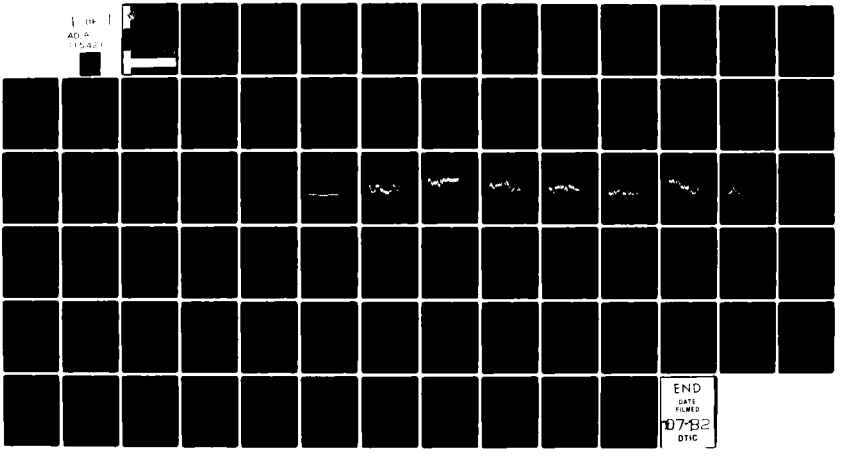
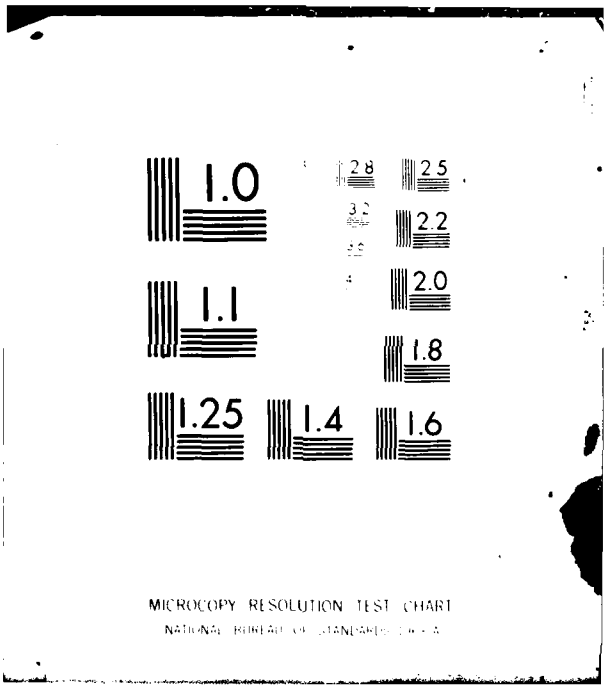
AD-A115 421 NAVAL TRAINING EQUIPMENT CENTER ORLANDO FL F/G 9/2
DESIGN OF A MICROPROCESSOR-CONTROLLED LINKAGE FOR SIMULATOR APP--ETC(U)
APR 82 L D HEALY
UNCLASSIFIED NAVTRAEQUIPC-IH-334 NL



END
DATE FILMED
07-82
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
MICROCOPY RESOLUTION TEST CHART

(12)

Technical Report: NAVTRAEQUIPCEN IH-334

Design of a Microprocessor-Controlled Linkage for Simulator Applications

Leonard D. Healy
Computer Systems Laboratory
Research Department
Naval Training Equipment Center
Orlando, FL 32813

April 1982

Final Report October 1980 - June 1981

NAVTRAEQUIPCEN IH-334

DESIGN OF A MICROPROCESSOR-CONTROLLED LINKAGE FOR SIMULATOR APPLICATIONS

LEONARD D. HEALY
Computer Systems Laboratory

April 1982

Approved:

C. F. SUMMER
Acting Head, Computer Systems Laboratory

DAVID P. GLENN
Director of Research

NAVAL TRAINING EQUIPMENT CENTER

ORLANDO, FL 32813

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NAVTRAEQUIPCEN IH-334	2. GOVT ACCESSION NO.: AD-A115 421
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): DESIGN OF A MICROPROCESSOR-CONTROLLED LINKAGE FOR SIMULATOR APPLICATIONS	5. TYPE OF REPORT & PERIOD COVERED: Final Report for period October 1980 - June 1981
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): LEONARD D. HEALY	8. CONTRACT OR GRANT NUMBER(s):
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Computer Systems Laboratory, Research Department, Naval Training Equipment Center, Orlando, FL 32813	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NAVTRAEQUIPCEN Task 0742
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Training Equipment Center, Orlando, FL 32813	12. REPORT DATE: April 1982
	13. NUMBER OF PAGES: 78
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

computers
microcomputers/microprocessors
minicomputers
real-time simulation systems
computer architecture
input/output
computer interface

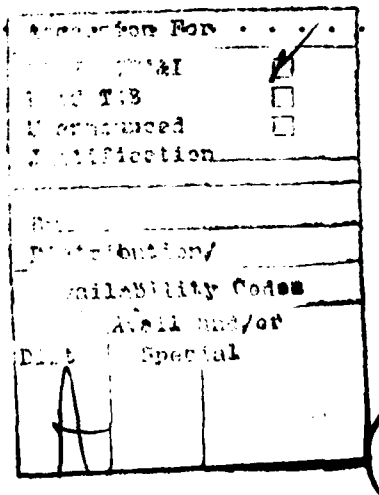
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes a method of linking a host computer to external devices and presents an implementation of such a system using multiple microcomputers. The technique used takes advantage of the statistical properties of simulator data to reduce the amount of data transferred, thereby insuring that those variables that change are updated with minimum delay. Implementation of linkage using multiple microcomputers provides a modular system easily tailored to various simulator applications.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TABLE OF CONTENTS



LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (CONTINUED)

LIST OF TABLES

SECTION I

INTRODUCTION

Simulators require a means of connecting the digital computer, which represents information by bit patterns in its memory, to external simulation equipment, which represents information by switch closures, indicator lights, control stick deflection, etc. The linkage between the computer and the external devices must handle two classes of data: (1) analog data, where the computer information is translated into a physical quantity such as a voltage or a current, and (2) discrete data, where the information in the computer is translated into on/off signals such as switch closures. The translation from analog and discrete data to the computer representation of information is also required.

Most simulators perform the conversion in hardware that is physically adjacent to the central computer, but this requires hundreds of interconnection lines. A block diagram of such a system is shown in figure 1. This method provides the quickest possible response time. The host computer can provide and accept data from the conversion equipment at a rate limited only by the speed of conversion. The limitation in input-output (I/O) rate is limited only by the speed of the conversion equipment itself.

The disadvantage in having the conversion equipment at the computer is the cabling requirement. Cable runs of fifty to one hundred feet are not uncommon, and the mass of wire presents a significant problem. In addition, many of the wires carry analog signals which are more susceptible to noise than digital signals. This dictates shielding of signal lines and a more elaborate grounding system than is required for digital signal transmission.

Placing the conversion equipment at the place where the signal is used rather than at the digital computer avoids the cabling problems described. Such a system is illustrated in figure 2. Conversion equipment is located at the student station (cockpit), at the instructor station, and at the display equipment. Data are transmitted between the computer and the remote sites via a serial data transmission line consisting of only a few signal lines. The result is a great saving in cabling cost and complexity.

The disadvantage in the serial approach is the time delay that results from serial transmission. The digital computer system can provide data one to two orders of magnitude faster than data can be transmitted over a serial line. Delay in the transmission of data adds directly to computational delays, resulting in an overall lag between student input to a simulator and the correct simulator response.

Advances in digital computer technology have resulted in the means to retain the saving in cabling provided by serial data transmission without accepting the inherent delay imposed by this mode of transmission. The key is use of intelligent processing in the interface to provide the same information transfer without the high data rate. Projected technology advances further favor this approach. The drastic reduction in the cost of digital data processing hardware that has occurred in the past decade shows no signs of

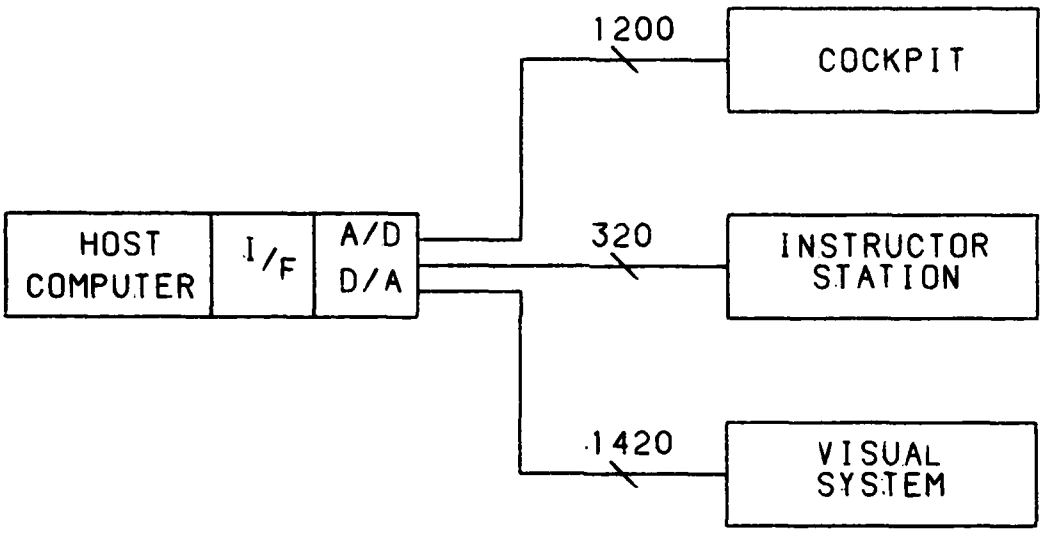


Figure 1. Block Diagram of Parallel Linkage System.

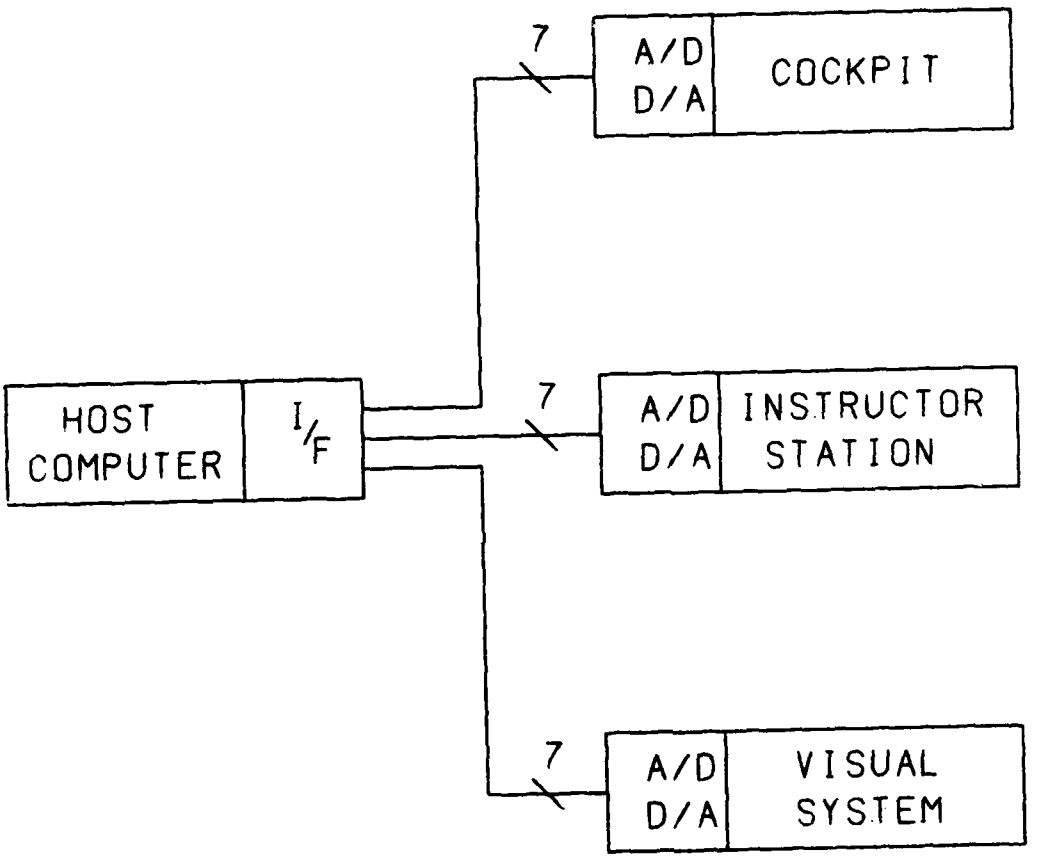


Figure 2. Block Diagram of Serial Linkage System.

slowing. On the other hand, the cost of cables and installation gives no indication of an equivalent cost reduction.

SYSTEM ORGANIZATION

The system organization proposed consists of a microcomputer based processor at the interface to the host computer and a microcomputer based processor at distribution nodes where conversion is performed. A block diagram of this organization is shown in figure 3. Transfer of data from the host computer to the master controller is accomplished by a high-speed parallel data transfer. The master controller processes these data and sends information to the slave processors via a serial transmission line. At the same time, the slave processors transmit data to the master controller. When the data transfer is complete, the master processor transmits the data it has received to the host computer.

The unique feature of this system is its ability to use extrapolation of data to reduce transmission bandwidth. The conventional method of treating data in simulators is to transmit data at a fixed update rate determined by the maximum bandwith of the data. For example, it is common to transmit information concerning aircraft dynamics 30 times a second. On the other hand, engine data are often transmitted only 5 times a second.

The proposed system does not transmit data at a fixed rate. Data are transmitted between the master processor and a slave processor only when the processor requiring the data cannot extrapolate the data within accuracy bounds without outside intervention. A slowly varying signal is transmitted at a low rate. When the same signal begins to change rapidly, the information is transmitted at a high rate.

Reduction in data rate based upon statistics does present the danger of the worst-case condition when all signals are varying at their maximum rates. A prime feature of this system is that it degrades gracefully and in the worst case produces the same result now produced by a conventional serial linkage system. And the worst-case degradation occurs only when all signals are varying rapidly, an event during which the student pilot is less likely to be affected by the delay.

SYSTEM REQUIREMENTS

The primary design requirement is to configure a linkage system that incorporates the use of prediction to reduce bandwidth into a viable general-purpose linkage sytem applicable to a variety of simulators. This dictates a need for modular design, standard bus interfaces, and software flexibility. The application demands a fail-soft capability in the event of a heavy data load and automatic self-verification.

MODULAR DESIGN. The system must be designed so that it can be expanded incrementally to meet the input-output requirements of different classes of trainers. This leads to the selection of distributed processing for the master and slave processors. These processors consist of one or more microcomputers performing the required functions, allowing the processing capability to be tailored to meet the needs of different systems.

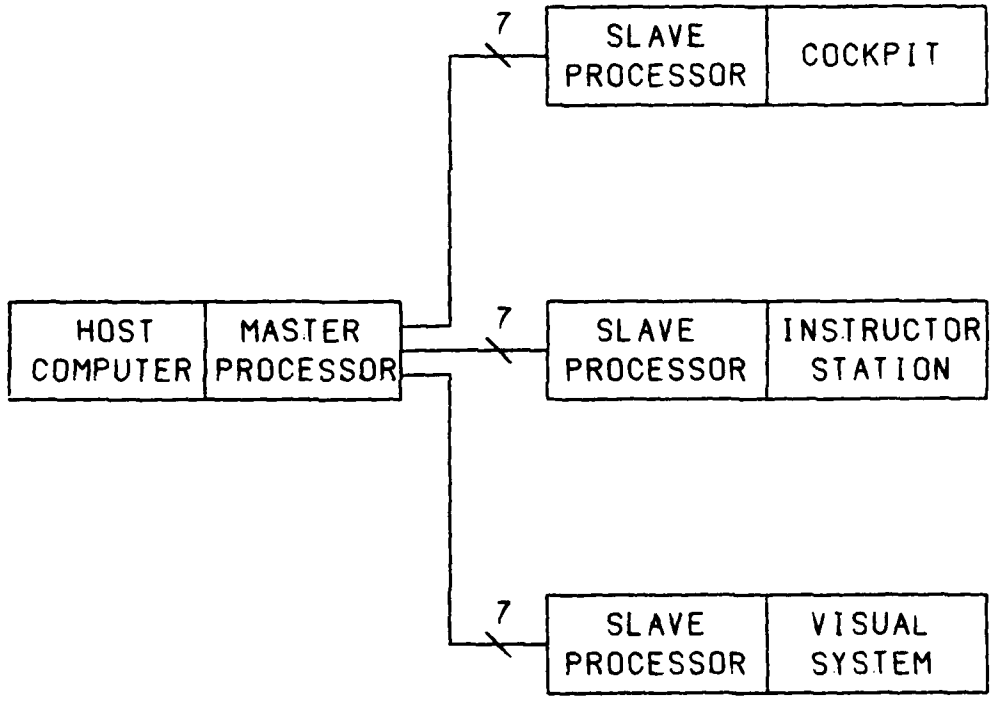


Figure 3. Block Diagram of Extrapolation Linkage System.

BUS INTERFACES. It is desirable that the linkage system use a standard bus interface such as MIL-STD-1553B. However, the breadboard system being designed to prove the concept must interface with an existing simulator. Therefore, the design presented in this report will match the existing interface, with consideration of a standard interface deferred until the concept is proven.

SOFTWARE FLEXIBILITY. The computer software to perform the linkage operations in the master processor and the slave processors must provide several different degrees of flexibility. First, in order to be generally useful the software must allow for the variation in simulators and their application. Secondly, the software must be able to accomodate addition, deletion or change in signal assignments during the operational life of a simulator. Finally, the initial application to a breadboard system to prove a concept dictates that the software be able to perform tests and generate statistics on the operation of the system during the evaluation phase.

FAIL-SOFT CAPABILITY. Although the system uses a statistical approach to give improved performance for most cases, the system must be designed to handle the full data transmission load within the simulator frame time.

SELF-VERIFICATION. The system must have self-test features to verify its operation and to aid in trouble detection and correction. The verification capability will include an on-line system which monitors system operation during the dead time in each frame when no data are being transmitted and an off-line system when allows a complete error detection and diagnostic program to be run when the linkage system is in test mode.

A PREVIEW

This preview serves to put the development of the architecture for the linkage system in perspective. The first step was the choice of extrapolation of signals to reduce bandwidth and the use of a distributed microprocessor configuration for implementation of the master and slave processors. Section II provides an analysis of the extrapolation methods available and provides a rational basis for the selection of a particular method.

Section III considers the overall system architecture. It develops guidelines for hardware implementation and techniques for basic functions such as interfacing with the host computer, signal prediction, and data transmission. Section IV describes the software requirements for the prototype linkage and the support software required to assist in interfacing the linkage system to the host computer.

Section V completes the design by describing the overall system operation and some additional features that may be added to future linkage systems. Section VI provides a summary of the linkage system characteristics and suggests areas for further work.

SECTION II

EXTRAPOLATION METHODS

Extrapolation is the key to the operation of the proposed linkage system. The degree to which extrapolation of signals is successful sets the bound on the effectiveness of the linkage system. In addition, extrapolation of signals is the major computation performed by the master and slave processors and therefore significantly affects hardware cost.

The extrapolation techniques considered are various orders of Lagrange interpolation applied to prediction of the next value in a sequence. Lagrange interpolation was chosen over more common interpolation methods such as splines or Hermite interpolation, because the other methods require knowledge of the derivative of the function. The comparison of extrapolation techniques is performed by applying the methods to sample data taken from a flight simulator. The evaluation criteria used are (1) the reduction in the number of data points that must be transmitted to keep the estimated value of the function within the bounds set for the problem, and (2) the complexity of the hardware required to implement the prediction.

