF
      RETURN
C
 100  NOMHS(DEST)=NOMHS(DEST)+1
      IS=NOMHS(DEST)
C
 110  IF(LSIZE(MID).EQ.0) GO TO 120
      IF(LSIZE(MID).LT.NOMHS(MID)) THEN
           IS=LSIZE(MID)
      END IF
c ----- check the status of block,if not,scheduled queue events.
c
      IR=LSIZE(MID)
      IP=NOMHS(MID)
```

```
C
      DO 500 INO=1,IS
 800  CALL QREMOV(1,MID,DELAY,QTIME,JOBTQ,TASKQ,SERVQ,IB,PREVQ,MIDQ,
     &DESTQ,FLAQ,BTIME)
C
      IF(MHSDIR.EQ.1) GO TO 600
      IF(IR.LE.0)   GO TO 500
      IF(RFLOW(MIDQ,PREVQ).GT.0) THEN
           IF(IR.GT.IP) THEN
                IR=IR-1
                IF(IR.GT.0) THEN
                     CALL GOTOQU(2,MID,QTIME,JOBTQ,TASKQ,SERVQ,IB,PREVQ,
     $               MIDQ,DESTQ,FLAQ,BTIME)
                     GO TO 800
                 END IF
                 GO TO 500
            ELSE
                 CALL GOTOQU(2,MID,QTIME,JOBTQ,TASKQ,SERVQ,IB,PREVQ,
     $            MIDQ,DESTQ,FLAQ,BTIME)
                 GO TO 500
            END IF
      ELSE IF(IB.EQ.1) THEN
           IBLOK(PREVQ,MIDQ)=IBLOK(PREVQ,MIDQ)-1
      END IF
C
 600  STANO=PREVQ
      RFLOW(PREVQ,MIDQ)=RFLOW(PREVQ,MIDQ)+1
      DIST=ABS(LDIS(PREVQ,MIDQ))
      TR=TIME+DIST/VEL
      CALL SCHEDUL(TR,4,JOBTQ,TASKQ,SERVQ,PREVQ,MIDQ,DESTQ,FLAQ,BTIME)
      CALL STATMH(DELAY,JOBTQ,STANO)
 500  CONTINUE
C
      IF(K.EQ.1) RETURN
      GO TO 130
 120  BUSY=0.
      CALL TIMEST(BUSY,DEST)
      IF(K.EQ.1) RETURN
C
 130  IF(FLAG.EQ.10) GO TO 200
      IF(FLAG.GE.4.AND.FLAG.LE.6) GO TO 150
      NIND=0
      GO TO 200
C
 150  I=2
 300  IF(I.EQ.SERV) GO TO 400
      DEST=DEST-NSGRO(I)
      I=I+1
      GO TO 300
 400  NIND=DEST-1
```

```
C
 200  IF(FLAG.EQ.10) THEN
          THRU=TIME-ATIME
          IDX4=NOSTA+NOWS+NTYP+JOBT
          CALL SAMPST(THRU,IDX4)
          NSUM=NFIN(JOBT)
          NSUM=NSUM+1
          NFIN(JOBT)=NSUM
          CALL LOADST(1,0,0,0,0,0,0.)
          RETURN
      ELSE IF(FLAG.EQ.9) THEN
          CALL LOADST(1,0,0,0,0,0,0.)
          RETURN
      ELSE IF(FLAG.EQ.1) THEN
          TR=TIME
          CALL SCHEDUL(TR,1,JOBT,TASK,SERV,0,0,0,2,ATIME)
          RETURN
      ELSE IF(FLAG.EQ.4) THEN
          TR=TIME
          CALL SCHEDUL(TR,1,JOBT,TASK,SERV,0,NIND,0,4,ATIME)
      ELSE IF(FLAG.EQ.3.OR.FLAG.EQ.6) THEN
          TR=TIME
          TASK=TASK-1
          CALL FINSERV(JOBT,TASK,ROUTE,NOSERV,SERV)
          CALL SCHEDUL(TR,1,JOBT,TASK,SERV,0,0,0,3,ATIME)
          RETURN
      ELSE
          TR=TIME
          TASK=TASK+1
          CALL FINSERV(JOBT,TASK,ROUTE,NOSERV,SERV)
          CALL SCHEDUL(TR,1,JOBT,TASK,SERV,0,0,0,2,ATIME)
          RETURN
      END IF
      RETURN
      END
C
C
      SUBROUTINE LOADST(STANO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25),NOSTA
      INTEGER*2 START MID,DEST,TYEVT,NCHWS(45),NBLOK(25)
      INTEGER*2 STANO,JOBT,TASK,SERV,NIND,FLAG,NSER,AJOB
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,ATIME,TR,DELAY
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
```

```
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/RAND/NVALUE,PROD(10)
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      IF(STANO.EQ.1) GO TO 100
         CALL ERR(202,'LOADST')
         RETURN
C
 100  IF(TYEVT.EQ.1) GO TO 105
      IF(TYEVT.EQ.2) GO TO 120
 105  IF(FINTYP.EQ.1) GO TO 110
      IF(FINTYP.EQ.2) GO TO 200
      IF(LENGTH.LT.TIME) GO TO 120
      GO TO 110
 200  IF(NTOT.GE.NPRO) GO TO 120
 110  IF(JOBT.EQ.0) THEN
         LS=LSIZE(1)
      ELSE
         LS=LSIZE(1)+1
      END IF
c
      IF(LS.GE.QS(1,1)) GO TO 120
 255  CALL DISTRI(2,TYARR,MARRVT,TAVE,TUNI,ETIME)
      TR=TIME+ETIME
         IF(FINTYP.EQ.3) THEN
               IF(TR.GT.LENGTH) THEN
                  IF(TIME.GT.(LENGTH-MARRVT).AND.TIME.LT.LENGTH) GO TO 120
                  GO TO 255
               END IF
         END IF
      CALL RANDI(2,AJOB)
      NSUM=NUTY(AJOB)
      NSUM=NSUM+1
      NUTY(AJOB)=NSUM
      CALL FINSERV(AJOB,1,ROUTE,NOSERV,NSER)
      NTOT=NTOT+1
      CALL SCHEDUL(TR,1,AJOB,1,NSER,0,0,0,1,TR)
c
 120  IF(JOBT.EQ.0) RETURN
      TR=TIME
      CALL SCHEDUL(TR,3,JOBT,TASK,SERV,0,0,0,1,ATIME)
      RETURN
      END
C
C
      SUBROUTINE MACHINE(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
```

