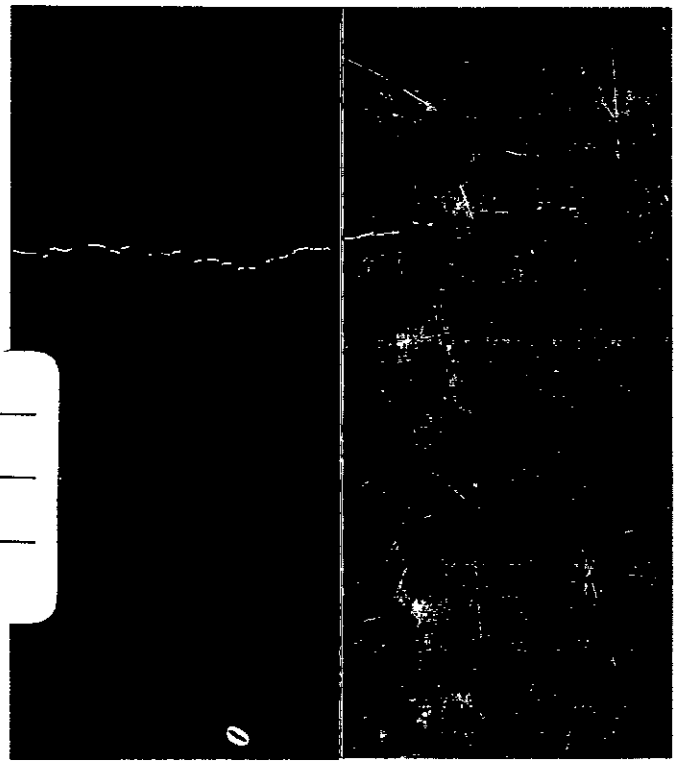
2(BIX)

SPERRY RAND



SPACE SUPPORT
DIVISION

Report No.: SP-209-0339-3

Date: November 4, 1970

GENERAL PURPOSE SIMULATOR SYSTEM STUDY

Prepared For
COMPUTATION LABORATORY
of
GEORGE C. MARSHALL SPACE FLIGHT CENTER
CONTRACT NO. NAS8-25608

By
SPACE SUPPORT DIVISION
DIVISION OF SPERRY RAND CORPORATION
HUNTSVILLE, ALABAMA

SUBMITTED BY: J. Scoggins
J. Scoggins

and

J. Taras
J. Taras

APPROVED BY: R. A. Rossler
R. A. Rossler

TABLE OF CONTENTS

GENERAL PURPOSE SIMULATION SYSTEM STUDY PART 1 (CONTINUED)

GENERAL PURPOSE SIMULATION SYSTEM STUDY PART 3 (CONTINUED)

ACKNOWLEDGEMENTS

The authors wish to express their sincere appreciation to their colleagues at Computer Sciences Corporation for their significant contribution to this project, particularly to P. A. Brown and C. O. Rigby for the development of the "Response and Equation Evaluation Definition", and for their assistance in reformating the vehicle equations for the proposed shuttle system configurations.

GENERAL INTRODUCTION

Contained herein is a report of a study performed for the Computation Laboratory of the Marshall Space Flight Center. The study is an evaluation of the DEE-6 Simulator System to determine how it could be modified for shuttle applications.

Assistance in the analyses of the software system was rendered by personnel of the Computer sciences Corporation as part of their activities in support of the Computation Laboratory. This assistance is gratefully acknowledged.

The simulation was originally performed using an SDS-930 computer as a subsystem simulator with two RCA-110A computers configured as a Mobile Launch Facility and Launch Control Center. The processors and associated interface equipment were provided by the Astrionics Laboratory and software support was provided by the Computation Laboratory. The joint effort provided the simulator with a sophisticated software package capable of scheduling for evaluation discrete and analog equations representing the electrical and mechanical subsystems of the Saturn V vehicle.

Sperry Rand reports parts 1 through 3 are detailed studies of the simulation hardware and software performed through the first 3 of the 4 quarterly contract periods. A critical analysis of the simulator was performed and recommendations for improving the simulator are included in the summaries of each of the reports.

The report for the fourth quarter has been placed first in the document for reader convenience and because it contains the hardware and software modifications that were proposed as the result of the studies of the first three quarters. The reader is invited to review the system proposals in Appendices A and B of report four because these hardware configurations along with the software package are capable of simulating multiple shuttle subsystems.

The software structure has been altered to improve equation response and to reduce equation length. The executive program will still be capable of solving the ten distinct types of analog equations as defined by the Transfer Equation Preparation Manual prepared by Sperry Rand. In addition, the software will be capable of solving multiple discrete signals with cross-coupling between discrete and analog equations.

The newly developed simulation system will be called the Marshall Real Time Digital Simulator (MARDIGS). The proposed simulator will be capable of responding to the ground checkout computer and data management system by means of a data bus for flight program verification. The simulator consists of a sophisticated executive program capable of scheduling and solving discrete and analog equations representing the response of a shuttle subsystem. The hardware is configured so that parallel or multiprocessing techniques are used to meet the response required of the subsystems. The concept of using a digital computer for subsystem simulation provides an inexpensive means of testing flight programs without the investment for flight hardware.

SUMMARY

Several years ago the Marshall Space Flight Center developed and demonstrated a Saturn V Real-Time Launch Vehicle and Ground Support Equipment Software Simulator. The simulator performs the functions of the Saturn V vehicles by scheduling for evaluation in a control sequence time-dependent logical equations for discrete functions and polynomials or tables for the analog functions. The intent of the study was to investigate how this simulator can be optimally used for new programs such as Space Shuttle and Space Station and to identify modifications necessary for operation with these new programs.

The repertoire of analog equations available for simulating analog functions make the MARDIGS language especially attractive for shuttle subsystem simulation. The equations themselves may be written in segments to define an analog variable during different phases of operation (Boost, Orbit, Docking, Reentry, Approach and Landing Phases). In addition the frequency of solution of the analog equations can be adjusted in each of the phases.

The cross coupling of the analog and discrete equations make the language especially attractive for simulation and as proposed in Appendices A and B of the final report, are capable of simulating multiple systems. If necessary, the single data bus interface may be fanned out to represent several interfaces with the single processor communicating with each of the interface units.

A number of improvements have been proposed to upgrade the digital system to an effective simulator. The most effective modifications are:

1. Making all equations resident in core memory. This single item provides the most effective means of improving the response of the simulator. The necessity of accessing the equations from the disk was eliminated and on subsequent solution of equations eliminates the need for searching through buffers for the ID of the equation.

A computer program requesting the solution of several analog equations was run on the UNIVAC 1108 computer and an equation solution improvement factor of up to 50 was obtained. For the comparison of results, see sections 3.7 and 3.8 of report 3 and appendix C of the final report.

2. Providing a separate processor(s) for the solution of analog equations. The parallel processing of analog equations improves the response considerably since the additional processor(s) solves only analog equations.

3. Providing external clocks and eliminating cycle stealing by the processor for updating internal memory locations.

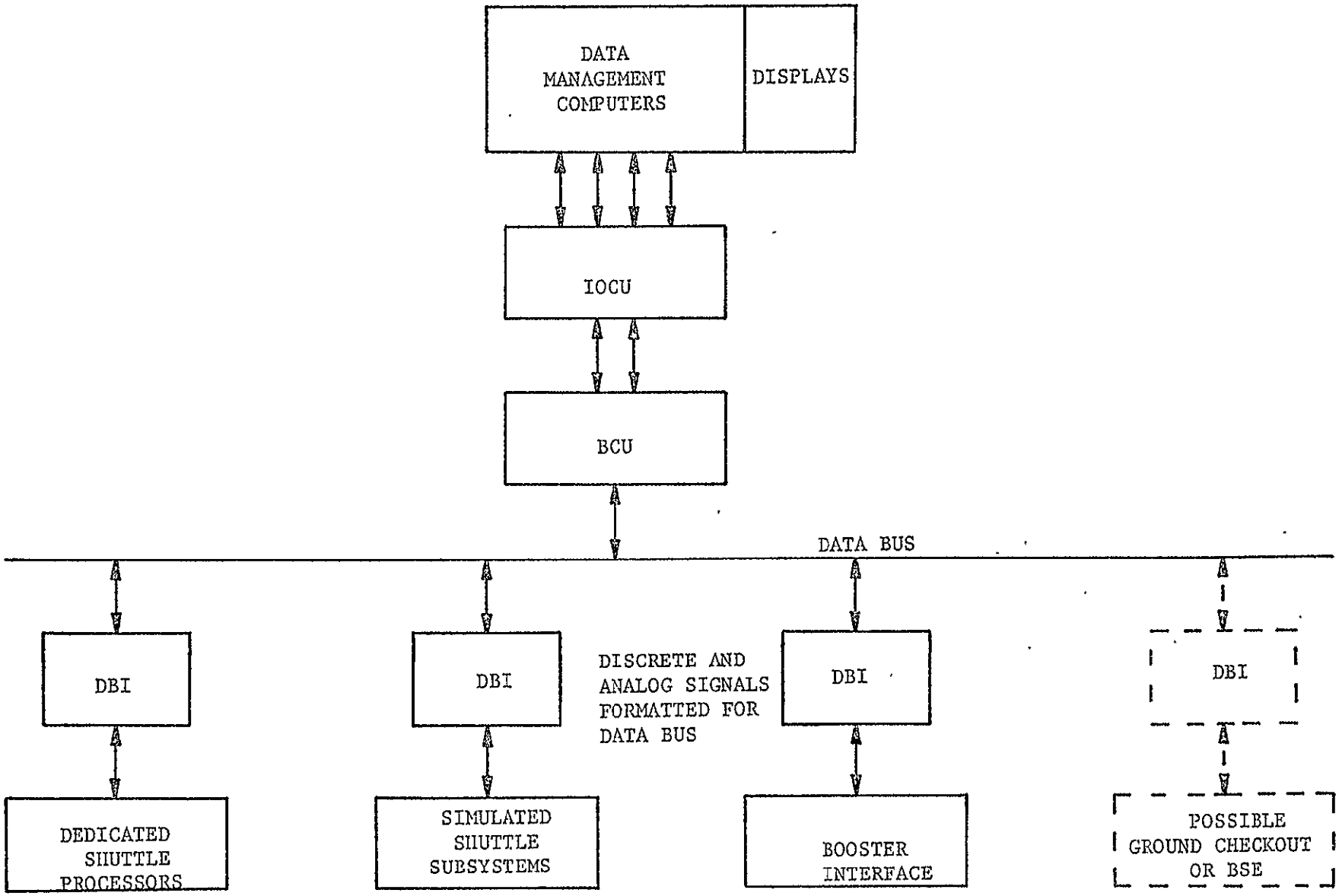
4. Developing a new equation format for all equations. This format was established in Report 2 Appendix B (CSC entry) and was found to reduce the length of the equations.

5. Providing hardware floating point operations to eliminate time consuming software subroutines.

6. Providing a single boolean status variable for each analog segment of an equation (Final Report, Appendices A and B). This single status variable will reside in shared memory and will improve the analog computational capability of the simulator considerably.

7. Making selectable the variables to be recorded on magnetic tape. Eliminating variables not normally monitored improves the system response considerably.

The improvements made and the hardware configurations proposed in the final report provide a hardware complex with a sophisticated executive program capable of simulating multiple subsystems on the shuttle data bus. As indicated in figure 1 the subsystem simulator is capable of accepting and sending discrete signals formatted for data bus transmission. In addition, the analog equations are solved within a millisecond time frame providing analog signals of proper response time for shuttle simulation.


DATA MANAGEMENT COMPUTERS
DISPLAYS
IOCU
BCU
DATA BUS
DBI
DBI
DISCRETE AND ANALOG SIGNALS FORMATTED FOR DATA BUS
DBI
DBI
DEDICATED SHUTTLE PROCESSORS
SIMULATED SHUTTLE SUBSYSTEMS
BOOSTER INTERFACE
POSSIBLE GROUND CHECKOUT OR BSE

FIGURE

GENERAL PURPOSE SIMULATOR
SYSTEM STUDY
FINAL REPORT

Report No: SP-209-0339-3
Date: October 31, 1970

General Purpose Simulator System Study Final Report

Prepared for

George C. Marshall
Space Flight Center
Contract No. NAS8-25608

By

Space Support Division
Division of Sperry Rand Corporation
Huntsville, Alabama

Submitted by J. Scoggins
J. Scoggins

and J. Taras
J. Taras

Approved by R. A. Rossler
R. A. Rossler

ABSTRACT

This the fourth and final report of contract NAS8-25608 describes the philosophy behind the Shuttle Data Management System and describes schemes of using the MARDSLC simulation language for subsystem simulation and checkout.

TABLE OF CONTENTS

Introduction

A simulation language has been developed and used at the Marshall Space Flight Center to simulate space vehicle subsystems. The simulation language called MARDIGS has the ability of scheduling for solution, discrete and analog equations representing the electrical and mechanical characteristics of the space vehicle's subsystem.

The language has been successfully demonstrated on the Instrument Unit (IU) of Saturn V vehicle and the switch panel of ATM. The ability of the language to schedule equations for solution makes the languate especially attractive for Space Shuttle subsystem simulation. The nature of the Space Shuttle launch and recovery dictates that the subsystems sample the various sensors at different sample rates for each of the major modes of Shuttle operation. For this reason it becomes very important that the simulation system schedule for solution the various equations representing the characteristics of the Shuttle subsystems.

The MARDIGS simulation language has been proven capable of simulating the electrical and mechanical subsystems of the Saturn V vehicle. The first three reports of this contract delt with the capability and limitations of the existing software and hardware. Suggestions and recommendations were made for the improvements of both. This, the fourth and final report describes the philosophy behind the Space Shuttle Data Management System and provides various schemes for simulating the major subsystems aboard the Booster and Orbiter.

Shuttle System Philosophy

In past manned spaceflight programs, a great deal of expense has been attributed to launch test complexes and operations support personnel. In addition a great deal of time was consumed for a series of pre-launch test activities. Since the Shuttle is a reusable vehicle it is most important that all preflight testing be accomplished on-board the orbiter to reduce the cost of vehicle checkout and minimize test time. In addition post-flight maintenance activity can be expedited and simplified by making the in-flight on-board capability effecient enough for fault isolation to line replaceable units.

The requirements of the Space Shuttle Orbiter and Booster are such that both vehicles must be capable of autonomous operation during all of the mission phases and operating modes. This capability will reduce to a minimum the amount of ground support needed for in-flight manuevers. Such a scheme would also reduce the amount of Ground Support Equipment and support personnel required for launch.

While an autonomous mode of operation would reduce the amount of ground support needed for a mission, the scheme would require a sophisticated Inertial Navigation System and an elaborate means of checkout and fault isolation.

A Strapdown Inertial Reference Unit (SIRU) with a dodecahedron mounting of sensors appears to be the most promising of the inertial navigation configurations. This system, developed by the Massachusetts Institute of Technology provides a major increase in reliability, making the concept especially attractive for long-duration space missions. The concept also requires a third fewer gyros and accelerometers. In addition all of the sensors are identical and readily accessible and easily replaced This inertial scheme should satisfy all the requirements of the Shuttle mission and conforms with the Line Replaceable Unit scheme (LRU) proposed by Shuttle contractors [2]. Naturally a new inertial concept such as the SIRU will require an enormous amount of testing before being accepted for the Shuttle mission.

· Integrated Avionics System (IAS)

The Integrated Avionics System (IAS) concept proposed for Shuttle provides maximum reusability of the space vehicle with a minimum of turnaround time. In addition the IAS scheme provides for on-board systems checkout and assures success for Astronaut initiated missions.

An autonomous system such as that proposed for Shuttle requires a sophisticated processor arrangement with an extremely strong and flexibile man/computer interface capability.

Without ground support, the crew must be able to monitor and analyze with ease, information pertinent to a successful mission. Critical data and trend analysis must be available to the Astronauts from each of the eight major subsystems. Present studies indicate that the best method of providing mission critical data to the crew is via CRT displays [1], [2]. Elaborate schemes of CRT displays and display control computers have been proposed to meet the display requirements of the following Shuttle modes of operation;

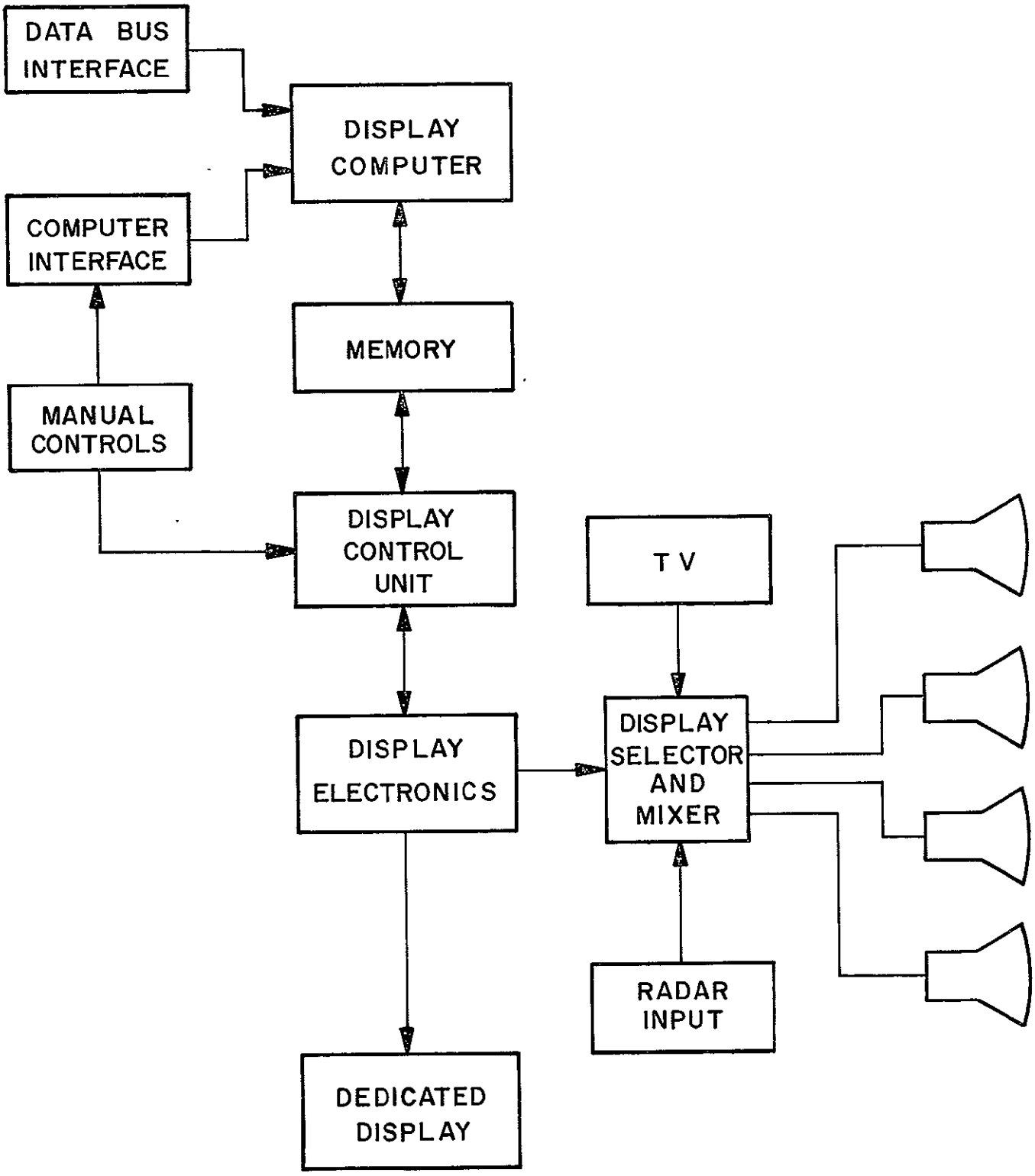
1. Boost Phase
2. Orbit Phase
3. Docking Phase
4. Reentry Phase
5. Approach Phase
6. Landing Phase

The proposed crew stations allow Shuttle flight control by one crew member with information available to both crew members during periods of critical activity. During non-critical maneuvers the redundant displays may be used to perform subsystem checkout, malfunction analysis, configuration display or general housekeeping activities. With the displays being under computer control (with Astronaut intervention) the entire display complement may be programmed to display different information during various modes of shuttle flight. Critical events can be programmed to take precedence over previously displayed variables.

The types of displays (CRT, Plasma, Electroluminescent, etc.) and the number of displays to be used on the orbiter has not yet been established. A great deal of research is presently being done to establish the types of displays to be implemented and the amount of data compression required for display purposes.

Considerations have been given to using a computer-managed display (Figure 1 and Figure 2). Such a configuration would eliminate the requirement for redundant electronic assemblies for each display system by combining the functions in a computer-managed unit. Such a device would reduce the weight and power requirements considerably. For a more complete description of the computer-managed display system see Section 7.0 of Reference 1.

As seen in Figures 1 and 2 the data to be displayed may be obtained from the "Data Bus" or from the central computer. The variables to be plotted will be specified automatically by the central computer depending on Shuttle mode of operation or will be requested by the crew through a control panel or a keyboard.



COMPUTER MANAGED DISPLAY

FIGURE I.

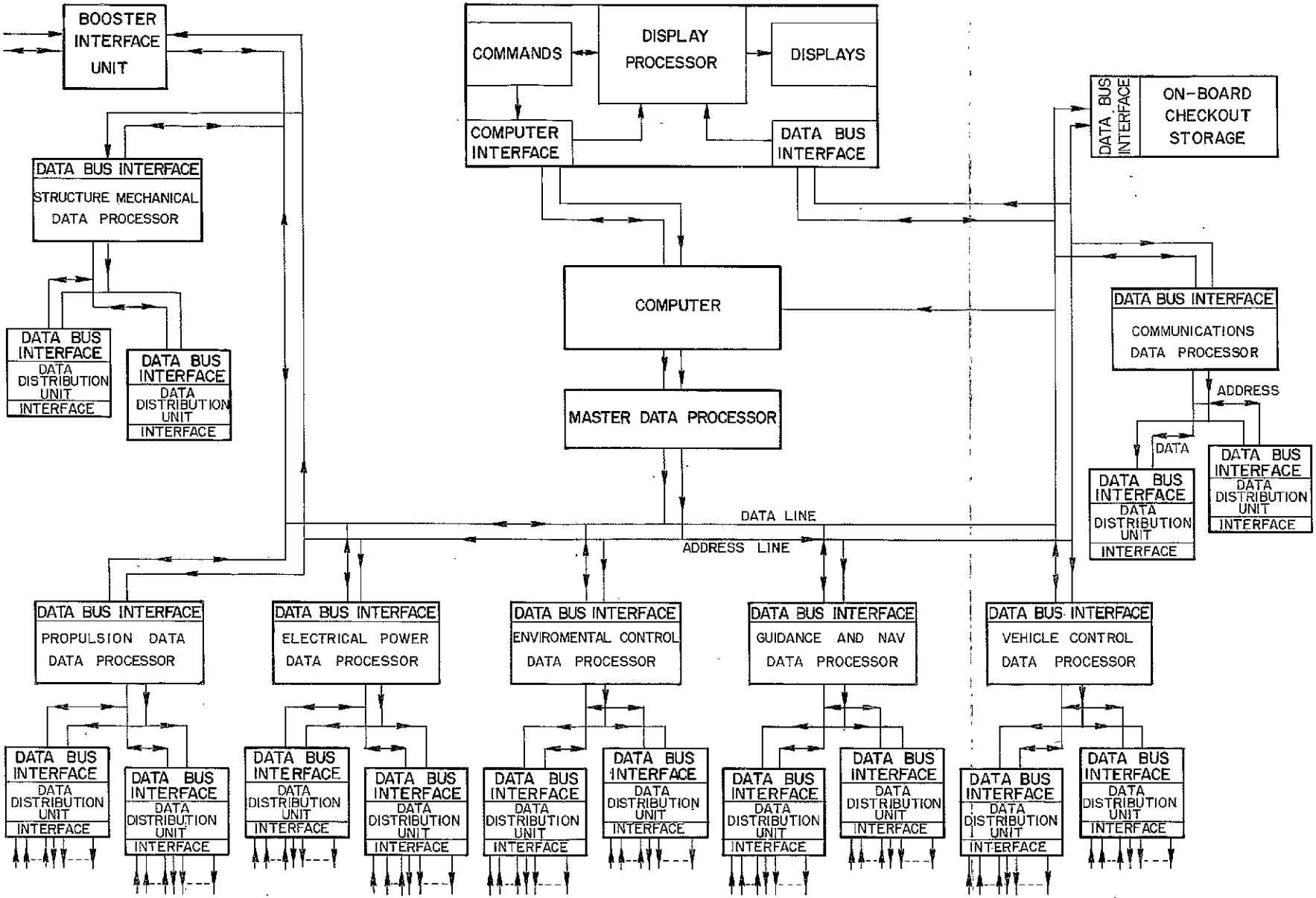


FIGURE 2

FOLDOUT FRAME

FOLDOUT FRAME

2

Subsystem Checkout

Since the checkout system of the Shuttle will be flown, human engineering becomes a prime factor in design. Obviously a large portion of the checkout will be automatic and will be performed within the subsystems. Continuous or periodic tests will be performed within the subsystems with the necessary electronics to allow monitoring within each LRU. Should the measurements not meet the preset tolerances a fault will be indicated within the cockpit.

Built-In-Test

The decentralized Built-In-Test (BIT) concept eliminates the need for an enormous amount of on-board data collection and analysis [2]. The proposed scheme (BIT) would eliminate for the most part the sampling of sensors by the central computer complex and free the computer for a more detailed analysis of the faulty system. Supplemental tests would be performed upon crew request and the results of the tests would be displayed for convenience. The detailed status analysis would allow an inflight decision on the best method to proceed. The crew would have the option of switching in a backup system or proceeding with the present degraded system.

In general the proposed Built-In-Test (BIT) would permit crew controlled launch capability, increase the probability of mission success, and allow a rapid turn around capability. As proposed [2], the on-board checkout system would provide the following characteristics:

- Automatic continuous subsystem monitoring
- Crew initiated supplemental tests
- All failure data available for crew display
- Permanent record of malfunctions
- Capability for monitoring trend data
- Monitor all vehicle subsystems
- Full in-flight testing ability.

As indicated in the preceeding paragraphs the display system will play a major role in all phases of the Shuttle mission. The memory requirement of such a sophisticated display system would be enormous. Since the individual subsystems require independent test programs for detailed analysis, all of the diagnostic tests will be contained in bulk memory. It will be necessary for the Checkout, Monitor, and Fault Isolation Computer (CMFI) to fetch and execute the test program selected.

Past studies of manned flight vehicles indicate a need for improved man/computer interfaces. The checkout scheme proposed for Shuttle will require a major improvement in display technology and the interaction of man with the display computers.

The most promising method of man/computer interface for display checkout is by "Reprogrammable Switches" supplemented by a keyboard console. This method if implemented would eliminate the cluttering of switch panels and would reduce to a minimum the amount of crew interaction.

Due to the limited response by man, the amount of interaction required for test purposes must be limited. In the "Reprogrammable Switch" concept the crew members would be permitted to select tests via designated switches. Reprogramming of the switches and cueing the operator would permit optional tests on the same subsystem switches.

Postflight Maintenance

Preliminary studies by Shuttle contractors indicate that an external checkout scheme will be needed for postflight fault isolation. While a great deal of subsystem fault isolation may be accomplished by on-board computers, the depth of fault isolation will be limited to detecting failures for which the crew may correct.

Several factors have a bearing on the extent of fault isolation that may be accomplished on-board. The amount of instrumentation required for checkout would be limited due to the added weight and the degradation to the subsystem itself. The amount of instrumentation allowable would in most cases be limited to detecting a failure to a subsystem component and not to the individual part within the component. In general the fault detection scheme would isolate faults within a subsystem to the lowest level unit that can be conveniently removed from the vehicle.

The three fundamental considerations that must be given in designing a checkout system for Shuttle are:

a. Probability of crew and vehicle survivability.

b. Probability of a successful mission.

c. Cost of total program.

To accomplish the above described tasks there must be a great deal of trade-offs between methods of checkout. To insure a successful and safe mission at a reasonable program cost requires a great deal of development in automatic checkout systems. In addition to relieving the crew of elaborate subsystem analysis, there is the problem of fetching large test programs from a bulk memory device and executing the test procedures. Storage limitations would reduce the number of test routines allowable for subsystem checkout. The test routines would probably be limited in depth due to required crew analysis; necessitating the use of a ground checkout computer for detailed postflight subsystem fault isolation.

Data Bus Concept

The Intergrated Avionics System (IAS) proposed for Shuttle involves a large amount of control and data transmission between the computer complex and subsystems. The placement of the subsystems and sensors relative to the computers requires a method of multiplexing data to eliminate massive cable harnesses. Studies to date indicate a substantial weight savings in electronics when multiplexed data transmission schemes are implemented.

Of the two possible schemes of multiplexing (time and frequency), time division multiplexing was chosen as the most appropriate mode of multiplexing data and commands for the Data Bus. The message in itself will be transmitted via a shielded, twisted-pair cable. The selection of this medium of transmission came as a result of extensive study by NASA and contractor personnel.

Of nine available signal types available for Data Bus transmission seven were eliminated due to poor transmission characteristics. The remaining signal types available (Bi-Phase and Level Shift Keying) were chosen as the most suitable for Shuttle. Figure 3 shows both of the waveforms considered. While Level Shift Keying with sine wave modulation (LSK-SW) appears to be the most suitable for Shuttle, the final selection of coding methods has not been made.

Independent studies on coupling methods indicate that transformer coupling rather than direct coupling will be used. This is to be expected since this method of transmission provides complete ground isolation and reduces the transients inherent in direct coupled transmission lines. The sine wave modulation techniques chosen as the mode of transmission simplifies the task of transformer coupling the individual subsystems. In addition this method of ac coupling allows the addition or deletion of Data Bus Interfaces (DBI) with no effect on the remaining interfaces. This becomes a desirable characteristic since different space missions might require a different hardware configuration.

The IAS proposed for Shuttle is configured in such a manner that data established in one subsystem will be required by other subsystems. It would become highly desireable to permit the exchange of data between subsystems without loading the main data bus between the computers and the subsystems. Such a scheme would obviously require a sub-bus and the need for such a bus will be established after the Shuttle data requirements have been firmly established. If the need for a sub-bus should exist it would be imparitive that the Data Bus Interfaces (DBI) be of standard design to eliminate multiple design and development.

Data Controller

Three basic concepts have been considered for the control of data between the Shuttle subsystems. A basic description of each method follows:

A. Fixed Frame Mode

In this mode data is assigned a particular frame or time slot in which the data is to be transmitted. The data transmission would be repeated within the particular frame or time slot each time a synchronization word is generated.

B. Request-Reply Mode

In this mode the computer would request via the data bus certain data from a subsystem. The specified subsystem would respond via the data bus and supply the requested data. This method consumes a great deal of data bus time and is ineffective when large volumes of data need to be transferred between subsystems.

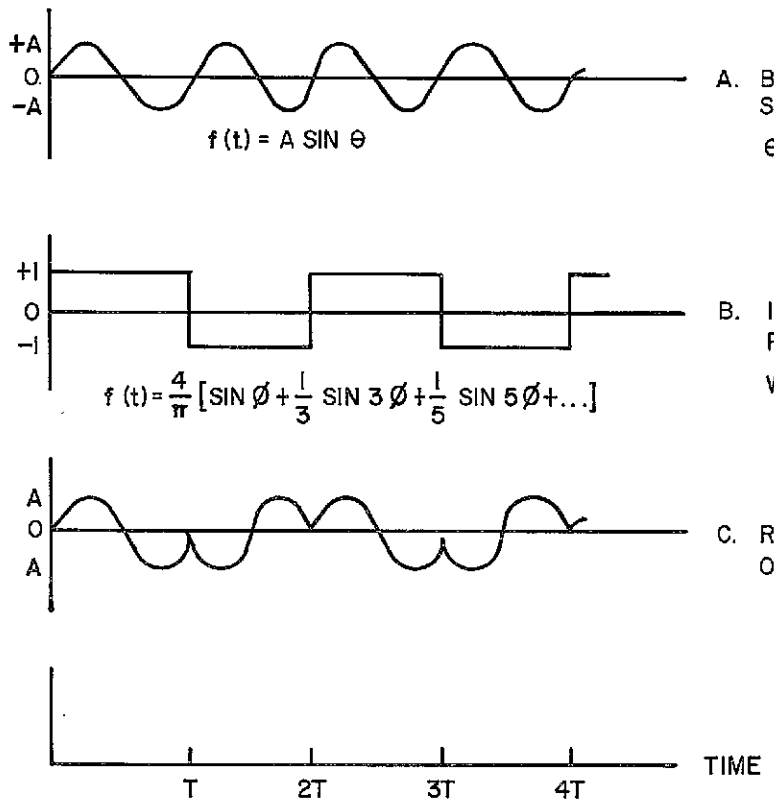
C. Subsystem Polling

A very attractive means of data distribution is via the method of polling. In this method of data distribution the individual subsystems are interrogated in an orderly manner to determine if they have data to transmit. If by this "roll call" method of interrogation the subsystem does have information to transmit, the interrogating device would initiate the data transfer in serial form.

The Data Controller used for performing the polling activities and initiating data transfer may be a small computer controlled logical device or a small computer itself.

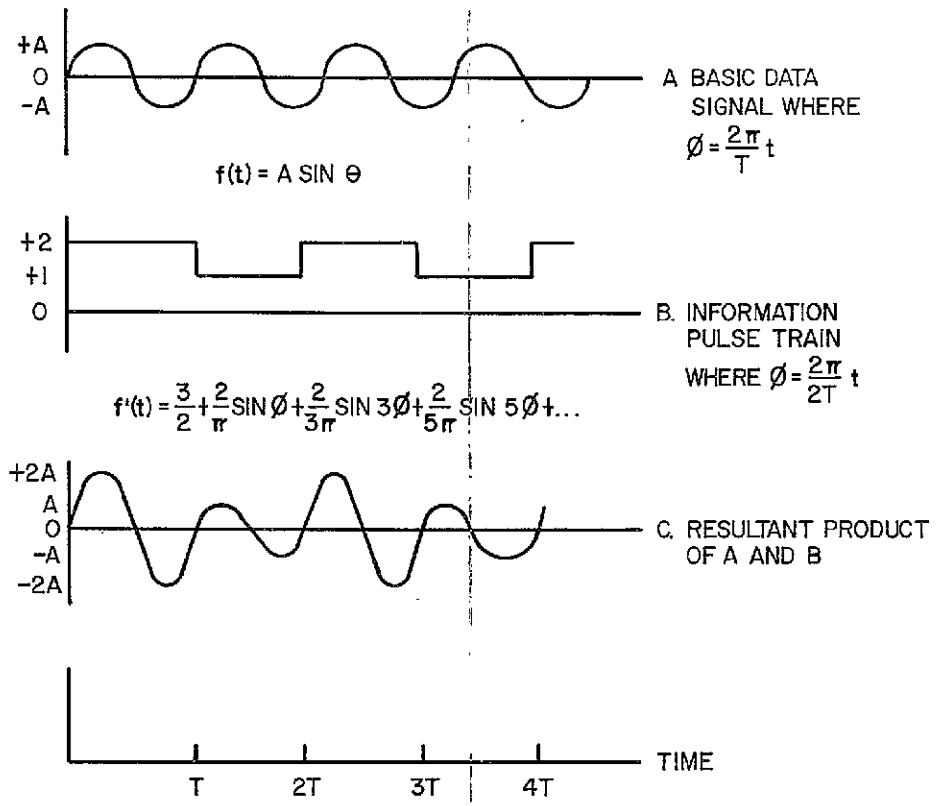
The Data Controller method of data distribution is by far the most efficient means of data transfer since the device is external to the central computer intervention.

At any time the computer has a critical need of refreshed data, it may interrupt the sequence and obtain the information directly from the subsystem via the data bus. The sequence would then continue in the previously established sequence. The computer at any time may command the Data Controller to to change the sequence by supplying the new programmed sequence.



BI-PHASE DATA WAVEFORMS

FOLDOUT FRAME 1



LEVEL SHIFT DATA WAVEFORMS

FOLDOUT FRAME 2

FIGURE 3

Data Management Concept

Four basic processor configurations were considered by Shuttle contractors for use as the data management system. The types considered and a basic description of each follows;

Uniprocessor - This the most basic of available computer configurations, consists of a single processor unit (PU) which may use a number of memory modules. In this configuration the single processor may be small and dedicated to a particular task or may be large and capable of handling multiple tasks.

Federated Computers - In a federated configuration several uniprocessors are assigned and each may be capable of handling one or more subsystem functions. The only data paths between processors would be via the data bus.

Dual Processors - This computer configuration would consist of two dedicated processors and independent memory modules for each processor. The advantage of such a computer configuration would be the communication link between processors, reducing data exchange via the data bus.

Multiprocessor - In the multiprocessor configuration any of a number of processors may have access to all memory modules and input/output units. The executive program determines which processor is idle and assigns a given program to the appropriate processor. Obviously such a system is multitask oriented and would perform the necessary computations for all subsystems.

Many considerations were made by Shuttle contractors and the individual laboratories of MSFC in the initial or conceptial design of the Shuttle Management System.

It is beyond the scope of this report to perform a detailed analysis on the requirements effecting the selection of a data management system; however there are some considerations that play a major role in the selection of the computer complex. The most inportant of these are:

Location of Subsystems - A preliminary study of the Shuttle indicates that the hardware for individual subsystems will be physically distributed from the front to the rear of the vehicle. Such a distribution requires Data Distribution Units (DDU) whereby the data from the sensors are collected, converted to digital form, formatted, and applied to a data bus on request.

Checkout, Monitoring, Fault Isolation (CMFI) - Some of the aspcts of checkout and fault isolation were described in previous sections. Generally, it would be necessary to have one processor for checkout and fault isolation. It would be necessary to maintain the individual subsystem tests on a bulk memory device since the tests would vary in nature and depth from subsystem to subsystem.

In addition to performing the checkout routines and isolating the fault to a replaceable unit the processor must be capable of monitoring consumables and all critical test points. Most of the monitoring to be performed will be in the form of limit checks.

Guidance, Navigation and Control - These three important functions may be defined in very general terms as follows;

A. Guidance - The computation of maneuvers necessary to reach a given attitude and trajectory.

B. Navigation - The determination of attitude and velocity using onboard equipment.

C. Control - The execution of the maneuver or the correction to the attitude or velocity.

The boundaries between these functions are sometimes not well defined since the Control function is dependent on the Guidance function which in turn is dependent on the Navigation function.

Preliminary studies indicate an extensive computational capability will be required for GN&C due to the computations required and the large amount of digital filtering to be performed.

Selected Computer Complex

At present two different schemes have been proposed for the Data Management System.

The first of the proposed computer schemes (Figure 2) consists of dedicated processors for each of the individual subsystems. The problem with such a federeated computer complex is immediately obvious. The number of computers required would be approximately 30 including the redundant processors. Different types of computers with different speeds and word lengths would be required to satisfy the subsystem requirements. The interface for the individual processors would be complex and data transfer schemes via the data bus would be complicated. Such a scheme would create a great deal of duplication and cause high penalities in memory requirement in addition to the added weight and power problems.

The second management system considered was a multiprocessor arrangement with additional dedicated proccessors (Figure 4) where needed. The additional processors would be assigned depending on the subsystems physical location and the locations of the sensors to be sampled. The advantages of such a computer complex would be;

a. Modularization and minimization of memory requirements.

b. Grouping computations to reduce data exchange between individual computers. v.e. (Guidance and Navigation Computations)

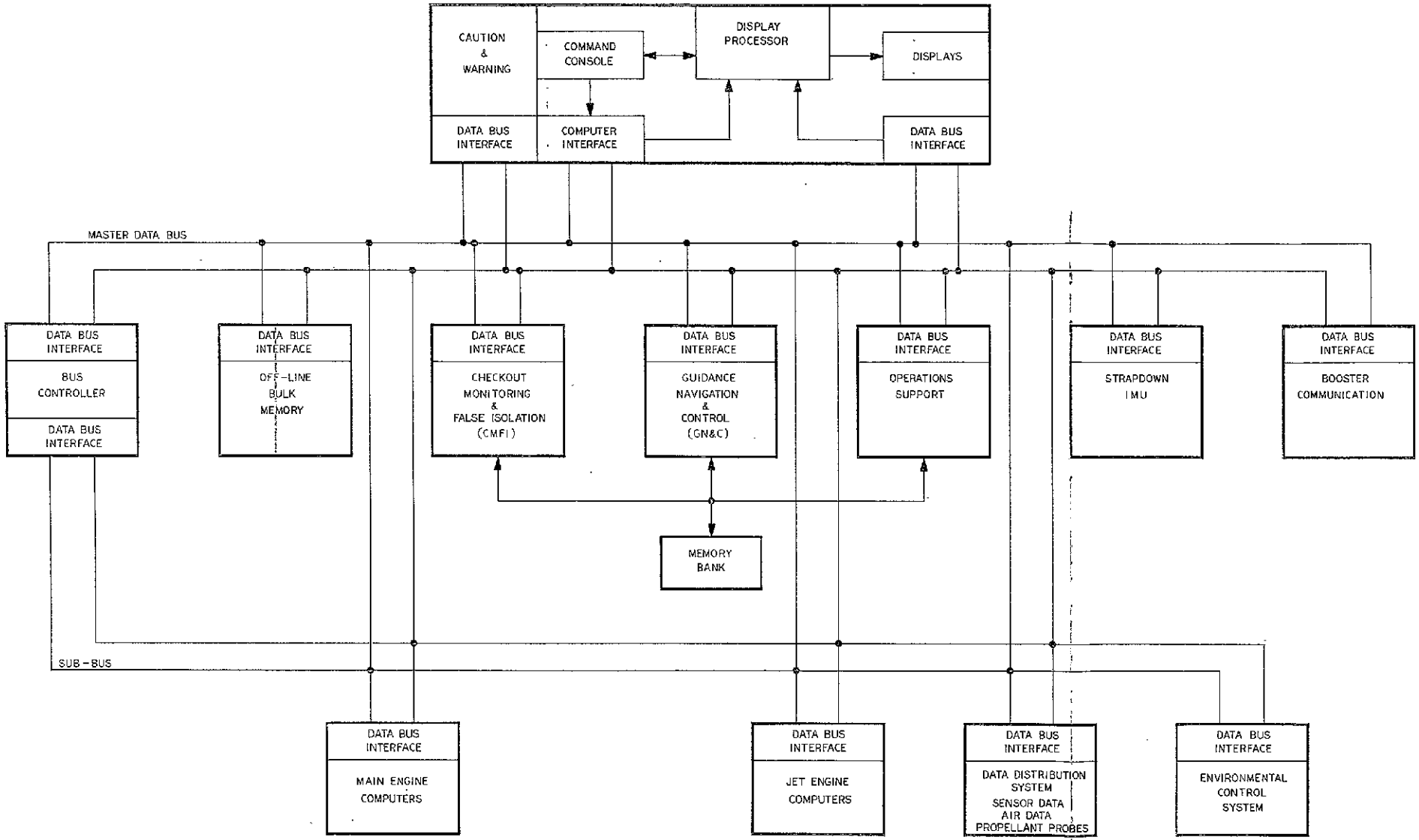
c. Reduction of weight and power requirements.

d. Provides for easier software changes for different missions.

To date the final computer complex configuration has not been established. While the concept of an autonomous Shuttle vehicle is most attractive, the cost of developing such a vehicle would be prohibitive. Therefore a tradeoff between the previously described computer concepts can be expected.

The depth to which the onboard checkout system performs its test will most likely be limited. To conserve weight, power, and cost GSE will most probably be used to a greater extent than originally planned.

The latest reports from the Shuttle contractors indicate a mix of onborad and ground checkout systems will be used; expecially in the maintenance and pre-launch modes of operation.



FOLDOUT FRAME 1

FIGURE 4

FOLDOUT FRAME 2

Simulator Application

The latest contractor studies indicate a trend toward centralizing the checkout computer and associated electronics. The onboard and ground checkout computers will communicate with the subsystems via the data bus. The checkout scheme will be simplified to some extent due to the fact that the ground checkout computer will perform the detailed tests of the individual subsystems. The onboard checkout computer would be relieved of this burden in the pre-launch and post-landing phases.

It is most important that the ground checkout computer be capable of checking all modes of operation of the subsystems; and it is equally important that the software of the ground and onboard checkout computers be completely checked under real time conditions. The most efficient and inexpensive means of program verification would be by simulating the subsystems by digital computers, using a data bus interface as a means of communication.

The DEE-6 simulator consisting of an SDS-930 computer and associated electronics has been considered as an appropriate computer system for simulating Shuttle subsystems. The computer and software were demonstrated in simulating the Instrument Unit of Saturn V. The Transfer Equation Preparation Manual [3] has been sufficiently documented to aid the engineer in writing the equations to simulate the given subsystem. The equations themselves represent discrete and continuous (analog) functions and as such may represent multimode subsystem functions. The flexibility of the equations permits the simulation of any phase of Shuttle operation from pre-launch to post-landing and maintenance operations.

A study was made of the existing simulator to determine what modifications or improvements were needed for Shuttle program verification. The specifications of the checkout system has not been adequately defined to determine which of the modifications proposed are actually needed.

There is the possibility that all of the modifications proposed for the SDS-930 would still not be adequate for subsystem simulation.

From this standpoint parallel studies were made of existing computer systems to determine which computers might be capable of simulating all of the individual Shuttle subsystems. The results of these studies are included in the proper appendices.

Conclusions and Recommendations

Previous studies indicate a need for improvements to the DEE-6 simulator in order to adquately simulate the Shuttle subsystems. The modifications while entirely feasible represents a substantial hardware modification. The impact on the existing software would be considerable if all of the proposed hardware modifications were incorporated. Additional studies would be required to determine if the DEE-6 simulator would be adequate for the intended application.

Additional computer configurations were considered and are enclosed in apppendices C through E as originally submitted by the computer firms. In cases where test runs were performed the results of the tests are included.

Tests were run on the UNIVAC 1108 computer since the system was available. (See Appendix C) The results of the test indicate the multiprocessor configuration will out perform the best SDS-930 configuration available by a considerable margin. The test program was written in Fortran for convenience but the response of the system could be improved appreciably by writing the subroutines in assembly language. The equations themselves could be written in Fortran for the convenience of the engineers thereby reducing the pre-processing computations now taking place.

The UNIVAC 1108 is proposed as the most suitable processor for the following reasons:

1. The multiprocessing capability will improve the equation response time considerably.
2. Internal runs in the real time mode may be performed for a given subsystem data base. The stimuli would be from a card reader and the output would be to tape. The system clocks would be used eliminating the need for expensive hardware.
3. The post history data evaluation may be performed on an off-line basis using another computer system.

4. No hardware modifications are needed and no expense would be incured since the runs would be internal. The existing UNIVAC 1108 could be used as it exists.

5. If the tests prove feasable the DDP-224 computer presently attached to the UNIVAC 1108 could be used to interface with the data bus and checkout system.

The SEL 840 MP (Appendix B) was chosen as the 2nd most feasible concept due to the computational ability and the existing software system. This configuration would require the least amount of modifications, would incur the least amount of expense, would provide the best software capability of the small computer concept and could be made operational before any other system proposed. In addition this hardware/software configuration with the existing analog capability provides more flexibility than any of the other configurations studied.

The SDS-930/SEL 840A configuration (Appendix A) was chosen as the 3rd most likely candidate due to the limited amount of modifications needed to upgrade the simulator for Shuttle simulation. This configuration provides the least amount of flexibility of all configurations studied and requires a greater expense to incorporate than the multiprocessor proposal (Appendix A).

Additional studies were made of larger computer systems in case the Shuttle simulation requires greater computational ability than the previously proposed systems. The results of these studies are included in appendecies C through E.

Appendix A

Proposal
Minimum SDS-930 Configuration

Philosophy of SDS-930 Configuration

The SDS-930 computer was chosen as the central processor for the simulator since the software for the simulation has been developed. The proposed hardware changes will have some impact on the existing software in the preprocessing and real time phases.

Previous studies of the simulator (Sperry reports 1, 2, and 3) indicate a need for equation response improvements in all areas; particularly in the solution of analog equations. To meet the response requirements it becomes necessary to improve the equation solution scheme and to perform parallel processing.

The proposed hardware and software combination should be adequate to simulate any individual subsystem on the Shuttle. With minor modifications to the existing SDS/RCA interface equipment the SDS-930 should be capable of communicating with a checkout computer via the data bus. The SDS-930 is capable of formatting and commutating the data, eliminating the need for an additional interface computer. Since the Shuttle computer configuration and I/O data format has not been firmly established, the modifications required of the SDS interface link cannot be detailed at this writing. The hardware modifications for the existing SDS data link, excluding the actual data bus interface has been estimated at $7,000.

Proprosed Scheme

Since an improvement in simulation response is needed the hardware configuration in figure A2 has been proposed. This configuration incorporates all of the improvements suggested in Sperry reports 1 through 3.

The SDS-930 will be used as presently configured with additional transfer buffers on W channel, and shared memory modules and a shared clock on E channel as indicated in figure A2. No major modifications are required of the SDS-930 computer.