EXTRAPOLATION FORMULA

The prediction method used for extrapolation is based upon (1) equally-spaced values of the independent variable and (2) use of only the last N values of the function to determine the best estimate of the next value. The method is to fit a polynomial of degree N-1 through the N values and use that polynomial to compute the estimate.

The general polynomial of degree N-1 can be expressed

$$P_N(x) = a_0 + a_1 x + a_2 x^2 + \ldots\ldots + a_N x^N = \sum_{k=0}^{N} a_k x^k$$

The condition that the polynomial pass through a set of sample points

$$(x_i, y_i)$$

produces N+1 equations of the form

$$y_i = P_N(x_i) = \sum_{k=0}^{N} a_k x^k$$

for i = 0, 1, 2, 3,....., N.

This set of N+1 linear equations can be solved for the coefficients

$$a_0, a_1, \ldots\ldots, a_N$$

and the result can be used to determine values of the polynomial at any point.

An alternate formulation of the expression for the polynomial due to Lagrange is

$$P_N(x) \approx \sum_{i=0}^{N+1} y_i \frac{\pi_i(x)}{\pi_i(x_i)}$$

where

$$\pi_i(x) = (x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_N)$$

For the case considered here, the sample points are successive values of the dependent variable

$$y_0, y_1, \ldots . y_N .$$

Because of the equal spacing of the sample points, the corresponding values of the independent variable

$$x_0, x_1, \ldots . x_N$$

can be replaced with integer values with no loss in generality. (This is done by translation and scaling of the x-axis.) Thus, the corresponding x values are made 0, 1,.....,N.

With the substitution of integer values for x in the polynomial expression, the equation above becomes

$$P_N(x) = \sum_{i=0}^{N} y_i \left[\frac{x(x-1)\ldots(x-i-1)(x-i+1)\ldots(x-N)}{i(i-1)\ldots(2)(1)(-1)(-2)\ldots(i-N)} \right]$$

Extrapolation is to be used to predict the points in the sequence. Substituting N+k for x yields

$$y_{N+k} = \sum_{i=0}^{N} y_i \left[\frac{(N+k)(N+k-1)\ldots(N+k-i-1)(N+k-i+1)\ldots(k)}{i(i-1)\ldots(2)(1)(-1)(-2)\ldots(i-N)} \right]$$

This expression can be rewritten in terms of factorials to give

$$y_{N+k} = \sum_{i=0}^{N} y_i \left[(-1)^{i+1} \frac{(N+k)!(N+k-i-1)!}{i!(N-i)!(k-1)!(N+k-i)!} \right]$$

which can be expressed in terms of binomial coefficients

$$y_{N+k} = \sum_{i=0}^{N} y_i \left[(-1)^{i+1} \binom{N+k}{i} \binom{N+k-i-1}{k-1} \right].$$

For prediction of the next sample point following the N known points, this reduces to

$$y_{N+1} = \sum_{i=0}^{N} y_i \left[(-1)^{i+1} \binom{N+1}{i} \right].$$

This is the extrapolation formula used in the following analysis to determine the most suitable choice of N for use in the linkage system.

COMPUTATIONAL COMPLEXITY

A major consideration in the choice of an extrapolation scheme is its cost in computation time and storage. The cost in storage is the easiest to estimate, because the storage requirement is simply the number of computer words required to save the N past values required for any order of polynomial fit that might be selected. Therefore, the storage required for each prediction algorithm is N.

The computation time required is not as simple to assess. For this comparison, the computation time will be measured by the number of instructions executed, without regard to instruction type. In making this comparison, instructions like multiply which take a long time are replaced by add and shift operations.

The estimate of instructions required is made in two steps (1) instructions to update the storage area holding the N points to be considered and (2) instructions required to compute the value of the next point. Begin by considering the update of storage. After a computation, the N data values must be moved one position in storage and the oldest value discarded. This requires N load operations to store each data element in its correct place, but only N-1 load operations because the most recent is in a register rather than in storage. The data shifting operation takes 2N-1 instructions.

The number of instructions required to calculate the next point must be determined by implementing the equation given above. The form of the equation for the next point given above would require summation of N products. For solution on a microcomputer, the solution of the sum of products is best reduced to add and shift operations. Each case must be handled differently, because the shift and add implementation depends on the coefficients that must be used at each step. Table 1 indicates the computational steps required for values of N from 1 to 5.

Figure 4 shows the total computation steps required, the sum of the steps indicated in Table 1 and the 2N-1 steps to update storage. These data illustrate the cost growth associated with the use of high-order polynomials in prediction. A high order prediction algorithm can be justified only if it can be shown to be very effective in reducing the number of data points transmitted.

PREDICTION EXAMPLES

The effectiveness of the different orders of prediction algorithm will be examined by looking at two kinds of examples. The first set of examples use sinusoids of different frequencies and amplitudes, and the second set uses the simulator-derived data described earlier.

SINUSOIDAL EXAMPLES. Sinusoidal examples were chosen to illustrate the frequency dependence of extrapolation. Application of the prediction methods to sinusoids of various frequencies shows how the prediction improves as the frequency of the input signal decreases.

The first illustration shows the effect of using a zero order polynomial (sample and hold) to predict successive values of a one hertz sinusoid. Figure 5 shows the input waveform and indicates at the top those points at which the prediction accuracy was sufficient to meet a 12-bit accuracy criterion. Figure 6 shows the same information for a 10-bit accuracy criterion.

Figures 7 and 8 show sampling of the first part of a 0.1 hertz sine wave and the positions at which samples are required for the 12-bit and 10-bit error criteria. The same data were obtained for sine waves from one hertz down to 0.001 hertz and for different orders of polynomial fit. The results are shown in Table 2 and in Figure 9.

In addition to the illustration of the frequency dependence of the prediction, the examples show that higher-order prediction is not necessarily better. This is a typical result of polynomial prediction, where the higher

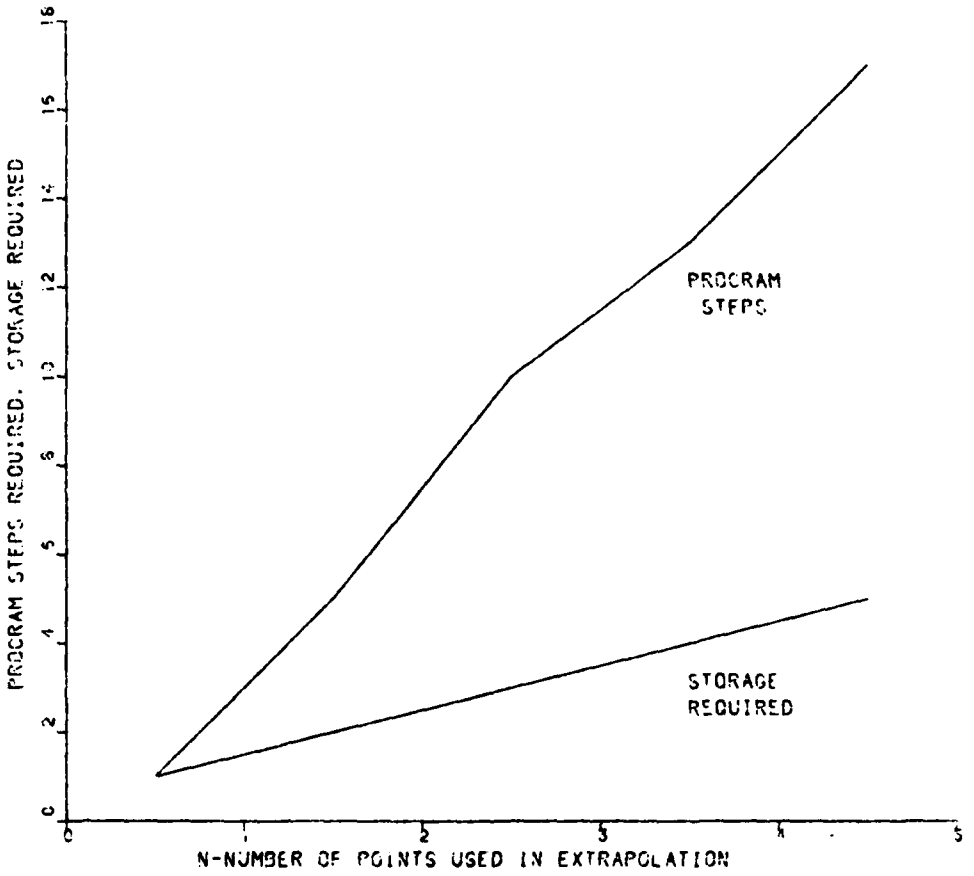


Figure 4. Computation and Storage Requirements.

TABLE 1. COMPUTATION STEPS REQUIRED FOR EXTRAPOLATION

COMPUTATION STEP	FUNCTION
N=1	
LOAD y(0)	y(0)
N=2	
LOAD y(1)	y(1)
ADD y(1)	2y(1)
SUB y(0)	2y(1)-y(0)
N=3	
LOAD y(2)	y(2)
SUB y(1)	y(2)-y(1)
LOAD [y(2)-y(1)]	y(2)-y(1)
SHIFT LEFT 1	2[y(2)-y(1)]
ADD [y(2)-y(1)]	3[y(2)-y(1)]
ADD y(0)	3y(2)-3y(1)+y(0)
N=4	
LOAD y(3)	y(3)
ADD y(1)	y(3)+y(1)
SUB y(2)	y(3)-y(2)+y(1)
SHIFT LEFT 1	2[y(3)-y(2)+y(1)]
SUB y(2)	2y(3)-3y(2)+2y(1)
SHIFT LEFT 1	2[2y(3)-3y(2)+2y(1)]
SUB y(0)	4y(3)-6y(2)+4y(1)-y(0)
N=5	
LOAD y(2)	y(2)
SUB y(3)	y(2)-y(3)
SHIFT LEFT 1	2[y(2)-y(3)]
ADD y(4)	y(4)-2y(3)+2y(2)
SUB y(1)	y(4)-2y(3)+2y(2)-y(1)
LOAD [y(4)-2y(3)+2y(2)-y(1)]	y(4)-2y(3)+2y(2)-y(1)
SHIFT LEFT 2	4[y(4)-2y(3)+2y(2)-y(1)]
ADD [y(4)-2y(3)+2y(2)-y(1)]	5[y(4)-2y(3)+2y(2)-y(1)]
ADD y(0)	5y(4)-10y(3)+10y(2)-5y(1)+y(0)

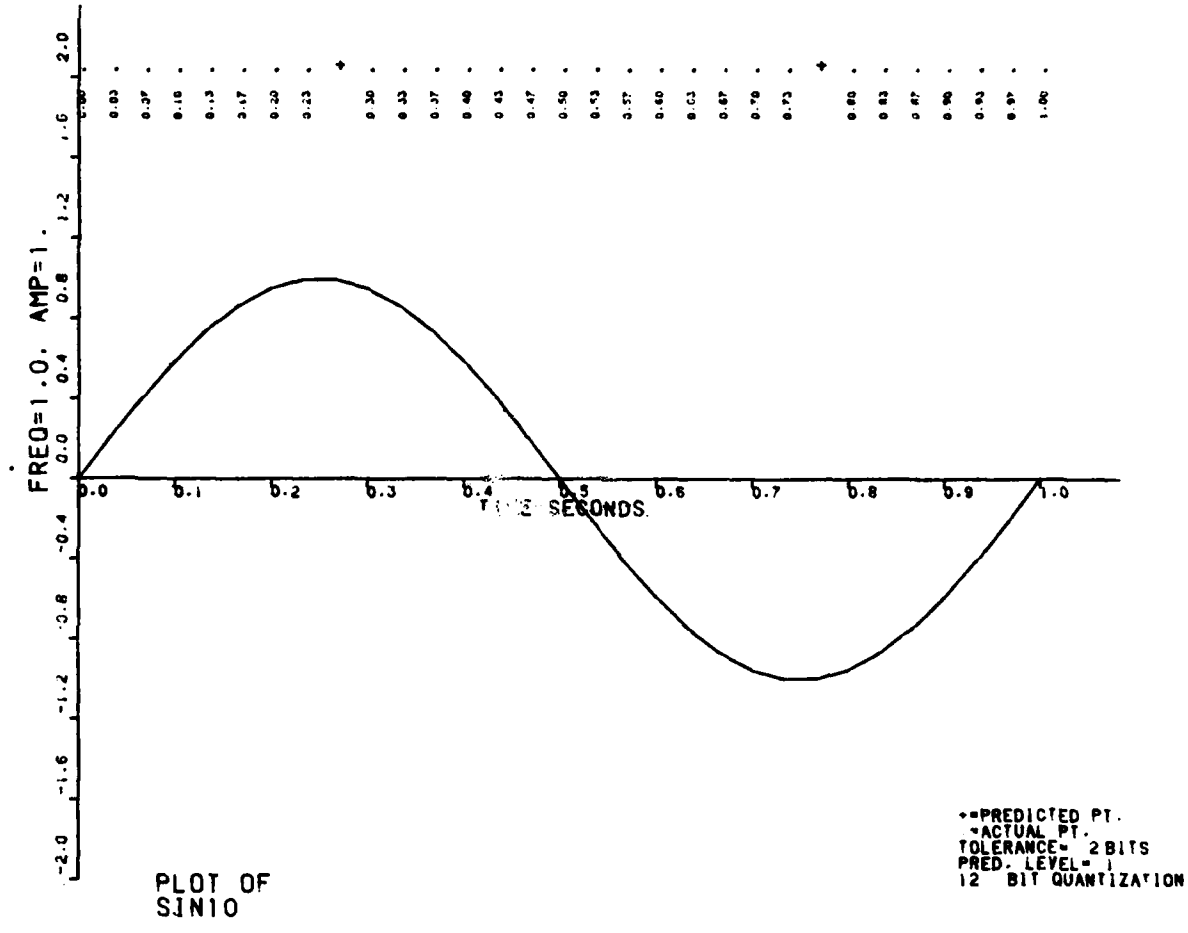


Figure 5. Extrapolation of a 1 Hz. Sinusoid to 12-bit Accuracy.

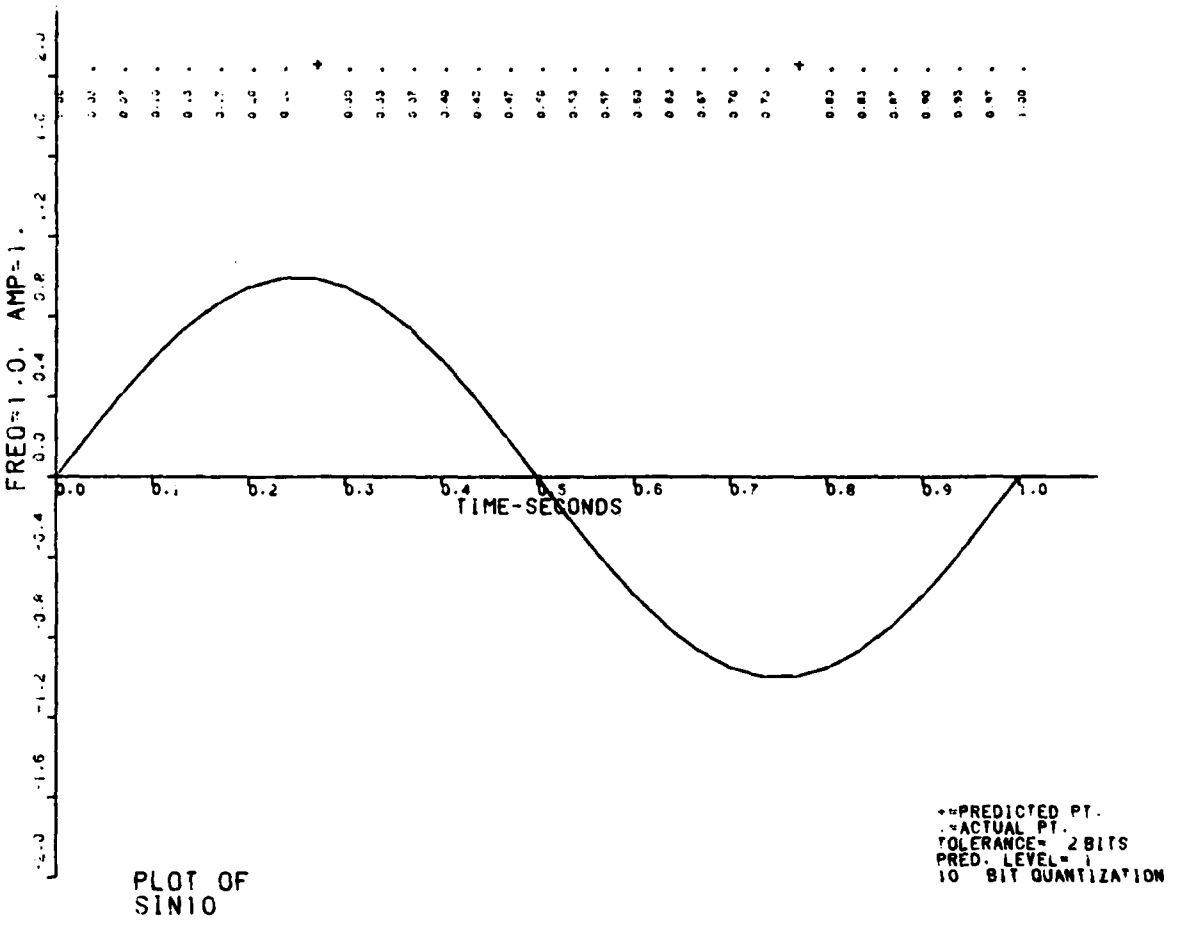


Figure 6. Extrapolation of a 1 Hz. Sinusoid to 10-bit Accuracy.

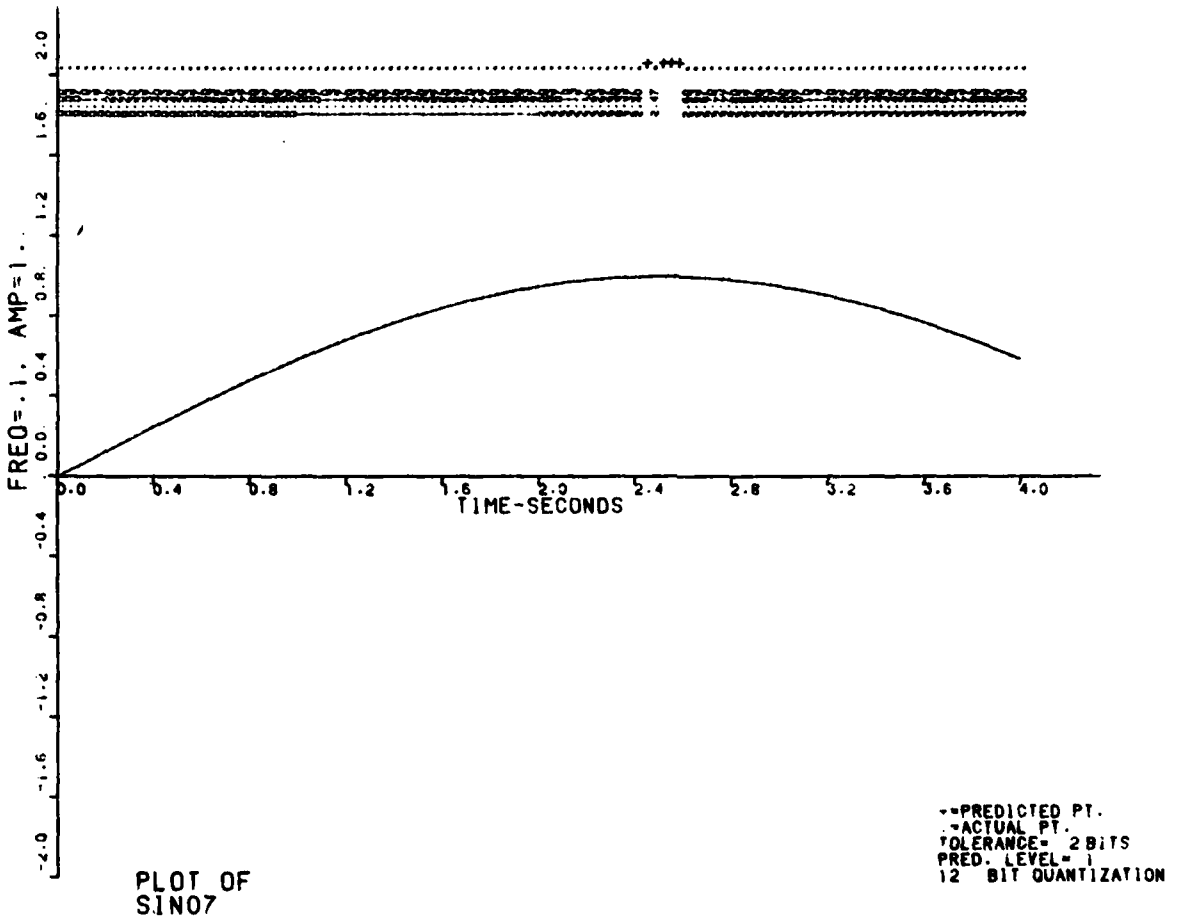


Figure 7. Extrapolation of a 0.1 Hz. Sinusoid to 12-bit Accuracy.

