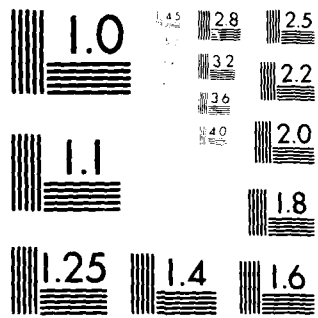


AD-A085 994

UNITED TECHNOLOGIES RESEARCH CENTER EAST HARTFORD CONN F/6 9/2
MICROPROCESSOR REQUIREMENTS FOR IMPLEMENTING MODERN CONTROL LOG--ETC(U)
APR 80 R W GUILE, J R KRODEL, F A FARRAR F49620-79-C-0078
UNCLASSIFIED UTRC/R80-944590-1 AFOSR-TR-80-0443 NL

1 of 1
AD-A085 994

END
DATE
FILMED
8-80
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AFOSR-TR- 80-0443

R80-944590-1

LEVEL III

PD 75-060

(12)

**MICROPROCESSOR REQUIREMENTS
FOR IMPLEMENTING
MODERN CONTROL LOGIC**

ADA 085994

**ROBERT W. GUILÉ
JAMES R. KRODEL
FLORENCE A. FARRAR**

**UNITED TECHNOLOGIES RESEARCH CENTER
EAST HARTFORD, CONNECTICUT 06108**

CONTRACT F49620-79-C-0078

April 1980

**FINAL REPORT FOR PERIOD
1 MARCH 1979 — 29 FEBRUARY 1980**

Approved for public release; distribution unlimited

**PREPARED FOR
AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
DEPARTMENT OF THE AIR FORCE
BOLLING AIR FORCE BASE
WASHINGTON, D.C. 20332**

**DTIC
ELECTE
JUN 25 1980**

DGC FILE COPY

80 6 11 002

Approved for public release;
distribution unlimited.

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DDC
This technical report has been reviewed and is
approved for public release IAW AFR 190-12 (7b).
Distribution is unlimited.
A. D. BESS
Technical Information Officer

REPORT DOCUMENTATION NUMBER		D INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER 18 AFOSR/TR-80-0443	2. GOVT ACCESSION NO. AD-A085994	3. RECIPIENT'S CATALOG NUMBER 9	
4. TITLE (and Subtitle) Microprocessor Requirements for Implementing Modern Control Logic		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report, 1 March 1979-29 February 1980	
6. AUTHOR(s) Robert W. Guile James R. Krodel Florence A. Farrar		7. PERFORMING ORG. REPORT NUMBER	
9. PERFORMING ORGANIZATION NAME AND ADDRESS United Technologies Research Center Silver Lane East Hartford, Connecticut 06108		8. CONTRACT OR GRANT NUMBER(s) 15 F49620-79-C-0078	
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research/AM Bolling Air Force Base Washington, D. C. 20332		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2304/A1 17 A1	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE 11 Apr 1980	
12/87		13. NUMBER OF PAGES 75	
		14. SECURITY CLASS. (of this report) Unclassified	
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited. The United States Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation hereon.			
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)			
14 UTRC/R80-944590-1			
18. SUPPLEMENTARY NOTES Preliminary Results presented at Institute of Electrical and Electronic Engineers Conference on Decision and Control on 7 December 1979 in Fort Lauderdale, Florida			
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Digital Control Implementation Microprocessor Control Accuracy Requirements Computational Requirements Memory Requirements			
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) A demonstration of the use of microprocessors for implementing linear quadratic Gaussian (LQG) control was conducted. The demonstration consisted of simulating linear system dynamics on an analog computer and implementing LQG control and estimation dynamics on a microprocessor. Two cases were studied: a single input second order system and a four input fifth order system. The second order system was controlled using an Intel 8080 8-bit microprocessor and the fifth order system was controlled using a 16-bit Digital Equipment			

DD FORM 1 JAN 73 1473 K

Unclassified
 SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)
 409252 *LM*

←
out
→
Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Corporation LSI 11/2 microprocessor. Key requirements addressed ~~in this study~~ included microprocessor requirements for: (1) word size; (2) computational capability, including arithmetic and input/output operations, and (3) memory requirements. The requirements were compared against predicted requirements made using previously developed analytic techniques.

The implementation involved developing general purpose algorithms required for implementing LQG control and estimation. These algorithms consisted of matrix/vector multiplication, vector addition and input/output service routines. The same algorithms were employed in both the second order and the fifth order demonstration.

A

Unclassified

R80-944590-1

FINAL TECHNICAL REPORT
REPORT R80-944590-1

MICROPROCESSOR REQUIREMENTS FOR IMPLEMENTING MODERN CONTROL LOGIC

ROBERT W. GUILÉ

JAMES R. KRODEL

FLORENCE A. FARRAR

Research sponsored by the Air Force Office of Scientific Research (AFOSR)
United States Air Force, under contract No. F49620-79-C-0078. The
United States Government is authorized to reproduce and distribute reprints
for governmental purposes notwithstanding any copyright notation hereon.

DTIC
ELECTE
JUN 25 1980
S D
C

Approved for public release, distribution unlimited

**UNITED TECHNOLOGIES
RESEARCH CENTER**



**UNITED
TECHNOLOGIES**

EAST HARTFORD CONNECTICUT 06108

79-03-108-8

FOREWORD

This final technical report documents research performed from 1 March 1979 to 29 February 1980 under Air Force Office of Scientific Research (AFOSR) Contract F496209-79-C-0078. This research program was conducted at United Technologies Research Center (UTRC), East Hartford, Connecticut 06108. Major Charles L. Nefzger served as the AFOSR Scientific Officer.

This report is issued as UTRC Report R80-944590-1.

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	

Microprocessor Requirements for Implementing
Modern Control Logic

TABLE OF CONTENTS

	<u>Page</u>
FOREWORD	i
TABLE OF CONTENTS.	ii
SUMMARY.	1
RESULTS AND CONCLUSIONS.	2
INTRODUCTION	4
CONTROL OF NONLINEAR STOCHASTIC SYSTEMS.	6
System Description.	6
Control Design Approach	7
Digital Implementation.	14
LINEAR SYSTEM RESULTS.	18
Microprocessor Requirements-Linear Systems.	18
APPLICATION AND VERIFICATION OF MICROPROCESSOR REQUIREMENT PROCEDURES	21
Second Order System	22
Fifth Order System	23
REFERENCES	27
LIST OF SYMBOLS.	29
TABLES	32
FIGURES	
APPENDIX A - MICROPROCESSOR SURVEY	A-1
APPENDIX B - INTEL 8080 SOFTWARE FOR LQG CONTROLLER.	B-1
APPENDIX C - LSI-11 SOFTWARE FOR LQG CONTROLLER.	C-1

Microprocessor Requirements for Implementing
Modern Control Logic

SUMMARY

A demonstration of the use of microprocessors for implementing linear quadratic Gaussian (LQG) control was conducted. The demonstration consisted of simulating linear system dynamics on an analog computer and implementing LQG control and estimation dynamics on a microprocessor. Two cases were studied, a single input second order system and a four input fifth order system. The second order system was controlled using an Intel 8080 8-bit microprocessor and the fifth order system was controlled using a 16-bit Digital Equipment Corporation LSI 11/2 microprocessor. Key requirements addressed in this study included microprocessor requirements for (1) word size (2) computational capability including arithmetic and input/output operations and (3) memory requirements. The requirements were compared against predicted requirements made using previously developed analytic techniques

The implementation involved developing general purpose algorithms required for implementing LQG control and estimation. These algorithms consisted of matrix/vector multiplication, vector addition and input/output service routines. The same algorithms were employed in both the second order and the fifth order demonstration.

Analytic procedures were developed for establishing microprocessor requirements for control of nonlinear systems. Techniques were described for designing nonlinear controls based on the application of LQG theory in which the nonlinear system dynamics were approximated by a series of linearized system descriptions at a number of operating points. Linear control and estimation methodology was applied to the linearized descriptions resulting in a series of piecewise optimal state variable feedback controls. Techniques are described for synthesizing the nonlinear control and estimation equations by scheduling the piecewise optimal gains between operating points. Microprocessor requirements were then predicted for the piecewise optimal configuration in terms of computational operation and total memory required.

This research was performed for the U.S. Air Force Office of Scientific Research under Contract F49620-79-C-0078.

RESULTS AND CONCLUSIONS

1) The implementation of modern control and estimation techniques based on linear quadratic Gaussian (LQG) methodology was successfully demonstrated using commercially available microprocessors. The demonstration was conducted for both a single input second order linear system and a four input fifth order linear system. The linear system dynamics were simulated on an analog computer and the LQG control was implemented using microprocessors. An eight bit Intel 8080 microprocessor was used to control the second order system and a sixteen bit Digital Equipment Corporation LSI 11/2 microprocessor was used to control the fifth order system.

2) Analytic techniques developed in the Phase I portion of the program were applied to predict the microprocessor requirements prior to implementation. These techniques resulted in procedures for addressing key issues associated with microprocessor implementation including word size, computational requirements, and memory. Application of the prediction techniques resulted in estimates for both the second order and the fifth order system in terms of these key issues. The actual requirements resulting from this study agree with predicted values. However, the particular characteristics of the interface between the microprocessor and the analog system i.e., the D/A converters and A/D converters may lead to additional software being required to ensure proper scaling. Also, additional software requirements arise because of the detailed characteristics of the hardware multiplication function. These two features are device dependent and the requirements in terms of software to work with them will depend on the devices selected. The analytic prediction techniques are very accurate in specifying the number and type of operation i.e., addition, multiplication, with some qualification. In the actual implementation it was found that additional software was required beyond that predicted. This additional software was required to (1) properly account for necessary scaling between the microprocessor and the analog simulation and (2) allow for signed multiplication and division when using a hardwired multiplication option on the LSI 11/2.

3) The Phase I study indicated that the number of operations required per sample time as well as the memory requirements were dependent on the structure of the system. Transformation could be used to express the system state vector in a new coordinate system which in turn could be used as a basis for designing the LQG control. Typical transformations include both the Jordan canonical and the Companion form. While transformation to one of these forms offers the potential for increased speed and reduced memory requirements, it was found that it is not always practical to use this approach. A transformation was used for the fifth order system in an attempt to reduce the number of analog computer components needed. However, when the control and estimation matrices for the

transformed systems were calculated it was found that they contained numbers ranging over eleven orders of magnitude. The large range of values could not be accommodated by the 16 bit word size of the LSI 11/2.

Although it is possible to reduce the number of operations and memory required by state vector transformation there is no guarantee that the result will be amenable to implementation on a (fixed point) microprocessor. The use of such transformations should be with caution. Evaluating a particular microprocessor for implementing LQG control should be done first on the basis of the physical system description. Resort to alternate forms may reduce the apparent number of computations and memory required but the scaling issue discussed above must be considered.

4) The second order system validation using on Intel 8080 microprocessor required 4.7 ms computation time compared with the predicted value of 4.25 ms. The actual memory used was 483 words of which 468 were PROM and 15 were RAM. The predicted values were 490 words including 15 words of RAM and 475 words of PROM.

The fifth order system validation using an LSI 11/2 microprocessor required 14.45 ms computation time compared with the predicted value of 9.68 ms. The discrepancy is due to the additional software required to execute signed multiplication which was not accounted for in the prediction. The actual memory used was 422 words of which 109 words were RAM and 313 words were PROM. The predicted memory requirements were 304 words including 121 words of RAM and 183 words of PROM. The differences between the actual and estimated memory requirements is primarily due to the additional software for signed multiplication.

5) Microprocessor requirements for implementing control and estimation for nonlinear systems were defined. The nonlinear control and estimation structure was developed based on a piecewise linear approach which approximates the nonlinear system by a series of linearized systems evaluated at various operating points of the system.

Optimal linear control and estimation designs are developed for each operating point based on LQG theory. These designs are coupled together by scheduling the control and estimation matrices between operating points using linear interpolation as a function of the state of the nonlinear system.

Microprocessor requirements for implementing the nonlinear configuration indicate the largest impact is in the additional memory required compared to the linear system. The additional memory results from the requirement that control and estimation matrices must be stored for each operating point considered. However, the increased memory is a linear function of the number of operating points. This coupled with the facts that 1) the memory requirements for LQG are small and 2) large capacity memories have been available for some time indicates that the memory requirements for nonlinear implementation are not excessive.

INTRODUCTION

Over the past several years use of modern control methodology -- in particular, linear quadratic Gaussian (LQG) theory -- has gained increased recognition as an effective design tool for control of nonlinear multivariable stochastic systems (Refs. 1-8). The referenced studies have been conducted under a combination of AFOSR, Office of Naval Research (ONR), Air Force Aero Propulsion Laboratory (AFAPL), NASA-Lewis and Pratt & Whitney Aircraft (P&WA) support. In these as well as many other aerospace applications the primary impetus for application of modern LQG control concepts is improved system performance combined with the advent of digital electronic control implementation. Digital electronics provide the means by which complex controllers associated with LQG theory can be implemented. The current trend both within as well as outside the aerospace controls community toward increased use of digital electronics -- in particular, microprocessors -- will lead to increased use of modern control logic including system identification, modeling, estimation, and multivariable control methodologies (Ref. 9). In addition, use of microcomputer controllers will lead, in many instances, to reduced control cost (Refs. 10 and 11), lighter and smaller controls (Ref. 12), lower power requirements and integrated circuit reliability (Ref. 13). Recent studies (Ref. 14) have demonstrated that existing microprocessor can be used to implement algorithms for parameter identification of relatively simple, low-order dynamic systems.

However, prior to widespread use of microprocessors for modern control logic implementation key issues associated with microprocessor implementation of LQG control and estimation concepts must be addressed and resolved. These issues include (1) accuracy, (2) computational capability, (3) memory, and (4) interface requirements (Ref. 15). These requirements depend upon system dynamics as well as upon the particular control algorithm employed. Defining these requirements will establish criteria for selecting the appropriate computer system for control implementation.

To address these important issues a two phase two year program directed toward establishing microprocessor requirements for implementing modern control logic was initiated at UTRC under AFOSR support in 1978. The Phase I study (Ref. 16) was concerned with establishing analytic techniques for evaluating microprocessor requirements for implementing modern control and estimation for linear systems. These techniques were applied to two selected examples, a single input second order system and a four input fifth order system. The implementation requirements for each system were identified and a candidate microprocessor was selected as a suitable device for implementing each of the system control and estimation algorithms.

This report presents results for the Phase II effort of the program which was directed toward verification of the analytic procedures of Phase I. The verification was conducted for both the second order and the fifth order cases analyzed in Phase I. The procedures involved simulating the system dynamics on an EAI/1000 analog computer. The control and estimation dynamics were implemented on an Intel 8080 microprocessor for the second order case and a Digital Equipment Corporation LSI-11/2 microprocessor for the fifth order case. Comparisons are presented between the predicted microprocessor requirements and those realized using the hybrid simulation approach mentioned above.

The Phase I techniques have also been extended to the general class of nonlinear systems. An analysis is presented which treats the nonlinear problem as a series of linear problems by linearizing the dynamics at a set of operating points. The LQG design methodology is applied at each of the operating points to develop a series of piecewise linear optimal control and estimation configurations. The microprocessor requirements for implementing this approximation to the optimal nonlinear solution are presented.

CONTROL OF NONLINEAR STOCHASTIC SYSTEMS

In this section the general nonlinear stochastic regulation problem is presented. The nonlinear stochastic system model is defined first. This model is general in nature and applicable to a broad class of estimation and control problems.

A technique for synthesizing feedback control of the general system is then presented. The approach is to represent the nonlinear system as a set of linear systems defined throughout the operating regime of the nonlinear plant. Linear quadratic Gaussian (LQG) control design is then reviewed as applied to the linearized plant description. A technique for extending the linear design to the nonlinear plant is then presented which uses gain scheduling within the controller. In the final part of this section the digital implementation of the nonlinear and linear controllers are discussed.

System Description

The system model is shown in Fig. 1 with provision for estimation and regulation algorithms included. The system consists of a nonlinear plant, control actuators, and sensors. The control actuators are physical devices which translate commanded inputs into actual plant inputs. This translation is not exact and, therefore, process noise is included to account for actuator uncertainties. This process noise also models external plant disturbances and system-to-system parameter variations. Plant state variables are generated through plant dynamics and the actual inputs. These state variables in conjunction with the actual inputs govern the plant output response. Plant state variables are available only through sensors which contain inherent lags and nonlinearities. These sensors indicate which state variables or combinations thereof can be measured. Sensor noise is included to account for measurement inaccuracies. Modeling actuator and measurement uncertainties by stochastic processes translates these physical uncertainties into mathematically tractable representations. The statistical properties of these random processes define the process and sensor inaccuracies.

System dynamics are given by the differential and algebraic equations

$$\begin{aligned}\dot{x}(t) &= f(x(t), u(t), \xi(t)) \\ y(t) &= g(x(t), u(t)) \\ z(t) &= h(x(t)) + n(t)\end{aligned}\tag{1}$$

where x represents the n -dimensional system state vector, u represents the m -dimensional actuator input vector, y represents the p -dimensional plant output vector, and z represents the l -dimensional measurement vector. The random process vectors ξ and η represent white zero-mean Gaussian m -dimensional actuator and l -dimensional sensor noise, respectively. The dot notation denotes differentiation with respect to time. The vector functions f , g , and h are assumed continuous and twice differentiable in all their arguments. Note that f includes all dynamics associated with the plant, actuators and sensors. The initial state vector is assumed to be a Gaussian random variable with known mean. The random vectors $x(0)$, $\xi(t)$, and $\eta(t)$ are assumed independent with known covariances. The statistics of the system uncertainties are defined by

$$\begin{aligned}
 E\{x(0)\} &\triangleq \bar{x}(0) \\
 E\{(x(0)-\bar{x}(0))(x(0)-\bar{x}(0))'\} &\triangleq S(0) \\
 E\{\xi(t)\} &= 0 \\
 E\{\xi(t)\xi'(\tau)\} &= Q(t)\delta(t-\tau) \\
 E\{\eta(t)\} &= 0 \\
 E\{\eta(t)\eta'(\tau)\} &= R(t)\delta(t-\tau) \\
 E\{(x(0)-\bar{x}(0))\xi'(t)\} &= 0 \\
 E\{(x(0)-\bar{x}(0))\eta'(t)\} &= 0 \\
 E\{\xi(t)\eta'(\tau)\} &= 0
 \end{aligned} \tag{2}$$

where $\delta(t-\tau)$ is the Dirac delta function and the prime denotes a transpose.

Control Design Approach

The controller must generate inputs to the actuators to achieve desired system performance as determined by actual plant state and output response. The time response of actual plant variables, rather than measured variables, is therefore the key quantity that enters into an assessment of system performance. The controller must determine time evolution of the actuator inputs--the only system variables which can be directly adjusted--to satisfactorily control time evolution of actual plant state and output variables.

The nonlinear stochastic feedback regulator design depends on (1) the dynamics of the system, (2) the levels of uncertainty in the system, and (3) performance criteria that specify satisfactory time evolution of the system inputs and outputs. The design issue is complicated by the interplay between system dynamics, the stochastic nature of the problem, and the effects of deterministic commanded inputs. However, a design philosophy that separates the deterministic and stochastic aspects of the problem can be adopted.

The design approach involves first linearizing the system dynamics of Eq. 1, applying linear quadratic regulator (LQR) control techniques to the linear deterministic description, developing estimator equations for the stochastic portion and finally developing the nonlinear control by combining the control and estimation equations.

The separation theorem (Ref. 17) allows optimal solution of the control and filter problems separately for linear systems. In general, if the over-all nonlinear quadratic stochastic control problem could be solved, the resulting optimal design would not obey the separation property. Since at the present time the combined optimal nonlinear estimation and control problem cannot be solved, the separation concept and a linear quadratic Gaussian approach is employed to arrive at a set of related problems that can be solved.

Linearized System Description

For steady-state regulation, the controller must maintain the actual plant variables as close as possible to the steady-state operating point in the presence of plant disturbances. In these small-signal situations, the nonlinear system of Eq. (1) can be described by a linearized perturbational model which approximates the dynamic behavior of the nonlinear system in a small region about the steady-state operating point. Linear dynamics are determined by expanding the nonlinear system functions f , g , and h (Eq. (1)) in a Taylor series expansion about the steady-state operating point (x_{ss}). Retaining only first-order terms in the Taylor series expansion results in the perturbational equations of the system dynamics.

$$\begin{aligned}\delta\dot{x}(t) &= A\delta x(t) + B \delta u(t) + \xi(t) \\ \delta y(t) &= C\delta x(t) + D \delta u(t) \\ \delta z(t) &= E\delta x(t) + \eta(t)\end{aligned}\tag{3}$$

where δx , δu , δy , and δz represent state, input, output and measurement perturbations, respectively, and statistical properties are as previously defined for Eq. (1). The A ($n \times n$), B ($n \times m$), C ($p \times n$), D ($p \times m$) and E ($l \times n$) matrices are given by

$$A = \frac{\partial f}{\partial x}, B = \frac{\partial f}{\partial u}, C = \frac{\partial g}{\partial x}, D = \frac{\partial g}{\partial u}, E = \frac{\partial h}{\partial x} \quad (4)$$

evaluated at the point $x = x_{ss}$.

