NASA CONTRACTOR
REPORT

NASA CR-61376

COMPUTER SYSTEM SIMULATION AND ANALYSIS

By T. Williams and J. E. Weatherbee
Computer Sciences Corporation
Field Services Division
Aerospace Systems Center
8300 S. Whitesburg Drive
Huntsville, Ala. 35802

March 6, 1972

Prepared for

NASA-GEORGE C. MARSHALL SPACE FLIGHT CENTER
Marshall Space Flight Center, Alabama 35812

TECHNICAL REPORT STANDARD TITLE PAGE

1. REPORT NO.	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.
NASA CR-61376		

4. TITLE AND SUBTITLE	5. REPORT DATE
Computer System Simulation and Analysis	March 6, 1972
	6. PERFORMING ORGANIZATION CODE

7. AUTHOR(S)	8. PERFORMING ORGANIZATION REPORT #
T. Williams and J.E. Weatherbee	

9. PERFORMING ORGANIZATION NAME AND ADDRESS	10. WORK UNIT NO.
Computer Sciences Corporation Field Services Division, Aerospace Systems Center 8300 S. Whitesburg Drive Huntsville, Alabama 35802	11. CONTRACT OR GRANT NO. NAS8-21805

12. SPONSORING AGENCY NAME AND ADDRESS	13. TYPE OF REPORT & PERIOD COVERED
National Aeronautics and Space Administration Washington, D.C. 20546	Contractor Report
	14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

16. ABSTRACT

A simulation model has been developed as a tool for studying different computer system configurations. The model simulates a computer system in terms of the number and speed of hardware subsystems and the runs processed by the system in terms of equivalent adds for computation and data bits transferred for I/O. The model was written in the GPSS simulation language.

17. KEY WORDS	18. DISTRIBUTION STATEMENT
Computer System Configuration Simulation Model System Throughput	Bobby Hodges B. Hodges Computer Systems Division MSFC/Computation Laboratory Unclassified-unlimited

19. SECURITY CLASSIF. (of this report)	20. SECURITY CLASSIF. (of this page)	21. NO. OF PAGES	22. PRICE
Unclassified	Unclassified	22	$3.00

MSFC - Form 3292 (May 1969)

TABLE OF CONTENTS

ACKNOWLEDGEMENT

The authors wish to acknowledge Dr. H. Kerner (S&E-COMP-C), who initiated the present MSFC program of computer systems evaluation through simulation which formed the basis for the work described in this report.

The Project Monitor, Mr. B. Hodges (S&E-COMP-C) is also acknowledged for his help and guidance throughout the course of the project and in the preparation of this report.

Finally, thanks is due to Dr. J. B. White (S&E-ASTR-CA) for the liaison provided with the SUMC project group and his contribution to the many discussions relating to the applicability of simulation to the SUMC project.

COMPUTER SYSTEM SIMULATION AND ANALYSIS

SUMMARY

This report describes a simulation model developed as a tool for use in studying different computer system configurations. The model was developed specifically for use in evaluating proposed configurations for the Space Ultrareliable Modular Computer (SUMC) being developed at MSFC Huntsville by the Computer Division of the Astrionics Laboratory.

Two major elements comprise the simulated environment; the system configuration being simulated and the workload being processed by the simulated system. The principal use of the simulator is in comparing the relative merits of two or more configurations by simulating these systems when processing a given workload.

A configuration is simulated in terms of the number and speed of the system central processors, the number of I/O data channels and the speeds of the I/O peripherals. The model also provides for the simulation of configurations with dedicated processors for non-user or system processing.

The workload processed by the simulated system is described in terms of the number of equivalent adds performed during computation and the number of data bits transferred while performing I/O.

The model is written in the GPSS simulation language and uses the statistics gathering features of that language to measure the performance of a configuration as it processes a particular workload.

The throughput performance and cost-effectiveness of two systems, one with and one without a dedicated Executive system processor, are evaluated as an example of how the model may be used.

INTRODUCTION

The advent of the third generation of computers has been accompanied by the realization that the choice of a computer configuration is no longer the relatively simple task it once was. The concept of multiprogramming allows the computing system to process more than one task concurrently, thus enabling more efficient utilization to be made of system resources such as central processing unit and input-output subsystems. A great amount of effort, resulting in many different approaches, has been expended on the problem of matching a modern computer system to its data processing workload. In order to accomplish such an optimization, however, detailed knowledge of both the proposed system and workload are required. The Systems and Computers Evaluation and Review Technique (SCERT)[1] provides the means for computer selection by making available a library of hardware and software performance factors for most of the commonly available commercial systems. The proposed data processing load must be described in detail so that a mathematical model of each program can be constructed. This technique is limited in that each Executive System considered is a fixed package and cannot be altered by the user.