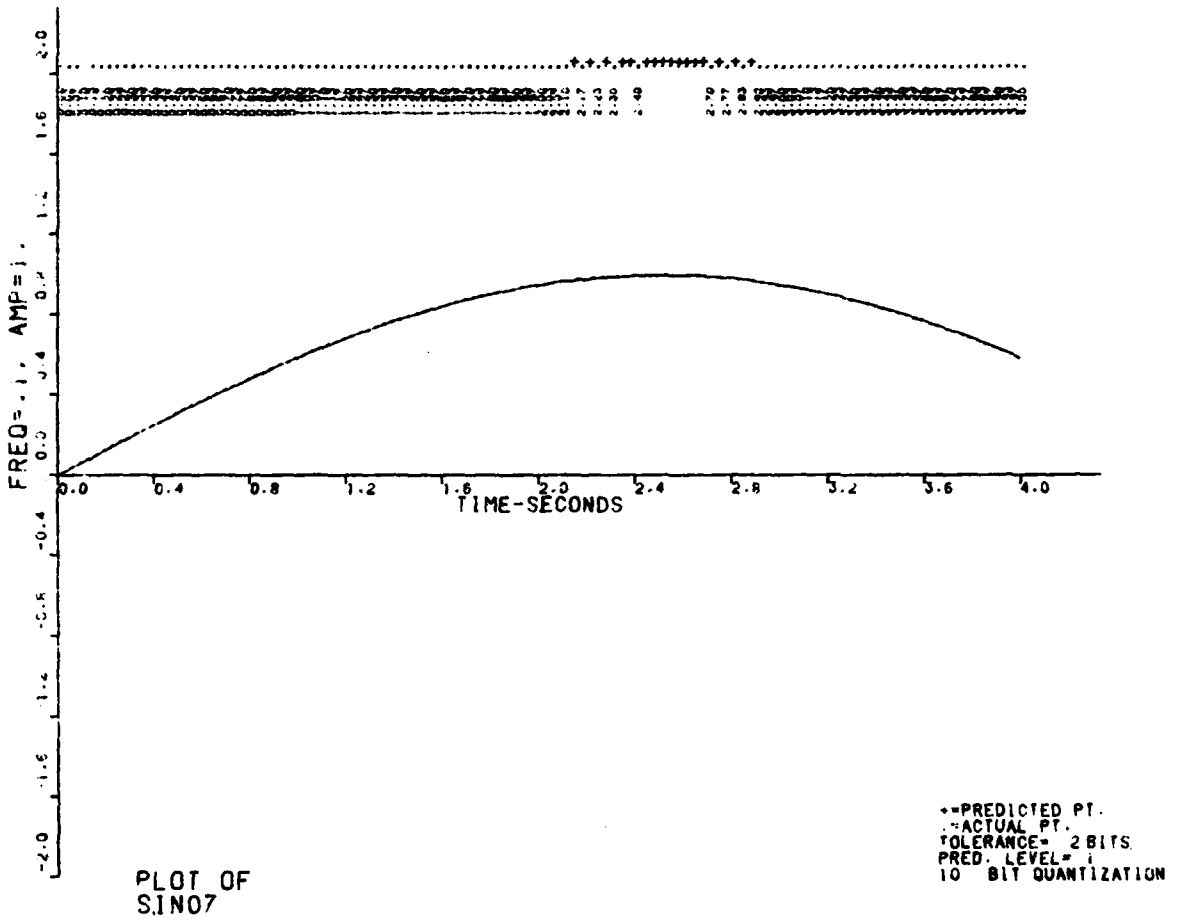


Figure 8. Extrapolation of a 0.1 Hz. Sinusoid to 10-bit Accuracy.

TABLE 2. EXTRAPOLATION EFFECTIVENESS AT DIFFERENT FREQUENCIES

FREQUENCY (HERTZ)	EFFECTIVENESS (RATIO)							
	10-BIT QUANTIZATION				12-BIT QUANTIZATION			
	N=1	N=2	N=3	N=4	N=1	N=2	N=3	N=4
.0001	.995	.991	.986	.982	.982	.964	.945	.927
.0002	.991	.982	.973	.964	.964	.927	.891	.854
.0005	.977	.955	.932	.909	.909	.818	.681	.389
.001	.955	.909	.864	.818	.818	.825	.684	.452
.002	.909	.818	.678	.392	.661	.806	.644	.324
.005	.773	.873	.769	.605	.276	.843	.722	.511
.01	.618	.862	.752	.575	.131	.811	.643	.427
.02	.275	.838	.704	.520	.065	.834	.680	.520
.05	.108	.762	.555	.404	.027	.620	.415	.221
.1	.050	.639	.411	.175	.017	.401	.374	.209
.2	.046	.395	.388	.243	.020	.099	.388	.217
.5	.016	.081	.306	.113	.016	.016	.129	.274
1.0	.094	.000	.000	.156	.094	.000	.000	.000

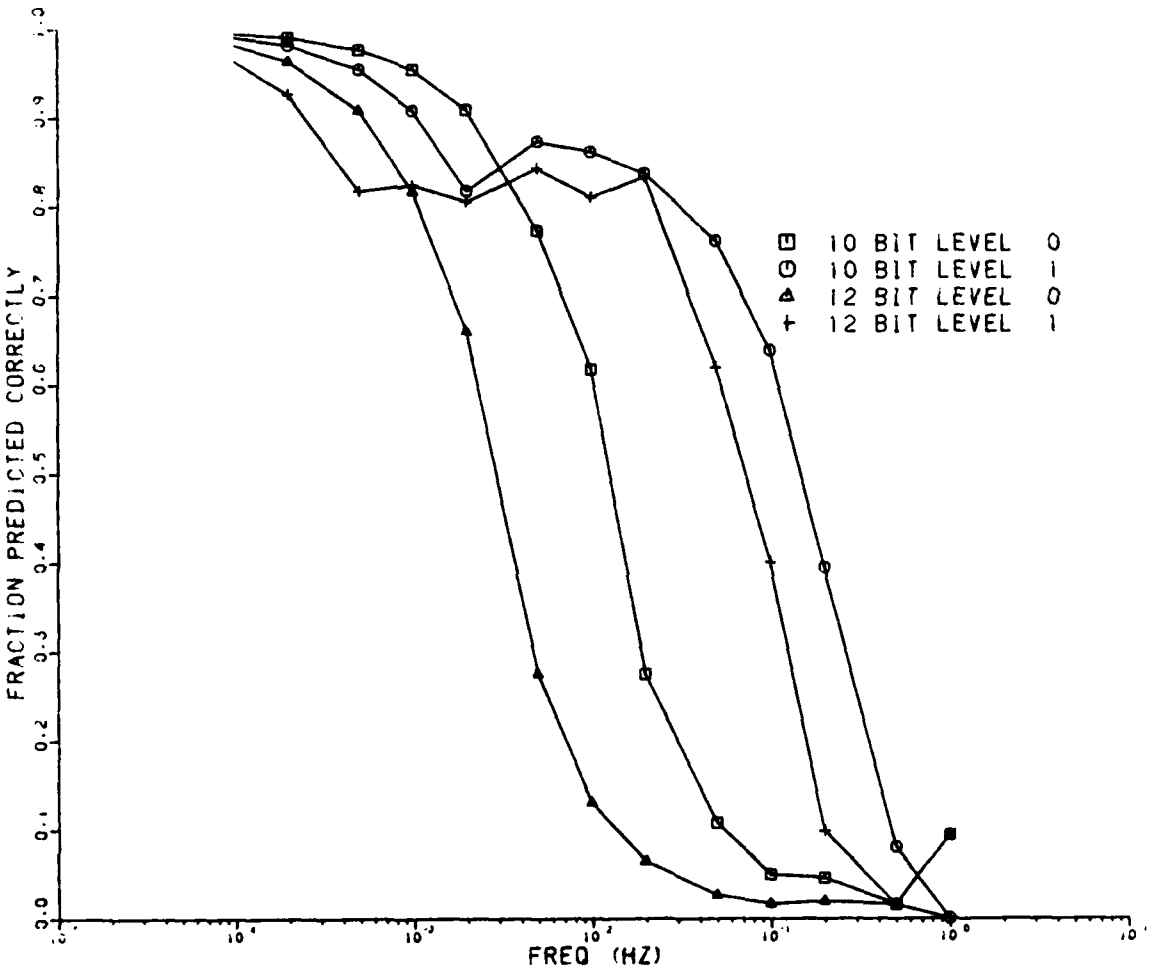


Figure 9. Summary of Extrapolation Results for Sinusoids.

order polynomials produce oscillations that actually give a poorer result than lower order polynomials. Another contributor to the problem with high order polynomials in prediction is the roundoff error. All polynomial fits of higher order than one will give a perfect fit to a straight line, but in practice the higher order polynomials do not perform well because of roundoff error in the computation.

SIMULATOR-DERIVED-DATA EXAMPLES. A second set of examples uses data obtained from the results of a motion test experiment performed on the Visual Training Research System. The flight conditions and variables recorded are shown in Tables 3 and 4. Although the data cannot be said to be typical, they serve to illustrate the effect of the various prediction methods and the saving that can be obtained by using prediction. Figures 10 and 11 show examples of the data used to evaluate the extrapolation methods.

Figures 12 through 19 show the number of variables that must be transmitted at each computation cycle for each of the nine experiments. This is an illustration of the results that can be achieved by using the statistical approach instead of blindly transmitting a block of data at each frame.

CHOICE OF METHOD

Clearly, use of zero order prediction is indicated by the data examined. Both the computational complexity and the effectiveness of the extrapolation methods indicate that a simple sample and hold algorithm is the best method for use in the linkage system. There are undoubtedly pathological cases where other extrapolation schemes do better, but the simplicity of sample and hold extrapolation and its proven effectiveness for the sample functions considered lead to its selection for the prototype linkage system.

TABLE 3. FLIGHT CONDITIONS FOR DATA GATHERING

SEQUENCE NUMBER	FLIGHT CONDITIONS
1	Carrier Landing
2	30-degree S Turn
3	Left/Right Radical
4	Aileron Rolls
5	60-degree S Turn
6	Extending Speed Brake
7	Lowering Flaps
8	Cycling of Landing Gear and Flaps

TABLE 4. VARIABLES RECORDED

SEQUENCE NUMBER	VARIABLE
1	Engine Thrust - Left
2	Engine Thrust - Right
3	Angle of Attack
4	Longitudinal Axis Acceleration
5	Lateral Axis Acceleration
6	Vertical Axis Acceleration
7	Sideslip Angle
8	Cosine of Angle of Attack
9	Roll Acceleration
10	Pitch Acceleration
11	Yaw Acceleration
12	Direction Cosine L3
13	Longitudinal Center of Gravity
14	Direction Cosine M3
15	Direction Cosine N1
16	Direction Cosine N2
17	Direction Cosine N3
18	Roll Rate
19	Pitch Rate
20	Yaw Rate
21	Sine of Angle of Attack
22	Sine of Sideslip Angle
23	Gross Weight
24	True Aircraft Roll
25	True Aircraft Pitch
26	Change in True Aircraft Roll
27	Change in True Aircraft Pitch
28	Manually Induced Freeze
29	Cosine of Sideslip Angle

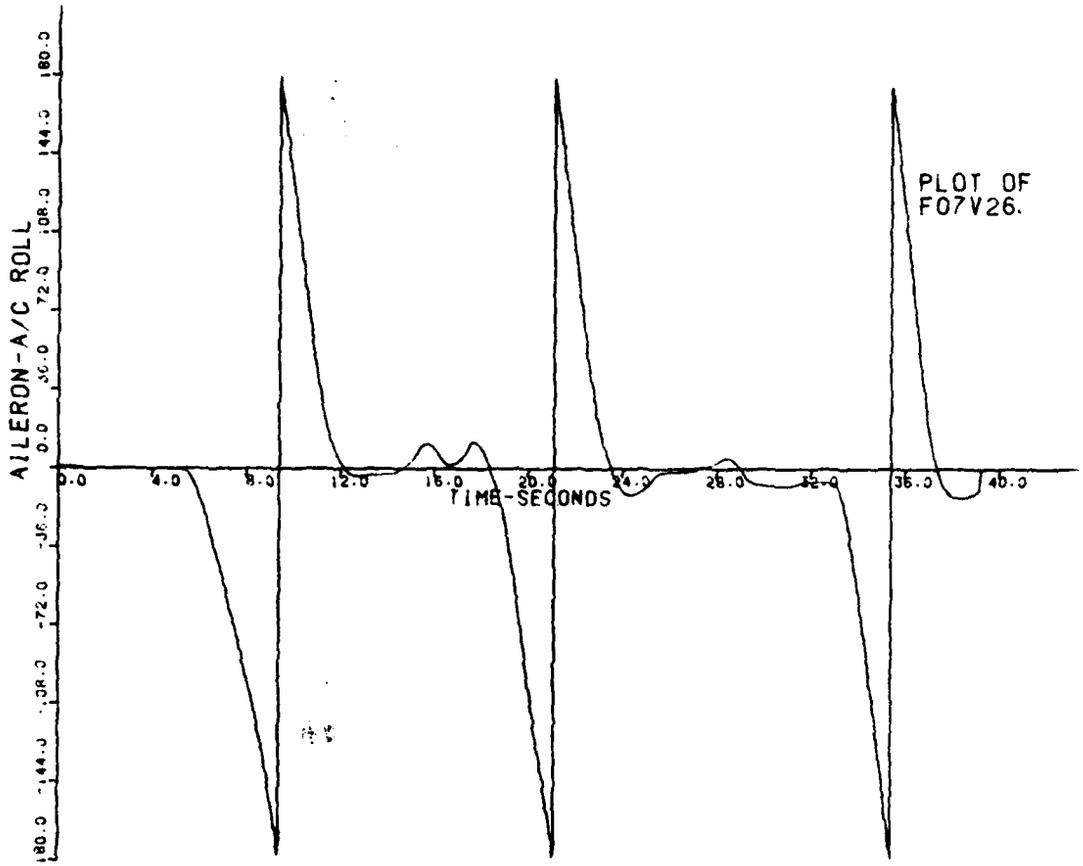


Figure 10. Roll Angle During Aileron Rolls.

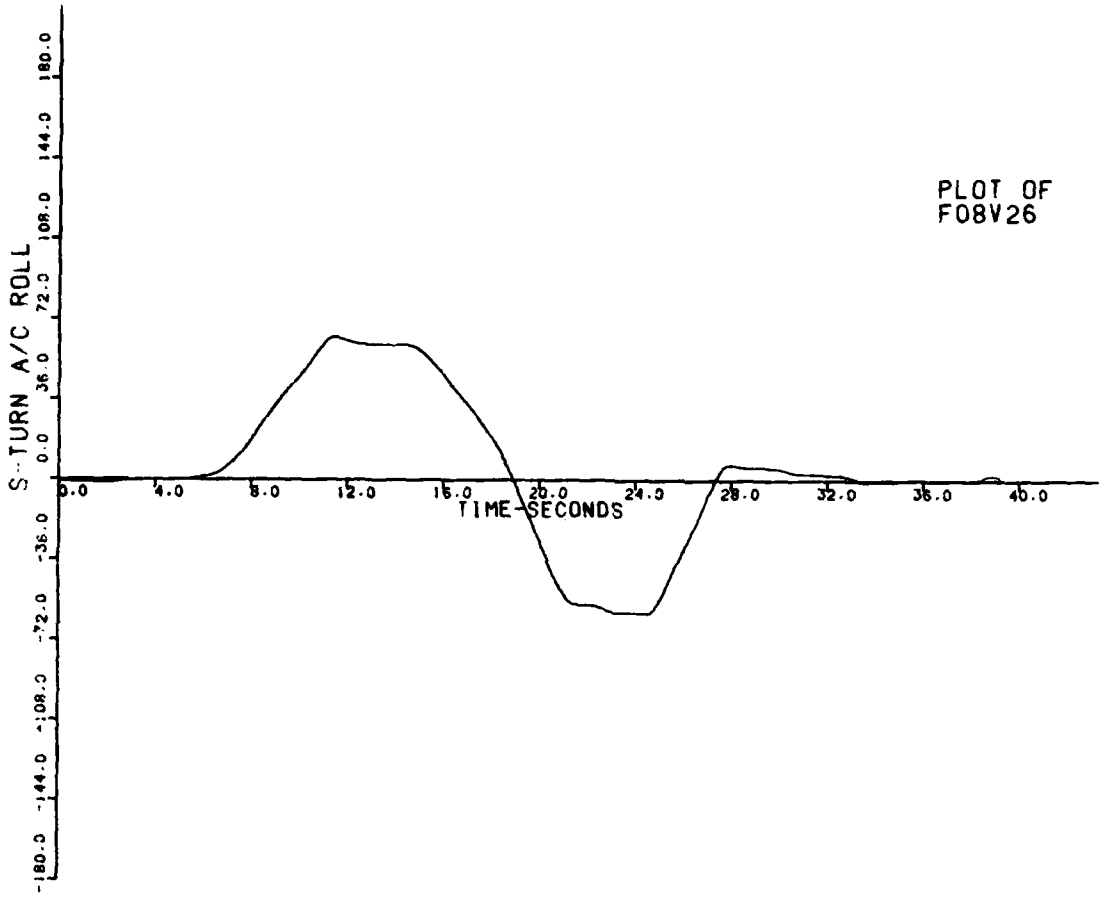


Figure 11. Roll Angle During 60-degree S Turn.