Deterministic Control

A systematic technique for deterministic multivariable nonlinear system control design based on linear quadratic regulator theory -- specifically, the piecewise-linear/piecewise-optimal (PLPO) control technique -- was developed (Ref. 2). The deterministic control design procedure assumes no uncertainties, i.e., it is assumed that (1) no actuator errors exist, (2) no plant disturbances occur, (3) all state and output variables are measured perfectly, and (4) actuator and plant dynamics and parameters are known exactly. Under these assumptions, plant state and output variables can be determined for any given commanded inputs.

The analytical PLPO method is based on linearizing the system about a set of closely spaced steady-state operating points and applying linear optimization methods at each point. A single nonlinear control problem is thereby reduced to a series of linear control problems. This permits the use of established analytical and numerical methods associated with linear quadratic control theory. At each operating point, an optimal linear feedback controller is generated by minimizing a quadratic performance criterion. Weighting factors within each performance criterion enable the control designer to satisfy performance specifications by trading-off system response against control actuation rates. Nonlinear feedback control is then constructed by combining the series of linear controllers into a single nonlinear controller whose feedback gains vary with system state.

LQR theory applied at any operating point given an optimal incremental control for the linearized system dynamics described by Eq. 3. The general form of this control is

$$\delta u^* = G(x_{ss}) \delta x \quad (5)$$

where $G(m \times m)$ represents the optimum feedback gain matrix for operation at the steady-state point x_{ss} . The PLPO technique applied to Eq. 5 gives the algorithm for implementation the nonlinear control:

$$u^* = \int_{x(0)}^{x(t)} G(x(\tau)) dx(\tau) + u(0) \quad (6)$$

Since the controller gain matrix G is defined at a series of design points along the system steady-state operating line, on-line interpolation is used to determine values for G between the operating points.

State Estimation Design

The stochastic aspects of the problem are reintroduced for the estimation portion of control system design. In addition to plant, actuator and model uncertainties, the fact that all state variables cannot be measured and that any measurement is subject to sensor errors must be taken into account. The objective of this step is to design an estimator or filter that generates, on the basis of past and present sensor measurements, estimated plant state and output variables as close as possible to actual plant state and output variables at any instant of time.

Linear filtering theory may be applied to nonlinear systems by continually updating a linearization around the current state estimates. The resulting estimation algorithm is the extended Kalman filter. The system dynamics for the extended Kalman filter are represented by the nonlinear deterministic system equations. The estimated state variables $\hat{x}(t)$ and output variables $\hat{y}(t)$ are generated by the nonlinear differential and algebraic equations

$$\begin{aligned} \dot{\hat{x}}(t) &= f(\hat{x}(t), u(t), 0) + K(t) (z(t) - h(\hat{x}(t))) \\ \hat{y}(t) &= g(\hat{x}(t), u(t), 0) \end{aligned} \quad (7)$$

where the $n \times 1$ Kalman gain matrix $K(t)$ depends on (1) the partial derivatives $\partial f/\partial x$ and $\partial h/\partial x$ evaluated with respect to the estimated states $\hat{x}(t)$, and (2) the sensor and driving noise statistics. Extended Kalman filtering theory calls for the matrix $K(t)$ to be calculated in real time since it is coupled to the current state estimates through the relinearization procedure. This on-line gain calculation often results in filter divergence (Ref. 18) due primarily to (1) on-line linear system approximations required for on-line gain calculations, (2) model-mismatch between the filter model and the actual system, and (3) mismatch between actual system noise statistics and those statistics assumed in calculating the gains. In addition, $\frac{n(n+1)}{2}$ differential

equations must be solved to calculate the gain matrix $H(t)$. The derivation of the extended Kalman filter is presented in Ref. 17.

In Ref. 5 techniques were developed for large-signal filtering logic with off-line gain calculation to (1) avoid the divergence problems associated with on-line gain calculation, and (2) reduce the computational complexity of the filtering logic. The filtering logic was defined based on representing the system by reduced-order models to further reduce the computational complexity of the estimation of algorithms. In addition, model-mismatch compensation techniques were established to eliminate bias errors due to modeling inaccuracies, system-to-system variations, and system degradation. Results obtained in the Ref. 3 UTRC study directed toward stochastic small-signal regulation of nonlinear multivariable dynamic systems indicate that Kalman filtering methodology with model-mismatch compensation is an effective means for achieving accurate estimation. In addition, the Ref. 3 study showed that improved estimation leads to improved stochastic regulation.

Off-line Kalman gain calculation is based on the linearized system dynamical description of Eq. 3. For this problem the solution to the state estimation problem found in Ref. 17 was used. The Kalman filter dynamics are described by

$$\begin{aligned}\dot{\delta\hat{x}}(t) &= A\delta\hat{x}(t) + B\delta u(t) + K(z(t) - E\delta\hat{x}(t)) \\ \delta\hat{y}(t) &= C\delta\hat{x}(t) + D\delta u(t)\end{aligned}\tag{8}$$

The $n \times l$ constant gain matrix K is a function of the known A and B matrices and the specified system noise statistics.

Model-Mismatch Compensation

The linear perturbational model (Eq. (3)) represents an approximate relationship between state, input, output and measurement perturbations because second- and higher-order terms were neglected in the Taylor series expansion. Derivation of the Kalman gains, however, assumes that Eq. (3) represents an exact model of the physical process. To compensate for this inherent model-mismatch an $n \times l$ vector e , which represents the resulting error between actual and estimated state perturbations, is defined. This definition leads to a second linear estimation problem -- i.e., estimation of the error vector e -- described by

$$\begin{aligned}\dot{e}(t) &= (-KE+A)e(t) - K\eta(t) + B\xi(t) + v(t) \\ \dot{v}(t) &\stackrel{\Delta}{=} \gamma(t) \\ r(t) &= Ee(t) + \eta(t) \\ r(t) &\stackrel{\Delta}{=} \delta z(t) - E\delta\hat{x}(t)\end{aligned}\tag{9}$$

where the n -dimensional vector γ represents white zero-mean Gaussian n -dimension model-mismatch uncertainty with intensity N ; i.e., $E \gamma(t)\gamma'(t) = N(t)\delta(t-\tau)$. The designer selects the model-mismatch matrix N to reflect uncertainty about the model dynamics; i.e., the larger the intensity matrix N the more uncertain the designer is that the system model is the same as the physical process. The Kalman filter for the system described by Eq. (9) is given by

$$\begin{aligned}\dot{\hat{e}}(t) &= (-KE+A)\hat{e}(t) + \hat{v}(t) + K_1(r(t)-E\hat{e}(t)) \\ \dot{\hat{v}}(t) &= K_2(r(t)-E\hat{e}(t))\end{aligned}\quad (10)$$

where the $n \times l$ and K_2 matrices are upper and lower partitions, respectively, of the $2n \times l$ Kalman filter gain matrix. An improved perturbational state estimate is obtained by adding the estimated error to the original perturbational state estimate. After algebraic manipulation -- which alters the Kalman gains and isolates the compensator gains -- the estimator is described by

$$\dot{\delta\hat{x}}(t) = A\delta\hat{x}(t) + B\delta u(t) + K_f(\delta z(t)-E\delta\hat{x}(t)) + \int_0^t (K_c(\delta z(\tau)-E\delta\hat{x}(\tau)))d\tau \quad (11)$$

where the n -dimensional vector $\hat{\delta}$ is the improved perturbational state estimate and

$$\begin{aligned}K_f &= K + K_1 \\ K_c &= K_2.\end{aligned}\quad (12)$$

The derivation of Eq. (11) is presented in Ref. 19.

Integrating Eq. (11) along the system trajectory leads to

$$\dot{\hat{x}}(t) = f(\hat{x}(t), u(t), 0) + K_f(z(t)-h(\hat{x}(t))) + \int_0^t K_c(z(\tau)-h(\hat{x}(\tau)))d\tau \quad (13)$$

where

$$\begin{aligned}f(\hat{x}(t), u(t), 0) &= \int_0^t (A\delta\hat{x}(\tau) + B\delta u(\tau))\delta\tau \\ h(\hat{x}(t)) &= \int_0^t E\delta\hat{x}(\tau)\delta\tau\end{aligned}\quad (14)$$

and f , h represent the system model employed in the filter. The Kalman and compensator gains K_f and K_c are functions of the operating condition. The block diagram of this filter is shown in Fig. 2. The implementation requirements for this nonlinear filter depend on the form assumed for the deterministic system models f and h .

The PLPO techniques described above and used to generate the control law of Eq. 6 are now applied to the filter equation. The procedure requires formulating expressions for the two functions shown in Eq. 14. Path integrals of the linearized system dynamics (Eq. (3)) at a series of operating points define the piecewise-linear model. From Eq. (3) the piecewise-linear model dynamics are given by

$$f(\hat{x}(t), u(t), 0) = \int_{\hat{x}(0)}^{\hat{x}(t)} A(\hat{x}_j) d\hat{x} + \int_{u(0)}^{u(t)} B(\hat{x}_j) du \quad (15)$$

$$h(\hat{x}(t)) = \int_{\hat{x}(0)}^{\hat{x}(t)} E(\hat{x}_j) d\hat{x} + h(\hat{x}(0)) .$$

Since the A, B, C, D, and E matrices are defined at a series of design points along the system steady-state operating line, interpolation based on a selected state x_j is used to determine values for these matrices between design points. The filter gains K_f and K_c are also functions of operating condition.

Combined Estimation and Control

The third and final step in control synthesis for stochastic nonlinear systems using the separation approach would involve combining the PLPO deterministic control and stochastic estimation algorithm developed in this study into a unified feedback controller. The resulting stochastic controller must estimate the system states and outputs from the noise-corrupted system measurement data. Based on these estimated system variables, the control must generate actuator inputs to achieve satisfactory system performance. The overall system structure is shown in Fig. 1.

Summarizing the results of the control and filter design outlined above produces the combined nonlinear equation to be implemented

$$u(t) = \int_{\hat{x}(0)}^{\hat{x}(t)} G(\hat{x}(\tau)) d\hat{x}(\tau) + u(0) \quad (16)$$

$$\dot{\hat{x}}(t) = f(\hat{x}(t), u(t), 0) + K_f(z(t) - h(\hat{x}(t))) \quad (17)$$

$$+ \int_0^t K_c(z(\tau) - h(\hat{x}(\tau))) d\tau$$

where f , h are defined in Eq. 15 for the piecewise linear model.

Digital Implementation

The nonlinear equations to be implemented in the microprocessor are given by Eqs. 16 and 17. The implementation is based on using rectangular (Euler) integration which is the simplest form of integration in terms of the computations required per sample interval.

To code the estimation algorithms on a digital computer the filter equations for state estimation may be represented by (1) state prediction equations, and (2) state update equations. Filter equations based on Euler integration which predict state variables at time $t + \Delta t$, given measurements to time t , are described by

$$\hat{x}(t + \Delta t/t) = \hat{x}(t/t) + f(\hat{x}(t/t), u(t), 0) \Delta t \quad (18)$$

where Δt represents the known sampling interval. If a more accurate integration method (e.g., Runge-Kutta) is employed, the sampling interval may be increased; however, the computational requirements will also be increased. The notation $(t + \Delta t/t)$ represents filter prediction of system states at time $t + \Delta t$ given measurements to time t . The filter update equations are given by

$$\hat{x}(t + \Delta t/t + \Delta t) = \hat{x}(t + \Delta t/t) + (K_f \Delta t) r(t + \Delta t) + \sum_{\tau=0}^{t+\Delta t} (K_c \Delta t^2) r(\tau) \quad (19)$$

$$r(t + \Delta t) = z(t + \Delta t) - h(\hat{x}(t + \Delta t/t)).$$

Equations (18) and (19) indicate that the state prediction computational requirements are primarily dependent on the system model in the filter; whereas, the state update computational requirements are primarily dependent on the gain calculation. Output and measurement estimate computational requirements are primarily dependent on the output and measurement models employed in the filtering algorithm. Therefore, computational requirements for the filtering algorithms are functions of (1) the system model employed in the filter, and (2) filter gain calculation. Digital implementation of the control equation follows from Eq. 16 again assuming Euler integration

$$u(t + \Delta t) = u(t) + G(\hat{x}_j)(\hat{x}(t + \Delta t/t + \Delta t) - \hat{x}(t/t)) \quad (20)$$

Equation 20 indicates that the control computational requirements are primarily dependent on the gain calculation. All gain scheduling is assumed to be implemented using linear univariate interpolation. For any of the matrices contained in Eqs. 18 thru 20 the algorithm for gain calculation is

$$P = m_1 (\hat{x}_j - (x_j)_i) + P_i \quad (21)$$

$$m_1 = \frac{P_{i+1} - P_i}{(x_j)_{i+1} - (x_j)_i}$$

where p denotes any matrix parameter, and the index i represents a steady-state operating condition such that $(x_j)_i \leq \hat{x}_j \leq (x_j)_{i+1}$.

Microprocessor Requirements

The requirements placed on a single microprocessor to implement the non-linear control and estimation equation are discussed in this section. This study did not include an investigation of partitioning the computational tasks among two or more microprocessors although this option is a potentially powerful alternative.

The microprocessor requirements which are discussed include (1) number of computations required per sample interval and (2) memory requirements necessary for storing the gains and the sampled state values. The methodology closely follows that developed in Ref. 16. The number of computations required per sample interval determine the time to execute the control and filter equation. This time must be less than the sample time which is used in the closed loop design. The sample time in turn must be sufficiently fast to guarantee that (1) the dynamics of the physical plant are adequately represented by the sampled values and (2) the digital implementation of Eqs. 18 through 20 is stable. A technique for examining the stability of the equation was presented in Ref. 16. This technique can be used to study the stability of these different equations for a particular system description. Guaranteeing that the sample update is fast enough to adequately represent the nonlinear dynamics of the physical system requires that detailed simulation be used. For linear systems the evaluation can be achieved without recourse to detailed simulation by considering the eigenvalues of the system. The maximum sample time in this case can be determined by application of the sampling theorem.

Once the maximum sample time is determined the applicability of a particular microprocessor can be evaluated. This evaluation requires first determining the number of computations required per sample interval. These computations consist of signed multiplication and signed addition as well as delay associated with the input and output operations.

Computational Requirements

The computational requirements for signed multiplication and signed addition are determined from Eqs. 18 through 21. This determination involves the systematic analysis of these equations in terms of the dimensions of the dynamic variables and matrices.

In Phase I it was shown that the linear discrete controller implementation requirements could be altered by transforming the estimated state vector. To change state coordinates for the linear system the estimated state vector \hat{x} is transformed through the equation

$$\hat{w} = T^{-1} \hat{x} \quad (22)$$

where T is an $n \times n$ nonsingular constant matrix. This technique was used to transform the system description to both Jordan canonical form and the Companion form. The digital implementation requirements were then evaluated for the linear system in the Standard, Jordan, and Companion forms.

For the transformation technique to be applicable to the nonlinear case requires that a T matrix be stored in the microprocessor as a function of operating point. This requirement follows from the fact that the linearized system dynamical description of Eq. 3 is a function of operating point.

Therefore, for nonlinear implementation additional memory is required if the transformation technique is to be used. The approach taken in developing the implementation requirements for the nonlinear system is to consider only the standard form of the system. This avoids the necessity to store the T matrix as a function of operating point.

Filter computational requirements per sampling interval are determined from the prediction and update relationship of Eqs. 18 and 19. The gain matrices K_f and K_c are scheduled according to the univariate interpolation algorithm of Eq. 21. The computational operations for implementing the control law of Eq. 20 consist of matrix/vector multiplication and vector addition as well as the interpolation required by Eq. 21 for scheduling the matrix G between operating points.

In addition to the operations associated with the filter and control equation there are operations required for input and output (I/O) between sample intervals. These operations consist of measuring l outputs of the plant and providing n inputs to the plant. These I/O operations are used for both the filter and control equations. Table I summarizes the computational operations required for each sample interval of the PLPO implementation. They consist of multiplications, additions, and I/O operations. These operations may be used to predict the computation time required by the microprocessor.

Computational times are evaluated for any candidate microprocessor once the benchmark times for signed multiplication and signed addition are determined. The I/O times are dependent on the characteristics of the A/D and D/A converters used. The characteristics of a representative set of microprocessor and A/D and D/A converters are shown in Ref. 16 and repeated in Appendix A.

Memory Requirements

Memory requirements depend upon (1) the system model and (2) the computer code including temporary storage to implement the control and filter algorithms. System model memory requirements are a function of model structure as well as

system state, input, and output orders. System model requirements will not vary with microprocessor. On the other hand, the computer code and temporary storage requirements will vary with microprocessor as well as system model.

Memory may be either random access memory (RAM) or programmable read only memory (PROM). RAM is generally used to store temporary data such as measurements and intermediate calculations. PROM would be used for storing constants necessary for implementing the control and filter equations. Memory requirements are determined from the filter and control relationships of Eqs. 18 through 20 and the interpolation algorithm of Eq. 21. The results are summarized in Table II where the type of memory, either RAM or PROM is also indicated. For the scheduled gain matrices, G , K_f and K_c the total number of storage locations depends on the number of operating points used in designing the PLPO structure. In Table II, K represents the number of steady-state operating points selected for system linearization.

LINEAR SYSTEM RESULTS

The Phase I work reported in Ref. 16 dealt with the microprocessor requirements for implementation of modern control for linear systems. A major task conducted under the present research was to verify the analysis and prediction of microprocessor requirements for linear systems. The validation was aimed at verifying the prediction for two problems. One problem was a single input second order system and one was a four input five state system representing linearized F100 engine dynamics. The second order validation consisted of implementing the modern control on an Intel 8080 microprocessor. An analog simulation of the second order system was interfaced to the microprocessor. The control for the four input fifth order system was implemented on a Digital Equipment Corporation LSI 11/2 microprocessor. The linearized F100 engine dynamics were simulated on an Electronics Associates Incorporated Model 1000 analog computer.

This section reviews the microprocessor requirements for implementing modern control for linear systems. The validation results are presented by first discussing the matrix and vector mathematical algorithms used in the microprocessor. Results from this implementation phase are then presented and compared with the predictions.

Microprocessor Requirements-Linear Systems

For linear systems the matrices of Eq. 3 are constant and the δ notation does not apply. Therefore the filter and control dynamics can be represented by

$$\begin{aligned}\dot{\hat{x}}(t) &= F\hat{x}(t) + H(z(t) - E\hat{x}(t)) \\ u^*(t) &= G\hat{x}(t) \\ F &\triangleq A + BG\end{aligned}\tag{23}$$

where the notation $(\hat{\quad})$ denotes the estimate of the variable in parentheses, G ($m \times n$) represents the deterministic feedback control gain matrix and H ($n \times l$) represents the steady-state Kalman filter gain matrix.

The formulation developed in Phase I included provision for transforming the coordinate system using a transformation matrix T . The reason for the transformation is to represent the system by a new state vector

$$\hat{w} = T^{-1} \hat{x}\tag{24}$$

whose dynamic equation may be of a form to reduce the number of computations required in implementing the filter and control equation. The transformed equations, from Eqs. 23 and 24 are

$$\begin{aligned}\dot{\hat{w}}(t) &= F_T \hat{w}(t) + H_T (z(t) - E_T \hat{w}(t)) \\ u(t) &= G_T \hat{w}(t)\end{aligned}\tag{25}$$

where $F_T = T^{-1}(F)T$, $H_T = T^{-1}H$, $E_T = ET$ and $G_T = GT$. However, the matrix F_T depends upon the selected transformation matrix T . Note that $T=I$ results in the standard form, i.e., $F_T=F$.

Implementing the control and estimation dynamics of Eq. 25 is done by solving the filter equation in two steps as discussed earlier. This process involves solving (1) state prediction equations and (2) state update equations. The result may be expressed in the following form

$$\begin{aligned}\hat{w}(k+1) &= \phi_D \hat{w}(k) + H_D z(k+1) \\ u(k+1) &= G_D \hat{w}(k+1)\end{aligned}\quad (26)$$

where

$$\begin{aligned}\phi_D &= (I - T^{-1} HET\Delta t) (I + \phi) \\ \phi &= T^{-1} FT\Delta t + \frac{(T^{-1} FT)^2 \Delta t^2}{2!} + \frac{(T^{-1} FT)^3 \Delta t^3}{3!} + \dots \\ H_D &= T^{-1} H\Delta t \\ G_D &= GT\end{aligned}\quad (27)$$

and k denotes the k^{th} sample time.