```
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 WSNO,JOBT,TASK,SERV,NIND,FLAG,TYEVT
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,ATIME,DELAY,TR
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      K=CTASK(JOBT,TASK)
      CALL PLAN(OPTION,WSNO,IK,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      IF(IK.EQ.0) GO TO 100
      RETURN
C
 100  CALL CHEKTOL(K,NIND,TOC)
      DELAY=0
      A1=MSERVT(JOBT,TASK,SERV)
      NGWS(WSNO)=NGWS(WSNO)+1
      CALL DISTRI(2,TYSER,A1,TAVE,TUNI,ETIME)
      TR=TIME+ETIME+TOC
      CALL SCHEDUL(TR,2,JOBT,TASK,SERV,0,0,0,FLAG,ATIME)
      CALL STATWS(DELAY,TR,2,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      RETURN
      END
C
C
      SUBROUTINE INSPECT(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 INSPEC(10,10)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 WSNO,JOBT,TASK,SERV,NIND,FLAG,TYEVT
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,ATIME,DELAY,TR
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
```

```
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      CALL PLAN(OPTION,WSNO,IK,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      IF(IK.EQ.0) GO TO 100
      RETURN
C
 100  DELAY=0
      A1=MSERVT(JOBT,TASK,SERV)
      NGWS(WSNO)=NGWS(WSNO)+1
      CALL DISTRI(2,TYSER,A1,TAVE,TUNI,ETIME)
      TR=TIME+ETIME
      CALL SCHEDUL(TR,2,JOBT,TASK,SERV,0,NIND,0,FLAG,ATIME)
      CALL STATWS(DELAY,TR,2,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      RETURN
      END
C
C
      SUBROUTINE OTHERS(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 WSNO,JOBT,TASK,SERV,NIND,FLAG,TYEVT
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,ATIME,TR,DELAY
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      IF(NIDL(SERV,NIND).GT.0) THEN
         DELAY=0.
         A1=MSERVT(JOBT,TASK,SERV)
         CALL DISTRI(2,TYSER,A1,TAVE,TUNI,ETIME)
         TR=TIME+ETIME
         CALL SCHEDUL(TR,2,JOBT,TASK,SERV,0,0,0,FLAG,ATIME)
         CALL STATWS(DELAY,TR,2,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      ELSE
         TR=TIME
         CALL GOTOQU(2,WSNO,TR,JOBT,TASK,SERV,FLAG,0,0,0,0,ATIME)
      END IF
      RETURN
      END
C
C
```

```
C
      SUBROUTINE DEFINE(NO,DEST,JOBT,TASK,SERV,NIND,FLAG)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 NSER,NEST,NOTA,PSER,PRST,JOBT,TASK,SERV,NIND,FLAG
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      GO TO(100,200,300) NO
C
 100  CALL INDEX(DEST,N2,N3,N4,JOBT,SERV,NIND,FLAG)
      RETURN
C
 200  NOTA=TASK+1
      CALL FINSERV(JOBT,NOTA,ROUTE,NOSERV,NSER)
      NEST=ROUTE(JOBT,NOTA,NSER)
      CALL INDEX(DEST,N2,N3,N4,JOBT,NSER,NEST,FLAG)
      RETURN
C
 300  NOTA=TASK-1
      CALL FINSERV(JOBT,NOTA,ROUTE,NOSERV,PSER)
      PRST=ROUTE(JOBT,NOTA,PSER)
      CALL INDEX(DEST,N2,N3,N4,JOBT,PSER,PRST,FLAG)
      RETURN
      END
C
C
      SUBROUTINE INDEX(N1,N2,N3,N4,JOBT,SERV,NIND,FLAG)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 JOBT,SERV,NIND,FLAG,TOTST,NCHWS(45),NBLOK(25)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      TOTST=NOSTA+NOWS
C
      IF(FLAG.EQ.1) GO TO 100
      IF(SERV.EQ.2) GO TO 200
      IF(SERV.EQ.3) GO TO 300
      N1=1
      N2=NOSTA
      J=2
 500  IF(J.EQ.SERV) GO TO 400
```

```
      N1=N1+NSGRO(J)
      N2=N2+NSGRO(J)
      J=J+1
      GO TO 500
C
 400  N1=N1+NIND
      N2=N2+NIND
      GO TO 600
C
 300  N1=1+NSGRO(2)+NIND
      N2=NOSTA+NSGRO(2)+NIND
      GO TO 600
C
 200  N1=1+NIND
      N2=NOSTA+NIND
      GO TO 600
C
 100  N1=1
      N2=0
 600  N3=TOTST+JOBT
      N4=TOTST+NTYP+JOBT
      RETURN
      END
C
C
C
      SUBROUTINE AVAIL(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 INSPEC(10,10),NOWS,NOSTA,NCHWS(45),NBLOK(25)
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 WSNO,JOBT,TASK,SERV,NIND,EXTRA,TYEVT,FLAG
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,PROBD(10),LTIME,TR
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/INSP/INSPEC
      COMMON/TWO/IXX
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      EXTRA=0
      IF(FLAG.EQ.4) GO TO 100
```

```
C
      IF(IXX.EQ.1) GO TO 150
      IF(LSIZE(WSNO).GE.QS(SERV,NIND)) GO TO 150
         N=0
         GO TO 300
c
 150  DO 200 J=1,NSGRO(SERV)
      IF(SERV.EQ.2) THEN
           EXTRA=AVAMA(NIND,J)
      ELSE IF(SERV.EQ.3) THEN
           EXTRA=INSPEC(NIND,J)
      ELSE
           N=0
           GO TO 300
      END IF
C
      CALL INDEX(N1,N2,N3,N4,JOBT,SERV,EXTRA,FLAG)
      IF(NIDL(SERV,EXTRA).GT.0.AND.NGWS(N2).EQ.0) THEN
           N=1
           NIND=EXTRA
           NGWS(N2)=NGWS(N2)+1
           NCHWS(N2)=NCHWS(N2)+1
           GO TO 300
      ELSE
           N=0
      END IF
 200  CONTINUE
C
 300  CALL CHANGE(WSNO,N,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      RETURN
C
c---- schedule changed work station and calculated statistics.---
C
 100       A1=MSERVT(JOBT,TASK,SERV)
           CALL DISTRI(2,TYSER,A1,TAVE,TUNI,ETIME)
           TR=TIME+ETIME
           CALL SCHEDUL(TR,2,JOBT,TASK,SERV,0,NIND,0,4,ATIME)
           IF(NIDL(SERV,NIND).LE.0) THEN
                     DELAY=0.
                     BUSY=1.
                     CALL SAMPST(DELAY,WSNO)
                     IDX1=NOSTA+NOWS+JOBT
                     CALL SAMPST(DELAY,IDX1)
                     CALL TIMEST(BUSY,WSNO)
           END IF
      RETURN
      END
C
C
C
```