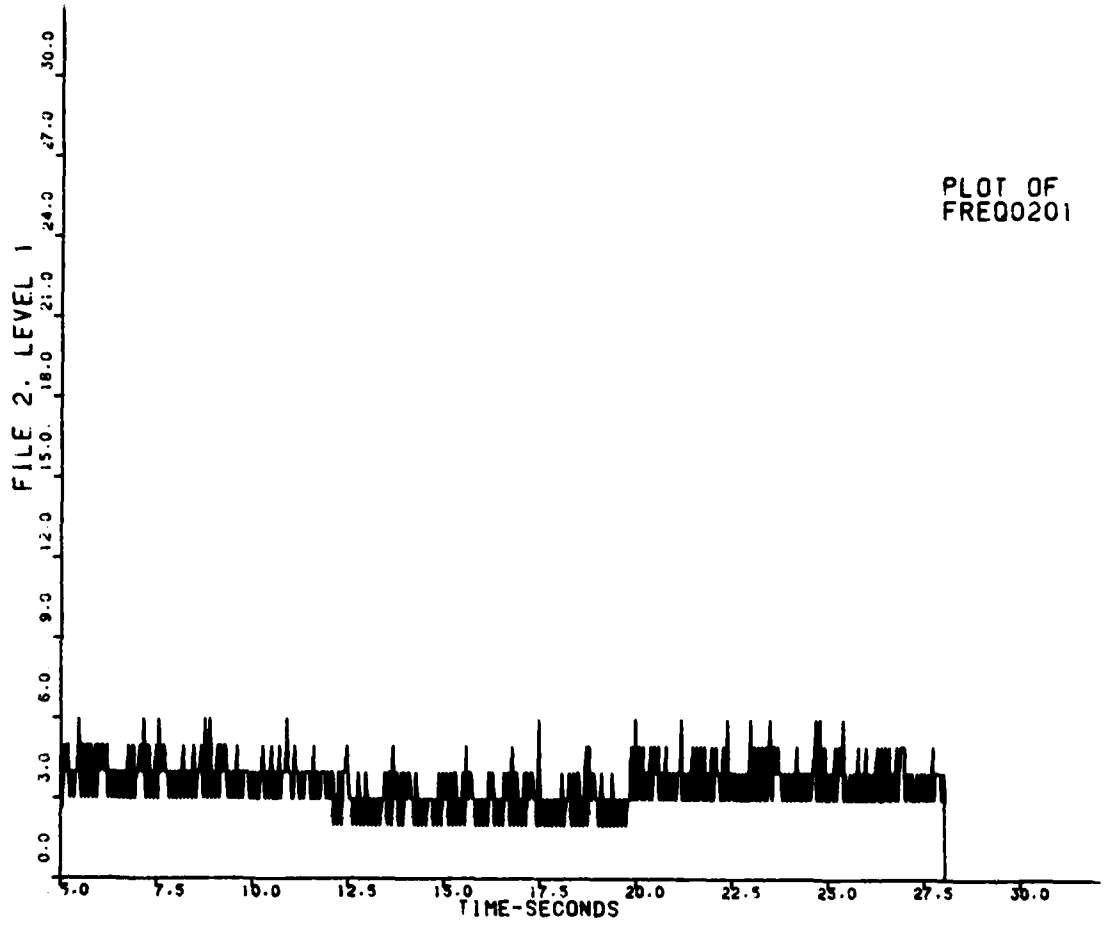


Figure 12. Data Transmitted During Carrier Landing.

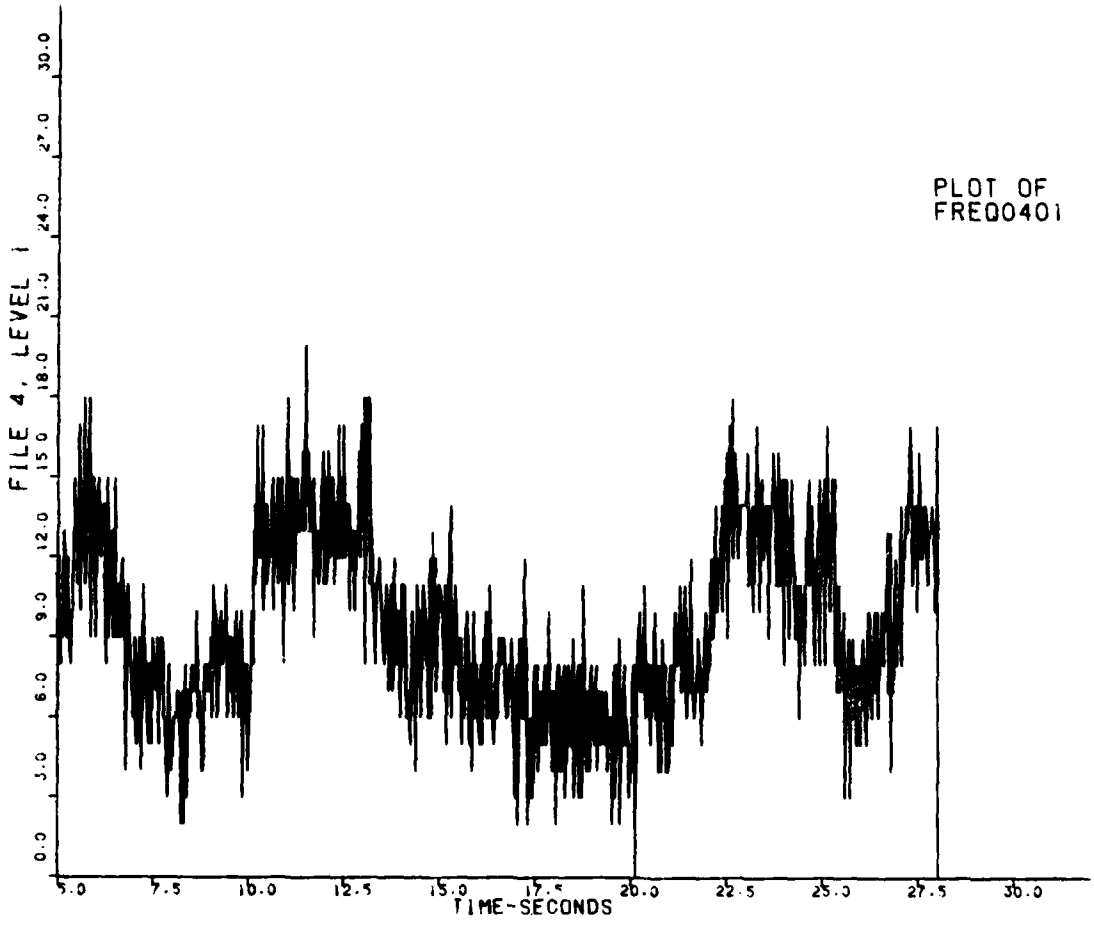


Figure 13. Data Transmitted During 30-degree S Turn.

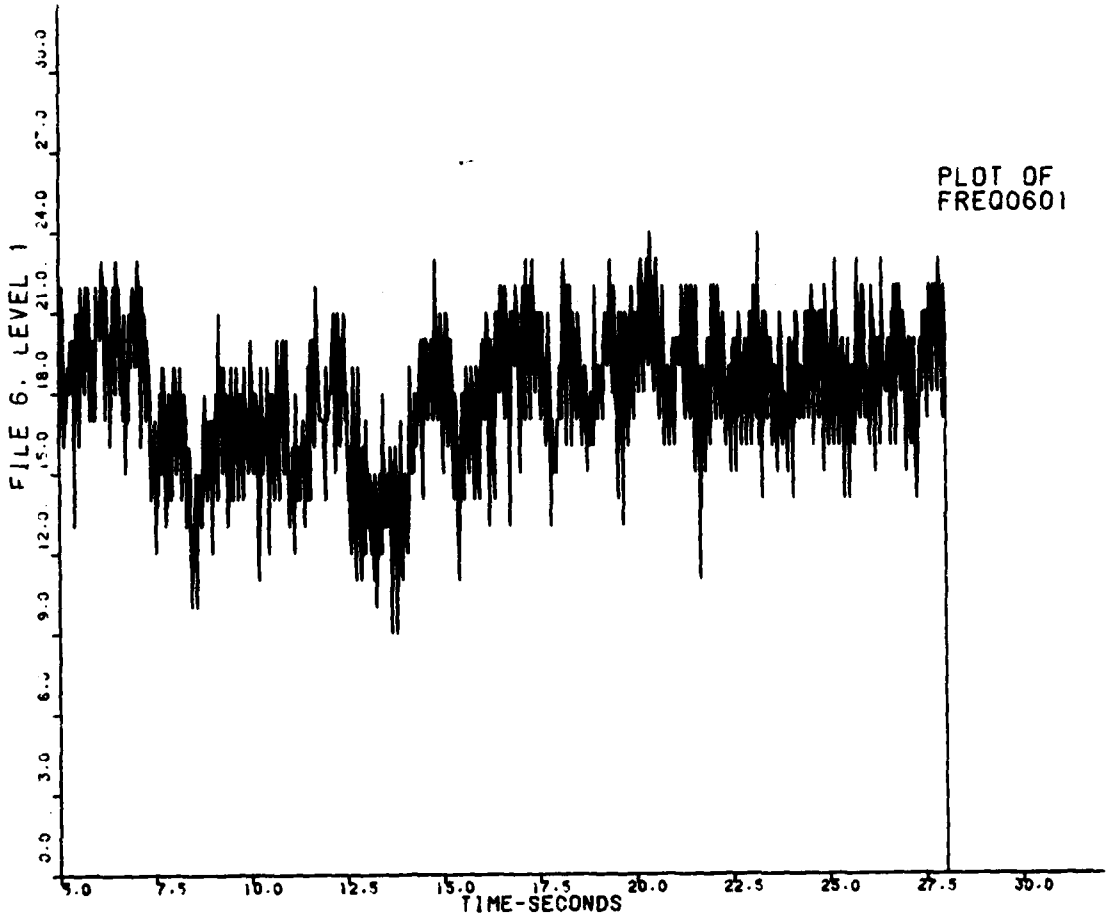


Figure 14. Data Transmitted During Left/Right Radical.

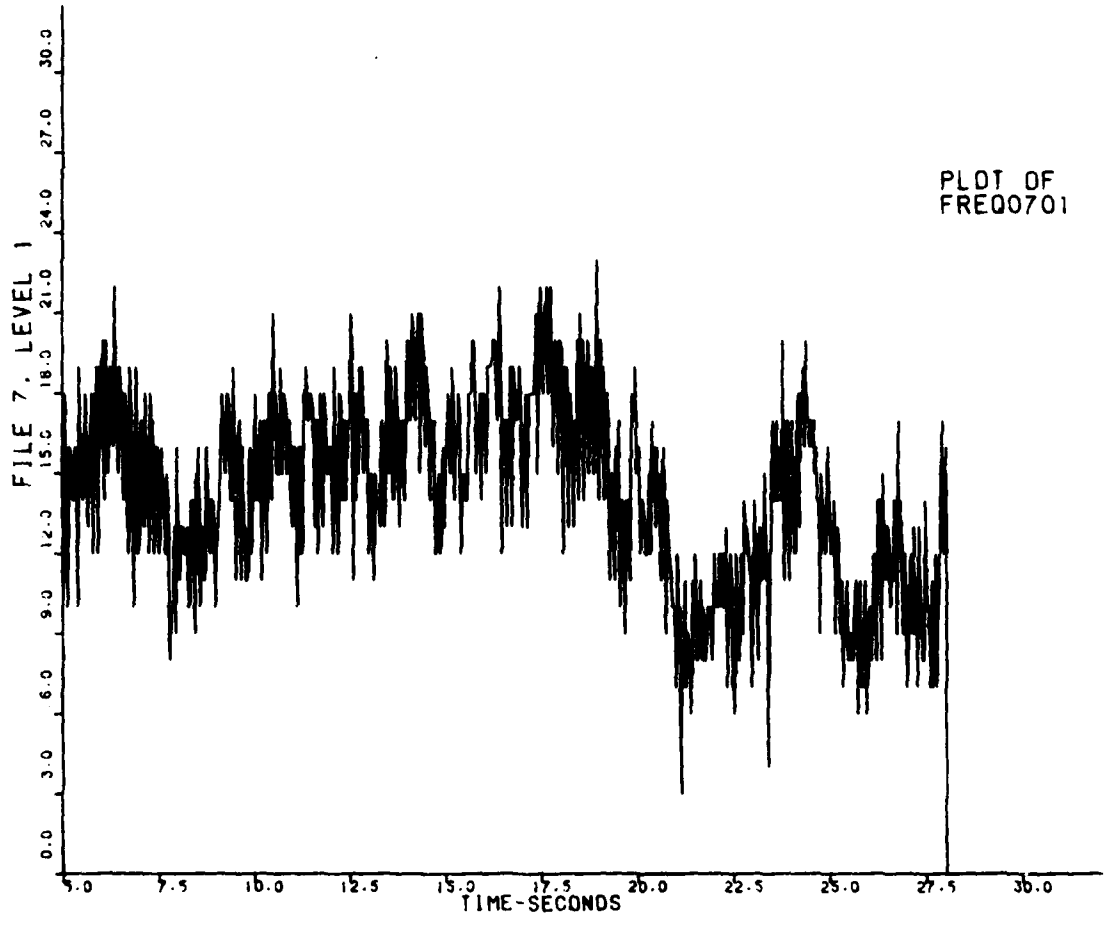


Figure 15. Data Transmitted During Aileron Rolls.

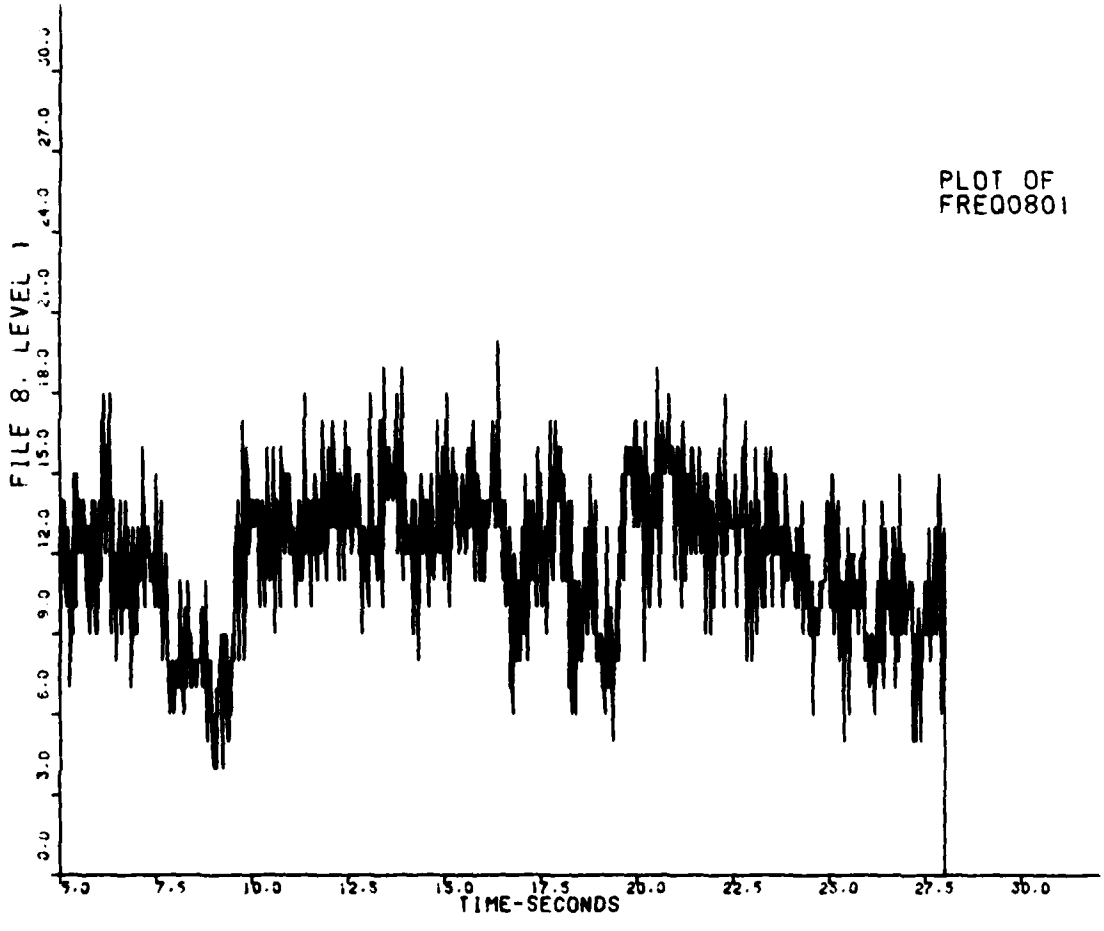


Figure 16. Data Transmitted During 60-degree S Turn.

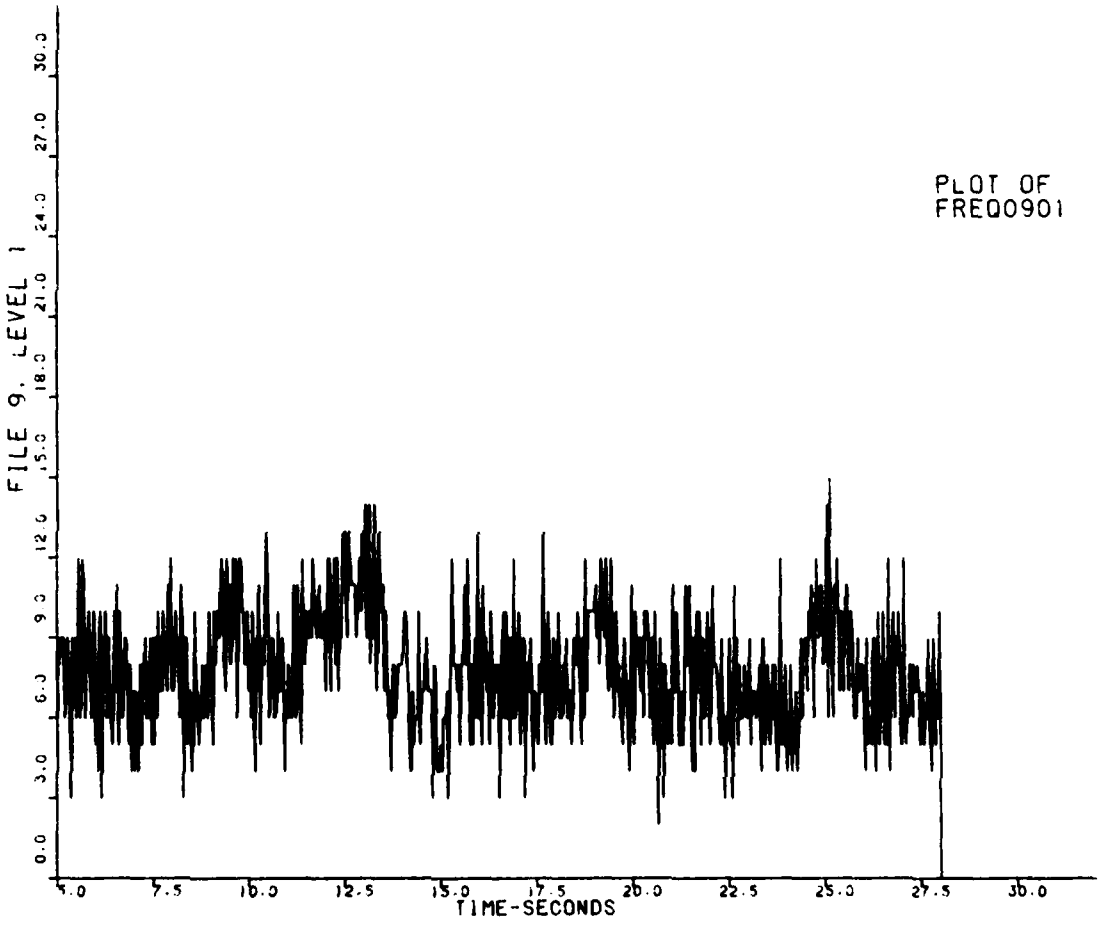


Figure 17. Data Transmitted During Extending Speed Brake.