The matrices of Eq. 27 are computed off-line and stored in the microprocessor for use in implementing the filter and control relationships of Eq. 26.

Analytic techniques were established in Phase I for predicting the microprocessor requirements for implementing Eq. 26. The specific requirements addressed included (1) accuracy (word length) requirements, (2) computational requirements, and (3) memory requirements.

Accuracy Requirements

The effect of finite word length on the overall response of the controlled system was evaluated using a performance index approach. A Univac 1100 series digital computer with a 36 bit word length was used to generate system response against which the responses for smaller word length configuration could be compared. The performance index

$$J = \int_0^{\infty} [(y^* - y^t)' Q (y^* - y^t) + (u^* - u^t)' R (u^* - u^t)] dt \quad (28)$$

where y^* = output response vector with 36-bit controller
 y^t = output response vector with b -bit controller
 u^* = control vector with 36-bit controller
 u^t = control vector with b -bit controller
 Q, R = weighting matrices

was defined. The performance index J represents performance degradation due to finite word lengths less than the accurate 36-bit word length. As the number of bits in the computer word approaches 36, J approaches zero.

A procedure was developed for reducing this integral equation to an algebraic equation. The resulting expression allows the numerical evaluation of Eq. 28 to be performed very simply on the computer without the need to simulate the actual system, control, and filter dynamics.

Computational Requirements

The computational requirements for implementing Eq. 26 were expressed in terms of the number of multiplications, and additions required as well as the I/O operations required for each sample interval. Three cases were considered corresponding to three transformation matrices T , defined in Eq. 25. The three structures were (1) Standard, (2) Jordan canonical, and (3) Companion. Table III summarizes the results for the number of operations required for the LQG implementation.

Memory Requirements

Total storage requirements were defined for the filter and control dynamics represented by Eq. 26. The memory requirements resulted from analysis of the dimension of the vectors and matrices of Eq. 26. The storage was required for (1) past state estimates, $w(k)$, (2) current state estimates, $w(k+1)$, (3) measurements $z(k+1)$, (4) control $u(k+1)$, and the gain (G_D) and filter matrices (ϕ_D, H_D) of Eq. 26.

The memory requirements for implementing LQG control and estimation are summarized in Table IV for the three system structures discussed above.

APPLICATION AND VERIFICATION OF MICROPROCESSOR
REQUIREMENT PROCEDURES

The procedures reviewed in the previous section were applied to two candidate systems and the microprocessor requirements were predicted. The systems selected for examination were (1) a single input second order plant and (2) a four input fifth order F100 turbofan engine linearized at sea level static military operation. The verification of the procedures was carried out for both of the selected examples. This verification consisted of (1) simulating the system dynamics on an analog computer, (2) implementing (coding) the control and filter equation in a microprocessor, and (3) comparing the results with those predicted using the procedures discussed above.

Before discussing the results of the validation experiments it is important to describe the matrix and vector operations which are required to implement the control and filter dynamics of Eq. 26. The operations required i.e., vector/matrix multiplication and vector addition are generic to the structure of the problem. Any efficiencies which can be realized in performing these operations will translate directly into reduced computational time.

A block diagram of the complete simulation system is shown in Fig. 3. The system dynamics are simulated on an analog computer. Control and filter dynamics are implemented in a microprocessor which is interfaced to the analog computer using A/D and D/A converters as shown.

The software to implement the control and filter dynamics consists of three matrix/vector multiplications ($\Phi_D w$, $H_D z$, and $G_D w$) and one vector addition ($\Phi_D w + H_D z$). Other operations are required to save past state estimates, service the clock, interrupt and output the control via the D/A converter. The clock interrupt service routine (1) performs a timing check to assure that all control computations are completed within the sample time and (2) reads in measurement data via the A/D converter.

The overall block diagram of the control software described above is shown in Fig. 4. The block diagram of the matrix/vector multiplication code is displayed in Fig. 5. Figure 5 includes several minor changes (e.g., the order in which pointers are initialized and updated) which were made in the matrix/vector multiplication block diagram presented in the Phase I report. These changes result in more efficient microprocessor implementation. The matrix/vector multiplication algorithm is not changed but rather the way the algorithm is implemented has been slightly modified. Block diagrams of the vector addition code, the store state estimates, and the interrupt service routine are shown in Fig. 6.

Second Order System

The second order system dynamics in the form of Eq. 3 are represented by the A, B, C, D and E matrices of Table V. The control and filter dynamics for the second order system are in the form of Eq. 26. The constant matrices G_D and H_D are also shown in Table V.

The second order system control and filter equations were coded on an Intel 8080 microprocessor with software multiply. The second order system dynamics were implemented on a special purpose analog computer.

The improvements discussed above in the matrix/vector multiplication algorithms were incorporated into the 8080 code. These improvements result in reduced execution time. The code changes include (1) more efficient use of the registers in the multiplication algorithm (11.5% reduction in cycle time) and (2) more efficient memory (data array) accessing as well as more efficient use of the registers in the matrix/vector multiplication algorithm (reduction in cycle time dependent on system order, e.g., a 24% reduction in the matrix/vector multiplication control algorithm is obtained for a 2×2 matrix times a 2×1 vector). In addition, the interface software was added to the preliminary code. A complete listing of the code is shown in Appendix B.

Comparison of Results

The analytic procedures developed in Ref. 16 were applied to the second order system. Conclusions resulting from applying these prediction techniques were (1) an 8 bit word length is sufficient, (2) a minimum sample time based on the number of computations of 4.25 ms was predicted for the 8080, and (3) 490 words of memory would be required.

The validation of the prediction techniques consisted of (1) comparing the execution times and memory requirements between the predicted values and those resulting from the actual implementation and (2) comparing dynamic response between that predicted and that actually achieved. This dynamic response comparison was used to verify the accuracy requirement predictions.

Table VI summarizes the predicted values for computation time and memory requirements and also shows the computation time and memory actually achieved with the implemented system. The minimum predicted sample time of 4.25 ms compares well with the measured value of 4.7 ms.

The actual memory requirements are also in good agreement with the predicted values. Actual memory used was 483 words of which 468 were PROM and 15 were RAM. The predicted values were 490 words including 15 words of RAM and 475 words of PROM.

Validation of the word length requirements shows that the prediction techniques are quite accurate. This is important since the word length prediction was treated using an analytic formulation based on the cost function approach described above. This allows the designer to simply solve an algebraic matrix equation to assess the effect of finite word length. Application of this technique to the second order system showed that an 8 bit microprocessor was adequate for implementing the second order control and filter equations. An 8 bit microprocessor was simulated as described in Ref. 16 and the resulting output response was recorded. Figure 7 shows this predicted response and also shows the response recorded using the analog computer/8080 system. The good agreement between these results indicate that the prediction techniques are valid for assessing the effect of word length on system response.

Fifth Order System

Validation of the prediction techniques was also carried out for a four input fifth order linear system. The system dynamics represent the linearized description of an F100 turbofan engine operating at sea level static military operating conditions. The linearized engine dynamic description, in the form of Eq. 3 are represented by the A, B, C, D and E matrices of Table VII.

Simulating these dynamics on the EAI 1000 analog computer requires implementing the A and B matrices of Table VII. The number of amplifiers and potentiometers required for instrumenting the matrices as shown exceeded the number available on the EAI 1000. The large number of analog computer components required is a result of the A and B matrices being in standard form i.e., all elements of the matrices were nonzero. An attempt was made to transform the system description to both the Jordan canonical form and the Companion form by using the appropriate T matrix transformation as described in Eq. 24. This approach did not work, however, since the resultant closed loop F_T matrix of Eq. 24 became ill conditioned. That is, the magnitude of the elements of F_T varied over a wide range. This wide range of values in turn required more than 16 bits of accuracy to allow stable closed loop operation. The number of analog components required was reduced by eliminating several of the smallest terms in the A and B matrices and approximating other terms. The resultant A and B matrices are shown in Table VIII. These matrices are compatible with the number of components available on the EAI 1000. Since the control (G_D) and filter matrices (H_D, ϕ_D) of Eq. 26 are dependent on the A and B matrices, new values for these matrices were computed for the modified A and B matrices. The matrix values for $G_D, H_D,$ and ϕ_D are shown in Table IX. These values are those used in the microprocessor implementation.

The modified system dynamics were required to allow analog computer implementation. However, to verify that the changes did not significantly affect the dynamic description of the system a comparison was made between the responses

for the original A and B matrices and modified versions. Figure 8 depicts the close agreement between the two cases for the x_1 state (incremental fan turbine inlet temperature) response. Other results which were obtained show similar agreement and it was concluded that changing the A and B matrices did not noticeably alter the closed loop dynamics.

Prior to implementing the closed loop experiment the analog simulation was checked against the expected response to verify that the analog representation was correct. This was done by comparing the unforced ($u=0$) response of the analog system against a digital computer simulation. This procedure was of considerable value in uncovering and correcting wiring errors on the analog computer. The analog computer implementation is shown in Fig. 9. The analog simulation of the unforced system compares well with the digital simulation. Figure 10 compares the x_5 state (incremental after burner pressure) responses.

Microprocessor Implementation

Based on application of the prediction techniques it was decided to use an LSI-11 16 bit microprocessor with hardware multiply and divide options. The matrix/vector operation, the interrupt service routines and the state store algorithm discussed previously and shown by the flowcharts of Figs 4, 5, and 6 were coded on the LSI-11. Several changes were incorporated into the final version of the code compared to the preliminary code developed during Phase I. Three changes were in the areas of (1) maximizing the use of registers to reduce the memory access times and (2) improving the techniques for table accessing. Additionally, since the hardware multiply and divide option of the LSI-11 are for unsigned operations, code was developed to enable the handling of signed operations. The complete code for the LSI-11 implementation is shown in Appendix C.

Comparison of Results

Comparisons are presented for (1) accuracy requirements, (2) computational speed, and (3) memory requirements. In each case the comparison is made between the results predicted using the techniques of Phase I and the results obtained from the actual implementation. Application of the prediction techniques indicate that the fifth order system can be implemented for the standard structure of the system using (1) 16 bit word length, (2) a minimum sample time of 9.68 ms based on the computational requirements and (3) 304 words of memory (183 words of PROM and 121 words of RAM).

The predicted results for a simulated 16 bit word length are compared against results from the EAI 1000/LSI-11 system. The predicted results were obtained by using the Phase I simulation techniques and the modified system dynamics discussed above.

Accuracy requirements are compared as in the second order case by examining the dynamic responses. Figure 11 shows the predicted response of the F100 engine model perturbational afterburner pressure response (x_5) and the response obtained from the hardware implementation. Additional comparisons are shown in Figs. 12 through 14. Figure 12 compares the small signal fan speed response and Fig. 13 shows the small signal fan turbine inlet temperature response. Figure 14 compares the compressor variable vane (u_3) control. The agreement between the predicted and actual responses is good considering that the voltage levels of the responses on the analog computer were 100 mV and less. These low levels were required to keep the maximum voltage levels in the analog simulation below the 5 volt power supply limit of the computer. These low voltage levels result in a lower signal to noise ratio at the outputs of the A/D converters than would be possible with higher signal levels. This noise represents measurement noise analogous to the $n(t)$ term in the definition of the system dynamics of Eq. 3. The optimal estimator gains calculated for the fifth order system were predicted on a noise covariance of 0.01. The actual noise present in the analog simulation was beyond the control of the experiment and therefore the precalculated gains are not optimal for the analog system noise levels. With this proviso, however, it can be concluded that the agreement between the predicted responses and the actual responses is good. This validates the conclusion that a 16 bit word is adequate for the fifth order system and indicates that the accuracy prediction techniques developed in Phase I are applicable.

Computational requirements as predicted by the Phase I techniques were 9.68 ms to perform the control and filter calculations. This time consisted of 8.23 ms for arithmetic calculation and 1.45 ms for input and output operations. The computation time for the experiment was determined using a frequency counter. The time required to execute the code was 14.45 ms and the input and output operations required 1.32 ms. The total time between samples required by the system was therefore 15.77 ms. The large discrepancy (15.77 ms vs 9.68 ms) is due to the nature of the hardware multiply instruction. The Phase I estimate of the time required for a multiply was 40.5 μ s. However, this figure was in error. The actual time required in the LSI-11 for a hardware multiply ranges from a low value of 37.0 μ s to a maximum of 68.0 μ s. The exact value for a multiply depends on the value of the two numbers being multiplied. In addition, the Phase I prediction did not account for the additional software required to achieve a signed multiplication. This added software increases the multiply time to 161 μ s on the LSI-11. Since there are 70 multipliers required per sample interval the Phase I estimate was in error by 8.43 ms which is the difference between the actual multiply time of 161 μ s and the Phase I time of 40.5 μ s times the 70 multiplication required. Using the correct value of 161 μ s and applying the Phase I prediction techniques results in a total time of 16.6 ms. This figures compares well with the measured time of 15.77 ms.

Table X summarizes the comparison between the predicted computation times and those determined in the equipment. This table shows the originally predicted times which did not account for the additional multiply software and the values predicted after the additional software was involved in the analysis. Table X also shows the predicted and actual memory requirements for the fifth order system.

Memory requirements as predicted using the Phase I methods were 304 16 bit words of which 121 words would be RAM and 183 would be PROM. The actual LSI-11 memory requirements are 422 words of which 109 words are RAM and 313 words are PROM. The reduction in the number of RAM words required (109 compared to 121) is a result of the efficient use of registers for temporary storage. The increase in PROM in the experiment (313 compared to 183) was due primarily to the software required for implementing the signed multiply and divide operations (46 words required) and the scaling routines necessary to interface with the D/A converters (55 words).

Table X shows the memory requirements using the original prediction techniques, updated prediction techniques and those actually required. The original prediction values were updated to include the additional memory required to store the code for implementing the signed multiply and divide routines (46 words) and the scaling software required (55 words) to interface to the D/A converters. Both of these operations are in the category of program code rather than storage required for the basic LQG structure. These additional requirements result from the specific characteristics of the microprocessor used. In general there is no effective way to predict the additional memory (or computation time) without detailed consideration of the particular microprocessor.

In the Phase I effort it was concluded that the memory required to implement LQG control on microprocessors was modest. Although the experiment summarized above did reveal that additional memory is required, the increase is small and the conclusion that memory requirements are not significant is unchanged. The caution to be exercised in predicting requirements is that subtle characteristics of the particular microprocessor selected for the application should be considered in arriving at detailed estimates.

In summary, the Phase I technique for analytically predicting word length requirements agrees very well with the actual results of the validation. The techniques for estimating computational requirements and memory, once updated to account for particular microprocessor characteristics (i.e., multiply times for signed operations) were found to produce good estimates for the actual requirements.

REFERENCES

1. Michael, G. J. and G. S. Sogliero: Key Control Assessment for Linear Multivariable Systems. Proceedings, 1976 IEEE Conference on Decision and Control, December 1-3, 1976, Clearwater Beach, Florida, pp. 1120-1125.
2. Michael, G. J. and F. A. Farrar: An Analytical Method for the Synthesis of Nonlinear Multivariable Feedback Control. United Technologies Research Center Report M941338-2, Final Technical Report prepared under Department of the Navy Contract N00014-72-C-0414, June 1973 (DDC Accession No. AD 962797).
3. Michael, G. J. and F. A. Farrar: Stochastic Regulation of Nonlinear Multivariable Dynamic Systems. United Technologies Research Center Report ONR-CR215-219-4P, Final Technical Report prepared under Department of the Navy Contract N00014-73-C-0281, March 1976.
4. Michael, G. J. and F. A. Farrar: Development of Optimal Control Modes for Advanced Technology Propulsion Systems. United Technologies Research Center Report N911620-2, Annual Technical Report prepared under Department of the Navy Contract N00014-73-C-0281, March 1974 (DDC Accession No. AD 775337).
5. Farrar, F. A. and G. J. Michael: Large-Signal Estimation for Stochastic Nonlinear Multivariable Dynamic Systems. Office of Naval Research Report ONR-CR215-247-1, Annual Technical Report prepared under Department of the Navy Contract N00014-76-C-0710, March 1977.
6. Farrar, F. A. and G. J. Michael: Nonlinear Stochastic Control Design for Gas Turbine Engines. Office of Naval Research Report ONR-CR212-247-2F, Draft Final Technical Report prepared under Department of the Navy Contract N00014-76-C-0710, submitted to ONR, June 1978.
7. Farrar, F. A. and G. J. Michael: Multivariable Control Synthesis for F100 Engine Stability Reset Operation. United Technologies Research Center Report UTRC77-23, Technical Report prepared under P&WA Government Products Division Engineering Order Supplement No. 802707, March 9, 1977.
8. DeHoff, R. L. and W. E. Hall: Multivariable Control Design Principles with Application to the F100 Turbofan Engine. Proceedings, 1976 Joint Automatic Control Conference, Lafayette, Indiana, July 1976.
9. Spang III, H. A.: The Challenge of Microprocessors. IEEE Control Systems Newsletter, October 1976.

REFERENCES (Cont'd)

10. Barnich, R. G.: Designing Microcomputers into Your Products. Instrument Society of America ISA-76 International Conference and Exhibit, Houston, Texas, October 1976.
11. Grossman, R. M.: "Micro" Peripherals - Moving to Meet the Microcomputer Challenge, EDN. February 5, 1976, pp. 38-47.
12. Bishop, P. G.: Microprocessors: Computing in Miniature. Physics in Technology, March 1976, pp. 47-53.
13. Larson, A.: Some Questions and Answers about Microprocessors. Proceedings of the IFAC 6th World Congress, Boston/Cambridge, Massachusetts, August 1975.
14. Hopkins, H. G.: An 8 Bit Microprocessor Identifies Second Order Transfer Functions. Proceedings, 1976 IEEE Conference on Decision and Control, Clearwater, Florida, December 1976.
15. Farrar, F. A.: Microprocessor Implementation of Advanced Control Modes. Proceedings, 1977 Summer Computer Simulation Conference, Chicago, Illinois, July 1977, pp. 339-342.
16. Farrar, F. A., and R. S. Eidens: Microprocessor Requirements for Implementing Modern Control Logic. United Technologies Research Center Report R79-944258-2, Final Report prepared under Air Force Office of Scientific Research Contract F49620-78-C-0017, March 1979.
17. Bryson, A. E., Jr. and Y. C. Ho: Applied Optimal Control. Ginn and Company, Waltham, Mass., 1969.
18. Fitzgerald, R. J.: Divergence of the Kalman Filter. IEEE Trans. Auto. Control, Vol. AC-16, No. 6, December 1979, pp. 736-747.
19. Athans, M. A.: The Compensated Kalman Filter Symposium on Nonlinear Estimation, San Diego, September, 1971.