Special purpose simulators have been written [2,3,4,5,6,7] which allow the user to select a hardware/software system for a prescribed data processing workload. These models are usually deterministic simulations of the algorithms of the Executive System which allow the user to evaluate various job scheduling philosophies and algorithms as constrained by a hardware configuration, which is also specified by the user. Such models depend, for their validity, upon the fidelity of the description of the workload. Various generalized job-mix descriptions have been used[3], as well as mixes more indicative of the particular application[4,7,8].

Computer systems have also been studied using mathematical descriptions of the system and its workload[9,10]; this method is not as flexible as that of simulation, however, due to the fact that an analytical solution quickly becomes intractable when the system is described to any great level of detail.

The bulk of the work reported so far in the literature is concerned with optimizing the performance of an established system. No work has been apparent to the authors, however, which addresses the problem of establishing the basic configuration of a new computing system.

In the preliminary specification of a computer system two requirements usually dominate: (a) the rate at which the computer processes data, i.e. its throughput rate, and (b) the system cost. A basic issue affecting both throughput and cost is the question of where the processing required by the executive system should be done. Two philosophies have emerged during the development of the computer to its present level of sophistication. The first of these, represented by the UNIVAC 1108 system, uses a central processor (or multiple processors) for

all computation to be performed by the system, ranging from complex calculations programmed by a user to the processing of relatively simple algorithms used by the executive program in supervising the processing of the total system load. The other philosophy utilizes the central processor for user programs only and provides a simpler peripheral processor to monitor the total system activity; this type of architecture is employed, for example, by Control Data Corp. in their 6000 and 7000 series machines.

Since it is not obvious how these two approaches compare in terms of throughput and cost, the present study was undertaken in order to develop a method whereby two such configurations could be compared in terms of their throughput, normalized with respect to their relative costs, when processing a pre-defined workload.

A simulation model was developed and consists of two major elements, (a) the configuration being simulated and (b) the workload being processed. A configuration is simulated in terms of the number and speed of the system central processors, the number of I/O data channels and the speeds of the I/O peripherals.

As an example of the application of the model, a study was performed in relation to the proposed earth-orbiting space station. Since design information is, as yet, not well defined, the results represent a first-cut at the basic system configuration for the Space Ultrareliable Modular Computer (SUMC). It proved possible to obtain gross estimates of the space station user and executive system overhead requirements in terms of average computation rates and average data transfer rates, but no information was available on processing or I/O transfer iteration rates, task memory requirements, the operating system task scheduling philosophy or peripheral storage subsystem specifications.

Based upon the information available, a workload is described in terms of the number of equivalent adds performed during computation and the number of data bits transferred while performing I/O. By varying the relative amounts of computation and data transferred during I/O, a spectrum of job-mixes was defined.

In processing these job-mixes on the different computer configurations, the criterion of a balanced system is established, i.e. one in which the queues for central processing unit and I/O channels are approximately matched. This condition ensures that system delays are not localized.

The performance of two systems, one with and the other without a special purpose processor (SPP) dedicated to performing executive functions only, is compared. The model is written in the GPSS simulation language.

SECTION II. SIMULATOR LOGIC

Figure 1 shows the logic flow as the simulated system processes a run. A run is created by a GPSS GENERATE block and is entered into the simulated system where it moves immediately to the run queue. In a typical computer system this event corresponds to the reading of a program run card.

The rate at which runs arrive for processing is determined by the particular application under investigation. If the run arrival pattern of the proposed workload is known, a function can be constructed which causes runs to be entered at this predetermined rate. If, however, the run arrival rate is unknown or if the purpose of a simulation is to determine the capability of the simulated system operating under a constant load, a fixed number of runs may be entered into the system at the beginning of the simulation (at simulated time zero). The remainder of the simulation is then devoted to processing as many of these runs as is necessary to evaluate the performance of the simulated system. The latter method is desirable when comparing two different systems processing the same workload, and was used in the work to be described in this report.

The rate at which runs leave the run queue and become active is determined by a parameter which limits the number of concurrent active runs. In a typical computer system the active time of a run corresponds to the time the run spends in core. Even though core acquisition and utilization are not modeled deterministically, placing a maximum on the number of concurrent active runs is an indirect way of specifying core capacity for the system. For example, if the average core requirement of a typical run is known to be 30K words of core memory and 6 runs are to be active concurrently, the system must have at least 180K words of memory.