The minimum configuration proposed is the SDS-930 computer with an SEL 840 computer and associated periperal equipment as a slave processor. The SEL 840A was chosen because of its I/O capability

and the available floating point hardware. Any 24-bit machine with comparable I/O structure, speed, and computational ability may be used.

In general, the SDS-930 computer will process all switch actions (card reader input), discrete functions, and timed discrete equations. In accomplishing the latter the SDS-930 will use the existing DEE-6 simulator clocks. The equations to be processed by the SDS-930 may be listed as follows:

	Type
S_{xxxx}	Switch
A_{xxxx}	Discrete In
B_{xxxx}	Discrete In
D_{xxxx}	DDAS Discrete
X_{xxxx}	Discrete Out
Y_{xxxx}	Discrete Out
V_{xxxx}	(Discrete) Internal Variable
C_{xxxx}	Segment Status (for analog)

The last variable C_{xxxx} is an analog status variable introduced to eliminate the amount of variable status checking and data transferring by the SEL 840A computer. As an example the following equation is given:

$$E_{xxxx} = A_1 * B_1 * V_1 * A_{30} * B_{30}/\text{analog args}/\$$$

$$C_{xxxx}$$

In the proposed scheme the SDS-930 will solve one additional discrete equation of the form;

$$C_{xxxx} = A_1 * B_1 * V_1 * A_{30} * B_{30}$$

This eliminates the need of the SEL 840A computer acquiring all of the individual status variables from the SDS-930 computer. The SEL 840A would now need to check only one status (C_{xxxx}) located in the shared memory module. The analog equation would reside in the SEL 840A computer in the form of;

$$E_{xxxx} = C_{xxxx}/\text{analog args}/\$.$$

Since the C_{xxxx} equation would be a cross reference of all variables affecting it, the burden of establishing the status of the variable C_{xxxx} would be on the SDS-930 computer thus enhancing the analog computational ability of the SEL 840A.

To adequately simulate a subsystem it is necessary to maintain all of the equations in core memory. A study was performed to determine what impact would be made on the present simulator software. In addition, to reducing the present equation length by using relative addresses for equation ID, and reformatting the equations, a number of subroutines may be completely eliminated from the real time program. For a detailed explanation of the equation formatting see Sperry Report 3, CSC Insert, Section 2.

A functional drawing (Figure A1) has been included to define the data exchanges between processors and the types of solutions each processor performs. In the functional diagram a second variable (T) has been introduced and will be solved by the SEL 840A processor. The equation represents a threshold analog equation of the form;

$$T_{xxxx} = E_{xxxx}//\text{pick-up, drop-out}//\$ \text{ cross references}$$

In essence the analog equation E_{xxxx} has passed through a threshold value when the equation T_{xxxx} is scheduled for evaluation. When this occurs the SEL 840A checks to see what cross references are affected by the threshold crossover. If the cross references are discrete equations their ID's (P. C./Rel-Addr) are sent to the Queue of the SDS-930 computer and if the cross references are analog equations their ID is scheduled in the Queue of the SEL 840A.

Data Exchange

The simulator has been configured to minimize data exchange between processors. To accomplish this end, all discrete equations and timed discrete equations are processed in the SDS-930 while the analog equations and threshold equations are processed in the SEL 840A computer (Figure A1). The new variables (C and T) have been introduced to aid in optimuizing the simulation language.

The arguments to be passed between processors via the isolation buffers will consist of one coded word per queue entry. The word will be transmitted (only when buffer is empty) simultaneously with an interrupt. The argument will consist of a Processor Code (P.C.) and Relative Address of the equation to be solved. The receiving computer will fetch the argument, making the isolation buffer available for further transmission, and place the equation in the proper queue to be solved.

The shared memory devices will be pulse isolated devices and will be addressable by either processor as a unit device. Shared memory device 1 will contain the status of the following variables;

Shared Memory 1: E Status
F Status
T Status
C Status

Shared memory module 2 will maintain the floating point values of the E and F analog equations with two words per value (SEL 840A format). The SEL 840A will notify the SDS 930 (Group/Sub-Group) when an analog equation has changed value ($\pm$5 millivolts) and the SDS-930 will fetch the floating point value from the shared memory, convert to a fixed point value, format the variable, and seat in the proper table for commutating via the SDS interface to the data bus interface.

Provisions have been made for recording all values on history tapes (Figure A1). The history tape will consist of 4 word records of the following format for each of the processors;

	DISCRETE FORMAT	ANALOG FORMAT
Word 1	Time	Time
Word 2	Identification	Identification
Word 3	Old Status	Old Value
Word 4	New Status	New Value

The history tape capability has been included as a convenient means of analyzing the subsystem simulation performance. As such, a shared time of day clock has been included in the design and will be used to read the nearest 0.1 millisecond.

Provisions should be included in the software to select individual variables to be recorded on the history tape. This is a necessary feature as the recording process of "all" variables degrades the computational ability of the simulator considerably. To achieve this goal it would be necessary to update the individual equations via the card reader prior to the real time simulation.

The suggested means of selecting the variables to be recorded is by setting or resetting the sign bit of the first word (word 0) of the individual equations. The normal state (+) would inhibit the recording and the opposite state (-) would permit recording. With such a scheme no recording would take place unless the operator has taken positive action (card deck) in requesting the record.

Analog Clocks

In keeping with the scheme of external clocks for analog equations, three clock controllers and associated clocks would eliminate table compressing for expired analog equations and would enhance the analog simulation capability of the simulator.

The external clocks consist of three individual control units and a number of associated clocks. The nominal number of clocks has been set at 8 per unit however the capability can be expanded to 15 per unit for a maximum of 45 active analog equations. It is anticipated that this is the maximum number of equations that the processor could handle.

The equations are to be assigned so that the equations having the smallest Δt (10 ms) will have the highest priority. These equations will

be associated with interrupt Group 1, levels 1 through 15. The next set of equations will have a Δt of 20 milliseconds and will be associated with interrupt Group 2, levels 1 through 15. The third group of clocks have a variable Δt setup by the processor and have the largest Δt and the least priority. The priority arrangement has been chosen to give the highest frequency equations the greatest priority.

The clock controllers will be designed to have available on request from the computer, the highest priority clock not in use. For devices 21 and 22 the clock will become active on request and will continue interrupting the processor every Δt time until the clock is halted. Device controller 23 will operate in a similar manner however the clock will not become active until the desired Δt is sent to the available clock.

Shared Memory

The shared memory controllers will be identical in design and will contain the necessary status values in unit 11 and the analog values (in double word format) in unit 12. The devices will communicate via pulse isolation devices to eliminate ground loop problems between the processors.

Analog Capability

The analog computer has been included as a possible addition only. Standard interface designs for the SEL 840A exist for most of the popular analog computers available. The digital-to-analog and analog-to-digital converters are off the shelf items available from SEL.

Estimated Cost (Minimum Configuration)

It is assumed that a SEL 840A computer and associated perperials are available for mating to the DEE-6 Simulator. The periperals have been assigned unit numbers to conform to the logical device number (LDN) assignments within the SEL 840A Executive software.

The special devices have been given available device numbers as indicated in figure A2 and the estimated cost of the devices is as follows;

Assignment	Description	Material Cost
Unit #20	Double Buffered Clock	$ 2,300
Unit #24	Isolation Buffer 1	1,500
Unit #25	Isolation Buffer 2	1,500
Unit #21	Clock Controller 1	2,000
Unit #22	Clock Controller 2	2,000
Unit #23	Clock Controller 3	2,000
	Fixed Δt clock units (30)	2,000
	Programmable clock units (15)	2,000
	840A Interrupt modules (2) (Group 2 and 3)	5,000
Unit #11	Shared Memory Controller 1	1,500
Unit #12	Shared Memory Controller 2	1,500
	Shared Memory Module 1	11,000
	Shared Memory Module 2	11,000
	SDS Hardware Modifications	7,000
	Total Material Cost (Minimum Requirement)	$52,300

Manpower Requirement

A table has been included to reflect the manpower requirement needed to incorporate the modifications previously described. A definition of responsibilities is included for each position.

Systems Analyst

Function:

1. Develop new software concept.
2. Define detailed hardware changes.

3. Insure integration of hardware and software.
4. Provide a math model.

Programmer

Function:

1. Incorporate new software scheme.
2. Assist in hardware interface checkout (diagnostics)
3. Aid in analyzing math model.

Engineer

Function:

1. Design special units.
2. Provide design modifications for existing SDS interface.
3. Aid technician in hardware checkout.
4. Assist in system integration.

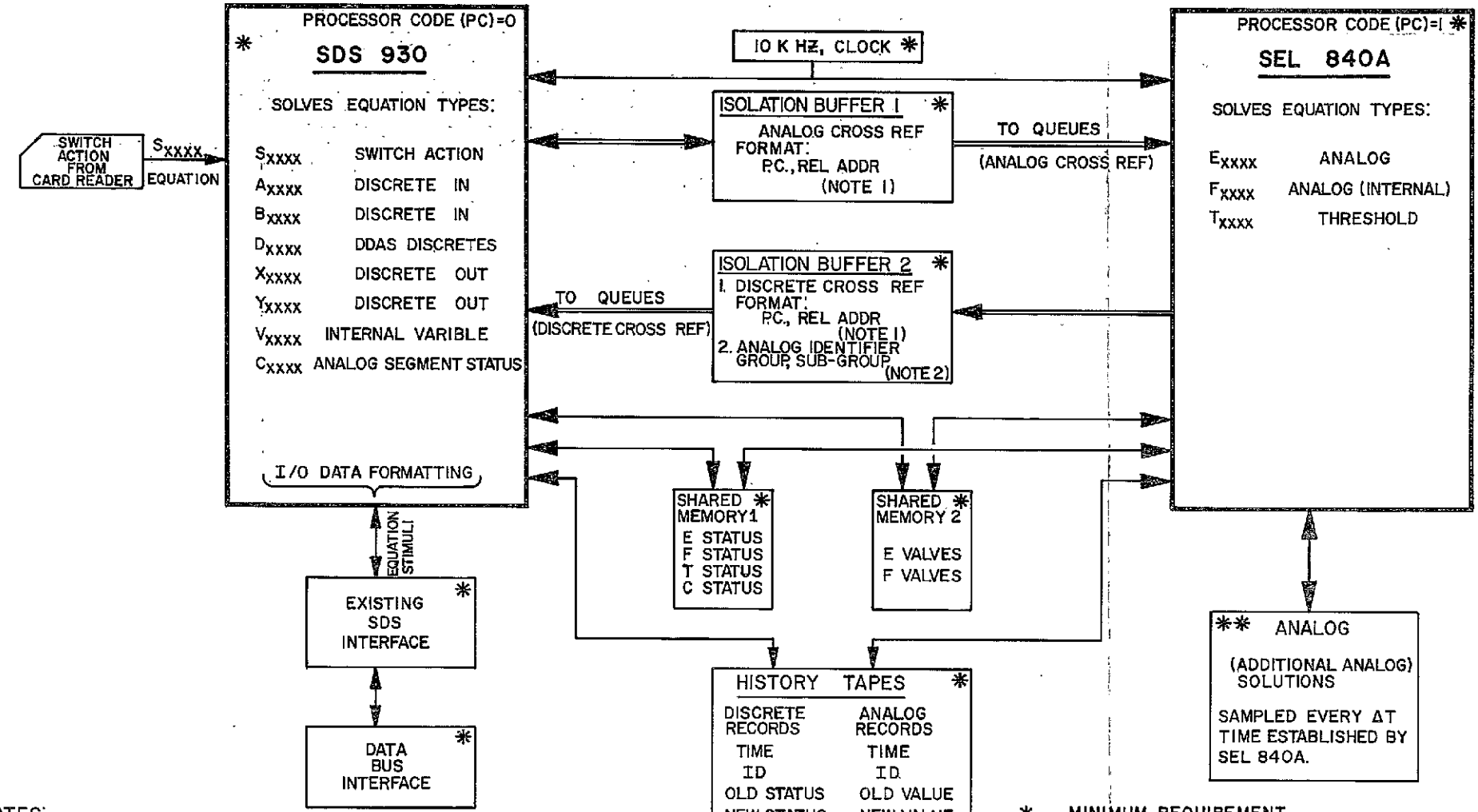
Technician

Function:

1. Wire special units.
2. Modify SDS data link
3. Assist engineer in checkout

The projected manpower effort to incorporate the modifications and provide system integration is;

Systems analyst	18 man months
Programmer	18 man months
Engineer	12 man months
Technician	12 man months



NOTES:

1. ALL ARGUMENTS PASSED IN 24 BIT WORDS WITH FOLLOWING BIT ASSIGNMENT.

BITS 0—2	3—5	15—23
NOT USED	PROCESSOR CODE	RELATIVE ADDRESS

2. AFTER SOLUTION OF ANALOG EQUATION BY SEL 840A THE ANALOG GROUP AND SUB-GROUP IS PASSED TO SDS 930.

GROUP	SUB-GROUP

* MINIMUM REQUIREMENT

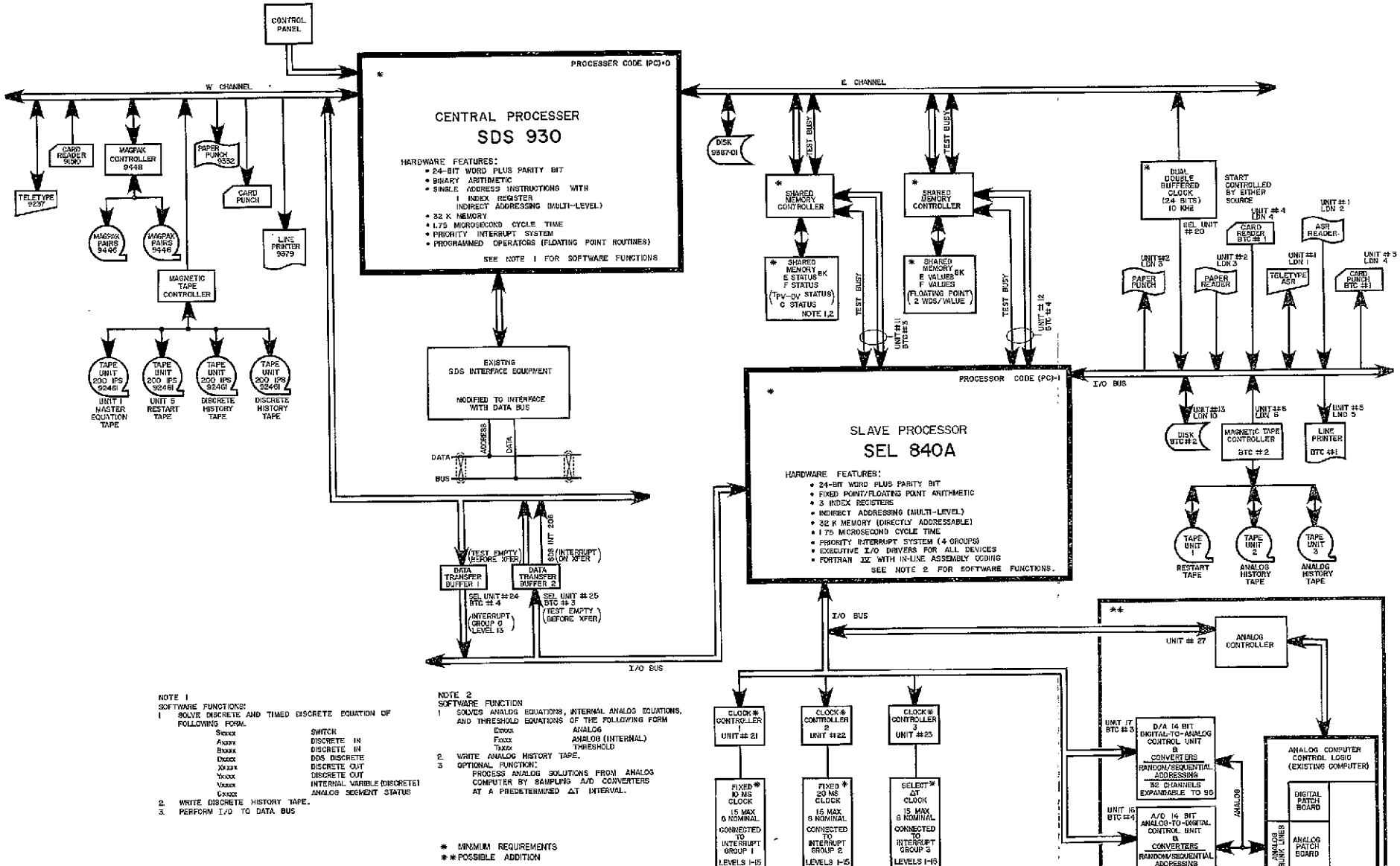
** POSSIBLE ADDITION

FOLDOUT FRAME 1

FOLDOUT FRAME 2

DEE-6 SIMULATOR FUNCTIONAL DIAGRAM

FIGURE A-1



FOLDOUT FRAME

FOLDOUT FRAME 2

FIGURE A2

Appendix B

Proposal
SEL 840 Multiprocessor

Philosophy of SEL 840 MP Configuration

The SEL 840 MP configuration was chosen as a possible candidate for simulating the shuttle subsystem(s), using the MARDSLVC simulation language. The multiprocessor arrangement was chosen over the SDS 930 configuration due to the expanded memory capacity and parallel processing capability.

The 840 MP is especially attractive for simulating analog functions (as defined in the MARDSLVC language) because of the floating point hardware. In addition, the memory available should be large enough to maintain the data base of any Shuttle subsystem. This in itself is an important feature as pointed out in previous reports.

The existing 840 MP is configured as needed except for the additional clocks needed for each of the processors, and the I/O computer needed for recording history data and communicating with the data bus. Not drawn in figures B1 and B2 is the CI 5000 analog computer and computer controller associated with the SEL 840 MP. The interface for the analog computer has been built and debugged and is addressed as unit 27 in the SEL addressing scheme. The associated converters (A/D and D/A) have been installed in the processor and are addressed as units 16 and 17 respectively. This added analog capability would enhance the simulation of the vehicle subsystems since the higher frequency equations could be handled by the analog computer.

The solution of the equations will be broken into 3 catagories; (1) discrete, (2) timed-discrete, and (3) analog. The discrete equations and timed discrete equations will be processed by multiprocessor 0. The stimuli for initializing the equations will be from the switch action profile from the SEL 810A processor or from the discrete function from the data bus and I/O computer. The types of equations to be processed by processor stage 0 are;

	Type
S_{xxxx}	Switch
A_{xxxx}	Discrete In
B_{xxxx}	Discrete In

Type	
D_{xxxx}	DDAS Discrete
X_{xxxx}	Discrete Out
Y_{xxxx}	Discrete Out
V_{xxxx}	Internal Variable
C_{xxxx}	Segment Status

The C status is a new variable introduced to eliminate the enormous amount of data passing that would otherwise be required. The variable C in itself represents the status (off or on) of an analog segment and is processed by processor 0 since it is a discrete function. An example is introduced here to clarify the use of the new variable.

The new equation scheme will require one status for each segment of an analog equation. Therefore in the equation;

$$E_{xxxx} = A_1 * B_1 * V_1 * A_{30} * B_{30}/\text{analog args}/\$$$

$$C_{xxxx}$$

the new variable C_{xxxx} is introduced to replace the previous status variables, thus eliminating the need of the analog processor (processors 1 or 2) of determining their status. It now becomes necessary for processor stage 0 to determine the value of the status variable (C_{xxxx}) and update the values in shared memory when required.

The analog processors would be required to check only one status variable of the equation;

$$E_{xxxx} = C_{xxxx}/\text{analog args}/\$$$

thus enhancing the analog computation capability considerably. Since Processor Stage 0 will solve all equations of the form;

$$C_{xxxx} = A_1 * B_1 * V_1 * A_{30} * B_{30}\ \$C.R.$$

the analog processors would be relieved of this burden and the amount of data exchanges between processors would be reduced considerably.

Extensive studies of the MARDSLVC simulation language has proven the need for maintaining the entire subsystem data base in core memory. If this scheme is adopted many of the subroutines presently being used may be eliminated. In addition, many of subroutines used for processing the equations may be reduced in length due to the new equation format developed by CSC (See Sperry Report 3, CSC Insert, Section 2). The equations themselves will be reduced in length due to the new equation format.

The functional drawing (figure B1) has been included to define the data exchanges between processors and the types of solutions each processor performs. The functional diagram delineates two processors for the solution of analog equations; one processor dedicated for analog solutions of odd numbered ID's and a second processor for analog solutions of even numbered ID's. This parallel method of solving analog equations will both increase the number of analog equations that may be processed, and increase the maximum frequency of the equations in themselves.

The analog processors will use the status variables (C_{xxxx}) within the shared memory to determine the status (true, false) of each segment of the analog equation. These status values were previously established by Processor Stage 0. The analog processors will then solve the scheduled equation, set the analog status (E status, F status to off or on), store the floating point values in shared memory, and present a 4 word record to the I/O computer in the following format;

	Format
Word 1	Time
Word 2	Identification
Word 3	Old value
Word 4	New value

The analog processor in addition to solving the analog equations will solve one additional equation termed the Threshold or T type equation.

The Threshold equation is of the form;

$$T_{xxxx} = E_{xxxx}//\text{Pick-up, Drop-out}//\$ \text{ C.R.}$$

and schedules the additional equations (cross references) for evaluation when the analog equation (E_{xxxx}) passes through a threshold value (pick-up value or drop-out value).

Multiprocessor Stage 0 will be responsible for all discrete and timed discrete equations and will send 4 word history records to the I/O computer in the following format:

	Format
Word 1	Time
Word 2	Identification
Word 3	Old Status
Word 4	New Status

The I/O computer (SEL 840A or equivalent) will be responsible for recording the data.

In a real time simulator the logging of data should be minimuized to the greatest extent possible. This is imparative with the MARDSLVC simulation language since there are an enormous amount of variables processed. For this reason it becomes necessary to make the variables to be recorded be selective prior to the realtime simulation. Provisions should be incorporated in the hold mode of the simulation to suppress all recording unless positive action has been taken (card deck) specifing the variable to be recorded. In the case of analog equations, an additional input should be required specifying the limits for recording analog values.

The suggested means of selecting the variables to be recorded is by setting or resetting the sign bit of the first word (word 0) of the selected equation. The normal state (+) would inhibit the recording and the opposite state (-) would permit recording. In the case of analog equations the limit set for recording would have an additional effect. With such a scheme no recording would take place unless the operator has taken positive action.

Clock Structure

Two clock subsystems and corresponding interrupts will be used with processor stages 1 and 2 to aid in solving analog equations. The clock subsystems (type B) will automatically assign equations with the smallest Δt the highest priority within the clock subsystem. The device will then notify the computer of the assigned clock so there will be a correlation between the equation ID and the clock assigned. The combined clocks and interrupt structure will provide for a maximum of 90 analog equations to be active at a given time.

An additional clock subsystem (type A) will be supplied for use by Multiprocessor Stage 0. This subsystem will be used for timed discrete functions and will be limited to 20 bits in length with a clock rate of 1 KHz. The interrupts will operate on a priority scheme with a maximum of 45 timed discrete functions with associated interrupts.

Shared Memory

The shared memory capacity of the SEL 840 MP is expandable from 8 K to 65 K words in 8 K increments. The following status tables are included in the shared memory and are addressable by all 3 processors;

E	Status
F	Status
T	Status
C	Status
E	Values (2 floating point words)
F	Values (2 floating point words)

The shared memory of course can be accessed by only one processor at a given time and this is done on a priority basis.

Buffers have been provided between the individual multiprocessor stages to allow data transfer between processors without using the shared memory. This scheme would allow data transfer between processors and would still permit the 3rd processor to access shared memory.

History Buffers

Isolation buffers will be provided between the multiprocessor stages and the I/O processor and will be used for transferring history data in 4 word records. The data will be sent/received using the block transfer channels (BTC) of the SEL processors.

Input-Output Processor

The I/O processor will have a duplicate discrete status buffer and a table containing fixed point values for all analog (external or E type) equations. The discrete values will be refreshed each time processor 0 sends over the four word record representing a change in the discrete status. The analog table will be refreshed each time an analog equation is solved.

The I/O processor will be responsible for receiving the stimuli from the data bus and checkout computer and notifying Multiprocessor Stage 0. In addition the I/O computer will be responsible for formatting the analog data and commutating the data to the data bus. The analog data that is formatted and output will be the last calculation performed by the analog processors.

Minicomputer

The SEL 810A computer has been interfaced with the SEL 840 multiprocessor and serves a useful function in both the real time and post simulation phases. In the real time phase the 810A computer will simulate manual switch actions by reading a switch action deck in a pre-simulation phase and using the available clock will present a switch action profile to Multiprocess Stage 0. Multiprocessor Stage 0 in turn will use the switch input to trigger additional equations.

In the post simulation phase the 810A processor with the display may be used for monitoring the history tape and analyzing particular plots. If so desired the history tape could be used for monitoring a certain

6

variable, recording the value of the variable within certain time limits on a second tape, and obtaining a hard copy of the plot on the Benson-Lehner 120 Plotter (Reference 5).

Multiprocessor Advantage

The multiprocessor configuration has a definite advantage over the SES proposal (Appendix A) in several areas. Besides adding an additional processor for parallel processing of analog equations, the software system of the 840 MP is developed to aloow in-line assembly language coding in the Fortran compiler. This is an expecially attractive feature as the subroutines for the simulator may be easily written in Fortran and the bit manipulation requirement can be handled by in-line assembly language coding.

The memory capacity of the multiprocessor is 32 K words of private memory plus 65 K of shared memory between processors for a total of 163,840 words. A memory capacity of this magnitude should be capable of maintaining the data base of a shuttle subsystem in resident care memory.

The existing floating point hardware is far superior to the software subroutines or Programmed Operators (POP) of the SDS-930. This is a most improtant feature in solving analog equations. The subroutines that were coded for the DEE-6 simulator for this floating point operation consume a great deal of time and should be eliminated. (See Sperry report SP-209-0339-1, Appendix A, page 12.)

The clocks in the SEL 840 MP system are external eliminating the updating of memory cells within the processor. This in itself eliminates cycle stealing from the processor for the prupose of updating clocks, and permits the processor to spend more time processing equations.

The external clock and priority interrupt configuration will reduce the number of redundant tables presently being used for processing the analog equations. The single most important feature however would be eliminating the need for compressing corresponding clock tables when an analog equation or discrete equation has expired. This single improvement justifies the expense of external clocks.

The SEL 810A processor presently attached to the SEL 840 MP would enhance the post simulation phase considerably by aiding in data reduction. A great deal of analysis could be performed by a single stage of the MP system and the SEL 810A display system and this display feature in conjuction with the Benson-Lehner Plotter Routine (Reference 5) would be a valuable tool in data reduction.

Analog Capability

An analog controller and CI5000 analog computer is presently configured to the 840 MP system. This additional analog capability could be used if required.

Estimated Cost

The greatest part of the simulator has been configured leaving the interface linkage, clocks, and expanded memory requirements as the only expense. No modifications to the multiprocessor would be required; only additional units would be added and the memory expanded to meet the simulation requirements. The estimated cost for upgrading the simulator to meet the shuttle requirements would be;

		Total Material Cost
Memory expansion 8 K modules	(1)	$21,000
B Type clock subsystems	(2)	6,000
A Type clock subsystem	(1)	2,000
History buffers to SEL 840	(3)	3,000
Multi-buffered clock	(1)	2,000
Buffers between MP stages	(6)	2,000
	Total	36,000

Manpower Requirement

A table has been included to reflect the manpower requirement needed to incorporate the modifications proposed. A definition of responsibilities is included for each position.

Systems Analyst

Function:

1. Develop new software concept.
2. Define detailed hardware changes.
3. Insure integration of hardware and software.
4. Provide a math model.

Programmer

Function:

1. Incorporate software scheme.
2. Assist in checkout (diagnostics).
3. Aid in analyzing math model.

Engineer

Function:

1. Design special units.
2. Assist in system integration.

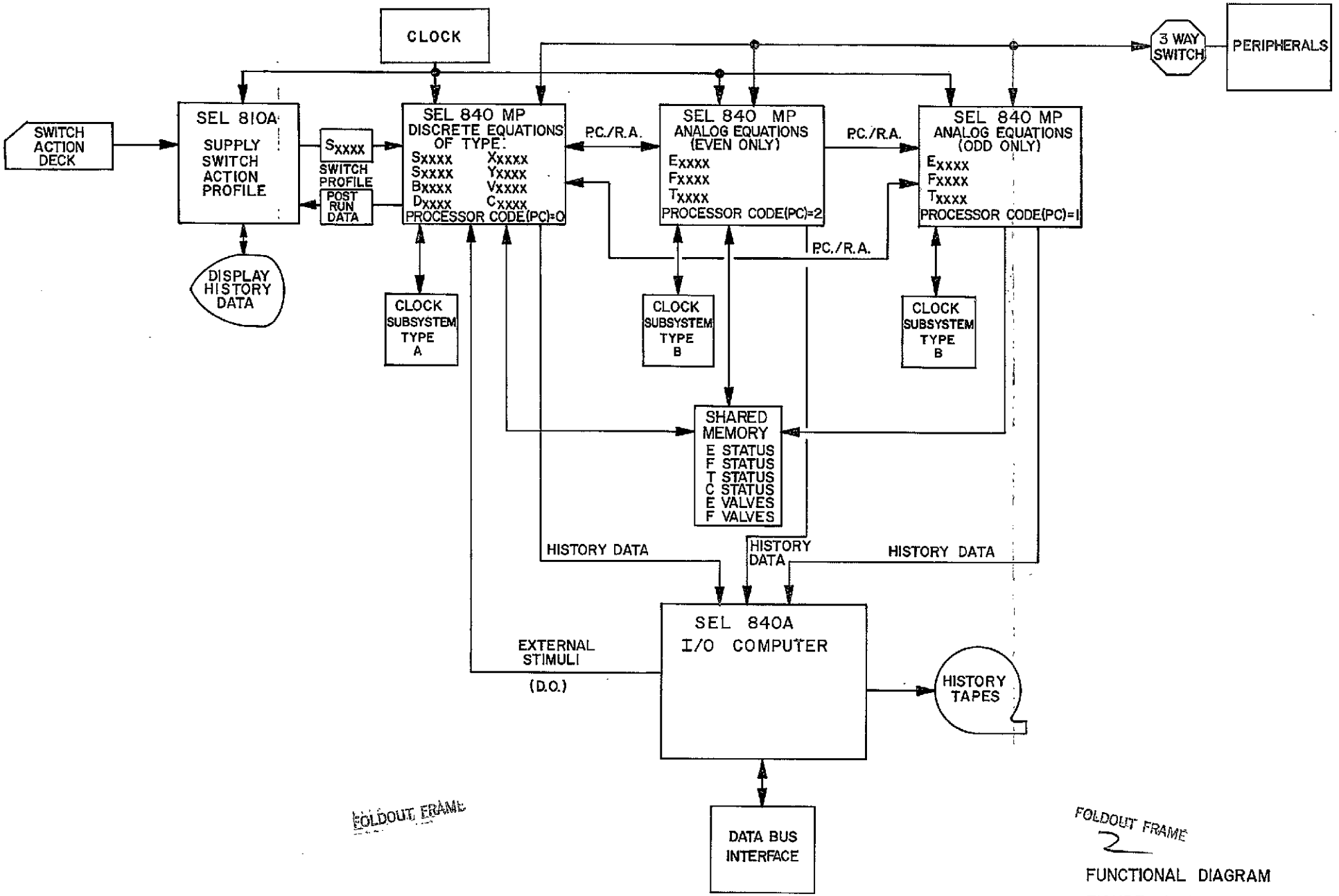
Technician

Function:

1. Wire special units.
2. Assist in checkout.

The manpower effort to incorporate the modifications and provide system integration would be;

Systems Analyst	12 man months
Programmer	12 man months
Engineer	3 man months
Technician	4 man months



FOLDOUT FRAME

FOLDOUT FRAME 2

FUNCTIONAL DIAGRAM
FIGURE B-1

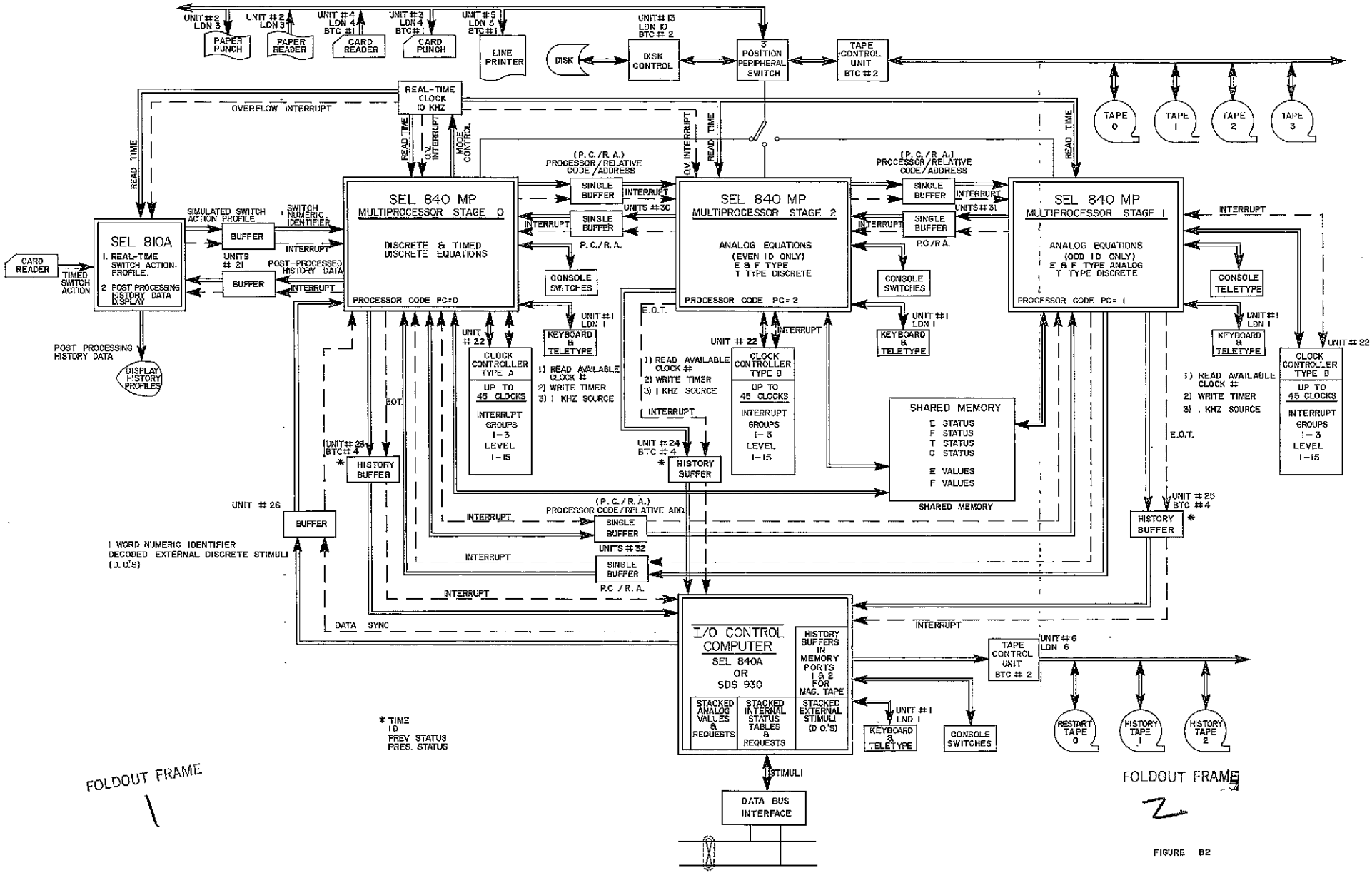

UNIT #2 LDN 3
PAPER PUNCH
UNIT #2 LDN 3
PAPER READER
UNIT #4 LDN 4 BTC #1
CARD READER
UNIT #3 LDN 4 BTC #1
CARD PUNCH
UNIT #5 LDN 5 BTC #1
LINE PRINTER
DISK
UNIT #13 LDN 10 BTC # 2
DISK CONTROL
3 POSITION PERIPHERAL SWITCH
TAPE CONTROL UNIT BTC # 2
TAPE 0
TAPE 1
TAPE 2
TAPE 3
REAL-TIME CLOCK 10 KHZ
OVERFLOW INTERRUPT
READ TIME
O.V. INTERRUPT
MODE CONTROL
SEL 810A
1. REAL-TIME SWITCH ACTION PROFILE.
2 POST PROCESSING HISTORY DATA DISPLAY
CARD READER
TIMED SWITCH ACTION
SIMULATED SWITCH ACTION PROFILE
SWITCH NUMERIC IDENTIFIER
BUFFER
INTERRUPT
UNITS # 21
POST-PROCESSED HISTORY DATA
POST PROCESSING HISTORY DATA
DISPLAY HISTORY PROFILES
SEL 840 MP
MULTIPROCESSOR STAGE 0
DISCRETE & TIMED DISCRETE EQUATIONS
PROCESSOR CODE PC=0
(P. C./R. A.) PROCESSOR/RELATIVE CODE/ADDRESS
SINGLE BUFFER
UNITS #30
P. C./R. A.
CONSOLE SWITCHES
UNIT #1 LDN 1
KEYBOARD & TELETYPE
SEL 840 MP
MULTIPROCESSOR STAGE 2
ANALOG EQUATIONS
(EVEN ID ONLY)
E & F TYPE
T TYPE DISCRETE
PROCESSOR CODE PC= 2
UNITS #31
P.C./R.A.
SEL 840 MP
MULTIPROCESSOR STAGE 1
ANALOG EQUATIONS
(ODD ID ONLY)
E & F TYPE ANALOG
T TYPE DISCRETE
PROCESSOR CODE PC= 1
CONSOLE TELETYPE
UNIT #22
CLOCK CONTROLLER TYPE A
UP TO 45 CLOCKS
INTERRUPT GROUPS 1-3
LEVEL 1-15
1) READ AVAILABLE CLOCK #
2) WRITE TIMER
3) 1 KHZ SOURCE
CLOCK CONTROLLER TYPE B
UP TO 45 CLOCKS
INTERRUPT GROUPS 1-3
LEVEL 1-15
E.O.T.
UNIT #23 BTC #4
UNIT #24 BTC #4
UNIT #25 BTC #4
HISTORY BUFFER
SHARED MEMORY
E STATUS
F STATUS
T STATUS
C STATUS
E VALUES
F VALUES
UNIT # 26
BUFFER
1 WORD NUMERIC IDENTIFIER DECODED EXTERNAL DISCRETE STIMULI (D. O.'S)
(P. C./R. A.) PROCESSOR CODE/RELATIVE ADD.
UNITS #32
DATA SYNC
I/O CONTROL COMPUTER
SEL 840A OR SDS 930
HISTORY BUFFERS IN MEMORY PORTS 1 & 2 FOR MAG. TAPE
STACKED ANALOG VALUES & REQUESTS
STACKED INTERNAL STATUS TABLES & REQUESTS
STACKED EXTERNAL STIMULI (D.O.'S)
TAPE CONTROL UNIT BTC # 2
UNIT #6 LDN 6
RESTART TAPE 0
HISTORY TAPE 1
HISTORY TAPE 2
*TIME ID
PREV STATUS
PRES. STATUS
STIMULI
DATA BUS INTERFACE
FOLDOUT FRAME 1
FOLDOUT FRAME 2

FIGURE B2

Appendix C

Proposal

Univac 1108

The system proposed by Univac is a stripped down version of the existing 1108 multiprocessor system located at the Computation Laboratory of the Marshall Space Flight Center. It is a three multiprocessor configuration equipped with 262K of core memory. The core memory is shared by all of the multiprocessors, and is provided in 65,536 word increments. The system as proposed offers several attractive features:

1. The simulation concept can be tried out on the existing Univac 1108 multiprocessor system before the procurement of any hardware. In this fashion a more definite hardware requirement can be established and only those items necessary for the simulation can be procured.

2. All eight of the shuttle subsystems can be simulated simultaneously by the proposed system providing for a more accurate checkout of the software.

3. The system can be constructed from existing off the shelf hardware with proven histories of reliability.

4. The system features modular construction which allows for ease in expandibility and maintainibility.

To determine the feasibility of the proposed system three test programs were run in real time on the existing 1108 multiprocessor system located at the computation Laboratory. All three test programs specified the solution of polynomial-type analog equations. Each analog equation had one "OR" term and three status variables. The three test programs scheduled the simultaneous evaluation of 5, 26, and 51 analog equations respectively. The first two test programs specified are evaluation interval of 5 milliseconds. The evaluation interval for the equations of the third test program was varied. Five equations specified an evaluation interval of 5 milliseconds, 20 equations specified 20 milliseconds, and 26 equations specified an evaluation interval of 40 milliseconds. The evaluation results were recorded on a magnetic drum for the first two test problems and one magnetic tape for the third program. FORTRAN was used for the evaluation of all the programs.

The post history analysis of the test programs revealed that the 1108 system solved all equations correctly. Each evaluation took approximately 0.5 millisecond. The Univac people felt that the solution time could be improved if assembly language programming was used exclusively and if the system was solely devoted to the solution of the equations. i.e., the system was not time shared.

Following is the Univac proposal for Shuttle Simulator.

1.0 INTRODUCTION

The UNIVAC 1108 Multiprocessor System has been selected for the Shuttle Simulator. This choice of hardware is based on preliminary technical discussions with the study contractor. At the same time, it is understood that the simulator requirements are still fluid, and the Univac configuration may change as more visibility is gained into the system requirements. This is highly important from a system cost factor if an 1106 Processor and memory could be used in lieu of the 1108 Processor and its memory. The use of an 1106 processor represents approximately 1.5 million dollars reduction in hardware costs. The only difference between the 1106 and the 1108 is the cycle time. The 1106 is half as fast as the 1108 with both using the same operating system and peripheral subsystems.

The capabilities of the 1108 hardware and software makes it unnecessary to add additional hardware for the simulator operation. The use of external control clocks is not required with the 1108 System. This is an intrinsic capability of the 1108 operating system. Plus, sharing of the 262K core memory between all three 1108 Processors eliminates the need of a 8K shared memory.

The system hardware cost does not include an I/O computer and its interface box, although the 1108 System has available at present a handler for such devices. The I/O computer and its interface box would be quite similar to that already available and interfacing with the 1108 System on a real time basis at MSFC.

The availability of the UNIVAC 1108 System at MSFC would make the system development of the Simulator very timely. This would allow software development to go forward without a pressing hardware operational date.

The information contained in this report includes the following:

1. Original Simulator flow diagram received from the study contractor (Figure 1).

2. Univac hardware configuration diagram (Figure 2).

3. Table of Univac hardware versus System requirements.

4. Univac hardware cost.

5. Description of each major piece of Univac hardware to be used in the Simulator.

All items described in this report are standard off-the-shelf Univac hardware appearing in the current GSA Federal Supply Schedule, GS-00S-84538. The normal delivery time for the hardware is 90 to 120 days, except for the Central Processing Units. The CPU delivery time may require a maximum of six months.

1108, 3x0 SYSTEM COST

Type or Feature No.	Description	Qty Req.	Purchase Price	Monthly Rental	Monthly Maintenance
F1053-99	1108 Multiprocessor	3	1,888,200	43,200	6,975
4009-99	Display Console	3	97,875	2,250	750
7005-70	262K Core Memory	1	1,830,000	41,765	2,490
2506-00	1108 Availability Control Unit (ACU)	1	52,200	1,200	115
0954-99	Multi-Module Access (MMA)	3	169,650	3,900	150
F0879-00	MMA Expansion	3	7,800	240	-----
0955-04	Shared Peripheral Interface (SPI)	3	58,725	1,350	75
0955-05	SPI Cabinet Expansion Kit	3	52,200	1,200	60
F0789-00	SPI Expansion from 2 to 3 cpu's	3	10,440	240	50
F5012-00	FH432/FH1782 Drum Control Unit	1	82,515	1,885	260
F0929-00	Write Lockout Capability	1	1,040	25	5
F0930-00	Second cpu Interface	1	17,905	415	25
F0930-01	Third cpu Interface	1	2,675	65	5
F6015-00	FH1782 Drum Unit	1	117,210	2,680	260
5008-12	VIII C Control	1	62,430	1,430	120
F0627-04	Translator	1	4,410	105	5
0859-00	Uniservo 8C, 7 Track Non simultaneous	4	129,940	3,000	440
3030-02	9300 Processor	1	26,500	605	150
7007-08	Storage - 8K	1	23,230	530	45

1108, 3x0 SYSTEM COST (Continued)

Type or Feature No.	Description	Qty Req.	Purchase Price	Monthly Rental	Monthly Maintenance
0711-02	Card Reader	1	6,315	145	60
0603-04	Card Punch	1	6,315	145	60
3536-98	Uniscope 100	1	3,415	70	25
F1245-05	Interface - cpu	1	1,280	40	-----
F1241-01	Character Gen Expansion 64-96	1	720	15	-----
F1242-02	Keyboard, Alpha Numeric/ Numberic.	1	425	15	-----
F1095-02	1108/9000 ICCU	1	8,920	205	45
	Totals		$4,662,745	$106,720	$12,170

SIMULATOR REQUIREMENTS VERSUS UNIVAC HARDWARE CONFIGURATION

Simulator Requirements Per Figure 1	Univac Hardware Configuration Per Figure 2
Multiprocessor #0	1108 Multiprocessor
Multiprocessor #1	1108 Multiprocessor
Multiprocessor #2	1108 Multiprocessor
Real Time Clock	Standard equipment with 1108 Processor
LSI Memory Arrays, 0, 1, and 2, 8K shared memory	Included in 262K, 36 bit word shared memory with overlapping and interleaving capability
Programmed Controlled Clocks	Function of 1108 operating system Real Time Clock routine
Buffers	Not required because each processor has access to 262K shared memory
Keyboard and Teletype and Console	Standard equipment with 1108 Processor
Peripheral Switch	Shared Peripheral Interface (SPI) controls the access of the three (3) cpu's to the units in a shared peripheral subsystem
Minicomputer Real Time Switch Action Profile	Function of 1108 operating system Real Time Clock routine and 9300 Card Reader
Minicomputer Post Processing History Data Display	Uniscope 100 interface with single or multistation configuration
Disk Control and Storage	Fastrand Drum (FH-1782) and control
Line Printer, Card Reader, and Card Punch	9300 with 600 LPM Printer, 600 CPM Reader and 75-200 CPM Punch
I/O Control Computer	Any control computer may be interfaced to the 1108 Multiprocessor System through a arbitrary device handler

2.0 SYSTEM DESCRIPTION

2.1 General

The UNIVAC 1108 System is a general purpose, high performance unit- and multi-processor system, making good use of the latest advances in computer design, systems organization, and programming technology. Its modular structure permits the selection of systems components to fulfill most efficiently the speed and capacity requirements for applications ranging from a basic job-shop system to the comprehensive public utility computing complex.

As the workload increases, this modularity also enables the addition of input/output subsystems and main storage and even additional processors to provide the full multiprocessor system. Among the principal features of the 1108 System are:

- Common resource systems organization.
- Equality among multiple UNIVAC 1108 Central Processors.
- Multiple input/output controllers.
- Large, modular, partity-checked, high speed main storage.
- Overlapped and interleaved main storage access.
- Redundancy among systems components.
- Program address relocation.
- Storage protection.
- Partial-word addressability in 6, 9, 12, an 18-bit portions as well as full-word (36 bits) and double-word (72 bits) addressing.
- High speed, random access, auxiliary storage.
- Privileged mode for the Executive system.
- Guard mode for user programs.

2.2 System Components

The UNIVAC 1108 System is organized to allow multiple processors to perform a number of tasks simultaneously under the direction of a common Executive control system. A multiprocessor system requires a much higher degree of modularity in organization than does a unit-processor system. It must be divisible into individual logical components with the following properties:

- Each system component must have more than one access path.
- Priority logic must resolve possible access conflicts.
- The failure of any individual component must not prevent continued operation of the system.
- System components must be logically removable for servicing without disabling the system.

The system is constructed of six types of components:

- Central Processors
- Input/Output Controllers
- Main Storage Modules
- Auxiliary Storage Subsystems
- System Interconnection Components
- Peripheral Subsystems

2.2.1 Central Processor

Each processor can perform all functions required for the execution of instructions including arithmetic, input/output, and Executive control. In a multiprocessor configuration, all processors are equal - the test of a true multiprocessor system. Included in each processor is its own set of 125-nanosecond integrated circuit control registers providing multiple accumulators, index registers, input/output access control registers, and special-use registers.

2.2.2 Main Storage

The main storage of the UNIVAC 1108 System is expandable in 65,536 word increments up to a maximum of 262,144 thirty-six bit words. The main storage read/restore cycle time is 750 nanoseconds. Up to four logical banks for instruction/data reference overlapping provide the capability for an effective cycle time of 375 nanoseconds. In addition, two-way interleaving of storage modules is provided to reduce the probability of access conflicts.

2.2.3 Auxiliary Storage

The auxiliary magnetic drum storage subsystems are an integral part of each UNIVAC 1108 System. Up to eight FH-432 or FH-1782 magnetic drums, or any combination of the two types, amy be attached to one or two control units. Both types of drums can transfer data at 1,440,000 characters per second. The average access time of the FH-432 is 4.3 milliseconds, that of the FH-1782 is 17 milliseconds.