LIST OF SYMBOLS

A	Constant $n \times n$ matrix in linear system dynamic description
B	Constant $n \times m$ matrix in linear system dynamic description
C	Constant $p \times n$ matrix in linear system dynamic description
D	Constant $p \times m$ matrix in linear system dynamic description
E	Constant $l \times n$ matrix in linear system dynamic description
e	$n \times 1$ error vector used to represent bias errors due to model mismatched Kalman filter
F	Constant $n \times n$ matrix used to describe optimal deterministic closed loop system dynamics
F_T	Constant $n \times n$ matrix used to describe transformed optimal closed loop system dynamics
f	Nonlinear $n \times 1$ vector function describing rate of change of system state vector
G	Constant $m \times n$ optimal deterministic closed loop feedback gain matrix
G_D	Constant $m \times n$ optimal deterministic closed loop feedback gain matrix for microprocessor implementation
g	Nonlinear $p \times 1$ vector function describing system output vector
H	Constant $n \times l$ Kalman filter gain matrix
H_D	Constant $n \times l$ Kalman filter gain matrix for microprocessor implementation
h	Nonlinear $l \times 1$ vector function describing measurement vector
I	Identity matrix
i	General subscript
J	Performance index

LIST OF SYMBOLS (Cont'd)

R	Number of operating points used in linearizing nonlinear system
K_c	$n \times l$ compensator gain matrix for estimator with model mismatch compensation
K_f	$n \times l$ Kalman filter gain matrix for estimator with model mismatch compensation
K	Discrete time
l	Dimension of system measurement vector z
m	Dimension of system control vector u
n	Dimension of system state vector x
P	Dimension of system output vector y
Q	Constant $p \times p$ matrix used in J
R	Constant $m \times m$ matrix used in J
RAM	Random access memory
PROM	Programmable read only memory
T	Constant $n \times n$ transformation matrix
U	$m \times 1$ control vector
U^*	$m \times 1$ optimal control vector
W	$n \times 1$ transformed state vector
x	$n \times 1$ system state vector
y	$p \times 1$ system output vector
z	$l \times 1$ system measurement vector

LIST OF SYMBOLS (Cont'd)

- η 1×1 sensor noise vector
- ξ $m \times 1$ process noise vector
- Φ Constant $n \times n$ closed loop system matrix
- Φ_D Constant $n \times n$ closed loop system matrix used in microprocessor implementation
- $\delta ()$ Small signal (linearized) representation of variable

TABLE I
 COMPUTATIONAL REQUIREMENTS
 PLPO Implementation - Standard Structure

Function	Application	Number of Operations
Addition	Filter	$n(n-1+5l)$
	Control	$n(m+1)$
Multiplication	Filter	n^2+3nl
	Control	$n(m+l)$
Interface	Input	m
	Output	l

TABLE II
 MEMORY REQUIREMENTS
 PLPO Implementation

Variable	Memory Type	Number of Locations
Past state estimate $(\hat{x}(t + \Delta t/t))$	RAM	n
Current state estimate $(\hat{x}(t + \Delta t/t + \Delta t))$	RAM	n
Measurement $(z(t + \Delta t))$	RAM	l
Control $(u(t + \Delta t))$	RAM	m
Control matrix G	PROM	$\bar{k}n^2$
Kalman matrix K_f	PROM	$\bar{k}nl$
Kalman matrix K_c	PROM	$\bar{k}nl$

TABLE III
 COMPUTATION REQUIREMENTS FOR LINEAR SYSTEMS
 Number Of Operations Per Sample Interval

Function	Structure	Standard	Jordan Canonical	Companion
Addition	Filter	$n(n-1) + n(l-1) + n$	$n(l-1) + n$	$(n-1) + n(l-1) + n$
	Control	$m(n-1)$	$m(n-1)$	$m(n-1)$
Multiplication	Filter	$n^2 + nl$	$n + nl$	$n + nl$
	Control	nm	nm	nm
Interface	Input	m	m	m
	Output	l	l	l

TABLE IV
MEMORY REQUIREMENTS FOR LINEAR SYSTEMS

Variable	Memory Type	Memory (Words)		
		Standard Structure	Jordan Canonical Structure	Companion Structure
Past state estimate ($\hat{w}(k)$)	RAM	n	n	n
Current state estimate ($\hat{w}(k+1)$)	RAM	n	n	n
Measurement ($z(k+1)$)	RAM	l	l	l
Control ($u(k+1)$)	RAM	m	m	m
System matrix (ϕ_D)	PROM	n^2	n	n
Kalman gain matrix (H_1)	PROM	nl	nl	nl
Control gain matrix (G_D)	PROM	mn	mn	mn

TABLE V
SECOND-ORDER SYSTEM, CONTROL, AND FILTER MATRICES

Matrix	Matrix Elements	
A	0.0 -2.0	1.0 -4.0
B	0.0 2.0	
C	1.0	0.0
D	0	
E	1.0	0.0
G	-0.871	-0.207
H_D	1.130 -0.360	
ϕ_D	.873 - .265	.071 .632

TABLE VI

COMPARISON OF PREDICTED AND ACTUAL COMPUTATION TIME AND MEMORY

Second Order System
Intel 8080 Microprocessor

	Computation time - msec	Memory Requirements - bytes	
		RAM	PROM
Prediction	4.25	15	475
Actual	4.70	15	468

TABLE VII

FIFTH-ORDER F100 ENGINE MODEL DYNAMICS

Engine Model Linearized at Sea-Level Static Military Operation

<u>States</u>	<u>Outputs</u>	<u>Controls</u>
Fan turbine inlet temperature	Airflow	Jet exhaust area
Main burner pressure	Fan stability margin	Fan inlet guide vanes
Fan speed	Compressor stability margin	Compressor variable vanes
Compressor speed	Thrust	Main burner fuel flow
Afterburner pressure	High Turbine inlet temperature	

Matrix	Matrix Elements				
A	-34.013	-9.303	12.037	-2.398	-1.254
	4.389	-38.762	-4.221	28.480	14.729
	-4.755	2.287	-0.400	-1.546	-2.200
	2.046	1.062	-0.729	-2.150	-0.624
	4.150	-8.814	-0.167	7.477	1.099
B	0.766	0.546	-0.813	17.095	
	0.056	1.341	7.737	8.641	
	0.156	-1.176	-0.416	2.034	
	-0.136	-0.024	-0.555	-0.378	
	-4.729	0.874	1.617	0.223	
C	-0.042	0.063	0.013	-0.054	1.404
	1.045	0.092	-0.060	-0.028	-0.050
	0.386	0.100	-0.217	0.170	-0.095
	0.305	-0.326	-0.458	0.584	-0.538
	-0.183	-0.564	0.394	-0.165	0.394
D	1.044	0.001	-0.013	0.002	
	-0.015	-0.003	-0.013	-0.044	
	-0.043	0.278	0.035	-0.155	
	-0.101	0.281	0.137	-0.041	
	0.073	0.047	-0.091	0.050	
E	1.0	0	0	0	0
	0	1.0	0	0	0
	0	0	1.0	0	0
	0	0	0	1.0	0
	0	0	0	0	1.0

TABLE VIII

APPROXIMATE FIFTH-ORDER F100 ENGINE MODEL DYNAMICS

A, B matrices represent approximations to those in Table VII which are required to allow EAI 1000 analog computer implementation

Matrix	Matrix Elements				
A	-34.013	-9.0	12.037	-2.2	0.0
	4.4	-38.762	-4.221	28.480	14.729
	-4.4	2.287	0.0	-1.546	-2.200
	2.046	1.062	-0.729	-2.2	-0.8
	4.4	-9.0	0.0	7.477	0.8
B	0.0	0.0	0.0	17.095	
	0.0	1.2	7.737	8.641	
	0.14	-1.2	-0.5	2.034	
	-0.14	0.0	-0.5	-0.378	
	-4.729	0.874	1.617	0.0	

TABLE IX

CONTROL AND FILTER MATRICES FOR FIFTH ORDER IMPLEMENTATION

Matrices Used For Approximate Dynamic Description of Table VIII

Matrix	Matrix Elements				
G_D	1.602	-1.183	2.224	0.148	5.53
	0.012	3.074	-0.341	-0.903	-0.223
	-2.942	-5.064	5.544	-2.222	8.148
	-4.362	0.749	-0.652	-0.092	-0.811
H_D	0.153	.047	.514	.061	.001
	.084	.026	.304	.093	.000
	.012	.007	.087	.063	.000
	.001	.001	.034	.077	.000
	.002	.000	.007	-0.005	.000
ϕ_D	.107	- .010	- .471	- .104	- .041
	- .180	.107	.027	.057	.696
	- .095	.026	.854	- .044	- .081
	.033	.042	- .087	.899	- .062
	- .008	- .085	- .093	.041	.650

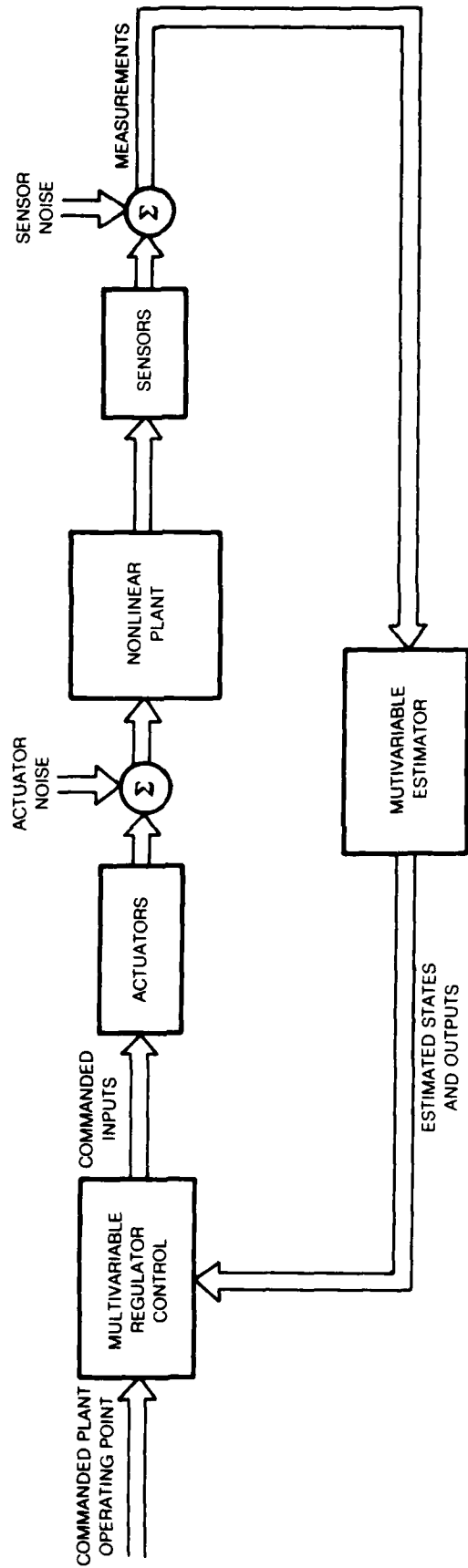
TABLE X

COMPARISON OF PREDICTED AND ACTUAL COMPUTATION TIMES AND MEMORY

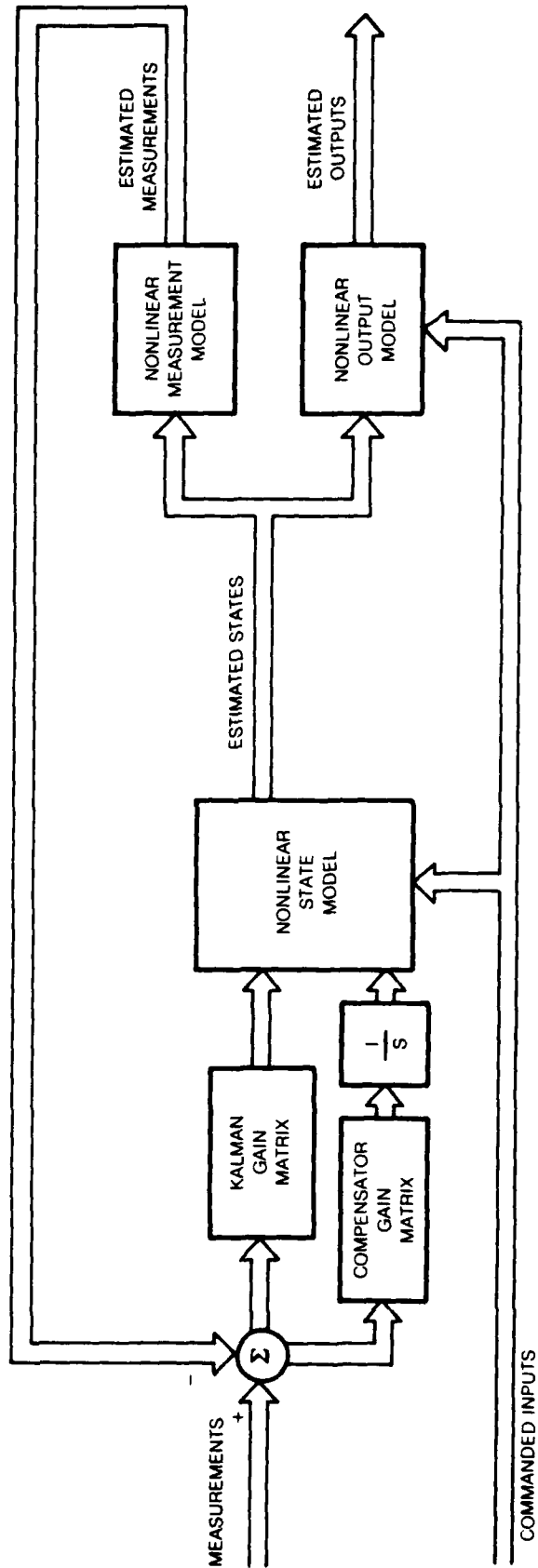
Fifth Order System
LSI 11/2 Microprocessor
Updated Prediction Accounts for Additional Multiply Software

	Computation Time - msec	Memory Requirements - Words	
		RAM	PROM
Original Prediction	9.68	121	183
Updated Prediction	16.6	121	284
Actual	15.77	109	313