```
      SUBROUTINE PLAN(TYPE,WSNO,IK,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 TYPE,IK,WSNO,JOBT,TASK,SERV,NIND,FLAG,TYEVT
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 FAILR(2:3,10),MAT1,MAT2,TREC
      REAL*4 MARRVT,LTIME,TR,ETIME,RN
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/INSP/INSPEC
      COMMON/TWO/IXX
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      IXX=0
      IK=1
C
      IF(TYPE.EQ.1) GO TO 100
      IF(NIDL(SERV,NIND).GT.0) GO TO 200
      CALL AVAIL(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      RETURN
C
 200  IF(TYPE.EQ.2) GO TO 500
      IF(TYPE.GE.3) GO TO 400
 500  CALL DISTRI(2,5,0.,0.,TUNI,RN)
 550  IF((1.-RN).LE.FAILR(SERV,NIND)) THEN
           IXX=1
           NIDL(SERV,NIND)=NIDL(SERV,NIND)-1
           IF(NIDL(SERV,NIND).NE.0) CALL ERR(011,'PLAN')
           CALL DISTRI(2,TYREC,TREC,TAVE,TUNI,ETIME)
           TR=TIME+ETIME
           CALL SCHEDUL(TR,2,0,NIND,2,SERV,WSNO,0,0,0.)
           CALL AVAIL(WSNO,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      ELSE
           IK=0
      END IF
      RETURN
C
 400  IF(MAT1.GT.TIME.AND.MAT2.LE.TIME) THEN
            NIDL(SERV,NIND)=NIDL(SERV,NIND)-1
            IF(NIDL(SERV,NIND).NE.0) CALL ERR(012,'PLAN')
            TR=MAT2
            CALL SCHEDUL(TR,2,0,NIND,3,SERV,WSNO,0,0,0.)
            CALL CHANGE(WSNO,0,JOBT,TASK,SERV,NIND,FLAG,ATIME)
```

```
      ELSE IF(TYPE.EQ.3) THEN
             IK=0
             RETURN
      ELSE
             CALL DISTRI(2,5,0.,0.,TUNI,RN)
             GO TO 550
      END IF
      RETURN
C
 100  IF(NIDL(SERV,NIND).GT.0) THEN
              IF(NIDL(SERV,NIND).NE.1) CALL ERR(013,'PLAN')
              IK=0
      ELSE
              CALL CHANGE(WSNO,0,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      END IF
      RETURN
      END
C
C
C
C
      SUBROUTINE CHEKTOL(K,NIND,TOC)
      INTEGER*2 N,TYEVT
      INTEGER*2 MACHT(10,25),NTOL(10),LTOL(10),AVAMA(10,10),TYTOL
      INTEGER*2 FINTYP,MHSDIR,NPRO,TYARR,TYSER,TYMAT,TYLT,TYREC,OPTION
      REAL*4 FAILR,MAT1,MAT2,TREC
      REAL*4 LTIME,TOC
      COMMON/MODEL/FINTYP,MHSDIR,FAILR,MAT1,MAT2,TREC,NPRO,OPTION,TIREC
      COMMON/SERVT/TYARR,TYSER,TYMAT,TYREC,TYLT,TAVE,TYEVT,TUNI(2)
      COMMON/MACH/MACHT,NTOL,LTOL,AVAMA,LTIME,TYTOL
C
      N=0
      IS=MACHT(NIND,1)
C
      DO 100 I=1,NTOL(NIND)
      IF(MACHT(NIND,I).EQ.K) THEN
          N=1
          IK=I
          GO TO 200
      END IF
 100  CONTINUE
C
 200  IF(N.EQ.0) THEN
          CALL ERR(202,'CHEKTOL')
          RETURN
      ELSE IF(IS.EQ.K) THEN
          TOC=0
      ELSE
          CALL DISTRI(2,TYTOL,LTIME,TAVE,TUNI,TOC)
          MACHT(NIND,1)=K
```

```
          MACHT(NIND,IK)=IS
      END IF
      RETURN
      END
C
C
      SUBROUTINE CHEKWS(JOBT,TASK,SERV,FLAG,NIND,NO)
      INTEGER*2 JOBT,TASK,SERV,FLAG,NIND,NO
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),VEL,SMHS,LOAD,LENGTH
      REAL*4 MARRVT,LTIME
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      IF(FLAG.GE.4) THEN
          NIND=TRNSFR(7)
          NO=1
      ELSE
          NIND=ROUTE(JOBT,TASK,SERV)
          NO=0
      END IF
C
      IF(NIND.EQ.0) THEN
          CALL ERR(201,'CHEKWS')
      END IF
      RETURN
      END
C
C
      SUBROUTINE STATWS(DELAY,TM,EV,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 JOBT,TASK,SERV,NIND,FLAG,EV
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),LENGTH,MARRVT
      REAL*4 DELAY,BUSY,TM
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      CALL INDEX(N1,N2,N3,N4,JOBT,SERV,NIND,FLAG)
      CALL SAMPST(DELAY,N2)
      CALL SAMPST(DELAY,N3)
C
      BUSY=0.
      CALL TIMEST(BUSY,N2)
C
```