2.2.4 Interconnection Components

It is an essential characteristic of multiprocessor systems that there must be provision for sharing of all of main storage and all I/O subsystems by all processors in the system. This sharing must be on the basis of both established priorities and reactive priorities that may change during processing. In the 1108 System, Multiple Module Access Units (MMA) provide this access to the storage modules by several processors, and the Shared Peripheral Interface (SPI) similarly enables the sharing of I/O subsystems.

2.2.4.1 Shared Peripheral Interface

The Shared Peripheral Interface (SPI) controls the access of up to four Central Processors (CPU) and Input/Output Controller I/O channels to the units in a shared peripheral subsystem. Access to shared peripheral subsystems is determined primarily by time of request. If two requests are made simultaneously, the Central Processor or Input/Output Controller on the lower numbered SPI Input/Output channel receives priority. In case of a busy condition, or a priority conflict, the Executive automatically stacks the request until the SPI channel is available. First-level queuing is handled in the SPI itself. The Executive stacks longer queues and keeps track of the number of I/O function requests outstanding.

Input/Output subsystems may be either single- or dual-channel. Single-channel subsystems perform one I/O operation at a time and therefore require one control unit, one I/O channel, and, in multiprocessor systems, only one SPI. Dual channel subsystems can execute two operations simultaneously using different peripheral units in the subsystem. Both of these operations may originate in the same processor or they may come from different processors.

Different processors can be connected to such a dual-channel subsystem through two SPI Units. Two input/output channels from each Processor or Input/Output Controller take separate paths. The failure of an SPI or one of the pair of Control Units affects only one of the two paths to a peripheral subsystem. Therefore, all peripheral units are still accessible through the second SPI and Control Unit.

2.2.4.2 Multiple Module Access Unit

The Multiple Module Access Units (MMA) allow the sharing of individual storage modules by up to three Central Processors and two Input/Output Controllers on a fixed priority basis.

The MMA recognizes the storage access requests on a priority basis with the lower channel retaining the higher priority. Input/Output Controllers which require higher priority are therefore connected to the lower numbered interfaces. Upon recognition of a storage access request the MMA connects the address lines, the CPU or IOC data lines, and the write control signals to the storage module. The MMA then sends the storage acknowledgement back to the recognized processor and provides the drive necessary to transfer the data to the processor requesting it.

2.2.4.3 Availability Control Unit

Because of the system availability requirement the multiprocessor system must have a means for partitioning the system for specific jobs or for maintenance purposes. The Availability Control Unit (ACU) performs this and related functions as follows:

- Partitions the multiprocessor system hardware into independent systems.
- Takes units off-line for maintenance without disrupting operation of the rest of the system.
- Protects main storage in event of a power failure in the CPU or IOC.
- Automatically initiates a recovery sequence after a failure.

The ACU partitions the hardware into specific configurations by disabling and enabling the interface between units. It can set up as many as three logically independent configurations which run concurrently under control of the Executive system. The possible configurations can be pre-specified for a given site. At the same time the ACU can take units off-line for maintenance.

The ACU is an independent unit with its own power supply logically situated between the peripheral subsystems, the central processors, input/output controllers, and main storage. It can interface with three Central Processors through one I/O channel of each, two IOC's, four banks of main storage, and six multiple-access peripheral subsystems. Additional peripheral subsystems, to a maximum of 24, can be added in groups of six. The ACU includes a control panel, physically located at the operator's console, that indicates all partitioning currently in effect and also shows which units are off line. It also has manual controls to switch units on or off line.

The automatic recovery sequence is based on a resettable system timer in the ACU. The period of this timer can be set to times varying from one to fifteen seconds. Unless the Executive system resets this timer within its period, the ACU assumes that a catastrophic malfunction has occurred and it initiates an automatic recovery sequence. The processor can then interrogate the ACU to determine which units are on line and available for use.

2.2.5 Operator's Control Console

The UNIVAC 1108 Operator's Control Console subsystem is a free-standing input/output device for directing and monitoring the operation of the CPU. A multiprocessor configuration includes one console subsystem for each CPU. The various activities of the system can be apportioned among the available consoles so that the total system will be utilized to best advantage. The console is always connected to input/output channel 15.

The basic Control Console includes the following components:

- Keyboard and CRT Display

  The keyboard and CRT display enable the operator to monitor the performance of the system. The keyboard is a standard four-bank keyboard which can generate 63 Fieldata codes. A row of eight interrupt keys is located immediately above it. The CRT can display 16 lines of 64 characters each.

- UNIVAC Pagewriter

  The UNIVAC Pagewriter provides a hard copy of all messages for a permanent record of all completed transactions between the operator and the Executive system. The Pagewriter prints lines of up to 80 characters each at a rate of 25 characters per second.

- Day Clock

  The day clock on the console displays the time of day in hours, minutes, and hundredths of minutes. It furnishes the time of day to the CPU every 600 milliseconds and sends a day clock interrupt signal to the CPU every 6 seconds. The day clock may be manually disabled from the operator's console.

  On a multiprocessor system any one day clock may be selected to be active either externally or by program.

3.0 MAIN STORAGE

3.1 General

The main storage of the UNIVAC 1108 Multiprocessor System is a high performance, fast access repository for instructions and data. Its design fully supports the concepts of multiprogramming, multiprocessing, modularity, and reliability around which the entire UNIVAC 1108 Multiprocessor System is constructed. Among its featured characteristics are:

- 750 nanosecond read/restore cycle time
- 65,536, 131,072, 196,608, or 262,144 thirty-six bit words
- Parity checking on all storage references
- Access by up to three processors and two IOC's
- Modular expansion two, four, six or eight 32,768-word modules (one, two, three, or four 65,536-word module paris or banks)
- Hardware storage protection - lockout boundaries establishable in 512-word increments
- Relative addressing and dynamic program relocatability through program base registers
- On-line serviceability - module pairs may be removed for servicing without stopping the entire system
- Overlapping/interleaved main storage access to boost processor performance and to minimize access conflicts among processors

While these features are all discussed generally as storage features, many of them such as relative addressing, storage protection, overlapping/interleaving are actually functions of each processor. With proper multiprocessor system organization, the main storage then becomes a set of components of the system which are allocatable in the same manner as peripheral devices. In realizing this objective, the design departs from the traditional close integration of the processor and the storage elements in the following ways:

- The main storage is composed of independently accessible modules, yet it presents a continuous addressing structure to the processors.
- In order to service more than one processor, a method of establishing priority among processors (CPU's and IOC's) at each module is provided in case two or more processors attempt to reference the same module simultaneously.

- To ensure that a processor will wait for access to storage, the processor and the module communicate on a request/acknowledge basis.

With these considerations in mind, the storage modules (via the MMA) become passive components which perform the following functions:

- Grant storage access to a number of processors on a priority basis
- Accept an address from any processor
- Store or retrieve a word at that address
- Issue an acknowledgement signifying that a storage reference has been completed
- Check parity on all data and deliver an interrupt signal to the processor requesting access should a parity error occur.

This processor module relationship has significant advantages for the immediate as well as the future needs of the system. Addition of processors or banks of storage is simplified. It becomes a simple matter to add processors or storage elements, or to replace them with improved equipment, module by module, as technology advances.

3.2 Storage Module

The basic storage module includes 32,768 words of ferrite core array. Each word is 36 bits long, and carries two additional parity bits in nonaddressable levels, one bit for each half word. The main components of the module are a 15-bit address register, a 36-bit read/restore register, parity checking circuits, and request/acknowlege circuits.

The 15-bit address register of each storage module provides addressing for 32,768 words. Since an 18-bit address is generated within the processor at each storage reference, three bits are available for selection of one of the eight possible storage modules.

Parity is checked on reading or calculated on writing for each storage access. If a parity error is detected, the storage bank sends a parity error interrupt signal to the processor which it is currently serving, and rewrites the word in its incorrect form to ensure subsequent data errors when the word is again referenced. Preservation of the error in this way facilitates fault location, since the Executive can determine whether the failure is transient or is associated with a marginal or complete failure of the module.

3.3 Multiple Module Access Unit

In a multiprocessor system, an MMA unit is connected between each pair of main storage modules and the processors which may reference it. The MMA unit furnishes five priority-ordered processor connection paths (one for

each CPU and one for each IOC) to each of the modules of the pair. Should an access conflict occur among processors, the MMA grants storage access to the processor having the highest priority, then the next, and so on. Communications between a processor and a single storage module can, therefore, be asynchronous; if the storage module is busy servicing one processor, a passive wait cycle is induced in others of lower priority that may be referencing it. Because a delay in honoring an input/output transfer can result in an undesirable "go-around" on drum, reread or rewrite on tape, or actual loss of data in the case of real-time input, I/O Controllers are ordinarily attached to the higher priority inputs of the MMA, followed by CPU's, which have built-in precedence of I/O over computational activities.

3.4 Packaging

Two 32,768-word storage modules (module pair) within a single cabinet constitute a bank. An adjacent cabinet contains DC power supplies for operation of the bank and the associated MMA.

3.5 Storage Capacity

Available storage capacity ranges from 65,536 words to the system maximum of 262,144 words, in steps of 65,536 words, according to the following:

65,536 words (two modules) - Minimum for unit processor system

131,072 words (four modules) - Minimum for multiprocessor system

196,608 words (six modules)

262,144 words (eight modules)

3.6 Addressing

Two special techniques for referencing the main storage modules are used to increase processor performance and to reduce the occurrence of multiprocessor access conflicts. The first, called overlapping, enables the CPU to retrieve the current operand and the nest instruction simultaneously; the second, called interleaving, enables two processors (CPU's, IOC's, or CPU and IOC) to access a pair of modules with minimum access conflicts.

3.6.1 Overlapping

The CPU can determine whether its current operand and next instruction lie in different storage modules, and if they do it retrieves the two words in parallel, at an effective 100% performance increase.

The overlapping feature permits the separation of the instruction data of a program into separate physical banks. Furthermore, the base register of the CPU allows either the instruction or data area of a program to be relocated independently - a significant advantage in storage compacting to overcome program fragmentation.

3.6.2 Interleaving

Interleaving is a method of addressing a main storage module pair (bank) in an even/odd fashion. It significantly reduces storage access conflicts in a multiprocessor system, and thereby increases overall system performance. With interleaving, one module of a pair contains all even numbered locations and the other contains all odd numbered locations. Thus, in a fully-expanded eight module system, modules 0, 2, 4, 6 are referenced for even addresses while modules 1, 3, 5, 7 are referenced for odd. The even/odd module pairs consist of modules 0 and 1, 2 and 3, 4 and 5, and 6 and 7.

For a practical example, substitute the letters A, B, C, D for the modules contained in two banks, and assume data are being stored sequentially by a program. (The same assumption may be made for instructions being executed sequentially by the same program.) With the overlapping feature, assume processor number 1 starts executing instructions and retrieving data, with the instruction area in bank 1 and the data area in bank 2. For simplicity, assume the starting instruction and data addresses are at even numbered locations. The processor will then reference module A-B-A-B... for sequential instructions, and C-D-C-D... for sequential data locations. In any single storage interval, either modules A-C or B-D will be busy while their alternates will be idle. If another processor starts an identical process, but referencing an odd address to begin with, both processors may run concurrently without one impeding the operation of the other.

Assuming that both processors in the above example started at even addresses, the processor with lower priority passively waits one storage cycle after which the two are again in synchronization and may operate simultaneously.

3.7 Storage Protection

To prevent inadvertent program reference to out-of-range storage addresses, the 1108 Processor includes a hardware storage protection feature. The controlling element in this feature is the Storage Limits Register.

The Storage Limits Resister (SLR) can be loaded by the Executive system to establish allowable operating areas for the program currently in execution. These areas are termed the program instruction (I) and data (D) areas. Before control is given to a particular program, the Executive loads the SLR with the appropriate I and D boundaries.

Before each main storage reference, the processor performs a limits check on the address, comparing against the limits of either the I or D field of the SLR. An out- of-limits address generates a guard mode interrupt, thereby allowing the Executive to regain control and take appropriate action.

3.7.1 Storage Protection Modes

The Executive system can establish two different modes of storage protection by means of control fields in the Processor State Register (PSR). Normally, the Executive itself operates in open mode; that is, the Storage Limits Register may be loaded but the PSR is set to disregard this, and the Executive can reference any location in main storage.

3.7.1.1 Privileged Mode

Another mode can be established in the PSR for privileged programs. This privileged mode protects against out-of-bounds writes. Privileged programs (such as real-time programs or Executive-controlled subroutines) may enter non-alterable (re-entrant) subroutines, which are part of the Executive. Though these privileged programs are assumed to be thoroughly checked out, the system is still fully protected against unexpected occurrences since write protection is in effect.

3.7.1.2 User Program Mode

In the user program mode, read, write, and jump storage protection is in effect. Therefore, user programs are limited to those areas assigned by the Executive. If the user program reads, writes, or jumps to an out-of-limits address, an interrupt returns control to the Executive for remedial action.

Read/jump protection allows the Executive to stop the program at the point of error, terminate it, and provide diagnostic information to the programmer thereby minimizing wasted time and smoothing the checkout process.

A particular advantage of read/jump protection is that classified (confidential) programs can be confidently run together; they are fully protected from audit (inadvertent or otherwise) by other programs.

3.8 Relative Addressing

Relative addressing is a feature of great significance in multiprogramming, time-sharing, and real-time operations, for it allows storage assignments for one program (the one going into execution) to be changed dynamically by the Executive to provide continuous storage for operation of another program, and it permits programs to dynamically request additional main storage according to processing needs. An additional advantage is that systems programs stored in auxiliary storage may be brought in for operation in any available area without complicated relocation algorithms.

Relative addressing is provided for through base registers contained within the CPU. Two separate registers control the basing of the program instruction and data bank, and a third register controls the selection of the appropriate base register.

4.0 PERIPHERAL SUBSYSTEMS

4.1 Available Peripheral Equipment

Peripheral subsystems are attached to the 1108 processor or to the independent I/O controller through general purpose input/output channels, which have no restriction as to the manner in which peripheral subsystems may be attached. The governing factor for peripheral attachment is the transfer rate of the devices in the subsystem. Since the channels are numbered in order or priority, real time equipment or equipment with very high transfer rates should be attached to the lower numbered channels which have the higher priority.

With this adaptable input/output arrangement, the UNIVAC 1108 System can communicate with many real time devices such as analog/digital converters, key sets, communication terminals, tracking and radar systems, display systems, and other information processing systems.

4.2 FH-1782 Magnetic Drum Subsystem

The Flying Head 1782 (FH-1782) magnetic drum is identical to the FH-880 drum used on the UNIVAC 1107 Thin-Film Memory Computer except that the storage capacity is 2-2/3 times greater; this increase is achieved partly by an increase in the number of data tracks to 1,536 and partly by an increase in the recording density. Each track has its own read/write head, and average access time is unchanged at 17 milliseconds.

A single FH-1782 drum stores 2,097,152 words, equivalent to 12,582,912 characters. Up to eight FH-1782 drums can be accommodated in a single subsystem giving a subsystem capacity of 100,663,296 characters.

4.2.1 Uniservo VIII C Magnetic Tape Subsystem

A Uniservo VIII C subsystem can have up to 16 magnetic tape units, and can incorporate either one or two Control Units attached to one or two input/output channels for single or dual channel operator.

Uniservo VIII C Units may be specified with seven- or nice-track mode. In seven-track mode one parity and six data bits are recorded in each frame across the width of the tape. A single 6-bit alphanumeric character, or a 6-bit binary value may be stored per frame. In nine-track mode one parity and eight data bits are recorded in each frame across the width of the tape.

Data packing density is set either by the program or by a manual switch on each unit to either 200, 556 or 800 frames per inch. Physical tape speed is 120 inches per second giving maximum transfer rate of 24,000, 66,720 and 96,000 characters per second (in seven-track mode), or bytes per second (in nine-track mode).

Even higher transfer rates result if 6-bit characters are read or written in nine-track mode. This method of operation yields transfer rates of 32,000, 88,960, and 128,000 characters per second.

The rewinding rate is 240 inches per second; a full reel of 2,400 feet can be rewound in 120 seconds. The 800 frame per inch packing density will normally be used, the 200 and 556 densities being used only for compatibility purposes.

Reading may take place with the tape moving either forward or backward, an ability valuable for saving rewind time especially during sort/merge operations. Writing takes place when the tape is moving forward only.

Data may be recorded in variable-length blocks under program control, with character and block (horizontal and vertical) parity. A read-after-write head allows immediate verification of all data written. Under the control of the software Input/ Output Handler, repeated read and write operations are undertaken in an attempt to recover from an error.

Programming problems with this tape subsystem are insignificant since the Control Unit, combined with the Executive Input/Output Handler, deals with all operations except the system response to a nonrecoverable error.

Uniservo VIII C Magnetic Tape Units are fully compatible with IBM 727, 729 Mod I through VI, and 7330 units in seven-track mode, and with IBM 2400 series Models I through 3 units in nine-track mode, and with industry-compatible units produced by other manufacturers. The Uniservo VIII C control unit can be furnished with a hardware translator to convert between tape code and Fieldata code, thus ensuring tape compatibility among installations.

Two different Uniservo VIII C tape subsystems are available: the Uniservo VIII C/VI C subsystem and the fully simultaneous Uniservo VIII C subsystem.

Uniscope 300 Visual Communication Terminal Subsystem

The UNIVAC Uniscope 300 Terminal is designed for applications requiring direct user interaction with the central system. These Terminals can be located at or near the site, or they may be operated as remote stations over standard data communication facilities.

The Uniscope Terminal itself includes a keyboard and CRT display, a core storage unit to store data as it is typed or received and displayed on the CRT, and control circuits. As the operator types a message on the keyboard, the data is accumulated in the storage unit and displayed on the CRT with a cursor marking the location of the next character to be typed. He can then make changes before releasing the message for transmission to the computer. Messages from the computer are similarly displayed and can be altered and returned to the computer by the operator.

4.3.1 The Keyboard

The keyboard includes alphanumeric keys, cursor controls and editing keys, and function keys. The alphanumeric section is similar to the standard electric typewriter in layout and operation. The character set has 56 symbols ordinarily, but up to 96 characters are available. The cursor controls can set the cursor to any point on the screen, so that deletions and changes can be made while composing or editing messages. Up to 40 different function keys can be added to the keyboard. The meanings of 35 of these function keys can be varied by means of overlays that fit over the group. The overlays are cards each of which has an edge formed according to an identifying code; when an overlay is in position, the coded edge causes the operation of some combination of seven switches controlling the significance of the function keys. On the face of the overlay, and appearing adjacent to the function keys when the overlay is in place, are markings to indicate the corresponding use of each key. One hundred and twenty-two different overlays can be used, enabling representation of as many as 4000 different functions. Typically, the different overlays can be used to identify stations, operators, applications, or security requirements. In addition to initiating a function, each function key displays a unique symbol on the CRT.

4.3.2 CRT Display

Sixteen lines of 64 characters each can be displayed on the 5 inch by 10 inch screen. The data is actually stored in a 1024 character core storage unit and is regenerated on the screen 60 times per second.

4.3.3 Subsystem Configurations

The subsystem can be used in single- and multi-station configurations.

The single-station terminal is completely self-contained. It interfaces dierectly with the data communications equipment through a standard EIA interface. Transmission rate of the line must be greater than 2000 characters per second. It can be used individually in a private line circuit or a number of them can be connected to a multi-point parity line. In this latter arrangement the unit responds to a poll code from the central system.

The multi-station configuration provides a more economical arrangement where a large number of terminals is required. This configuration requires a multi-station control unit. The control unit is modular; it provides I/O message buffering, character generation, and control logic for up to 24 terminals of 1024-digit capacity or up to 48 terminals if 512-digit capacity. In this configuration each terminal is completely independent of all others. The terminals can be located as much as 1500 feet from the control unit.

Optionally the control unit can be equipped to communicate over two separate lines. This makes possible independent, concurrent communication (input and/or output) over the two lines; or, if desired; one line can be reserved as backup.

In another configuration two control units can be connected to the same set of terminals with one unit switched on and the other off to provide a backup control unit. The units can be switched on or off manually or by programming. By this means, one extra control unit can serve as a spare for as many as 48 sets of displays.

Polling techniques allow single-station and multi-station units to be mixed freely on one multipoint party line. This capability increases line utilization and significantly decreases line costs.

4.3.4 Reliability

In both single- and multi-station configurations the subsystem checks the character parity and message parity of incoming messages, and generates them for outgoing messages. Erroneous messages are retransmitted automatically on request. A simple message acknowledgement system ensures that no messages are lost or duplicated.

To further improve reliability, the central system can perform marginal tests on the storage unit.

4.3.5 Special Features

Several special features of the subsystem contribute especially to its use as an on-line device.

A pair of special function codes enable the program to insert or delete a line of copy. The insert function moves the line marked by the cursor and all lines below it down one line, the bottom line moving off the screen. It then inserts the new text. The delete function erases the line marked by the cursor and all lines move up one line, leaving the bottom line blank. By means of these functions the program can rearrange data on the screen with a minimum of new transmission.

Of particular interest is the "roll and scroll" technique implemented by use of the insert and delete functions. In this case text on the screen appears to roll up or down with the screen remaining filled. Typically, this is useful for scanning through large tables or other lengthy text that is too large for the screen. As soon as the required part of the table is located, the operator can stop the rolling by means of a function key.

The capability of split-screen operation also increases the versatility of the terminal. This makes possible simultaneous, independent display of several messages. In this case an end-of-field symbol designates the end of each separate message. Line insert and delete functions can then be used on the messages separately and each one can be rolled and scrolled without interfering with the others.

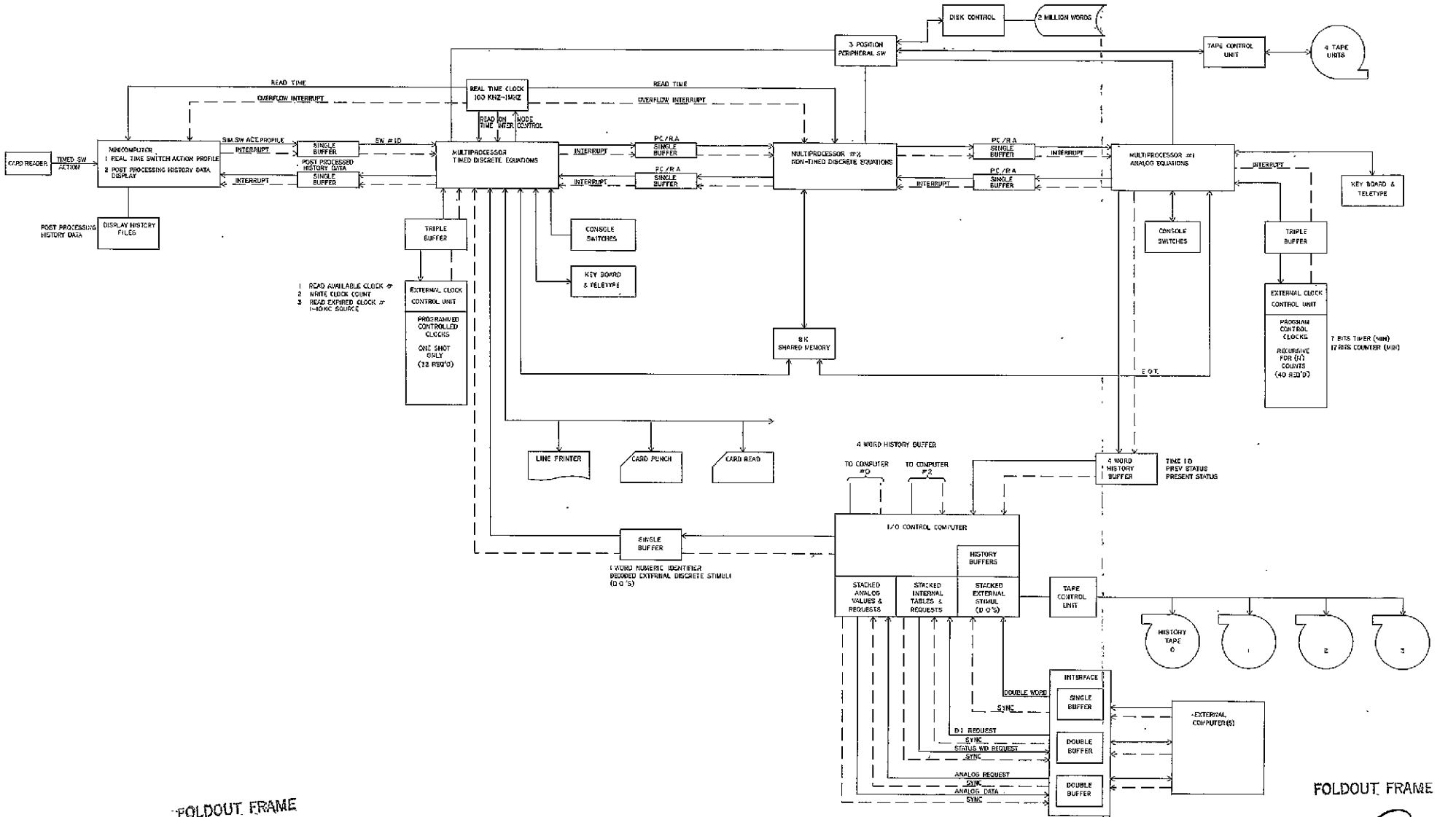
For emphasis, specific messages or parts of messages from the system can be programmed to blink. In addition, unsolicited messages from the central system are accompanied by an audible signal. Such a message can be programmed for immediate display, overriding other output, or it may be withheld pending operator response.

4.4 Program Controlled Clocks

The EXEC VIII Real-Time Clock routine is used by the system for timing various activities such as input/output functions, operator responses, CPU usage time for each run, etc. The Timing Routine makes provision for its use by an object program as well as by the system. The routine is so designed that many events may be simultaneously timed, and many interrupts may be simultaneously requested. Since only one Real-Time Clock and one clock interrupt are available on the 1108, the Real-Time Clock Routine essentially acts as a multiplexor and creates an environment such that any routine may operate as though it had exclusive access to the clock and to the associated interrupt. The Real-Time Clock Routine is available to the user by means of Executive Requests.

Under EXEC VIII an Activity is defined as a division of a program which may be executed independently of other portions of the program. It is possible that a program may contain many activities which may be executed asynchronously. The following coding contains an example of how a user program may utilize the executive timing routine. COMPUTE is to be executed every five milliseconds.

```
          AXR$
          ____
          ____
          L,U       A1,5          Load A1 with 5
          S         A1, COCK1     Store 5 in CLOCK1
          L,U       A0,COMPUTE    Load address of compute in A0.
          ER        FORK$         Activate compute
          ____
          ____
COMPUTE   L         A3,VALUE      Load A3 with VALUE
          A         A3,VALUEX     Add VALUEX to VALUE
          S         A3,VALUE      Store new VALUE
          L         A1,CLOCK1     Load A1 with VALUE of CLOCK
          ER        TWAIT$        Timed waid of 5 milliseconds
          J         COMPUTE       Jump to COMPUTE
```

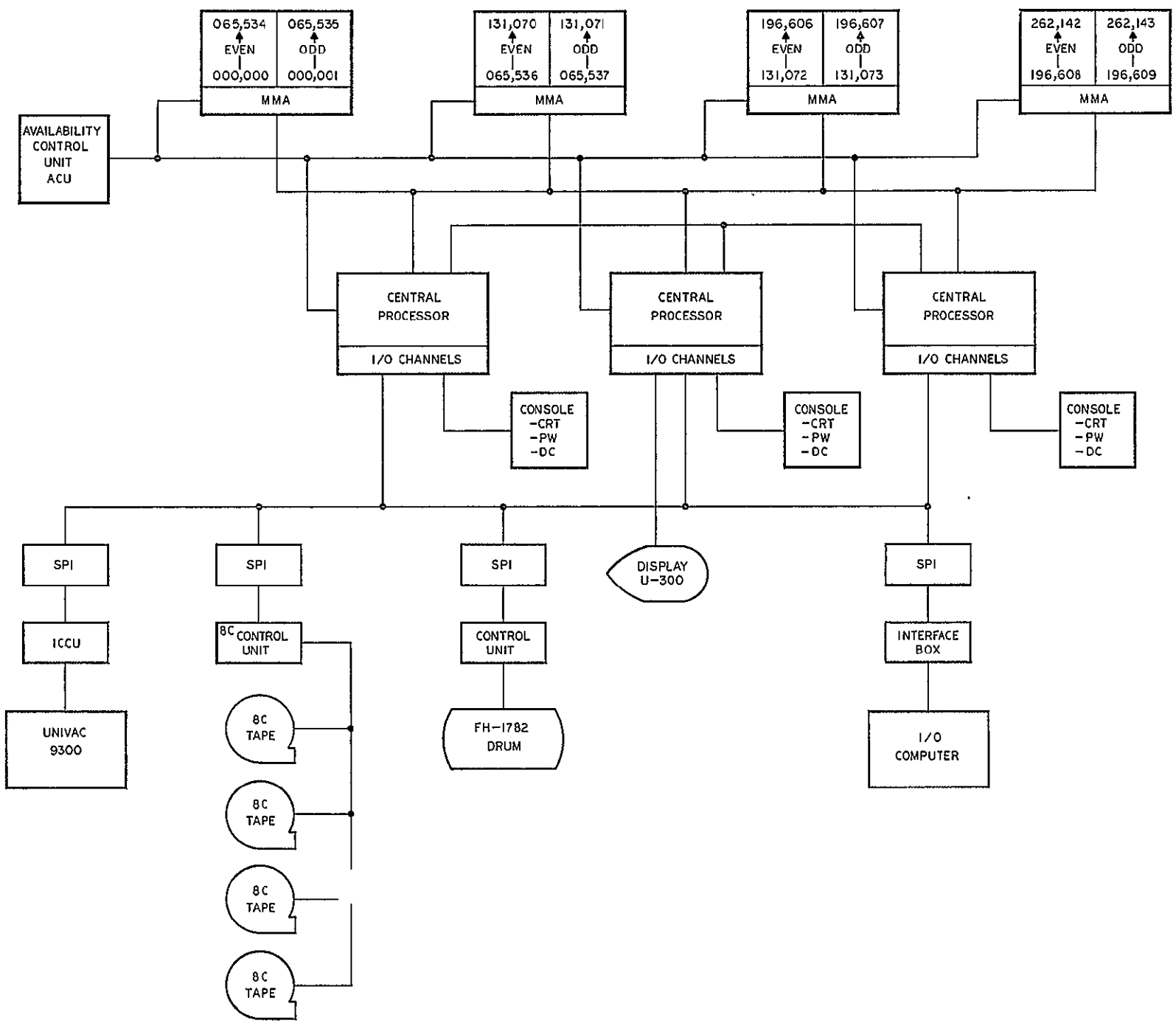


FOLDOUT FRAME

FOLDOUT FRAME

2

FIGURE I


065,534
EVEN
000,000
065,535
ODD
000,001
MMA
131,070
EVEN
065,536
131,071
ODD
065,537
MMA
196,606
EVEN
131,072
196,607
ODD
131,073
MMA
262,142
EVEN
196,608
262,143
ODD
196,609
MMA
AVAILABILITY CONTROL UNIT ACU
CENTRAL PROCESSOR
I/O CHANNELS
CENTRAL PROCESSOR
I/O CHANNELS
CENTRAL PROCESSOR
I/O CHANNELS
CONSOLE -CRT -PW -DC
CONSOLE -CRT -PW -DC
CONSOLE -CRT -PW -DC
SPI
SPI
SPI
DISPLAY U-300
SPI
ICCU
8C CONTROL UNIT
CONTROL UNIT
INTERFACE BOX
UNIVAC 9300
8C TAPE
8C TAPE
8C TAPE
8C TAPE
FH-1782 DRUM
I/O COMPUTER

Appendix D

Proposal
CDC 6400

Control Data Corporation in studying the shuttle simulation requirements, considered their 6000 series computers as possible candidates. Of this series the 6200, 6400, 6500, 6600 appear to have the computational capability for a complete shuttle simulation.

The 6400 system appears to be the most likely candidate of the 6000 series due to it's computational ability vs. cost. The system in itself is unique in the method the instructions are stacked in 60 bit words. This capability allows the equations to be pre-assembled into machine language for rapid execution. For a more complete description of this concept see the enclosed proposal by CDC.

The 6400 processor has seven Peripheral Processor Units (PPUs) associated with the CPU. In reality these are seven additional small processors and as such may be used for processing equations. This additional computational ability should improve the equation response time appreciably.

Two of the PPUs would be dedicated for timed equations (analog and timed discrete) and would eliminate the need for hardware clocks. Each PPU would be capable of maintaining 90 timed events for a total of 180 events. The clock simulation would take place simultaneously with the CPU evaluating Boolean equations and would not degrade the equation solution capability while updating the clocks.

Of the CDC systems considered the 6400 was chosen as the best processor configuration for simulating all subsystems of the shuttle. The cost of this simulator including the necessary peripherals was estimated to be $1,800,000. Of the large processors considered (See Appendices C through E) this is the least expensive.

For a more detailed explanation of the proposed simulator see the enclosed CDC proposal.

INTRODUCTION

Control Data is pleased to submit this solution to the computer requirements of the Shuttle Program. Control Data's approach is built around the 6400 which is a member of Control Data's 6000 Series super computers.

Control Data has thoroughly evaluated the present and future requirements for the Shuttle Program and the results of this evaluation is to favor the large computer approach. Large, medium and small computers were considered in the solution of the Shuttle Program task, however, the large computer concept proved to be extremely beneficial in several areas as follows:

1. Pre-processing - the magnitude of the pre-processing requirements will necessitate a large computer if it is to be handled expediently. The 6000 Computer is designed for high volume throughput.
2. Cost Performance - because of the speeds required, the 6000 provides the best cost performance and a much less complicated network of computer systems.
3. Expandability - the initial configuration will adequately handle the initial Shuttle requirements. The addition of peripheral processors, central processors, mass storage devices and Extended Core Storage will enable the initial configuration to be further enhanced handle future simulation requirements.

4. Reliability - the 6000 Series is extremely reliable and has been field proven since its announcement in 1963.

5. Flexibility - the 6000 Series offers flexibility by providing computing power necessary to conduct Shuttle Program simulation tasks as well as other batch processing.

6. Design Concepts - the 6000 is designed so that it is modular in nature thus providing ease of expansion. Speed - the proposed 6000 can adequately handle the Boolean and analog equations in the required time limit. This time can be further enhanced by adding additional peripheral devices.

8. Elimination of Clocks - the proposed 6000 system will enable the Shuttle Program to eliminate the need for clocks. These clock functions will be simulated by one or more peripheral processors. Interrupts to the 6000 will be created by the peripheral processor monitoring programs. The elimination of these clocks is in itself a unique advantage for the 6000 computer concept.

The attached document describes Control Data's approach for simulation of the Shuttle Program. Please contact us if additional information is desired.

SHUTTLE SYSTEM REQUIREMENTS

Based on discussions with Sperry and Computer Science personnel, certain design requirements influenced Control Data Corporations recommendations.

1. Large volumes of Boolean equations to be solved
2. Large number of real-time clocks necessary to drive system
3. Ability to interface with analog equipment
4. Large core requirements
5. Realization of multiple processors
6. Some volume of history output during simulation
7. Very stringent timing requirement to run in real-time environment

CONTROL DATA'S SOLUTION

Control Data feels that it is uniquely qualified to present a system realistically capable of meeting the above requirements. Our existing line of 6000 computers is ideally suited for the Apollo shuttle simulation.

The 6000 Series consist of the 6200, 6400, 6500, 6600 and 6700 computers.

The 6400 has one central processor with a unified arithmetic unit. The 6500 is identical to the 6400 except that it has two independent central processors, each having a unified arithmetic unit. The 6600 again has a single processor but has an arithmetic unit consisting of ten independent functional units. This allows

the 6600 to simultaneously execute up to ten instructions, thus achieving greater CPU speed. In addition, the 6600 has a high speed instruction stack that is capable of holding up to 27 instructions each of which can be recalled without a reference to central memory.

A 6400 with 49K core and 125K of Extended Core Storage (ECS) was found to possess the computing capability necessary for a shuttle simulation.

A unique feature of the 6000 Series Computer System is its isolation of the central processor from peripheral and "overhead" operations. This is accomplished through the use of a minimum of seven peripheral processor units (PPU's). Control of the entire computer system can reside in a PPU, thus freeing the CPU for computation.

In the shuttle simulation we have made full use of the PPU to handle the data manipulation and actually perform independent simulations necessary to the overall system.

COMPATIBILITY AND EXPANSION

In addition to the present existence of the 6000 computer systems, it is important to recognize the inherent compatibility of the series. This is to say that all 6000 computers from the 6200 through 6700 are software compatible. When a new operating system is released, it is for the entire series. Likewise, application programs capable of running one 6000 will run on all 6000's.

Equally important is the expansion capability within the 6000 series. The 6200 is field upgradable (no exchange of CPU) to a 6400, 6500 or 6700. Due to the uniqueness of the 6600 arithmetic unit, it requires an exchange of CPU. However, as mentioned above, the 6600 is software compatible with the remaining 6000 series. Increased throughput performance is achieved as one progresses upward from the 6200 through the 6700.

The 6400 recommended for the shuttle simulation would have seven of the above mentioned PPU's. This number can be increased to eight, nine, ten or twenty should the need arise. Also the 49K core and 125K ECS, both with 60 bit words, are expandable to 131K and 2 million words respectively.

PROPOSED SIMULATION TECHNIQUES

Boolean Equations

For the purpose of this proposal, those areas most critical with regard to time and volume have been emphasized. The most critical area appears to be the large Boolean equations, both in size and number, to be evaluated. Due to this volume and complexity, it has been determined that these equations should be pre-assembled into executable instructions and stored on ECS.

To accomplish this task, the status bits tables, once sorted during pre-processing, would be assigned to core locations. The addresses of these bits would then be available to a pre-assembler to use in constructing executable code. This code, as stated, would be stored on ECS. Thus when an equation was to be solved its code could be loaded into core and executed.

An example of this technique is as follows:

The equation

$$V100 = S100 * V200 + S101$$

would be pre-assembled into the following general form:

1. Load S100 status {address of S100 would be pre-assembled into the load instruction}
2. Save S100 status temporarily
3. Load V200 status
4. AND S100 and V200
5. Save result
6. Load S101
7. OR S101 with result of step 4
8. Store result in V100 status

Due to the packing of instructions on the 6000 computers, these instructions could be held in only three words of core during execution.

The idea of pre-assembly would have to be extended to allow the necessary equations access to status bits for modification when required.

To further speed the solution of Boolean equation, a PPU would be assigned to overlap the loading of required code while the previously loaded equation was being executed.

To summarize this technique, the pre-assembly procedure should be carried to the extent necessary to eliminate unpacking, masking and table searching during the real-time simulation. To optimize execution further, it will be necessary to store each variable's status in a single word. On the Saturn this would require 22,175 words; even if double on Apollo, only 40,000 words would be required. The memory required to load executable

equations from ECS should not excede 300 words for a 700 variable equation. This would leave ample space for buffering equations from ECS. With these figures in mind, a 49K central memory would be sufficient.

Using the described pre-assembly procedure, a 700 term equation could be evaluated in from one to one and a half milliseconds on the 6400. If the equations were unpacked and the status located in a table, a 700 variable equation would require three to four milliseconds. These figures have been substantuated by actual Boolean evaluations on a CDC 6400.

A 6600 would reduce the evaluation time to .5 milliseconds for a 700 variable equation.

Clock Simulation

The large number of clocks necessary to drive the simulation is another critical area. Control Data is in a unique position in that it can use peripheral processors (PPU's) to simulate these clocks. Rather than build 120-180 clocks and an interface to make them computer accessible, 90 clocks can be simulated on a single PPU. In this incident, two PPU's would be required.

Every millisecond, the PPU would decrement a location representing a clock. After each decrement, the location would be tested for zero. If zero, an associated location would be flagged to indicate to the CPU that a certain Boolean equation was to be evaluated. This clock simulation would take place simultaneously with the CPU evaluating Boolean equations.

The table to indicate which equation to evaluate would be in the CPU and would contain an address for retrieving the equation from ECS.

ANALOG EQUATIONS

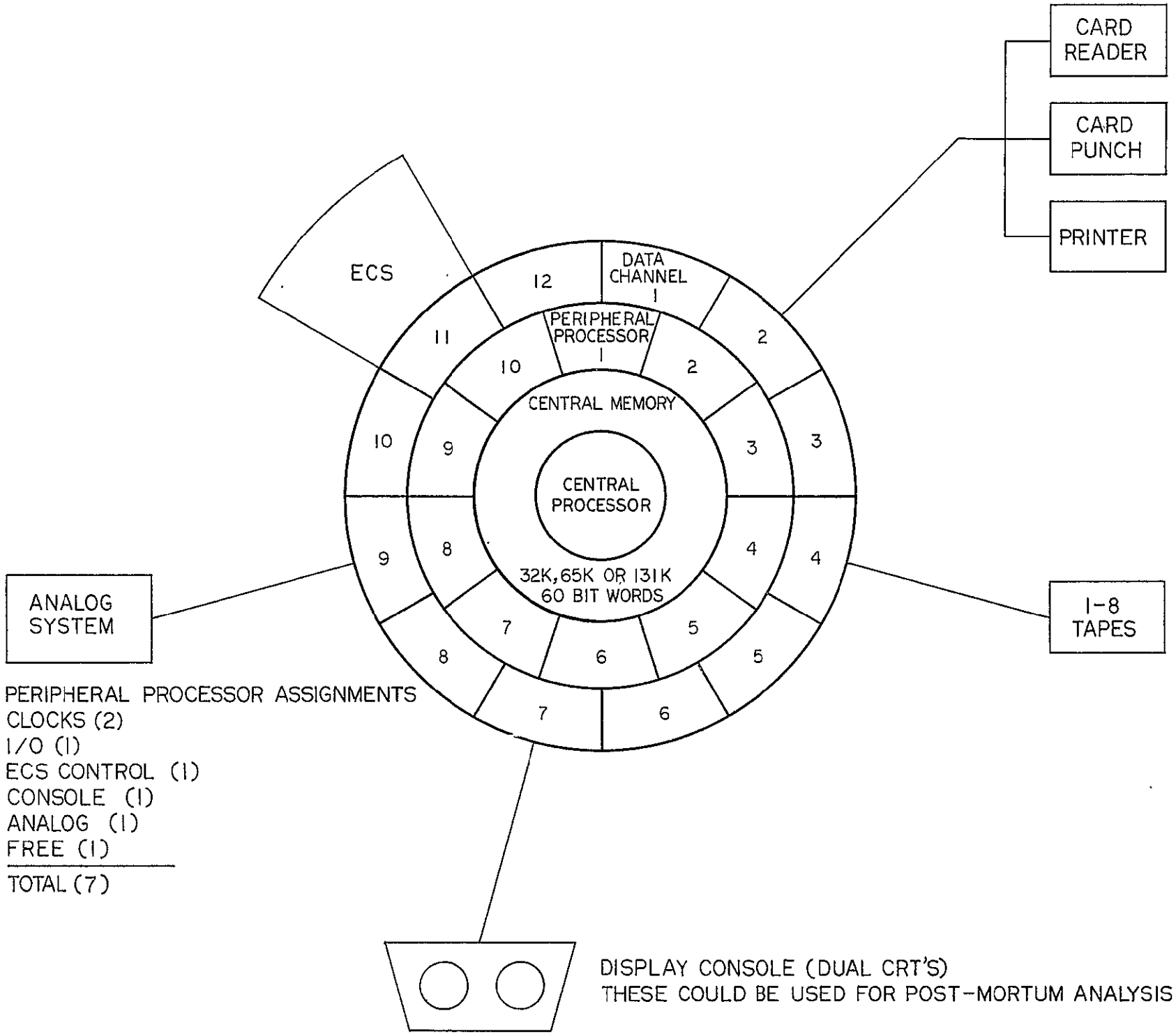
In this area of the simulation, two directs or a combination of the two are possible.

If practical, a PPU or multiple PPU's would be used to simulate or provide analog values. At present it is not known if the PPU's can handle this task. Should the PPU approach not be feasible, the analog equations would simply be moved to the CPU. Here they could be updated on some time interval.

ANALOG SYSTEM

The idea of hybrid computing is not new to the 6000 or to Control Data. In fact the 6400 has been used to build the world's largest hybrid computer system.

Interfaces exist and can operate under the control of a PPU, thus keeping the CPU free for computation as previously explained.



PERIPHERAL PROCESSOR ASSIGNMENTS
CLOCKS (2)
I/O (1)
ECS CONTROL (1)
CONSOLE (1)
ANALOG (1)
FREE (1)

TOTAL (7)

Appendix E

Proposal
SEL 88 Multiprocessor

Systems Engineering Laboratories has proposed 3 Systems 88 processors for simulating the Shuttle with a Systems 86 processor as an I/O computer. The proposed system includes the necessary clocks to simulate timed equations (analog and timed discrete) and the necessary peripheral equipment for preprocessing, data logging, and post simulation analysis.

The system proposed includes an extended memory capability and may be reduced in size to meet the Shuttle data base requirements. The cost of the peripherals may also be reduced by providing a 4-way peripheral switch.

The system as proposed should be adequate for simulating all Shuttle subsystems and the I/O computer should be capable of interfacing with the data bus for subsystem checkout.

The system proposed by SEL was estimated to cost $2,555,750. This price could be reduced however by limiting the memory size and reducing the amount of peripheral equipment. This combined reduction should bring the total cost to less than $2,000,000. It is anticipated that the proposed simulator would perform comparibly with the other large processors proposed.

For a more detailed description of the SEL simulator see the enclosed figure and SEL work statement.

WORK STATEMENT

Item	Model	Description	Qty.	Unit	Total
1	8800	SYSTEMS 88 Central Processor	3	61,000	183,000
2	8810	System Integrity Features	3	2,500	7,500
3	8820	Floating Point Unit	3	25,000	75,000
4	8850	Highest 6 External Priority Interrupts	3	600	1,800
5	8651	Priority Interrupt Module (16 Levels)	3	3,600	10,800
6	8836	Core Memory Module 8,192 Words (32 Bits) with Page Protect and Byte Parity, 4 Ports	46	34,500	1,587,000
7	8600	SYSTEMS 86 Central Processor	1	55,000	55,000
8	8610	System Integrity Features	1	2,500	2,500
9	8650	Highest 6 External Priority Interrupts	1	600	600
10	8651	Priority Interrupt Module (16 Levels)	1	3,600	3,600
11	8631	Core Memory Module 8,192 Words (32 Bit), One Port, Page Protect, Byte Parity	.5	30,500	152,500
12	N/A	CPU to CPU Links	7	9,000	63,000
13	8871	Internal Timer	1	2,500	2,500
14	N/A	Program Controlled Clock Counter	1	6,000	6,000
15	N/A	High Speed Parallel DCC	2	5,000	10,000
16	6519	Mag Tape Controller	2	15,000	30,000

WORK STATEMENT (Continued)

Item	Model	Description	Qty.	Unit	Total
17	6512	Mag Tape Drive	8	20,000	160,000
18	6411	Moving Head Disc	1	39,000	39,000
19	6362	600 LPM Line Printer	1	38,000	38,000
20	6221	Card Reader/Punch 500/100 CPM	1	34,000	34,000
21	N/A	Peripheral Switch	1	25,000	25,000
22	81-000B	810B Control Processor with 8K Words (16 Bit) Memory and ASR-33 Console	1	33,500	33,500
23	81-471-01A	Card Reader, 200 CPM	1	6,000	6,000
24	80-816	Display Console	1	25,950	25,950
25	810230B	Block Transfer Control	1	1,500	1,500
26	81-235B	Computer Graphics Processor	1	2,000	2,000
					$2,555,750

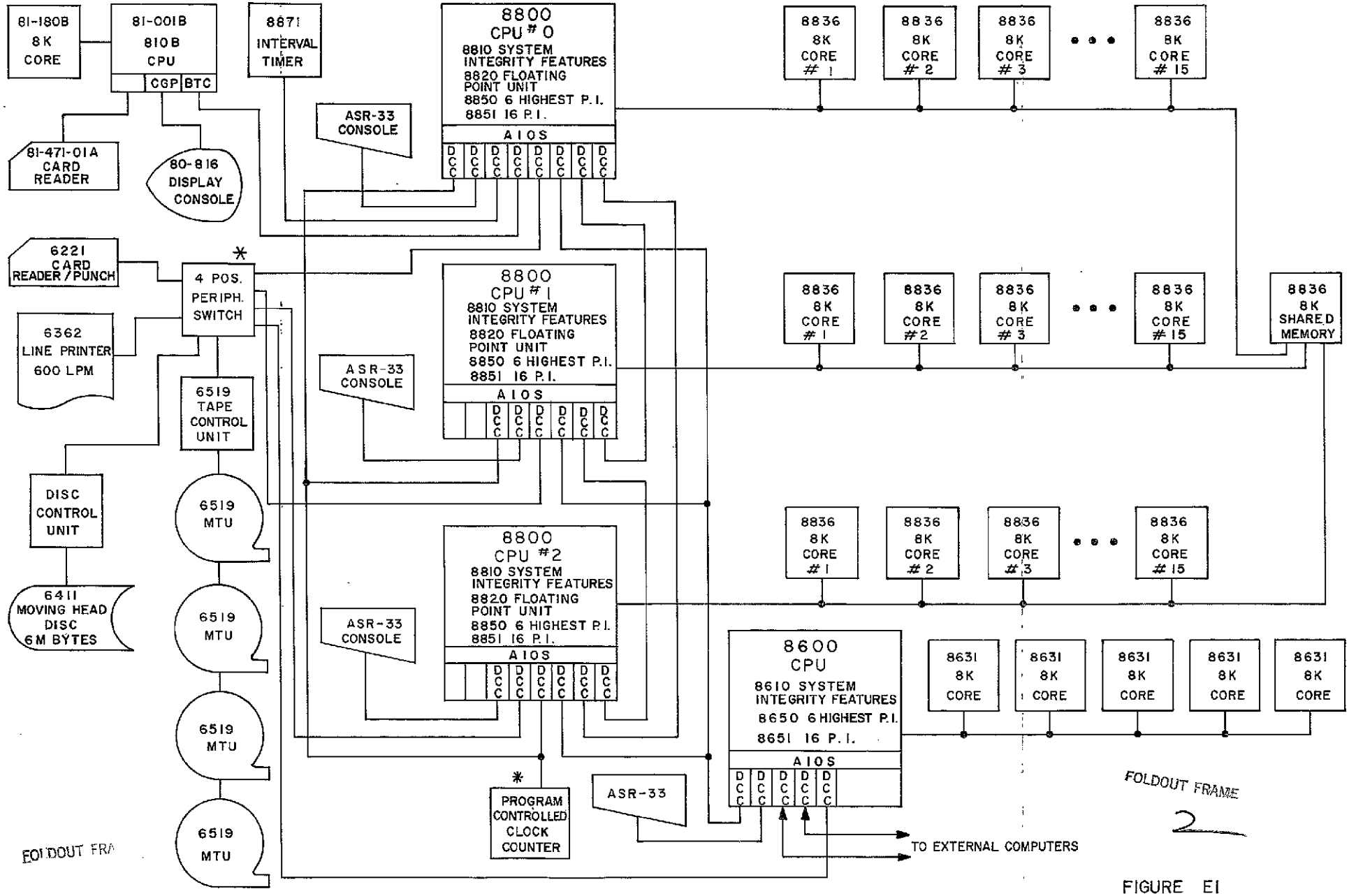

81-180B 8K CORE
81-001B 810B CPU
CGP
BTC
8871 INTERVAL TIMER
8800 CPU # 0
8810 SYSTEM INTEGRITY FEATURES
8820 FLOATING POINT UNIT
8850 6 HIGHEST P.I.
8851 16 P.I.
AIOS
DCC
8836 8K CORE # 1
8836 8K CORE # 2
8836 8K CORE # 3
8836 8K CORE # 15
ASR-33 CONSOLE
81-471-01A CARD READER
80-816 DISPLAY CONSOLE
6221 CARD READER / PUNCH
4 POS. PERIPH. SWITCH
8800 CPU # 1
8836 8K SHARED MEMORY
6362 LINE PRINTER 600 LPM
6519 TAPE CONTROL UNIT
DISC CONTROL UNIT
6519 MTU
8800 CPU #2
6411 MOVING HEAD DISC 6M BYTES
8600 CPU
8610 SYSTEM INTEGRITY FEATURES
8650 6 HIGHEST P.I.
8651 16 P.I.
8631 8K CORE
PROGRAM CONTROLLED CLOCK COUNTER
TO EXTERNAL COMPUTERS
FOLDOUT FRAME 2
FOLDOUT FRA

FIGURE E1

References

[1] A Conceptial Design of the Space Shuttle Integrated Avionics System, NASA TM X-53987, Fred E. Degesu, February 3, 1970.