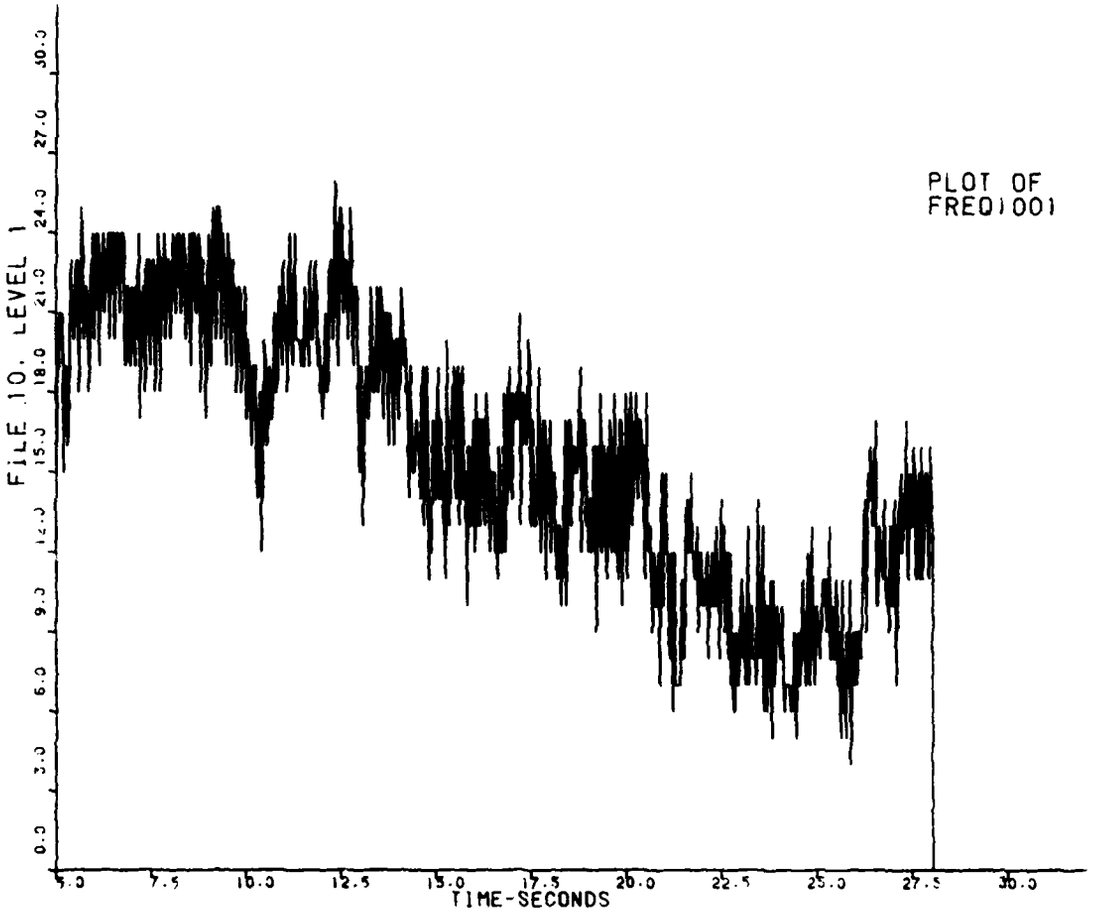


Figure 18. Data Transmitted During Lowering Flaps.

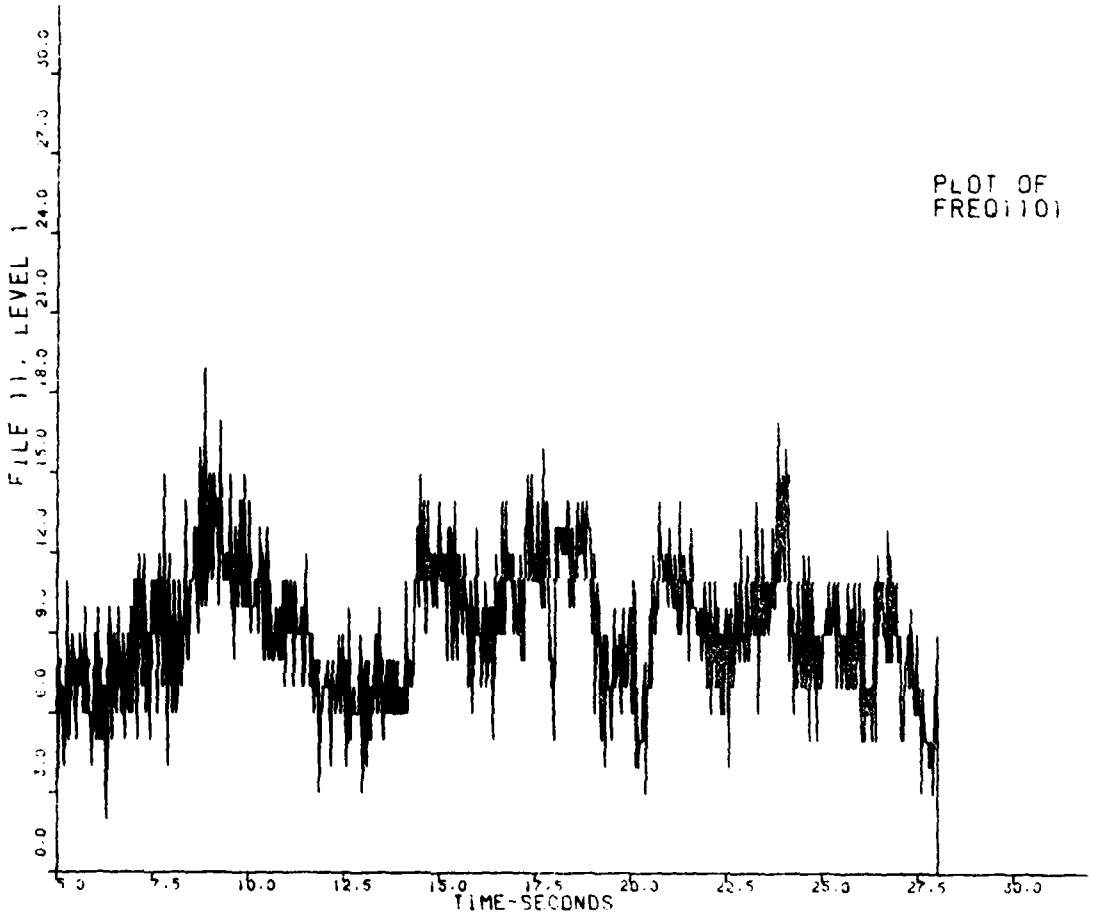


Figure 19. Data Transmitted During Cycling of Landing Gear and Flaps.

SECTION III

HARDWARE DESIGN

Design of the linkage system requires translation of the general requirements given in Section I into a specific hardware arrangement. The first step is selection of a criterion for system evaluation. The general requirements and the evaluation criterion are then used to develop the processor hardware configuration and the basic techniques for implementing the system. Major items to be considered are efficient methods for handling the common operations in data transmission: (1) mass transfer of data from the host computer to the master processor, (2) determining what data are to be sent and their routing, (3) selecting a transmission format, and (4) handling conflicts when several slave processors are ready to transmit data at the same time.

The design problem is to devise an "effective" implementation of a linkage system for use in simulators. A precise definition of "effective" is elusive, but it tends toward a goal of low cost and minimum delay introduced. The use of extrapolation reduces the delay introduced by the system at some hardware cost. The implementation is "effective" if the additional hardware cost is nominal and the systen offers a substantial reduction in delay time.

The goal can be achieved by selecting an extrapolation method that reduces the amount of data that must be transmitted, designing hardware that provides fast processing at low cost, and making proper selections in choosing which functions should be implemented in special-purpose hardware and which should be implemented by microprocessor. The extrapolation method was selected in Section II. The hardware configuration is selected in this section.

HARDWARE ORGANIZATION

The basic block diagram showing the concept of a serial linkage system instead of a parallel linkage was presented in Figure 2. A more detailed block diagram of such a system is illustrated in Figure 20. This diagram represents the linkage system now used for the Visual Technology Research Simulator (VTRS). It is instructive to examine this linkage system for two reasons: (1) it provides an example of the simulator linkage problem to be solved, and (2) the design described in this report is a system to replace the present VTRS linkage with one that exhibits less time delay.

The VTRS presents an ideal tool for demonstration of a new concept, because: (1) it provides a research tool on which to test the new design, and (2) much of the existing interface hardware can be used with the new design, making it unnecessary to provide new circuit cards to connect the linkage to the external equipment. Although the use of an existing interface does limit the mode of signal transmission, this does not interfere with the demonstration of the advantages of using extrapolation instead of transmitting all data over the linkage. The design described here will use the seven wire, one megahertz signal transmission line used in the VTRS. Later models can be expected to use a standard bus such as that described by MIL-STD-1553B.

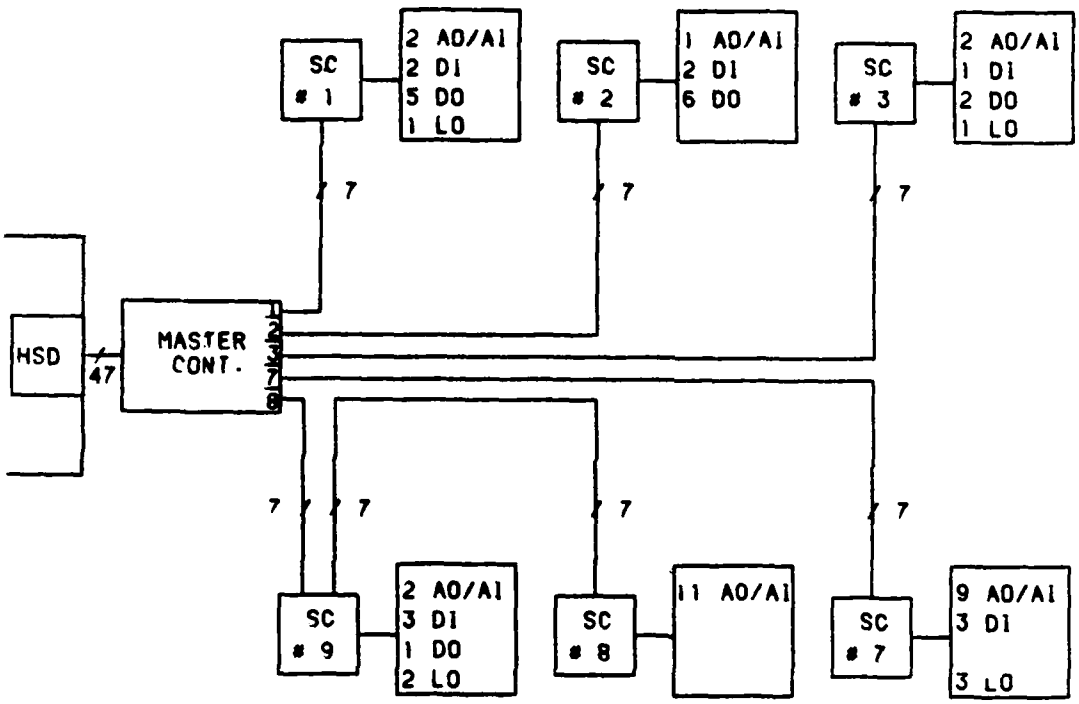


Figure 20. Block Diagram of VTRS Linkage.

VTRS LINKAGE. The existing VTRS linkage system uses a 3-level hierarchy to transmit signals between the central computer and the external devices. The functions performed at each level are described below.

Master Controller. The master controller provides the interface with the Systems Engineering Laboratories (SEL) computer via the High-Speed Data Interface (HSD). The master computer distributes data received by direct memory access (DMA) from the SEL memory to the external devices via the subcontrollers. It also collects data from the external devices and transmits these data to the SEL memory. Data transfer between the Master Controller and the Subcontrollers is in serial form.

Subcontroller. The subcontroller provides the interface between the serial data lines from the Master Controller and the system cards which actually supply the signals to the interface devices. In addition to its data transmission functions, the Subcontroller provides selection signals to the system cards and converts analog signals received from the system cards into digital data for transmission to the Master Controller.

System Card. The system card provides the connection to the individual devices. In the VTRS there are four types of system cards, each designed for a specific type of external device. Table 5 describes the capability of each type of card.

The VTRS system shows the advantage of using serial transmission. The system cards provide a capacity of 400 analog lines and 2500 discrete lines. The use of the serial linkage system achieves a considerable saving by eliminating 50 to 100 foot runs of thousands of wires.

INTERFACE DESIGN. The new linkage system will replace the master controller and subcontrollers with intelligent processors that decide what data are to be transferred and resolve conflicting requirements for access to the serial data transmission line. Figure 21 shows the block diagram for the system. The functions performed by the processors are shown in Table 6. The data transfer sequence begins with a signal from the host computer to the master processor (MP). This signal is transmitted to each slave processor (SP) via the serial data link. Steps 1-4 are performed and the data transfer for that cycle is complete. This sequence is repeated for each update cycle of the simulator.

The test and verification performed after the update sequence is done on a time available basis. In general there will be time available between the completion of the data transfers and the command from the host computer to begin the next set of data transfers. During this interval, the master and slave processors transmit data back and forth for verification. The test and verification program scans data storage and causes successive data entries to be used. Data in question are retransmitted and corrected, then reverified.

If the retransmission corrects the problem, the error is recorded in the error log for later use in troubleshooting. If the error cannot be corrected, a fault indication is sent to the host computer as part of the next update cycle. In this way, intermittent errors are corrected without interrupting the simulator operation. An error count maximum, set by the system program, is provided so that a great number of intermittent errors can cause a fault indication to be sent to the host computer.

TABLE 5. SYSTEM CARD TYPES

DESIGNATOR	DESCRIPTION
AI/AO	The analog card provides the interface to devices requiring a voltage level. The card performs the digital to analog conversion, but sends analog signals to the subcontroller for conversion. The card provides 7 analog inputs and 8 analog outputs. Signals are in the range &+ 10 volts.
DI	The digital input card provides an interface for 64 discrete signals. Signals are 0 or 5 volts.
DO	The digital output card provides an interface for 96 discrete signals. Signals are 0 to 5 volts.
LO	The light driver output card provides an interface for 64 discrete signals. The output can sink up to 250 milliamperes from an external source of up to 35 volts.

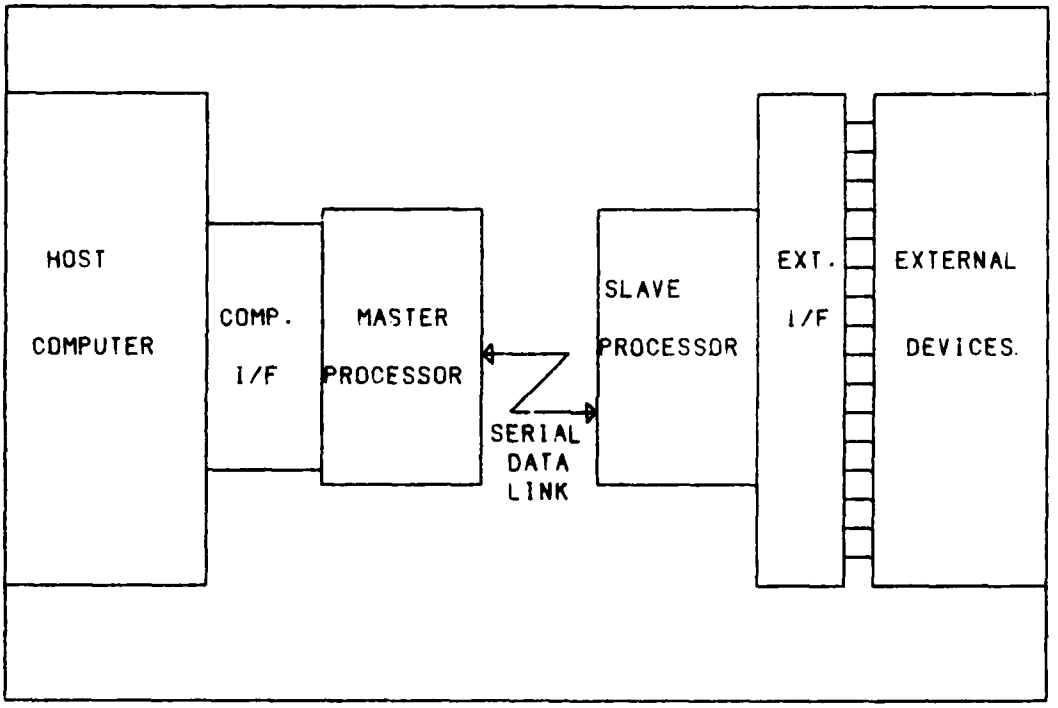


Figure 21. Block Diagram of Extrapolation Linkage System.

TABLE 6. FUNCTIONS PERFORMED BY PROCESSORS

MASTER PROCESSOR (MP)	SLAVE PROCESSOR (SP)
Data Transfer: HOST to MP	Data Transfer: Device to SP
Determine changes including prediction determine slave location	Determine changes including prediction
Data interchange changed output data to SP	Data interchange changed output to MP
Data transfer: MP to HOST	Data transfer: SP to Devices
Verification comparisons	Verification comparisons

MASTER PROCESSOR. Figure 22 shows a block diagram of the MP. The MP is connected to the SEL computer through the existing VTRS interface and to the SP's through a serial data link. Since the MP and SP are new designs, the interface between the master processor and the slave processors is not fixed by the existing system. However, use of the present 7-signal cable configuration was chosen to avoid new design in an area not directly related to proving the feasibility of the new system. Table 7 shows the signals transmitted between MP and SP.

Data transfers between the master processor and the SEL computer are performed by direct access to the SEL memory. This provides a 32-bit parallel transfer at rates up to 834,000 words per second. Transfers, once initiated by the computer, continue without further computer intervention. Timing of the transfers is governed by a handshaking protocol in which the data sending device provides an information ready signal and waits for an acknowledgement from the receiving device. The physical interface consists of 32 data lines and 15 control lines. The control signals required are shown in Table 8.

The connections shown in Figure 22 indicate data flow. Control connections from the controller to each element are not shown. The functions of the various elements are described below.

Controller. The controller provides the interface signals to the host computer to cause data transfers to be performed and provides gating and timing signals within the MP.

Memory Map. The memory map provides data to the controller relating memory location in the host computer to data type, data location in the SP and data location in the data memory of the MP.

Comparator. The comparator determines whether data memory matches the input data. If the two do not match, the input data are sent to the transmit buffer. In the test and verification mode, data that do not agree are sent to the error log.

Data Memory. Data memory holds the present value of each variable. Data for analog quantities are held in both floating-point format and integer format to save computation time. The integer-format data are updated on each cycle, but the floating-point format data are updated only when there is a change.

Transmit Buffer. Data received from the host computer that do not match the corresponding data in data memory are stored in the transmit buffer. Data in this buffer are transmitted to the appropriate SP via the serial transmission line.

Receive Buffer. The receive buffer stores data from the SP until they can be processed. These data are used to update the information sent to the host computer.

Transmitter/Receiver. The transmitter takes parallel data from the transmit buffer, formats them, and sends them serially to the SP. At the same time, the receiver accepts serial data from the SP's and stores them in the receive buffer.

TABLE 7. SERIAL LINK INTERFACE SIGNALS

SIGNAL	DESCRIPTION
Frame Strobe	A timing signal indicating the frame timing. A frame consists of 18 clock intervals, but only 16 correspond to data.
Clock Output	A sequence of 16-bit bursts of clock signals which indicate the time when data and address outputs should be sampled.
Data Output	A 16-bit serial word giving the data from the MP to the SP.
Address Output	A 16-bit serial word giving the address of the external device and the operating mode requested.
Clock Input	A sequence of 16-bit bursts of clock signals which indicate the time when data and address inputs should be sampled.
Data Input	A 16-bit serial word giving the data from the SP to the MP.
Address Input	A 16-bit serial word giving the address of the source of the data transmitted.