```
      NIDL(SERV,NIND)=NIDL(SERV,NIND)-1
      IF(NIDL(SERV,NIND).NE.0) THEN
             CALL ERR(020,'STATWS')
      ELSE
           BUSY=1.
           CALL TIMEST(BUSY,N2)
      END IF
      RETURN
      END
C
C
      SUBROUTINE CHANGE(WSNO,N,JOBT,TASK,SERV,NIND,FLAG,ATIME)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 JOBT,WSNO,TASK,SERV,NIND,FLAG,IDX2,N
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),LENGTH,MARRVT
      REAL*4 TR
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      IF(N.EQ.0) GO TO 200
      IF(N.EQ.1) GO TO 300
         CALL ERR(602,'CHANGE')
         RETURN
C
 300  TR=TIME
      NIDL(SERV,NIND)=NIDL(SERV,NIND)-1
      IF(NIDL(SERV,NIND).NE.0) CALL ERR(040,'CHANGE')
      CALL SCHEDUL(TR,3,JOBT,TASK,SERV,0,NIND,0,4,ATIME)
      RETURN
C
 200  TR=TIME
      CALL GOTOQU(2,WSNO,TR,JOBT,TASK,SERV,0,0,0,0,2,ATIME)
      RETURN
      END
C
C
      SUBROUTINE PASS(TYPE,START,MID,DEST,JOBT,TASK,SERV,FLAG,ATIME)
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25)
      INTEGER*2 JOBT,TASK,SERV,MID,DEST,FLAG,NCHWS(45),NBLOK(25)
      INTEGER*2 ISTA,DIS,DIR,NO,START,DIST,TYPE
      REAL*4 VEL,SMHS,LOAD
      REAL*4 ATIME,TR
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/LOC/CORDX,CORDY,NOMHS
```

```
      COMMON/MHS/VEL,SMHS,LOAD
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      ISTA=0
      DIS=0
      TR=0
      DIR=0
      NO=0
      IKK=0
C
      IF(MID.EQ.0) THEN
          ISTA=START
          IKK=1
      ELSE IF(MID.EQ.DEST) THEN
          ISTA=START
          NO=MID
      ELSE
          ISTA=MID
      END IF
C
      IF(NO.GT.0) GO TO 350
      DIS=LDIS(ISTA,DEST)
      IF(DIS.GT.0) THEN
         DIR=1
      ELSE IF(DIS.LT.0) THEN
          DIR=-1
      ELSE
          CALL ERR(701,'PASS')
          RETURN
      END IF
C
      CALL CHECK(DIR,ISTA,NO)
C
 350  IF(TYPE.EQ.1) GO TO 100
      IF(IKK.EQ.0) GO TO 500
      IF(NOMHS(ISTA).LE.0) GO TO 400
 500  IF(RFLOW(ISTA,NO).EQ.99) THEN
            CALL ERR(702,'PASS')
            RETURN
      ELSE IF(RFLOW(NO,ISTA).EQ.0.AND.IBLOK(NO,ISTA).EQ.0) THEN
              RFLOW(ISTA,NO)=RFLOW(ISTA,NO)+1
              DIST=ABS(LDIS(ISTA,NO))
              TR=TIME+DIST/VEL
              CALL SCHEDUL(TR,4,JOBT,TASK,SERV,ISTA,NO,DEST,FLAG,ATIME)
              IF(IKK.EQ.1) THEN
                     DELAY=0.
                     CALL STATMH(DELAY,JOBT,ISTA)
              END IF
              RETURN
      ELSE IF(RFLOW(NO,ISTA).GE.1.OR.IBLOK(NO,ISTA).GT.0) THEN
```

```
          IF(IKK.EQ.0)  THEN
               NOMHS(ISTA)=NOMHS(ISTA)+1
          END IF
          TR=TIME
          IBLOK(ISTA,NO)=IBLOK(ISTA,NO)+1
          NBLOK(ISTA)=NBLOK(ISTA)+1
          CALL GOTOQU(2,ISTA,TR,JOBT,TASK,SERV,1,ISTA,NO,DEST,FLAG,
     &ATIME)
           RETURN
      ELSE
           CALL ERR(703,'PASS')
           RETURN
      ENDIF
C
 100  IF(IKK.EQ.0) GO TO 600
      IF(NOMHS(ISTA).EQ.0) GO TO 400
 600  DO 200 I=1,NOSTA
C
      IF(RFLOW(ISTA,I).NE.99) THEN
          RFLOW(ISTA,I)=RFLOW(ISTA,I)+1
          NO=I
          DIST=ABS(LDIS(ISTA,NO))
          TR=TIME+DIST/VEL
          CALL SCHEDUL(TR,4,JOBT,TASK,SERV,ISTA,NO,DEST,FLAG,ATIME)
          IF(IKK.EQ.1) THEN
                CALL STATMH(0.,JOBT,ISTA)
          END IF
          RETURN
      END IF
 200  CONTINUE
 400  TR=TIME
      CALL GOTOQU(2,ISTA,TR,JOBT,TASK,SERV,0,ISTA,NO,DEST,FLAG,ATIME)
C
      RETURN
      END
C
C
      SUBROUTINE CHECK(DIR,P1,NUM)
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 I,NO,NUM,DIR,P1
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      I=0
      NO=0
      NUM=0
C
      IF(DIR.EQ.1) THEN
          CALL MIN(P1,NUM,NO)
          RETURN
```

```
      ELSE IF(DIR.EQ.-1) THEN
          CALL MAX(P1,NUM,NO)
          RETURN
      ELSE
          CALL ERR(704,'CHECK')
          RETURN
      END IF
      RETURN
      END
C
C
      SUBROUTINE MIN(P1,NO,IN)
      INTEGER*2 P1,NO,IN
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25),NOSTA,NCHWS(45),NBLOK(25)
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/GENS/NIDL,LDIS,RFOW
      IN=0
      NO=0
C
      DO 100 K=1,NOSTA
      IF(LDIS(P1,K).LT.0) GO TO 100
         IN=IN+1
      IF(P1.EQ.K) GO TO 100
         MINM=LDIS(P1,K)
         NO=K
         J=K+1
         GO TO 300
 100  CONTINUE
C
      RETURN
 300  DO 200 I=J,NOSTA
C
      IF(LDIS(P1,I).LT.0) GO TO 200
         IN=IN+1
      IF(P1.EQ.I) GO TO 200
      IF(MINM.GT.LDIS(P1,I)) THEN
         MINM=LDIS(P1,I)
         NO=I
      END IF
 200  CONTINUE
      RETURN
      END
C
C
      SUBROUTINE MAX(P1,NO,IM)
      INTEGER*2 P1,NO,IM
      INTEGER*2 NIDL(10,10),LDIS(25,25),RFLOW(25,25),QS(3,10)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25),NOSTA,NCHWS(45),NBLOK(25)
```