[2] A Two-Stage Fixed Wing Space Transportation System. Volume II Preliminary Design, McDonnel Douglas, December 15, 1969.

[3] Transfer Equation Preparation Manual for Launch Vehicle and Ground Support Equipment System Simulator, L. W. Brooks, L. J. Gellman, J. A. Stahley, Prepared for NASA, Marshall Space Flight Center (R-ASTR-ESA), Sperry Rand Corporation, Huntsville, Alabama.

[4] Space Shuttle Final Report, Volume VII, Integrated Electronics, General Dynamics, October 31, 1969.

[5] SEL 840 Plotter Program for Generating a Benson-Lehner 120 Plotter Tape, W. M. McCombs, Prepared for NASA, Marshall Space Flight Center, Sperry Rand Corporation, Huntsville, Alabama.

GENERAL PURPOSE SIMULATOR SYSTEM STUDY
PART I

Report No.: SP-209-0339
Date: April 16, 1970

GENERAL PURPOSE SIMULATOR SYSTEM STUDY PART I

Prepared for
George C. Marshall Space Flight Center
Contract No. NAS8-25608

By
Space Support Division
Division of Sperry Rand Corporation
Huntsville, Alabama

SUBMITTED BY: J. Scoggins
J. Scoggins

AND

J. Taras
J. Taras

APPROVED BY: R. A. Rossler
R. A. Rossler

ABSTRACT

A simulation system has been developed at the Marshall Space Flight Center to enable efficient checkout of programs written for the RCA-110A launch support computers.

Sperry Space Support Division is presently studying the hardware and software configuration of the system. Reports will be written in four parts describing how the simulator could be modified to act as a general purpose simulator for future space missions.

TABLE OF CONTENTS

INTRODUCTION

Marshall Space Flight Center has developed a simulation system that can simulate discrete and analog functions of the Saturn V vehicle, the Electrical Support Equipment (ESE), and Digital Data Acquisition System (DDAS). The simulation of the Instrument Unit portion of the Saturn V vehicle was demonstrated in January 1969.

The potential of such a simulator is readily realizeable as a breadboard simulator for launch program checkout and an inexpensive simulator for launch crew training.

PURPOSE

The purpose of this study is to determine from available documents how the existing hardware and software systems may be used for future space missions. If the magnitude of the math models prove the present computer configuration to be unsuitable for the missions involved alternate computers and hardware configurations will be proposed.

SCOPE

The documents supporting the simulator (see references) indicate that the software is well developed to perform the functions required of a simulator. Unfortunately the limitations of the hardware impose great burdens on the software in the real-time mode of operation. In particular the I/O operations, data commutation, and updating of time interval counters places great constraints on the software and the amount of throughput of the CPU.

In general the various phases of the simulator lend themselves readily adaptable to the transfer equations written for the simulator. The real-time mode of operation however, is tightly interlaced with and unnecessarily constrained by the hardware. It appears that a great amount of relief is needed here to separate the hardware and software functions into their proper configurations.

The proper hardware configuration would allow the processing of many more equations for a given period of time. Increasing the throughput by an appreciable amount would allow the simulation of extemely large math models.

In keeping with the theory that eliminating the individual problems will decrease the magnitude of the overall problem, a study has been made to determine the problems in the real-time mode of operation. The results of this study are described in further sections. Due to the limited amount of documentation available on the software, only functional descriptions are included. More detailed descriptions will be included in future reports as documentation becomes available and the writer becomes more familiar with the existing system.

EXISTING SIMULATOR PROBLEMS (REAL-TIME MODE)

As described earlier the configuration of the hardware is the limiting factor of the existing simulator. An attempt will be made to positively identify these limitations and suggest methods of improving the overall simulator. The reader must have some degree of knowledge of the existing system to appreciate any suggestions offered within this report.

A. General Limitations

1. To upgrade the present simulator to be a truly generalized simulator some modifications must be made to the existing hardware and software. In particular the system must be capable of accepting and processing analog input signals. Such a change could be provided between the computer and external interface but would require modifications to the existing software. If this added capability is deemed necessary, the method by which the added feature is to be implemented should be planned immediately.

2. In general, in processing analog signals the software must be capable of processing both negative and positive signals. Normally all such computations are performed by the arithmetic unit of the CPU and are handled in floating point form. The present simulator uses a special floating point subroutine for a special floating point format and is unable to handle negative numbers.

An additional consideration is the accuracy required of the system to be simulated. Fourteen bits of accuracy for D/A and A/D converters are standard items on simulation systems with ±100 volt reference voltages the rule rather than the exception. Also it is quite possible that telemetry requirements of future space missions might require more accuracy than is presently available from the special floating point subroutines.

Coupled with the above mentioned problems is the added overhead time required for floating point computations by subroutines in the real-time mode of operation. It appears that floating point computations by hardware methods would reduce overhead time in the real-time mode, improve the resolution of analog signals, and increase the throughput of the CPU in general.

B. Specific Limitations

1. Memory - Due to the limited size of the core memory only a small portion of the data base may be resident in memory at a given time. To process a given equation several steps are required before the solution to the equation is obtained; all of which contribute heavily to the overhead time of the real-time mode of operation.

The driver portion of the real-time program must search core to see if the equation is resident in memory, this in itself requiring a great deal of time.

If the equation is not resident in memory the disc retrieval routine is called loading the equation into a general buffer in core.

The equation is moved from the general buffer into a specific location in core and scheduled for evaluation. If the core size was increased substantually allowing more equations to be resident in core, most of the time required for loading the equations from RAD would be eliminated.

In addition it is quite conceivable that the method by which the memory is searched for the equation could be improved upon further decreasing the time required to process an equation.

2. Clocks - Associated with the simulator are 190 internal elapse time counters (ETC). The ETC's are preloaded with a negative count by the driver portion of the real-time program and are incremented every millisecond by the interface hardware. The updating of each counter requires 2 cycles of memory access time or 0.665 milliseconds of each millisecond to update the internal clocks.

Each clock that is active corresponds to a single timed analog equation to be evaluated. A more efficient means of evaluation would be to allow all equations to be evaluated at specified increments of time, be under one clock control. This eliminates the condition of having several consecutive clocks with the same count. This would update the driver portion of the program to a monitor mode of operation and reduce the number of interrupts the CPU must process for a given amount of time.

The hardware clock subsystem presently part of the simulator would be totally unacceptable in a 3rd generation computer. Possibly an improved method of implementing such a scheme would be to locate the clocks external to the CPU whereby they may be incremented by a selectable external source (0.1 KC, 1.0 KC, 10.0 KC) and would not cycle steal from the CPU for updating of the clocks. The clocks would not require refreshing by the driver program except when a different evaluation time is required.

ADDITIONAL HARDWARE REQUIREMENTS

A truly general-purpose simulator would require an I/O structure of great flexibility and speed. Being highly flexible the I/O structure would allow for several external systems to be connected simultaneously eliminating the need for recabling for each different simulator to take place. Such a configuration would allow for a shuttle simulation to take place on a given work shift, a space station simulation on a second shift and a Saturn launch program checkout or launch crew checkout to be performed on a quick turn around basis.

A high speed input/output structure is needed to perform the commutation of data needed to refresh any external units (DDAS) and to perform general high speed input/output operations.

Considering the cycle stealing scheme normally attributed to I/O commutations and the large data tables necessary for commutation purposes it becomes necessary to designate a small general purpose computer to perform the I/O functions previously handled by the SDS 930 computer.

SPECIAL CONSIDERATIONS

The Digital Data Acquisition System (DDAS) performs two basic functions:

1. Commutating the analog or transducer data.
2. Responding to requests from both of the RCA 110A computers.

Both functions are hardware functions and are performed automatically. The only software requirement is that of initial set-up. In the DDAS system each positive 0- to 5.0-volt transducer signal is represented by a 10-bit word. Two transducer words are combined to form a 20-bit computer word and are - sequentially stored in a commutated data table located in the upper 16K core memory of the 930 computer. The commutation rate is fixed and it is established by the 3.6-kHz tuning fork oscillator. The word location within the commutated data table is determined by the DDAS logic and in particular the following address counters:

1. A 3-bit data receiving station (DRS) counter.
2. A 2-bit muliplexer counter
3. A 4-bit frame counter
4. A 5-bit channel counter

The above format was chosen to correspond to the actual format of the Saturn vehicle telemetry. Any deviations from this commutation sequence would render the system unusable. For example, the above address counters are fixed word counters with no provisions made for altering the address counting sequence. A requirement for the increase in the number of bits in any of the four address counters would automatically dictate the redesign of the commutator logic. Also, the additional requirement for either a sign bit or the increase in the

number of data bits per transducer word would dictate larger registers than presently available in the DDAS.

Other inherent limitations of the existing DDAS are its excessive memory access time requirements for the commutation of data and the apparent large commutation error resulting from the simultaneous request for memory access by both the commutator and RCA 110A computers. As is stated in the DEE-6 Simulator System Technical Manual, 12.6 percent of the memory access time is taken up by the commutation process. Furthermore the memory access time is directly proportional to the commutator channel rate. The 12.6 percent figure is in itself very excessive considering the burden imposed on the 930 computer for the solution of equations and interrupt processing. However, with the same scheme it is impossible to process present commutation rates which range an order of magnitude or better than the 3.6-kHz rate. This would require better than 100 percent of the available memory access time.

The simultaneous request for memory access by both the commutator and either of the RCA 110A computers will cause a commutation rate error. The commutator requires 35 out of a total of 278 microseconds (3.6 kHz) available to access five address pairs and ten transducer values and to store five 20-bit data words in the commutated data table. Assuming the requests for memory access by either of the 110A computers to be random and independent of the commutation process, then the probability, P, of a rate error is

$$\frac{T_{com}}{T_{ch}} = \frac{35}{278} = 0.126$$

If n random requests are considered during a particular run and if each request is independent of the previous request then the probability, P_E, of at least one rate error is

$$P_E = \sum_{k=1}^{n} C_k^n P^k (1-P)^{n-k}$$

where

C_k^n is the binominal coefficient or n items taken k at a time
and P is the probability of error for each individual request.

Substituting for n = 5 and P = 0.126 than the probability of at least one commutation rate error becomes 0.492. As n or the commutation rate increases the probability of at least one rate error increases. To eliminate this problem requires the conceptual redesign of the DDAS simulation system.

As a solution to the DDAS problem Sperry Rand recommends the incorporation of a small general purpose computer to handle the multiplexing and transferring of data between the simulation computer and external system. Also recommended is the redesign of DDAS hardware to allow for programmable commutation rates, programmable commutation sequence, and programmable word length. The request for data by the 1108 computers (or equivalent) can be handled on a priority bases therby eliminating the commutation rate errors of the existing system. This approach will provide the needed flexibility to handle almost all of the existing digital data acquisition systems. Also, it will relieve the overhead and allow a more efficient equation evaluation.

SUMMARY

There appears to be many areas where the present simulator could be improved or updated to become a general-purpose simulator as discussed in the previous sections.

A review of the reports and documents compiled by Computer Sciences Corporation and SDS indicate that there is an immediate need for improving the existing system. The system errors and aborts encountered in the real-time mode of operation indicate that the overhead time for processing interrupts and I/O operations is too large.

While moving some of the input/output operations to a satellite computer would relieve the CPU of some overhead time, there still remains the question of how large a data base the CPU can adequately handle. With the simulation system disassembled to the extent that it presently is, this information cannot be obtained.

NASA previously issued a task directive (TD) to study ways by which the method of solving Boolean equations could be improved upon in the real-time mode of operation. No further effort was expended in this area once the simulation system was disassembled. The need for such a study is still very much in need and could contribute heavily in reducing redundant tables and large amounts of overhead time.

In conclusion it is suggested that further studies be made on the real-time portion of the MARDSLVC simulator. An effort should be made to separate the software and hardware responsibilities and update the software to operate more as an executive system. Such a system would be truly generalized and capable of handling very large math models.

REFERENCES

[1] L. W. Brooks, J. A. Stahley Real-Time Digital Simulation of The Saturn V System. Sperry Rand Engineering Review, Systems-Computer Applications 1969.

[2] Marshall Space Flight Center, Astrionics Laboraroty VLF-39-1 Digital Data Acquisition System (DDAS) for Saturn V, Interim Technical Report.

[3] L. W. Brooks, L. J. Gellman, J. A. Stahley Transfer Equation Preparation Manual for Launch Vehicle and Ground Support Equipment System Simulator. Prepared for NASA, Marshall Space Flight Center (R-ASTR-ESA), Sperry Rand Corporation, Huntstille, Alabama.

[4] H. Trauboth, C. Rigby, P. Brown Real-Time Space Vehicle and Ground Support Systems Software Simulator for Launch Programs Checkout. Proceedings of Spring Joint Computer Conference, Atlantic City, May 1970.

GENERAL PURPOSE SIMULATOR SYSTEM STUDY PART 2

Report No. SP-209-0339-1
Date: July 20, 1970

General Purpose Simulator System Study Part 2

Prepared for

George C. Marshall Space Flight Center
Contract No. NAS8-25608

Prepared By

Space Support Division
Division of Sperry Rand Corporation
Huntsville, Alabama

Submitted by: J. Scoggins
J. Scoggins

and

J. Taras
J. Taras

Approved by: R Vigil for 7-22-70
R. A. Rossler
R. A. Rossler

ABSTRACT

Contained herein is the analysis of the limitations of a breadboard simulator which simulates the Electrical Support Equipment and Digital Data Acquisition System for the Instrument Unit of the Saturn V vehicle. The analysis is made to determine if the breadboard simulator can be adapted to the electrical and electro-mechanical systems of the Shuttle and Orbital Workshop. Recommendations resulting from this analysis are also included.

TABLE OF CONTENTS

INTRODUCTION

Sperry Rand is performing a study of a breadboard simulator which simulates the Electrical Support Equipment (ESE) and Digital Data Acquisition System (DDAS) for the Instrument Unit (IU) of the Saturn V vehicle. The first objective is to determine how the breadboard simulator can be modified to simulate the electrical and electromechanical devices of the Shuttle or Orbital Wrokshop. If this is not possible, the study will determine if a large simulator system using the same language can be used. If this is not feasible, the study will develop plans for the most efficient system for Shuttle and Orbital Workshop simulation.

To design the most efficient simulator, it is necessary to identify the limitations of the breadboard simulator. This study should indicate the most practical steps to take to improve the breadboard simulator. The results of the IU simulation were analyzed to identify the problem of simulating the ESE and DDAS of the IU. It is assumed that the Shuttle and Orbital Workshop will present similar problems since the equations defining the electro-mechanical systems should be similar. The preliminary results of this study are contained in the analysis portion of this report.

As discussed in report SP-209-0339, the hardware configuration has placed great constraints on the breadboard simulator. To have a measure of these constraints and an appreciation for the amount of equation processing the simulator is capable of achieving, an additional study is being performed by Sperry Rand and Computer Science Corporation (CSC).

Several steps have been taken to provide for a measure of equation processing the simulator may accomplish. Listings were obtained of the Boolean equations for all stages of the Saturn V vehicle. Flowcharts of the breadboard real time simulator were drawn by CSC. The subroutines were timed to get an appreciation of the time involved for the processing of equations. Unique

equations were established containing the timing information established above. From the timed equations a capability study was performed and the results included in the analysis section of this report.

SCOPE

This report discusses the limitations of the breadboard simulator as it presently exists. It is hoped that this study will reveal all limitations of the simulator and establish the most logical or efficient means of modifying the breadboard simulator for future space missions.

QUALITATIVE ANALYSIS
(IU STAGE)

An analysis of the IU simulation was made to determine the amount of processing or equation solving the computer performed for given periods of time. Three samples, representing the heaviest load times, were chosen for the analysis. Since all limitations of the simulator are amplified during the period of maximum equation activity, the necessity for modifications to improve system response should be revealed during this period.

No attempt is made in this report to determine if the proper response exists for the system since only the testing computer (RCA-110A or equivalent) may establish the criteria for the test. Rather, it is hoped that this study will give proper insight into the necessary modifications should the need for improvement to system response exists.

Sample 1

For switch action S8021, 62 equations were evaluated. None of the equations were resident in memory and although eight of the equations were timed analog, the Boolean terms did not allow them to be scheduled for evaluations. The statistical information for this evaluation is as follows;

$$\frac{\text{2915 WORDS (TOTAL)}}{\text{62 EQUATIONS}} = \text{47 WORDS/EQUATION}$$

$$\frac{295}{62} = 4.76 \text{ "OR" SEGMENTS/EQUATION}$$

$$\frac{336}{62} = 5.42 \text{ CROSS REFERENCES/ EQUATION}$$

A total period of 2.125 seconds elapsed between the initial switch action and the resulting discrete action. Therefore,for this particular period the equation rate was:

$$\frac{62 \text{ EVALUATIONS}}{2.125 \text{ SEC}} = 29.2 \text{ EVALUATIONS/SECOND}$$

or

34.27 MS/EVALUATION

Some interesting things to observe about this particular example are;

1. None of the equations were resident in memory when initially evaluated.

2. The equations although not large in number were unusually long in length.

3. An unusually large number of "OR" segments and cross reference terms existed for this particular example.

4. 12.9 percent of the equations evaluated were analog equations.

Sample 2

For switch action S0087, 428 evaluations were performed. None of the equations were resident in memory and 38 or 9.1 percent of the equations were analog equations. The statistical information for this evaluation is as follows:

$$\frac{11511 \text{ WORDS (TOTAL)}}{419 \text{ EQUATIONS}} = 27.5 \text{ WORDS/EQUATION}$$

$$\frac{1228 \text{ "OR" SEGMENTS}}{419 \text{ EQUATIONS}} = 2.93 \text{ "OR" SEGMENTS/EQUATION}$$

$$\frac{1619 \text{ CROSS REFERENCES}}{419 \text{ EQUATIONS}} = 3.86 \text{ CROSS REFERENCES/EQUATION}$$

A total period of 11.068 seconds elapsed between the initial switch action and the final computation of the internal variable associated with the switch, action. Therefore for this particular period the evaluation rate was:

$$\frac{428 \text{ EVALUATIONS}}{11.068 \text{ SECONDS}} = 38.7 \text{ EVALUATIONS/SECOND}$$

or

25.8 MS/EVALUATION

Some interesting things to observe about this particular example are;

1. None of the equations were resident in memory when initially evaluated.

2. The equations to be evaluated were much shorter in length (27.5) than in example 1 and were probably more representative of the data base equation length.

3. The initial term evaluated was a power bus with an unusually large number of cross references (385) to be solved.

4. The great majority of equations solved were discrete equations and internal variables.

5. Of special interest is the limited response of the simulator for this particular sample. The equations indicate that when power bus 6110 was turned on, power bus 6195 should have been turned on simultaneously. The history tape printout indicates that there was a delay of 7.501 seconds. It is apparent that large delays may occur for discrete and analog events when the simulator is required to do a large amount of data accessing and equation evaluation.

Sample 3

The results obtained for switch action 8328 are inconclusive due to the limited amount of information available for this example. However, some interesting results were obtained as follows;

1. None of the equations were resident in memory when initially evaluated.

2. Eight of the equations were timed analog equations requiring simultaneous solutions every eight milliseconds.

3. The ideal time frame for solution of all equations involved including internal variables and discrete variables should be less than 16 ms while 1077 ms were used for total equation evaluation.

4. When solving analog equations all associated equations and cross-references require solving before any additional analog equations are solved. The results of the IU history tape indicates that this criteria was not met since several additional analog equations were solved before the previous cross-references were solved.

5. The solving of the analog equations themselves fell out of the expected orderly sequence, suggesting that the eight analog equations and their cross-references could not be solved within the 8 ms interval alloted for their solution.

6. For switch action S8328 all analog equations should have been solved immediately and simultaneously including the analog equation E4117 for which there was a 172 ms delay. The internal variable V2581 should also have been solved immediately; however, there was an additional 357 ms delay. The discrete term B0829 should have been solved immediately; however, there was an additional 511 ms delay or a total of 1040 ms from switch action to the expected results. There is no obvious way to indicate if this response is adequate and no attempt is made in this report to do so. It is obvious however that the solution of the analog equations are taking more time than the interval of 8 ms intended for re-evaluation. As more analog equations are scheduled for evaluation the problems encountered become more acute.

MATHEMATICAL ANALYSIS

To further evaluate the simulator and determine the individual software subroutines effecting the response of the simulator an additional study has been made. While this mathematical analysis is preliminary in nature, it does give further insight into the nature of the system response. The example in this analysis should be regarded as a typical equation to be solved rather than the average equation solved by the simulator.

The information compiled for this analysis is a result of the flow charts and equations developed by CSC. The equations were relabeled for convenience and are included in Appendix A. The flow charts of the real-time program were redrawn for clarity and are included in Appendix B.

Example:

The example here is for a timed analog equation with the following format and characteristics;

$$E_{XXXX} = \ldots\ldots |ARG_1| + \ldots\ldots |ARG_2| + \ldots\ldots |ARG_3| + \ldots\ldots |ARG_4| \$$$

The following additional assumptions are made for solution of this analog equation.

1. There are eight such equations active simultaneously.

2. The equation is 51 words in length plus 2 cross-reference terms plus 4 pick-up, drop-out values or 57 words long and is to be contained in a 60 word buffer.

3. The equation is non-resident in memory initially.

4. The analog equation consists of four "OR" segments with the third "OR" term being true.

5. Assume five Boolean terms and six terms within the slashes for each "OR" segment.

6. The analog equation is to be evaluated every 8 ms.

7. Assume the analog equation type to be the normal non-additive type (T).

The following constants were also chosen for aiding in the equation solution.

K_1 = Number of buffers within buffer group. (10)

K_2 = 1, 3, 5 depending on Queue, i.e., special, timed, normal respectively.

K_3 = Number of words in equation.

K_4 = Boolean terms processed. (15)

K_5 = Number of "OR" terms processed.

K_6 = Number of related equations if analog equation evaluated changed in status or value.

K_7 = 0 if analog equation did not change in status or value; 1 if analog changed status or value.

K_8 = Relative position of desired entry in secondary timer tables. (4^{th})

K_9 = Number of related equations if discrete equation evaluated changed in status.

K_{10} = 0 if discrete equation did not change in status; 1 if discrete equation changed in status.

K_{11} = Queue priority = 1, 2, 3 for special, timed, normal respectively.

The time required to solve the example analog equation may be broken into five separate sections as follows.

t_1 = Real-time driver processing.

t_2 = Time to search for equation in memory and place in memory if not resident.

t_3 = Time to retrieve the analog equation from the disc. (17.5 ms)

t_4 = Interrupt process routine for the disc.

t_5 = Time to solve the analog equation.

t_6 = Time to record data on tape.

For the analog equation these computations are in the following sequenced order; (see Appendix A for equation terms).

$$T_1 = t_{\substack{\text{DRIVER} \\ \text{ANALOG}}} = .00175\left[220 + \frac{63 \text{ X } 40 + 80 + 33}{64}\right]$$

$$= 0.456 \text{ MS}$$

$$T_2 = t_{\substack{\text{EQINCORE} \\ \text{ANALOG}}} = X_{A1} + X_{B1} + X_{C1} + X_{D1}$$

$$= (K_{11}X_{A2} + X_{B2} + X_{C2} + X_{D2}) + X_{B1} + X_{C1} + X_{D1}$$

$$= K_{11}X_{A2} + X_{B2} + \left(X_{A3} + K_1X_{B3} + X_{\substack{C3 \\ D3}}\right) + X_{D2} + K_2X_{A5} + X_{B5} + X_{C1} + X_{D1}$$

$t_2 = 0.00175[(1)\ 45 + 50 + 47 + (10)\ (30) + 25 + 0 + (1)\ (10) + 70 + 30 + 30]$

$t_2 = 0.589$ MS

t_3 = Average Disc access time = 17.5 MS

t_4 = Disc interrupt process time = $t_{BDCINRPT}$

$t_4 = t_{BDCINRPT} = X_{A6} + K_3 X_{B6} + X_{C6} + X_{D6} + X_{E6}$

$= X_{A6} + K_3X_{B6} + X_{C6} + K_2X_{A5} + X_{B5} + X_{E6}$

$= 0.00175\,[62 + 18\ (57) + 53 + 10 + 70 + 26]$

$t_4 = 2.182$ MS

t_5 = Time to solve analog equation.

$t_5 = t_{\substack{ANALOG \\ AEQVALUE}} = 0.00175\,[X_{A10} + K_4X_{B10} + K_5X_{C10} + X_{D10} + K_6\ (30) + K_7\ (100) + K_8\ (11) + 102 + X_{J11} + X_{K11} + X_{L11} + X_{M11}]$

$t_5 = 0.00175\,[X_{A10} + K_4X_{B10} + K_5X_{C10} + X_{D10} + K_6\ (30) + K_7\ (100) + K_8\ (11) + 102 + (X_{A4} + X_{B4} + 4X_{D4} + X_{E4,\ F4} + X_{G4} + X_{H4}]$

$= 0.00175\,[73 + (15)\ (110) + 2\ (43) + 38 + 2\ (30) + 1\ (100) + 4\ (11) + 102 + 40 + 225 + 4\ (100) + 45 + 1225 + 139 + 80 + 83 + 64 + 2\ (185)]$

$t_5 = t_{\substack{ANALOG \\ AEQVALUE}} = 8.442$ MS

$t_6 = t_{tape} = 4 \frac{WDS.}{EQ.} \times 24 \frac{BITS}{WD} \times \frac{1\ SEC}{150\ IN.} \times \frac{1\ IN.}{200\ BITS} = 3.2$ MS

$T_{ANALOG} = t_1 + t_2 + t_3 + t_4 + t_5 + t_6$

$= t_{\substack{DRIVER \\ ANALOG}} + t_{\substack{EQINCORE \\ ANALOG}} + t_{ACCESS} + t_{BDCINRPT} + t_{\substack{ANALOG \\ AEQVALUE}} + t_{TAPE}$

$= 0.456$ MS + 0.589 MS + 17.5 MS + 2.182 MS + 8.442 MS + 3.2 MS

$T_{ANALOG} = 32.4$ MS

From the above analysis approximately 20 MS or 2/3 of the time required to process the equation is used for searching for the equation and retrieving the equation from the disc. The time required to solve the equation (8.442 MS) in itself exceeds the 8 MS interval requested for additional analog solutions of the same equation.

If the above analog equation should reach a pick-up or drop-out value, change in status i. e. (off-On), or change in magnitude by 1 millivolt or more an internal variable equation would be scheduled for evaluation of the form;

$$V_{XXXX} = E_{XXXX} \quad // \text{ Pick-Up Voltage, Drop-Out Voltage } // \$$$

A similar mathematical analysis may be performed for the solution of the internal variable with the following assumptions:

1. Length of internal variable equation is 17 words.
2. Equation not resident in memory..
3. Equation is logged in Timed Queue.
4. A change in state of the internal variable effects one additional discrete or B type equation.

Assuming in addition:

t_7 = Real-time driver processing

t_8 = Time to search for equation in memory and place in memory if not resident.

t_9 = Time to retrieve the internal variable equation from the disc.

t_{10}= Interrupt process routine for the disc.

t_{11}= Time to solve the internal variable equation.

t_{12}= Time to record data on tape.

For the solution of the internal variable equation the time required would be;

$$t_7 = t_{\substack{\text{DRIVER} \\ \text{DISCRETE}}} = 0.00175 \left[20 + \frac{63X40 + 80 + 33}{64}\right]$$

$$t_7 = 0.107 \text{ MS}$$

$$t_8 = t_{\substack{\text{EQINCORE} \\ \text{INT. VAR.}}} = 0.00175 \left[X_{A1} + X_{B1} + X_{C1} + X_{D1}\right]$$

$$= 0.00175 \left[(K_{11}X_{A2} + X_{B2} + X_{C2} + X_{D2}) + K_2X_{A5} + X_{B5} + X_{C1} + X_{D1}\right]$$

$$= 0.00175 \left[K_{11}X_{A2} + X_{B2} + (X_{A3} + K_1X_{B3} + X_{D3}) + K_2X_{A5} + X_{B5} + X_{C1} + X_{D1} + X_{D2}\right]$$

$$= 0.00175 \left[(2\ (45) + 50 + (47 + 10\ (30) + 25) + 35) + 3\ (10) + 70 + 30 + 30\right]$$

t_8 = TEQINCORE = 0.887 MS
INT. VAR.

t_9 = Average access time for disc = 17.5 MS

$$t_{10} = t_{BDCINRPT} = 0.00175 \left[X_{A6} + K_3X_{B6} + X_{C6} + X_{D6} + X_{E6}\right]$$

$$= 0.00175 \left[X_{A6} + K_3XB_6 + X_{C6} + (K_2XA_5 + X_{B5}) + X_{E6}\right]$$

$$= 0.00175 \left[62 + 17\ (18) + 53 + (3\ (10) + 70) + 26\right]$$

$$t_{10} = t_{BDCINRPT} = 0.957 \text{ MS}$$

$$t_{11} = t_{\substack{\text{AEQVALUE} \\ \text{INT. VAR.}}} = 0.00175 \left[X_{A10} + 1\ X_{B10} + K_9\ (30) + K_{10}\ (100) + X_{F10}\right]$$

$$= 0.00175 \left[73 + 1\ (110) + 38 + 2\ (30) + 1\ (100) + 140\right]$$

$t_{11} = t_{AEQVALUE} = 1.087$ MS
$t_{12} = 3.2$ MS

$$T_{\text{INT. VAR.}} = t_7 + t_8 + t_9 + t_{10} + t_{11} + t_{12}$$

$$= t_{\substack{\text{DRIVER} \\ \text{INT. VAR}}} + t_{\substack{\text{EQINCORE} \\ \text{INT. VAR.}}} + t_{DISC} + t_{BDCINRPT} + t_{\substack{\text{AEQVALUE} \\ \text{INT. VAR}}} + t_{TAPE}$$

$= 0.107 \text{ MS} + 0.887 \text{ MS} + 17.5 \text{ MS} + 0.957 + 1.087 \text{ MS} + 3.2 \text{ MS}$

$T_{\text{INT. VAR.}} = 23.738 \text{ MS}$

From the above analysis approximately 19.4 MS or 82 percent of the time required to process the equation is used to search for and retrieve the equation from the disc. The total time used thus far in processing the Analog equation and internal variable equation is:

$T = T_{\text{ANALOG}} + T_{\text{INT. VAR.}} = 32.4 \text{ MS} + 23.738 \text{ MS}$

$T = 56.138 \text{ MS}$

If in addition to solving the internal variable there is a cross-reference equation (B type) to solve. There will be additional time required for this solution as follows:

Assuming the following equation format and characteristics:

1. Equation of form BXXXX = ... + ...$
2. Equation 14 words in length
3. Two "or" segments
4. Assume second "OR" term true
5. Equation placed in timed Quene
6. Equation not resident in memory

Then the total time required to search for, access, and process the discrete equation would consist of the following terms:

t_{13} = Real time driver processing time

t_{14} = Time to search for equation in memory and place in memory if not resident

t_{15} = Time to retrieve the Discrete equation from disc.

t_{16} = Interrupt process routine for the disc

t_{17} = Time to solve the Discrete equation

t_{18} = Time to record data on tape

$$t_{13} = t_{\substack{DRIVER \\ DISCRETE}} = 0.00175 \left[20 + \frac{63X40 + 80 + 33}{64}\right] = 0.107 \text{ MS}$$

$$t_{14} = t_{\substack{EQINCORE \\ DUSCRETE}} = 0.00175 \left[X_{A1} + X_{B1} + X_{C1} + X_{D1}\right]$$

$$t_{14} = 0.00175 \left[K_{11}X_{A2} + X_{B2} + X_{A3} + K_1 XB_3 + X_{D3} + X_{D2} + K_2 X_{A5} + X_{B5} + X_{C1} + X_{D1}\right]$$

$$= 0.00175 \left[2\ (45) + 50 + 47 + 10\ (30) + 25 + 35 + 3\ (10) + 70 + 30 + 30\right]$$

$$t_{14} = t_{\substack{EQINCORE \\ DISCRETE}} = 0.809 \text{ MS}$$

$$t_{15} = t_{DISC} = 17.5 \text{ MS}$$

$$t_{16} = t_{BDCINRPT} = 0.00175 \left[X_{A6} + K_3 X_{B6} + X_{C6} + K_2 X_{A5} + X_{B5} + X_{E6}\right]$$

$$= 0.00175 \left[62 + 14\ (18) + 53 + 3\ (10) + 70 + 26\right]$$

$$t_{16} = t_{BDCINRPT} = 0.863 \text{ MS}$$

$$t_{17} = t_{\substack{DISCRETE \\ AEQVALUE}} = 0.00175 \left[X_{A10} + K_4 X_{B10} + X_{C10} + X_{D10} + K_9\ (30) + K_{10}\ (100) + X_{F10}\right]$$

$$= 0.00175 \left[73 + 6\ (110) + 43 + 38 + 1\ (30) + (100) + 140\right]$$

$$t_{17} = t_{\substack{DISCRETE \\ AEQVALUE}} = 1.114 \text{ MS}$$

$$t_{18} = t_{TAPE} = 3.2 \text{ MS}$$

The above results indicate that the total time to solve the discrete equation would be:

$$T_{DISCRETE} = t_{13} + t_{14} + t_{15} + t_{16} + t_{17} + t_{18}$$

$$= t_{\substack{DISCRETE \\ DRIVER}} + t_{\substack{EQINCORE \\ DISCRETE}} + t_{DISC} + t_{BDCINRPT} + t_{\substack{DISCRETE \\ AEQVALUE}} + t_{TAPE}$$

$$= 0.107 \text{ MS} + 0.809 \text{ MS} + 17.5 \text{ MS} + 0.863 \text{ MS} + 1.114 \text{ MS} + 3.2 \text{ MS}$$

$$T_{DISCRETE} = 23.593 \text{ MS}$$

The total time that has elapsed for the solution of these three equations becomes:

$$T_{TOTAL} = T_{ANALOG} + T_{INT.\ VAR} + T_{DISCRETE}$$

$$= 32.4 + 23.738 + 23.593 \text{ MS}$$

$$T_{TOTAL} = 79.731$$

The average time for the solution of three equations becomes 26.577 MS per solution which very nearly approaches the samples discussed in the Qualitative Analysis section of this report. It is interesting to note that if the equations had been resident in memory (they normally are not), the disc access time and disc interrupt subroutines would have been eliminated reducing the total process time to 23.3 MS for an average of 7.7 MS per equation.

Not computed in the above equations is the cycle stealing time required for DDAS commutation. This would increase the computational time of all subroutines by 13 percent. The results obtained in the Qualitative Analysis section also lacked the DDAS cycle stealing time since none of the stations were active for the IU simulation.

SUMMARY

The switch action and discrete out events for the IU cause many variations in the number and type of equations solved for each event. It becomes necessary for the breadboard simulator to solve the equations at a rate acceptable by program specifications for the particular space vehicle being simulated. It is hoped that the information contained within this report will aid in establishing the processing capability of the breadboard simulator. At present statistical information is being made to establish the processing rate of the simulator for various equation configurations. An additional report will be submitted containing the results of the study.

CONCLUSIONS AND RECOMMENDATIONS

The electrical-mechanical system of any space vehicle may be simulated by the MARDSLVC simulation language provided that the response of the simulator meets the specifications set forth for the individual elements being simulated. At present there is very little information available concerning the system responses of the Shuttle. However it should be safe to assume that the required system response will be similar to the Saturn V since similar systems will be simulated. If the Shuttle simulation requires a substantial improvement in system response the following improvements to the MARDSLV simulator should be considered:

1. Make all equations resident in core. This single item will be the most effective means of improving the response of the simulator.
2. Provide a separate processor for the solution of analog equations. Add external clocks to eliminate redundant data moving and superfluous equation management.
3. Provide parallel or multi-processing so that the Real-time driver portion of the program does not reduce the amount of equation processing taking place. Also parallel equation processing may be performed.
4. Provide for using hardware floating point operations to eliminate time consuming software subroutines.
5. Make all variables with discrete states or analog values available for immediate access for all processors through shared memory.
6. Provide a separate processor for communication with the vehicle computer.
7. Permit the history tape to be recorded by above processor through separate ports to memory. Such a configuration would reduce the cycle stealing from the main processor by the tape controller.

APPENDIX A

TIMED SUBROUTINES OF REAL-TIME PROGRAM

EQINCORE - checks queues for equations to evaluate; or equations to input; or input of a cross reference block

		Cycles
X_{A1}	check queues for equation to input.	t_{CHKEQ}
X_{B1}	get highest priority equation from disc I/O table.	t_{UPACK}
X_{C1}	Initiate input from disc.	30
X_{D1}	check indicated queue for input in memory.	30

$$t_{EQINCORE} = X_{A1} + X_{B1} + X_{C1} + X_{D1}$$

CHKEQ - check queues for an equation to be input.

X_{A2} check for open slot in disc I/O table which corresponds to this queue.

Special Queue	45	(1)	
Timed Queue	90	(2)	45 cycles/queue priority
Normal Queue	135	(3)	

X_{B2} check for buffer code and disc address in this queue. 50 cycles

X_{C2} check if equation is already in core t_{ECORE}

X_{D2} all queues empty; no equ, to input 35 cycles

$$t_{CHKEQ} = N(X_{A1}) + (X_{B1}) + t_{ECORE} + X_{D1}$$

where N is the queue priority 1, 2, or 3

ECORE - check the indicated equation buffer (buffer code is indicator) to see if the equation is already in memory

		Cycles
X_{A3}	Parameter setup each time ECORE is entered	47
X_{B3}	Time consumed for each buffer check (No. of buffers for IU is 1 min.; 50 max.)	30
X_{C3}	Additional time consumed when a match is found	33
X_{D3}	Additional time consumed when no match is found and the buffer is avaialble.	25

Only one of these (c,d) can be true at any one time.

$$t_{ECORE} = X_{A3} + N(X_{B3}) + X_{C3} + X_{D3}$$

Where N is the number of equations in the particular buffer group. The minimum for the IU is 1 and the maximum is 50. Consult buffer data deck to obtain the min/max for any stage.

There are 3 exits from this routine:

EXIT 1: Equation is in memory (match found)

a. Minimum time $X_{A3} + X_{B3} + X_{C3}$ = 110 cycles

b. Maximum time $X_{A3} + 50X_{B3} + X_{C3}$ = 1580 cycles

EXIT 2: Equation is not in memory and buffer is available

a. Minimum time is $X_{A3} + X_{B3} + X_{D3}$ = 102 cycles

b. Maximum time is $X_{A3} + 50X_{B3} + X_{D3}$ = 1572 cycles

EXIT 3: Equation not in memory and no buffer available

a. $X_{A3} + X_{B3}$ = 77 cycles

b. $X_{A3} + 50X_{B3}$ = 1547 cycles

CALOG - evaluate analog portion (within slash marks) of equation.

			Cycles
X_{A4}	Parameter setup		40
X_{B4}	Pick-up E type parameters		225
	or (CEFV)		
X_{C4}	Pick-up F type parameters		86
X_{D4}	Calculate time and set-up parameters:		100
X_{E4}	Adjust interval time if initial eval.		45
	or (CTIME)		
X_{F4}	Adjust interval time NOT initial evaluation		158
X_{G4}	Evaluate analog:	P) Polynomial	351
		T) normal non-additive	1225
		C) cyclic non-additive	1952
		A) normal additive	1385
		B) cyclic additive	2114
		G) summation	586
		H) multiplication	405
		U) update	290
X_{H4}	Store results (data word)		
		E-type	139
		F-type	58

UPACK - determines (1) the highest priority disc address to be input and (2) the approximate number of words to be input.

		Cycles
X_{A5}	Find highest priority address in the disc I/O table of six entries.	10 per entry
X_{B5}	Compute approximate word count for the desired disc address.	70

$t_{UPACK} = N(X_{A5}) + X_{B5}$

Where: N is a number 1, 3, or 5 depending on the queue from which the disc address was taken - special, timed, or normal, respectively.

Two exists are provided:

EXIT 1: disc address found and word count computed

a. Minimum: $1(X_{A5}) + X_{B5} = 10 + 70 = 80$ cycles

b. Maximum: $5(X_{A5}) + X_{B5} = 50 + 70 = 120$ cycles

EXIT 2: no disc address found

Minimum/Maximum: $5(X_{A5}) = 50$ cycles

BDCINRPT - disc interrupt routine

		Cycles
X_{A6}	Determine exact equation word count and compute its intended core address	62
X_{B6}	Transfer 1 word to intended core address.	18 each
X_{C6}	Compress disc I/O tables	53
X_{D6}	Call UPACK for next highest priority address in disc I/O table	t_{UPACK}
X_{E6}	Initiate next input if any.	26

$$t_{BDCINRPT} = X_{A6} + N(X_{B6}) + X_{C6} + t_{UPACK} + X_{E6}$$

Where: N is the number of words in the equation.

AELTIME - Interrupt routine for the elapsed time counters.
211

	Cycles
a. Cycle time for analog equation timer	153
b. Cycle time for pick-up/drop-out times	120

RMILCLK - Interrupt routine for millisecond clock
210

	Cycles
a. Cycles required to update internal clock	30

DDAS3 - Interrupt routine for DDAS commutation. The interrupt occurs after 500 words in the address pair table have been scanned. There are 9 blocks of 500 words.

	Cycles
a. Cycles required to refresh interface	37

ADOCHK - check DO log table for a DO word to process

ACOMPDO - Process DO word from log table

		Cycles
a.	Check for entry in DO log table	65.0
b.	Check entry for validity (error check)	61.5
c.	Obtain DO status word (previous) for comparison	84.5
d.	Check least significant bit	11.0
e.	Additional bits checked	10.0 each
f.	Index in memory? (if bit changed) (AEQLOG = 150.0)	15.0
g.	Index is in memory; log related equations into normal queue.	246.5
h.	Update internal status (1 bit change at a time) (ROUT = 35.0)	65.0
i.	New DO status word (24 bits) to history tape (AEQLOG = 100.0)	55.0
j.	Index NOT in memory; disc address of index to queue	163.5

$$t = X_a + X_b + X_c + X_d + Q(X_e) + R(X_f) + R(X_g) + (RX_h) + X_i + X_j$$

Where: Q is the number of bits interrogated (1-24)

R is the number of interrogated bits which had a change in status (1-24)

FLOATING POINT SUBROUTINES

HFSC - f. p. divide	140.5 cycles
HFSM - f. p. multiply	100.0 cycles
HFSS - f. p. subtract	136.0 cycles
HFSA - f. p. addition	114.0 cycles
CMINMAX - check f. p. limits	42 cycles
FTB - f. p. to binary	42 cycles
CCOTFP - DDAS to f. p.	78 cycles
CFPTCO - f. p. to DDAS	29 cycles

AEQVALUE - Controls the evaluation of a transfer equation

COMMON TO ALL EQUATIONS

		Cycles
X_{A10}	Initial processing	73
X_{B10}	Determine desired status of 1 status variable	110
X_{C10}	Process a "+" ("or") operation	43
X_{D10}	Process a "$" operator	38
X_{E10}	Log related equation into a specified queue <u>if</u> equation changed in status or in value. (cross reference)	N(30) + 100 N = # of related equs.
	Timed and non-timed discrete	
X_{F10}	Store evaluation results in history buffer	140
	Timed discrete only	
X_{G10}	Set pick-up or drop-out timer(s) for a timed discrete equation	N(67) + 37 N = # of timers to set
X_{H10}	Determine if pick-up or drop-out timer has expired (timed discrete)	N(4) + 70 N = # of clock cells to be checked

$$t = X_{A10} + M(X_{B10}) + P(X_{C10}) + X_{D10} + X_{E10} + X_{F10} + X_{G10} + X_{H10}$$

Discrete AEQVALUE — timed discretes only

M = # of status variables

P = # of "+" signs

VXXXX = EXXXX// , //$

= 567 cycles if either the pick-up or drop value was reaches.

Analog
"/ /"
only

X_{I11}	Pick-up analog equ. information from secondary (analog) timer tables	N(11) + 102 N = # of entries to be checked in the tables
X_{J11}	Evaluate analog portion (within slashes)	t_{CALOG}
X_{K11}	Reset analog timer	80
X_{L11}	Store analog results in history buffer	83
X_{M11}	Determine if analog value reached a threshold; P = pick-up values, drop-out values = D	64 + P(185) + D(185)

$$t = X_{A11} + W(X_{B11}) + P(X_{C11}) + X_{D11} + X_{E11} + X_{I11} + X_{J11} + X_{K11} + X_{L11} + X_{M11}$$

Analog
AEQVALUE

W = # of status variables

P = # of "+" signs

MAIN DRIVER

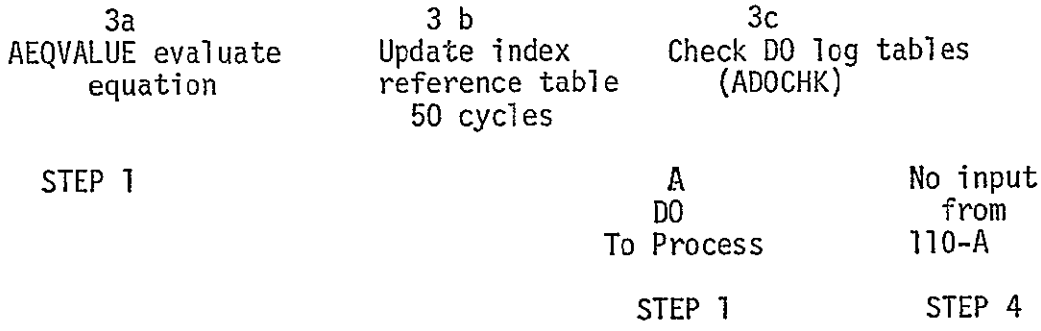
STEP 1: ACLKQUE - Move one disc address from the clock queue to the special queue provided the timed queue is empty.