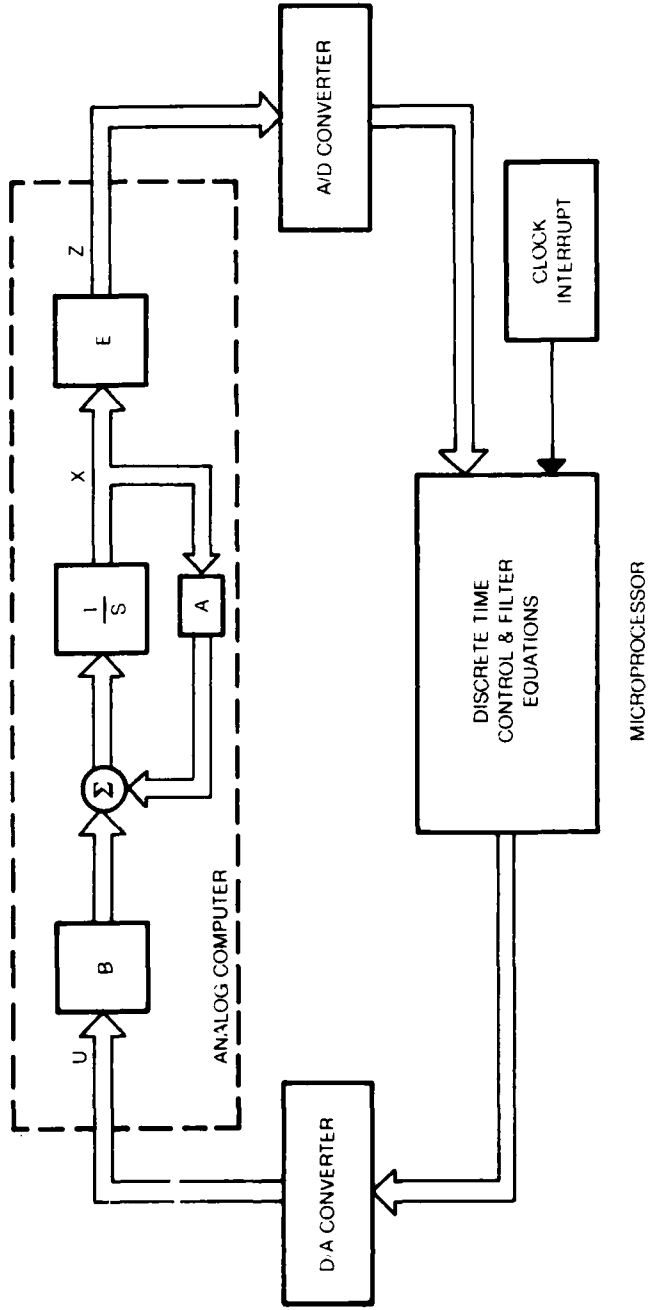
STOCHASTIC FEEDBACK REGULATOR STRUCTURE



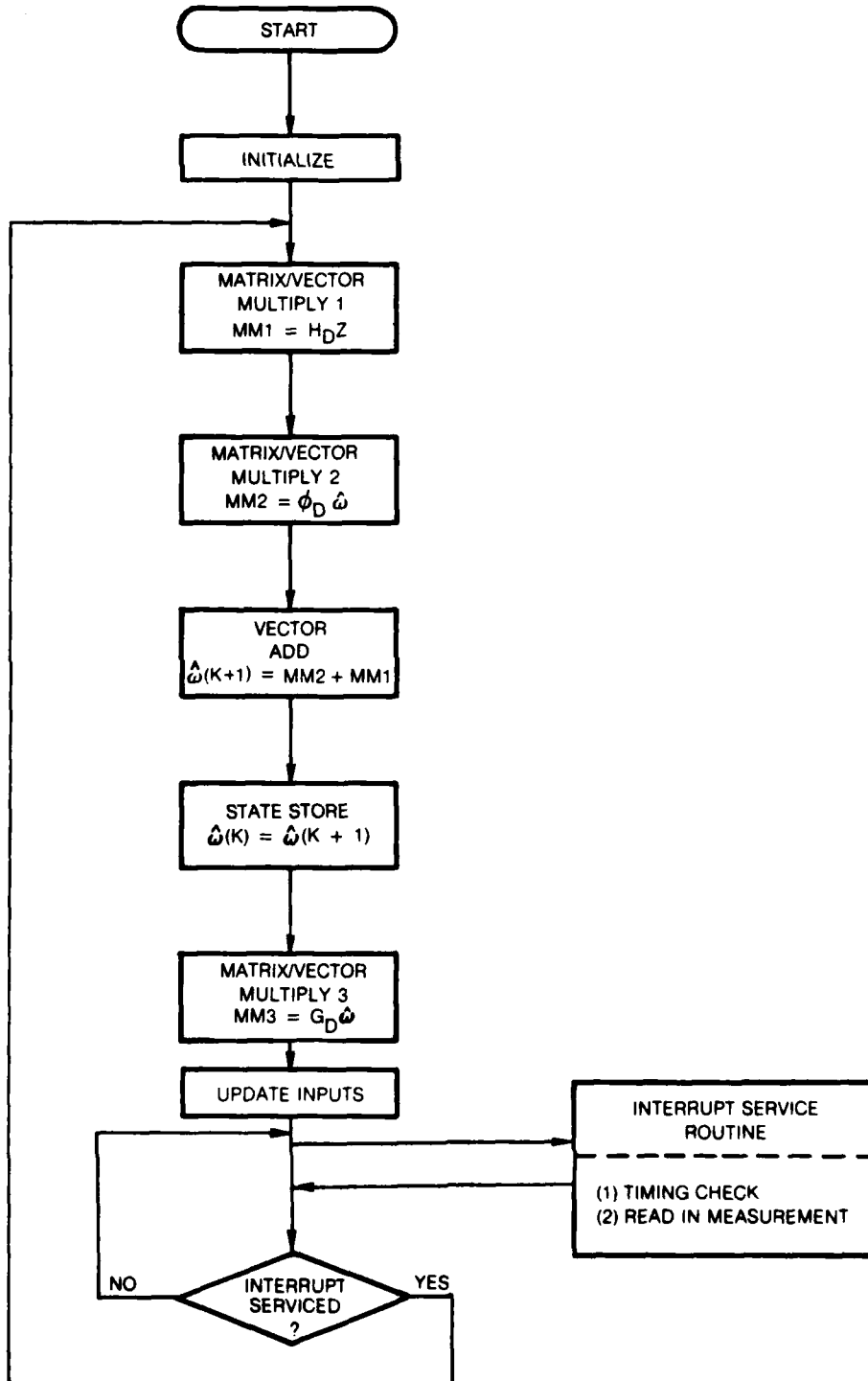
FILTER WITH MODEL-MISMATCH COMPENSATION



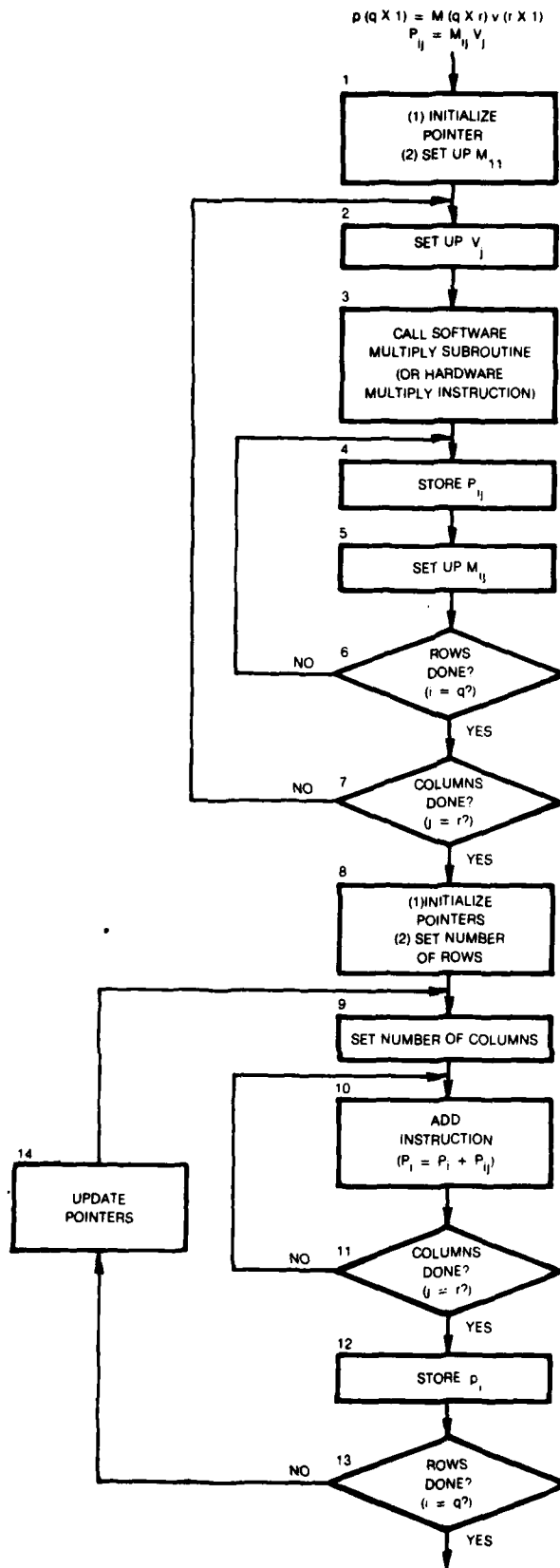
SYSTEM STRUCTURE FOR DEMONSTRATING MICROPROCESSOR CONTROL



CONTROL CODE BLOCK DIAGRAM

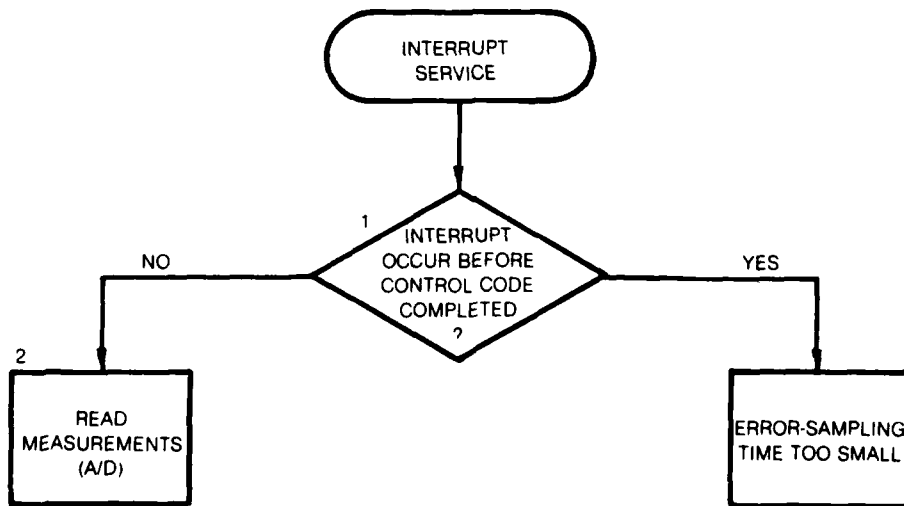
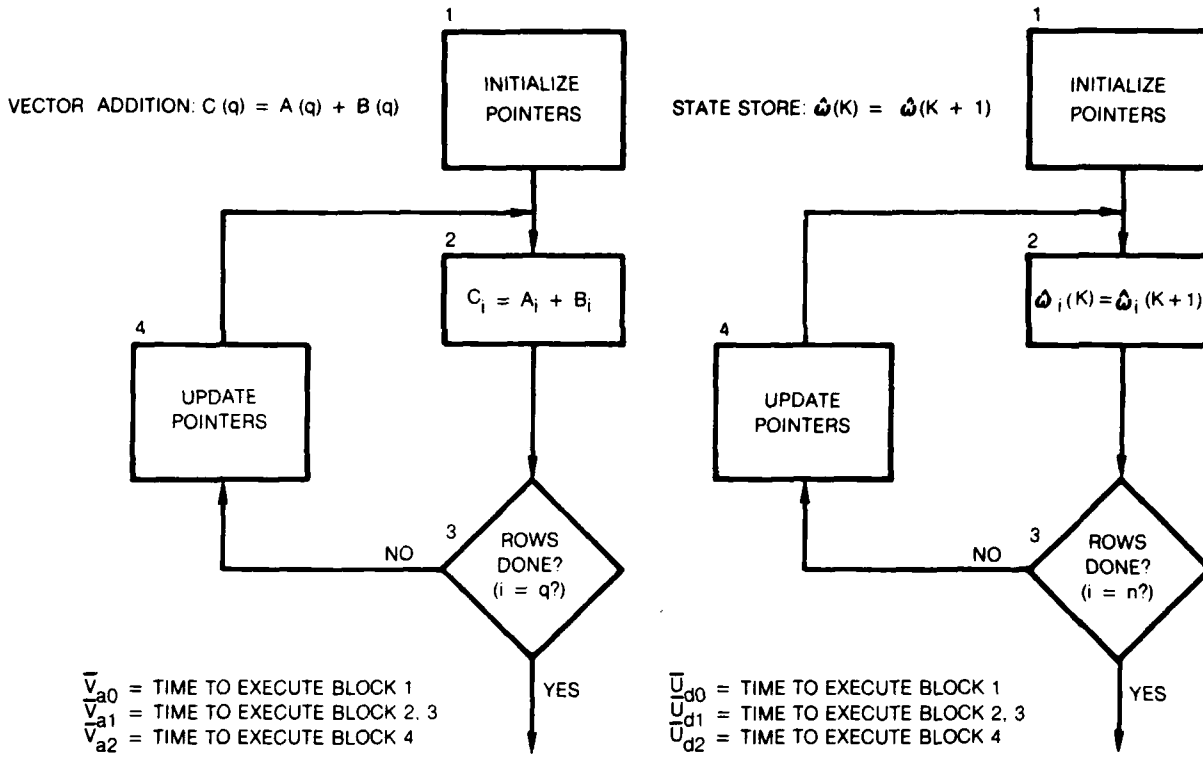


BLOCK DIAGRAM FOR MATRIX/VECTOR MULTIPLICATION



\bar{M} = TIME TO EXECUTE BLOCK 3
 L_{m0} = TIME TO EXECUTE BLOCKS 1 8
 L_{m1} = TIME TO EXECUTE BLOCKS 2 7
 L_{m2} = TIME TO EXECUTE BLOCKS 4 5 6
 A = TIME TO EXECUTE BLOCK 10
 L_{a1} = TIME TO EXECUTE BLOCK 11
 L_{a2} = TIME TO EXECUTE BLOCKS 9 12 13
 L_{a3} = TIME TO EXECUTE BLOCK 14

BLOCK DIAGRAM FOR VECTOR ADDITION, STATE STORE AND INTERRUPT SERVICE



\bar{T}_C = TIME TO EXECUTE BLOCK 1
 \bar{A}_D = TIME TO EXECUTE BLOCK 2
 \bar{D}_A = TIME TO EXECUTE BLOCK 3

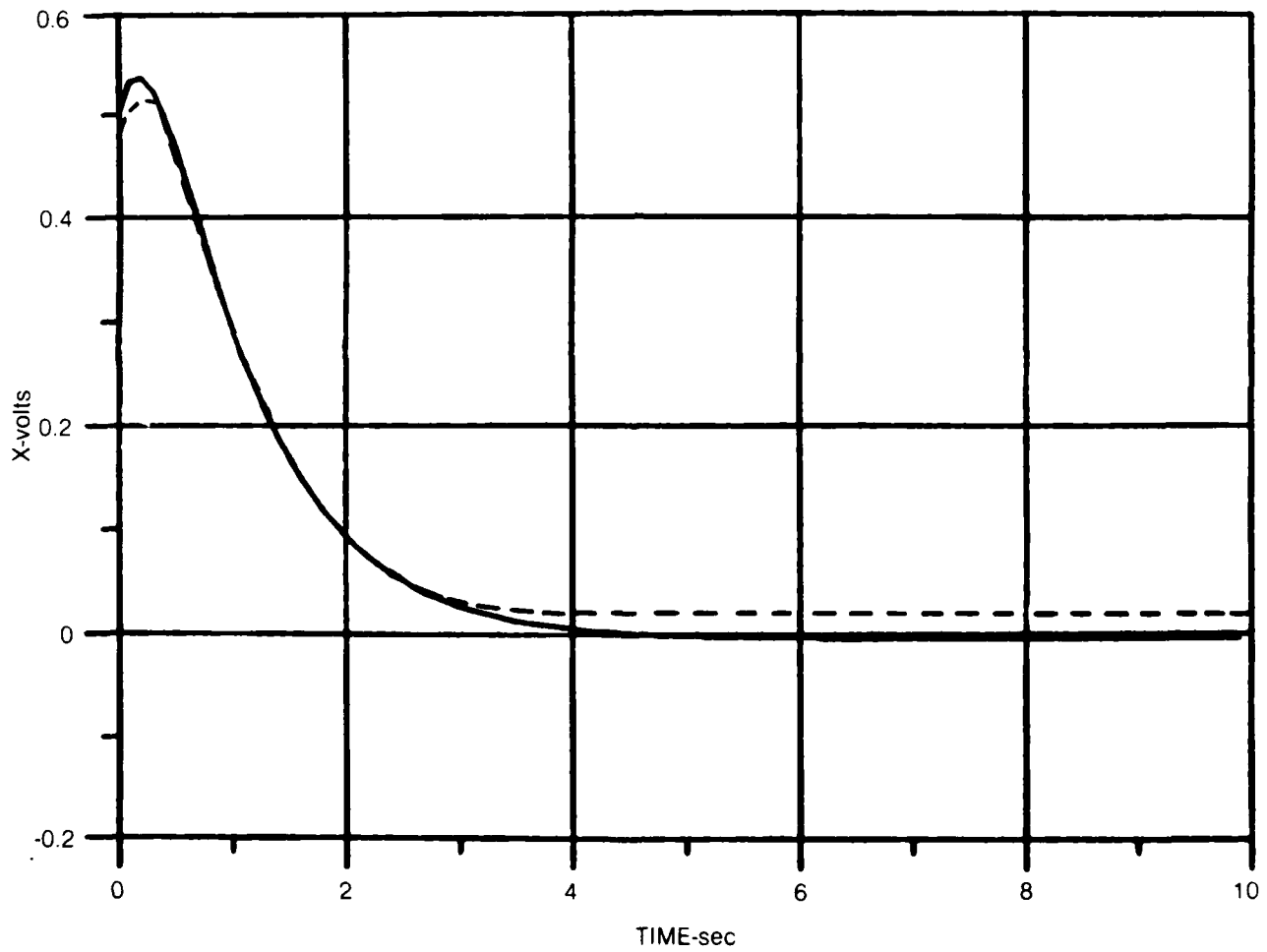
COMPARISON OF PREDICTED AND ACTUAL SECOND ORDER OUTPUT RESPONSE

RESPONSE TO INITIAL CONDITION $X_0 = (0.5, 0.5)$

STANDARD STRUCTURE WITHIN CONTROLLER

$t = 0.1$ sec. WORD LENGTH = 8 BITS

----- REAL TIME ANALOG SYSTEM/DIGITAL MICROPROCESSOR CONTROL
————— SIMULATED ANALOG SYSTEM/SIMULATED MICROPROCESSOR CONTROL



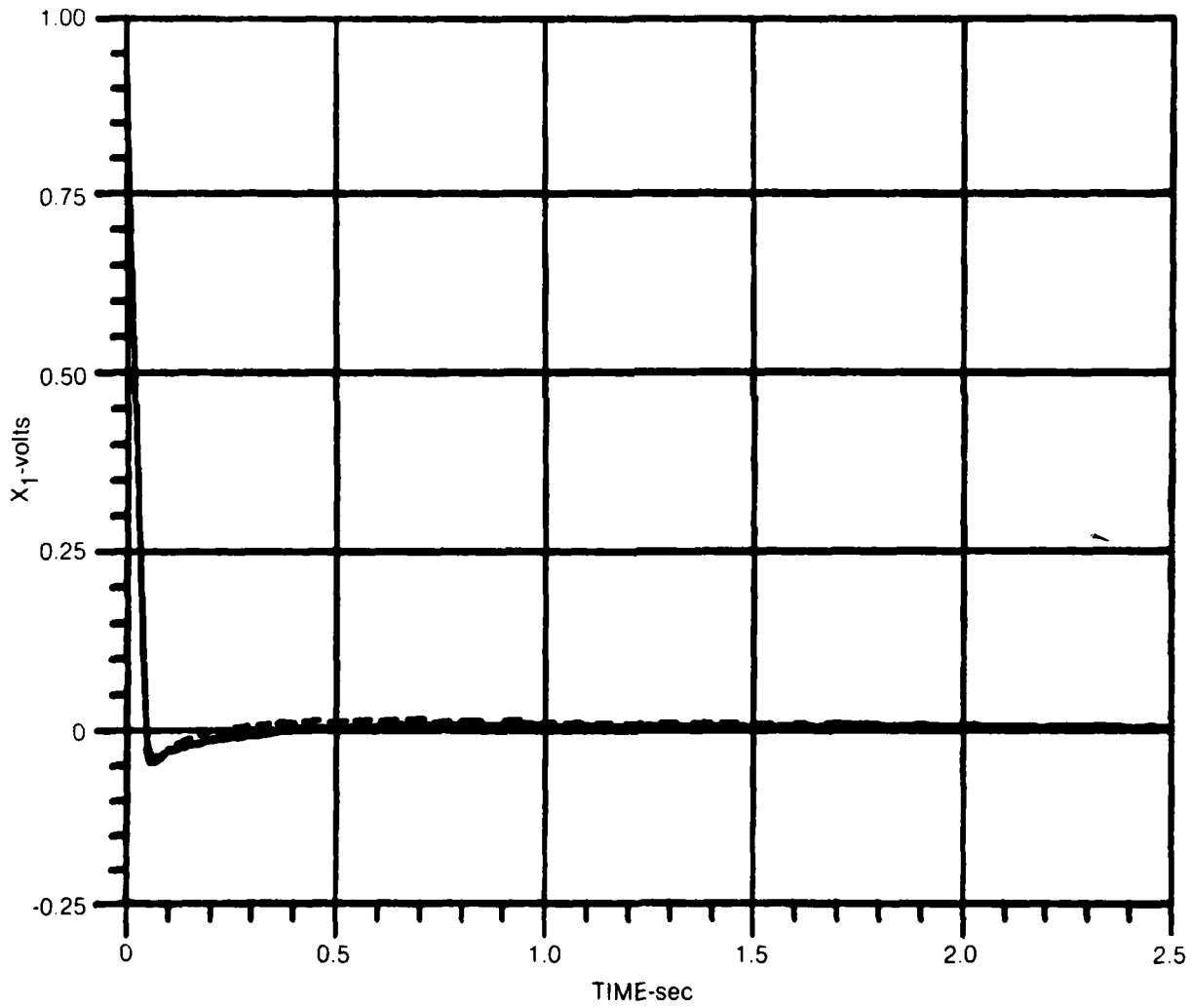
F100 ENGINE MODEL FAN TURBINE INLET TEMPERATURE RESPONSE

ORIGINAL A&B MATRIX vs MODIFIED A&B MATRIX

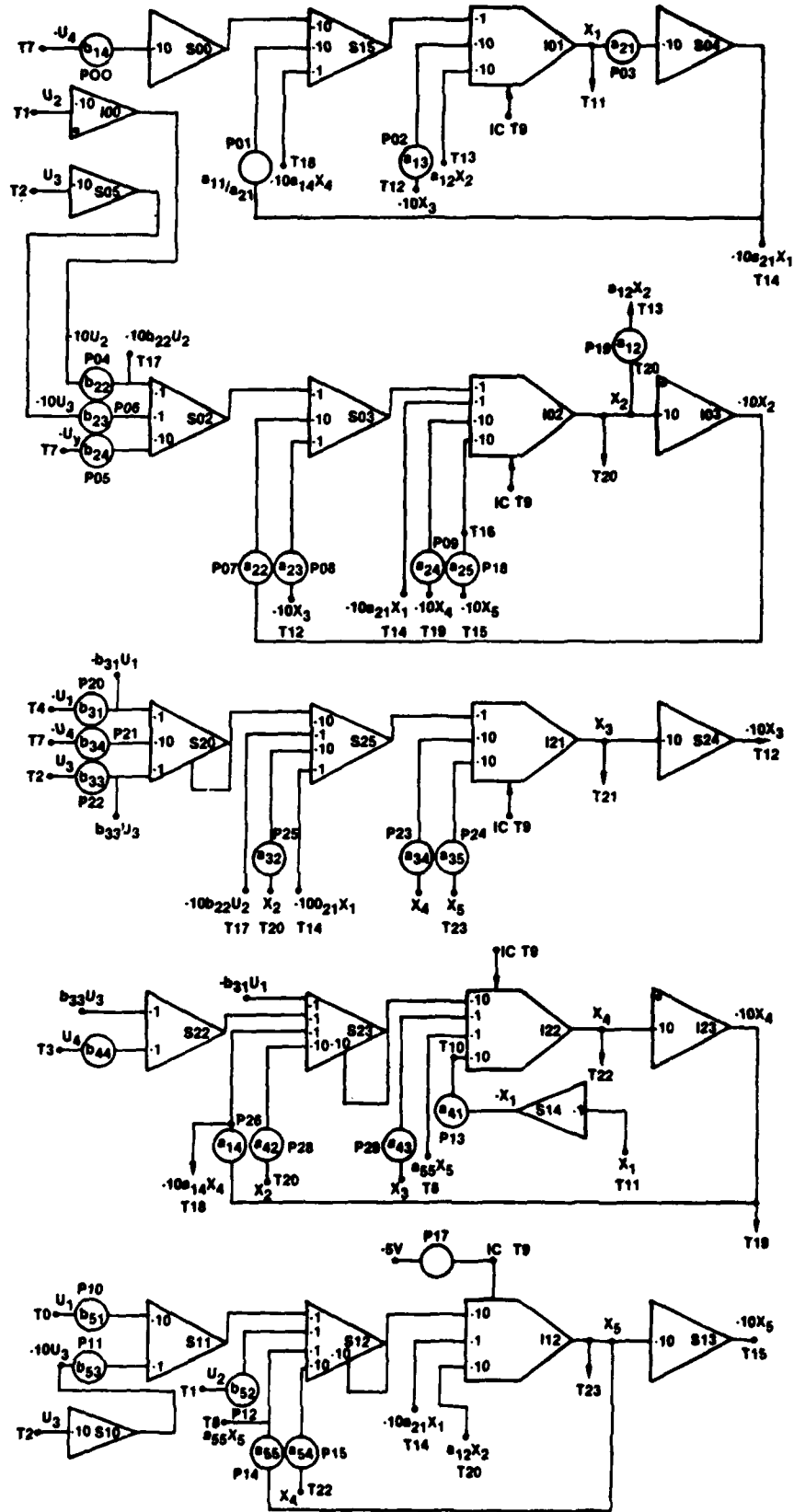
INITIAL CONDITION $x = 0.1$

$\Delta t = 0.025$ sec. WORD LENGTH = 16 BITS

— ORIGINAL MATRIX RESPONSE, UNIVAC SIMULATION
- - - MODIFIED MATRIX RESPONSE, UNIVAC SIMULATION



FIFTH ORDER ENGINE ANALOG SIMULATION

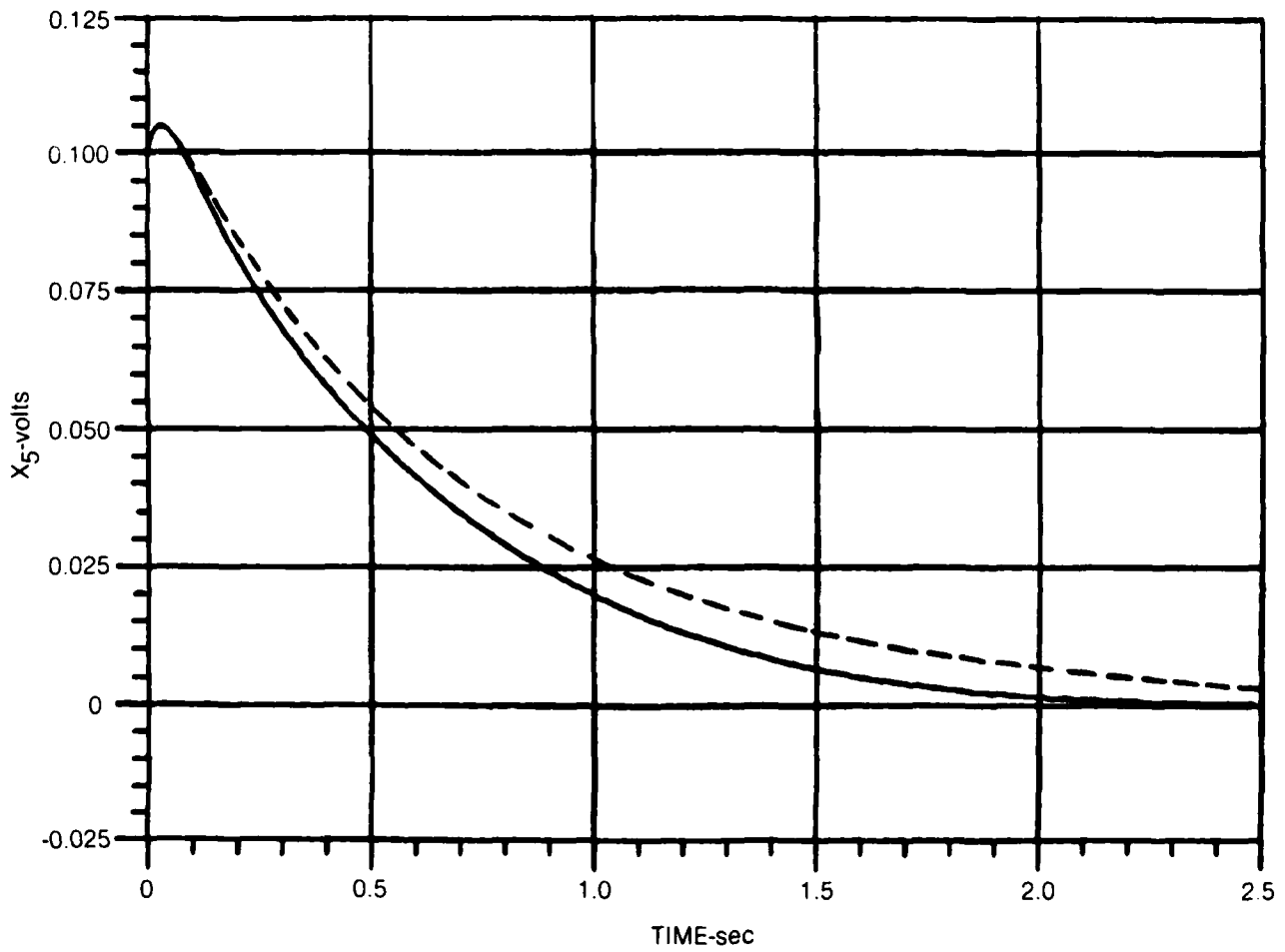


THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY REPRODUCED TO DDD

F100 ENGINE MODEL AFTER BURNER PRESSURE RESPONSE

ANALOG COMPUTER PROGRAM VERIFICATION
UNIVAC ZERO INPUT vs. ANALOG COMPUTER ZERO INPUT
INITIAL CONDITION $x = 0.1$
 $t = 0.025$ sec. WORD LENGTH = 16 BITS