When active, a run is in one of four possible states; in the CPU queue awaiting an available CPU, executing on a CPU, in a channel queue waiting for a channel to be free, or performing its I/O function. The duration of a run's active state is a function of the amount of computation and I/O performed by the run and the time spent in the CPU and channel queues. While the amount of computation and I/O performed by a run is independent of the configuration, the time required to perform these functions and the time spent in the queues are configuration dependent and are used in comparing the merits of different configurations.

Each time a run completes an I/O, a check is made to determine if the run has completed execution; if not, the run is directed to the CPU queue. When a run completes execution it is removed from the active state and terminated, allowing another run to become active.

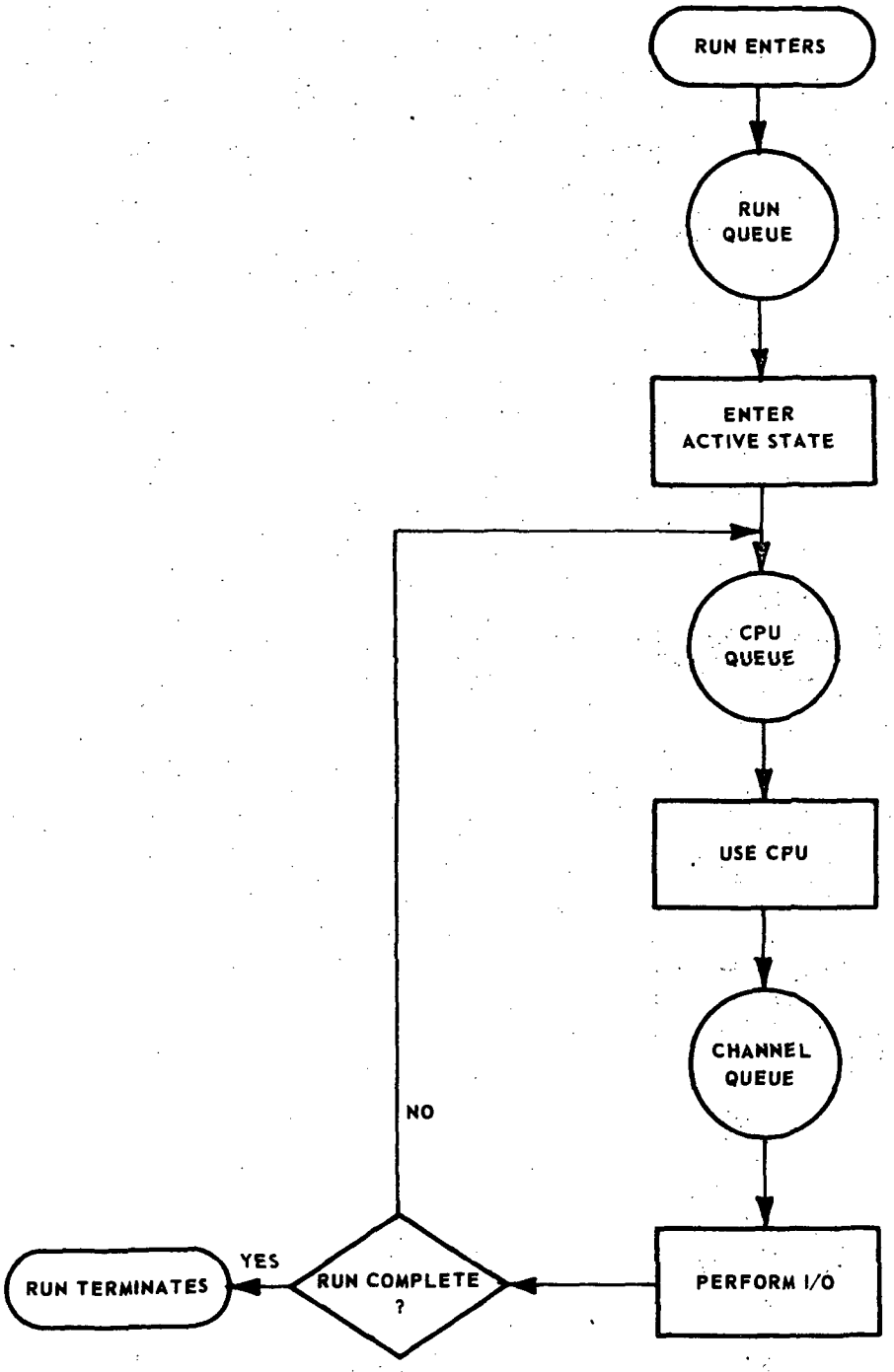

FIGURE 1. SIMULATOR LOGIC

SECTION III. THE SIMULATED ENVIRONMENT

The simulated environment is made up of the system being simulated and the workload being processed by the simulated system. Control over this environment is exercised through the various model input parameters.