Clock queue will hold the disc address of timed analog equations each time the "I" timer expires for that particular analog.

TIME: Minimum 20 cycles if timed queue is not empty.
Maximum 220 cycles if timed queue is empty.

STEP 2: BHISTINT - pseudo interrupt routine for history tape output
BHISTOUT - history tape output routine
33 cycles for BHISTINT
+40 cycles for BHISTOUT if (1) No buffer is ready,
(2) the channel is busy
+80 cycles to initiate actual output of a buffer
153 cycles

STEP 3: EQINCORE - Find or input equation to evaluate



STEP 4: ASWTCHK - Simulated switch input from card reader

		Cycles
X_{A13}	Initial setup each entry	58
X_{B13}	Determine if proper block of index is in core	45
X_{C13}	Locate information in the index block pertaining to this switch	75
X_{D13}	Update status of switch if it changed, and send event to history buffer	83
X_{E13}	Log effected equations into normal queue if the switch changed in status	200 + N(30) N = # of effected equations

APPENDIX B
MARDSLVC CAPABILITY STUDY
Response and Equation Evaluation Definition

Computer Sciences Corporation
Systems Development Operations
Aerospace Systems Center

November 4, 1970

MARDSLVC Capability Study

Response and Equation Evaluation Definition

Prepared by: P. A. Brown
P. A. Brown CSC

Approved by: C. O. Rigby
C. O. Rigby CSC

Approved by: H. Trauboth
Dr. H. H. Trauboth S&E-COMP-CS

In Support of the Systems Analysis Branch
Computation Laboratory, George C. Marshall
Space Flight Center, National Aeronautics
and Space Administration

TABLE OF CONTENTS

INTRODUCTION

The enclosed documentation represents a portion of the effort expended by Computer Sciences Corporation personnel in the development of a general purpose simulator utilizing the Saturn V Breadboard Simulator (MARDSLVC) concept, design, and capabilities.

SECTION 1

A) In order to determine the software capability, e.g., response time and equation evaluation time, of the MARDSLVC system, CSC personnel flow charted the major functions (subroutines) in the Real-Time Phase of MARDSLVC and derived equations for computing the cycle time required to perform each of these functions. Reference Appendix A and Figure 5 in Appendix A of this document.

To determine either 1) the response time to a particular external stimuli (simulated switch or discrete output) or 2) the cycle time for a particular equation evaluation requires effective utilization of Appendix A.

B) To determine the cycle time required for a particular evaluation, the following criteria should first be established.

1) The type of equation to be evaluated as described in Appendix B of this document.

2) The equation length, i.e., number of words.

3) The queue from which the equation is to be evaluated, e.g., Special, Timed, or Normal.

4) The location of the equation, e.g., in core memory or on the RAD.

5) The number and type of cross reference contained in the equation, e.g., other equations, timers, and/or pick-up values and drop-out values.

C) To determine the response time to an external stimuli, the following criteria should first be established.

1) The type of external stimuli, e.g., simulated switch or discrete output.

2) The queue is always Normal queue (lowest priority).

3) The location of the cross reference index block, e.g., in core memory or on the RAD. The index block contains the disc address of all equations effected by the particular stimuli.

4) The number of equations effected by the stimuli.

SECTION 2

A) Being more familiar 1) with the different types of transfer equations; 2) with their format within the MARDSLVC system; and 3) with the processing required for each type, CSC personnel also derived new formats for the four major groups of equation types. The major groups are analog equations, non-timed discrete equations, timed discrete equations, and threshold discrete equations, Figures 1, 2, 3 and 4, respectively. The new formats decrease the MARDSLVC format by a minimum of 5 computer words per equation. The decrease is attributed to the use of relative address words which eliminate the need for any logic operators such as logical "OR", slash marks, or dollar sign presently used in the MARDSLVC format. The relative address words eliminate the need to scan the equation for the location of a desired logical operator and allow the software to readily access pertinent information within the equation. A prior knowledge of the MARDSLVC system indicates that the relative address technique will substantially decrease processing time for all equation groups.

B) The reformatted analog equation, Figure 1, Section 2, decreases the MARDSLVC analog format by J+5 computer words. J is the number of segments (logical "OR" terms) in a MARDSLVC analog equation.

Eliminated from the MARDSLVC format are:

1	Disc address word
5	Keywords
J-1	Plus signs (logical "OR" operator)
1	Dollar sign operator
2J	Slash marks enclosing analog parameters
Total 3J+6	

Added to the format are:

1	Keyword (word 2 of Figure 1)
2J	Relative addresses in words 3 through 2J+2
Total 2J+1	

The resultant analog decrease is:

$$[(3J+6) - (2J+1)] = J+5 \text{ computer words/equation}$$

C) The reformatted non-timed and timed discrete equations, Figures 2 and 3, Section 2, decrease their MARDSLVC discrete format by 5 computer words. J is the number of segments (logical "OR" terms) in a MARDSLVC discrete equation.

Eliminated from the MARDSLVC format are:

1	Disc address word
5	Keywords
J-1	Plus signs (logical "OR" operators)
1	Dollar sign operator
Total J+6	

Added to the format are:

1	Keyword (word 2 of Figures
J	Relative addresses in words 3 through J+2
Total J+1	

The resultant discrete decrease is:

$$[(J+6) - (J+1)] = 5 \text{ computer words/equation}$$

D) The reformatted threshold discrete equation, Figure 4, Section 2, decreases the MARDSLVC format by 6 computer words.

Eliminated from the MARDSLVC format are:

1	Disc address word
5	Keywords
1	Dollar sign operator
Total 7	

Added to the format are:

1	Keyword (word 2 of Figure 4)
Total 1	

The resultant threshold discrete decrease is:

$$[(7) - (1)] = 6 \text{ computer words/equation}$$

J = # of equation segments
N = # of status variables
R = # of analog parameters

ANALOG EQUATION FORMAT

WORD	DEFINITION
0	# of words in the equation (P+1)
1	Equation identifier (group/sub-group)
2	Relative address (K+N+R) of related information/ # of equation segments (J)
3	Relative address K of segment 1 status variables
4	Relative address L of segment 2 status variables
.	. .
.	. .
.	. .
J+2	Relative address M of segment J status variables
J+3	Relative address K+N of segment 1 analog parameters
J+4	Relative address S of segment 2 analog parameters
.	. .
.	. .
.	. .
2J+2	Relative address T of segment J analog parameters
K	Status variable 1 of segment 1
K+1	Status variable 2 of segment 1
.	. .
.	. .
L	Status variable 1 of segment 2
L	
L+1	Status variable 2 of segment 2
.	. .
.	. .
M	Status variable 1 of segment J
M+1	Status variable 2 of segment J
.	. .
.	. .
K+N-1	Last status variable in equation

Figure 1, Section 2

ANALOG EQUATION
FORMAT (Continued)

WORD	DEFINITION
K+N	Analog parameters for segment 1
S	Analog parameters for segment 2
.	. .
.	. .
T	Analog parameters for segment J
K+N+R	1st related information*
K+N+R+1	2nd related information
.	. .
.	. .
P	Last related information

*Related information within an analog equation can be any one or all of the following types. Each type having the recommended format.

Code/Voltage
bits 0-2/bits 3-23
1) Pick-up Voltage; binary code value of 001

Code/Voltage
bits 0-2/bits 3-23
2) Drop-out Voltage; binary code value of 011

**P.C./Rel.Addr.
bits 3-5/bits 6-23
3) Related Equation; bits 0-2 not used

**P.C. = Processor code; allows multi-processors to be used.

Rel.Addr. = Relative address (memory address) of related equation within the indicated processor.

Figure 1, Section 2
(Continued)

J = # of equation segments
N = # of status variables

NON-TIMED DISCRETE
EQUATION FORMAT

WORD	DESCRIPTION
0	# of words in the equation (P+1)
1	Equation identifier (group/sub-group)
2	Relative address (K+N) of related information/ # of equation segments (J)
3	Relative address K of segment 1 status variables
4.	Relative address L of segment 2 status variables
.	
.	
.	
J+2	Relative address M of segment J status variables
K	Status variable 1 of segment 1 (group/sub-group)
K+1	Status variable 2 of segment 1
.	
.	
L	
L	Status variable 1 of segment 2
L+1	Status variable 2 of segment 2
.	
.	
M	Status variable 1 of segment J
M+1	Status variable 2 of segment J
.	
.	
K+N-1	Last status variable in equation
	bits 3-5/bits 6-23
K+N	1st related equation (P.C./REL.ADDR.)
K+N+1	2nd related equation
.	
.	
.	
P	Last related equation

Figure 2, Section 2

J = # of equation segments
N = # of status variables
Q = 2(#) of pick-up timers
R = 2(#) of drop-out timers

TIMED DISCRETE
EQUATION FORMAT

WORD	DESCRIPTION
0	# of words in the equation (P+1)
1	Equation identifier (group/sub-group)
2	Relative address (K+N) of related information/ # of equation segments (J)
3	Relative address K of segment 1 status variables
4	Relative address L of segment 2 status variables
. . .	
J+2	Relative address M of segment J status variables
K	Status variable 1 of segment 1 (group/sub-group)
K+1	Status variable 2 of segment 1
. .	
L	Status variable 1 of segment 2
L+1	Status variable 2 of segment 2
M	Status variable 1 of segment J
M+1	Status variable 2 of segment J
K+N-1	Last status variable in equation
K+N	bits 0-2/bits 3-23* 1st pick-up timer (code/time)
K+N+1	bits 3-5/bits 6-23 Equation related to timer (P.C./REL.ADDR.)
K+N+2	2nd pick-up timer

*Reference next page

Figure 3, Section 2

WORD	DEFINITION
	bits 3-5/bits 6-23
K+N+3	Equation related to timer (P.C./REL.ADDR.)
	bits 0-2/bits 3-23*
K+N+Q	1st drop-out timer (code/time)
K+N+Q+1	Equation related to timer
	bits 3-5/bits 6-23
K+N+Q+R	1st non-time related equation (P.C./REL.ADDR.)
K+N+A+R+1	2nd non-time related equation
P	Last non-time related equation

*Pick-up/drop-out timer recommended format is:

CODE/TIME
bits 0-2/bits 3-23
1) Pick-up timer; binary code value of 100

CODE/TIME
bits 0-2/bits 3-23
2) Drop-out timer; binary code value of 110

Figure 3, Section 2
(Continued)

THRESHOLD DISCRETE
EQUATION FORMAT

WORD	DEFINITION
0	# of words in the equation (P+1)
1	Equation identifier (group/sub-group)
2	Relative address (6) of related information/ # of equation segments (J)*
3	Analog equation identifier (group/sub-group) which controls (effects) this discrete
4	Critical pick-up threshold for analog in word 3 (pick-up value)
5	Critical drop-out threshold for analog in word 3 (drop-out value)
	bits 3-5/bits 6-23
6	1st related equation (P.C./REL.ADDR.)
.	
.	
P	Last related equation (P.C./REL.ADDR.)

*Threshold discrete equations will always have a segment number (J) equal to one. Reference Appendix B of this document for an example. All threshold discrete equations will have a minimum of 6 words (words 0-5); the only variable within their format will be the number of related equations (words 6-P).

Figure 4, Section 2

APPENDIX A

Subroutine	Description	Cycle Time
REAL-TIME DRIVER	The Real-Time Driver determines and directs all processing during a real-time simulation with the exception of the interrupt servicing subroutines. The Driver references primary subroutines which in turn reference secondary, etc., subroutines in order to evaluate a transfer equation. Primary subroutines referenced are as follows:	
Primary Subroutines		
STEP 1: ACKLQUE	Move one equation disc address from the clock queue to the special queue provided the timed queue is empty.	20 if Timed queue is not empty. 220 if Timed queue is empty.
	The Clock queue contains the disc address of those timed equations whose timers have expired (elapsed to zero). The expiration of a timer is controlled by an interrupt and is processed to the Clock queue by the interrupt servicing subroutine AELTIME contained on page 7 of this Appendix.	
STEP 2: BHISTINT	Pseudo interrupt servicing subroutine for history tape output.	33
STEP 3: BHISTOUT	History tape output routine.	
or	NO BUFFER TO OUTPUT.	or 40
	BUFFER READY AND OUTPUT STARTED.	or 120

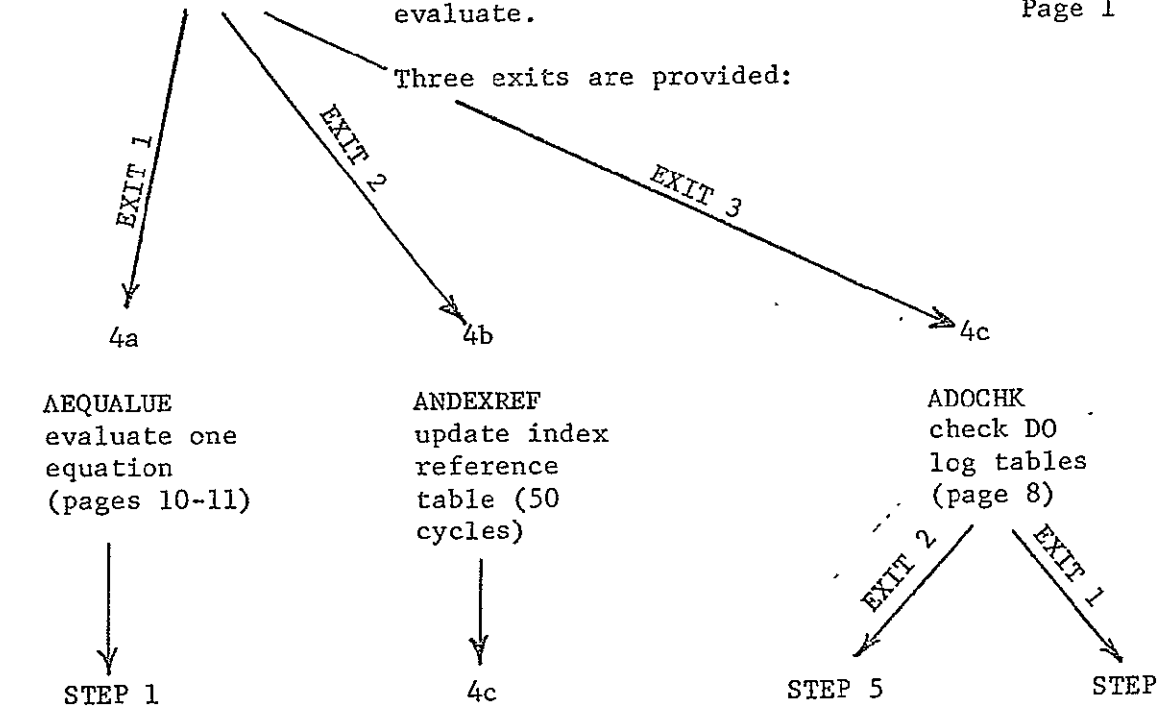
Primary Subroutines

Cycle Time

STEP 4: EQINCORE — Find or input equation to evaluate.

Appendix A
Page 1

Three exits are provided:



STEP 5: ASWTCKH — Input simulated switch from card reader provided BPT1 is set and reader is READY.

Appendix A
Page 12

STEP 1 - Recycle all steps continuously until termination breakpoint is SET by test conductor.

Subroutine	Description
EQINCORE	Checks the queues for equations to evaluate; or equations to input from the RAD; or input of a cross reference block from the RAD.

Parameters		Cycle Time
X_{A1}	Check queues for equation to input	t_{CHKEQ}
X_{B1}	Select highest priority equation from disc address I/O table	t_{UPACK}
X_{C1}	Initiate input from the disc	30
X_{D1}	Check indicated queue for completion of input from the disc	30

$$t_{EQINCORE} = t_{CHKEQ} + t_{UPACK} + X_{C1} + X_{D1}$$

Note: Parameters X_{C1} and X_{D1} are applicable only when the desired equation is not resident in memory.

Note: Three exits from this subroutine are provided:

Exit 1 is taken if an equation is in memory and ready for evaluation.

Exit 2 is taken if an index block of cross reference was previously input from the RAD.

Exit 3 is taken if all queues are empty; no equation to input or evaluate and no index block input.

Subroutine	Description
CHKEQ	Check queues for an equation to be input

Parameters		Cycle Time
X_{A2}	Check for an open slot in the disc address I/O table which corresponds to the indicated queue.	45 per queue
X_{B2}	Select the next equation disc address from the indicated queue.	50
X_{C2}	Determine if this equation is already in memory.	t_{ECORE}
X_{D2}	All queues are empty; no equation to input or evaluate.	35

$$t_{CHKEQ} = N(X_{A2}) + X_{B2} + t_{ECORE} + X_{D2} .$$

Note: N is a queue priority of 1, 2, or 3 for the Special Queue, Timed Queue or Normal Queue, respectively.

Parameter X_{D2} is applicable only when there is no equation to evaluate, i.e., the queues are empty; thus all other parameters are not applicable.

Subroutine	Description
ECORE	Check the appropriate equation buffer based on the equation length to determine if the equation is already in memory.

Parameters		Cycle Time
X_{A3}	Parameter setup each time ECORE is entered.	47
X_{B3}	Time consumed for each buffer checked with a minimum of 1 buffer to a maximum of 50 for the Instrument Unit of the Saturn V.*	30 per buffer
X_{C3}	Time consumed when a match is found (equation in memory).	33
X_{D3}	Time consumed when no match is found (equation not in memory) but a buffer is available for its input.	25

$$t_{ECORE} = X_{A3} + N(X_{B3}) + X_{C3} + X_{D3}$$

Note: Only parameter X_{C3} or X_{D3} is applicable at any one time.

* N is a variable parameter which can be defined only at run time based on the size of the data base (number of equations) and the amount of core available for equation buffers. Equation lengths are rounded up to the nearest unit of ten by the processing program (Initialization Phase). The test conductor establishes the exact number of buffers of a particular length via card input to the Initialization Phase program.

The number and length of equation buffers established for the IU stage and used in this analysis is as follows:

Number (N)	Length
10	20
50	30
40	40
20	50
10	60
5	70
4	80
4	90
5	100
4	200
3	300
1	800

Subroutine	Description
CALOG	Evaluate analog portion (within slash "/" marks) of an analog equation.

Parameters		Cycle Time
X_{A4}	parameter setup	40
X_{B4}	pick-up E type parameters	225
X_{C4}	pick-up F type parameters	86
X_{D4}	calculate interval time and set-up parameters	100
X_{E4}	adjust interval time if initial evaluation of analog	45
X_{F4}	adjust interval time not initial evaluation of of analog	158
X_{G4}	evaluate analog type:	
	P) Polynomial	351
	T) Normal non-additive	1225
	C) Cyclic non-additive	1952
	A) Normal additive	1385
	B) Cyclic additive	2114
	G) Summation	586
	H) Multiplication	405
	U) Update	290
X_{H4}	store evaluation results	139 E type 58 F type

$$t_{CALOG} = X_{A4} + X_{B4} + X_{C4} + X_{D4} + X_{E4} + X_{F4} + X_{G4} + X_{H4}$$

Note: Only parameter X_{B4} or X_{C4} is applicable at any one time based upon whether the analog equation is an E type or F type equation.

Only parameter X_{E4} or X_{F4} is applicable at any one time. Parameter X_{E4} is applicable the first time the analog is evaluated; thereafter, for each successive evaluation, parameter X_{F4} is applicable.

Parameter X_{H4} is based upon the selection of X_{B4} for E type equations or X_{C4} for F type equations.

Subroutine	Description
UPACK	Determine (1) the highest priority disc address to be input and (2) the approximate number of words to be input.

Parameters		Cycle Time
X_{A5}	Find the highest priority disc address in the disc address I/O table.	10
X_{B5}	Compute approximate word count for the desired disc address.	70

$$t_{UPACK} = N(X_{A5}) + X_{B5}$$

Note: N is a number 1, 3, or 5 depending upon the queue from which the disc address was taken - Special Queue, Timed Queue or Normal Queue, respectively.

Subroutine	Description
BDCINRPT	Disc interrupt routine

Parameters		Cycle Time
X_{A6}	Determine exact equation word count (length) and compute its intended memory address.	62
X_{B6}	Transfer 1 word to intended memory address.	18
X_{C6}	Compress disc address I/O tables by one entry each.	53
X_{D6}	Branch to the UPACK subroutine to select the next highest priority disc address to be input.	t_{UPACK}
X_{E6}	Initiate next input if any.	26

$$t_{BDCINRPT} = X_{A6} + N(X_{B6}) + X_{C6} + t_{UPACK} + X_{E6}$$

Note: N is the number of words in the equation.

Subroutine	Description
AELTIME	Priority interrupt servicing subroutine for system level interrupt 211. Responsible for processing the expiration of equation timers.

Parameters		Cycle Time
X_1 (AELTIME)	Time required to process the expiration of a timer associated with an analog equation (interval timer).	153
X_2 (AELTIME)	Time required to process the expiration of a pick-up or drop-out timer.	120

RMILCLK	Priority interrupt servicing subroutine for system level interrupt 210. Responsible for processing the millisecond clock interrupt.

Parameter		Cycle Time
X_1 (RMILCK)	Time required to update (add one) to internal real-time clock.	30

Subroutine	Description
ADOCHK	Check DO log table for a DO word from the RCA-110A to process.
ACOMPDO	Process DO word from log table.

	Parameters	Cycle Time	
X_{A8}	Check for DO word in log table.	65	
X_{B8}	Check word for validity.	61.5	
X_{C8}	Obtain previous status of this DO word from DO status table.	84.5	
X_{D8}	Check for a status change (bit change) beginning with least significant bit position (bit 24-right side).	10	per bit checked
X_{E8}	Check for associated index in memory <u>if</u> bit changed.	15	per bit change
X_{F8}	Log related equations into normal queue <u>provided</u>: 1) bit changed and 2) index is in memory.	246.5	
X_{G8}	Store new status bit in old status word <u>if</u> bit changed.	65	per bit change
X_{H8}	Store previous and present DO status word in history buffer.	55	
X_{I8}	Associated index <u>not</u> in memory; disc address of index to normal queue. DO word back to log table.	163.5	

$$t_{\substack{ADOCHK \\ ACOMPDO}} = X_{A8} + X_{B8} + X_{C8} + Q(X_{D8}) + R(X_{E8}) + R(X_{F8}) + R(X_{G8}) + X_{H8} + X_{I8}$$

Q = Number of bits interrogated (1 - 24)

R = Number of interrogated bits which had a change in status (1 - 24)

Note: Parameter X_{I8} is applicable only when the index of related equations associated with a bit change is not in memory. Exit 1 is taken.

If this condition occurs, parameter X_{H8} would not be applicable until the index was brought into memory and the entire DO word of 24 bits had been checked for changes.

The unfinished DO word is restored in the log table until all 24 bits have been checked.

Those bits, if any, which had a status change and had their associated index in memory will be rechecked; however, parameter X_{G8} will have previously been applicable and their new status will be reflected in the previous (old) DO status word. Thus, during subsequent rechecked of the same DO word, previously checked bits which changed and were completely processed will have effect only on the coefficient R for X_{D8}.

Considering the number of DO's (discrete outs) which were active during IU simulation, it is safe to assume that the associated index for any bit change is in memory.

Note: Two exits from this subroutine are provided:

Exit 1 is taken if the DO log table is not empty.

Exit 2 is taken if the DO log table is empty; only parameter X_{A8} is applicable in this case.

Subroutine	Description
ASWTCHK	Input simulated switch from the card reader.

Parameters		Cycle Time
X_{A9}	Initial setup each entry.	53
X_{B9}	Determine if associated switch index is in memory.	45
X_{C9}	Determine where the equations related to the switch are located in the index block.	75
X_{D9}	Update the status of the switch and place previous and present status in history buffer.	83
X_{E9}	Place related equation disc addresses from index block into normal queue	200 + N(30) N = # of related equation disc addresses

$$t_{ASWTCHK} = X_{A9} + X_{B9} + X_{C9} + X_{D9} + X_{E9}$$

Note: If the associated switch index block is not in memory, only parameters X_{A9} and X_{B9} are applicable.

If the above condition occurs, the subroutine will not input another simulated switch until the index block has been brought into memory and processing of the switch through all parameters has been completed.

Subroutine	Description
AEQVALUE	Controls the evaluation of a transfer equation.

Common Parameters		Cycle Time
X_{A10}	Initial processing each entry.	73
X_{B10}	Determine desired status of one status variable.	110 per variable
X_{C10}	Process one "+" (logical "OR") operator.	43 per "+"
X_{D10}	Process a "$" (end of equation) operator.	38
X_{E10}	Place related equation disc address(es) into specified queue if: 1) equation changed in status and/or 2) reached a critical value (pick-up/drop-out value).	N(30) + 100 N = number of related equation disc addresses

Discrete Parameters		Cycle Time
X_{F10}	Store evaluation results in appropriate status table and previous and present status in a history buffer.	140
X_{G10}	Set any associated pick-up/drop-out timers in clock cells	N(67) + 37 N = Number of associated timers.
X_{H10}	Determine if pick-up or drop-out timer associated with timed discrete has expired.	N(4) + 70 N = Number of clock cells active.

$$t_{\substack{DISCRETE \\ AEQVALUE}} = X_{A10} + M(X_{B10}) + P(X_{C10}) + X_{D10} + X_{E10} + X_{F10} + X_{G10} + X_{H10}$$

M = Number of status variables checked

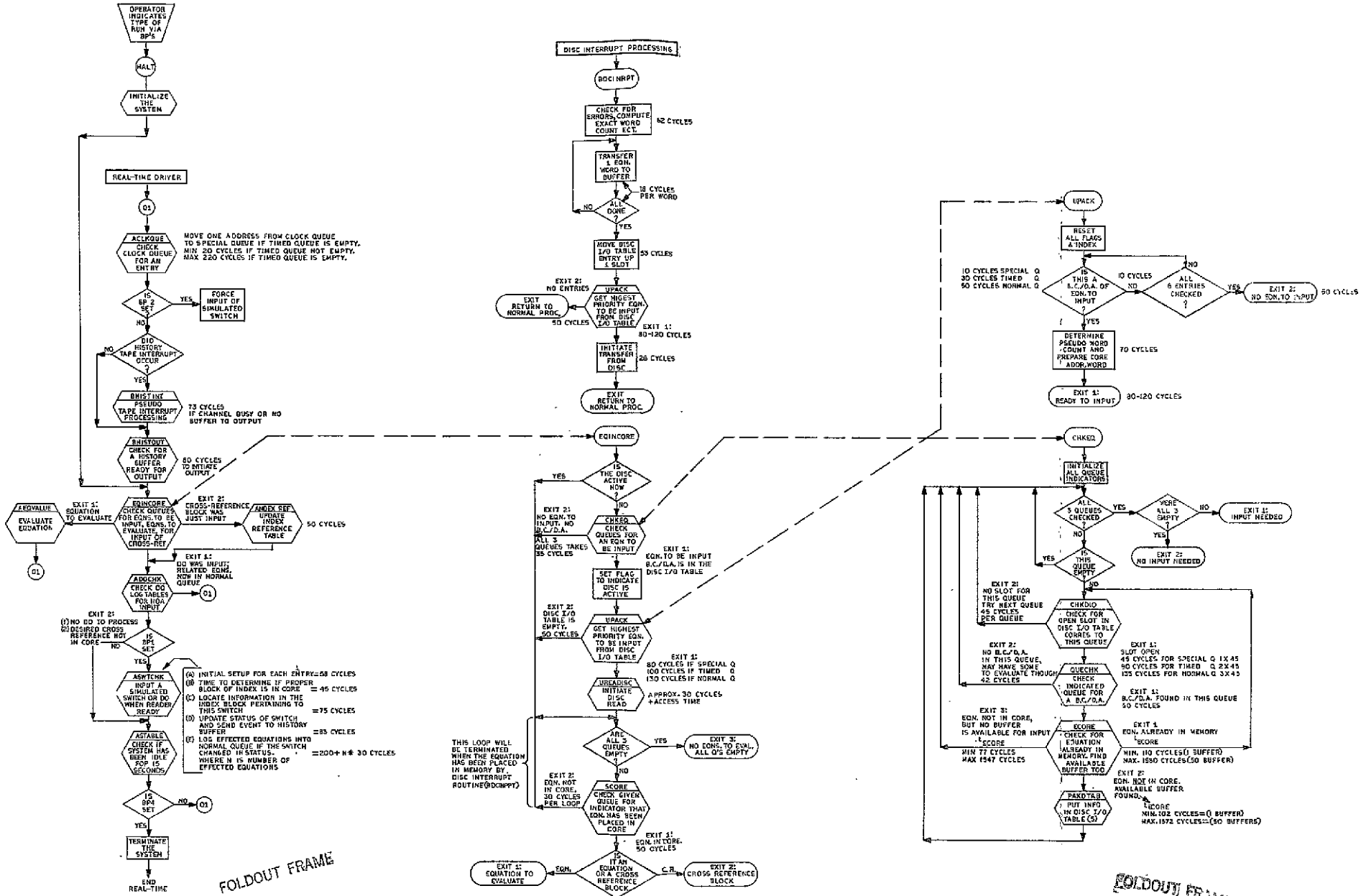
P = Number of "+" operators processed.

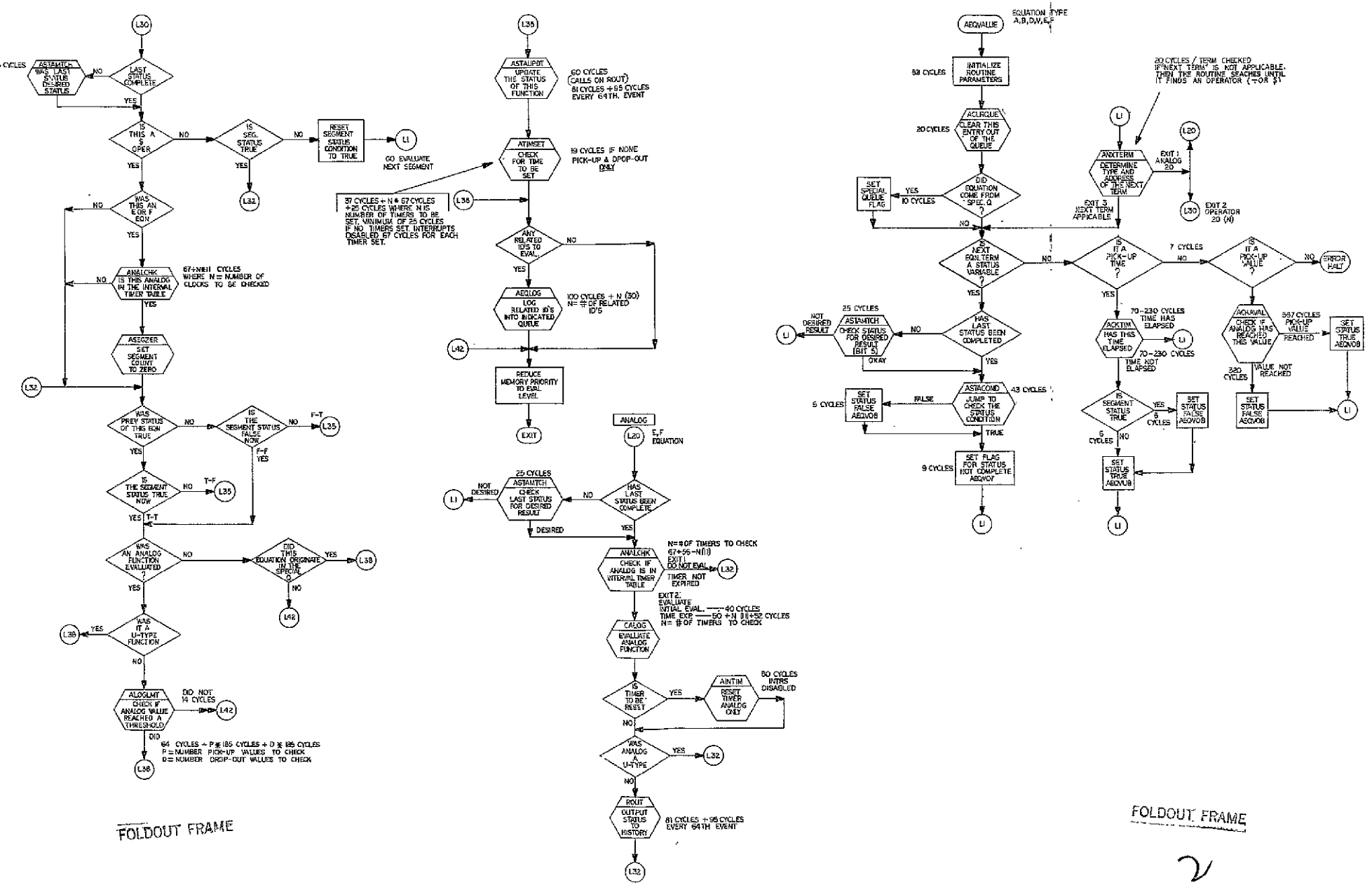
Subroutine	Description
AEQVALUE	Continued.

Analog Parameters		Cycle Time
X_{I11}	Pick-up analog equation information from secondary (analog) timer tables.	N(11) + 102 N = Number of entries before a match is found.
X_{J11}	Evaluate analog portion within slash ("/") marks only.	t_{CALOG}
X_{K11}	Reset analog timer.	80
X_{L11}	Store analog previous and present value in history buffer.	83
X_{M11}	Determine if analog reached a critical value; P = # of pick-up values, D = # of drop-out values.	64 + P(185) or 64 + D(185)

$$t_{\substack{ANALOG \\ AEQVALUE}} = X_{A10} + M(X_{B10}) + P(X_{C10}) + X_{D10} + X_{E10} + X_{I11} + t_{CALOG} + X_{K11} + X_{L11} + X_{M11}$$

Subroutine	Description	Cycle Time
HFSA	Floating point addition	114.0
FHSS	Floating point subtraction	136.0
HFSM	Floating point multiply	100.0
HFSC	Floating point divide	140.5
CCOTFP	DDAS counts to floating point	78
CFPTCO	Floating point to DDAS counts	29
FTB	Floating point to binary	42
CMINMAX	Check floating point limits	42


OPERATOR INDICATES TYPE OF RUN VIA BP's
HALT
INITIALIZE THE SYSTEM
REAL-TIME DRIVER
01
ACLKQUE
CHECK CLOCK QUEUE FOR AN ENTRY
MOVE ONE ADDRESS FROM CLOCK QUEUE TO S PECIAL QUEUE IF TIMED QUEUE IS EMPTY. MIN. 20 CYCLES IF TIMED QUEUE NOT EMPTY. MAX 220 CYCLES IF TIMED QUEUE IS EMPTY.
IS BP 2 SET
YES
FORCE INPUT OF SIMULATED SWITCH
NO
DIO HISTORY TAPE INTERRUPT OCCUR ?
NO
YES
BHISTINT
PSEUDO TAPE INTERRUPT PROCESSING
73 CYCLES IF CHANNEL BUSY OR NO BUFFER TO OUTPUT
BHISTOUT
CHECK FOR A HISTORY BUFFER READY FOR OUTPUT
80 CYCLES TO INITIATE OUTPUT
EQINCORE
CHECK QUEUES FOR EQNS. TO BE INPUT, EQNS. TO EVALUATE, FOR INPUT OF CROSS-REF
EXIT 1: EQUATION TO EVALUATE
EXIT 2: CROSS-REFERENCE BLOCK WAS JUST INPUT
XNDEX REF
UPDATE INDEX REFERENCE TABLE
50 CYCLES
EXIT 3: DO WAS INPUT; RELATED EQNS. NOW IN NORMAL QUEUE
EQVALUE
EVALUATE EQUATION
01
ADOCHK
CHECK DO LOG TABLES FOR HOA INPUT
01
EXIT 2: (1) NO DO TO PROCESS (2) DESIRED CROSS REFERENCE NOT IN CORE
IS BP3 SET
NO
YES
ASWTCHK
INPUT A SIMULATED SWITCH OR DO WHEN READER READY
(A) INITIAL SETUP FOR EACH ENTRY=58 CYCLES
(B) TIME TO DETERMINE IF PROPER BLOCK OF INDEX IS IN CORE =45 CYCLES
(C) LOCATE INFORMATION IN THE INDEX BLOCK PERTAINING TO THIS SWITCH =75 CYCLES
(D) UPDATE STATUS OF SWITCH AND SEND EVENT TO HISTORY BUFFER =85 CYCLES
(E) LOG EFFECTED EQUATIONS INTO NORMAL QUEUE IF THE SWITCH CHANGED IN STATUS. WHERE N IS NUMBER OF EFFECTED EQUATIONS =200+ N* 30 CYCLES
ASTABLE
CHECK IF SYSTEM HAS BEEN IDLE FOR 15 SECONDS
IS BP4 SET
NO
01
YES
TERMINATE THE SYSTEM
END REAL-TIME
FOLDOUT FRAME
DISC INTERRUPT PROCESSING
BDCINRPT
CHECK FOR ERRORS, COMPUTE EXACT WORD COUNT ECT.
62 CYCLES
TRANSFER 1 EQN. WORD TO BUFFER
18 CYCLES PER WORD
ALL DONE ?
NO
YES
MOVE DISC I/O TABLE ENTRY UP 1 SLOT
55 CYCLES
EXIT 2: NO ENTRIES
UPACK
GET HIGEST PRIORITY EQN. TO BE INPUT FROM DISC I/O TABLE
EXIT 1: 80-120 CYCLES
EXIT RETURN TO NORMAL PROC.
50 CYCLES
INITIATE TRANSFER FROM DISC
26 CYCLES
EXIT RETURN TO NORMAL PROC.
EQINCORE
IS THE DISC ACTIVE NOW ?
YES
NO
CHKEQ
CHECK QUEUES FOR AN EQN TO BE INPUT
EXIT 2: NO EQN. TO INPUT, NO B.C./D.A. ALL 3 QUEUES TAKES 35 CYCLES
EXIT 1: EQN. TO BE INPUT B.C./D.A. IS IN THE DISC I/O TABLE
SET FLAG TO INDICATE DISC IS ACTIVE
EXIT 2: DISC I/O TABLE IS EMPTY. 50 CYCLES
UPACK
GET HIGHEST PRIORITY EQN. TO BE INPUT FROM DISC I/O TABLE
EXIT 1: 80 CYCLES IF SPECIAL Q 100 CYCLES IF TIMED Q 130 CYCLES IF NORMAL Q
UREADISC
INITIATE DISC READ
APPROX. 30 CYCLES +ACCESS TIME
THIS LOOP WILL BE TERMINATED WHEN THE EQUATION HAS BEEN PLACED IN MEMORY BY DISC INTERRUPT ROUTINE(BDCINPT)
ARE ALL 3 QUEUES EMPTY ?
YES
EXIT 3: NO EQNS. TO EVAL. ALL Q'S EMPTY
NO
EXIT 2: EQN. NOT IN CORE, 30 CYCLES PER LOOP
SCORE
CHECK GIVEN QUEUE FOR INDICATOR THAT EQN. HAS BEEN PLACED IN CORE
EXIT 1: EQN. IN CORE. 50 CYCLES
IS IT AN EQUATION OR A CROSS REFERENCE BLOCK
EQN.
EXIT 1: EQUATION TO EVALUATE
C.R.
EXIT 2: CROSS REFERENCE BLOCK
UPACK
RESET ALL FLAGS & INDEX
IS THIS A B.C./D.A. OF EQN. TO INPUT ?
10 CYCLES SPECIAL Q 30 CYCLES TIMED Q 50 CYCLES NORMAL Q
10 CYCLES
NO
YES
ALL 6 ENTRIES CHECKED ?
NO
YES
EXIT 2: NO EQN. TO INPUT
50 CYCLES
DETERMINE PSEUDO WORD COUNT AND PREPARE CORE ADDR. WORD
70 CYCLES
EXIT 1: READY TO INPUT
80-120 CYCLES
CHKEQ
INITIALIZE ALL QUEUE INDICATORS
ALL 5 QUEUES CHECKED ?
YES
NO
WERE ALL 3 EMPTY ?
NO
YES
EXIT 1: INPUT NEEDED
EXIT 2: NO INPUT NEEDED
IS THIS QUEUE EMPTY ?
YES
NO
CHKDIO
CHECK FOR OPEN SLOT IN DISC I/O TABLE CONRES TO THIS QUEUE
EXIT 2: NO SLOT FOR THIS QUEUE TRY NEXT QUEUE 45 CYCLES PER QUEUE
EXIT 1: SLOT OPEN 45 CYCLES FOR SPECIAL Q 1X45 90 CYCLES FOR TIMED Q 2X45 135 CYCLES FOR NORMAL Q 3X45
QUECHK
CHECK INDICATED QUEUE FOR A B.C./D.A.
EXIT 2: NO B.C./D.A. IN THIS QUEUE, MAY HAVE SOME TO EVALUATE THOUGH. 42 CYCLES
EXIT 1: B.C./D.A. FOUND IN THIS QUEUE 50 CYCLES
ECORE
CHECK FOR EQUATION ALREADY IN MEMORY. FIND AVAILABLE BUFFER TOO
EXIT 3: EQN. NOT IN CORE, BUT NO BUFFER IS AVAILABLE FOR INPUT
ECORE
MIN 77 CYCLES MAX 1547 CYCLES
EXIT 4: EQN. ALREADY IN MEMORY
ECORE
MIN. 110 CYCLES(1 BUFFER) MAX. 1590 CYCLES(30 BUFFER)
EXIT 2: EQN. NOT IN CORE. AVAILABLE BUFFER FOUND.
PAKDTAB
PUT INFO IN DISC I/O TABLE (S)
ECORE
MIN. 102 CYCLES=(1 BUFFER) MAX. 1572 CYCLES=(30 BUFFERS)
FOLDOUT FRAME
2


L30
25 CYCLES
ASTAMTCH
WAS LAST STATUS DESIRED STATUS
NO
LAST STATUS COMPLETE
YES
IS THIS A S OPER
NO
IS SEG. STATUS TRUE
NO
RESET SEGMENT STATUS CONDITION TO TRUE
L1
GO EVALUATE NEXT SEGMENT
YES
L32
WAS THIS AN E OR F EQN
NO
YES
ANALCHK
IS THIS ANALOG IN THE INTERVAL TIMER TABLE
NO
67+N‡11 CYCLES WHERE N = NUMBER OF CLOCKS TO BE CHECKED
YES
ASEGZER
SET SEGMENT COUNT TO ZERO
L32
WAS PREV STATUS OF THIS EQN TRUE
NO
IS THE SEGMENT STATUS FALSE NOW
NO
F-T
L35
YES
F-F
YES
IS THE SEGMENT STATUS TRUE NOW
NO
T-F
L35
YES
T-T
WAS AN ANALOG FUNCTION EVALUATED ?
NO
DID THIS EQUATION ORIGINATE IN THE SPECIAL Q
YES
L38
NO
L42
YES
L38
WAS IT A U-TYPE FUNCTION
NO
ALOGLMT
CHECK IF ANALOG VALUE REACHED A THRESHOLD
DID NOT
14 CYCLES
L42
DID
64 CYCLES + P‡185 CYCLES + D‡185 CYCLES
P = NUMBER PICK-UP VALUES TO CHECK
D = NUMBER DROP-OUT VALUES TO CHECK
L38
FOLDOUT FRAME
L38
ASTAUPDT
UPDATE THE STATUS OF THIS FUNCTION
60 CYCLES (CALLS ON ROUT)
81 CYCLES + 95 CYCLES EVERY 64TH. EVENT
ATIMSET
CHECK FOR TIME TO BE SET
19 CYCLES IF NONE
PICK-UP & DROP-OUT ONLY
37 CYCLES + N‡67 CYCLES + 25 CYCLES WHERE N IS NUMBER OF TIMERS TO BE SET. MINIMUM OF 25 CYCLES IF NO TIMERS SET. INTERRUPTS DISABLED 67 CYCLES FOR EACH TIMER SET.
L38
ANY RELATED ID'S TO EVAL.
NO
YES
AEQLOG
LOG RELATED ID'S INTO INDICATED QUEUE
100 CYCLES + N (30)
N= # OF RELATED ID'S
L42
REDUCE MEMORY PRIORITY TO EVAL LEVEL
EXIT
ANALOG
L20
E,F EQUATION
25 CYCLES
ASTAMTCH
CHECK LAST STATUS FOR DESIRED RESULT
NOT DESIRED
L1
NO
HAS LAST STATUS BEEN COMPLETE
YES
DESIRED
ANALCHK
CHECK IF ANALOG IS IN INTERVAL TIMER TABLE
N=# OF TIMERS TO CHECK
67+56-N(11)
EXIT1
DO NOT EVAL
TIMER NOT EXPIRED
L32
EXIT2
EVALUATE
INITIAL EVAL. —— 40 CYCLES
TIME EXP —— 50 +N (11+52 CYCLES
N= # OF TIMERS TO CHECK
CALOG
EVALUATE ANALOG FUNCTION
IS TIMER TO BE RESET
YES
AINTIM
RESET TIMER ANALOG ONLY
80 CYCLES INTRS DISABLED
NO
WAS ANALOG A U-TYPE
YES
L32
NO
ROUT
OUTPUT STATUS TO HISTORY
81 CYCLES +95 CYCLES EVERY 64TH EVENT
L32
AEQVALUE
EQUATION TYPE A,B,D,W,E,F
58 CYCLES
INITIALIZE ROUTINE PARAMETERS
20 CYCLES
ACLRQUE
CLEAR THIS ENTRY OUT OF THE QUEUE
DID EQUATION COME FROM SPEC Q ?
YES
10 CYCLES
SET SPECIAL QUEUE FLAG
NO
IS NEXT EQN TERM A STATUS VARIABLE ?
NO
YES
HAS LAST STATUS BEEN COMPLETED
NO
25 CYCLES
ASTAMTCH
CHECK STATUS FOR DESIRED RESULT (BIT 5)
NOT DESIRED RESULT
L1
OKAY
YES
ASTACOND
JUMP TO CHECK THE STATUS CONDITION
43 CYCLES
FALSE
5 CYCLES
SET STATUS FALSE AEQVOB
TRUE
9 CYCLES
SET FLAG FOR STATUS NOT COMPLETE AEQVOY
L1
20 CYCLES / TERM CHECKED
IF "NEXT TERM" IS NOT APPLICABLE, THEN THE ROUTINE SEACHES UNTIL IT FINDS AN OPERATOR (-OR $)
L1
ANDTERM
DETERMINE TYPE AND ADDRESS OF THE NEXT TERM
EXIT 1 ANALOG 20
L20
EXIT 2 OPERATOR 20 (N)
L30
EXIT 3 NEXT TERM APPICABLE
IS IT A PICK-UP TIME ?
7 CYCLES
NO
YES
ACKTIM
HAS THIS TIME ELAPSED
70-230 CYCLES TIME HAS ELAPSED
L1
70-230 CYCLES TIME NOT ELAPSED
IS SEGMENT STATUS TRUE
YES
5 CYCLES
SET STATUS FALSE AEQVOB
5 CYCLES
NO
SET STATUS TRUE AEQVOB
L1
IS IT A PICK-UP VALUE ?
NO
ERROR HALT
YES
ACKVAL
CHECK IF ANALOG HAS REACHED THIS VALUE
567 CYCLES PICK-UP VALUE REACHED
SET STATUS TRUE AEQVOB
320 CYCLES
VALUE NOT REACHED
SET STATUS FALSE AEQVOB
L1
FOLDOUT FRAME
2

APPENDIX B

An example of each of the four major groups (types) of transfer equations within the MARDSLVC system are as follows:

1) Analog Equation

```
E100 = A1 * B1/C, 0, 0, 2500, 1000, 5000, 2000, I1000, M(200000)/

+ E500 + E200/G, 1.0, E500, -1.0, E200, M5000, NO, I1000, M(100000)/$
```

2) Timed Discrete

```
V100 = V99(2000,0)$
```

The equation V100 is true if V99 is true and has been true (timed) for a minimum of 2000 milliseconds. V100 is false whenever V99 is false (time of 0 milliseconds).

3) Non-timed discrete

```
A101 = V101 * B200 * Y1 + V101 * B201$
```

4) Threshold discrete (Analog internal variable)

```
V200 = E100//5000, 2500//$
```

The equation V200 is true when the value (voltage) for E100 is equal to or greater than 5000 millivolts.
V200 is false when E100 is equal to or less than 2500 millivolts.

GENERAL PURPOSE SIMULATOR SYSTEM STUDY

PART 3

Report No. SP-209-0339-2

DATE: October 5, 1970

General Purpose Simulator System Study Part 3

Prepared for

George C. Marshall Space Flight Center
Contract No. NAS8-25608

Prepared by:

Space Support Division
Division of Sperry Rand Corporation
Huntsville, Alabama

Submitted by: J. Scoggins
J. Scoggins

and

K. J. Taras
K. J. Taras

Approved by: R. A. Rossler
R. A. Rossler

TABLE OF CONTENTS

1.0 Scope

This report discusses the capabilities of the DEE-6 Simulator System based on the solution times of the equations. For the same data base this report can serve as this basis to determine the capabilities required for future simulation systems.

2.0 Introduction

In this report, a mathematical analysis of the DEE-6 simulator is made to establish its capability. The mathematical analysis is based on the Computer Sciences Corporation timed subroutines and the flow charts of the real-time program listed in Appendix A and B respectively of Sperry Rand's "General Purpose Simulator System Study" Part 2.