```
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/GENS/NIDL,LDIS,RFLOW,IBLOK(25,25),QS
C
      IM=0
      NO=0
C
      DO 100 K=1,NOSTA
      IF(LDIS(P1,K).GT.0) GO TO 100
         IM=IM+1
      IF(P1.EQ.K) GO TO 100
         MAXM=LDIS(P1,K)
         NO=K
         J=K+1
         GO TO 300
 100  CONTINUE
      RETURN
C
 300  DO 200 I=J,NOSTA
      IF(LDIS(P1,I).GE.0) GO TO 200
         IM=IM+1
      IF(P1.EQ.I) GO TO 200
      IF(MAXM.LT.LDIS(P1,I)) THEN
         MAXM=LDIS(P1,I)
         NO=I
      END IF
 200  CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE STATMH(DELAY,JOBT,STANO)
      INTEGER*2 ROUTE(10,25,10),CTASK(10,25),NTYP,NTASKS(10),NOSERV,
     &NSGRO(10),NUTY(10),NREW(10),NSCR(10),NFIN(10),NTOT,NGWS(45)
      INTEGER*2 CORDX(25),CORDY(25),NOMHS(25),NOSTA
      INTEGER*2 N1,N2,N3,N4,JOBT,STANO,NCHWS(45),NBLOK(25)
      REAL*4 BUSY,DELAY
      REAL*4 MSERVT(10,25,10),RREW(10),RDEF(10),LENGTH,MARRVT
      COMMON/LOC/CORDX,CORDY,NOMHS
      COMMON/JOB/ROUTE,MSERVT,CTASK,NTYP,NTASKS,NOSERV,RDEF,RREW,
     &NUTY,NREW,NSCR,NFIN,LENGTH,NSGRO,MARRVT,NTOT,NGWS
      COMMON/STAT/NOWS,NOSTA,NCHWS,NBLOK
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      CALL SAMPST(DELAY,STANO)
      N3=NOWS+NOSTA+JOBT
      CALL SAMPST(DELAY,N3)
C
      NOMHS(STANO)=NOMHS(STANO)-1
```

```
      IF(NOMHS(STANO).EQ.0) THEN
         BUSY=1.
         CALL TIMEST(BUSY,STANO)
      END IF
C
      RETURN
      END
C
C
      SUBROUTINE FINSERV(JOBT,TASK,ROUTE,NOSERV,N)
      INTEGER*2 ROUTE(10,25,10),NOSERV,JOBT,TASK,N
C
      N=0
      I=2
  200 IF(ROUTE(JOBT,TASK,I).GT.0) GO TO 100
      I=I+1
      IF(I.LE.NOSERV) GO TO 200
      CALL ERR(802,'FINSERV')
      RETURN
  100 N=I
      RETURN
      END
C
C
      SUBROUTINE ERR(NUM,SURB)
      INTEGER*2 NUM
      CHARACTER SURB*10
C
      WRITE(6,100) NUM,SURB
  100 FORMAT(10X,' ** ERROR ',I3,' ** IS IN SUBROUTINE(OR,FUN.) ',A10)
      RETURN
      END
C
C
      SUBROUTINE TEX(NAME,IX,IY,IZ,AT)
      INTEGER*2 IX,IY,IZ
      REAL*4 AT
      CHARACTER NAME*10
C
      WRITE(6,100) NAME,IX,IY,IZ,AT
  100 FORMAT('SUBROUTINE--',A10,3(I3,3X),F13.5)
      RETURN
      END
```

```
      SUBROUTINE SCHEDUL(TR,ET,JOBT,TASK,SERV,PREV,MID,DEST,FLAG,ATIME)
      INTEGER*2 JOBT,TASK,SERV,PREV,MID,DEST,FLAG,ET
      REAL*4 TR, ATIME
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      TRNSFR(1)=TR
      TRNSFR(2)=ET
      TRNSFR(3)=JOBT
      TRNSFR(4)=TASK
      TRNSFR(5)=SERV
      TRNSFR(6)=PREV
      TRNSFR(7)=MID
      TRNSFR(8)=DEST
      TRNSFR(9)=FLAG
      TRNSFR(10)=ATIME
C
      CALL FILE(3,50)
      RETURN
      END
C
C
      SUBROUTINE GOTOQU(NO,IFN,TM,JOBT,TASK,SERV,B1,PREV,MID,DEST,FLAG,
     &ATIME)
      INTEGER*2 JOBT,TASK,SERV,PREV,MID,DEST,FLAG,B1,NO,IFN
      REAL*4 ATIME,TM
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
      TRNSFR(1)=TM
      TRNSFR(2)=JOBT
      TRNSFR(3)=TASK
      TRNSFR(4)=SERV
      TRNSFR(5)=B1
      TRNSFR(6)=PREV
      TRNSFR(7)=MID
      TRNSFR(8)=DEST
      TRNSFR(9)=FLAG
      TRNSFR(10)=ATIME
      CALL FILE(NO,IFN)
      RETURN
      END
C
C
      SUBROUTINE TRNCOPY(JOBT,TASK,SERV,PREV,MID,DEST,FLAG,ATIME)
      INTEGER*2 JOBT,TASK,SERV,PREV,DEST,FLAG
      REAL*4 ATIME
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      JOBT=TRNSFR(3)
      TASK=TRNSFR(4)
      SERV=TRNSFR(5)
      PREV=TRNSFR(6)
```

```
      MID=TRNSFR(7)
      DEST=TRNSFR(8)
      FLAG=TRNSFR(9)
      ATIME=TRNSFR(10)
      RETURN
      END
C
C
      SUBROUTINE QREMOV(NO,IFN,DELAY,QTIME,JOBTQ,TASKQ,SERVQ,B2,PREVQ,
     &MIDQ,DESTQ,FLAQ,BTIME)
      INTEGER*2 JOBTQ,TASKQ,SERVQ,B2,PREVQ,MIDQ,DESTQ,FLAQ,NO,IFN
      REAL*4 QTIME,BTIME,DELAY
      COMMON/SYSTEM/LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
      CALL REMOVE(NO,IFN)
      QTIME=TRNSFR(1)
      JOBTQ=TRNSFR(2)
      TASKQ=TRNSFR(3)
      SERVQ=TRNSFR(4)
      B2=TRNSFR(5)
      PREVQ=TRNSFR(6)
      MIDQ=TRNSFR(7)
      DESTQ=TRNSFR(8)
      FLAQ=TRNSFR(9)
      BTIME=TRNSFR(10)
      DELAY=TIME-QTIME
      RETURN
      END
C
C
C
      SUBROUTINE INITLK
      INTEGER HEAD(50),LINKPR(1500),LINKSR(1500),LIST,NAR,ROW,TAIL(50)
      REAL MASTER(1500,10)
      COMMON /LLISTS/ HEAD,LINKPR,LINKSR,MASTER,NAR,TAIL
      COMMON /SYSTEM/ LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C *** INITIALIZE LINKS.
C
      DO 10 ROW=1,1500
      LINKPR(ROW)=0
      LINKSR(ROW)=ROW+1
10    CONTINUE
      LINKSR(1500)=0
C
C *** INITIALIZE LIST ATTRIBUTES.
C
      DO 20 LIST=1,50
      HEAD(LIST)=0
      TAIL(LIST)=0
```