A. The Simulated Workload

In defining the workload to be processed by the simulated system a run is defined in terms of the number of "equivalent adds" performed during computation and the number of bits of data transferred while performing I/O. This definition requires that all computation (adds, multiplies, divides, etc.) be expressed in an equivalent number of adds.

Each run is composed of a number of "compute-I/O loops" where each such loop consists of a number of equivalent adds (compute) followed by the transfer of a number of data bits (I/O). The values of these run attributes, as well as the action of the simulated system in processing the runs, are controlled through the use of four probability distribution functions. Three of these functions (PL, PC, and PI) are Poisson distributions with means specified by model input; the fourth (PCH) is a function defined by input data which gives the distribution of I/O's among the various data channels. These functions are sampled in the following manner during execution of the model (see Figure 2):

PL: Sampled once for each simulated run to give the number of compute-I/O loops for the run.

PC: Sampled for each compute-I/O loop of each simulated run to give the number of equivalent adds to be executed during the loop.

PI: Sampled for each compute-I/O loop of each simulated run to give the number of data bits to be transferred during the loop to a channel sampled from PCH.

PCH: Sampled for each compute-I/O loop of each simulated run to select a channel for data transfer.

The CPU time and I/O time required during each compute-I/O loop are calculated using the values obtained from PC and PI and the speeds of the system hardware. Note that the amount of work to be performed during a compute-I/O loop is configuration independent while the time required to perform the work is configuration dependent. Thus a particular job-mix can be run on different configurations to find the configuration which processes the job-mix in the most efficient or satisfactory manner.