To minimize the assumptions and thereby reasonably predict the capabilities of the simulator, Sperry Rand proceeded with an extensive study to classify the data base of the Instrument Unit. The data base was first classified by equation type. For each equation type the length of every equation was determined and grouped into logical class intervals. For each class interval the associated number of equations, the average number of OR terms per equation, and the average number of cross references per equation were established.

A computer program was written to fit a least square polynomial curve through the data. A sample of the program is shown in Appendix A of this report. The computer program calculated and printed out the coefficients for up to a fourth order polynomial. It also calculated the correlation between the data and terminated the calculation of the coefficients whenever a correlation of better than 98 percent was found.

The classified data are also used to calculate the mean equation length and the memory requirements of the various equation types. The mean equation length and its corresponding number of OR terms and number of cross references read from the curves are substituted into the applicable subroutines to determine the solution time for each equation type.

Figure 2-1a and b are simplified flow charts for the equation solution times. Figure 2-1a is the flow chart used when the equation is located in the RAD and Figure 2-1b is the flow chart used assuming the equation of interest is located in resident memory. The equation solution times

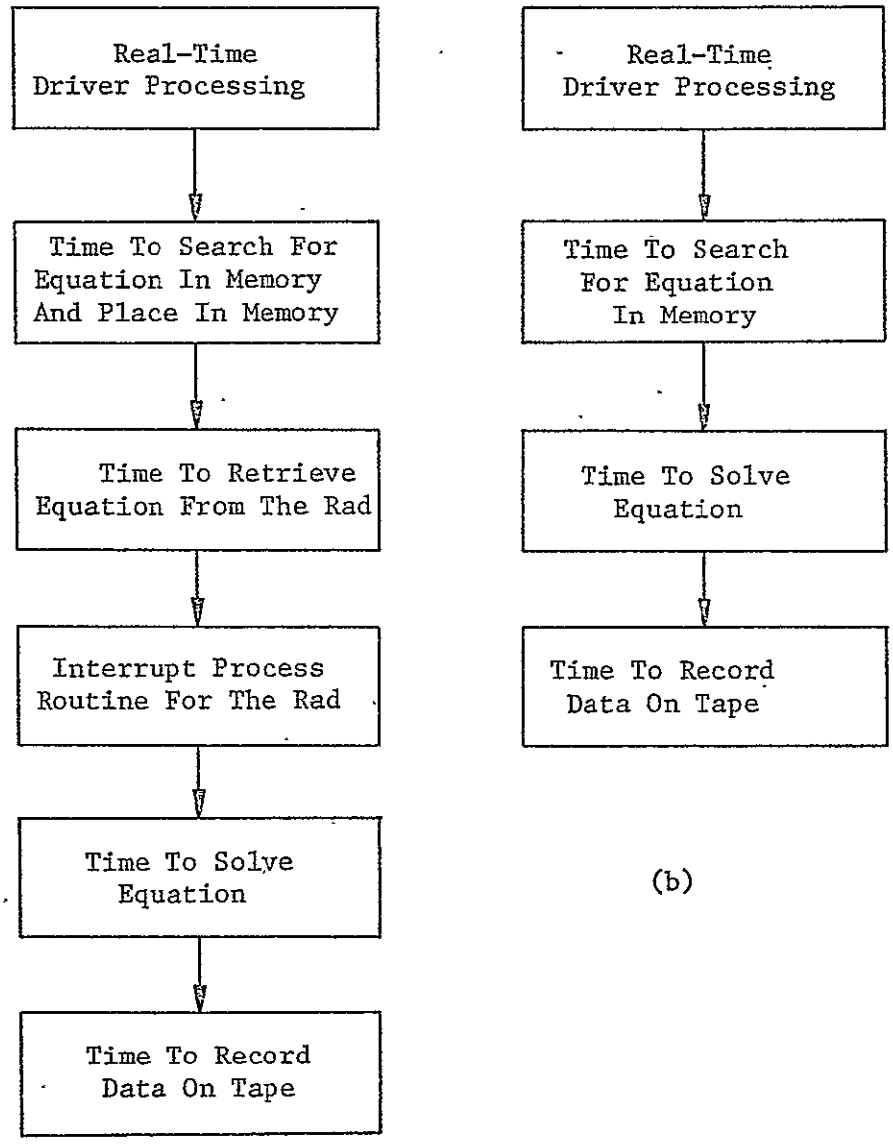


Figure 2-1 Equation Solution Time Flow Chart

(a) Equation Located In Rad

(b) Equation Located In Resident Memory

are then used to obtain a measure of the systems response time to a DO from RCA 110-A computer and to a switch action simulated by the card reader. From the equation solution times and system responses conclusions are drawn about the DEE-6 simulator capabilities.

3.0 Mathematical Analysis

In this section the solution time for each equation type is calcualted. The solution times are based on the Computer Sciences Corporation timed subroutines. The flow chart of the real-time problem shown in Appendix B of Sperry Rand's "General Purpose Simulator System Study" Part 2, and the Statistics of the Instrument Unit data base.

3.1 A Type Discrete Transfer Equations

3.1.1 Equation mean length : $\bar{X}_A = \frac{1}{NA} \sum_i X_{Ai} F_{Ai} = 11.23$ or 11 words

Where NA = 466 The total number of A type equations

X_{Ai} The median of the ith class interval (See Table 3.1)

F_{Ai} The number of A type equations within the ith interval (see Table 3.1)

3.1.2 Memory requirements: Number of words $= \sum_i (X_{Ai} - Y_{Ai}) F_{Ai} = 5254$ words

Where Y_{Ai} The average number of cross references in the ith class interval

3.1.3 Number of OR terms (A_{OR}) per transfer equation based on an equation mean length of 11 words (See curve B figure 3.1): $A_{OR} = 1$

3.1.4 Number of cross reference (A_{CR}) per transfer equation based on an equation mean length of 11 words: $A_{CR} = 0$

3.1.5 Number of status variables (Nst)/OR term

$$= \frac{\bar{X}_A - A_{CR} - O_A - 8}{A_{OR}} = 2$$

Where $O_A = 1$ Number of operators within the transfer equation

3.1.6 Time required to process an A type equation when the equation is not in resident memory.

3.1.6.1 $t_{eqincore} = 1.75 \times 10^{-6} (45m + 30n + 10k + 253) = 1.29$ milliseconds

Table 3.1 A Type Discrete Transfer Equations Grouped Data

Equation Length (Words) X_A	Frequency Of Occurance F_A	Average No. of OR Terms	Average No. of Cross References Y_A
10-11	404	1	0
12-13	45	1	0
14-15	9	2	0.8
16-17	6	3	0
18-19	2	1	7.5
Total	466		

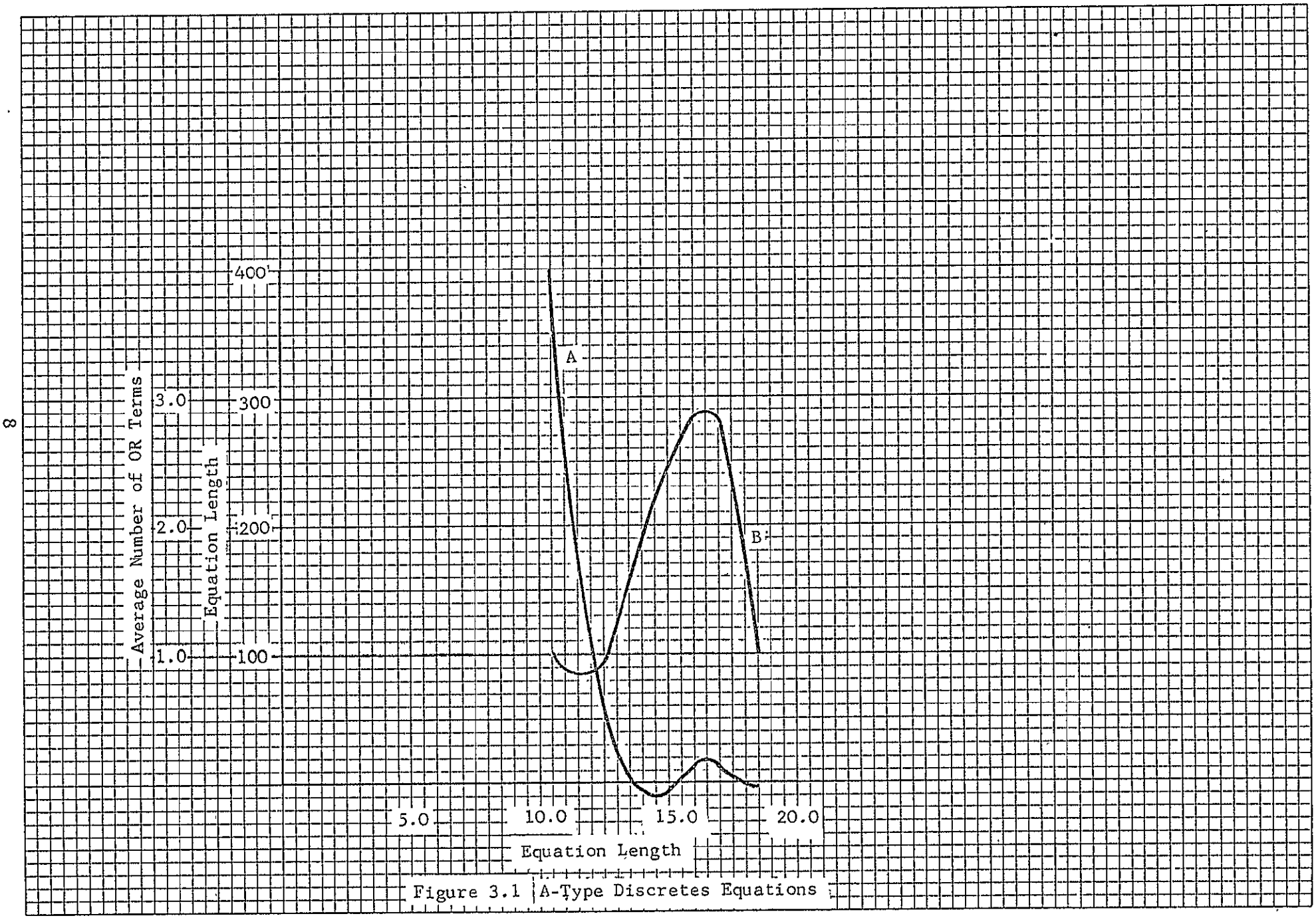


Figure 3.1 A-Type Discretes Equations

Where $t_{eqincore}$ is the time it takes the simulator to search for an equation to evaluate. It is obtained by combining the equations of the Equincore, Chkeq, Ecore, and Upack subroutines listed in Appendix A of Sperry Rand's "General Purpose Simulator System Study" Progress Report 2.

$m = 3$ The queue priority

$n = 10$ The buffer size alloted based on an equation mean length of less than 20 words

$k = 5$ The disc address queue priority

$t_{main\ driver} = 0.107$ millisecond (See Main Driver Subroutine listed in Appendix A of Sperry Rand's "General Purpose Simulator Study" Part 2.

3.1.6.2 $t_{bdcinrpt} = 1.75 \times 10^{-6} (18 \times +10k +211) = 0.803$ millisecond

Where $t_{bdcinrpt}$ is the time spent in the disc interrupt subroutine (For details see Appendix A of Sperry Rand's General Purpose Simulator System Study" Progress Report 2.)

$X = \bar{X}_A = 11$ Equation word length

3.1.6.3 Evaluation time of transfer equation assuming all of the status variables to be true:

$$t_{aeqvalue\ (max)} = (110\ Nst + 30\ A_{CR} + 43r + 100a + t_{\$} + 213)\ 1.75 \times 10^{-6}$$

$$= 1.19 \text{ milliseconds}$$

Where $t_{aeqvalue\ (max)}$ is the maximum time required to evaluate a discrete equation. (For details of the equation see the aeqvalue subroutine listed in Appendix A of Sperry Rand's "General Purpose Simulator System Study" Progress Report 2.)

N_{ST} = 3 The number of status variables checked

A_{CR} = 0 The number of related equations

r = 0 The number of plus operations

a = 1 Transfer equation changed in status

$t_{\$}$ = 38 Computer cycles to process $ operator

3.1.6.4 Minimum evaluation time of transfer equation assuming the first status variable checked is false:

$$t_{aeqvalue\ (min)} = 1.75 \times 10^{-6} (110\ N_{ST} + 30\ A_{CR} + 43r + 100a + t_{\$} + 213)$$

$$= 0.632 \text{ millisecond}$$

N_{ST} = 1 The number of status variables checked

A_{CR} = 0 The number of related equations

r = 0 The number of plus operations

a = 0 Transfer equation did not change in status

$t_{\$}$ = 38 Computer cycles required to process a $ operator

3.1.6.5 t_{tape} = 3.2 milliseconds The average time required to output results on history tape. (For details, see Sperry Rand's "General Purpose Simulator System Study" Progress Report 2.)

3.1.6.6 t_{access} = 17.5 milliseconds The average time required to access an equation from the disc.

3.1.6.7 Maximum time required to search, retrieve, access, evaluate, and output on history tape an A type transfer equation:

$$t_{\substack{A(max) \\ (\text{Equation in rad})}} = t_{\substack{main \\ driver}} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue} + t_{access}$$

$$+ t_{tape} = 24.094 \text{ milliseconds}$$

3.1.6.8 Minimum time required to search, retrieve, access, evaluate, and output on history tape an A type transfer equation:

$$t_{B(min)\ (Equation\ in\ rad)} = t_{main\ driver} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue} + t_{access} + t_{tape}$$

$$t_{B(min)\ (Equation\ in\ rad)} = 23.532 \text{ milliseconds}$$

3.1.7 Time required to process an A type equation when the equation is in memory:

3.1.7.1 Expected time spent to check the equation buffer to see if the equation is in resident memory:

$$E(t)_{Buffer} = P_1 t + 2P_2 t + \ldots n P_n t \qquad 3.1$$

Where $t = 30$ = Computer time in cycles spent for each buffer check

P_i for $i = 1,2,\ldots n$ = The probability that the equation is in the ith buffer.

Assuming $P_1 = P_2 \ldots = P_n = \frac{1}{n}$ then equation 3-1 can be written

$$E(t)_{Buffer} = \frac{t}{n}(1+2+\ldots+n)$$

Substituting the applicable values for n and t the expected time to check the equation buffer becomes:

$E(t)_{Buffer} = 165$ cycles

3.1.7.2 Time required to search the queues and buffer for equation to evaluate:

$$t_{eqincore\ (Equation\ in\ memory)} = 1.75 \times 10^{6} (45m + e(t)_{buffer} + 150) \qquad 3\text{-}2$$

$$= 0.787 \text{ millisecond}$$

Equation 3-2 is obtained by combining the equations for the eqincore, chkeq, and ecore subroutines listed in Appendix A of Sperry Rand's "General Purpose Simulator System Study" Part 2 with the terms pertaining to the retrieval of the equation from the buffer eliminated as before m = 3 is queue priority.

3.1.7.3 $t_{\substack{\text{main} \\ \text{driver}}} = 0.107$ millisecond

3.1.7.4 $t_{\substack{\text{aeqvalue(max)} \\ \left(\text{Equation in memory}\right)}} = 1.19$ milliseconds (See Section 3.1.6.3)

3.1.7.5 $t_{\substack{\text{aeqvalue(min)} \\ \left(\text{Equation in memory}\right)}} = 0.632$ millisecond (See Section 3.1.6.4)

3.1.7.6 $t_{\text{tape}} = 3.2$ milliseconds

3.1.7.7 Maximum time required to search, evaluate, and output on history tape an A type discrete equation when the equation is in resident memory:

$$t_{\substack{\text{A(max)} \\ \left(\text{Equation in resident memory}\right)}} = t_{\substack{\text{main} \\ \text{driver}}} + t_{\substack{\text{eqincore} \\ \left(\text{Equation in memory}\right)}} + t_{\substack{\text{aeqvalue(max)} \\ \left(\text{Equation in memory}\right)}} + t_{\text{tape}}$$

$= 5.284$ milliseconds

3.1.7.8 Minimum time required to search, evaluate, and output on history tape and A type transfer equation when the equation is in resident memory:

$$t_{\substack{\text{A(min)} \\ \left(\text{Equation in memory}\right)}} = t_{\substack{\text{main} \\ \text{driver}}} + t_{\substack{\text{eqincore} \\ \left(\text{Equation in memory}\right)}} + t_{\substack{\text{aeqvalue(min)} \\ \left(\text{Equation in memory}\right)}} + t_{\text{tape}}$$

$$t_{\substack{\text{A(min)} \\ \left(\text{Equation in memory}\right)}} = 4.726 \text{ milliseconds}$$

3.2 B Type Discrete Transfer Equation (Equation not in memory)

3.2.1 Equation mean length: $\bar{X}_B = \frac{1}{333} \sum_{i=1}^{5} X_{Bi} F_{Bi} = 16.59$ words

(See Table 3.2 for X_{Bi}, Y_{Bi} and F_{Bi} values)

3.2.2 Memory requirements: Number of words $= \sum_{i=1}^{5} (X_{Bi} - Y_{Bi}) F_{Bi} = 6333$ words

3.2.3 Number of OR terms (B_{OR}) associated with $\bar{X}_B$: $B_{OR} = 1$ (See curve B figure 3.2)

3.2.4 Number of cross-references (B_{CR}) associated with $\bar{X}_B$: $B_{CR} = 1$. (See curve C figure 3.2)

3.2.5 Number of status variables (N_{ST})/OR term

$$= \frac{\bar{X}_B - B_{CR} - O_B - 8}{B_{OR}} = 8$$

3.2.6 Time required to process a B type equation when equation is not in memory.

3.2.6.1 $t_{\text{main driver}} = 0.107$ millisecond

3.2.6.2 $t_{\text{eqincore}} = 1.75 \times 10^{-6} (45m + 30n + k + 253) = 1.29$ milliseconds.

Where $m = 3$ Normal Queue

$n = 10$ Buffer Size

$k = 5$ Normal Queue

3.2.6.3 $t_{\text{bdcrirpt}} = 1.75 \times 10^{-6} (18 \times X_B + 10_k + 211) = 0.904$ millisecond

Where $X_B = \bar{X}_B = 17$ Words

Table 3-2 B Type Discrete Transfer Equations Grouped Data

Equation Length (Words) X_B	Frequency Of Occurance F_B	Average No. of OR Terms	Average No. of Cross References Y_B
10-19	260	1.36	0.84
20-29	41	1.99	5.34
30-39	20	3.35	10.55
40-49	9	5.5	2.89
50-59	3	8.67	6.33
Total	333		

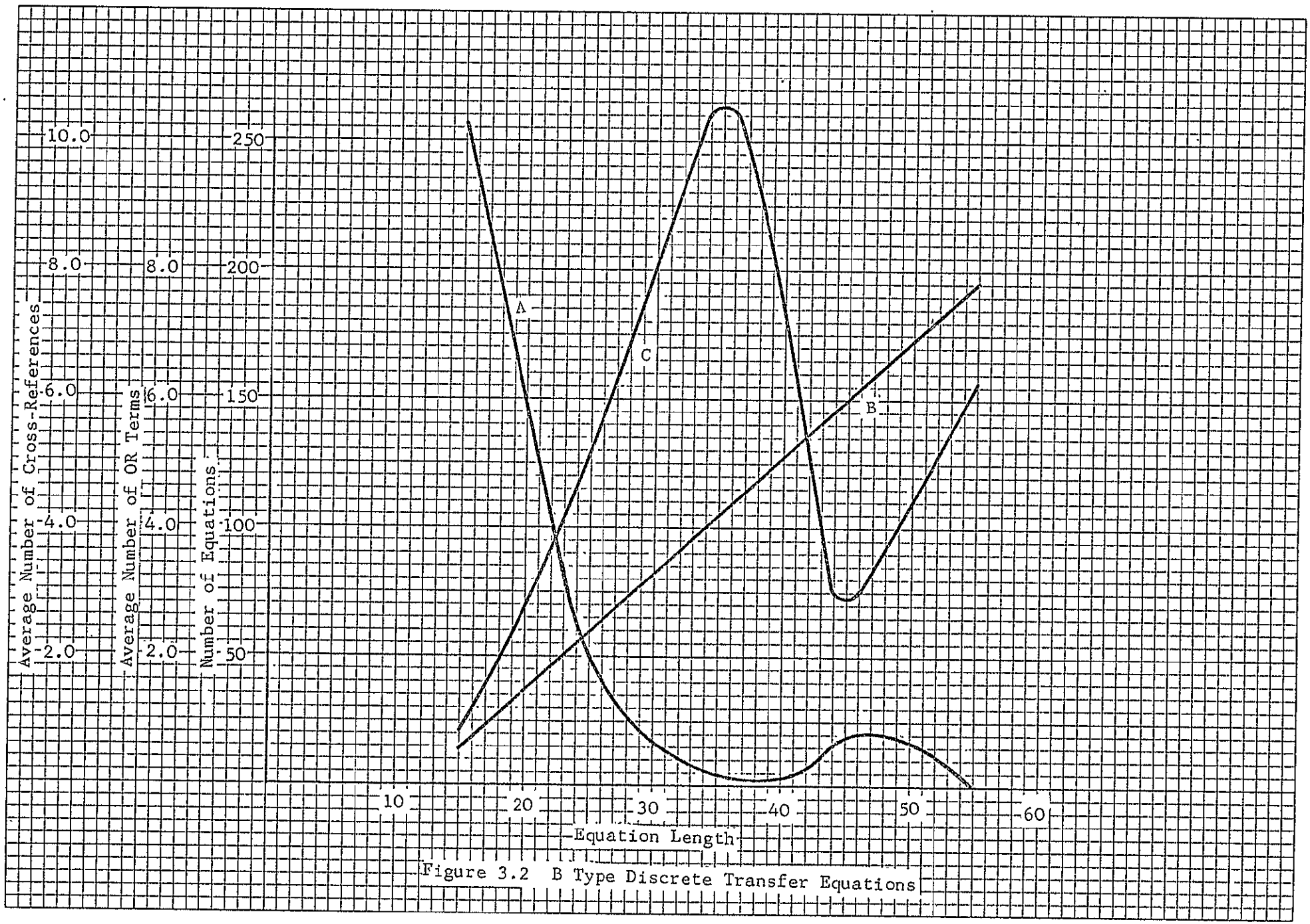


Figure 3.2 B Type Discrete Transfer Equations

3.2.6.4 Evaluation time of transfer equation if all of the status variables are true:

$$t_{aeqvalue(max)} = 1.75 \times 10^{-6} (110\ N_{ST} + 30B_{CR} + 43r + 100a + t_{\$} + 213) = 1.01 \text{ milliseconds}$$

Where N_{ST} = 8 Number of status variables checked
B_{CR} = 1 Number of related equations
r = 0 Number of + operators
a = 1 Equation change in status
$t_{\$}$ = 38 Computer cycles required to process a $ operator

3.2.6.5 Minimum evaluation time of transfer equation if the first status variable is false:

$$t_{aeqvalue(min)} = 1.75 \times 10^{-6} (110\ N_{ST} + 30B_{CR} + 43r + 100a + t_{\$} + 213) = 0.684 \text{ millisecond}$$

Where N_{ST} = 1 Number of status variables checked
B_{CR} = 1 Number of related equations
r = 0 Number of + operators
$t_{\$}$ = 38 Computer cycles required to process a $ operator
a = 0 Equation did not change in status

3.2.6.6 t_{Tape} = 3.2 milliseconds

3.2.6.7 t_{access} = 17.5 milliseconds

3.2.6.8 Maximum time required to search, retrieve, access, evaluate, and store on history tape a B type transfer equation

$$t_{B(max)\ (Equation\ in\ rad)} = t_{main\ driver} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue} + t_{access} + t_{tape}$$

$$t_{B(max)\ (Equation\ in\ rad)} = 25.01 \text{ milliseconds}$$

3.2.6.9 Minimum time required to search, retrieve, access, evaluate, and store on history tape a B type transfer equation:

$$t_{B(min)\ (Equation\ in\ rad)} = t_{main\ driver} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue(min)} + t_{access} + t_{tape}$$

$$t_{B(min)} = 23.685 \text{ milliseconds}$$

3.2.7 Evaluation of B type transfer equations with equation in resident memory

3.2.7.1 Expected time spent in checking the equation buffer to see if equation is in resident memory.

$$E(t)_{Buffer} = \frac{30}{n}(1 + 2 + \ldots + n) = 165 \text{ cycles}$$

Where n = 10 The equation buffer size

3.2.7.2 Time required to search the queues and buffer for equation to evaluate:

$$t_{eqincore\ (Equation\ in\ memory)} = 1.75 \times 10^{-6}\ (45m + E(t)_{buffer} + 150) = 0.787 \text{ millisecond}$$

Where m = 3 Normal Queue

3.2.7.3 $t_{\substack{\text{main} \\ \text{driver}}}$ = 0.107 millisecond

3.2.7.4 $t_{\substack{\text{aeqvalue(max)} \\ \left(\substack{\text{equation} \\ \text{in memory}}\right)}}$ = 2.01 milliseconds (See Section 3.2.6.4)

3.2.7.5 $t_{\substack{\text{aeqvalue(min)} \\ \left(\substack{\text{equation} \\ \text{in memory}}\right)}}$ = 0.684 millisecond (See Section 3.2.6.4)

3.2.7.6 t_{tape} = 3.2 milliseconds

3.2.7.7 Maximum time required to search, evaluate, and output on history tape a B type transfer equation when the equation is in resident memory:

$$t_{\substack{\text{B(max)} \\ \left(\substack{\text{Equation} \\ \text{in memory}}\right)}} = t_{\substack{\text{main} \\ \text{driver}}} + t_{\substack{\text{eqincore} \\ \left(\substack{\text{Equation} \\ \text{in memory}}\right)}} + t_{\substack{\text{aeqvalue(max)} \\ \left(\substack{\text{Equation} \\ \text{in memory}}\right)}} + t_{\text{tape}}$$

$$t_{\substack{\text{B(max)} \\ \left(\substack{\text{Equation} \\ \text{in memory}}\right)}} = 6.094 \text{ milliseconds}$$

3.2.7.8 Minimum time required to search, evaluate, and output on history tape a B type transfer equation when the equation is in resident memory.

$$t_{\substack{\text{B(min)} \\ \left(\substack{\text{Equation} \\ \text{in memory}}\right)}} = t_{\substack{\text{main} \\ \text{driver}}} + t_{\substack{\text{eqincore} \\ \left(\substack{\text{Equation} \\ \text{in memory}}\right)}} + t_{\substack{\text{aeqvalue(min)} \\ \left(\substack{\text{Equation} \\ \text{in memory}}\right)}} + t_{\text{tape}}$$

$$t_{\substack{\text{B(min)} \\ \left(\substack{\text{Equation} \\ \text{in memory}}\right)}} = 4.778 \text{ milliseconds}$$

3.3 Discrete Internal Variables Transfer Equations.

3.3.1 Equation mean length: $\bar{X}_{DIV} = \frac{1}{524} \sum_{i=1}^{12} X_{DIVi} = 28.3$ words

(See Table 3.3 for X_{DIVi}, Y_{DIVi}, and F_{DIVi} values)

3.3.2 Memory requirements; Number of words $= \sum_{i=1}^{12} (X_{DIVi} - Y_{DIVi}) F_{DIVi}$

3.3.3 Number of OR terms (DIV_{OR}) associated with $\bar{X}_{DIV}$:

$DIV_{OR} = 3$

(See curve B figure 3.3)

3.3.4 Number of cross-references (DIV_{CR}) associated with $\bar{X}_{DIV} = 3$
(See curve C figure 3.3)

3.3.5 Number of status variables N_{ST}/OR term:

$$N_{ST} = \frac{\bar{X}_{DIV} - DIV_{CR} - O_{DIV} - 8}{DIV_{OR}} = 5$$

Where $O_{DIV} = 3$ Number of operators

3.3.6 Time required to process a discrete internal variable transfer equation when equation is not in memory.

3.3.6.1 $t_{\substack{main \\ driver}} = 0.107$ millisecond

3.3.6.2 $t_{eqincore} = 1.75 \times 10^{-6} (45m + 30n + 10k + 253) = 3.39$ milliseconds

Table 3-3 Discrete Internal Variable Transfer Equation Grouped Data

Equation Length (Words) X_{DIV}	Frequency Of Occurance F_{DIV}	Average No. of OR Terms	Average No. of Cross References Y_{DIV}
10-19	196	1.58	2.07
20-29	167	2.77	5.42
30-39	92	3.03	6.75
40-49	15	3.70	10.91
50-59	25	4.99	15.10
60-69	6	6.50	25.20
70-79	2	7.50	31.50
80-89	13	9.08	33.50
90-90	4	10.25	20.00
100-109	2	9.5	2.72
120-129	1	20	3.72
150-159	1	16	7.82
Total	524		

Figure 3.3 Discrete Internal Variable Equations

Where m = 3 Normal Queue

n = 50 Buffer Size

k = 5 Normal Queue

3.3.6.3 $t_{bdcinrpt} = 1.75 \times 10^{-6} (18X_{DIV} + 10k + 211) = 1.34$ milliseconds

Where $X_{DIV} = \bar{X}_{DIV} = 28$ words

3.3.6.4 Maximum evaluation time of transfer equation (assuming the fifth status variable of the first two OR terms are false and all other status variables are true):

$$t_{aeqvalue(max)} = 1.75 \times 10^{-6} (110\ N_{ST} + 30\ DIV_{CR} + 43r + 100a + t_{\$} + 213) = 1.26 \text{ milliseconds}$$

$N_{ST} = 3$ Number of status variables checked

$DIV_{CR} = 2$ Number of related equations

$r = 0$ Number of + operators

$a = 0$ Equation did not change in status

3.3.6.6 $t_{tape} = 3.2$ milliseconds

3.3.6.7 $t_{access} = 17.5$ milliseconds

3.3.6.8 Maximum time required to search, retrive, access, evaluate, and output on history tape a discrete internal variable equation.

$$t_{DIV(max)}\ (\text{Equation in rad}) = t_{main\ driver} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue(max)} + t_{access} + t_{tape} = 28.957 \text{ milliseconds}$$

3.3.6.8 Minimum time required to search, retrieve, access, evaluate, and store on history tape a discrete internal variable equation:

$$t_{\substack{DIV(min) \\ (Equation \\ in\ rad)}} = t_{\substack{main \\ driver}} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue(max)} + t_{access} + t_{tape} = 26.797 \text{ milliseconds}$$

3.3.7 Evaluation of discrete an internal variable transfer equations with equation in memory

3.3.7.1 Expected time spend in checking the equation buffer to see if equation is in resident memory:

$$E(t)_{Buffer} = \frac{30}{n}(1 + 2 + \ldots + n) = 765 \text{ cycles}$$

Where n = 50 The equation buffer size

3.3.7.2 Time required to search the queues and equation buffer for equations to evaluate:

$$t_{\substack{eqincore \\ (Equation \\ in\ memory)}} = 1.75 \times 10^{-6}\,(45m + E(t)_{Buffer} + 150) = 1.837 \text{ milliseconds.}$$

Where m = 3 Normal Queue

3.3.7.3 $t_{\substack{main \\ driver}}$ = 0.107 millisecond

3.3.7.4 $t_{\substack{aeqvalue(max) \\ (Equation \\ in\ memory)}}$ = 3.42 milliseconds (see section 3.3.6.4)

3.3.7.5 $t_{\substack{\text{aeqvalue(min)} \\ \text{(Equation in memory)}}}$ = 1.26 milliseconds (see section 3.3.6.5)

3.3.7.6 t_{tape} = 3.2 milliseconds

3.3.7.7 Maximum time required to search, evaluate, and output on history tape a discrete internal variable equation when the equation is in resident memory:

$$t_{\substack{\text{DIV(max)} \\ \text{(Equation in memory)}}} = t_{\substack{\text{main} \\ \text{driver}}} + t_{\substack{\text{equicore} \\ \text{(Equation in memory)}}} + t_{\substack{\text{aeqvalue(max)} \\ \text{(Equation in memory)}}} + t_{\text{tape}}$$

$$t_{\substack{\text{DIV(max)} \\ \text{(Equation in memory)}}} = 8.554 \text{ milliseconds}$$

3.3.7.8 Minimum time required to search, evaluate, and output on history tape a discrete internal variable transfer equation when the equation is in resident memory:

$$t_{\substack{\text{DIV(min)} \\ \text{(Equation in memory)}}} = t_{\substack{\text{main} \\ \text{driver}}} + t_{\substack{\text{eqincore} \\ \text{(Equation in memory)}}} + t_{\substack{\text{aeqvalue(min)} \\ \text{(Equation in memory)}}} + t_{\text{tape}}$$

$$t_{\substack{\text{DIV(min)} \\ \text{(Equation in memory)}}} = 6.394 \text{ milliseconds}$$

3.4 DDAS Discrete Transfer Equations

3.4.1 Equation mean length (excluding last two entries of Table 3.4):

$$\bar{X}_{DDAS} = \frac{1}{177} \sum_{i=1}^{10} X_{DDASi} F_{DDASi} = 16.14 \text{ words}$$

(See Table 3.4 for X_{DDASi} and F_{DDASi} values)

3.4.2 Memory requirements: Number of words $= \sum_{i=1}^{12} (X_{DDASi} - Y_{DDASi}) F_{DDASi}$

$= 4498$ words

3.4.3 Number of OR terms ($DDAS_{OR}$) associated with $\bar{X}_{DDAS}$:

$DDAS_{OR} = 1$ (See curve B figure 3.4)

3.4.4 Number of cross-references ($DDAS_{CR}$) associated with $\bar{X}_{DDAS}$:

$DDAS_{CR} = 3.0$ (Rounded off to nears whole number)

3.4.5 Number of status variables (N_{ST})/OR term

$$= \frac{\bar{X}_{DDAS} - DDAS_{CR} - O_{DDAS} - 8}{DDAS_{OR}} = 4$$

Where $O_{DDAS} = 2$ Number of operators

3.4.6 Evaluation of DDAS discrete transfer equations with equation not in memory:

Table 3.4 DDAS Discrete Transfer Equations Grouped Data

Equation Length X_{DDAS}	Frequency F_{DDAS}	Average No. of OR Terms	Average No. of Cross References Y_{DDAS}
10-14	130	1.07	0.5
15-19	19	1.63	2.47
20-24	9	2.78	2.11
25-29	6	2.83	4.5
30-34	8	6.0	1.0
35-39	1	4.0	10.0
40-44	1	4.0	13.0
55-59	1	6.0	23.0
85-89	1	16.0	1.0
135-140	1	27.0	1.0
690-694	1	89.0	1.0
700-704	1	91.0	1.0
Total	179		

Figure 3.4 DDAS Discrete Equations

3.4.6.1 $t_{\text{main driver}}$ = 0.107 millisecond

3.4.6.2 $t_{\text{eqincore}} = 1.75 \times 10^{-6} (45m + 30n + 10k + 253) = 1.29$ milliseconds

Where m = 3 Normal Queue
n = 10 Buffer Size
k = 5 Normal Queue

3.4.6.3 $t_{\text{bdcinrpt}} = 1.75 \times 10^{-6} (18X_{\text{DDAS}} + 10k + 211) = 0.96$ millisecond

$X_{\text{DDAS}} = \overline{X}_{\text{DDAS}} = 16$ words

3.4.6.4 Maximum evaluation time or transfer equation (assuming all four status variables are true):

$$t_{\text{aeqvalue(max)}} = 1.75 \times 10^{-6} (110\ N_{ST} + 30 DDAS_{CR} + 43r + 100a + t_{\$} + 213) = 1.54 \text{ milliseconds}$$

Where N_{ST} = Number of status variables checked
$DDAS_{CR}$ = 3 Related equations
r = 0 Number of operators
$t_{\$}$ = 38 Computer cycles to process
a = 1 Equation changed in status

3.4.6.5 Minimum evaluation time of transfer equation (assuming the first status variable is false):

$$t_{\text{aeqvalue(min)}} = 1.75 \times 10^{-6} (110\ N_{ST} + 30\ DDAS_{CR} + 43r + 100a + t_{\$} + 213)$$

0.72 millisecond

Where N_{ST} = 1 Number of status variables checked

DDAS = 3 Number of related equations

r = 0 Number of + operators

a = 0 Equation did not change in status

3.4.6.6 t_{tape} = 3.2 milliseconds

3.4.6.7 t_{access} = 17.5 milliseconds

3.4.6.8 Maximum time required to search, retrieve, access, evaluate, and store on history tape a DDAS transfer equation:

$$t_{\substack{DDAS(max) \\ \left(\text{Equation in rad}\right)}} = t_{\substack{main \\ driver}} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue} + t_{tape} + t_{access}$$

$$t_{\substack{DDAS(max) \\ \left(\text{Equation in rad}\right)}} = 24.597 \text{ milliseconds}$$

3.4.6.9 Minimum time required to search, retrieve, access, evaluate, and store on history tape a DDAS transfer equation:

$$t_{\substack{DDAS(min) \\ \left(\text{Equation in rad}\right)}} = t_{\substack{main \\ driver}} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue} + t_{tape} + t_{access}$$

$$t_{\substack{DDAS(min) \\ \left(\text{Equation in rad}\right)}} = 23.777 \text{ milliseconds}$$

3.4.7 Evaluation of a DDAS discrete transfer equations with equation in memory.

3.4.7.1 Expected time spent in checking the equation buffer to see if equation is in resident memory:

$$E(t)_{Buffer} = \frac{30}{n}(1 + 2 + \ldots n) = 165 \text{ cycles}$$

Where n = 10 The equation buffer size

3.4.7.2 Time required to search the queues and buffer for equations to evaluate:

$$t_{\substack{eqincore \\ (\text{Equation in memory})}} = 1.75 \times 10^{-6} (45m + e(t)_{Buffer} + 150)$$

$$t_{\substack{eqincore \\ (\text{Equation in memory})}} = 0.787 \text{ millisecond}$$

Where m = 3 Normal Queue

3.4.7.3 $t_{\substack{main \\ driver}} = 0.107$ millisecond

3.4.7.4 $t_{\substack{aeqvalue(max) \\ (\text{Equation in memory})}} = 1.54$ milliseconds (See Section 3.4.6.4)

3.4.7.5 $t_{\substack{aeqvalue(min) \\ (\text{Equation in memory})}} = 0.72$ milliseconds (See Section 3.4.6.5)

3.4.7.6 Maximum time required to search, evaluate, and output on history tape a DDAS discrete transfer equation when the equation is in resident memory:

$$t_{\substack{\text{DDAS(max)} \\ \text{Equation} \\ \text{in memory}}} = t_{\substack{\text{main} \\ \text{driver}}} + t_{\substack{\text{eqincore} \\ \text{Equation} \\ \text{in memory}}} + t_{\substack{\text{aeqvalue(max)} \\ \text{Equation} \\ \text{in memory}}} + t_{\text{tape}}$$

$$t_{\substack{\text{DDAS(max)} \\ \text{Equation} \\ \text{in memory}}} = 5.624 \text{ milliseconds}$$

3.4.7.7 Minimum time required to search, evaluate, and output on history tape a DDAS discrete type transfer equation when the equation is in resident memory:

$$t_{\substack{\text{DDAS(min)} \\ \text{Equation} \\ \text{in memory}}} = t_{\substack{\text{main} \\ \text{driver}}} + t_{\substack{\text{eqincore} \\ \text{Equation} \\ \text{in memory}}} + t_{\substack{\text{aeqvalue(min)} \\ \text{Equation} \\ \text{in memory}}} + t_{\text{tape}}$$

$$t_{\substack{\text{DDAS(min)} \\ \text{Equation} \\ \text{in memory}}} = 4.814 \text{ milliseconds}$$

3.5 Power Bus Discrete Transfer Equations

3.5.1 Equation mean length: $\bar{X}_{PB} = \frac{1}{33} \sum_{i=1}^{5} X_{PBi} F_{PBi} = 36.71$ words

(See Table 3.5 for X_{PBi}, Y_{PBi} and F_{PBi} values)

3.5.2 Memory requirements: Number of words $= \sum_{i=1}^{8} (X_{PBi} - Y_{PBi}) F_{PBi}$

$= 1841$ words

3.5.3 Number of OR terms (PB_{OR}) associated with $\bar{X}_{PB}$: $PB_{OR} = 2$
(See curve B figure 3.5)

3.5.4 Number of cross-references (PB_{CR}) associated with $\bar{X}_{PB}$: $PB_{CR} = 18$
(See curve C figure 3.5)

3.5.5 Number of status variables (N_{ST})/OR term $= \frac{\bar{X}_{PB} - PB_{CR} - O_{PB} - 8}{PB_{OR}} = 5$

3.5.6 Evaluation of power bus discrete equations with equation not in memory.

3.5.6.1 $t_{\substack{main \\ driver}} = 0.107$ millisecond

3.5.6.2 $t_{eqincore} = 1.75 \times 10^{-6} (45m + 30n + 10k + 253) = 2.87$ milliseconds

Where m = 3 Normal Queue
n = 40 Buffer Size
k = 5 Normal Queue

3.5.6.3 $t_{bdcinrpt} = 1.75 \times 10^{-6} (18X_{PB} + 10k + 211) = 1.62$ milliseconds

Where $X_{PB} = \bar{X}_{PB} = 37$ words

Table 3.5 Power Bus Discrete Transfer Equations Grouped Data

Equation Length (Words) X_{PB}	Frequency Of Occurance F_{PB}	Average No. of OR Terms	Average No. of Cross References Y_{PB}
10-19	14	1.07	2.6
20-29	7	2.90	4.0
30-39	1	2.00	21.0
40-49	6	7.50	4.3
50-59	4	4.60	26.4
60-69	1	9.00	0
140-149	1	1.00	135
190-199	1	2.00	175
Total	35		

Figure 3.5 Power Bus Equation

3.5.6.4 Maximum evaluation time of transfer equation assuming first four status variables of the first OR term are true while the fifth is false and all five status variables of second OR term are true

$$t_{aeqvalue(max)} = 1.75 \times 10^{-6} (110\ N_{ST} + 30\ PB_{CR} + 43r + 100a + t_{\$} + 213) = 3.35 \text{ milliseconds}$$

Where N_{ST} = 9 Number of status variables checked

PB_{CR} = 18 Number of related equations

r = 1 Number of + operators

$t_{\$}$ = 38 Cycles required to process $ operator

a = 1 Equation changed in status

3.5.6.5 Minimum evaluation time of transfer equation (assuming the first status variable of each OR term is false):

$$t_{aeqvalue(min)} = 1.75 \times 10^{-6} (110\ N_{ST} + PB_{CR} + 100a + 43r + t_{\$} + 213) = 1.84 \text{ milliseconds}$$

Where N_{ST} = 2 Number of status variables checked

PB_{CR} = 18 Number of related equations

r = 1 Number of + operators

a = 0 Discrete equation did change in status

3.5.6.6 t_{tape} = 3.6 milliseconds

3.5.6.7 t_{access} = 17.5 milliseconds

3.5.6.8 Maximum time required to search, retrieve, access, evaluate, and store on history tape a power bus transfer equation:

$$t_{PB(max)\ (Equation\ in\ rad)} = t_{main\ driver} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue(max)} + t_{tape} + t_{access}$$

$$t_{PB(max} = 28.687 \text{ milliseconds}$$

3.5.6.9 Minimum time required to search, retrive, access evaluate, and store on history tape a power bus transfer equation:

$$t_{PB(min)\ (Equation\ in\ rad)} = t_{main\ driver} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue(min)} + t_{tape} + t_{access}$$

$$t_{PB(min)\ (Equation\ in\ rad)} = 27.138 \text{ milliseconds}$$

3.5.7 Evaluation of a power bus discrete transfer equation with equation in resident memory

3.5.7.1 Expected time spent in checking the equation buffer to see if equation is in the resident memory:

$$E(t)_{Buffer} = \frac{30}{n}(1 + 2 + \ldots n) = 615 \text{ cycles}$$

Where n = 40 The equation buffer size

3.5.7.2 Time required to search the queues and buffer for equations to evaluate:

$$t_{\left(\begin{array}{c}\text{eqincore}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = 1.75 \times 10^{-6} \ (45m + E(t)_{\text{Buffer}} + 150)$$

$$t_{\left(\begin{array}{c}\text{eqincore}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = 1.592 \text{ milliseconds}$$

Where m = 3 Normal Queue

3.5.7.3 $t_{\substack{\text{main}\\ \text{driver}}} = 0.107$ millisecond

3.5.7.4 $t_{\left(\begin{array}{c}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = 3.35$ milliseconds (See Section 3.5.6.4)

3.5.7.5 $t_{\left(\begin{array}{c}\text{aeqvalue(min)}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = 1.84$ milliseconds (See Section 3.5.6.5)

3.5.7.6 Maximum time required to search, evaluate, and output on history tape a power bus discrete transfer equation when the equation is in resident memory:

$$t_{\left(\begin{array}{c}\text{PB(max)}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = t_{\substack{\text{main}\\ \text{driver}}} + t_{\left(\begin{array}{c}\text{eqincore}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} + t_{\left(\begin{array}{c}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} + t_{\text{tape}}$$

$$t_{\left(\begin{array}{c}\text{PB(max)}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = 9.249 \text{ milliseconds}$$

3.5.7.7 Minimum time required to search, evaluate, and output on history tape a power bus discrete type transfer equation when the equation is in resident memory:

$$t_{\substack{PB(min) \\ \text{Equation} \\ \text{in memory}}} = t_{\substack{main \\ driver}} + t_{\substack{eqincore \\ \text{Equation} \\ \text{in memory}}} + t_{\substack{aeqvalue(min) \\ \text{Equation} \\ \text{in memory}}} + t_{tape}$$

$$t_{\substack{PB(min) \\ \text{Equation} \\ \text{in memory}}} = 6.737 \text{ milliseconds}$$

3.6 VXXX = EXXX// P.V., DV//$ C.R. Type Transfer Equations

3.6.1 Equation mean length: $\bar{X}_V = \frac{1}{254} \sum_{i=1}^{5} X_{Vi} F_{Vi} = 17.44$ words

(See Table 3.6 for X_{Vi}, Y_{Vi} and F_{Vi} values)

3.6.2 Memory requirements: Number of words $= \sum_{i=1}^{5} (X_{Vi} - Y_{Vi}) F_{Vi} = 4808$ words

3.6.3 Number of OR terms (V_{OR}) associated with $\bar{X}_V$: $V_{OR} = 1$

(See Table 3.6 and curve B of figure 3.6)

3.6.4 Number of cross-references (V_{CR}) associated with $\bar{X}_V$:

$V_{CR} = 1$ (See curve C figure 3.6)

3.6.5 Number of status values (N_{ST}) = 1

3.6.6 Evaluation of VXXX//P.V., D.V.//$ type transfer equation with equations not in memory:

3.6.6.1 $t_{\text{main driver}} = 0.107$ millisecond

3.6.6.2 $t_{\text{eqincore}} = 1.75 \times 10^{-6} (45m + 30n + 10k + 253) = 1.16$ milliseconds

Where m = 2 Timed Queue
n = 10 Buffer Size
k = 2 Timed Queue

3.6.6.3 $t_{\text{bdcinrpt}} = 1.75 \times 10^{-6} (18X_V + 10k + 211) = 0.710$ millisecond

$X_V = \bar{X}_V = 17$ words

Table 3.6 VXXX = EXXX//P.V., D.V.//$ Type
Transfer Equation Grouped Data

Equation Length (Words) X_V	Frequency Of Occurance F_V	Average No. of OR Terms	Average No. of Cross References
17	188	1	1
18	31	1	2
19	17	1	3
20	16	1	4
21	2	1	5
Total	254		

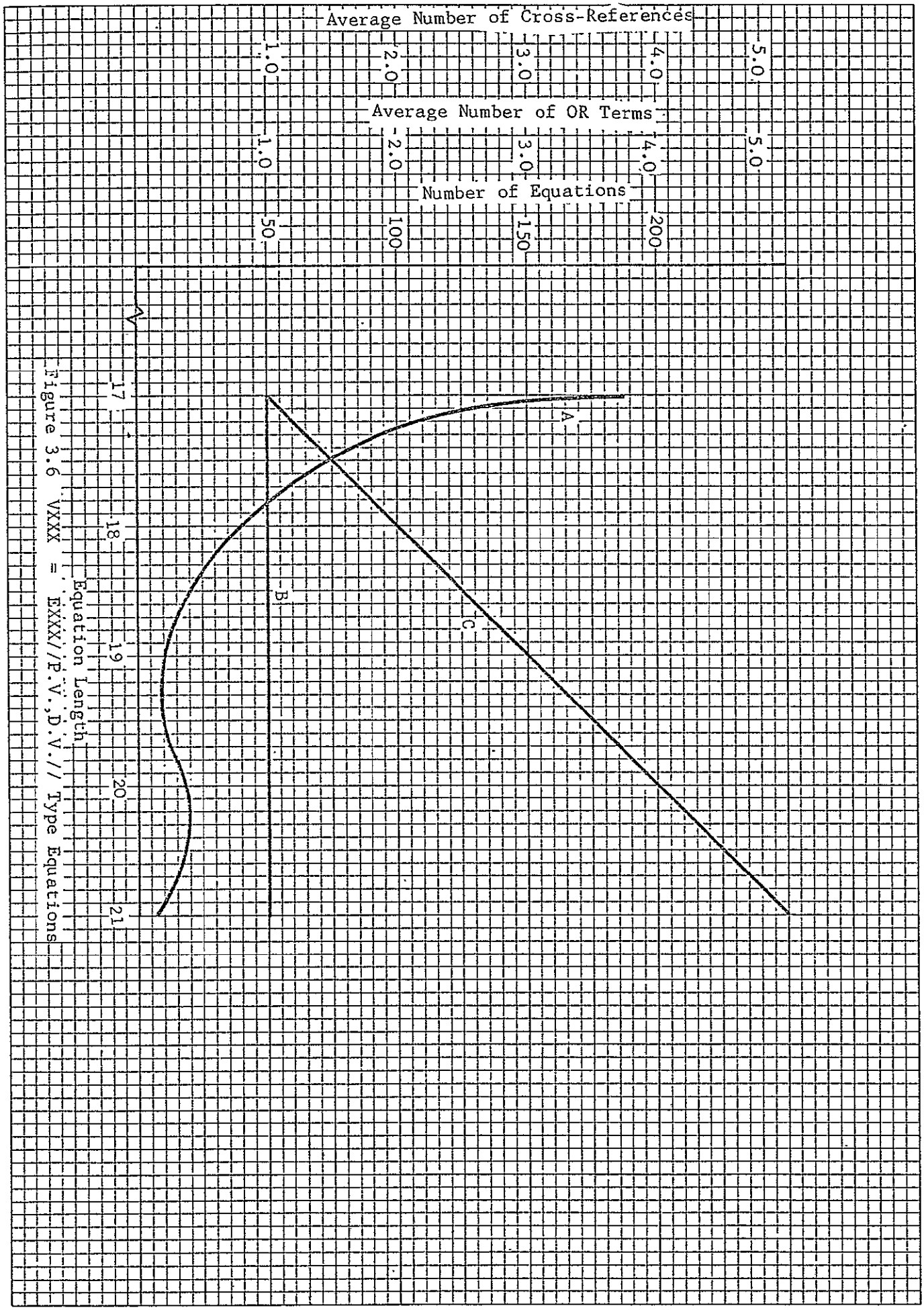


Figure 3.6 VXXX = EXXX//P.V.,D.V.// Type Equations

3.6.6.4 $t_{Vaeqvalue} = 1.75 \times 10^{-6} (110\ N_{ST} + 30\ V_{CR} + 100a + 43r + t_{\$} + 289)$

$t_{Vaeqvalue} = 0.992$ millisecond

Where $N_{ST} = 1$ Number of status variables checked

$V_{CR} = 1$ Number of related equations

$a = 1$ Equations changed in status

$t_{\$} = 38$ Computer cycles required to process $ operator

$r = 0$ Number of + operators

3.6.6.5 Time required to evaluate related equations:

$$t_{V\ CR} = t_{B(max)}\left(\begin{matrix}\text{Equation}\\ \text{in rad}\end{matrix}\right) P_B + t_{E(max)}\left(\begin{matrix}\text{Equation}\\ \text{in rad}\end{matrix}\right) P_E + t_{F(max)}\left(\begin{matrix}\text{Equation}\\ \text{in rad}\end{matrix}\right) P_F + t_{DIV(max}\left(\begin{matrix}\text{Equation}\\ \text{in rad}\end{matrix}\right)$$

$$P_{DIV} + t_{DDAS(max)}\left(\begin{matrix}\text{Equation}\\ \text{in rad}\end{matrix}\right) P_{DDAS} + t_{BUS}\left(\begin{matrix}\text{Equation}\\ \text{in rad}\end{matrix}\right) P_{BUS}$$

$t_{V\ CR}\left(\begin{matrix}\text{Equation}\\ \text{in rad}\end{matrix}\right) = 28.324$ milliseconds

$P_{BUS} = \frac{2}{379} = 0.005$ Probability that the related equation is a bus type.