TABLE 8. HOST COMPUTER INTERFACE SIGNALS

SIGNAL NAME	SYMBOL
Information Transfers:	
External Function Ready	EF
External Function Acknowledge	EFA
Status Ready	ISR
Status Acknowledge	ISA
Output Data Ready	ODR
Output Data Acknowledge	OA
Output Data End of Transfer	LWF
Input Data Ready	IDR
Input Data Acknowledge	IA
Input Data End of Transfer	DEB
Unusual Termination:	
Device Generated	EXT
HSD Generated	TDV
Other:	
Reset	IOR
Device Present	DP
External Address Control	EM

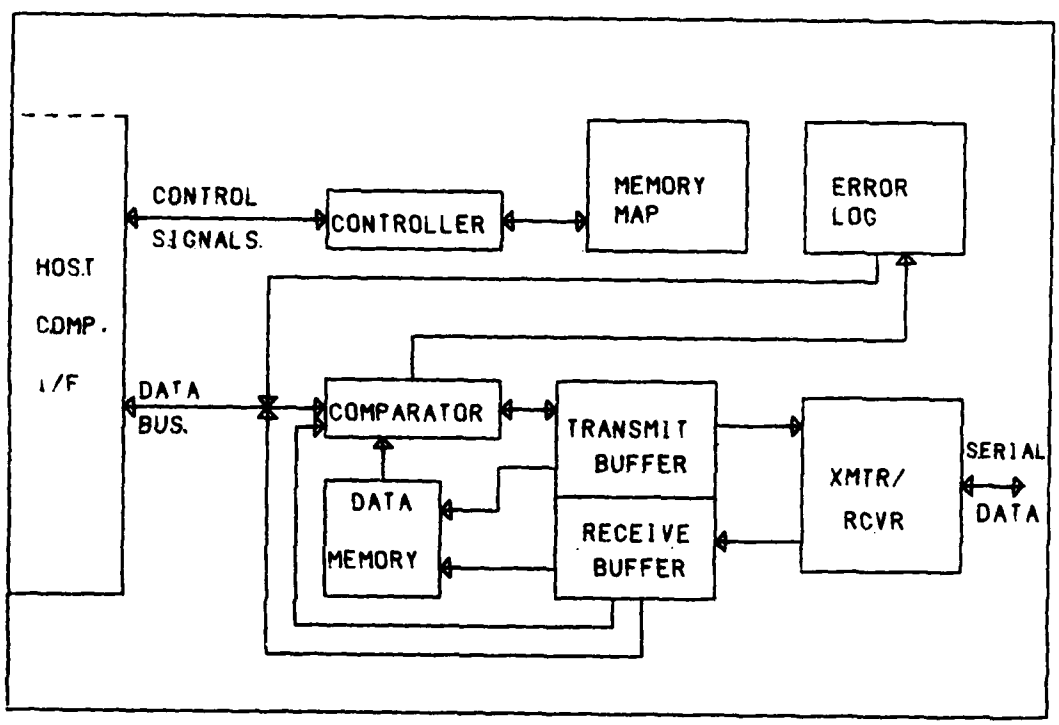


Figure 22. Block Diagram of Master Processor.

SLAVE PROCESSOR. Figure 23 shows a block diagram of the complete SP. In the prototype linkage, the blocks to the right of the bus are not implemented, and communication between the slave processor and the system cards is performed via the serial linkage now used by the subcontroller. Data and address information is provided in a serial data stream at a one megahertz clock rate. The signals that must be provided are listed in Table 9.

The SP's are connected to the MP via the 7-signal cable described above. The functions of the various elements in the SP are described below.

Controller. The controller provides the interface signals to the interface cards to cause data transfers to be performed and provides gating and timing signals within the SP.

Memory Map. The memory map provides data to the controller relating memory location in the SP to data type, interface card address, and data location in the data memory of the MP.

Comparator. The comparator determines whether data memory matches the input data. If the two do not match, the input data are sent to the transmit buffer. In the test and verification mode, data that do not agree are sent to the transmit buffer.

Data Memory. Data memory holds the present value of each variable. The memory contents are updated each time the information changes.

Transmit Buffer. Data received from the interface cards that do not match the corresponding data in data memory are stored in the transmit buffer. Data in this buffer are transmitted to the MP via the serial transmission line.

Receive Buffer. The receive buffer stores data from the interface card until they can be processed. These data are used to update the information sent to the MP.

Transmitter/Receiver. The transmitter takes parallel data from the transmit buffer, formats them, and sends them serially to the MP. At the same time, the receiver accepts serial data from the MP and stores them in the receive buffer.

Digital Storage. The digital storage is provided to hold discrete information sent from the SP to the interface. This is logically equivalent to data memory, but it is implemented in static registers in order to drive external devices. In the prototype linkage, this function is provided by the existing system card.

Switches. Switches controlled by the SP provide the multiplexing and demultiplexing to select elements in the digital storage. For analog inputs, an analog switch selects which analog signal is to be converted. This function is provided by the existing system card.

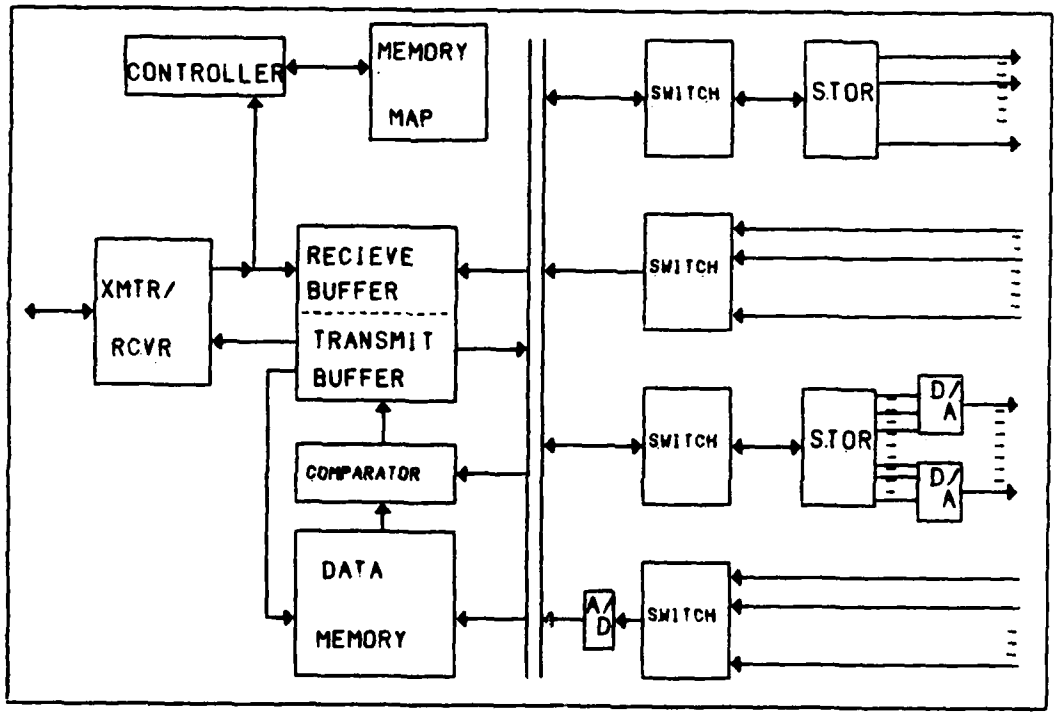


Figure 23. Block Diagram of Slave Processor.

TABLE 9. SYSTEM CARD INTERFACE SIGNALS

SIGNAL	DESCRIPTION
Clock	A sequence of 16-bit bursts of clock signals which indicate the time when data and address inputs should be sampled.
Data Output	A 16-bit serial word giving the data to be processed by the selected system card.
Address Output	A 16-bit serial word giving the address of the external device and the operating mode requested.
Input Data	A 16-bit serial word giving the data from the system card.
Analog Input	Since analog-to-digital conversion is performed external to the system card, analog input cards provide the selected analog signal.
Address Strobe	A control signal that indicates that a sub-controller match has occurred.
Card Enable	The card address portion of the address word is decoded external to the system card, and a separate enable output is provided for each card.
Card Select	This logic level enables the selected card. Only the selected card responds to the signals.
Address Strobe	This signal provides the clock to indicate presence of the card select signal. It sets the frame timing for the 16-bit serial data.
Read Strobe	Signal used by the system card to sample its input data.
Store Strobe	System used by the system card to sample its output data.
Read Strobe	Control signal to indicate that input data from the system card are to be read.
Reference Voltages	Reference voltages of 1.7 and 1.56 volts are required by the system cards.

IMPLEMENTATION

Microcomputers are the main component for implementing the functions described above, but there are many places where special-purpose logic units are required. The basic scheme selected for the implementation is a network of microcomputers arranged as a distributed processing system. This configuration is used to perform the bulk of the computations required to convert between floating-point and integer representation of values and to extrapolate values and determine which values must be transmitted.

The controller function is implemented by a single microprocessor which communicates with the network of microcomputers via individual direct memory access (DMA) channels to each microcomputer's private memory. Ancillary to the controller's microprocessor are several special-purpose logic units which handle data at higher rates than are possible with microprocessor control.

MICROCOMPUTER NETWORK. Figure 24 shows a block diagram of the multiple microcomputer configuration. In addition to the private memory assigned to each processor, there is a shared memory which holds the memory map and the data memory. The private memory is used only for programs, temporary storage during computation, and communication with the controller. The private memory is provided to avoid the time lost in waiting for memory access that would occur if all processors shared one memory.

Communication between the controller and the microprocessor network is performed in two ways. The controller assigns jobs by entering data into the microprocessor's private memory. The microprocessor performs the task and stores any result to be passed to other devices in its private memory. It also updates the shared memory as required. When the microprocessor has completed its task, it transmits an interrupt to the controller.

MASTER PROCESSOR. Figure 25 shows a block diagram of the master processor implementation. The functions of its various elements are described below.

Controller. The controller consists of a microprocessor with its own private memory and access to the shared memory. Its private memory is connected via DMA channels to transmitter and receiver. It is able to communicate through I/O operations with the interface logic to the HSD and the DMA channel of the private memory in the microprocessor network. It has interrupts inputs from each microprocessor in the microprocessor network and form the interface logic.

Interface Logic. The interface logic generates the signals necessary to transfer data between the host computer and the shared memory. It is controlled by an I/O channel from the controller. The interface logic interrupts the controller on a request for action from the host computer.

Transmitter. The transmitter accepts data from the controller's private memory via its DMA channel and sends these data in serial format to the SP. Control of the transmitter is by a separate I/O channel in the controller.

Receiver. The receiver accepts data from the SP via the serial data link and stores this information in the controller's private memory. Control of the receiver is by a separate I/O channel in the controller.

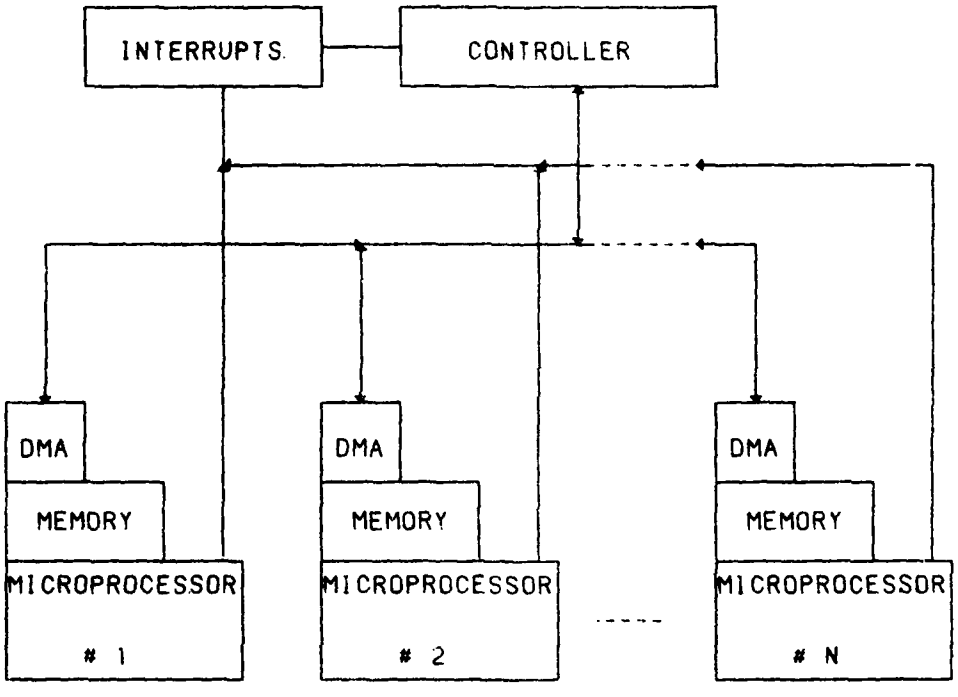


Figure 24. Block Diagram of Microcomputer Network.

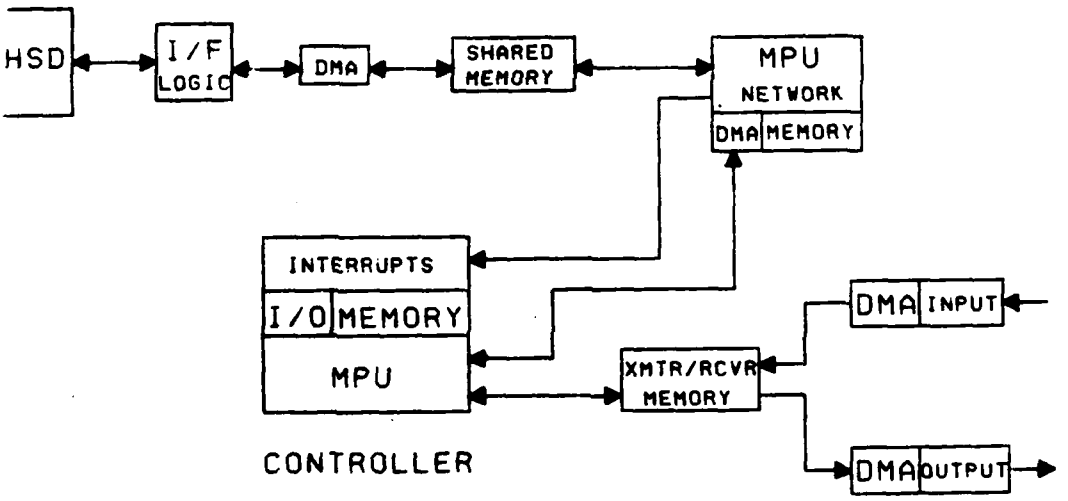


Figure 25. Master Processor Implementation.

SLAVE PROCESSOR. Figure 26 shows a block diagram of the slave processor implementation. The functions of its various elements are described below.

Controller. The controller consists of a microprocessor with its own private memory and access to the shared memory. Its private memory is connected via DMA channels to two sets of transmitters and receivers. It communicates through I/O channels, the DMA channels of the private memory in the microprocessor network, the transmitters and receivers connected to the serial transmission line to the MP, and the interface logic. It has interrupts inputs from each microprocessor in the microprocessor network.

Transmitter. There are two transmitters in the SP. The first transmitter accepts data from the controller's private memory via its DMA channel and sends these data in serial format to the MP. The second accepts data from the controller's private memory and sends these data in serial format to the interface cards. The first transmitter is controlled by a separate I/O channel from the controller. The second is controlled by the interface logic.

Receiver. There are two receivers in the SP. The first receiver accepts data from the MP via the serial data link and stores this information in the controller's private memory. The second receiver accepts data from the interface card serial transmission line and stores the information in the controller's private memory. The first receiver is controlled by an I/O channel from the controller. The second is controlled by the interface logic.

Interface Logic. The interface logic generates the control signals necessary to execute data transfers between the SP and the systems cards. This logic is controlled by a separate I/O channel from the controller. In turn, the interface logic controls the operation of the associated transmitter and receiver as well as the analog to digital converter.

Analog to Digital Converter. The analog to digital (A/D) converter accepts an analog signal from the selected system card and converts the voltage into a number which is stored in the controller's private memory. Operation of the A/D converter is controlled by the interface logic.

EVALUATION

The next step is to show that the design described meets the requirements set forth in Section I.

MODULAR DESIGN. The purpose in the modular design requirement is to allow the basic linkage system to be used on simulators having different numbers of signals to be processed. This requirement is met by the network of microprocessors used to perform most of the computation. The result is a single linkage channel that can handle a larger workload by adding identical processor stages to the basic unit.

BUS INTERFACES. The bus interfaces described are based upon the existing VTRS interface. However, the interface logic is a modular part of the design and can be changed without altering the basic configuration. Later models of this linkage system can be expected to use a standard serial linkage protocol such as MIL-STD-1553B. The computer interface matches a high-speed device designed

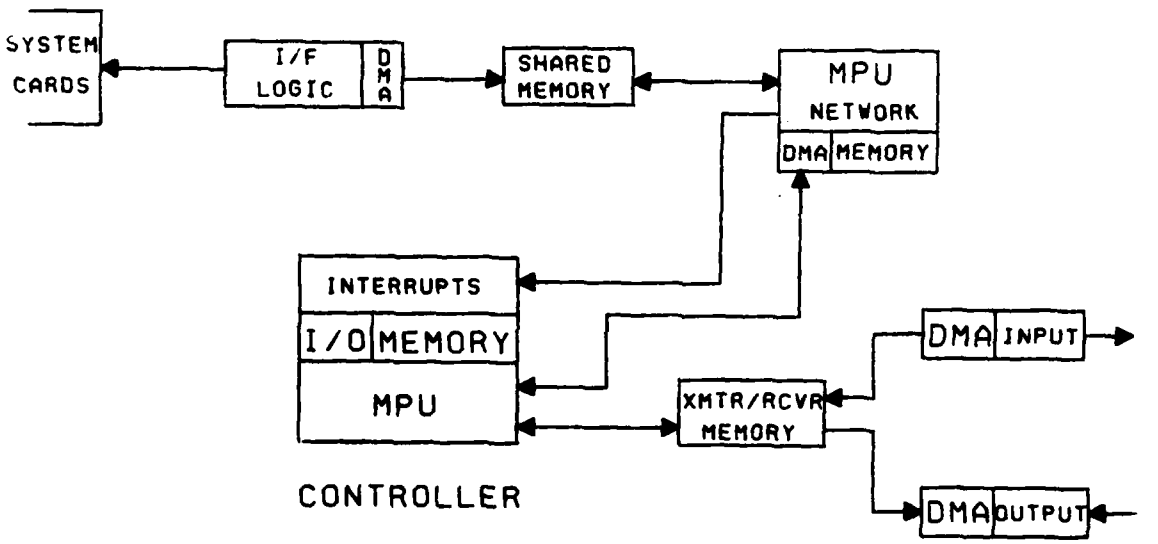


Figure 26. Slave Processor Implementation.

by the computer vendor. Use of another computer would require a similar design to match whatever interface is supplied by that computer manufacturer.

SOFTWARE FLEXIBILITY. The software requirements and how they are met is covered in Section IV. The hardware, with its use of microcomputers, supports a wide range of software to achieve the full flexibility specified.

FAIL-SOFT CAPABILITY. The fail-soft capability is achieved by providing the master and slave processors sufficient computational capacity to handle the full data transmission within one iteration cycle.

SELF-VERIFICATION. Self-verification is provided by programs within the master and slave processors which perform data exchange and verification during the dead time between the transmission of all updated values and the beginning of the next iteration cycle.

SECTION IV

SOFTWARE REQUIREMENTS

The software requirements for the linkage system encompass both real-time software and support software. The real-time software is that set of programs which are executed by the various computers in the system during the operation of the simulation. Support software includes aids for developing the real-time software, routines to assist in the allocation of data within the processors and assigning scale factors, and various utility programs.