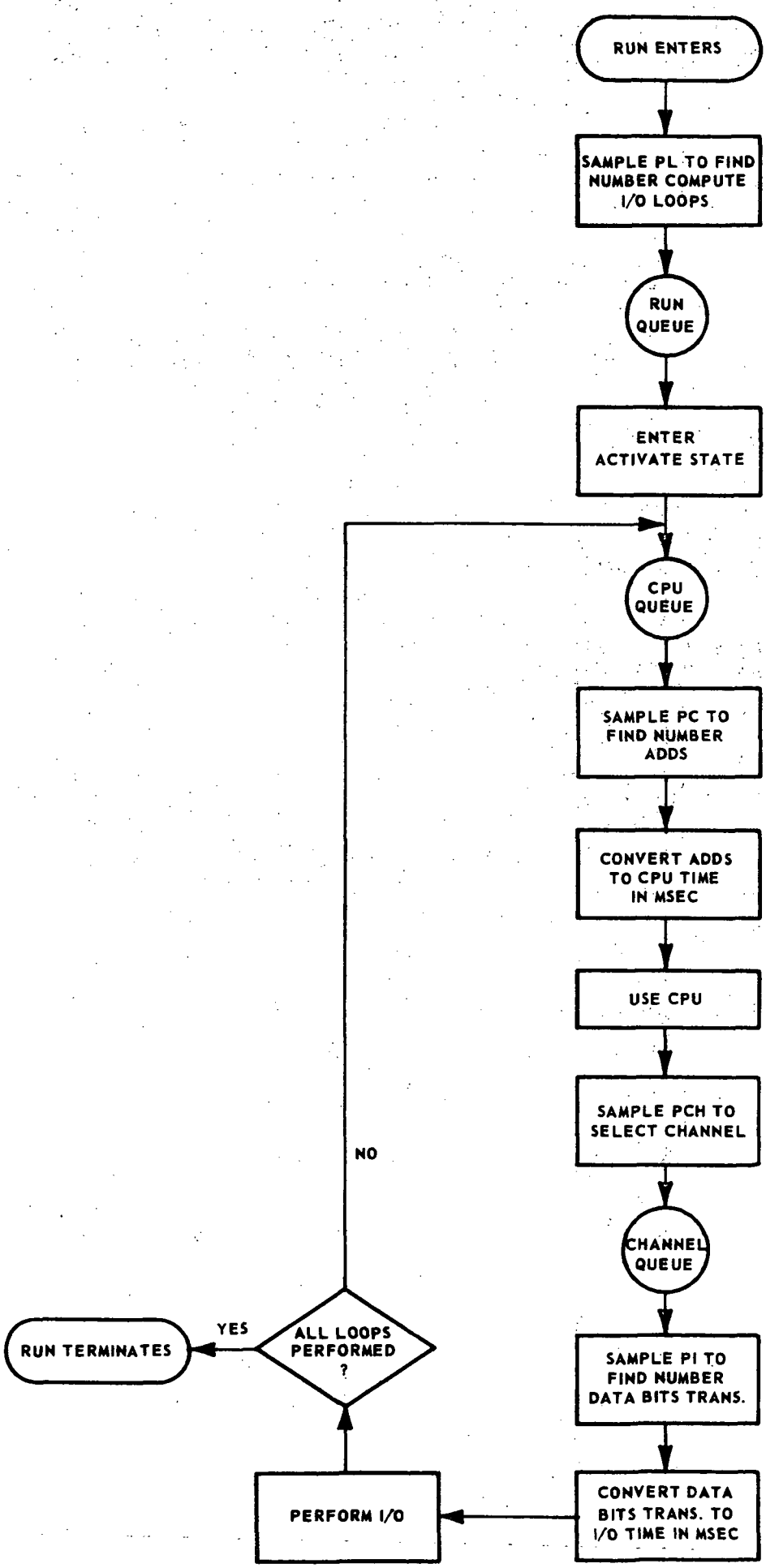

FIGURE 2. USE OF PROBABILITY DISTRIBUTION FUNCTIONS

B. The Simulated System

The hardware components of the configuration being simulated are specified in terms of the number and speed of each such component. CPU speed is measured in microseconds per add and I/O subsystem speed is a function of average access time (in milliseconds) and data transfer rate (in bits per millisecond).

An option is provided which includes in the system a specified number of special purpose processors to be used exclusively for executive system processing. The speed of these processors is independent of the CPU speed.

Since an executive system has yet to be developed for SUMC, certain simplifying assumptions were required in the development of scheduling algorithms for the model and in placing a load on the simulated system corresponding to the processing requirements of an executive system.

The following scheduling procedures were implemented in the model:

1. All runs have the same priority.
2. All system resources (CPU's, channels, etc.) are allocated on a first-come first-served basis.
3. All CPU's are assumed to be identical so that a single queue is required for runs awaiting an available CPU.
4. A queue is maintained for each data channel since each I/O is directed to a specific channel.
5. Once a run has control of a CPU it will retain control until I/O is required.
6. When a run performs I/O it loses control of the CPU on which it was executing.

These procedures provide the scheduling structure necessary to simulate SUMC configurations without imposing assumptions detrimental to a particular type of configuration.

In order to represent the requirements of the executive system during the course of processing a given stream of user runs, one executive task is arbitrarily created for each active user run. These executive tasks are executed in the same way as user runs with the following exceptions; the mean number of equivalent adds performed and the mean

number of data bits transferred, per compute-I/O loop, are fractions of the corresponding means for user runs. Thus executive processing requirements are expressed as fractions of user processing requirements and may be varied to represent various executive-to-user processing ratios.

SECTION IV. APPLICATION OF THE MODEL

The first application of the model involved a study of two distinctly different configuration philosophies. One of these philosophies is best illustrated by the UNIVAC 1108 configurations in which from one to three identical CPU's are used in conjunction with one or two input/output controllers (IOC'c), which control peripheral subsystems by providing a direct data path between main storage and the peripheral subsystems.

A second philosophy is illustrated by the CDC 6000 series configurations which utilize a single powerful CPU in conjunction with a number of less sophisticated peripheral processing units (PPU's). In addition to the PPU's which function like the UNIVAC IOC's, one is designated system monitor and handles all the executive system processing; thus the CPU is reserved for user processing only.

This study does not make an in-depth analysis of these two philosophies, but presents an approach for examining the cost-effectiveness of SUMC configurations with and without Executive Controllers (EXCN's) for all executive system processing. The simulator is used to determine the relative effectiveness of each configuration in terms of system throughput.

A. Preliminary Considerations

In studying different possible configurations, there is no SUMC performance criterion against which simulator results can be compared in evaluating a particular configuration. Thus the principal use of the model is in comparing the relative performance of two or more possible configurations. In an effort to make these comparisons more meaningful, the following three concepts were defined for the study under consideration; baseline job-mix, minimum configuration, and balanced configuration.

Before developing workloads upon which to exercise the various simulated configurations, a baseline job-mix was established to use as a standard. Since SUMC is a candidate computer for space station data processing, estimates of the space station data processing requirements (see Table 1) were used to formulate the baseline job-mix.