```
      LSIZE(LIST)=0
      LRANK(LIST)=0
20    CONTINUE
C
C *** INITIALIZE SYSTEM ATTRIBUTES.
C
      TIME=0.
      NAR=1
      LRANK(50)=1
      MAXATR=10
C
C *** INITIALIZE STATISTICAL ROUTINES.
C
      CALL SAMPST(0.,0)
      CALL TIMEST(0.,0)
      RETURN
      END
C
C
C
      SUBROUTINE FILE(OPTION,LIST)
      INTEGER AHEAD,HEAD(50),IHEAD,ITAIL,ITEM,LINKPR(1500),
     1LINKSR(1500),LIST,NAR,OPTION,ROW,TAIL(50),BEHIND
      REAL MASTER(1500,10),SIZE
      COMMON /LLISTS/ HEAD,LINKPR,LINKSR,MASTER,NAR,TAIL
      COMMON /SYSTEM/ LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C *** IF THE MASTER STORAGE ARRAY IS FULL, STOP THE SIMULATION.
C
      IF(NAR .GT. 0) GOTO 20
      PRINT 10,TIME
10    FORMAT(1H1,5X,'MASTER STORAGE ARRAY OVERFLOW AT TIME ',E10.3)
      STOP
C
C *** IF THE LIST VALUE IS IMPROPER, STOP THE SIMULATION.
C
20    IF((LIST .GE. 1) .AND. (LIST .LE. 50)) GOTO 40
      PRINT 30,LIST,TIME
30    FORMAT(1H1,I10,' IS AN IMPROPER VALUE FOR FILE LIST AT TIME ',
     1E10.3)
      STOP
C
C *** INCREMENT THE LIST SIZE.
C
40    LSIZE(LIST)=LSIZE(LIST)+1
C
C *** IF THE OPTION VALUE IS IMPROPER, STOP THE SIMULATION.
C
      IF((OPTION .GE. 1) .AND. (OPTION .LE. 4)) GOTO 60
      PRINT 50,OPTION,TIME
```

```
50    FORMAT(1H1,I10,' IS AN IMPROPER VALUE FOR FILE OPTION AT TIME ',
     1E10.3)
      STOP
C
C *** FILE ACCORDING TO THE DESIRED OPTION.
C
60    GOTO (300,200,100,100),OPTION
C
C *******************************************************************
C
C *** TEH LIST IS RANKED. DETERMINE THE ITEM ON WHICH THE LIST IS TO
C *** BE RANKED.
C
100   ITEM=LRANK(LIST)
C
C *** IF AN INVALID ITEM HAS BEEN SPECIFIED, STOP THE SIMULATION.
C
      IF((ITEM .GE. 1) .AND. (ITEM .LE. MAXATR)) GOTO 120
      PRINT 110,ITEM,LIST
110   FORMAT(1H1,I10,' IS AN IMPROPER VALUE FOR THE RAND OF LIST ',I2)
      STOP
C
C *** IF THIS IS NOT THE FIRST RECORD IN THIS LIST, CONTINUE.
C
120   IF(LSIZE(LIST) .EQ. 1) GOTO 400
C
C *** SEARCH THE LIST FOR THE PROPER LOCATION.
C
      ROW=HEAD(LIST)
130   IF(OPTION .EQ. 4) GOTO 140
C
C *** RANK THE LIST IN INCREASING ORDER.
C
      IF(TRNSFR(ITEM) .GE. MASTER(ROW,ITEM)) GOTO 160
C
C *** THE CORRECT LOCATION HAS BEEN FOUND.
C
      GOTO 150
C
C *** RANK THE LIST IN DECREASING ORDER.
C
140   IF(TRNSFR(ITEM) .LE. MASTER(ROW,ITEM)) GOTO 160
C
C *** THE CORRECT LOCATION HAS BEEN FOUND.
C
C *** INSERT BEFORE THE LAST RECORD EXAMINED.
C
150   IF(ROW .EQ. HEAD(LIST)) GOTO 300
C
C *** INSERT IN THE PROPER LOCATION BETWEEN THE PRECEDING AND
```

```
C *** SUCCEEDING RECORDS (BEHIND AND AHEAD).
C
      AHEAD=LINKSR(BEHIND)
      ROW=NAR
      NAR=LINKSR(ROW)
      IF(NAR .GT. 0) LINKPR(NAR)=0
      LINKPR(ROW)=BEHIND
      LINKSR(BEHIND)=ROW
      LINKPR(AHEAD)=ROW
      LINKSR(ROW)=AHEAD
C
C *** GOTO TRANSFER THE DATA.
C
      GOTO 500
C
C *** CONTINUE SEARCHING, CONSIDER THE NEXT ROW.
C
160   BEHIND=ROW
      ROW=LINKSR(BEHIND)
C
C *** IF THE LAST ROW CONSIDERED WAS NOT THE TAIL OF THE LIST,
C *** CONTINUE.
C
      IF(TAIL(LIST) .NE. BEHIND) GOTO 130
C
C *******************************************************************
C
C *** INSERT AFTER THA LAST RECORD IN THE LIST.
C
200   IF(LSIZE(LIST) .EQ. 1) GOTO 400
      ROW=NAR
      NAR=LINKSR(ROW)
      IF(NAR .GT. 0) LINKPR(NAR)=0
      ITAIL=TAIL(LIST)
      LINKPR(ROW)=ITAIL
      LINKSR(ITAIL)=ROW
      LINKSR(ROW)=0
      TAIL(LIST)=ROW
C
C *** GOTO TRANSFER THE DATA.
C
      GOTO 500
C
C *******************************************************************
C
C *** INSERT BEFORE THE FIRST RECORD IN THE LIST.
C
300   IF(LSIZE(LIST) .EQ. 1) GOTO 400
      ROW=NAR
      NAR=LINKSR(ROW)
```