Where $t_{B(max)}\left(\begin{matrix}\text{Equation}\\ \text{in rad}\end{matrix}\right) = 25.01$ milliseconds (See Section 3.2.6.8

$P_B = 0.174$ (See Table 3.6.1)

$t_{E(max)}$ (Equation in rad) = 31.337 milliseconds (See Section 3.8.6.7)

P_E = 0.235 (See Table 3.6.1)

$t_{F(max)}$ (Equation in rad) = 29.416 milliseconds (See Section 3.7.6.6)

P_F = 0.290 (See Table 3.6.1)

$t_{DIV(max)}$ (Equation in rad) = 28.957 milliseconds (See Section 3.3.6.8)

P_{DIV} = 0.161 (See Table 3.6.1)

$t_{DDAS(max)}$ (Equation in rad) = 24.597 milliseconds (See Section 3.4.6.8)

P_{DDAS} = 0.132 (See Table 3.6.1)

$t_{BUS(max)}$ (Equation in rad) = 28.687 milliseconds (See Section 3.5.6.8)

P_{BUS} = 0.005 (See Table 3.6.1)

3.6.6.6 t_{tape} = 3.2 milliseconds

3.6.6.7 t_{access} = 17.5 milliseconds

3.6.6.8 Time required to search, retrieve, access, evaluate, and store on history tape a VXXX = EXXX//P.V., DV//$ tape equation

$$t_V = t_{main\ driver} + t_{eqincore} + t_{bdcinrpt} + t_{aeqvalue} + t_{access} + t_{tape}$$

t_V = 23.667 milliseconds

Table 3.6.1 VXXX = EXXX//P.V., D.V.//$ Type Of Related Equations

Equation Length X	Frequency	Number and Type of Related Equations					
		B	E	F	DIV	DDAS	Bus
1	191	65	10	47	25	40	2
2	31	1	42	13	4	2	0
3	18	0	29	8	8	8	0
4	16	0	8	32	24	0	0
5	2	0	0	10	0	0	0
Totals	258	66	89	110	61	50	2

3.6.6.9 Time required to search, retrieve, access, evaluate and store on history tpae a VXXX = EXXX//P.V., D.V.//$ and its related equations:

$$t_{V\ total} = t_V + t_{\substack{V\ RE \\ \left(\begin{matrix}\text{Equation} \\ \text{in rad}\end{matrix}\right)}} = 51.991 \text{ milliseconds}$$

3.6.7 Evaluation of VXXX = EXXX//P.V., D.V.//$ transfer equation with equation in resident memory:

3.6.7.1 Expected time spend in checking the equation buffer to see if equation is in resident memory:

$$E(t)_{Buffer} = \frac{30}{n} (1 + 2 + \ldots n) \text{ 165 cycles}$$

Where n = 10 The equation buffer size

3.6.7.2 Time required to search the queues and buffer for equations to evaluate:

$$t_{\substack{eqincore \\ \left(\begin{matrix}\text{Equation} \\ \text{in memory}\end{matrix}\right)}} = 1.7 \times 10^{-6} (45m + E(t)_{Buffer} + 150)$$

$$t_{\substack{eqincore \\ \left(\begin{matrix}\text{Equation} \\ \text{in memory}\end{matrix}\right)}} = 0.709 \text{ millisecond}$$

Where m = 2 Timed Queue

3.6.7.3 $t_{\substack{main \\ driver}} = 0.107$ millisecond

3.6.7.4 t_{aeqvalue} (Equation in memory) = 0.992 millisecond, (See Section 3.6.6.4)

3.6.7.5 Time required to search, evaluate, and output on history tape a VXXX = EXXX//P.V.,D.V//$ transfer equation when the equation is in resident memory:

$$t_{V\ (\text{Equation in memory})} = t_{\text{main driver}} + t_{\text{eqincore}\ (\text{Equation in memory})} + t_{\text{aeqvalue(max)}\ (\text{Equation in memory})} + t_{\text{tape}}$$

$$t_{V\ (\text{Equation in memory})} = 5.008 \text{ milliseconds}$$

3.6.7.6 Time required to evaluate the expected related equations:

$$t_{V\ RE\ (\text{Equation in memory})} = t_{B(\max)\ (\text{Equation in memory})} P_B + t_{E(\max)\ (\text{Equation in memory})} P_E + t_{F(\max)\ (\text{Equation in memory})} P_F + t_{DIV(\max)\ (\text{Equation in memory})} P_{DIV} + t_{DDAS(\max)\ (\text{Equation in memory})} P_{DDAS} + t_{BUS(\max)\ (\text{Equation in memory})} P_{BUS}$$

$$t_{V\ RE\ (\text{Equation in memory})} = 7.955 \text{ milliseconds}$$

Where $t_{B(\max)\ (\text{Equation in memory})}$ = 6.094 milliseconds, (See Section 3.2.7.7)

P_B = 0.174 (See Table 3.6.1)

$T_{E(max)}$ (Equation in memory) = 11.406 milliseconds

P_E = 0.235 (See Table 3.6.1)

$t_{F(max)}$ (Equation in memory) = 7.082 milliseconds

P_F = 0.290 (See Table 3.6.1)

t_{DIV} (Equation in memory) = 8.554 milliseconds

P_{DIV} = 0.161 (See Table 3.6.1)

$t_{DDAS(max)}$ (Equation in memory) = 5.624 milliseconds

P_{DDAS} = 0.132 (See Table 3.6.1)

$t_{BUS(max)}$ (Equation in memory) = 8.247 milliseconds

P_{BUS} = 0.005 (See Table 3.6.1)

3.6.7.7 Time required to search, evaluate, and record on history tape a VXXX = EXXX//P.V.,D.V.//$ type transfer equation and its related equation when the equations are in resident memory:

$t_{V\ total}$ (Equation in memory) = $t_V + t_{V\ RE}$ = 5.008 + 7.955 = 12.963 milliseconds

3.7 F Type Analog Equation

3.7.1 Equation mean length (excluding the last three entries of Table 3.7):

$$\bar{X}_F = \frac{1}{219} \sum_{i=1}^{12} X_{Fi} F_{Fi} = 31.2 \text{ or } 31 \text{ words}$$

3.7.2 Memory requirements: Number of words $= \sum_{i=1}^{15} (X_{Fi} - Y_{Fi}) f_{Fi} = 8206$ words

3.7.3 Number of OR terms associated with $\bar{X}_F$
(See curve B figure 3.7): $F_{OR} = 2$ OR terms

3.7.4 Number of cross-references (F_{CR}) associated with $\bar{X}_F$
(See curve C figure 3.7): $F_{CR} = 2$ cross-references

3.7.5 Number of status variables (N_{ST})/OR term

$$N_{ST} = \frac{\bar{X}_F - F_{CR} - O_F - K - 8}{F_{OR}} = 2 \text{ (rounding off)}$$

Where $O_F = 6$ Number of operators

$K = 6$ Average number of entries inclosed within the /..../ marks per OR term

3.7.6 Evaluation of F type analog equations with equation not in memory:

3.7.6.1 $t_{\substack{\text{main} \\ \text{driver}}} = 0.456$ millisecond (See Sperry Rand's "General Purpose Simulator System Study" Part 2).

3.7.6.2 $t_{\substack{\text{eqincore} \\ \left(\substack{\text{Equation} \\ \text{in rad}}\right)}} = 1.75 \times 10^{-6} (45m + 30n + 10k + 253)$

$t_{\substack{\text{eqincore} \\ \left(\substack{\text{Equation} \\ \text{in rad}}\right)}} = 2.64$ milliseconds

Table 3.7 F Type Analog Equations

Grouped Data

Equation Length X_F	Frequency f_F	Average No. of OR Terms	Average No. of Cross References Y_F
15-24	97	1	0.35
25-34	69	1.64	2.03
35-44	29	1.69	1.83
45-54	14	2.57	2.28
55-64	2	3.5	5
65-74	1	4	6
75-84	1	4	0
85-94	2	8	0
95-104	0	0	0
115-124	2	11	0
125-134	1	9	0
135-144	1	9	0
215-224	1	14	0
245-254	1	13	13
625-634	1	16	16

222

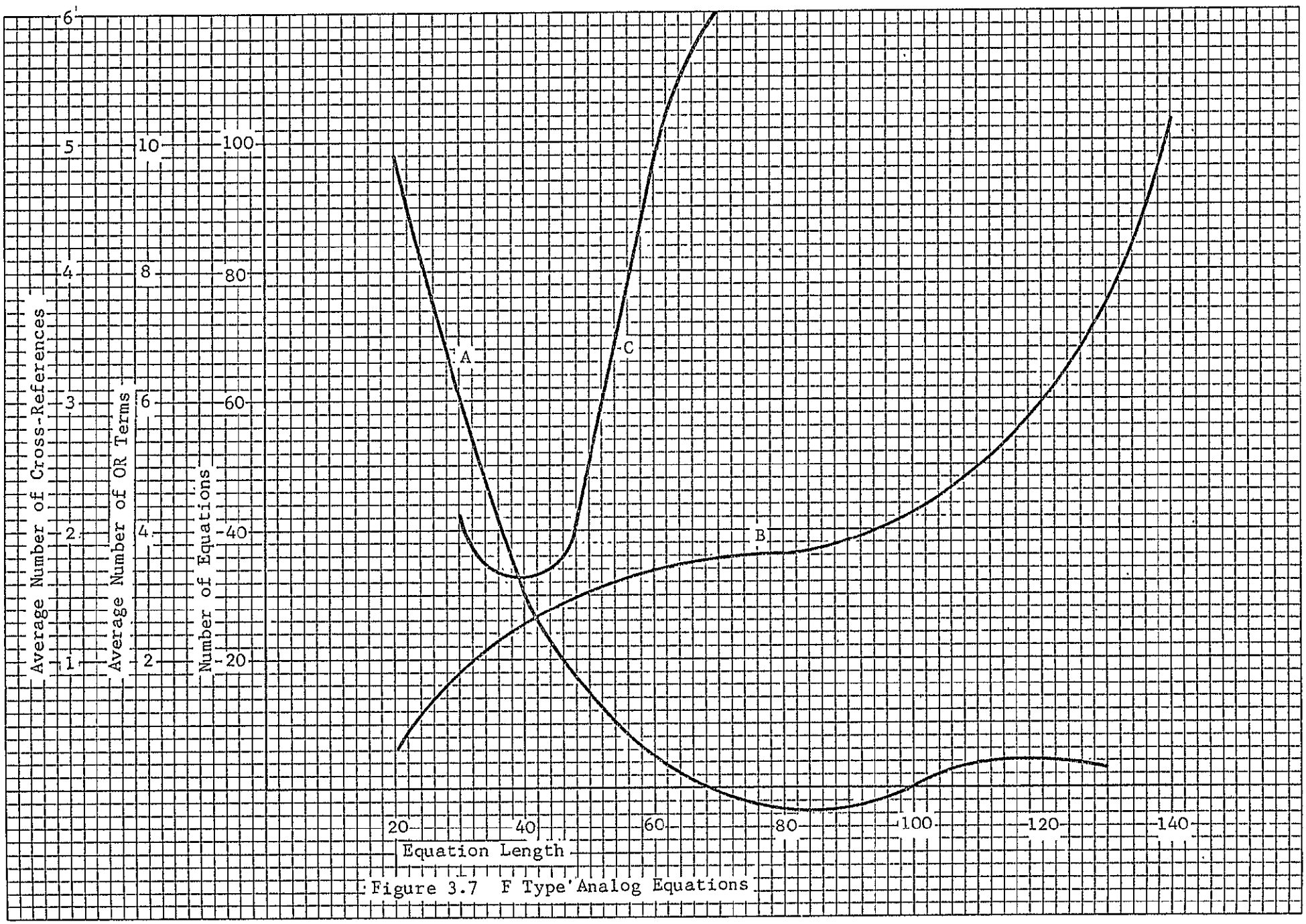


Figure 3.7 F Type Analog Equations

m = 1 Special Queue

n = 40 Buffer Size

k = 1 Special Queue

3.7.6.2 $t_{\left(\begin{matrix}\text{bdcinrpt}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 1.75 \times 10^{-6} (18X_F + 10k + 211)$

$t_{\left(\begin{matrix}\text{bdcinrpt}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 1.36$ milliseconds

Where $X_F = \bar{X}_F = 31$ Words

3.7.6.3 Maximum evaluation time of transfer equation (assuming the second status variable of the first OR term to be false and all other status variable to be true):

$$t_{\left(\begin{matrix}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 1.75 \times 10^{-6} \; 110 N_{ST} + 30 F_{CR} + 100a + 43r + 11 S + (185 P_1 + 370 P_2 + 555 P_3 + 740 P_4) P_{CR} + t_{\$} + 402 + t_{Calog}$$

$t_{\left(\begin{matrix}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 4.26$ milliseconds

Where $N_{ST} = 3$ Maximum number of status variables checked

$F_{CR} = 2$ Number of related equations

$a = 1$ Number of OR terms that changed in status

$r = 1$ Number of $\pm$ operators

$S = 4$ Relative position of desired entry in secondary timer table

P_1 = 0.327 (See Table 3.7.1)

P_2 = 0.265 (See Table 3.7.1)

P_3 = 0.265 (See Table 3.7.1)

P_4 = 0.082 (See Table 3.7.1)

P_{CR} = 0.221 (See Table 3.7.1)

$t_\$$ = 38 Computer cycles required to process a $ operator

$t_{E\ Calog}$ = 2.34 milliseconds (See Section 3.10)

3.7.6.4 Minimum evaluation time of transfer equation (assuming the first status variable of each OR term to be false, i.e., neither OR term changed in status).

$$t_{\left(\begin{matrix}\text{aeqvalue(min)}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 1.75 \times 10^{-6} (110\ N_{ST} + 43r + t_\$ + 156)$$

$$t_{\left(\begin{matrix}\text{aeqvalue(min)}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 0.8 \text{ millisecond}$$

Where N_{ST} = 2 Number of status variables checked

r = 1 Number of + operators

$t_\$$ = 38 Computer cycles required to process $ operator

3.7.6.5 Expected time for the solution of VXXX = EXXX//P.V., D.V.//$ C.R type related equations:

$$E(t)_{F_{CR}} = t_V\ P_{CR}\ (P_{V1} + 2P_{V2} + 3P_{V3} + 5P_{V5} + 6P_{V6} + 7P_{V7} + 8P_{V8})$$

$$E(t)_{F_{CR}} = 0.56\ t_V = 13.253 \text{ milliseconds}$$

Where t_V = 23.667 milliseconds — Time required to search retrieve, access, and evaluate VXXX = EXXX//P.V., D.V.//$ type equations.

P_{V1} = 0.265 (See Table 3.7.1)

P_{V2} = 0.327 (See Table 3.7.1)

P_{V3} = 0.163 (See Table 3.7.1)

P_{V5} = 0.061 (See Table 3.7.1)

P_{V6} = 0.061 (See Table 3.7.1)

P_{V7} = 0.020 (See Table 3.7.1)

P_{V8} = 0.040 (See Table 3.7.1)

P_{CR} = 0.221 (See Table 3.7.1)

3.7.6.6 Maximum time required to initially search, retrieve, access, evaluate, and output on history tape an F type analog transfer equation:

$$t_{\left(\begin{smallmatrix}\text{F(max)}\\ \text{Equation}\\ \text{in rad}\end{smallmatrix}\right)} = t_{\substack{\text{main}\\ \text{driver}}} + t_{\left(\begin{smallmatrix}\text{eqincore}\\ \text{Equation}\\ \text{in rad}\end{smallmatrix}\right)} + t_{\left(\begin{smallmatrix}\text{bdcinrpt}\\ \text{Equation}\\ \text{in rad}\end{smallmatrix}\right)} + t_{\left(\begin{smallmatrix}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in rad}\end{smallmatrix}\right)} + t_{\text{access}} + t_{\text{tape}}$$

$$t_{\left(\begin{smallmatrix}\text{F(max)}\\ \text{Equation}\\ \text{in rad}\end{smallmatrix}\right)} = 29.416 \text{ milliseconds}$$

3.7.6.7 Minimum time required to initially search, retrieve, access, evaluate, and record on history tape an F type analog transfer equation:

Table 3.7.1 F Type Related Equations

Number of Equations	Pick-Up Value	Drop-Out Value	V// //	DIV
13	1	1	1	0
3	1	1	2	0
3	0	0	0	1
13	2	2	2	0
3	3	3	5	0
2	3	3	6	1
8	3	3	3	0
1	4	4	7	2
2	4	4	8	0
1	4	4	6	0

Total 49

$P_{CR} = \frac{49}{222} = 0.221$ Probability that the transfer equation has related equations, pick-up, and drop-out values:

$P_1 = \frac{16}{49} = 0.327$ Probability of only one pick-up and one drop-out value.

$P_2 = \frac{13}{49} = 0.265$ Probability of two pick-up and two drop-out values.

$P_3 = \frac{13}{49} = 0.265$ Probability of three pick-up and three drop-out values.

$P_4 = \frac{4}{49} = 0.082$ Probability of four pick-up and four drop-out values.

$P_{V1} = \frac{13}{49} = 0.265$ Probability of only one related VXXX = EXXX //P.V., D.V.//$ type equations.

$P_{V2} = \frac{16}{49} = 0.327$ Probability of two related VXXX = EXXX //P.V., D.V.//$ type equations.

$P_{V3} = \frac{8}{49} = 0.163$ Probability of three related VXXX = EXXX //P.V., D.V.//$ type equations.

$P_{V5} = \frac{3}{49} = 0.061$ Probability of five related VXXX = EXXX //P.V., D.V.//$ type equations.

$P_{V6} = \frac{3}{49} = 0.061$ Probability of six related VXXX = EXXX //P.V., D.V.//$ type equations.

$P_{V7} = \frac{1}{49} = 0.020$ Probability of seven related VXXX = EXXX //P.V., D.V.//$ type equations.

$P_{V8} = \frac{2}{49} = 0.040$ Probability of eight related VXXX = EXXX //P.V., D.V.//$ type equations

$$t_{F(min)\ (\text{Equation in rad})} = t_{\text{main driver}} + t_{\text{eqincore}\ (\text{Equation in rad})} + t_{\text{bdcinrpt}\ (\text{Equation in rad})} + t_{\text{aeqvalue}\ (\text{Equation in rad})} + t_{\text{access}} + t_{\text{tape}}$$

$$t_{F(min)\ (\text{Equation in rad})} = 25.956 \text{ milliseconds}$$

3.7.6.8 Maximum time required to initially search, retrieve, access, evaluate, and output on history tape an F type equation and all of its related equations:

$$t_{F(max)\ \text{total}\ (\text{Equation in rad})} = t_{\text{main driver}} + t_{\text{eqincore}\ (\text{Equation in rad})} + t_{\text{bdcinrpt}\ (\text{Equation in rad})} + t_{\text{aeqvalue(max)}\ (\text{Equation in rad})} + E(t)_{F_{CR}\ (\text{Equation in rad})} + t_{\text{tape}} + t_{\text{access}}$$

$$t_{F(max)\ \text{total}\ (\text{Equation in rad})} = 42.669 \text{ milliseconds}$$

3.7.7.9 Minimum time required to initially search, retrieve, access, evaluate, and output on history tape an F type analog equation and all of its related equations:

$$t_{F(min)\ \text{total}\ (\text{Equation in rad})} = t_{\text{main driver}} + t_{\text{eqincore}\ (\text{Equation in rad})} + t_{\text{bdcinrpt}\ (\text{Equation in rad})} + t_{\text{aeqvalue(min)}\ (\text{Equation in rad})} + E(t)_{F_{CR}\ (\text{Equation in rad})} + t_{\text{access}} + t_{\text{tape}}$$

$$t_{\substack{F(min)\ total \\ (Equation\ in\ rad)}} = 39.209 \text{ milliseconds}$$

3.7.7 Evaluation of F type analog equations with equation in memory.

3.7.7.1 Expected time to check the equation buffer to see if equation is in memory:

$$E(t)_{Buffer} = \frac{t}{n}(1 + 2 + \ldots + n)$$

Where $t = 30$ Computer cycles required for each buffer check

$n = 40$ Buffer size

$E(t)_{Buffer} = 615$ cycles

3.7.7.2 Time required to search queues and buffer for equation to evaluate:

$$t_{\substack{eqincore \\ (Equation\ in\ memory)}} = 1.75 \times 10^{-6}\,(35m + E(t)_{Buffer} + 150)$$

$$t_{\substack{eqincore \\ (Equation\ in\ memory)}} = 1.4 \text{ milliseconds}$$

Where $m = 1$ Special Queue

3.7.7.3 $t_{\substack{main \\ driver}} = 0.456$ milliseconds

3.7.7.4 Maximum time required for the evaluation of the equation (assuming second status variable of the first OR term to be false and all other status variables to be true):

$$t_{\substack{\text{aeqvalue} \\ \left(\text{Equation in rad}\right)}} = 4.26 \text{ milliseconds} \quad \text{(See Section 3.7.6.3)}$$

3.7.7.5 Minimum time required for the evaluation of the equation (assuming all of the status variables of the first OR term to be true):

$$t_{\substack{\text{aeqvalue(min)} \\ \left(\text{Equation in memory}\right)}} = 1.75 \times 10^{-6}\ 110\ N_{ST} + 30F_{CR} + 100a + 43r + 11S + (185P_1 + 370P_2 + 555P_3 + 740P_4)\ P_{CR} + t_{\$} + 402 + t_{CR\ Calog} = 4.00 \text{ milliseconds}$$

Where $N_{ST} = 2$ Number of status variables checked

$r = 0$ Number of + operators

$t_{CR\ Calog} = 2.54$ milliseconds (See Section 3.10)

All other variables are the same as those of Section 3.7.6.3.

3.7.7.6 Expected time for the solution of VXXX = EXXX//P.V., D.V.//$ type related equations assuming the related equations are in resident memory.

$$E(t)_{\substack{F_{CR} \\ \left(\text{Equation in memory}\right)}} = t_{\substack{V \\ \left(\text{Equation in memory}\right)}} \cdot (P_{V1} + 2P_{V2} + 3P_{V3} + 5P_{V5} + 6P_{V6} + 7P_{V7} + 8P_{V8})$$

$$P_{CR} = 0.56\ t_{\substack{V \\ \left(\text{Equation in memory}\right)}} = 2.804 \text{ milliseconds}$$

Where $t_{V\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)}$ = 5.008 milliseconds — Time required to search and evaluate VXXX = EXXX//P.V., D.V.//$ type equations

P_{CR}, P_{V1} . . . P_{V8} (See Table 3.7.1)

3.7.7.7 Maximum time required to search, evaluate, and record on history tape an F type analog transfer equation when the equation is in resident memory:

$$t_{F(max)\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)} = t_{\text{main driver}} + t_{\text{eqincore}\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)} + t_{\text{aeqvalue(max)}\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)} + t_{\text{tape}}$$

$$t_{F(max)\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)} = 7.082 \text{ milliseconds}$$

3.7.7.8 Minimum time required to search, evaluate, and record on history tape an F type analog transfer equation when the equation is in resident memory:

$$t_{F(min)\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)} = t_{\text{main driver}} + t_{\text{eqincore}\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)} + t_{\text{aeqvalue(min)}\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)} + t_{\text{tape}}$$

$$t_{F(min)\left(\begin{smallmatrix}\text{Equation}\\ \text{in memory}\end{smallmatrix}\right)} = 7.056 \text{ milliseconds}$$

3.7.7.9 Maximum time required to search, evaluate, and output on history tape an F type analog equation and all of its related equations when the equations are in resident memory:

$$t_{F(max)\ total\ (Equation\ in\ memory)} = t_{main\ driver} + t_{eqincore\ (Equation\ in\ memory)} + t_{aeqvalue\ (Equation\ in\ memory)} + E(t)_{E_{CR}\ (Equation\ in\ memory)} + t_{tape}$$

$$t_{F(max)\ total\ (Equation\ in\ memory)} = 9.886 \text{ milliseconds}$$

3.7.7.10 Minimum time required to search, evaluate, and output on history tape an F type analog equation and all related equations when the equations are in resident memory:

$$t_{F(min)\ total\ (Equation\ in\ memory)} = t_{main\ driver} + t_{eqincore\ (Equation\ in\ memory)} + t_{aeqvalue\ (Equation\ in\ memory)} + E(t)_{E_{CR}\ (Equation\ in\ memory)} + t_{tape}$$

$$t_{F(min)\ total\ (Equation\ in\ memory)} = 9.860 \text{ milliseconds}$$

3.8 E Type Analog Transfer Equations

3.8.1 Equation mean length (excluding last three entries of table 3.8).

$$\bar{X}_E = \frac{1}{229} \sum_{i=1}^{11} X_{Ei} F_{Ei} = 40.00 \text{ Words}$$

3.8.2 Memory requirements: Number of words $= \sum_{i=1}^{14} (X_{Ei} + Y_{Ei}) F_{Ei} = 11638$ words

3.8.3 Number of OR terms associated with $\bar{X}_E$: $E_{OR} = 2.1$ or 2 OR terms

(See curve B, figure 3.8)

3.8.4 Number of cross references (E_{CR}) associated with $\bar{X}_E$:

(See curve C, figure 3.8)

$E_{CR} = 0.8$ or 1 related equation

3.8.5 Number of status variables (N_{ST})/OR term

$$N_{ST} = \frac{\bar{X}_E - E_{CR} - O_E - K - 8}{E_{OR}} = 7 \quad \text{Status variables per OR term}$$

Where $O_E = 6$ Number of operators

$K = 6$ Average number of entries inclosed within the / . . . ./ marks per OR term.

3.8.6 Evaluation of E type analog equations with equations not in memory.

3.8.6.1 $t_{\text{main driver}} = 0.456$ millisecond (See Sperry Rand's "General Purpose Simulator System Study" Part 2)

Table 3.8 E Type of Analog Equation

Grouped Data

Equation Length X_E	Frequency F_E	Average No. of OR Terms	Average No. of Cross References Y_E
15-24	102	1	0.09
25-34	43	2	0.56
35-44	15	1.8	0.2
45-54	20	3.0	0.8
55-64	9	5.0	1.66
65-74	11	4.45	0.91
75-84	7	5.72	0.58
85-194	3	6	2
95-104	2	4	0
105-114	14	5.06	0.286
115-124	3	9	2.67
165-174	4	8	2.25
175-184	5	8	3.2
245-254	2	17	3
	240		

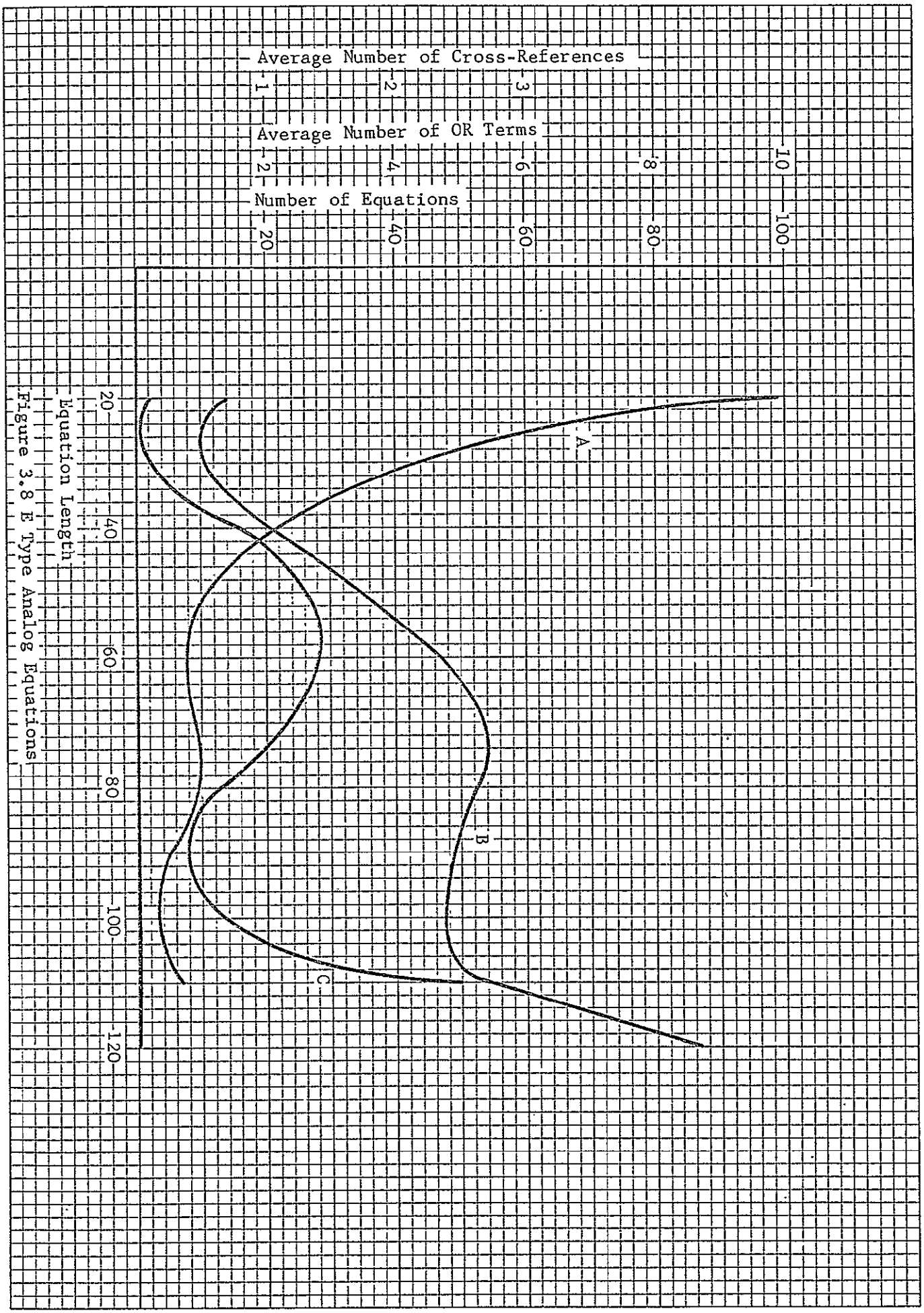


Figure 3.8 E Type Analog Equations

3.8.6.2 $t_{\substack{\text{eqincore} \\ \text{Equation} \\ \text{in rad}}} = 1.75 \times 10^{-6} (45m + 30n + 10k + 253)$

$t_{\substack{\text{eqincore} \\ \text{Equation} \\ \text{in rad}}} = 2.64$ milliseconds

Where m = 1 Special Queue
n = 40 Buffer Size
k = 1 Special Queue

3.8.6.3 $t_{\substack{\text{bdcinrpt} \\ \text{Equation} \\ \text{in rad}}} = 1.75 \times 10^{-6} (18X_E + 10k + 211)$

$t_{\substack{\text{bdcinrpt} \\ \text{Equation} \\ \text{in rad}}} = 1.65$ milliseconds

Where $X_E = \bar{X}_E = 40$ words

3.8.6.4 Maximum initial evaluation time of transfer equation (assuming the seventh status variable of the first OR term to be false and all other status variables of the transfer equation to be true):

$$t_{\substack{\text{aeqvalue} \\ \text{Equation} \\ \text{in rad}}} = 1.75 \times 10^{-6}(110\ N_{ST} + 30\ E_{CR} + 100a + 43r + 11S + (185P_1 + 370P_2 + 555P_3 + 740P_4)P_{CR} + t_{\$} + 402) + t_{Calog}$$

$$= 3.75 + 2.6 = 6.35 \text{ milliseconds}$$

Where N_{ST} = 13 Maximum number of status variables checked

E_{CR} = 1 Average number of related equations

a = 1 Number of OR terms that changed in status

r = 1 Number of + operators

S = 4 Relative position of desired entry in secondary timer table.

$t_{\$}$ = 38 Computer cycles to process $ operator

P_1 = 0.107 (See Table 3.8.1)

P_2 = 0.75 (See Table 3.8.1)

P_3 = 0.018 (See Table 3.8.1)

P_4 = 0.125 (See Table 3.8.1)

P_{CR} = 0.233 (See Table 3.8.1)

t_{Calog} = 2.6 milliseconds (See Section 3.10.1)

3.8.6.5 Minimum initial evaluation time of transfer equation (assuming the first status variable of each OR term to be false, i.e., neither OR term changed in status):

$$\left(\begin{matrix} t_{aeqvalue(min)} \\ \text{Equation} \\ \text{in rad} \end{matrix}\right) = 1.75 \times 10^{-6} (110\ N_{ST} + 43r + t_{\$} + 156)$$

$$= 0.8 \text{ millisecond}$$

Where N_{ST} = 2 Number of status variables checked

r = 1 Number of plus operators

$t_{\$}$ = 38 Computer cycles required to process $ operator

3.8.6.6 Expected time for the solution of VXXX = EXXX//P.V., D.V.//$ C.R. type related equations:

$$E(t)_{E_{CR}} = t_V(P_{V1} + 2P_{V2} + 3P_{V3} + 4P_{V4} + 6P_{V6})\ P_{CR}$$

$$E(t)_{E_{CR}} = 12.614 \text{ milliseconds}$$

Where $t_V = 23.667$ Time required to search, retrieve, and evaluate VXXX = EXXX//P.V., D.V.//$C.R. type equations.

$P_{V1} = 0.089$ (See Table 3.8.1)

$P_{V2} = 0.75$ (See Table 3.8.1)

$P_{V3} = 0.018$ (See Table 3.8.1)

$P_{V4} = 0.107$ (See Table 3.8.1)

$P_{V6} = 0.036$ (See Table 3.8.1)

$P_{CR} = 0.233$ (See Table 3.8.1)

3.8.6.7 Maximum time required to initially search, retrieve, access, evaluate, and output on history tape an E type analog transfer equation:

$$t_{\left(\begin{matrix}E(max)\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = t_{\substack{\text{main}\\ \text{driver}}} + t_{\left(\begin{matrix}\text{eqincore}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{bdcinrpt}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)}$$

$$+ t_{\text{access}} + t_{\text{tape}}$$

$$t_{\left(\begin{matrix}E(max)\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 31.796 \text{ milliseconds}$$

3.8.6.8 Minimum time required to initially search, retrieve, evaluate, and output on history tape an E type analog transfer equation:

$$t_{\left(\begin{matrix}E(min)\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = t_{\text{main driver}} + t_{\left(\begin{matrix}\text{eqincore}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{bdcinrpt}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{aeqvalue(min)}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)}$$

$$+ t_{\text{access}} + t_{\text{tape}}$$

$$t_{\left(\begin{matrix}E(min)\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 26.246 \text{ milliseconds}$$

3.8.6.9 $t_{\text{access}} = 17.5$ milliseconds

3.8.6.10 $t_{\text{tape}} = 3.2$ milliseconds

3.8.6.11 Maximum time required to initially search, retrieve, access, evaluate, and output on history tape an E type equation and all related equations.

$$t_{\left(\begin{matrix}E(max)\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = t_{\text{main driver}} + t_{\left(\begin{matrix}\text{eqincore}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{bdcinrpt}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)}$$

$$+ E(t)_{\left(\begin{matrix}E_{CR}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} + t_{\text{tape}} + t_{\text{access}}$$

$$t_{\left(\begin{matrix}E(max)\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 44.41 \text{ milliseconds}$$

3.8.6.12 Minimum time required to initially search, retrieve, access, evaluate, and output on history tape an E type analog equation and all related equations:

$$t_{E(min)\ (\text{Equation in rad})} = t_{\text{main driver}} + t_{\text{eqincore}\ (\text{Equation in rad})} + t_{\text{bdcinrpt}\ (\text{Equation in rad})} + t_{\text{aeqvalue(min)}\ (\text{Equation in rad})}$$

$$+ t_{\text{tape}} + t_{\text{access}} + E(t)_{E_{CR}\ (\text{Equation in rad})} = 38.860 \text{ milliseconds}$$

3.8.7 Evaluation of E type analog equations with equations in memory:

3.8.7.1 Expected time to check the equation buffer to see if the equation is already in memory:

$$E(t)_{\text{Buffer}} = \frac{t}{n}(1 + 2 + \ldots + n)$$

$$E(t)_{\text{Buffer}} = 615 \text{ Cycles}$$

Where $t = 30$ Computer cycles required for each buffer check

$n = 40$ Buffer size

3.8.7.2 Time required to search queues and buffer for equation to evaluate:

$$t_{\text{eqincore}\ (\text{Equation in memory})} = 1.75 \times 10^{-6}\ (45m + E(t)_{\text{Buffer}} + 150) = 1.4 \text{ milliseconds}$$

Where $m = 1$ Special Queue

3.8.7.3 $t_{\text{main driver}} = 0.107$ millisecond

3.8.7.4 Maximum time required for the evaluation of the equation (assuming seventh status variable of first OR term to be false and all other status variables to be true):

$$t_{\left(\begin{array}{c}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = 6.35 \text{ milliseconds (See Section 3.8.6.4)}$$

3.8.7.5 Minimum time required for the evaluation of the equation (assuming all of the status variables of the first OR term to be true):

$$t_{\left(\begin{array}{c}\text{aeqvalue(min)}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = 1.75 \times 10^{-6} (110\, N_{ST} + 30\, E_{CR} + 100a + 43r + 11S + (185P_1 + 370P_2 + 555P_3 + 740P_4)\, P_{CR} + t_{\$} + 402) + t_{Calog}$$

$$t_{\left(\begin{array}{c}\text{aeqvalue(min)}\\ \text{Equation}\\ \text{in memory}\end{array}\right)} = 3.12 \text{ milliseconds}$$

Where $N_{ST} = 7$ Number of status variables checked

$E_{CR} = 1$ Average number of related equations

$a = 1$ Number of OR terms that changed in status

$r = 0$ Number of plus operators

$S = 4$ Relative position of desired entry in secondary times table.

$P_1 = 0.104$ (See Table 3.8.1)

$P_2 = 0.75$ (See Table 3.8.1)

$P_3 = 0.018$ (See Table 3.8.1)

P_4 = 0.125 (See Table 3.8.1)

P_{CR} = 0.233 (See Table 3.8.1)

$t_{\$}$ = 38 Computer cycles to process a $ operator

t_{Calog} = 2.80 milliseconds (See Section 3.10.1).

3.8.7.6 Maximum time required to search, evaluate, and output on history tape an E type analog transfer equation when the equation is in resident memory:

$$t_{\left(\begin{matrix}\text{E(max)}\\ \text{Equation}\\ \text{in memory}\end{matrix}\right)} = t_{\begin{matrix}\text{main}\\ \text{driver}\end{matrix}} + t_{\left(\begin{matrix}\text{eqincore}\\ \text{Equation}\\ \text{in memory}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{aeqvalue(min)}\\ \text{Equation}\\ \text{in memory}\end{matrix}\right)} + t_{\text{tape}}$$

$$t_{\left(\begin{matrix}\text{E(max)}\\ \text{Equation}\\ \text{in memory}\end{matrix}\right)} = 11.406 \text{ milliseconds}$$

3.8.7.7 Minimum time required to search, evaluate, and output on history tape an E type analog transfer equation when the equation is in resident memory:

$$t_{\left(\begin{matrix}\text{E(min)}\\ \text{Equation}\\ \text{in memory}\end{matrix}\right)} = t_{\begin{matrix}\text{main}\\ \text{driver}\end{matrix}} + t_{\left(\begin{matrix}\text{eqincore}\\ \text{Equation}\\ \text{in memory}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{aeqvalue(min)}\\ \text{Equation}\\ \text{in memory}\end{matrix}\right)} + t_{\text{tape}}$$

$$t_{\left(\begin{matrix}\text{E(min)}\\ \text{Equation}\\ \text{in memory}\end{matrix}\right)} = 8.176 \text{ milliseconds}$$

3.8.7.8 Expected time for the solution of VXXX = EXXX//P.V., D.V.//$ type cross reference equations assuming cross reference equation is in memory:

Table 3.8.1 E Type Related Equations

No. of Equations	PV	DV	V
5	1	1	1
1	1	1	2
1	2	2	3
39	2	2	2
1	3	3	4
2	2	2	2
2	4	4	6
5	4	4	4

56

$P_1 = \frac{6}{56} = 0.104$ Probability of only one pick-up and one drop-out value.

$P_2 = \frac{42}{56} = 0.75$ Probability of two pick-up and two drop-out values.

$P_3 = \frac{1}{56} = 0.018$ Probability of three pick-up and three drop-out values.

$P_4 = \frac{7}{56} = 0.125$ Probability of four pick-up and four drop-out values.

$P_{CR} = \frac{56}{240} = 0.233$ Probability that the transfer equation has related equations, pick-up and drop-out values.

$P_{V1} = \frac{5}{50} = 0.089$ Probability of only and related VXXX = EXXX//P.B., D.V// Type equation.

$P_{V2} = \frac{42}{56} = 0.75$ Probability of two related VXXX = EXXX//P.V., D.V.//$ Type equations.

$P_{V3} = \frac{1}{56} = 0.018$ Probability of three related VXXX = EXXX// P.V., D.V.//$ Type equations.

$P_{V4} = \frac{6}{50} = 0.107$ Probability of four related VXXX = EXXX// P.V., D.V.//$ Type equations.

$P_{V6} = \frac{2}{50} = 0.036$ Probability of six related VXXX = EXXX// P.V., D.V.//$ Type equations

$$E(t)_{\substack{E_{CR} \\ (\text{Equation in memory})}} = t_{V\,(\text{Equation in memory})} P_{V1} + 2t_{V\,(\text{Equation in memory})} + 3t_{V\,(\text{Equation in memory})} P_{V3}$$

$$4t_{V\,(\text{Equation in memory})} P_{V4} + 6t_{V\,(\text{Equation in memory})} P_{V6}) P_{CR}$$

$$E(t)_{\substack{E_{CR} \\ (\text{Equation in memory})}} = 2.669 \text{ milliseconds}$$

Where $t_{V\,\text{Equation in memory}}$ = 5.008 milliseconds — Time required to search and evaluate VXXX = EXXX//P.V., D.V.//$C.R. type equations

P_{V1} = 0.089 (See Table 3.8.1)

P_{V2} = 0.75 (See Table 3.8.1)

P_{V3} = 0.018 (See Table 3.8.1)

P_{V4} = 0.107 (See Table 3.8.1)

P_{V6} = 0.036 (See Table 3.8.1)

P_{CR} = 0.233 (See Table 3.8.1)

3.8.7.9 Maximum time required to search, evaluate, and output on history tape an E type analog equation and all related equations when the equations are in resident memory:

$$t_{E(max)\ total\ (\text{Equation in memory})} = t_{\substack{\text{main} \\ \text{driver}}} + t_{eqincore\ (\text{Equation in memory})} + t_{aeqvalue(max)\ (\text{Equation in memory})} + E(t)_{\substack{E_{CR} \\ (\text{Equation in memory})}} + t_{tape}$$

$$t_{E(max)\ total\ \left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right)} = 10.845 \text{ milliseconds}$$

3.9 Analog Internal Variable Transfer Equations

3.9.1 Equation mean length (excluding last two entries of Table 3.9).

$$\bar{X}_{AIV} = \frac{1}{28} \sum_{i=1}^{4} X_{AIVi} F_{AIVi} = 18.7 \text{ or } 19 \text{ words.}$$

3.9.2 Memory requirements: Number of words $= \sum_{i=1}^{6} X_{AIVi} F_{AIVi} = 987$ words

3.9.3 Number of OR terms (AIV_{OR}) per transfer equation based on an equation mean length of 19 words (See curve B, figure 3.9): $AIV_{OR} = 1$ term

3.9.4 Number of cross references (AIV_{CR}) per transfer equation based on an equation mean length of 19 words (See curve C, figure 3.9): $AIV_{CR} = 1$.

3.9.5 Number of status variables (N_{ST})/OR term:

$$N_{ST} = \frac{\bar{X}_{AIV} - AIV_{CR} - O_{AIV} - K - 8}{AIV_{OR}} = 5$$

Where $O_{AIV} = 3$ Number of operators

$K = 2$ Number of entries enclosed within / . ./ marks per OR term.

3.9.6 Evaluation of an analog internal variable equations with equations not in memory:

3.9.6.1 $t_{\substack{main \\ driver}} = 0.456$ millisecond

Table 3.9 Analog Internal Variable Transfer Equation Grouped Data

Equation Length X_{AIV}	Frequency F_{AIV}	Average No. of OR Terms	Average No. of Cross References Y_{AIV}
16-17	17	1	1.12
18-19	4	1	1.2
22-23	1	1	0
24-25	6	1.67	5
223	1	18	18
240	1	2	20
Total	30		

Figure 3.9 Analog Internal Variables

3.9.6.2 $t_{\left(\begin{matrix}\text{eqincore}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 1.75 \times 10^{-6} (45m + 30n + 10k + 253)$

$t_{\left(\begin{matrix}\text{eqincore}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 1.064$ milliseconds

Where m = 1 Special Queue
n = 10 Buffer Size
k = 1 Special Queue

3.9.6.3 $t_{\left(\begin{matrix}\text{bdcinrpt}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 1.75 \times 10^{-6} (18X_{AIV} + 10k + 211)$

$t_{\left(\begin{matrix}\text{bdcinrpt}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 0.985$ millisecond

Where $X_{AIV} = \overline{X}_{AIV} = 19$ words

3.9.6.4 Maximum initial evaluation time of transfer equation (assuming all five status variables are true):

$$t_{\left(\begin{matrix}\text{aeqvalue(max)}\\ \text{Equation}\\ \text{in rad}\end{matrix}\right)} = 1.75 \times 10^{-6}(110\ N_{ST} + 30\ AIV_{CV} + 100a + 43r + 11S + t_{\$} + 402) + t_{Calog}$$

Where $N_{ST} = 5$ Maximum number of status variables checked

$AIV_{CR} = 1$ Number of related equations

$= 1$ Number of OR terms that changed in status

$r = 0$ Number of 1 operators

$S = 4$ Relative position of desired entry in secondary timer table.