REAL-TIME SOFTWARE

The process of data translation, extrapolation and transfer requires four different real-time computer programs to be executed in the linkage microcomputers. In addition, the host computer must be programmed to communicate with the linkage system via the interface.

HOST COMPUTER SOFTWARE. The part of the host computer program that communicates with the linkage is like the program used in any general-purpose computer to initiate an I/O channel. Transfers are initiated by a CD command which specifies an I/O command block (IOCB). The IOCB holds the information about the transfer to be performed. Table 10 shows the contents of the IOCB.

When the CD instruction is executed, the HSD transmits the contents of Word 0 of the IOCB to the external device, together with the external function code. This allows the device to determine what it is to do and what is expected of the HSD.

The flexibility of the I/O system is indicated by the functions available in the HSD operation code. Table 11 shows the function of each bit. Table 12 shows a sample IOCB used in the VTRS program.

MASTER PROCESSOR SOFTWARE. The MP compares the present value of each variable to its previous value to see if it is within tolerance. If not, the value is translated from the floating point format used in the host computer into a scaled fixed point value used for digital to analog conversion. This new value, together with addressing information, is placed in a buffer memory area for use by the transmitter logic. The MP also converts scaled fixed point values received from the SP's into floating point format for transmission to the host computer.

The software for the controller microprocessor consists of several real-time routines for handling the various data processing and data transfer operations and a utility program to allow loading of both the MP and the SP's from the host computer. These routines are described below.

Accept Data Block. This real-time routine is initiated by an external function command from the host computer. The controller sets the DMA channel that communicates with the HSD to accept the data transfer.

TABLE 10. INPUT-OUTPUT COMMAND BLOCK

WORD 0		
	Bits 0-7	HSD Operation Code
	Bits 8-15	Device Dependent Information
	Bits 16-31	Transfer Count
WORD 1		
FORMAT A		
	Bits 0-7	Not Used
	Bits 8-31	Buffer Memory Address
FORMAT B		
	Bits 0-31	Device Dependent Command
WORD 2		
	Bits 0-31	Reserved for Software
WORD 3		
	Bits 0-31	Error/Device Status

TABLE 11. BIT FUNCTIONS IN HSD OPERATION CODE

Bit 0	0/1 indicates output/input data transfer.
Bit 1	Command Transfer. Transmits contents of word 1 to external device as an External Function (EF).
Bit 2	Device Status Request. Accept Input Status Read (ISR) signal for external device and store status form device in data buffer.
Bit 3	Continue on Error. An error normally causes the HSD to terminate its operation. When either the command or data chaining is specified (bits 6 or 7), bit 3 set causes the next IOCB command specified by the command or data chain to be executed.
Bit 4	Interrupt on End of Block. This bit signals the HSD to generate an SI request when the current IOCB is completed.
Bit 5	Transfer in Channel. When this bit is set, the HSD uses word 1 on the current IOCB as the address of the next IOCB.
Bit 6	Chain Command. When this bit is set, the HSD executes the next IOCB in sequence upon completion of the present IOCB.
Bit 7	Data Chain. When this bit is set, the HSD executes the next IOCB in sequence (as in Command Chain) except that Command bits 0-3 are not used.

TABLE 12. SAMPLE INPUT-OUTPUT COMMAND BLOCK

WORD	COMMAND	DESCRIPTION
0	42000000	Command Transfer/Command Chain. Execute transfer then go to next block.
1	82820000	Information to be transferred to device.
2	00000000	Not used.
3	00000000	Not used.
0	8800000A	Input Transfer/Interrupt at End. Transfer 10 words from external device and interrupt at end.
1	00A00000	Address of beginning memory location where data are to be stored.
2	02000130	Information for use of software in processing data.
3	00000000	Not used.

Process Data from Host Computer. This real-time routine is initiated by completion of the data transfer from the host computer. The controller assigns the processing sequentially to each of the microprocessors in the network. The processors respond with interrupts each time a process is complete, and the controller assigns the next operation.

Data Transmission to SP. This real-time routine places the data identified by the data processing routine above, formats the data and places the data in a stack for transmission. This routine is more complex than it appears, because the DMA for transmission is automatically taking data from this stack.

Process Data from SP. This real-time routine takes data from the SP as they are received and assigns them sequentially to the next microprocessor in the network for conversion and formatting. This operation is carried on at the same time as processing data from the host computer.

Transmit Data Block. This real-time routine causes the HSD to accept a block of data from the MP via the DMA interface.

Verification. This real-time routine transmits data to the SP during the dead time after all data processing for the frame has been completed and responds to the replies from the SP. Data that are incorrect are retransmitted. Steady-state errors are reported to the host computer immediately; intermittent errors are recorded in the error log and reported only if the number of errors exceeds a preset threshold.

Statistics. This real-time routine operates during the data processing cycle to keep statistics on the number of data points transmitted. It keeps a record of peak and average number of data words sent and received over the serial transmission line.

Utility Programs. This set of non-real-time programs allows the host computer to load the MP and SP memories, interrogate these memories, and perform diagnostics on the MP and SP via the DMA interface.

SLAVE PROCESSOR SOFTWARE. The SP compares the present value of each of its stored variables to the values received from the system cards to see if it is within tolerance. If the new value is out of tolerance, the value and its address are placed in a buffer memory area for use by the transmitter logic.

The software for the controller microprocessor consists of a several real-time routines for handling the various data processing and data transfer operations and a utility program to allow loading of SP from the host computer via the MP. These routines are described below.

Accept Data Block. This real-time routine is initiated by a signal from the MP. The program causes the system cards to be sampled and the data transferred into the SP's memory via DMA.

Process Data from System Cards. This real-time routine is initiated by receipt of data from the system cards. The controller assigns the processing sequentially to each of the microprocessors in the network. The processors respond with interrupts each time a process is complete, and the controller assigns the next operation.

Data Transmission to MP. This real-time routine places the data identified by the data processing routine above, formats the data and places the data in a stack for transmission. This routine is more complex than it appears, because the DMA for transmission is automatically taking data from this stack.

Process Data from MP. This real-time routine takes data from the MP as they are received and assigns them sequentially to the next microprocessor in the network for conversion and formatting. This operation is carried on at the same time as processing data from the host computer.

Verification. This real-time routine accepts data from the MP during the dead time after all data processing for the frame has been completed and compares them to what is stored at the SP. Any discrepancies are reported to the MP so that the data can be retransmitted.

Utility Programs. This set of non-real-time programs allows the MP to load the SP memories, interrogate these memories, and perform diagnostics on the SP via the serial linkage.

MICROCOMPUTER NETWORK. A more detailed look at the processing programs in the microcomputer network is necessary in order to establish time and storage estimates. This part of the master or slave processors is the leading item in terms of both cost and performance.

The most significant operation performed by the microprocessor network in the MP is the conversion between binary and floating-point representation of data. Figures 27 and 28 show flowcharts for these conversions. A program was written for a 16-bit microprocessor to determine the time requirement. Conversion from binary to floating point notation takes 85 microseconds, and conversion from floating point to binary takes 55 microseconds.

TIME ESTIMATES. Analysis of the linkage system performance requires an estimate of the time required to perform each function. Table 13 shows the major functions and the estimated time to perform each function. These functions are assigned symbols for use in the general analysis shown in Section V.

SUPPORT SOFTWARE

Support software that must be provided to make the linkage system useful consists of programs to be run on the host computer to assist in assignment of data locations, developing scale factors and loading the MP and SP from the host computer. This software must be written before a linkage system can be put in the field, but it will not be available to provide help during the prototype development phase.

Software support during the development phase will consist of compilers, assemblers and utility packages available on the host computer. It will be necessary for the software developers to use these existing aids as effectively as possible to provide adequate support to test the prototype linkage system.

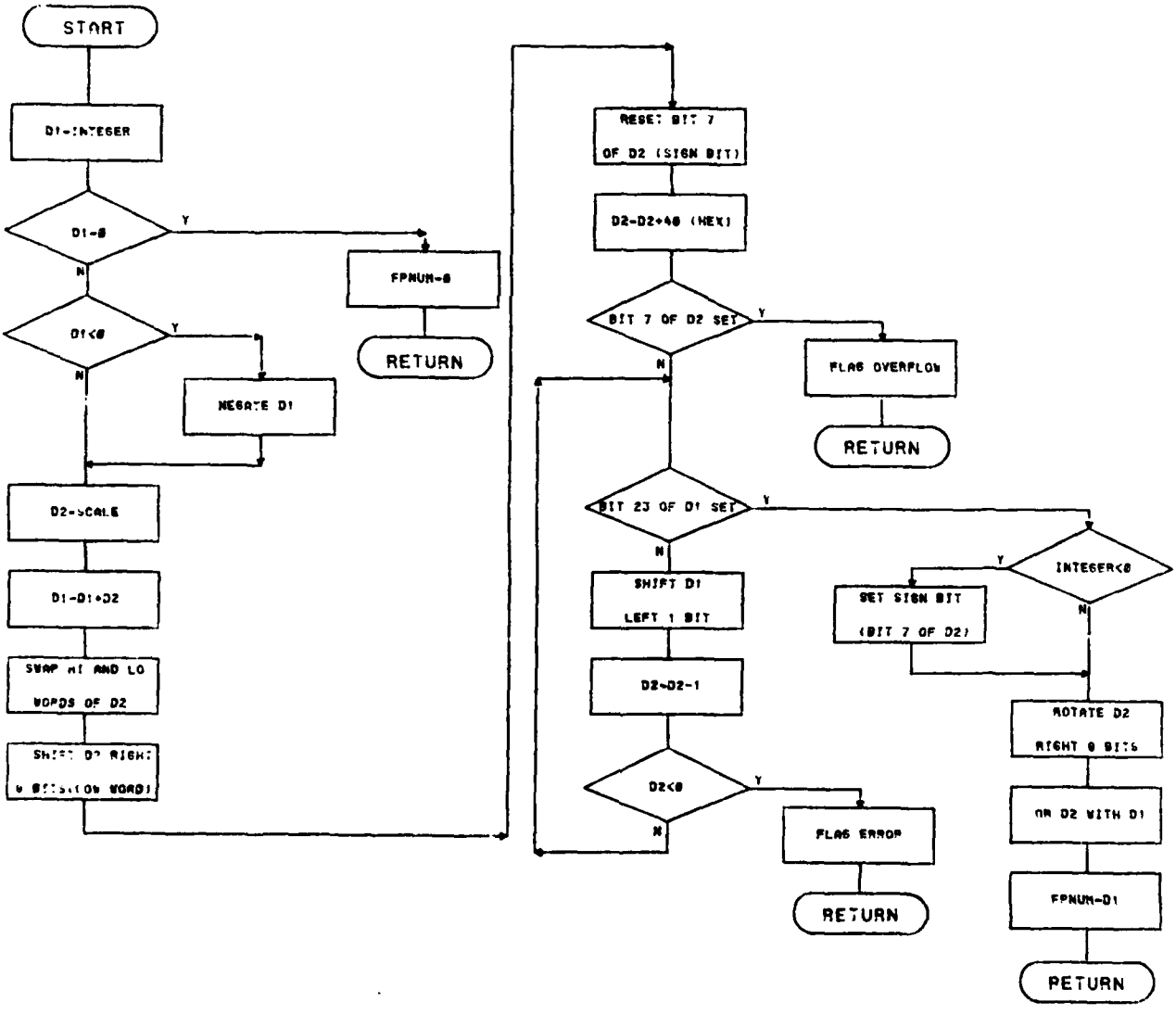


Figure 27. Flowchart for Binary to Floating Point Conversion.

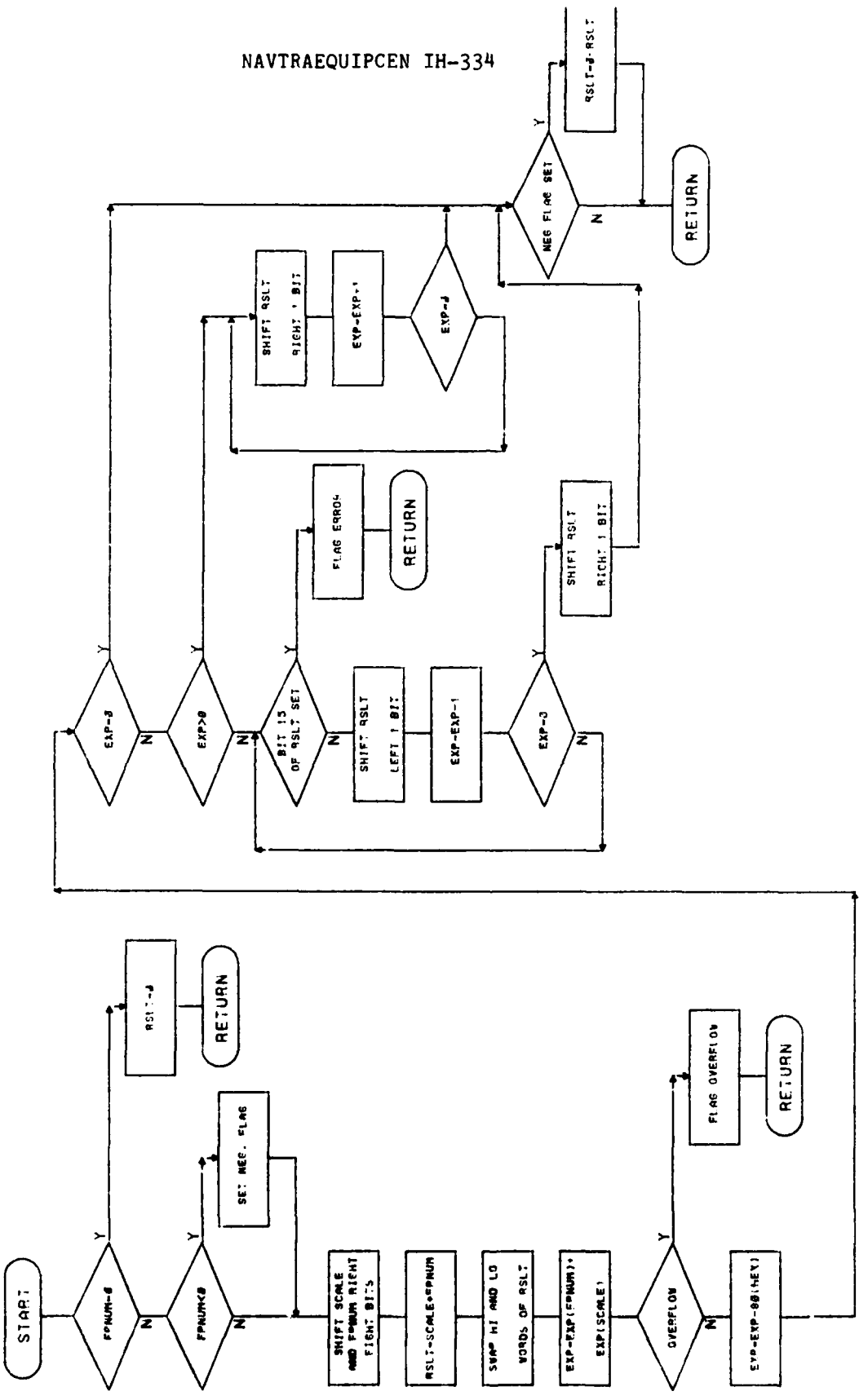


Figure 28. Flowchart for Floating Point to Binary Conversion.

TABLE 13. PROCESSING TIME ESTIMATES

FUNCTION	SYMBOL	TIME (Microseconds)
Transfer Data from Host to MP	X	1
Controller Functions	C	5
Floating Point Comparison	A	10
Integer to Floating Point Conversion	F	85
Floating Point to Integer Conversion	V	55
Discrete Comparison	D	7
Serial Transmission	K	18

SECTION V

SYSTEM OPERATION

Analysis of the linkage system to predict its performance is a necessary part of the design. This analysis provides: (1) an illustration of the linkages system's advantages over other approaches, and (2) a basis for selection of the number of microcomputers to be used in implementing the MP and SP's for a given application.

The processing to be performed by the linkage system is defined by the number and type of signals and the signals' statistical properties. The linkage hardware can be described in terms of its processing speed in performing each subtask related to handling these signals. The analysis task is to relate the hardware capability and the signal statistics in order to provide performance measures of the linkage system.

PROBLEM DESCRIPTION

Signals to be processed by the linkage system are classified by type (analog or discrete) and by direction of signal flow (input or output). In order to investigate the effectiveness of the linkage system, these signals must be described statistically. The following statistical model is assumed for this purpose:

a. the signals are independent of one another, and

b. each signal of a given type has the same fixed probability of change during each sampling interval.

Although these statistical properties cannot be said to be a true representation of the real signals, they provide a reasonable vehicle to illustrate the dependence of the linkage system performance upon signal properties.

Table 14 shows the signal parameters and the probability of change assigned to each. The number of signals of each type are based upon VTRS requirements. The probabilities are selected in the analysis to follow.

ANALOG SIGNALS. The probability of change for the analog signals examined in Section II shows too great a variation to base the entire analysis on a single average value. The probabilities of change for the input and output data will be assumed the same, and the single probability will be carried through the analysis as a variable. The average value that resulted from the data examined is shown in Table 14, but it will be used only for an overall estimate of system performance.

DISCRETE SIGNALS. Since no experimental data are available, the probability of change for discrete signals will be derived. Begin by considering each discrete signal to be independent and to have a probability p of changing during a time interval. Since discrete signals are handled in 16-bit words, the probability that a word is different from its last value is

Table 14. Signal Parameters

SIGNAL			CHANGE PROBABILITY	
TYPE	SYMBOL	VALUE	SYMBOL	VALUE
Analog Input	A_I	189	p_{AI}	.20
Analog Output	A_O	216	p_{AO}	.20
Discrete Input	D_I	44	p_{DI}	.0018
Discrete Output	D_O	112	P_{DO}	.0018

$$1 - (1 - p)^{16}$$

Now consider what value is appropriate for p. If a signal changes on the average of r times per second, the probability that it changes in a 1/30 second time interval is r/30. Substituting this in the expression above gives the results shown in Figure 29.

For the simulator problem being considered, an average rate for changing a discrete signal might be in the order of one every 5 minutes. Using this value for r gives gives a probability of a change in a word of 0.0018. This value is used for the analysis of the linkage performance.

PROCESSING TIME

The parameter of interest in evaluating the linkage system is the processing time required. This is dependent upon the hardware configuration chosen, particularly how many microcomputers are used in the computation network in the MP, and one purpose of this analysis is to provide a rational basic for the choice. There are three processing times to be considered in the analysis:

T_{NET} = *Time for the microcomputer network to process* all variables.

T_{CONT} = Time for the controller microprocessor to process all variables.

T_{LINK} = Time for the data to be transmitted over the serial transmission line.

In general, these processes occur simultaneously and the controlling time factor is the greatest of the three.

Each of these times can be expressed in terms of a time to perform each function and the number of variables to be processed. The values of the parameters used in this analysis are those shown in Table 13 in Section IV. The analysis consists of deriving the equations for the times defined above, using the results to determine what processor configuration should be used, and determining the characteristics of the final system.