```
      IF(NAR .GT. 0) LINKPR(NAR)=0
      IHEAD=HEAD(LIST)
      LINKPR(IHEAD)=ROW
      LINKSR(ROW)=IHEAD
      LINKPR(ROW)=0
      HEAD(LIST)=ROW
C
C *** GOTO TRANSFER THE DATA.
C
      GOTO 500
C
C ********************************************************************
C
C *** INSERT THE FIRST RECORD IN THE LIST.
C
400   ROW=NAR
      NAR=LINKSR(ROW)
      IF(NAR .GT. 0) LINKPR(NAR)=0
      LINKSR(ROW)=0
      HEAD(LIST)=ROW
      TAIL(LIST)=ROW
C
C ********************************************************************
C
C *** TRANSFER THE DATA.
C
500   DO 510 ITEM=1,MAXATR
      MASTER(ROW,ITEM)=TRNSFR(ITEM)
510   CONTINUE
C
C *** UPDATE THE AREA UNDER THE NUMBER IN LIST CURVE.
C
      SIZE=LSIZE(LIST)
      CALL TIMEST(SIZE,50+LIST)
      RETURN
      END
C
C
C
      SUBROUTINE REMOVE(OPTION,LIST)
      INTEGER HEAD(50),IHEAD,ITAIL,ITEM,LINKPR(1500),LINKSR(1500),LIST,
     1NAR,OPTION,ROW,TAIL(50)
      REAL MASTER(1500,10),SIZE
      COMMON /LLISTS/ HEAD,LINKPR,LINKSR,MASTER,NAR,TAIL
      COMMON /SYSTEM/ LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C *** IF THE LIST VALUE IS IMPROPER, STOP THE SIMULATION.
C
      IF((LIST .GE. 1) .AND. (LIST .LE. 50)) GOTO 20
      PRINT 10,LIST,TIME
```

```
10    FORMAT(1H1,I10,' IS AN IMPROPER VALUE FOR REMOVE LIST AT TIME ',
     1E10.3)
      STOP
C
C *** IF THE LIST IS EMPTY, STOP THE SIMULATION.
C
20    IF(LSIZE(LIST) .GT. 0) GOTO 40
      PRINT 30,LIST,TIME
30    FORMAT(1H1,5X,'UNDERFLOW OF LIST ',I2,' AT TIME ',E10.3)
      STOP
C
C *** DECREMENT THE LIST SIZE.
C
40    LSIZE(LIST)=LSIZE(LIST)-1
C
C *** IF THE OPTION VALUE IS IMPROPER, STOP THE SIMULATION.
C
      IF((OPTION .EQ. 1) .OR. (OPTION .EQ. 2)) GOTO 60
      PRINT 50,OPTION,TIME
50    FORMAT(1H1,I10,' IS AN IMPROPER VALUE FOR REMOVE OPTION AT TIME ',
     1E10.3)
      STOP
C
C *** IF THERE IS MORE THAN ONE RECORD IN THE LIST, CONTINUE.
C
60    IF(LSIZE(LIST) .EQ. 0) GOTO 300
C
C *** REMOVE ACCORDING TO THE DESIRED OPTION.
C
      GOTO (100,200),OPTION
C
C *********************************************************************
C
C *** REMOVE THE FIRST RECORD IN THE LIST.
C
100   ROW=HEAD(LIST)
      IHEAD=LINKSR(ROW)
      LINKPR(IHEAD)=0
      HEAD(LIST)=IHEAD
C
C *** GOTO TRANSFER THE DATA.
C
      GOTO 400
C
C *********************************************************************
C
C *** REMOVE THE LAST RECORD IN THE LIST.
C
200   ROW=TAIL(LIST)
      ITAIL=LINKPR(ROW)
```

```
      LINKSR(ITAIL)=0
      TAIL(LIST)=ITAIL
C
C *** GOTO TRANSFER THE DATA.
C
      GOTO 400
C
C *****************************************************************
C
C *** REMOVE THE ONLY RECORD IN THE LIST.
C
300   ROW=HEAD(LIST)
      HEAD(LIST)=0
      TAIL(LIST)=0
C
C *****************************************************************
C
C *** TRANSFER THE DATA.
C
400   LINKSR(ROW)=NAR
      LINKPR(ROW)=0
      NAR=ROW
      DO 410 ITEM=1,MAXATR
      TRNSFR(ITEM)=MASTER(ROW,ITEM)
410   CONTINUE
C
C *** UPDATE THE AREA UNDER THE NUMBER IN LIST CURVE.
C
      SIZE=LSIZE(LIST)
      CALL TIMEST(SIZE,50+LIST)
      RETURN
      END
C
C
      SUBROUTINE TIMING
      COMMON /SYSTEM/ LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C *** REMOVE THE FIRST EVENT FROM THE EVENT LIST.
C
      CALL REMOVE(1,50)
C
C *** CHECK FOR A TIME REVERSAL.
C
      IF(TRNSFR(1) .GE. TIME) GOTO 20
      PRINT 10,TRNSFR(2),TRNSFR(1),TIME
10    FORMAT(1H1,5X,' ATTEMPT TO SCHEDULING AN EVENT OF TYPE ',F3.0,
     1' AT TIME ',E10.3,' WHEN THE CLOCK IS ',E10.3)
      STOP
C
C *** ADVANCE THE SIMULATION CLOCK.
```

```
C
20    TIME=TRNSFR(1)
      NEXT=TRNSFR(2)
      RETURN
      END
C
C
C
      SUBROUTINE CANCEL(ETYPE)
      INTEGER AHEAD,BEHIND,HEAD(50),ITEM,LINKPR(1500),LINKSR(1500),NER,
     1ROW,TAIL(50)
      REAL ETYPE,HIGH,LOW,MASTER(1500,10),SIZE,VALUE
      COMMON /LLISTS/ HEAD,LINKPR,LINKSR,MASTER,NAR,TAIL
      COMMON /SYSTEM/ LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C *** SEARCH THE EVENT LIST.
C
      IF(LSIZE(50) .EQ. 0) RETURN
      ROW=HEAD(50)
      LOW=ETYPE-0.1
      HIGH=ETYPE+0.1
10    VALUE=MASTER(ROW,2)
      IF((LOW .LT. VALUE) .AND. (HIGH .GT. VALUE)) GOTO 20
C
C *** GOTO THE NEXT EVENT.
C
      IF(ROW .EQ. TAIL(50)) RETURN
      ROW=LINKSR(ROW)
      GOTO 10
C
C ********************************************************************
C
C *** CANCEL THIS EVENT.
C
20    IF(ROW .NE. HEAD(50)) GOTO 30
C
C *** REMOVE THE FIRST EVENT IN THE EVENT LIST.
C
      CALL REMOVE(1,50)
      RETURN
30    IF(ROW .NE. TAIL(50)) GOTO 40
C
C *** REMOVE THE LAST EVENT IN THE EVENT LIST.
C
      CALL REMOVE(2,50)
      RETURN
C
C *** REMOVE THIS EVENT WHICH IS SOMEWHERE IN THE MIDDLE OF THE EVENT
C *** LIST.
C
```