TABLE 1: SPACE STATION PROCESSING ESTIMATES

	Average Data Transfer Rate (Bits/Sec.)	Average Computation Rate (Operations/Sec.)
User Requirements	11.5×10^6	700×10^3
Executive Requirements	60×10^3	657×10^3

The minimum configuration, which possessed some features required of SUMC, was minimal in the sense that all other configurations considered in the study were derived by upgrading this configuration. Since SUMC must have the capability of operating in Triple Modular Redundency (TMR) mode, the minimum configuration contained three identical CPU's. These CPU's operated at a rate of two microseconds per add, a design specification of SUMC. No criteria have been established with regard to SUMC I/O subsystems, so the minimum configuration was arbitrarily given nine data channels together with I/O subsystems having a common data transfer rate of 4000 bits per millisecond.

The concept of a balanced configuration was derived to provide a criterion for identifying bottlenecks at the CPU's or I/O subsystems within a configuration. If CQ is the average time spent in the CPU queue and IQ is the average time spent in a channel queue, per compute-I/O loop, then the configuration being simulated is considered to be balanced with respect to the job-mix it is processing if $1/2 \leq CQ/IQ \leq 2$.

B. Study Procedures

Figure 3 illustrates the procedures used in the study. Five job-mixes were defined (See Table 2) and for each of these job-mixes, balanced configurations were determined with and without SPP's. Each configuration was then exercised against each job-mix to provide satistics for making the following evaluations:

1. Relative performance when processing a particular job-mix.

2. Relative performance across job-mix spectrum.

3. Relative cost.

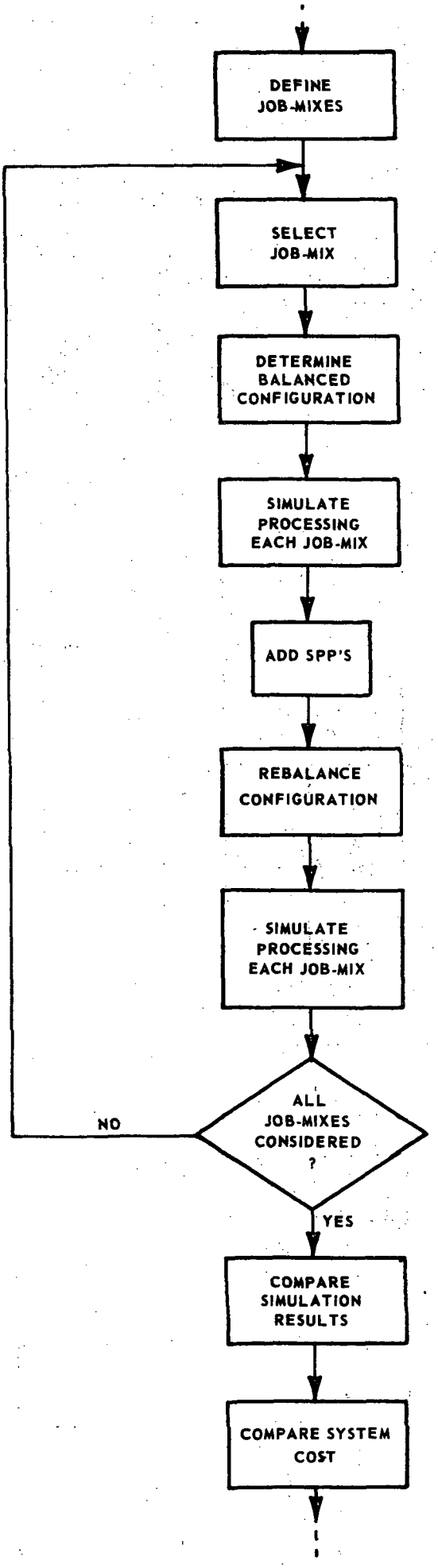

FIGURE 3. STUDY PROCEDURES

Table 2 shows a spectrum of job-mixes derived from the baseline mix of Table 1. The parameters are translated from "average data transfer rate" and "average compution rate" of Table 1 to "mean number of bits transferred per compute-I/O loop" and "mean number of adds per compute-I/O loop" in Table 2. The characteristic compute: I/O ratio of the user requirements of the baseline mix of Table 1 is duplicated in job-mix 3 in Table 2. The lower numbered mixes are more compute-bound and higher numbered more I/O bound than the baseline.