EQUATION DERIVATION. The time required by the microporcessor network to perform its functions is the time to perform the following operations: (1) comparison of all discrete variables, (2) comparison of all analog variables, (2) conversion of all output analog variables that have changed from floating point form to binary form, and (4) conversion of all input analog variables that have changed from binary form to floating point form. Since the total

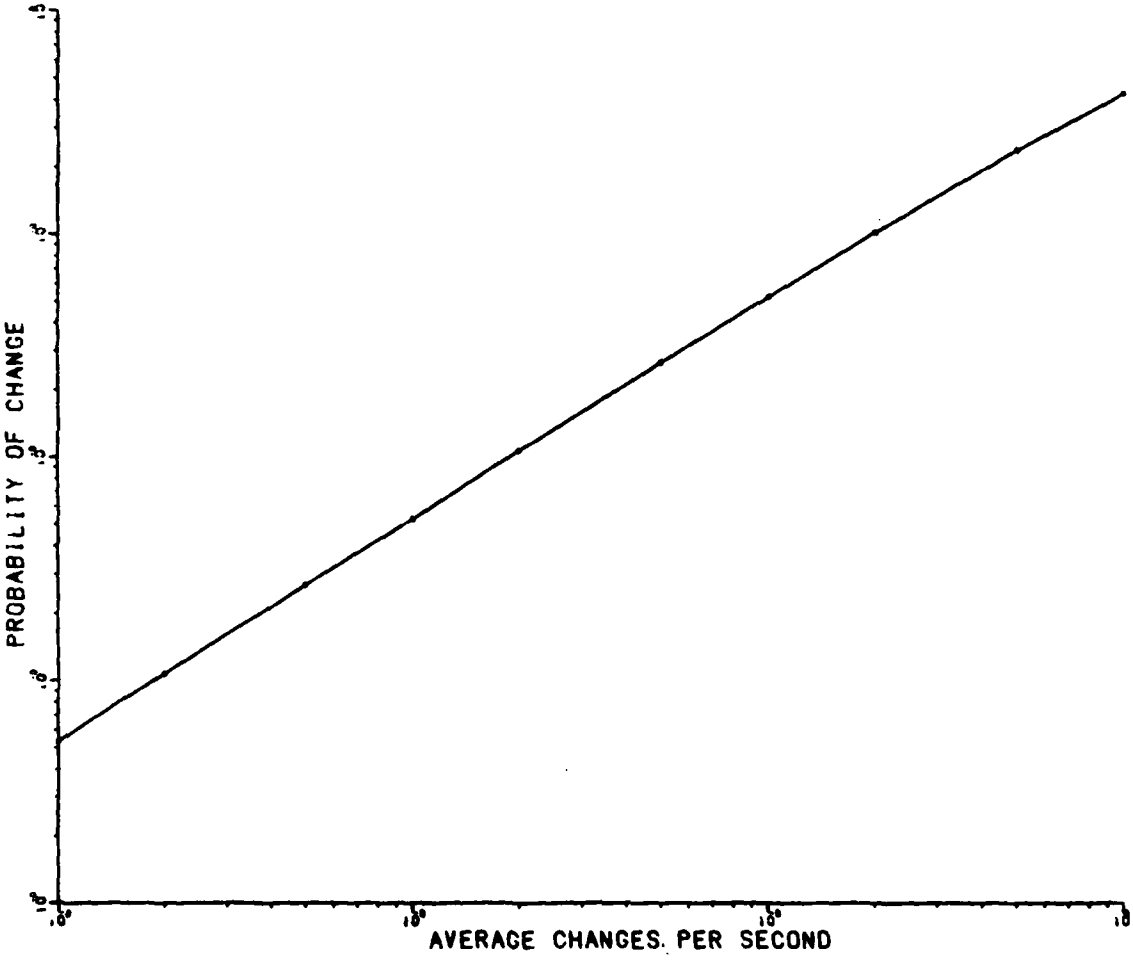


Figure 29. Probability That a Discrete Word Changes.

computation load is shared by n processors, the equation for the network processing time is:

$$T_{NET} = D(D_O) + A(A_O) + V(p_{AO}\, A_O) + F(p_{AI}\, A_I)$$

n

The processing time estimates and signal parameters used in the above equation are defined in Tables 13 and 14.

The time required by the controller to perform its functions is the time to transfer all variables from or to the host computer and the time spent by the controller in assigning the computations associated with these variables to the microprocessor network and storing the results in the correct buffer area. Because the controller processing time determines the overall system performance in some cases, the use of multiple microcomputers in implementing the controller must be considered. The equation for the controller processing time using m microprocessors is:

$$T_{CONT} = (X + C/m)(D_O + A_O + D_I + A_I)$$

The processing time estimates and signal parameters used in the above equation are defined in Tables 13 and 14.

The time required to transmit all data that change over the serial linkage is the product of the total number of words sent and the time to send each word. Since separate lines are used for sending and receiving, the time required is the greater of the sending or receiving time. The equation for the transmission time is:

$$T_{LINK} = K\, MAX[(p_{AO}\, A_O + p_{DO}\, D_O),(p_{AI}\, A_I + p_{DI}\, D_I)]$$

The processing time estimates and signal parameters used in the above equation are defined in Tables 13 and 14.

CONFIGURATION. The problem in choosing a configuration is to determine the number of microprocessors to use in implementing the microprocessor network and the controller. The basic idea is to match these two processing functions to the speed of the serial transmission line. Begin by matching the speed of the computation network to the speed of the serial line. The values used, except for the probability of change in the analog signals, are those shown in Tables 13 and 14. The effect of the unknown probability is best handled graphically.

The result indicates that 8 microprocessors are required for the VTRS application. Figure 30 shows the processing time for the serial linkage and the microprocessor network for this configuration. Observe the close match in processing time over the entire range of probabilities.

The second parameter to be determined is the number of microprocessors to be used in implementing the controller. This decision is a tradeoff based upon the point of diminishing returns. Consider first the cost of the MP in terms of equivalent microprocessors. The microcomputer network is implemented by 8 microprocessors, and the hardware external to the controller is equivalent to another 5 microprocessors. The minimum cost system, using a single microprocessor for the controller, is then equivalent to 14 microprocessors.

Observe that the minimum system meets the worst case criterion, because a single-microprocesser controller can handle all of the data faster than the serial linkage. However, when only 10 or 20 per cent of the data must be transmitted, the controller is the slowest element in the process. Figure 30 shows that for this case the network processing and the transmission time are on the order of 1000 microseconds. Figure 31 shows the processing time for the controller as more microprocessors are added. The plot shows the additional microprocessors as addition cost. The cost factor is based upon adding additional processors to the basic 14 element network. A four processor controller is chosen for the VTRS implementation on the basic of diminishing returns.

PERFORMANCE. System performance is shown in Figure 32. The limiting factor is the slowest of the three critical elements: controller, serial linkage, and microprocessor network for each processing load. For very low processing loads, the controller limits the speed of data transfer. For high loads, the linkage and processor network limit the speed. Compare the system performance to the 5.9 milliseconds required to transmit all data.

SLAVE PROCESSOR CONFIGURATION. A similar analysis was performed in order to determine the configuration of each slave processor. Because of the lesser computational load placed on the SP, a single microcomputer in the controller and a single microcomputer in the microcomputer network are sufficient to match the speed of the MP. The reasons for the reduced computational load are: (1) the SP's process less than one-third the number of signals handled by the MP, and (2) the SP's do not perform any binary to floating point conversion.

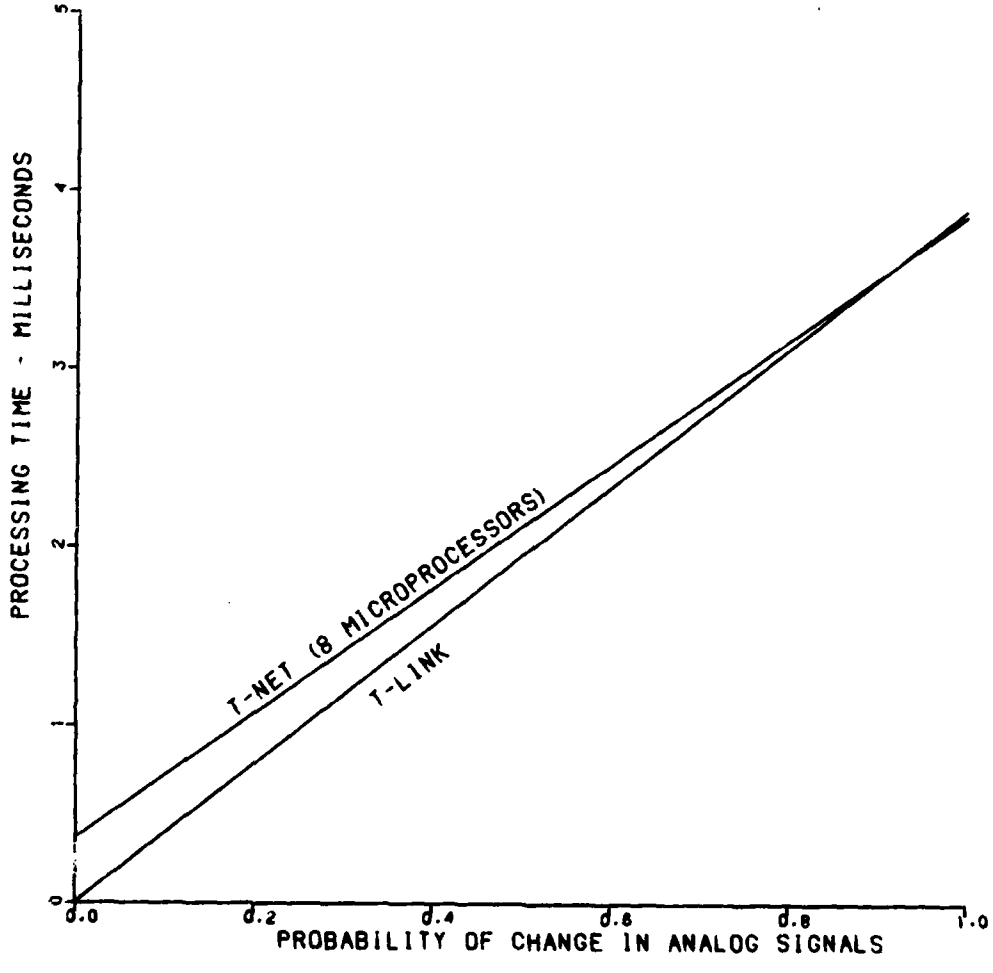


Figure 30. Processing Time for Serial Linkage and Microcomputer Network.

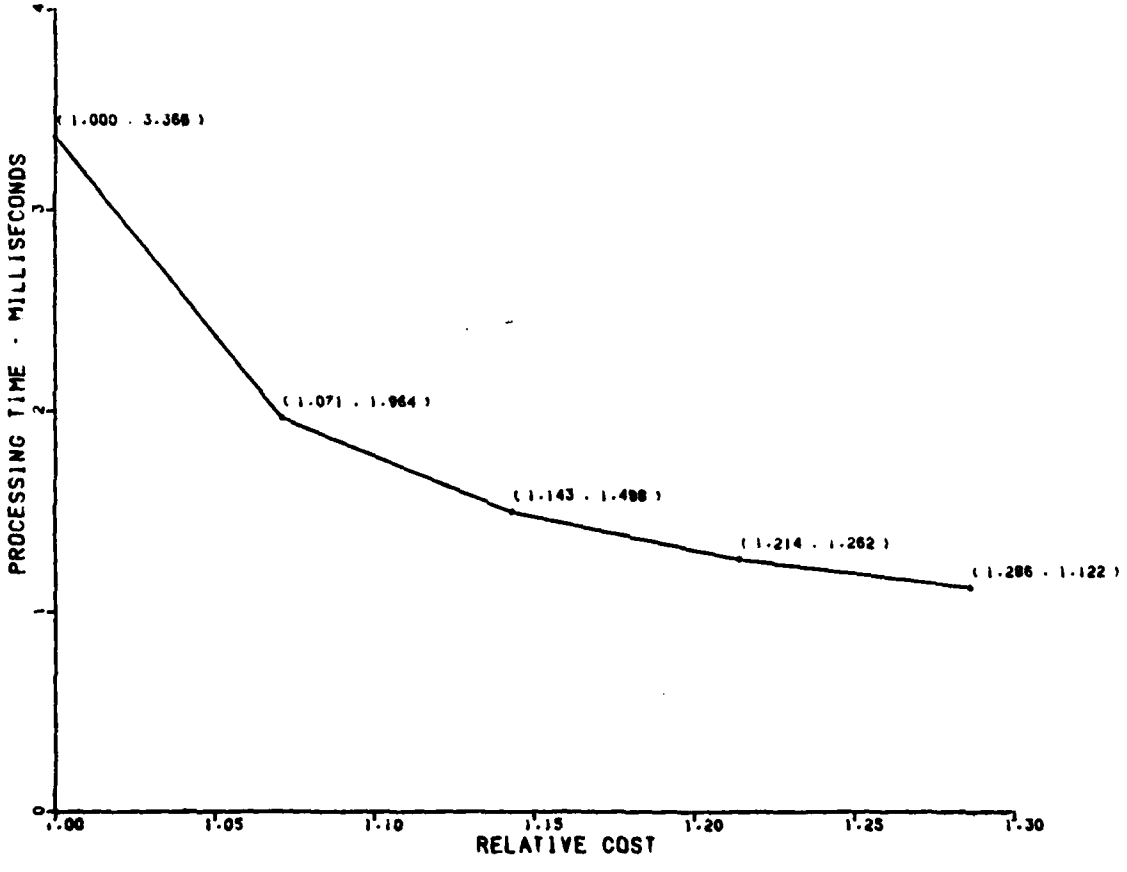


Figure 31. Processing Time for Controller.

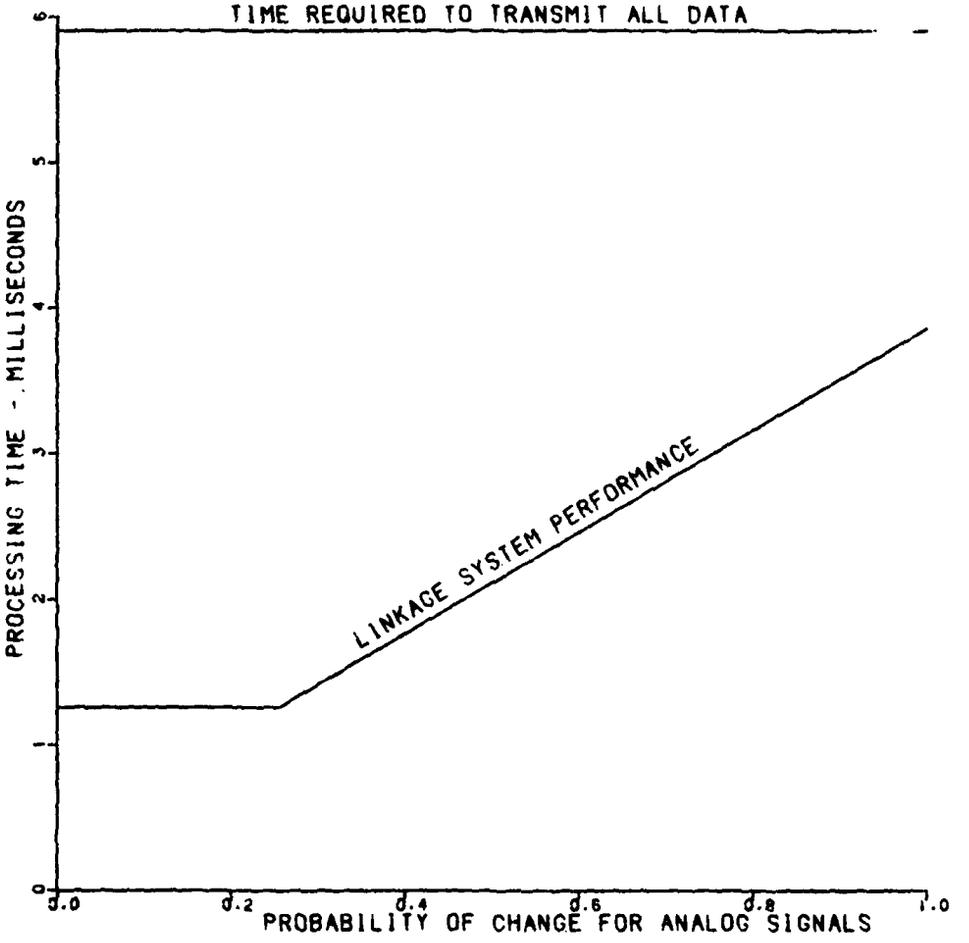


Figure 32. System Performance.

SECTION VI

SUMMARY AND CONCLUSIONS

The linkage architecture that has been developed provides an effective solution to the problem of linking a host computer to external devices in a simulator. It takes advantage of the statistical properties of simulator data to reduce the information that is transferred, thereby insuring that those variables that do change are updated with minimum delay.

The design approach assures rapid response to control inputs when the trainee is performing a critical tracking task or similar high-precision maneuver. Only in the case of violent changes does the system degrade to the slower response time typical of a conventional serial linkage. And in that case, the rapid response is not usually critical to the trainees performance.

The architecture based upon use of multiple microcomputers provides other features not found in present linkage systems. The linkage hardware handles the scaling problem, allowing analog data to be represented in the host computer in floating-point notation in natural units. Because the unit is microcomputer controlled, it offers an unsurpassed diagnostic capability. The serial interface provided lends itself to use in test and measurement of simulator performance not related to maintenance operations. If this standard interface were adopted for general use, quality assurance personnel could obtain information independent of unique hardware of a particular trainer.

The microcomputer implementation is based upon a technology that is continuing to provide increased capability at lower cost. Features that require multiple processors in the present design can be expected to be performed by many fewer integrated circuits in the future. The approach presented offers a viable, long-term solution to simulator interfacing.

AREAS FOR FURTHER RESEARCH

Further development of the linkage system requires some experience in the use of the extrapolation technique in a simulator application. The VTRS offers a research vehicle to test the concept. It is recommended that implementation and testing be done in three steps.

The first step is a verification of the advantages of the statistical approach. The method is to build those parts of the master processor associated with the controller and interfacing to the host computer. The microprocessors in the controller can be programmed to gather the statistical information during actual training tasks. The data can be used to validate the approach and to refine the design of the processors.

The next step is to complete a master processor and two slave processors to interface with the VTRS system cards. This is the minimum system needed to test the concept in actual operation. By selecting the cockpit as the location for the two slave processors, the improvement in response time can be demonstrated by actual experiment.

Finally, the existing system cards can be replaced by a design better suited to microcomputer control. At this step the present serial transmission format can be replaced with one based upon a standard serial interface such as MIL-STD-1553A. This will complete the demonstration of a suitable linkage system for general use in training devices.

NAVTRAEQUIPCEN IH-334

DISTRIBUTION LIST

Naval Training Equipment Center (50)
Orlando, FL 32813

Defense Technical Info Center (12)
Cameron Station
Alexandria, VA 22314

PM TRADE, US ARMY (2)
Attn: SE
Orlando, FL 32813

ALL OTHERS RECEIVE ONE COPY

Chief of Naval Education & Training
Code N-331
Pensacola, FL 32508

Naval Air Systems Command
AIR 340D
Washington, DC 20350

Naval Air Systems Command
AIR 340F
Washington, DC 20350

Naval Air Systems Command
AIR 360B, Attn: B. Zempolich
Washington, DC 20350

Naval Air Systems Command
Library AIR 50174
Washington, DC 20350

Chief of Naval Material, Navy Dept.
Code MAT 08T
Washington, DC 20360

Chief of Naval Material, Navy Dept.
MAT 08Y
Washington, DC 20360

Chief of Naval Material, Navy Dept.
MAT 09Y
Washington, DC 20360

Chief of Naval Research
ONR 461
800 North Quincy Street
Arlington, VA 22217

Naval Air Systems Command
AIR 413
Washington, DC 20350

AF ASD/ENE
Attn: Mr. A. Doty
Wright-Patterson AFB, OH 45433

AF ASD/ENO
Attn: Mr. Phil Babel
Wright-Patterson AFB, OH 45433

Dept. of Electrical & Computer
Engineering
Attn: Dr. Harold Stone
University of Massachusetts
Amherst, MA 01003

Chief of Naval Operations
OP-115
Attn: LCDR J. Lawhon
Washington, DC 20350


ATE
LME