```
40    AHEAD=LINKSR(ROW)
      BEHIND=LINKPR(ROW)
      LINKSR(BEHIND)=AHEAD
      LINKPR(AHEAD)=BEHIND
      LINKSR(ROW)=NAR
      LINKPR(ROW)=0
      NAR=ROW
      LSIZE(50)=LSIZE(50)-1
C
C *** PLACE THE ATTRIBUTES OF THE CANCELED EVENT IN THE TRNSFR ARRAY.
C
      DO 50 ITEM=1,MAXATR
      TRNSFR(ITEM)=MASTER(ROW,ITEM)
50    CONTINUE
C
C *** UPDATE THE AREA UNDER THE NUMBER IN LIST CURVE.
C
      SIZE=LSIZE(50)
      CALL TIMEST(SIZE,100)
      RETURN
      END
C
C
C
      SUBROUTINE SAMPST(VALUE,VARIBL)
      INTEGER IVAR,NOBS(50),VARIBL
      REAL MAX(50),MIN(50),SUM(50),VALUE
      COMMON /SYSTEM/ LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C *** IF THE VALUE IS IMPROPER, STOP THE SIMULATION.
C
      IF((VARIBL .GE. -50) .AND. (VARIBL .LE. 50)) GOTO 20
      PRINT 10,VARIBL,TIME
10    FORMAT(1H1,I10,' IS AN IMPROPER VALUE FOR A SAMPST VARIABLE ',
     1' AT TIME ',E10.3)
      STOP
C
C *** EXECUTE THE DESIRED OPTION.
C
20    IF(VARIBL) 300,100,200
C
C ***********************************************************************
C
C *** INITIALIZE THE ROUTINE.
C
100   DO 110 IVAR=1,50
      SUM(IVAR)=0.
      MAX(IVAR)=-1.E+30
      MIN(IVAR)=1.E+30
      NOBS(IVAR)=0
```

```
110   CONTINUE
      RETURN
C
C ********************************************************************
C
C *** COLLECT DATA.
C
200   SUM(VARIBL)=SUM(VARIBL)+VALUE
      IF(VALUE .GT. MAX(VARIBL)) MAX(VARIBL)=VALUE
      IF(VALUE .LT. MIN(VARIBL)) MIN(VARIBL)=VALUE
      NOBS(VARIBL)=NOBS(VARIBL)+1
      RETURN
C
C ********************************************************************
C
C *** REPORT THE RESULTS.
C
 300  IVAR=-VARIBL
      IF(NOBS(IVAR).EQ.0) THEN
          TRNSFR(1)=0.
          TRNSFR(2)=0.
          TRNSFR(3)=0.
          TRNSFR(4)=0.
          RETURN
      ELSE
          TRNSFR(2)=NOBS(IVAR)
          TRNSFR(1)=SUM(IVAR)/TRNSFR(2)
          TRNSFR(3)=MAX(IVAR)
          TRNSFR(4)=MIN(IVAR)
      END IF
C
      RETURN
      END
C
C
      SUBROUTINE TIMEST(VALUE,VARIBL)
      INTEGER IVAR,NOB(100),VARIBL
      REAL AREA(100),MAX(100),MIN(100),PREVAL(100),TLVC(100),VALUE
      COMMON /SYSTEM/ LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C *** IF THE VARIABLE VALUE IS IMPROPER, STOP THE SIMULATION.
C
      IF((VARIBL .GE. -100) .AND. (VARIBL .LE. 100)) GOTO 20
      PRINT 10,VARIBL,TIME
10    FORMAT(1H1,I10,' IS AN IMPROPER VALUE FOR A TIMEST VARIABLE ',
     1' AT TIME ',E10.3)
      STOP
C
C *** EXECUTE THE DESIRED OPTION.
C
```

```
20    IF(VARIBL) 300,100,200
C
C ********************************************************************
C
C *** INITIALIZE THE ROUTINE.
C
100   DO 110 IVAR=1,100
      AREA(IVAR)=0.
      MAX(IVAR)=-1.E+30
      MIN(IVAR)=1.E+30
      PREVAL(IVAR)=0.
      TLVC(IVAR)=TIME
      NOB(IVAR)=0
110   CONTINUE
      TRESET=TIME
      RETURN
C
C ********************************************************************
C
C *** COLLECT DATA.
C
200   AREA(VARIBL)=AREA(VARIBL)+(TIME-TLVC(VARIBL))*PREVAL(VARIBL)
      IF(VALUE .GT. MAX(VARIBL)) MAX(VARIBL)=VALUE
      IF(VALUE .LT. MIN(VARIBL)) MIN(VARIBL)=VALUE
      PREVAL(VARIBL)=VALUE
      TLVC(VARIBL)=TIME
      NOB(VARIBL)=NOB(VARIBL)+1
      RETURN
C
C ********************************************************************
C
C *** REPORT THE RESULTS.
C
 300  IVAR=-VARIBL
      IF(NOB(IVAR).EQ.0) THEN
          TRNSFR(1)=0.
          TRNSFR(2)=0.
          TRNSFR(3)=0.
          TRNSFR(4)=0.
          RETURN
      ELSE
          AREA(IVAR)=AREA(IVAR)+(TIME-TLVC(IVAR))*PREVAL(IVAR)
          TLVC(IVAR)=TIME
          TRNSFR(1)=AREA(IVAR)/(TIME-TRESET)
          TRNSFR(2)=MAX(IVAR)
          TRNSFR(3)=MIN(IVAR)
          TRNSFR(4)=NOB(IVAR)
      END IF
C
      RETURN
```

```
      END
C
C
      SUBROUTINE FILEST(LIST)
      INTEGER ILIST,LIST
      COMMON /SYSTEM/ LRANK(50),LSIZE(50),MAXATR,NEXT,TIME,TRNSFR(10)
C
C
C *** COMPUTE SUMMARY STATISTICS FOR THE LIST.
C
      ILIST=-(50+LIST)
      CALL TIMEST(0.,ILIST)
      RETURN
      END
```