TABLE 2: JOB-MIX DESCRIPTIONS

Job-Mix	Mean No. Bits Trans. per Compute-I/O Loop	Mean Number Adds per Compute-I/O Loop
1	200,000	24,000
2	224,000	21,000
3	264,000	16,000
4	288,000	13,000
5	316,000	10,000

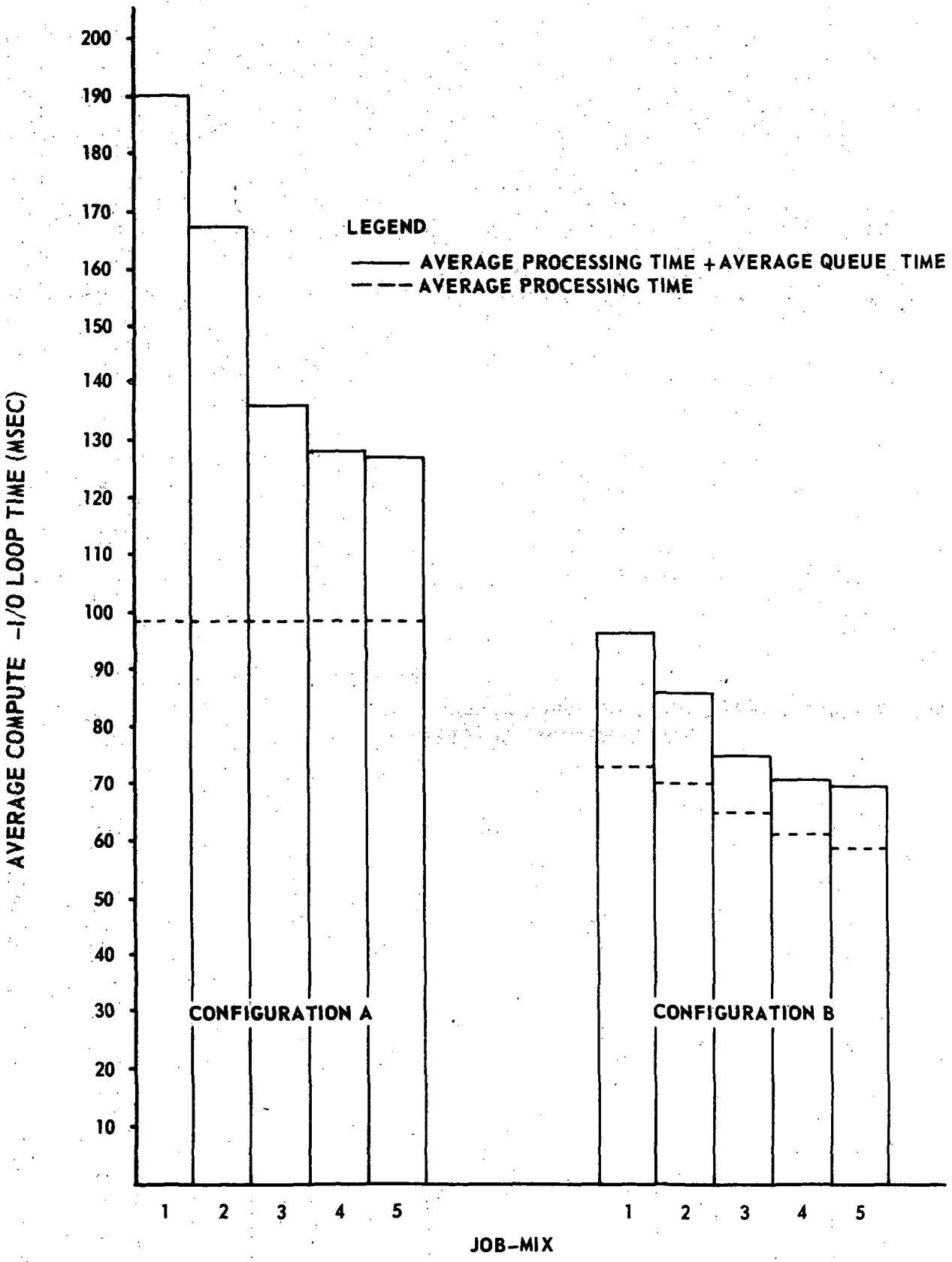

FIGURE 4. COMPARISON OF TWO CONFIGURATIONS

1. Performance Analysis. Figure 4 illustrates a type of performance analysis possible with the simulation results. Two statistics are represented in each histogram: Average compute and I/O processing time required per compute-I/O loop (dashed line), and average total time (compute and I/O processing times plus CPU and I/O queue times) per compute-I/O loop, represented by the solid line.