$t_{\$} = 38$ Computer cycles to process a $ operator

$t_{Calog} = 0.728$ millisecond

$$t_{\substack{aeqvalue(max) \\ (\text{Equation in rad})}} = 2.765 \text{ milliseconds}$$

3.9.6.5 Minimum initial evaluation time of transfer equation (assuming the first status variable checked is false):

$$t_{\substack{aeqvalue(min) \\ (\text{Equation in rad})}} = 1.75 \times 10^{-6} (110\ N_{ST} + 43r + t_{\$} + 156)$$

Where $N_{ST} = 1$ Number of status variables checks

$r = 0$ Number of + operators

$t_{\$} = 38$ Computer cycles required to process $ operator

$$t_{\substack{aeqvalue(min) \\ (\text{Equation in rad})}} = 0.515 \text{ millisecond.}$$

3.9.6.6 Maximum expected time to evaluate related equation:

$$t_{\substack{AIV_{CR(max)} \\ (\text{Equation in rad})}} = t_{\substack{B(max) \\ (\text{Equation in rad})}} P_B + t_{\substack{DIV(max) \\ (\text{Equation in rad})}} P_{DIV} + t_{\substack{E(max) \\ (\text{Equation in rad})}} P_E + t_{\substack{F(max) \\ (\text{Equation in rad})}} P_F$$

Where $t_{\left(\begin{matrix}B(max)\\ Equation\\ in\ rad\end{matrix}\right)}$ = 25.01 milliseconds (See Section 3.2.6.8)

$P_B = \frac{3}{50} = 0.06$ The probability that the related equation is a B type.

$t_{\left(\begin{matrix}DIV(max)\\ Equation\\ in\ rad\end{matrix}\right)}$ = 28.957 milliseconds (See Section 3.3.6.8)

$P_{DIV} = \frac{42}{50} = 0.84$ The probability that the related equation is a discrete internal variable type.

$t_{\left(\begin{matrix}E(max)\\ Equation\\ in\ rad\end{matrix}\right)}$ = 31.796 milliseconds (See Section 3.8.6.7)

$P_E = \frac{4}{50} = 0.08$ Probability that the related equation is of the E type.

$t_{\left(\begin{matrix}F(max)\\ Equation\\ in\ rad\end{matrix}\right)}$ = 29.416 milliseconds (See Section 3.7.6.6)

$P_F = \frac{1}{50} = 0.02$ Probability that the related equation is of the F type.

$t_{AIV_{\left(\begin{matrix}CR(max)\\ Equation\\ in\ rad\end{matrix}\right)}}$ = 28.949 milliseconds

3.9.6.7 Minimum expected time to evaluate related equation:

$$t_{\left(\begin{matrix}\text{AIV}_{\text{CR}}(\text{min}) \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)} = t_{\left(\begin{matrix}\text{B(min} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)} P_B + t_{\left(\begin{matrix}\text{DIV(min)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)} P_{DIV} + t_{\left(\begin{matrix}\text{E(min)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)} P_E + t_{\left(\begin{matrix}\text{F(min)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)} P_F$$

Where $t_{\left(\begin{matrix}\text{B(min)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)}$ = 23.685 milliseconds (See Section 3.2.6.9)

$t_{\left(\begin{matrix}\text{DIV (min)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)}$ = 26.797 milliseconds (See Section 3.3.6.9)

$t_{\left(\begin{matrix}\text{E(min)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)}$ = 26.246 milliseconds (See Section 3.8.6.8)

$t_{\left(\begin{matrix}\text{F(min)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)}$ = 25.956 milliseconds (See Section 3.7.6.7)

$t_{\left(\begin{matrix}\text{AIV}_{\text{CR}}(\text{min}) \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)}$ = 26.549 milliseconds

3.9.6.8 Maximum time required to initially search, retrieve, access, evaluate, and output on history tape an analog internal variable transfer equation:

$$t_{\left(\begin{matrix}\text{AIV(max)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)} = t_{\begin{matrix}\text{main} \\ \text{driver}\end{matrix}} + t_{\left(\begin{matrix}\text{eqincore} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{bdcinrpt} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)} + t_{\left(\begin{matrix}\text{aeqvalue(max)} \\ \text{Equation} \\ \text{in rad}\end{matrix}\right)}$$

$$+ t_{\text{access}} + t_{\text{tape}}$$

$$t_{\substack{AIV(max) \\ \left(\substack{Equation \\ in\ rad}\right)}} = 25.970 \text{ milliseconds}$$

3.9.6.9 Minimum time required to initially search retrieve, access, evaluate, and output on history tape an analog internal variable transfer equation and its related equation:

$$t_{\substack{AIV(max)\ total \\ \left(\substack{Equation \\ in\ rad}\right)}} = t_{\substack{AIV(max) \\ \left(\substack{Equation \\ in\ rad}\right)}} + t_{\substack{AIV_{CR}(max) \\ \left(\substack{Equation \\ in\ rad}\right)}} = 54.919 \text{ milliseconds}$$

3.9.6.10 Minimum time required to initially search retrieve, access, evaluate, and output on history tape an analog internal variable transfer equation:

$$t_{\substack{AIV(min) \\ \left(\substack{Equation \\ in\ rad}\right)}} = t_{\substack{main \\ driver}} + t_{\substack{eqincore \\ \left(\substack{Equation \\ in\ rad}\right)}} + t_{\substack{bdcinrpt \\ \left(\substack{Equation \\ in\ rad}\right)}} + t_{\substack{aeqvalue(min) \\ \left(\substack{Equation \\ in\ rad}\right)}}$$

$$+ t_{access} + t_{tape}$$

$$t_{\substack{AIV(min) \\ \left(\substack{Equation \\ in\ rad}\right)}} = 23.720 \text{ milliseconds}$$

3.9.6.11 Minimum time required to initially search, retrieve, access, evaluate, and output on history tape an analog internal variable equation and its related equation:

$$t_{\substack{AIV(min) \\ \left(\substack{Equation \\ in\ rad}\right)}} = t_{\substack{AIV(min) \\ \left(\substack{Equation \\ in\ rad}\right)}} + t_{\substack{AIV_{CR}(min) \\ \left(\substack{Equation \\ in\ rad}\right)}} = 50.269 \text{ milliseconds}$$

3.9.7 Evaluation of an analog internal variable transfer equation with equation in resident memory.

3.9.7.1 Expected time spent in checking the equation buffer to see if equation is in resident memory:

$$E(t)_{Buffer} = \frac{30}{n}(1 + 2 + \ldots n) = 165 \text{ cycles}$$

Where $n = 10$ The equation buffer size

3.9.7.2 Time required to search the queues and buffer for equations to evaluate:

$$t_{\left(\begin{matrix}\text{eqincore} \\ \text{Equation} \\ \text{in memory}\end{matrix}\right)} = 1.75 \times 10^{-6} (45m + E(t)_{Buffer} + 150)$$

$$t_{\left(\begin{matrix}\text{eqincore} \\ \text{Equation} \\ \text{in memory}\end{matrix}\right)} = 0.630 \text{ millisecond}$$

Where $m = 1$ Special Queue

3.9.7.3 $t_{\text{main driver}} = 0.456$ millisecond

3.9.7.4 $t_{\left(\begin{matrix}\text{aeqvalue(max)} \\ \text{Equation} \\ \text{in memory}\end{matrix}\right)} = 2.765$ milliseconds (See Section 3.9.6.4)

3.9.7.5 $t_{\left(\begin{matrix}\text{aeqvalue(min)} \\ \text{Equation} \\ \text{in memory}\end{matrix}\right)} = 0.515$ millisecond (See Section 3.9.6.5)

3.9.7.6 Maximum time required to search, evaluate, and output on history tape and analog internal variable transfer equation when the equation is in resident memory:

$$t_{\substack{AIV(max) \\ (\text{Equation in memory})}} = t_{\substack{main \\ driver}} + t_{\substack{eqincore \\ (\text{Equation in memory})}} + t_{\substack{aeqvalue(max) \\ (\text{Equation in memory})}} + t_{tape}$$

$$t_{\substack{AIV(min) \\ (\text{Equation in memory})}} = 4.891 \text{ milliseconds}$$

3.9.7.7 Minimum time required to search, evaluate, and output on history tape an analog internal variable type transfer equation when the equation is in resident memory:

$$t_{\substack{AIV(min) \\ (\text{Equation in memory})}} = t_{\substack{main \\ driver}} + t_{\substack{eqincore \\ (\text{Equation in memory})}} + t_{\substack{aeqvalue(min) \\ (\text{Equation in memory})}} + t_{tape}$$

$$t_{\substack{AIV(min) \\ (\text{Equation in memory})}} = 4.891 \text{ milliseconds}$$

3.9.7.8 Maximum expected time to evaluate related equations when the related equations are in resident memory:

$$t_{AIV(max)} = t_{\substack{B(max)P_B \\ (\text{Equation in memory})}} + t_{\substack{DIV(max)P_{DIV} \\ (\text{Equation in memory})}} + t_{\substack{E(max)P_E \\ (\text{Equation in memory})}} + t_{\substack{F(max)P_F \\ (\text{Equation in memory})}}$$

$$t_{\substack{AIV_{CR(max)} \\ (\text{Equation in memory})}} = 8.065 \text{ milliseconds}$$

Where: $t_{B(max)}$ (Equation in memory) = 6.094 milliseconds (See Section 3.2.7.7)

$t_{DIV(max)}$ (Equation in memory) = 8.554 milliseconds (See Section 3.3.7.6)

$t_{E(max)}$ (Equation in memory) = 11.406 milliseconds (See Section 3.8.7.6)

$t_{F(max)}$ (Equation in memory) = 7.082 milliseconds (See Section 3.7.7.7)

3.9.7.9 Minimum expected time to evaluate related equations when the related equations are in resident memory:

$$t_{AIV_{CR}(min)}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) = t_{B(min)}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) P_B + t_{DIV(min)}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) P_{DIV}$$

$$+ t_{E(min)}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) P_E + t_{F(min)}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) P_F$$

$t_{AIV_{CR}(min)}$ = 6.453 milliseconds

Where: $t_{B(min)}$ (Equation in memory) = 4.778 milliseconds (See Section 3.2.7.8)

$t_{DIV(min)}$ (Equation in memory) = 6.394 milliseconds (See Section 3.3.7.8)

$t_{E(min)}$ (Equation in memory) = 8.176 milliseconds (See Section 3.8.7.7)

$$t_{F(min)}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) = 7.056 \text{ milliseconds (See Section 3.7.7.8)}$$

3.9.7.10 Maximum time required to search, evaluate, and output on history tape an analog internal variable transfer equation and its related equations when all equations are in resident memory:

$$t_{AIV(max)\ total}\left(\begin{matrix}\text{Equation}\\ \text{in memroy}\end{matrix}\right) = t_{AIV(max)}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) + t_{AIV_{CR(max)}}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right)$$

$$= 15.116 \text{ milliseconds}$$

3.9.7.11 Minimum time required to search, evaluate, and output on history tape an analog internal variable transfer equation and its related equations when all equations are in resident memory:

$$t_{AIV(min)\ total}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) = t_{AIV(min)}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right) + t_{AIV_{CR(min)}}\left(\begin{matrix}\text{Equation}\\ \text{in memory}\end{matrix}\right)$$

11.344 milliseconds

3.10 Evaluation of Analog Equation (Portion Within the Slash Marks)

3.10.1 Expected time for the evaluation of an E type transfer equations (The portion within the slash marks):

$$E(t)_{E\text{-Calog}} = 225\ P_E + (86\ P_{1F} + 172\ P_{2F} + 1032\ P_{12F} + 1118\ P_{13F})\ P_F\ P_G + 200\ P + 351\ P_P + 1225\ P_T + 586\ P_G + 347 \cdot 1.75\times10^{6} \qquad 3\text{-}10$$

$E(t)_{E\text{-Calog}}$ = 2.6 milliseconds

(Equation 3-10 is based on the calog subroutine listed in Appendix A of Sperry Rand's "General Purpose Simulator System Study" Part 2 and Table 3-10.)

In Equation 3-10

P_E	The probability of an E parameter, given that it is a G type equation
P_F	The probability of an F parameter, given that it is a G type equation
P_{1E}, P_{2E}	The probability of 1E or 2E type parameters respectively given that the parameter was an E type
P_{1F}, P_{2F}, P_{12F}, P_{13F}	The probability of 1F, 2F 12F, or 13F type parameters respectively given that the equation contained F type parameters
P_G, P_T, P_P, P_{GP}	The probalities that the equation was of the G, T, P, and G or P type respectively.

3.10.2 Expected time spent for the evaluation of F type transfer equation (The portion within the slash marks).

Transfer Equation Type	Equation Description				Probability of Occurance				
	No. Of Polynomial Equations (P)	No. Of Normal Non-Additive Equations (T)	No. of Sumation Equations (G)	No. of Multiplication Equations (H)	P_P	P_T	P_G	P_H	P_{GH}
E	146	423	71	0	0.228	0.660	0.111	0	0
F	30	176	210	28	0.067	0.396	0.472	0.063	0.536

Transfer Equation Type	I,M,(M),N Parameters Within Slash Marks		Number Of E And F Type Parameters* Within Slash Marks											
	No of Parameters/Term	No. of Terms	1E	2E	1F	2F	3F	12F	13F	15F	18F	1E & 1F		
E	2	423	48	0	9	28	0	2	6	0	0	0		
	4	217												
F	2	176	9	10	20	168	7	0	0	1	1	15		
	4	268												

Probability Of Occurance													
P_E	P_F	P_{1E}	P_{2E}	P_{1F}	P_{2F}	P_{3F}	P_{12F}	P_{13F}	P_{15F}	P_{18F}	P_{EF}		
0.516	0.484	1	0	0.2	0.622	0	0.044	0.133	0	0	0		
0.082	0.852	0.473	0.527	0.101	0.852	0.035	0	0	0.005	0.005	0.064		

*There is at least one E or F Type parameter in every G and H Type Equation. P and T Type equations have no E and F parameters.

$$E(t)_{F\text{-}Calog} = 1.75\times10^{6}\{[(225\ P_{1E} + 450\ P_{2E} + 155\ P_{EF} + (86\ P_{1F} + 172\ P_{2F} + 258\ P_{3F} + 1290\ P_{15F} + 1548\ P_{18F})P_{F}]\ P_{GH} + 200\ P_{T} + 400\ P_{GHP} + 351\ P_{P} + 1225\ P_{T} + 586\ P_{G} + 405\ P_{H} + 211\}$$

$$E(t)_{F\text{-}Calog} = 2.54 \text{ milliseconds}$$

Where: P_{15F}, P_{18F} The probability of 15F or 18F type parameter respectively, given that the equations contained F type parameters

P_{EF} The probability that the equation contained 1E and 1F type parameters.

P_{15F}, P_{18F} The probability of 15F or 18F type parameters respectively, given that the equation contains F type parameters.

P_{GH} The probability that the equation is either G or H type.

P_{GHP} The probability that the equation is G, H, or P type.

3.10.3 Expected time spent for the evaluation of analog internal variable transfer equations, the portion within the slash marks:

$$E(t) = 1,75\times10^{6}\ (40 + 86 + 290) = 0.728 \text{ millisecond}$$

3.11 DOs

3.11.1 Average number of cross references per DO

$$\bar{X}_{DO} = \frac{1}{277} \sum_{i=1}^{11} X_{DO_i} F_{DO_i} = 1.9 \text{ related equations}$$

(See Table 3-11 for X_{DO_i} and F_{DO_i} values)

3.11.2 Memory Requirements:

$$\text{Number of words} = \sum_{i=1}^{9} X_{DO_i} F_{DO_i} = 525 \text{ words}$$

3.11.3 Response Time To A DO

The response time to a DO from the 110-A computer is dependent on the number and type of transfer equations on the queue stacks waiting to be evaluated, the number and type of related equations scheduled for evaluation by the DO, and the time required to process the DO through the adochk and acompdo subroutines. Since the test conditions change from one request to another, it becomes difficult to predict with any certainty the response time of a DO without the following simplifications:

1) There are no equations in the queue stacks to be evaluated.
2) The DO will schedule for evaluation only one related equation.
3) Thirteen out of twenty-four bits are interrogated for a change in status.

With these assumptions the problem reduces to simply adding the times for checking and processing a DO through the log tables (See adochk and acompdo subroutines listed in Appendix A of Sperry Rand's "General Purpose Simulator System Study" Part 2) and predicting the type of equation scheduled for evaluation by the DO. The expected time (t_{DO}) for the evaluation of the transfer equation is simply.

$$t_{DO} = \sum_{i=1}^{n} t_i P_i \qquad 3\text{-}11$$

In equation 3-11, t_i is the time spent by the simulator to search, access, evaluate, and record on history tape the ith type of transfer equation and P_i is the probability of occurance for the ith type of of transfer equation. t_i for each equation type has been calculated in sections 3.1 through 3.9. P_i can be established from the listing of the related equations for the DOs of the instrument unit grouped in Table 3-12.

Only the maximum solution times of each equation type for both equation not in memory and equation in memory will be used to calculate the response time to a DO. If desired the minimum response times can be obtained by simply replacing the maximum solution times with the minimum solution time.

$$t_{\left(\begin{array}{l}\text{Maximum response to a DO}\\ \text{with equation not in memory}\end{array}\right)} = t_{Acompdo} + t_{A(max)}P_A \quad \text{(Initial Evaluation)}$$

$$+ t_{B(max)}P_B \quad \text{(Equation in rad)} + t_{E(max)}P_E \quad \text{(Equation in rad)} + t_{F(max)}P_F \quad \text{(Equation in rad)}$$

$$+ t_{DIV(max)}P_{DIV} \quad \text{(Equation in rad)} + t_{DDAS(max)}P_{DDAS} \quad \text{(Equation in rad)}$$

$$+ t_{Bus(max)}P_{Bus} \quad \text{(Equation in memory)}$$

Where: $t_{Acompdo}$ = 1.283 milliseconds By the simulator to search and process a DO through the log Tables.

$$t_{\left(\begin{array}{l}\text{Maximum response time to a DO}\\ \text{with equation not in memory}\end{array}\right)} = 28.621 \text{ milliseconds}$$

$$t_{\left(\begin{array}{l}\text{Maximum response to a DO}\\ \text{with equation in memory}\end{array}\right)} = t_{Acompdo} + t_{A(Max)}{}^{P}{}_{A} + t_{B(max)}{}^{P}{}_{B}$$

$$+ t_{E(max)}{}^{P}{}_{E} + t_{F(max)}{}^{P}{}_{F} + t_{DIV(max)}{}^{P}{}_{DIV}$$

$$t_{DDAS(max)}{}^{P}{}_{DDAS} + t_{Bus(max)}{}^{P}{}_{Bus}$$

$$t_{\left(\begin{array}{l}\text{Maximum response time to a DO}\\ \text{with equation in memory}\end{array}\right)} = 8.933 \text{ milliseconds}$$

Table 3-11 DO Cross References

Grouped Data

Equation Length (X_{DO})	Frequency Of Occurrence (F_{DO})	Frequency Of Occurrence By Equation Type						
		A	B	E	F	DIV	DDAS	BUS
1	211	5	124	1	5	60	10	6
2	41	0	27	3	2	39	11	0
3	8	0	4	0	1	18	1	0
4	2	0	5	1	1	1	0	0
5	4	0	5	4	4	6	1	0
6	1	0	0	3	3	0	0	0
8	1	0	0	0	0	8	0	0
14	3	0	1	12	0	29	0	0
15	2	0	2	12	0	16	0	0
17	2	0	0	0	0	34	0	0
20	3	0	5	4	5	46	0	0
Totals	277	5	173	40	21	257	26	6

Probability of Occurrence	P_A=	P_B=	P_E=	P_F=	P_{DIV}=	P_{DDAS}=	P_{BUS}=
	.009	.329	.076	.040	.490	.044	.001

3.12 Switches

3.12.1 Average number of cross references per switch action

$$\overline{X}_S = \frac{1}{41.5} \sum_{i=1}^{9} X_{S_i} F_{S_i} = 1.41 \text{ equations}$$

(See table 3-12 for X_{S_i} and F_{S_i} values)

3.12.2 Memory Requirements:

$$\text{Number of words} = \sum_{i=1}^{9} X_{S_i} F_{S_i} = 586 \text{ words}$$

3.12.3 Response Time To a Switch Action

A switch action in the DEE-6 simulator system is simulated by the card reader. It schedules for evaluation certain transfer equations provided there are no equations in the queue stacks waiting to be evaluated or DO from the 110-A computer to be processed and provided the card reader is ready. Since the test conditions change from switch action to switch action it becomes difficult to predict with any certainty the response time to a switch without the following simplifications:

1) There are no equations in the queue stack waiting to be evaluated.
2) There are no DOs to be processed
3) The card reader has been turned on
4) A switch action will schedule for evaluation only one transfer equation

With these simplifications the problem reduces to simply adding the overhead times spent to process a switch action (See a swtchk subroutine listed in Appendix A of Sperry Rands "General Purpose Simulator System Study" Part 2), the time it takes the card reader to read a card, plus the time it takes to solve the equation specified by the switch action.

Table 3-12 Switch Cross References
Grouped Data

Equation Length (X_S)	Frequency Of Occurance (F_S)	Frequency of Occurance by Equation Type						
		A	B	E	F	DIV	DDAS	BUS
1	329	252	6	1	0	52	7	13
2	53	32	8	8	0	4	8	6
3	13	1	6	0	0	26	1	5
4	11	3	7	3	3	26	4	1
5	1	0	0	0	5	0	0	0
6	2	0	4	2	0	6	0	0
7	4	0	1	0	0	27	0	0
10	1	0	0	0	0	10	0	0
13	1	0	3	3	0	7	0	0
Total	415	288	35	17	5	198	20	25

Probability of Occurance	P_A=.490	P_B=.059	P_E=.029	P_F=.009	P_{DIV}=.337	P_{DDAS}=.034	P_{BUS}=.042

Again the expected time for the evaluation of the transfer equation is simply:

$$t_S = \sum_{i=1}^{n} t_i P_i$$

Where t_i is the time spent by the simulator to search, access, evaluate and record on history tape the ith type of transfer equation and P_i is the probability of occurance of the ith type of transfer equation. t_i for each type of equation has been computed in Sections 3.1 through 3.9. P_i can be established from the listings of the related equations for the switches of the Instrument Unit grouped in table 3-12.

Again, only the maximum solution times of each equation type will be used to calculate the response time to a switch action.

$$t_{\left(\begin{array}{l}\text{Maximum response time to a switch}\\ \text{with equation not in memory}\end{array}\right)} = t_{Aswtchk} + t_{Card\ reader}$$

$$+ t_{A(max)}P_A + t_{B(max)}P_B$$
$$\left(\begin{array}{l}\text{Equation}\\ \text{in rad}\end{array}\right) \quad \left(\begin{array}{l}\text{Equation}\\ \text{in rad}\end{array}\right)$$

$$+ t_{E(max)}P_E + t_{F(max)}P_F$$
$$\left(\begin{array}{l}\text{Equation}\\ \text{in rad}\end{array}\right) \quad \left(\begin{array}{l}\text{Equation}\\ \text{in rad}\end{array}\right)$$

$$+ t_{DIV(max)}P_F + t_{DIV(max)}$$
$$\left(\begin{array}{l}\text{Equation}\\ \text{in rad}\end{array}\right) \quad \left(\begin{array}{l}\text{Equation}\\ \text{in rad}\end{array}\right)$$

$$+ t_{DDAS(max)}P_{DDAS} + t_{BUS(max)}P_{BUS}$$
$$\left(\begin{array}{l}\text{Equation}\\ \text{in rad}\end{array}\right) \quad \left(\begin{array}{l}\text{Equation}\\ \text{in rad}\end{array}\right)$$

$$t_{\left(\begin{array}{l}\text{maximum response time to a switch}\\ \text{with equation not in memory}\end{array}\right)} = 0.842 + 150 + 26.268 = 177.11 \text{ milliseconds}$$

$$t_{\left(\begin{array}{l}\text{maximum response time to a switch}\\ \text{with equation}\end{array}\right)} = t_{\text{Aswtchk}} + t_{\text{Card reader}} + t_{\text{A(max)}}P_{\left(\begin{array}{l}\text{Equation}\\ \text{in memory}\end{array}\right)\text{A}} + t_{\text{B(max)}}P_{\left(\begin{array}{l}\text{Equation}\\ \text{in memory}\end{array}\right)\text{B}} + t_{\text{E(max)}}P_{\left(\begin{array}{l}\text{Equation}\\ \text{in memory}\end{array}\right)\text{E}} + t_{\text{F(max)}}P_{\left(\begin{array}{l}\text{Equation}\\ \text{in memory}\end{array}\right)\text{F}} + t_{\text{DIV(max)}}P_{\left(\begin{array}{l}\text{Equation}\\ \text{in memory}\end{array}\right)\text{DIV}} + t_{\text{DDAS(max)}}P_{\left(\begin{array}{l}\text{Equation}\\ \text{in memory}\end{array}\right)\text{DDAS}} + t_{\text{BUS(max)}}P_{\left(\begin{array}{l}\text{Equation}\\ \text{in memory}\end{array}\right)\text{BUS}}$$

$$t_{\left(\begin{array}{l}\text{max. response time to a switch}\\ \text{with equation in memory}\end{array}\right)} = 0.842 + 150 + 6.763 = 157.605 \text{ milliseconds}$$

4.0 Conclusion and Recommendations

Table 4-1 is a summary of the mathematical analysis of the DEE-6 simulator system. The entire Instrument Unit data base consists of 2326 equations corresponding to 62,733 words. The DEE-6 simulator can provide storage for up to 625,000 24-bit words. Assuming similar data basis for the remaining Saturn stages, the entire Saturn data base would only occupy 50.9 percent of the DEE-6 simulator storage capacity. This more than adequately satisfies the storage requirements of the Saturn data base.

The solution times for the average discrete transfer equations located in the rad range from a minimum of 23.532 milliseconds for the A type to a maximum of 28.957 milliseconds for the discrete internal variable type equation. Conversely, if the equation were to be located in resident memory its solution time would be substantially reduced. It would range from a minimum of 4.726 milliseconds to a maximum of 8.554 milliseconds.

The average solution time for the analog type transfer equation is somewhat longer. It ranges from a minimum of 23.720 milliseconds for the analog internal variable equation type located in the RAD to a maximum of 31.796 milliseconds for E type. This however, is somewhat deceiving since all of the related equation must be solved before another solution of the analog equation takes place. This substantially extends the solution time of the analog equation. When the related equations are taken into consideration (See Table 4-1) the solution time for the average analog equation located in the rad ranges from a minimum of 38.860 milliseconds to a maximum of 54.919 milliseconds. On the other hand if the analog equation and all of its related equations are located in resident memory the solution time would be reduced to a minimum of 9.86 milliseconds and a maximum of 15.116 milliseconds. In the DEE-6 simulator system, only 156 equations out of 2326 can be located in resident memory at any one time. It is highly unlikely that the analog

Table 4-1. Summary of Memory Requirements and Solution Times

Equation Type	Memory Req. (Words)	Solution Times			
		Equation Not In Memory		Equation In Memory	
		Max. Time (msec)	Min. Time (msec)	Max. Time (msec)	Min. Time (msec)
A Type Discrete	5254	24.094	23.532	5.284	4.726
B Type Discrete	6333	25.010	23.685	6.094	4.778
Discrete Internal Variable	17551	28.957	26.797	8.554	6.394
DDAS Discrete ,	4498	24.597	23.777	5.624	4.814
Power BUS Discrete	2347	28.687	27.137	8.247	6.737
E Type Analog	11638	31.796	26.246	11.406	8.176
E Type Analog and its Related Equations		44.410	38.860	14.175	10.845
F Type Analog	8206	29.416	25.956.	7.082	7.056
F Type Analog and its Related Equations		42.669	39.209	9.886	9.860
Analog Internal Variable	987	25.970	23.720	7.051	4.891
Analog Internal Variables and its Related Equations		54.919	50.269	15.116	11.344
VXXX = EXXX//P.V.,D.V./$	4808	23.667		5.008	
VXXX = EXXX//P.V.,D.V.//$ and its Related Equations		51.991		12.963	
Response Time to A Switch Action	525	177.110		157.605	
Response Time to A DO	586	28.621		8.933	
Total	62733				

equation of interest and all of its related equations would be in resident memory. Therefore, to insure the solution of an analog equation by the DEE-6 simulator, its solution interval must be greater than 55 milliseconds. A substantial number of analog equations in the Instrument Unit specify a solution interval of 8 milliseconds. It appears that these equations could not be solved correctly by the DEE-6 simulator system.

To insure the solution of an analog equation with a solution interval of 8 milliseconds, would require an extensive modification of both the DEE-6 simulator hardware and software. For example all equations must be located in resident memory. The software must be made more efficient by eliminating the search and retrieval subroutines. During peak loads some parallel solutions might be necessary. Detail description for the improvement of the DEE-6 simulator system will be given in Sperry Rand's final report.

APPENDIX A

Sample Computer Program
(Polynomial Least Square Fit)

```
C       PROGRAM TO CALCULATE POLYNOMIAL LEAST SQUARE FIT
C       EQUATION LENGTH VS NO. OF OR TERMS (F-TYPE ANALOG)
        DIMENSION X(30),Y(30),W(30),ALPHA(30),BETA(30),C(30),T1(30),
       1T2(30),T3(30),A(30),YAVE(30),YY(30),IND1(1),IND2(1)
        READ(5,10)(X(I),I=1,13)
   10   FORMAT(13F6.0)
        READ(5,20)(Y(I),I=1,13)
   20   FORMAT(13F6.2)
        READ(5,25)YAVE(1)
   25   FORMAT(1F10.2)
        READ(5,30)N,L,J
   30   FORMAT(1I2,2I1)
        K=0

   40   CALL ORTHLS(X,Y,W,N,L,J,C,ALPHA,BETA,K,T1,T2,T3,IND1)
   50   CALL COEFS(J,C,ALPHA,BETA,K,A,T1,T2,T3,IND2)
C
        WRITE(6,60)(A(I),I=1,5)
   60   FORMAT(5E16.6)
C
        YDS=0.
        YNS=0.
        DO 70 I=1,13
        YY(I)=A(1)+A(2)*(X(I))**1.0+A(3)*(X(I))**2.0+A(4)*(X(I))**3.0
       1+A(5)*(X(I))**4.0
        WRITE(6,85)(YY(I))
   85   FORMAT(1E15.6)
        YN=(YY(I)-YAVE(1))**2.0
        YNS=YN+YNS
        YD=(Y(I)-YAVE(1))**2.0
        YDS=YD+YDS
   70   CONTINUE
C
        RHO=SQRT(YNS/YDS)
        WRITE(6,80)(RHO)
   80   FORMAT(E16.6)
C
        IF(K.GE.4) CALL EXIT
        IF(RHO.GT.0.98) CALL EXIT
        IF(K.LT.4) K=K+1
        GO TO 40
C
        END
```

APPENDIX B

Example of a Switch Action

EXAMPLE OF A SWITCH ACTION

1.0 Description of Example

A switch action was chosen to demonstrate the response time of the DEE-6 Simulator System and to utilize the solution time results calculated in sections 3.1 through 3.10 of Sperry Rand's "General Purpose Simulator System Study Part 3." The example chosen is switch action S8021 and it is taken from the post history printouts of the Instrument Unit. Preliminary results of this example were given in Sperry Rand's "General Purpose Simulator Study Part 2." Essentially switch action S8021 called for the solution of 66 equations in four solution levels (see figure 1).

In level one, switch action S8021 called for the solution of three Discrete Internal Transfer Equations. Associated with these equations are 127 related equations. The DEE-6 Simulator will schedule for evaluation only those related equations that are specified by the transfer equations which changed in status. In level one only one transfer equation changed in status. This equation called for the solution of 54 related equations. These equations form solution level two as delineated in figure 1.

In level two, both analog and discrete types of transfer equations will be scheduled for evaluation. The analog equations, types E and F, will be placed in the special queue and will be evaluated first. The schedule for evaluation will be on a first come first serve basis. The discrete equations will be placed in the normal queue and will be scheduled for evaluation when all of the analog equations and their related equations have been evaluated, i.e., when the special queue is empty. Again, the schedule for evaluation will be on a first come first serve basis.

In level two, three discrete transfer equations and all of the analog transfer equations changed in status. These equations in turn scheduled eight discrete types of transfer equations for evaluation in level three. (See figure 1 for the specific type of transfer equation involved for each case). The E Type of analog equations specify an evaluation interval of 8 milliseconds. The minimum solution time for the E type equation when the equation is located in resident memory exceeds the specified

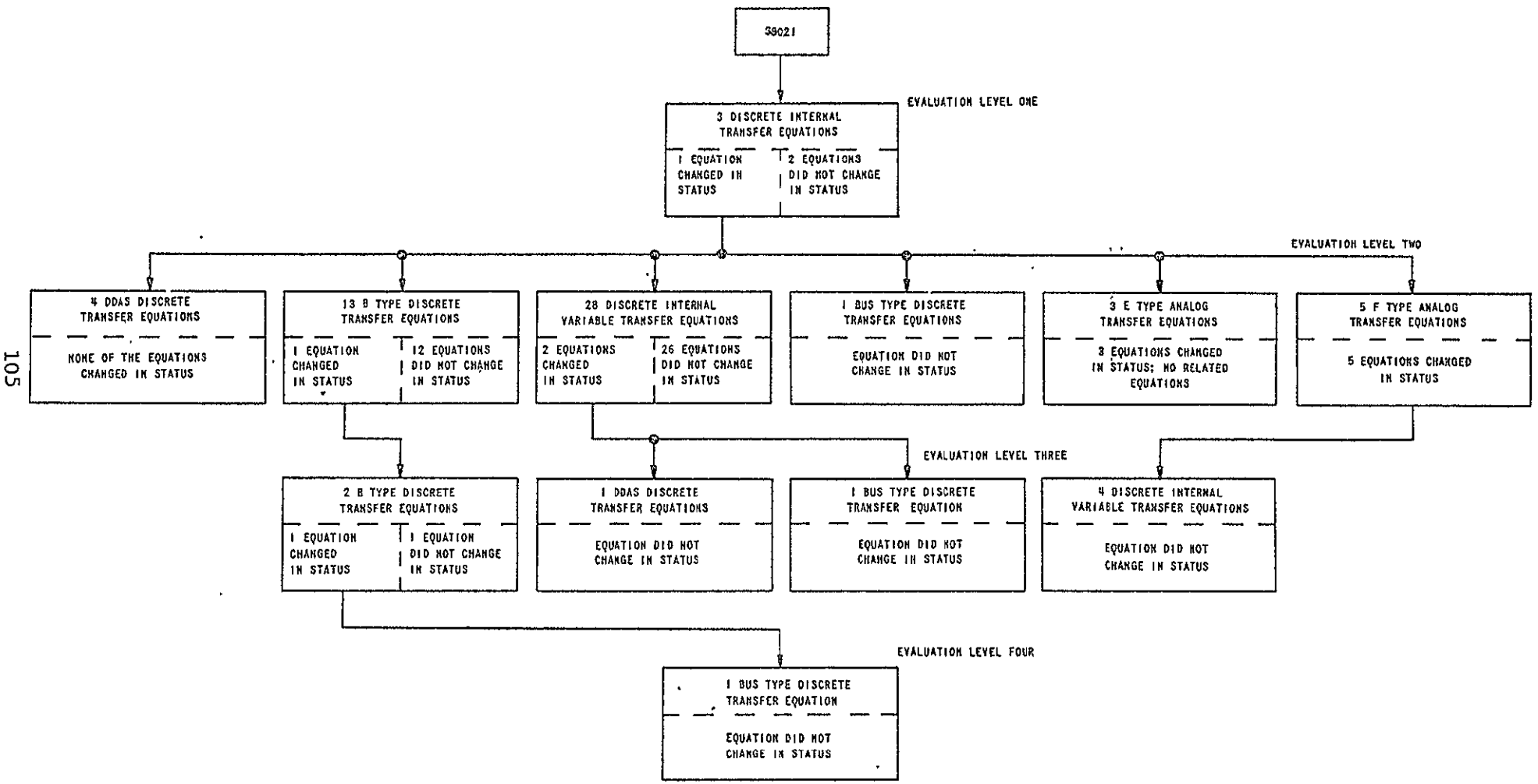


Figure 1. Equation Evaluation Scheduled by Switch Action S8021

8-millisecond evaluation interval (see section 3.8.7.7). It is highly unlikely, then, that the DEE-6 Simulator can solve these equations correctly even though there are no related equations to add to their solution time. For this example it will be assumed that a solution will be provided by the simulator although not necessarely correct. The number of evaluations for both the E and F type of analog transfer equations is dependent on the status logic. From the available data there is no way to determine with any degree of certainty how many evaluations will take place. To allow the calculation for the response time of the switch action it will be assumed that a total of 10 evaluations for each equation will take place from the time the status logic changed from false to true and back. In all cases the equations for the second through the tenth evaluation will be assumed to be in resident memory. This is a reasonable assumption since the analog equations are placed in the special queue and their solution takes preference over the discrete equations.

The F type of analog equations specify an evaluation interval of 80 milliseconds. The simulator should handle this without any difficulty.

The five, F type analog equations called for the evaluation of four Discrete Internal Variable Equations. In the example and under the preceding assumptions, these discrete internal variable equations will be evaluated twice; the first time when the status logic of the analog equation becomes true and the second time immediately after the status logic reverts back to false. All other discrete equations in levels one through four will be evaluated only once.

In level three all of the equations are the discrete type. Out of the eight equations scheduled for evaluation only one equation changed in status. Associated with this equation is one related equation which becomes evaluation level four. The fourth level equation did not change in status thereby terminating the evaluation for the chosen example.

The total response to the switch action S8021 is the solution of all 66 equations evaluated in the described sequence. To obtain a measure of the total response time the results of sections 3.1 through 3.10 will be used. The response time is first calculated assuming all 66 equations are located in the RAD and it will be repeated assuming all equations are located in resident memory. In each case a maximum and a minimum response time will be calculated based on the maximum and minimum time it takes to process each equation through the AEQVALUE subroutine.

2.0 Response Time Calculation to Switch Action S8021

The response time calculations for the switch action S8021 will be based on the assumptions of the preceding section . Two maximum and two minimum response times will be calculated based on the maximum and minimum times required to process the equations by the AEQVALUE subroutine One set of maximum and minimum calculations will assume that all equations initially reside in the RAD and the other set will assume all equations in resident memory. For repeated evaluations the equations are assumed to be in resident memory.

2.1 Maximum response time to switch action S8021, assuming all equations are initially located in the RAD:

$$t_{\substack{S8021\ (max) \\ (\text{Equations in RAD})}} = 15\, t_{\substack{B\ (max) \\ (\text{Equation in RAD})}} + 35\, t_{\substack{DIV\ (max) \\ (\text{Equation in RAD})}} + 5\, t_{\substack{DDAS\ (max) \\ (\text{Equation in RAD})}}$$

$$+ 3\, t_{\substack{pB\ (max) \\ (\text{Equation in RAD})}} + 3\, t_{\substack{E\ (max) \\ (\text{Equation in RAD})}} + 3\, n_E t_{\substack{E\ (max) \\ (\text{Equation in Memory})}} + 5\, t_{\substack{F\ (max) \\ (\text{Equation in RAD})}}$$

$$+ 5\, n_F t_{\substack{F\ (max) \\ (\text{Equation in Memory})}} + 4\, m_F t_{\substack{DIV\ (max) \\ (\text{Equation in Memory})}}$$

$$t_{\substack{S8021\ (max) \\ (\text{Equation in RAD})}} = 2501.027 \text{ milliseconds}$$

Where:

$$t_{\substack{B\ (max) \\ (\text{Equation in RAD})}} = 25.01 \text{ milliseconds (see section 3.2.6.9)}$$

$t_{DIV\ (max)}$ (Equation in RAD) = 28.957 milliseconds (see section 3.3.6.8)

$t_{DDAS\ (max)}$ (Equation in RAD) = 24.597 milliseconds (see section 3.4.6.8)

$t_{PB\ (max)}$ (Equation in RAD) = 28.687 milliseconds (see section 3.5.6.7)

$t_{E\ (max)}$ (Equation in RAD) = 31.796 milliseconds (see section 3.8.6.7)

$t_{F\ (max)}$ (Equation in RAD) = 29.416 milliseconds (see section 3.7.6.6)

$t_{E\ (max)}$ (Equation in Memory) = 11.406 milliseconds (see section 3.

$t_{F\ (max)}$ (Equation in Memory) = 7.082 milliseconds (see section 3.7.7.7)

$t_{DIV\ (max)}$ (Equation in Memory) = 8.554 milliseconds (see section 3.

$n_E = 9$ The number of repeated evaluations of the E type analog transfer equations

$n_F = 9$ The number of repeated evaluations of the F type analog transfer equations

$m_F = 1$ The number of repeated evaluations of the Discrete Internal Variable Transfer Equations. (These are the related equations specified by the F type analog transfer equations)

!.2 Minimum response time to switch action S8021, assuming all equations are located in the RAD:

$$t_{S8021\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) = 15\ t_{B\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) + 35\ t_{DIV\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) + 5\ t_{DDAS\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) + 3\ t_{PB\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right)$$

$$+ 3\ t_{E\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) + 3\ n_E t_{E\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 5\ t_{F\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) + 5\ n_F t_{F\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right)$$

$$+ 4\ m_F\ t_{DIV\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right)$$

$$t_{S8021\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) = 22651.840 \text{ milliseconds}$$

Where:

$$t_{B\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) = 23.685 \text{ milliseconds (see section 3.2.6.9)}$$

$$t_{DIV\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in RAD}\end{matrix}\right) = 26.797 \text{ milliseconds (see section 3.3.6.8)}$$

$t_{DDAS\ (min)}$ (Equation in RAD) = 23.777 milliseconds (see section 3.4.6.9)

$t_{PB\ (min)}$ (Equation in RAD) = 27.138 milliseconds (see section 3.5.6.9)

$t_{E\ (min)}$ (Equation in RAD) = 26.246 milliseconds (see section 3.8.6.8)

$t_{E\ (min)}$ (Equation in Memory) = 8.176 milliseconds (see section 3.8.7.7)

$t_{F\ (min)}$ (Equation in RAD) = 25.956 milliseconds (see section 3.7.6.7)

$t_{F\ (min)}$ (Equation in Memory) = 7.056 milliseconds (see section 3.7.7.8)

$t_{DIV\ (min)}$ (Equation in Memory) = 6.394 milliseconds (see section 3.3.7.8)

2.3 Maximum response time to switch action S8021, assuming all equations are located in resident memory:

$$t_{S8021\ (max)}\ (\text{Equation in Memory}) = 15\ t_{B\ (max)}\ (\text{Equation in Memory}) + 35\ t_{DIV\ (max)}\ (\text{Equation in Memory}) + 5\ t_{DDAS\ (max)}\ (\text{Equation in Memory}) + 3\ t_{PB\ (max)}\ (\text{Equation in Memory})$$

$$+ 3\, t_{E\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 3\, n_E t_{E\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 5\, t_{F\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 5\, n_F t_{F\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right)$$

$$+ 4\, m_F t_{DIV\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right)$$

$$t_{S8021\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) = 1177.163 \text{ milliseconds}$$

Where:

$$t_{B\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) = 6.094 \text{ milliseconds (see section 3.2.7.7)}$$

$$t_{DDAS\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) = 5.624 \text{ milliseconds (see section 3.4.7.7)}$$

$$t_{PB\ (max)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) = 9.249 \text{ milliseconds (see section 3.5.7.6)}$$

2.4 Minimum response time to switch action S8021, assuming all equations are located in resident memory:

$$t_{\substack{S8021\ (min)\\ \text{Equation}\\ \text{in Memory}}} = 15\, t_{B\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 35\, t_{DIV\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 5\, t_{DDAS\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 3\, t_{PB\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right)$$

$$+ 3\, t_{E\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 3\, n_E t_{E\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right) + 5\, t_{F\ (min)}\left(\begin{matrix}\text{Equation}\\ \text{in Memory}\end{matrix}\right)$$

$$+ 5 \cdot n_F \cdot t_{F\ (min)_{\left(\begin{array}{c}\text{Equation}\\ \text{in Memory}\end{array}\right)}} + 5\ m_F\ t_{DIV\ (min)_{\left(\begin{array}{c}\text{Equation}\\ \text{in Memory}\end{array}\right)}}$$

$t_{S8021\ (min)_{\left(\begin{array}{c}\text{Equation}\\ \text{in Memory}\end{array}\right)}}$ = 955.221 milliseconds

Where

$t_{B\ (min)_{\left(\begin{array}{c}\text{Equation}\\ \text{in Memory}\end{array}\right)}}$ = 4.778 milliseconds (see section 3.2.7.8)

$t_{DDAS\ (min)_{\left(\begin{array}{c}\text{Equation}\\ \text{in Memory}\end{array}\right)}}$ = 4.814 milliseconds (see section 3.4.7.8)

$t_{PB\ (min)_{\left(\begin{array}{c}\text{Equation}\\ \text{in Memory}\end{array}\right)}}$ = 6.737 milliseconds (see section 3.5.7.7)

3.0 Additional Conclusions and Recommendations

The Instrument Unit data base consists of 2326 equations. Of these, 462 are of the E and F type analog equations. Out of the 462 equations, 372 specify and evaluation interval of 10 milliseconds or less. From sections 3.8.7.7 and 3.7.7.8 the minimum evaluation time for the E and F type equations (when the equations are located in resident memory) are 8.176 and 7.056 milliseconds respectively. It is doubtful that the DEE-6 Simulator can solve these equations correctly; particularly, if related equations are involved and more than two evaluations are required. The remaining 1954 equations of the Instrument Unit data base are either discrete or analog with an evaluation interval of greater than 10 milliseconds. It appears that the DEE-6 Simulator can solve these correctly although not necessarily within the desired vehicle response time.

The DEE-6 Simulator queue assignments for the Instrument Unit data base are as follows:

1.	Normal queue	1000 words
2.	Timed queue	512 words
3.	Special queue	50 words
4.	Clocked queue	50 words
	Total	1612 words

It takes one memory location to reference one equation. Based on the above queue assignments the DEE-6 Simulator can schedule for evaluation more than 1600 equations. The queue allotments are established during the preprocessing phase of the program and can be increased if desired.

The preceding example has shown that the response time of the DEE-6 Simulator can be reduced by a factor of two if all of the equations are located in resident memory. Farther reductions in the response time can be achieved by more efficient software and by parallel evaluations. For example, with ample memory the existing cross

reference identification address of each equation can be replaced with a relative storage address thereby, eliminating all of the existing search and retrieval subroutines. This coupled with the parallel processing capability of another computer could substantially reduce the equation evaluation time and thus the response time of the simulator. Details of the concept are provided in Sperry Rand's final report.

TABLE 49

HEMATOLOGY DATA*

Subject	Day of Study	RBC	HGB	HCT	WBC	MCV	MCH	MCHC	RC
	C 1	4.88	15.7	46	6300	95	32	34.5	.6
	C 9	5.43	15.7	47	6600	87	28.5	33.5	1.4
	B 2	5.50	16.2	46.5	6800	85.5	29	35	1.5
6	B14	4.52	16.4	47	7200	104	36	36	1.1
	B28	5.15	15.5	47	7200	92	30	33	.4
	A 7	4.98	14.9	42	6500	85	30	36	1.2
	A13	5.11	15.3	43	7000	85	29.5	35.5	2.0
	C 9	5.02			7500	82	27.5	34	.8
	B 2	4.69	12.8	39	6900	84	27	33	1.7
	B14	4.80	14.3	40	5700	84	29.5	35.5	1.1
7	B28	4.80	13.3	42	5500	88.5	27.5	32	1.2
	R 7	4.58	13.9	39	5600	86.5	30	36	1.7
	R14	4.52	12.8	37	5700	83	27	34.6	1.2
	C 1	5.64	15.1	46	7100	83	27	31	1.0
	C 9	4.75	13.7	40	6300	85	35	29	.5
	B 2	4.74	14.3	40	5200	87	30	35	3.3
8	B14	4.75	14.3	41	6100	87	30	35	1.0
	B28	5.80	14.3	45	8000	78.5	24.5	32	[illegible]
	R 7	4.44	13.1	38	9000	86.5	30	34.5	[illegible]
	R14	4.50	13.5	38	6600	85	30	36	[illegible]

DESIGN • DEVELOPMENT • STUDIES •

ENGINEERING SUPPORT

- RANGE INSTRUMENTATION
- AUTOMATIC CHECKOUT
- GUIDANCE AND CONTROL SYSTEMS
- TELEMETRY
- ELECTRONIC POWER SYSTEMS
- CONFIGURATION MANAGEMENT
- FLIGHT DYNAMICS AND SIMULATION
- ELECTRICAL AND ELECTRONIC SYSTEMS
- SPACE AND SATELLITE COMMUNICATIONS
- NAVIGATION SYSTEM ANALYSIS
- RELIABILITY ANALYSIS

FABRICATION

SPERRY RAND

SPACE SUPPORT DIVISION
HUNTSVILLE, ALABAMA