— UNIVAC RESPONSE WITH ZERO INPUT
- - - ANALOG COMPUTER RESPONSE WITH ZERO INPUT



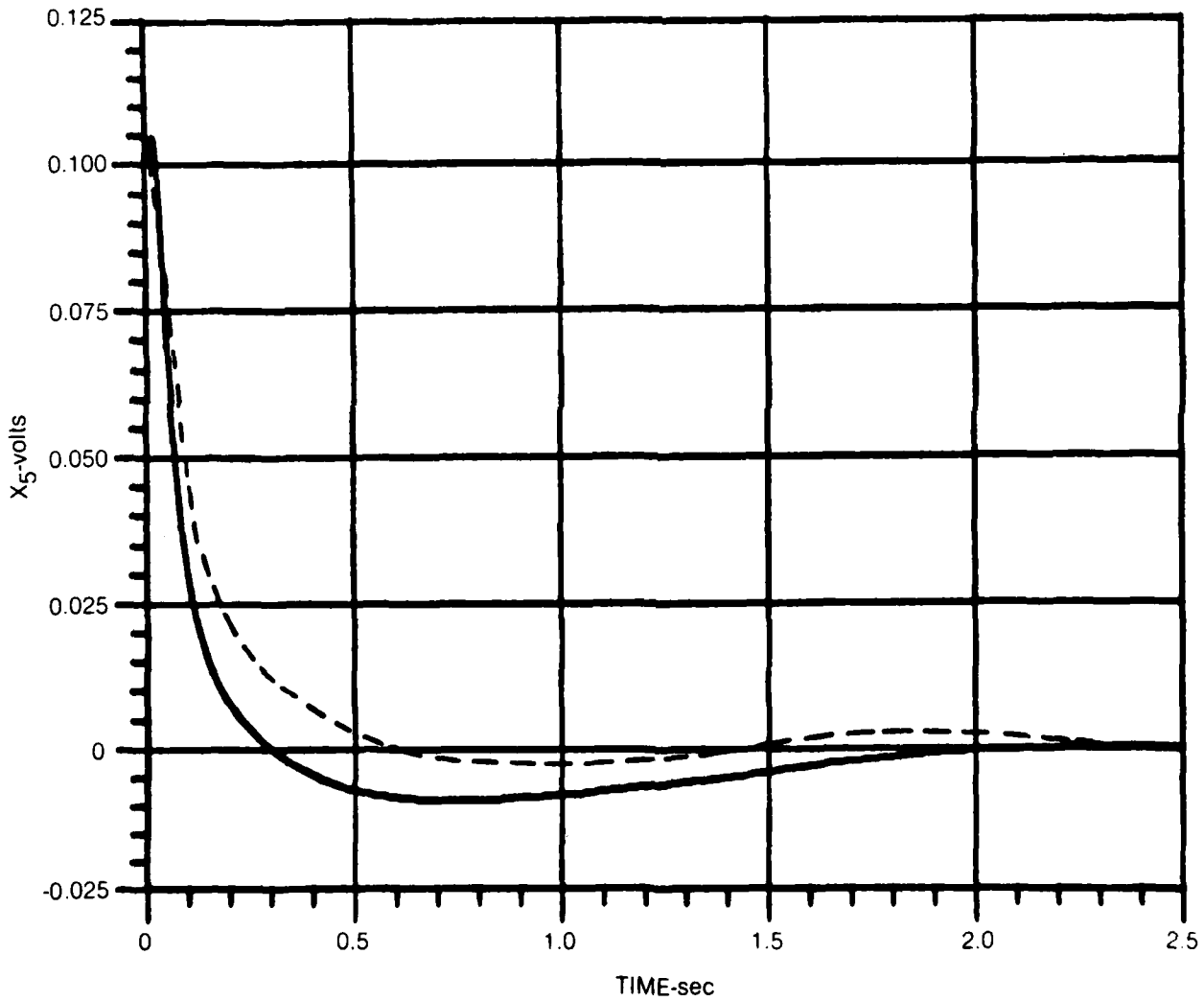
F100 ENGINE MODEL AFTER BURNER PRESSURE RESPONSE

UNIVAC SIMULATED RESPONSE vs. ANALOG COMPUTER RESPONSE

INITIAL CONDITIONS: $X = 0.1$

$t = 0.025$ sec. WORD LENGTH = 16 BITS

—— UNIVAC SIMULATED RESPONSE
- - - - ANALOG COMPUTER RESPONSE

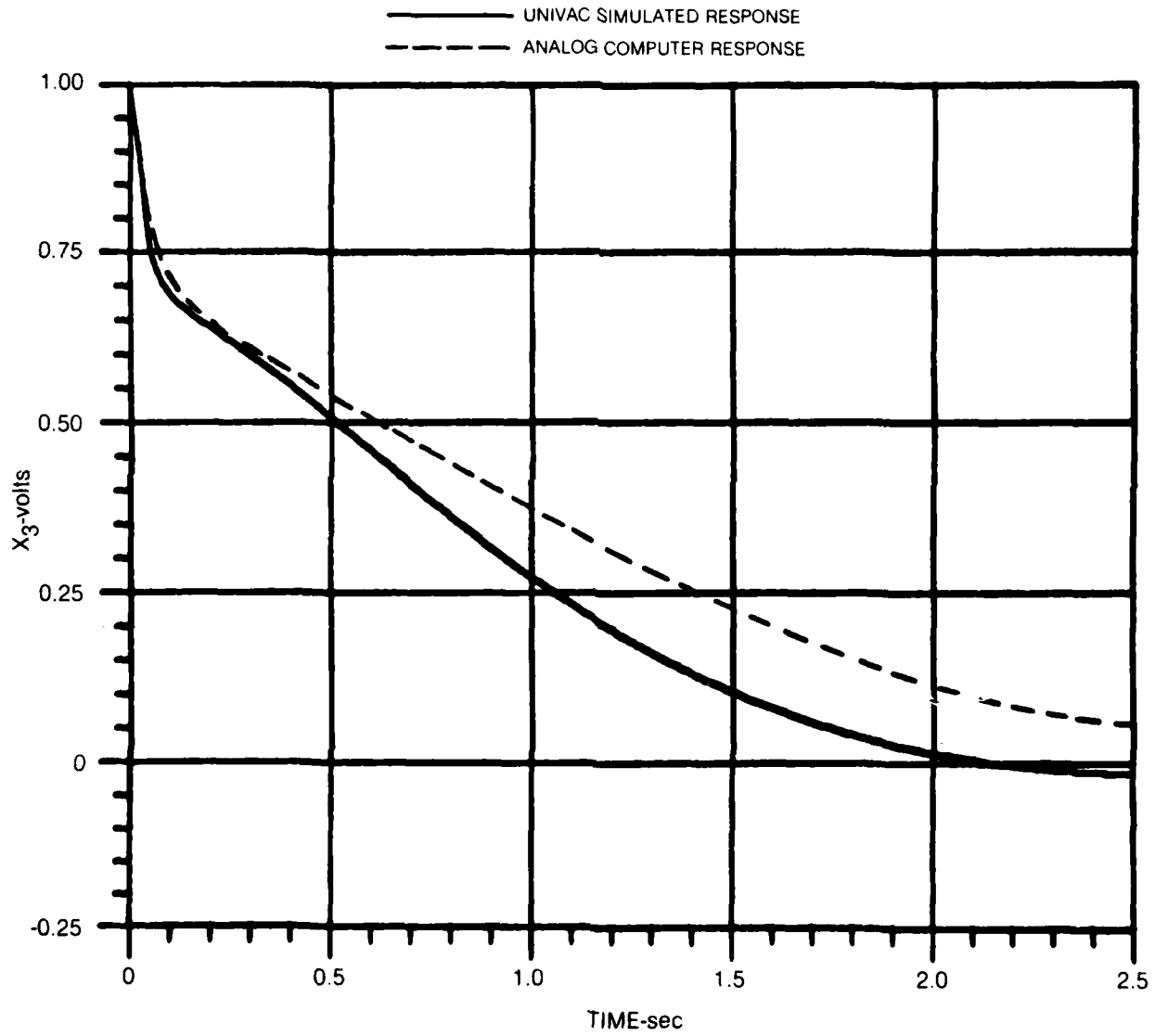


F100 ENGINE MODEL FAN SPEED RESPONSE

UNIVAC SIMULATED RESPONSE vs. ANALOG COMPUTER RESPONSE

INITIAL CONDITIONS: $X = 0.1$

$t = 0.025$ sec. WORD LENGTH = 16 BITS



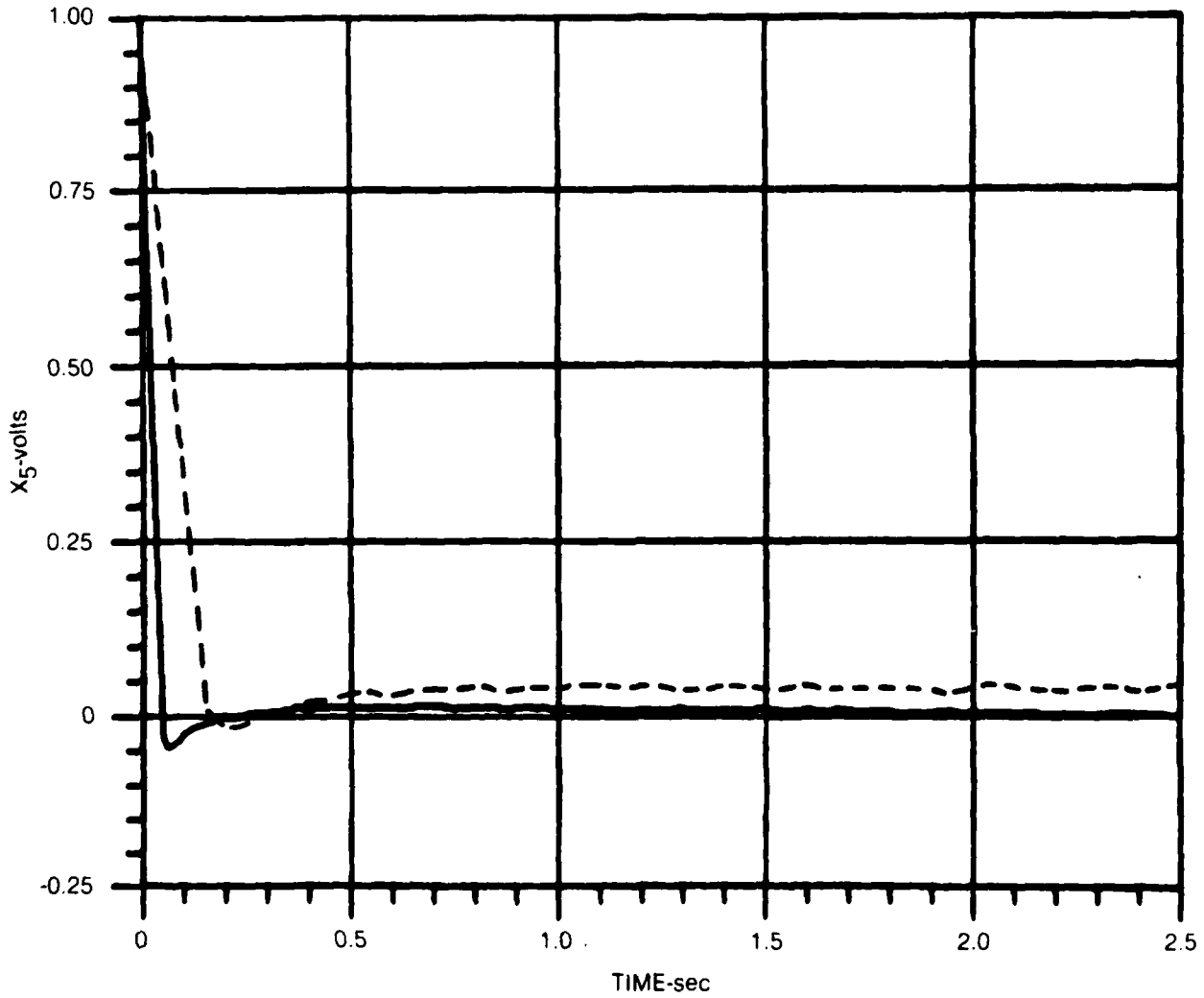
F100 ENGINE MODEL FAN TURBINE INLET TEMPERATURE RESPONSE

UNIVAC SIMULATED RESPONSE vs ANALOG COMPUTER RESPONSE

INITIAL CONDITIONS: $X = 0.1$

$t = 0.025$ sec. WORD LENGTH = 16 BITS

———— UNIVAC SIMULATED RESPONSE
- - - - - ANALOG COMPUTER RESPONSE



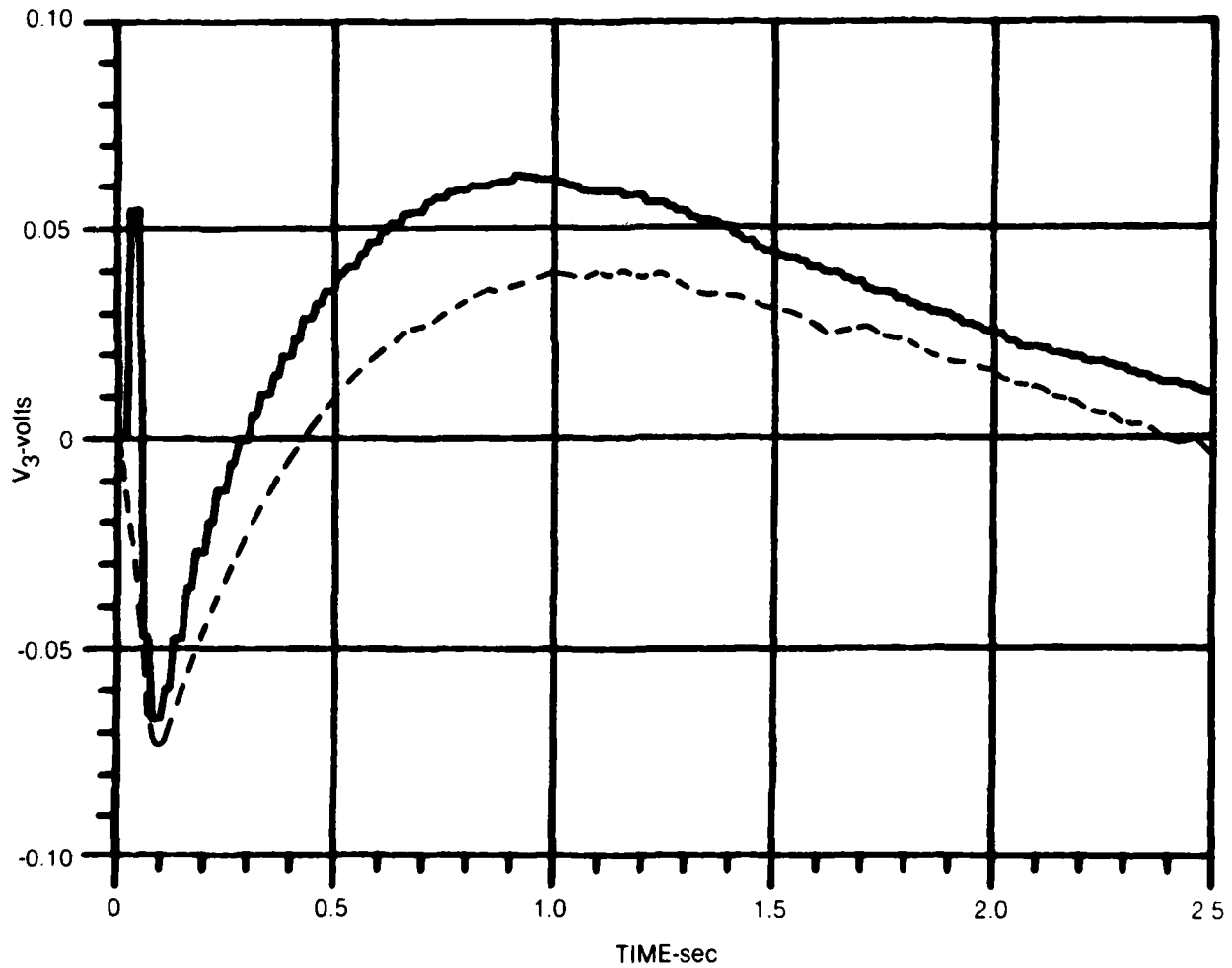
F100 ENGINE MODEL COMPRESSOR VARIABLE VANE PARAMETER

UNIVAC SIMULATED RESPONSE vs. ANALOG COMPUTER RESPONSE

INITIAL CONDITIONS $X = 0.1$

$t = 0.025$ sec. WORD LENGTH = 16 BITS

—— UNIVAC SIMULATED RESPONSE
- - - - ANALOG COMPUTER RESPONSE



APPENDIX A

MICROPROCESSOR SURVEY

Characteristics of (1) microprocessors, (2) A/D and D/A converters, and (3) hardware multipliers are presented in this Appendix. The characteristics tabulated here were obtained from Electrical Design News (EDN) 1976-1978 as well as from TRW product sheets. Microprocessor characteristics -- including word length, internal registers, indexed addressing capabilities, and multiply instruction capability -- are listed in Table A-I. The A/D and D/A characteristics -- including word length, conversion time, and technology -- are shown in Table A-II. Table A-III displays multiplier characteristics including word length, multiply time, and technology.