Configuration A is the minimum configuration and is balanced for job-mix 3, the baseline job-mix. Configuration B was obtained by adding three SPP's to Configuration A thereby removing all executive system overhead from the CPU's and rebalancing for job-mix 3; this was achieved by doubling the data transfer rate of the I/O devices.

In Configuration A, the ratio of compute to I/O times were chosen so that the average total compute and I/O times were constant across the spectrum of job-mixes. The worst queueing situation occurs for the most compute bound mix (job-mix 1) as can be seen from Figure 4 and Table 3, where the queue serving the three CPU's represents a severe bottleneck.

This condition is alleviated, however, in Configuration B, where the CPU queueing time is reduced by a factor of four (see Table 3) due to the removal of the non-user tasks from the CPU's to the SPP's. Table 3 also shows that the I/O queueing times are reduced for all job-mixes because of the faster peripheral devices. Additionally, Configuration B exhibits a lower average processing time in all cases: this again is due to the faster peripheral devices since the CPU speed was held constant for both configurations.

It is of interest to note that the variation in average processing time per compute-I/O loop was more nearly constant in Configuration B, varying by 36.5% across the job-mix spectrum. This compares with a variation of 48.1% for Configuration A. Hence, for the situation considered where the processing load was known in only a gross form (see Table 1) Configuration B has the advantage that its throughput is less sensitive to changes in the nature of the processing load.

2. System Cost. A further factor which bears on the desirability of a particular configuration is the cost of the system. For example, configuration A processed job-mix 3 at a rate of 136 milliseconds per compute-I/O loop whereas configuration B required only 75 milliseconds per loop, a reduction of forty-five (45%) percent. However, configuration B is a more expensive system than configuration A (three additional SPP's and I/O devices twice as fast). In evaluating the two configurations this difference in cost must be taken into account along with the difference in processing capabilities. One way to assess these two factors in a common expression is to find a value for "Throughput per Dollar" for each configuration. The following expression could be used to find Throughput per Dollar:

$$\text{Throughput/Dollar} = \frac{1/(\text{Average Loop Time})}{\text{System Cost}}$$

where,

$$\text{System Cost} = (\text{No. CPU's})*(\text{CPU Cost})+(\text{No. SPP's})*(\text{SPP Cost})+$$
$$(\text{No. Channels})*(\text{I/O Device Cost})$$

Of course this assessment is possible only for a definite configuration where reasonable cost estimates are available for the system components.

SECTION V. CONCLUSIONS

A simple model has been derived to enable a study of a basic computer configuration to be made. Two configurations were examined, one with and the other without special purpose processors dedicated to the processing of system tasks. It is concluded that for the space station workload description considered, that the system having the additional dedicated processors is the more "broadly tuned" in that its throughput rate is less sensitive to the compute-I/O characteristic of the job-mix being processed.

A figure of merit was derived based upon the throughput per dollar ratio for both systems.

REFERENCES

(1) D.J. Herman and F.C. Ihrer, "The use of a computer to evaluate computers", 1964 Spring Joint Computer Conference, AFIPS Proc. Vol. 24, pp 383-395.

(2) N.R. Nielsen, "An approach to the simulation of a time-sharing system", 1967 Fall Joint Computer Conference, AFIPS Proc. Vol. 31, pp 419-428.

(3) G.R. Cassir and A.G. Abrahart, "The design and use of a general purpose real-time system simulator", The Computer Bulletin, pp 149-152, August 1968.

(4) F.C. Holland and R.A. Merikallio, "Simulation of a multiprocessing system using GPSS", IEEE Trans. on Systems Science and Cybernetics, Vol. SSC-4, pp 395-400, November 1968.

(5) N.R. Nielsen, "The simulation of time-sharing systems", Comm. ACM, Vol. 10, pp 397-412, July 1967.

(6) R.A. Merikallio and F.C. Holland, "Simulation design of a multi-processing system", 1968 Fall Joint Computer Conference, AFIPS Proc. Vol. 33, pp 1399-1410.

(7) J.H. Katz, "An experimental model of system/360", Comm. ACM, Vol. 10, pp 694-702, November 1967.

(8) G.K. Hutchinson and J.N. Maguire, "Computer systems design and analysis through simulation" 1965 Fall Joint Computer Conference, AFIPS Proc. Vol. 27, pp 161-167.

(9) H. Hellerman and H.J. Smith, Jr., "Throughput analysis of some idealized input, output and compute overlap configurations", Computing Surveys, Vol. 2, pp 111-118, June 1970.

(10) D.P. Gaver, Jr., "Probability models for multiprogramming computer systems", Journal ACM, Vol. 14, pp 423-438, July 1967.