TABLE A-1
REPRESENTATIVE MICROPROCESSORS

Characteristics Microprocessor	Second Source	Technology	Word Length-- Data: Address (Bits)	On-Chip Clock	Internal Register	Power Supplies (Volts)	Indexed Addressing	Instruction Execution Time (μsec)	Multiply Instruction	ADC On-Chip
Intel 8080/8085	Y	NMOS	8;16	N/Y	7	+5,+12/+5	N	2.0-8.5	N	N
Zilog Z80	Y	NMOS	8;16	N	12	+5	Y	2.0	N	N
Motorola 6800/6802	Y	NMOS	8;16	N/Y	2	+5	Y	2.0-5.0	N	N
Zilog Z8	N	NMOS	8;16	Y	124	+5	Y	1.5-2.2	N	N
Mostek 3870	Y	NMOS	8;16	Y	64	+5	Y	2.0	N	N
Intel 8048 (Family)	Y	NMOS	8;16	Y	-	+5	-	1.4-10.0	N	Y(1)
TI 9900/SBP 9900A	Y	NMOS/I ² L	8,16;16/8;14	Y	16	+5	Y	5.0-10.0	Y	N
Intersil IM6100	Y	CMOS	12;12	N	-	+5	N	5.0	N	N
National Semiconductor 8900	N	NMOS	16;16	N	4	+5,+12	Y	10.0	N	N
Intel 8086	N	NMOS	16;16	N	7	+5	Y	0.6-3.4	Y	N
Zilog Z8000	Y	NMOS	16;24	N	16	+5	Y	0.7-17.5	Y	N
Motorola 68000	-	NMOS	16;24	N	16	+5	Y	-	Y	N
Intel 3000	Y	Bipolar	2 BIT SLICE	N	-	+5	-	-	-	N
Motorola 10800	-	Bipolar/ECL	4 BIT SLICE	N	-	-5.2,-2.0	-	-	-	N
DEC LSI 11/2	-	-	16;16	N	8	-	Y	2.0-232.0	Y	N

(1) Intel 8022 only

TABLE A-II
 REPRESENTATIVE A/D AND D/A CONVERTERS

Converter Type	Manufacturer	Model	Word Length (bits)	Conversion Time (μ sec)	Technology
A/D	TRW	TDC1007J	8	35×10^{-3}	Bipolar
	TRW	TDC1001J	8	400×10^{-3}	Bipolar
	TRW	TDC1002J	8	1	Bipolar
	Analog Devices	AD75705	8	40	CMOS
	Datel	ADC-MC88C	8	500	Bipolar
	Analog Devices	AD7570L	10	120	CMOS
	Datel	ADC-HX12B	12	20	Hybrid
	Analog Devices	AD572BD	12	25	---
	Micre Networks	ADC80	12	25	Hybrid
	National Semiconductor	ADC1210	12	50	Hybrid
D/A	TRW	TDC1016J	8	35×10^{-3}	Bipolar
	Analog Devices	AD7523JN	8	100×10^{-3}	---
	Datel	DAC-UP88	8	2	Bipolar
	National Semiconductor	DAC0800	8	135	Bipolar
	Datel	DAC-088	8	150	Bipolar
	TRW	TDC1017J	10	50×10^{-3}	Bipolar
	Analog Devices	AD7541KN	12	1	---
	Datel	DAC-HK12B	12	3	Hybrid
	Harris Semiconductor	H1-5612	12	85	Bipolar
	Harris Semiconductor	H1-562	12	200	Bipolar
	Datel	DAC-HA12B	12	500	Hybrid
	Analog Devices	AD7531	12	500	Hybrid

TABLE A-III
 REPRESENTATIVE MULTIPLIERS

Manufacturer	Model	Word Length (bits)	Multiply Time (nsec)	Technology	Accumulator
TRW	TDC1008J	8	70	TTL	Y
MONOLITHIC MEMORIES	57558	8	100	—	N
TRW	MPY-8AJ	8	130	TTL	N
TRW	MPY-12A	12	150	TTL	N
TRW	TDC1003J	12	175	TTL	Y
TRW	TDC1010J	16	115	TTL	Y
TRW	MPY-16A	16	160	TTL	N
AMD	9511	16	42000	—	N


```

76: 0002      US2 EQU 2      ; 8 OF ELEMENTS IN VECTOR
77: 0002      MS2 EQU 2      ; 8 ROWS IN MATRIX
78: 5300      DMA2 EQU 5300H  ; DATA MATRIX START ADDR 82
79: 5400      PMMA2 EQU 5400H ; PARTIAL MATRIX MPY START ADDR
80: 5500      MMRA2 EQU 5500H ; MATRIX MPY RESULT START ADDR
81:           ;
82:           ;
83: 0002      US3 EQU 2      ; 8 OF ELEMENTS IN VECTOR
84: 0001      MS3 EQU 1      ; 8 ROWS IN MATRIX
85: 5600      DMA3 EQU 5600H  ; DATA MATRIX START ADDR 83
86: 5700      PMMA3 EQU 5700H ; PARTIAL MATRIX MPY START ADDR
87: 5800      MMRA3 EQU 5800H ; MATRIX MPY RESULT START ADDR
88:           ;
89:           ;
90:           ;
91: 5800      UOUT EQU 5800H   ; OUTPUT VAR LOCATION
92: 5300      STRIPC EQU 5300H ; STRIP CHART VAR LOCATION
93:           ;
94:           ;
95: 3C3D      ORG 3C3DH       ;
96: 3C3D      C38941        JMP INTR      ; SET UP INTERRUPT SERV ROUT.
97:           ;
98:           ;
99: 4000      ORG 4000H      ;
100:          ;
101:          ; INITIALIZATION
102:          ;
103: 4000      F3           DI          ; DISABLE INTERRUPTS
104: 4001      310040       LXI SP,4000H ; SET UP STACK POINTER
105:          ;
106:          ;
107:          ;
108:          ; PROGRAM START:
109:          ;
110:          ; MULTIPLY MATRIX MD (KALMAN FILTER GAIN MATRIX) WITH
111:          ; VECTOR Z (SYSTEM MEASUREMENT VECTOR).
112:          ; MD IS 2 ROWS X 1 COL
113:          ; Z IS 1 ROW X 1 COL
114:          ;
115: 4004      STRT1:
116: 4004      3E01         MUI A,US1   ; ESTABLISH VECTOR SIZE
117: 4006      110051       LXI D,PMMA1 ; PARTIAL MATRIX MPY START ADDR
118: 4008      210050       LXI H,DMA1   ; DATA MATRIX START ADDR
119:          ;
120: 400C      COLN1:
121: 400C      F5         PUSH PSW     ; START NEXT COLUMN MPY, SAVE
122: 400D      0602       MUI B,MS1   ; ESTABLISH MATRIX ROW SIZE
123: 400F      4E         MOV C,M     ; GET OPERAND 81
124: 4010      23         INX H       ; POINT TO OPERAND 82
125:          ;
126: 4011      ROWN1:
127: 4011      C5         PUSH B      ; MPY THE ELEMENTS IN EACH ROW
128: 4012      ES        PUSH H      ; SAVE B,C,M & L
129: 4013      66        MOV H,M     ; GET OPERAND 82
130: 4014      CD4F41    CALL MULT ; 8 BIT SIGNED
131: 4017      7C        MOV A,M     ; PREPARE TO STORE
132: 4018      12        STAX D      ; SAVE PARTIAL MATRIX MPY
133: 4019      13        INX D      ; ADJUST SAVE POINTER
134: 401A      E1        POP H      ; RESTORE H & L
135: 401B      23        INX H      ; POINT TO NEXT OPERAND
136: 401C      C1        POP B      ; RESTORE VECTOR COUNT (B)
137:          ; RESTORE OPERAND 81 (C)
138: 401D      05        DCR B       ; COLUMN ALL DONE ?
139: 401E      C21140    JNZ ROWN1  ; NO
140: 4021      F1        POP PSW     ; YES, RESTORE VECT SIZE
141: 4022      3D        DCR A       ; ALL MULTIPLIES COMPLETE ?
142: 4023      C20C40    JNZ COLN1  ; NO
143:          ; YES, SUM PARTIALS TO COMPLET
144: 4026      0602       MUI B,MS1   ; REESTABLISH MATRIX ROW SIZE
145: 4028      110052     LXI D,PMMA1 ; MATRIX MPY RESULT START ADDR
146: 402B      210051     LXI H,PMMA1 ; PARTIAL MAT MPY START ADDR
147:          ;
148: 402E      SUMA1:
149: 402E      0E01       MUI C,US1   ; REINITIALIZE VECTOR SIZE
150: 4030      AF        XRA A       ; CLEAR REG A

```

```

151: 4031 D5          PUSH D          ; OBTAIN POINTER OFFSET IN D&E
152: 4032 1600       RUI D,0        ; CLEAR UPPER PORTION OF D&E
153: 4034 1E02       RUI E,RS1     ; SET OFFSET, MATRIX ROW SIZE
154:
155: 4036             ; SUBM1:
156: 4036 86          ADD M          ; A = A + HL
157: 4037 19          DAD D          ; HL = HL + DE (DE = OFFSET)
158: 4038 0D          DCR C          ; ALL TERMS SUMMED ?
159: 4039 C23640     JNZ SUBM1     ; NO
160: 403C D1          POP D          ; YES, RESTORE RESULT ADDR
161: 403D 12          STAX D         ; STORE RESULT
162: 403E 05          DCR B          ; MATRIX RPY COMPLETE ?
163: 403F CA5240     JZ DONE1      ; YES
164: 4042 13          INX D          ; NO, POINT TO NEXT RESULT ADDR
165: 4043 D5          PUSH D         ; ADJUST TERM POINTER TO LAST
166: 4044 210051     LXI H,PMMA1   ; GET BASE ADDR
167: 4047 3E02       RUI A,RS1     ; GET MATRIX ROW SIZE
168: 4049 90          SUB B          ;
169: 404A 5F          MOV E,A       ; GET OFFSET
170: 404B 1600       RUI D,0       ; CLEAR UPPER PORTION OF DE
171: 404D 19          DAD D          ; HL = PMMA + US - REG B
172: 404E D1          POP D          ;
173: 404F C32E40     JMP SUBM1     ;
174:
175: 4052             ; DONE1:
176:
177:
178:             ; MULTIPLY MATRIX PHID (CLOSED LOOP SYSTEM MATRIX) UI
179:             ; VECTOR U (PAST STATE VECTOR).
180:             ; PHID IS 2 ROWS X 2 COLS
181:             ; U IS 2 ROWS X 1 COL
182:
183: 4052 3E02       RUI A,US2     ; ESTABLISH VECTOR SIZE
184: 4054 110054     LXI D,PMMA2   ; PARTIAL MATRIX RPY START ADDR
185: 4057 210053     LXI H,DMA2    ; DATA MATRIX START ADDR
186:
187:             ; COLN2:
188: 405A 405A       PUSH PSU      ; START NEXT COLUMN RPY, SAVE
189: 405B 0602       RUI B,RS2     ; ESTABLISH MATRIX ROW SIZE
190: 405D 4E          MOV C,M       ; GET OPERAND R1
191: 405E 23          INX H         ; POINT TO OPERAND R2
192:
193:             ; ROWN2:
194: 405F 405F       PUSH B        ; RPY THE ELEMENTS IN EACH ROW
195: 4060 E5          PUSH H        ; SAVE B,C,H & L
196: 4061 66          MOV M,M       ; GET OPERAND R2
197: 4062 CD4F41     CALL MULT     ; 8 BIT SIGNED
198: 4065 7C          MOV A,M       ; PREPARE TO STORE
199: 4066 12          STAX D        ; SAVE PARTIAL MATRIX RPY
200: 4067 13          INX D         ; ADJUST SAVE POINTER
201: 4068 E1          POP H         ; RESTORE H & L
202: 4069 23          INX H         ; POINT TO NEXT OPERAND
203: 406A C1          POP B         ; RESTORE VECTOR COUNT (B)
204:             ; RESTORE OPERAND R1 (C)
205: 406B 05          DCR B         ; COLUMN ALL DONE ?
206: 406C C25F40     JNZ ROWN2    ; NO
207: 406F F1          POP PSU       ; YES, RESTORE VECT SIZE
208: 4070 3D          DCR A         ; ALL MULTIPLIES COMPLETE ?
209: 4071 C25A40     JNZ COLN2    ; NO
210:             ; YES, SUM PARTIALS TO COMPLET
211: 4074 0602       RUI B,RS2     ; REESTABLISH MATRIX ROW SIZE
212: 4076 110055     LXI D,PMMA2   ; MATRIX RPY RESULT START ADDR
213: 4079 210054     LXI H,PMMA2   ; PARTIAL MAT RPY START ADDR
214:
215:             ; SUBM2:
216: 407C 407C       PUSH D        ; REINITIALIZE VECTOR SIZE
217: 407E AF          XRA A         ; CLEAR REG A
218: 407F D5          PUSH D        ; OBTAIN POINTER OFFSET IN D&E
219: 4080 1600       RUI D,0       ; CLEAR UPPER PORTION OF D&E
220: 4082 1E02       RUI E,RS2     ; SET OFFSET, MATRIX ROW SIZE
221:
222: 4084             ; SUBM2:
223: 4084 86          ADD M         ; A = A + HL
224: 4085 19          DAD D         ; HL = HL + DE (DE = OFFSET)
225: 4086 0D          DCR C         ; ALL TERMS SUMMED ?

```

```

226: 4087 C28440 JNZ SUMB2 ; NO
227: 408A D1 POP D ; YES, RESTORE RESULT ADDR
228: 408B 12 STAX D ; STORE RESULT
229: 408C 05 DCR B ; MATRIX MPY COMPLETE ?
230: 408D CAA040 JZ DONE2 ; YES
231: 4090 13 INX D ; NO, POINT TO NEXT RESULT ADDR
232: 4091 D5 PUSH D ; ADJUST TERM POINTER TO LAST
233: 4092 210054 LXI H,PNRA2 ; GET BASE ADDR
234: 4095 3E02 RUI A,MS2 ; GET MATRIX ROW SIZE
235: 4097 90 SUB B
236: 4098 5F ROU E,A ; GET OFFSET
237: 4099 1600 RUI D,0 ; CLEAR UPPER PORTION OF DE
238: 409B 19 DAD D ; ML = PNRA + US - REG B
239: 409C D1 POP D
240: 409D C37C40 JNP SUMA2
241:
242:
243: ; DO VECTOR ADDS TO COMPUTE M(K+1) I.E.
244:
245: ; M(K+1)1 = PHIDZU(K)1 + HBSZ(K)1
246: ; . . .
247: ; . . .
248: ;
249: ; M(K+1)N = PHIDZU(K)N + HBSZ(K)N
250: ;
251: ;
252: 40A0 DONE2: LXI H,DMA3 ; GET M(K+1) STORAGE AREA ADDR
253: 40A0 210056 PUSH H ; TEMP SAVE
254: 40A3 E5 LXI H,PNRA1 ; OBTAIN HDEZ RESULT START A
255: 40A4 210052 LXI D,PNRA2 ; OBTAIN PHIDZU RESULT START A
256: 40A7 110055 RUI B,US3 ; OBTAIN # OF TERMS TO ADD
257: 40AA 0602 RUI C,MS3 ; OBTAIN OFFSET TO STORE SUMS
258: 40AC 0E01 INR C
259: 40AE 0C
260:
261: 40AF ; ADDNXT: XRA A ; CLEAR A REG
262: 40AF AF ADD R ; GET GDSZ TERM
263: 40B0 86 XCHG ;
264: 40B1 EB ADD R ; ADD PHIDZU TERM
265: 40B2 86 ADD R ; SET CARRY = 0
266: 40B3 3F STC ;
267: 40B4 3F CMC ;
268: 40B5 17 RAL ; ADJUST FOR 2.0 SCALING
269: 40B6 DAC440 JC NEG ; NEG OR POS ?
270: 40B9 B7 ORA A ; SET CONDITION CODE
271: 40BA F2BF40 JP NOFLO ; POSITIVE, ANY OVERFLOW ?
272: 40BD F67F ORI 7FH ; YES, FORCE TO LARGEST 8
273:
274: 40BF ; NOFLO: ANI 7FH ; FORCE TO POSITIVE 8
275: 40BF E67F JNP SCAL1
276: 40C1 C3C640
277:
278: 40C4 ; NEG: ORI 80H ; FORCE TO NEGATIVE 8
279: 40C4 F680
280:
281: 40C6 ; SCAL1: XTHL ; GET STORAGE ADDRESS
282: 40C6 E3 ROU R,A ; STORE SUM
283: 40C7 77 DCR B ; CHECK IF ALL TERMS ADDED
284: 40C8 05 JZ DONE4 ;
285: 40C9 CAD640 ROU A,L ; ADJUST STORAGE ADDRESS
286: 40CC 7D ADD C
287: 40CD B1 ROU L,A
288: 40CE 8F XTHL ; RESTORE POINTERS
289: 40CF E3 XCHG ;
290: 40D0 EB INX H ; POINT TO NEXT GDSZ TERM
291: 40D1 23 INX D ; POINT TO NEXT PHIDZU TERM
292: 40D2 13 INX B ; ADD NEXT TERMS
293: 40D3 C3AF40 JNP ADDNXT
294:
295:
296: ; UPDATE M(K) WITH NEWLY CALCULATED M(K+1)
297: ;
298: ;
299: 40D6 DONE4: POP H ; RESTORE STACK
300: 40D6 E1

```

```

301: 40D7 2E02      MUI  L,US3      ; GET 8 OF TERMS TO STORE
302: 40D9 E5        PUSH H          ; TEMP SAVE
303: 40DA 210056    LXI  H,DMA3    ; GET U(K+1) STORAGE AREA ADDR
304: 40DD 0E01      MUI  C,RS3     ; GET OFFSET OF U(K+1)
305: 40DF 0C        INR  C         ;
306: 40E0 110053    LXI  B,DMA2    ; GET U(K) STORAGE AREA ADDR
307: 40E3 0502      MUI  B,RS2     ; GET OFFSET OF U(K)
308: 40E5 04        INR  B         ;
309:
310: 40E6          ; STRNXT:
311: 40E6 7E        MOV  A,M       ; GET U(K+1)
312: 40E7 12        STAX D         ; STORE INTO OLD U(K)
313: 40E8 E3        XTHL          ; GET 8 OF TERMS REMAINING TO
314: 40E9 2D        DCR  L         ; ANY LEFT ?
315: 40EA CAF740    JZ   BONES     ; NO
316: 40ED E3        XTHL          ; YES, RESTORE POINTER
317: 40EE 7D        MOV  A,L       ; ADJUST FOR NEXT U(K+1)
318: 40EF 81        ADD  C         ;
319: 40F0 6F        MOV  L,A       ;
320: 40F1 7B        MOV  A,E       ; ADJUST FOR NEXT U(K)
321: 40F2 80        ADD  B         ;
322: 40F3 5F        MOV  E,A       ;
323: 40F4 C3E640    JMP  STRNXT    ;
324:
325:
326:          ;
327:          ; MULTIPLY MATRIX GD (CLOSED LOOP FEEDBACK MATRIX) UI
328:          ; VECTOR U (NEXT STATE VECTOR).
329:          ; GD IS 1 ROW X 2 COLS
330:          ; U IS 2 ROWS X 1 COL
331:
332: 40F7          ; DONES:
333: 40F7 E1        POP  H         ; RESTORE STACK
334: 40F8 3E02      MUI  A,US3     ; ESTABLISH VECTOR SIZE
335: 40FA 110057    LXI  D,PMA3    ; PARTIAL MATRIX NPY START ADDR
336: 40FD 210056    LXI  H,DMA3    ; DATA MATRIX START ADDR
337:
338: 4100          ; COLN3:
339: 4100 F5        PUSH PSU       ; START NEXT COLUMN NPY, SAVE
340: 4101 0601      MUI  B,RS3     ; ESTABLISH MATRIX ROW SIZE
341: 4103 4E        MOV  C,M       ; GET OPERAND 01
342: 4104 23        INX  H         ; POINT TO OPERAND 02
343:
344: 4105          ; ROWN3:
345: 4105 CS        PUSH B         ; NPY THE ELEMENTS IN EACH ROW
346: 4106 E5        PUSH H         ; SAVE B,C,M & L
347: 4107 66        MOV  M,M       ; GET OPERAND 02
348: 4108 CD4F41    CALL MULT     ; 8 BIT SIGNED
349: 4109 7C        MOV  A,M       ; PREPARE TO STORE
350: 410A 12        STAX D         ; SAVE PARTIAL MATRIX NPY
351: 410B 13        INX  D         ; ADJUST SAVE POINTER
352: 410C E1        POP  H         ; RESTORE H & L
353: 410D 23        INX  H         ; POINT TO NEXT OPERAND
354: 410E C1        POP  B         ; RESTORE VECTOR COUNT (B)
355: 410F 05        DCR  B         ; RESTORE OPERAND 01 (C)
356: 4110 05        DCR  B         ; COLUMN ALL DONE ?
357: 4111 05        DCR  B         ; NO
358: 4112 C20541    JNZ  ROWN3    ; YES, RESTORE VECT SIZE
359: 4113 F1        POP  PSU       ; ALL MULTIPLIES COMPLETE ?
360: 4114 3D        DCR  A         ; NO
361: 4115 C20041    JNZ  COLN3    ; YES, SUM PARTIALS TO COMPLET
362: 4116 0601      MUI  B,RS3     ; REESTABLISH MATRIX ROW SIZE
363: 4117 110058    LXI  D,PMA3    ; MATRIX NPY RESULT START ADDR
364: 4118 210057    LXI  H,PMA3    ; PARTIAL MAT NPY START ADDR
365:
366: 4122          ; SUMA3:
367: 4122 0E02      MUI  C,US3     ; REINITIALIZE VECTOR SIZE
368: 4123 AF        XRA  A         ; CLEAR REG A
369: 4124 D5        PUSH D         ; OBTAIN POINTER OFFSET IN D&E
370: 4125 D5        PUSH D         ; CLEAR UPPER PORTION OF D&E
371: 4126 1600      MUI  D,0       ; SET OFFSET, MATRIX ROW SIZE
372: 4127 1E01      MUI  E,RS3     ;
373:
374: 412A          ; SURB3:
375: 412A 86        ADD  M         ; A = A + ML
376: 412B 19        DAD  D         ; ML = ML + DE (DE = OFFSET)
377: 412C 0D        DCR  C         ; ALL TERMS SUMMED ?

```

```

376: 412D C22A41 JNZ SUMB3 ; NO
377: 4130 D1 POP D ; YES, RESTORE RESULT ADDR
378: 4131 12 STAX B ; STORE RESULT
379: 4132 05 DCR B ; MATRIX MPY COMPLETE ?
380: 4133 CA4641 JZ DONE3 ; YES
381: 4136 13 INX D ; NO, POINT TO NEXT RESULT ADDR
382: 4137 D5 PUSH D ; ADJUST TERM POINTER TO LAST
383: 4138 210057 LXI H,PMMA3 ; GET BASE ADDR
384: 413D 3E01 MUI A,MS3 ; GET MATRIX SIZE
385: 413D 90 SUB B
386: 413E 5F MOV E,A ; GET OFFSET
387: 413F 1600 MUI D,0 ; CLEAR UPPER PORTION OF DE
388: 4141 19 DAD D ; HL = PMMA + US - REG B
389: 4142 D1 POP D
390: 4143 C32241 JMP SUMA3
391:
392: 4146 ;
393: 4146 37 ; DONE3:
394: 4147 3F STC ; SET CARRY = 0
395: 4148 FB CMC ; CHECK FOR INTERRUPT COMPLETI
; NEEDED FOR 1ST PASS ONLY
396:
397: 4149 ;
398: 4149 D24941 ; WAITLP:
399: 414C C30440 JNC WAITLP ; WAIT FOR INTERRUPT
; INTERRUPT SERVICED
; CALCULATE NEXT OUTPUT
400:
401:
402: ; SUBROUTINE 'MULT' --- 8 BIT SIGNED MULTIPLY
403:
404: ; HL = H&C
405:
406: ; INPUTS: C - MULTIPLICAND 8 BIT SIGNED
407: ; H - MULTIPLIER 8 BIT SIGNED
408:
409: ; OUTPUTS: H&L - PRODUCT 16 BIT SIGNED
410:
411: ; DESTROYS: A,B,C,H,L
412:
413: ;
414: 414F ; MULT:
415: 4150 7C MOV A,H ; CHECK SIGN OF MULTIPLIER (H)
416: 4151 B7 ORA A
417: 4154 F26E41 JP MULMP ; H IS POSITIVE
418: 4155 3C CMA ; MULTIPLIER (H) IS NEGATIVE
419: 4156 67 INR A ; TAKE 2'S COMPLIMENT
420: 4157 79 MOV A,C ; CHECK SIGN OF MULTIPLICAND
421: 4158 B7 ORA A
422: 4159 F26341 JP MULOS ; INPUTS HAVE OPPOSITE SIGNS
423: 415C 2F CMA ; MULTIPLICAND (C) IS
424: 415D 3C INR A ; TAKE 2'S COMPLIMENT
425: 415E 4F MOV C,A
426:
427: ;
428: 415F ; MULSS:
429: 4162 CD7941 CALL INUL ; SAME SIGN, MULTIPLY AND RETU
430: 4162 C9 RET
431:
432: ;
433: 4163 ; MULOS:
434: 4163 CD7941 CALL INUL ; H & C HAVE OPPOSITE SIGNS
435: 4166 2B DCX H ; TAKE 2'S COMPLIMENT OF PRODU
436: 4167 7D MOV A,L
437: 4168 2F CMA
438: 4169 6F MOV L,A ; 2'S COMP OF L
439: 416A 7C MOV A,H
440: 416B 2F CMA
441: 416C 67 MOV H,A ; 2'S COMP OF H
442: 416D C9 RET ; RETURN WITH FINAL RESULT IN
443:
444: ;
445: 416E ; MULMP:
446: 416E 79 MOV A,C ; H (MULTIPLIER) IS POSITIVE
447: 416F B7 ORA A ; CHECK SIGN OF MULTIPLICAND
448: 4170 F25F41 JP MULSS ; MULTIPLICAND (C) IS NEGATIVE
449: 4173 2F CMA ; TAKE 2'S COMPLIMENT
450: 4174 3C INR A
451: 4175 4F MOV C,A
452: 4176 C36341 JMP MULOS ; DO OPPOSITE SIGN MULTIPLY

```

```

451:                                     ;
452:                                     ; SUBROUTINE 'INUL' --- 8 BIT UNSIGNED FRACTIONAL
453:                                     ;
454:                                     ; INPUTS: C - MULTIPLICAND           8 BIT UNSIGNED
455:                                     ;           N - MULTIPLIER           8 BIT UNSIGNED
456:                                     ;
457:                                     ; OUTPUTS: ML - PRODUCT           16 BIT UNSIGNED
458:                                     ;
459:                                     ; DESTROYS: A,B,M,L
460:                                     ;
461: 4179                                INUL:
462: 4179 0600                            RUI  B,0           ; CLEAR FOR FOLLOWING 'DAD' IN
463: 417B 68                               ROU  L,B           ; CLEAR BOTTOM HALF OF ML
464: 417C 3E08                            RUI  A,B           ; INITIALIZE LOOP COUNTER
465:                                     ;
466: 417E                                INUL1:
467: 417E 29                               DAD  H           ; SHIFT RESULT
468: 417F D28341                          JNC  INUL2       ; IF MSB SET, ADD MULTIPLICAND
469: 4182 09                               DAD  B           ; ML = ML + BC
470:                                     ;
471: 4183                                INUL2:
472: 4183 3D                               DCR  A           ; DECREMENT & TEST LOOP COUNT
473: 4184 C27E41                          JNZ  INUL1       ;
474: 4187 29                               DAD  H           ; ADJUST FOR FRACTIONAL RPY
475: 4188 C9                               RET
476:                                     ;
477:                                     ;
478:                                     ; INTERRUPT SERVICE ROUTINE
479:                                     ;
480:                                     ; FUNCTIONAL DESCRIPTION:
481:                                     ;
482:                                     ; THIS ROUTINE IS ENTERED WHEN THE CLOCK (DELTA T)
483:                                     ; GOES OFF. DELTA T IS A SQUARE WAVE CLOCK INPUT
484:                                     ; FROM A FUNCTION GENERATOR. HENCE IT IS PRESETTABLE
485:                                     ; IT PROVIDES:
486:                                     ;
487:                                     ; A) A DELTA T TIME STEP
488:                                     ; B) A MEANS OF INDICATING END OF CONVERSION FOR
489:                                     ;
490:                                     ; A2D MAX CONVERSION TIME IS .05 MS. HENCE DELTA T
491:                                     ; BE SET HIGHER
492:                                     ;
493:                                     ; THE ROUTINE READS IN A 12 BIT A2D INPUT Z(K), WHICH
494:                                     ; TRUNCATES THE LEAST SIGNIFICANT 4 BITS SINCE ONLY 8
495:                                     ; BITS ARE NEEDED.
496:                                     ;
497:                                     ; THE ROUTINE ALSO OUTPUTS U(K) TO THE ANALOG SYSTEM
498:                                     ; MEANS OF A D2A. THE D2A IS 8 BIT MEMORY MAP IO.
499:                                     ; 2 OUTPUTS ARE PROVIDED: 1 TO THE SYSTEM
500:                                     ;                               1 TO A STRIP CHART RECORDER
501:                                     ;
502:                                     ;
503:                                     ; IN ADDITION, A CHECK IS DONE TO SEE IF THE
504:                                     ; INTERRUPT OCCURRED DURING CONTROL CODE
505:                                     ; CALCULATIONS, WHICH COULD RESULT IN
506:                                     ; INACCURATE CONTROL COMMANDS. IF THIS
507:                                     ; ERROR OCCURS, THEN PGM CONTROL IS
508:                                     ; PASSED TO THE MONITOR.
509: 4189                                INTR:
510: 4189 F3                               DI           ; DISABLE INTERRUPTS
511: 418A F5                               PUSH  PSU     ; SAVE A
512: 418B E5                               PUSH  M       ; SAVE ML
513:                                     ;
514:                                     ; CHECK IF CONTROL CODE COMPLETED
515:                                     ;
516: 418C 33                               INX  SP       ; CHECK IF CONTROL CODE COMPLE
517: 418D 33                               INX  SP       ; FIND RET ADDR FROM INTERRUPT
518: 418E 33                               INX  SP
519: 418F 33                               INX  SP
520: 4190 214941                          LXI  M,UAITLP ; GET WAIT LOOP HI ADDR
521: 4193 7C                               ROU  A,M
522: 4194 E3                               XTHL        ; GET INTERRUPT RET HI ADDR
523: 4195 94                               SUB  M        ; ARE THEY EQUAL ?
524: 4196 C8D841                          JNZ  ERR1     ; NO, INDICATE ERR, STOP PGM
525: 4199 7D                               ROU  A,L     ; GET INTERRUPT RET LO ADDR

```

```

526: 419A E3      XTHL      ; GET WAIT LOOP LO ADDR
527: 419B 95      SUB       L      ; ARE THEY EQUAL ?
528: 419C C2D841   JNZ      ERR1   ; NO, INDICATE ERR, STOP PGM
529: 419F 3B      DCX      SP     ; YES, CONTROL CODE WAS COMPLE
530: 41A0 3B      DCX      SP     ; RESTORE STACK PNTR TO NORMAL
531: 41A1 3B      DCX      SP
532: 41A2 3B      DCX      SP
533:
534:                ; READ A/D
535:
536: 41A3 3E82     MVI      A,82H  ; SET A/D READ CHANNEL
537: 41A5 D3E7     OUT      0E7H  ; AMPLIFIER INPUT
538: 41A7 DBE5     IN       0E5H  ; READ LOW BYTE
539: 41A9 0F      RRC      ; ADJUST FOR 4 BIT THROUAWAY
540: 41AA 0F      RRC
541: 41AB 0F      RRC
542: 41AC 0F      RRC
543: 41AD E60F     ANI      0FH    ; PRESERVE LOW NIBBLE
544: 41AF 6F      MOV      L,A    ; TEMP SAVE
545: 41B0 DBE4     IN       0E4H  ; READ HIGH BYTE
546: 41B2 0F      RRC      ; ADJUST FOR HIGH NIBBLE
547: 41B3 0F      RRC
548: 41B4 0F      RRC
549: 41B5 0F      RRC
550: 41B6 E6F0     ANI      0F0H  ; PRESERVE HIGH NIBBLE
551: 41B8 B5      ORA      L      ; MERGE TO FORM 8 BIT INPUT
552: 41B9 210050   LXI     M,DMA1 ; STORE INPUT IN MATRIX DATA A
553: 41BC 77      MOV      H,A
554:
555:                ;
556:                ; OUTPUT D/A (U)
557:                ; MEMORY MAPPED I/O
558:                ;
559:                ;
560: 41BD 21005B   LXI     H,UOUT  ; OUTPUT U
561: 41C0 7E      MOV      A,M
562: 41C1 2100F7   LXI     H,0F700H
563: 41C4 77      MOV      H,A
564: 41C5 210053   LXI     H,STRIPC ; OUTPUT MEMORY LOC TO STRIP C
565: 41C8 7E      MOV      A,M
566: 41C9 2101F7   LXI     H,0F701H
567: 41CC 77      MOV      H,A
568: 41CD E1      POP      H
569: 41CE F1      POP      PSU
570: 41CF 37      STC      ; INDICATE INTERRUPT COMPLETE
571: 41D0 FB      EI
572: 41D1 C9      RET      ; ENABLE INTERRUPTS & RETURN
573:
574:                ; CONTROL CODE NOT COMPLETED, ERROR
575:                ;
576: 41D2          ERR1:
577: 41D2 CF      RST     1      ; BRANCH TO MONITOR IMMEDIATLY
578:
579:                ;
580:                ; DATA MATRIX STORAGE AREA
581:                ;
582:                ; THE FOLLOWING DATA IS FOR THE 2ND ORDER SYSTEM
583:                ; ALL 8'S ARE REPRESENTED AS FRACTIONS WHERE:
584:                ; +8 = FRAC.812B+0.5
585:                ; -8 = 2'S COMP OF (-FRAC.812B+0.5)
586:                ;
587:                ; *****
588:                ;
589:                ; NOTE: GAINS BELOW ARE FOR T = 0.1 SEC
590:                ;
591:                ; *****
592:                ;
593:                ; Z - INPUT VECTOR FROM ASD
594:                ; MD - MATRIX
595:                ;
596:                ; MD = .113 = .057 WHEN SCALED ON 8.0
597:                ;     -.036 = -.018
598:                ;
599: 5000          ORG     5000H
600:

```

```

601: ; NOTE: MD(1) & MD(2) HAVE BEEN COMPLIMENTED
602: ; DUE TO INVERTED INPUT.
603: ;
604: 5000 00 ; DB 0H ; Z(K) VECTOR A/D INPUT 01
605: 5001 08 ; DB 0F9H ; MD MATRIX ROW1 COL1
606: 5002 16 ; DB 03H ; MD MATRIX ROW2 COL1
607: ;
608: ;
609: ; U(K) - PAST STATE VECTOR
610: ; PHID - CLOSED LOOP SYSTEM MATRIX
611: ;
612: ; PHID = .87264 .07127 0 .43632 .03563
613: ; -.86524 .63205 0 -.13262 .31603
614: ; ;
615: ; WHEN SCALED ON 2.0
616: 5300 ; ORG 5300H
617: 5300 40 ; DB 40H ; U(K) PAST STATE 01 INIT .5
618: 5301 38 ; DB 38H ; PHID MATRIX ROW1 COL1
619: 5302 F0 ; DB 0F0H ; PHID MATRIX ROW2 COL1
620: 5303 40 ; DB 40H ; U(K) PAST STATE 02 INIT .5
621: 5304 05 ; DB 05H ; PHID MATRIX ROW1 COL2
622: 5305 28 ; DB 28H ; PHID MATRIX ROW2 COL2
623: ;
624: ;
625: ; U(K+1) - PRESENT STATE VECTOR
626: ; GD - CONTROL GAIN MATRIX
627: ;
628: ; GD = -1.742 -.41412 0 -.871 -.20706
629: ; ;
630: ; WHEN SCALED ON 2.0
631: 5600 ; ORG 5600H
632: 5600 00 ; DB 0H ; U(K+1) PRESENT STATE 01
633: 5601 02 ; DB 02H ; GD MATRIX ROW1 COL1
634: 5602 00 ; DB 0H ; U(K+1) PRESENT STATE 02
635: 5603 E6 ; DB 0E6H ; GD MATRIX ROW1 COL2
636: ;
637: ;
638: ; END
639: NO PROGRAM ERRORS
640: ;
641: ;
642: ;
643: ;
644: ; 01
645: ;
646: A 0007 ADDNX 40AF B 0000 C 0001
647: COLN1 400C COLN2 405A COLN3 4100 D 0002
648: DMA1 5000 DMA2 5300 DMA3 5600 DONE1 4052
649: DONE2 40A0 DONE3 4146 DONE4 40DG DONE5 40F7
650: E 0003 ERR1 41D2 H 0004 INUL 4179
651: INUL1 417E INUL2 4183 INTR 4189 L 0005
652: N 0006 NARA1 5200 NARA2 5500 NARA3 5800
653: NS1 0002 NS2 0002 NS3 0001 NULNP 416E
654: NULOS 4163 NULSS 415F NULT 414F NEG 40C4
655: NOFLO 40BF PRA1 5100 PRA2 5400 PRA3 5700
656: PSU 0006 ROUN1 4011 ROUN2 405F ROUN3 4105
657: SCAL1 40C6 SP 0006 STRIP 5300 STRNX 40E6
658: STRT1 4004 SURA1 402E SURA2 407C SURA3 4122
659: SURB1 4036 SURB2 4084 SURB3 412A UOUT 5800
660: US1 0001 US2 0002 US3 0002 MATL 4149
661: ;

```

SYMBOL TABLE

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO BDC

APPENDIX C

LSI-11 SOFTWARE FOR LQG CONTROLLER

KALMAN FILTER/CONTROLLER

PROJECT: MICROPROCESSOR IMPLEMENTATION OF MODERN CONTROL
PROGRAMMER: J. KRODEL, DIGITAL COMPUTER LAB
DATE: 15-APR-80
VERSION: 00.00
REVISION: 00.00

FUNCTIONAL DESCRIPTION:

THIS PROGRAM CONTROLS A 5TH
ORDER SYSTEM USING MODERN CONTROL
METHODS. THE BASIC EQUATIONS FOLLOW

$$\hat{W}_{K+1} = PHID * \hat{W}_K + HD * Z_K$$

$$\hat{W}_K = \hat{W}_{K+1}$$

$$U_{K+1} = GD * \hat{W}_{K+1}$$

WHERE:

Z_K = SYSTEM MEASUREMENT VECTOR

HD = KALMAN FILTER GAIN MATRIX

\hat{W}_K = PAST STATE ESTIMATE VECTOR

$PHID$ = CLOSED LOOP SYSTEM MATRIX

\hat{W}_{K+1} = NEXT STATE ESTIMATE VECTOR

GD = CLOSED LOOP FEEDBACK GAIN MATRIX

U_{K+1} = SYSTEM INPUT VECTOR

FOR THE FIFTH ORDER SYSTEM:

Z = 5X1 VECTOR

HD = 5X5 MATRIX

\hat{W} = 5X1 VECTOR

$PHID$ = 5X5 MATRIX

GD = 4X5 MATRIX

U = 4X1 VECTOR

REVISION HISTORY:

ANALOG INTERFACE:

THIS INTERFACE ALLOWS THE USER TO INVESTIGATE BOTH THE
ANALOG AND DIGITAL COMPUTER STATES AT ANY POINT IN
TIME OF THE SIMULATION/CONTROLLER. THIS
PROGRAM ACTS AS A SLAVE TO THE ANALOG MACHINE WITH

RESPECT TO 'MODE' OPERATION. THE 3 MODES ON THE ANALOG COMPUTER ARE:

I.C. (INITIAL CONDITIONS)
OP (OPERATE)
HLD (HOLD)

THE OVERLOAD CONDITION WILL ALSO BE HANDLED.

WHEN IN I.C. MODE THE PROGRAM DECIPHERS THIS MODE, AND RESETS THE INTERNAL CLOCK AND AWAITS FOR THE OPERATE MODE. ONCE IN OPERATE MODE, THE PROGRAM ALLOWS THE CLOCK TO RUN FREE AND CONTROL OF SIMULATION BEGINS.

IF THE HOLD MODE IS ENTERED, THE CLOCK IS IMMEDIATELY HALTED, AND THE PROGRAM CAN BE SET TO BREAKPOINT UNDER ODT TO EXAMINE THE PRESENT STATUS OF THE LSI-11 CONTROLLER. LIKEWISE, ANY INFORMATION OF THE SIMULATOR CAN BE OBTAINED VIA THE ANALOG COMPUTERS DIGITAL KEY PAD. WHEN THE OPERATE MODE IS LATER ENTERED THE CLOCK STARTS AGAIN AT PRECISELY THE POINT IT WAS STOPPED.

THE ANALOG COMPUTER HAS THE CAPABILITY OF GOING INTO THE HOLD STATE WHEN ANY AMPLIFIER IS OVERLOADED. IF THE SYSTEM BECOMES OVERLOADED THEN AGAIN THE INTERNAL CLOCK IS HALTED, AND LIKEWISE THE PROGRAM CAN BE HALTED UNDER ODT CONTROL.

NOTE: FOR CORRECT OPERATION OF THIS CODE, PATCHES MUST BE MADE TO THE ANALOG COMPUTER. BELOW ARE LISTED THE NECESSARY PATCHES.

UPF = UNIVERSAL PATCH PANEL
HIO = HYBRID I/O PATCH PANEL
DPP = DIGITAL PATCH PANEL

UPF.OVL - UPF.HOLD *ALLOWS HOLD STATE WHEN OVERLOAD OCCURS
DPP.CLK - HIO.RQSTA *INTERRUPT FOR I/O UPDATE
UPF.ABAR - HIO.LSI-11 DIGITAL INPUT 1 *ALLOWS CODE TO DETECT....
UPF.A - HIO.LSI-11 DIGITAL INPUT 0 *....ANALOG COMPUTER MODE.

NOTE: CLK BUS DOES NOT WORK UNLESS IN OPERATE MODE.

ABAR A ! ANALOG COMPUTER MODE

0 0 ! DON'T CARE
0 1 ! I. C. MODE
1 0 ! OPERATE MODE
1 1 ! HOLD MODE

EXTERNAL GLOBL

.GLOBL HBRKPT ; SYSTEM IN HOLD ODT BREAK POINT

EQUATES

DIOCSR = 170000 ; LSI-11 DIGITAL INPUT/OUTPUT STATUS REG
DIOIN = 170004 ; LSI-11 DIGITAL INPUT REG
TIMPSW = 340 ; CLOCK INTERRUPT PSW
DAOUT0 = 176750 ; D/A OUTPUT REG
DAOUT1 = 176752 ; D/A OUTPUT REG
DAOUT2 = 176754 ; D/A OUTPUT REG
DAOUT3 = 176756 ; D/A OUTPUT REG

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COLLECTOR TO DDC


```

; **** SYSTEM IN HOLD ****
;   CLOCK STOPS AUTOMATICALLY (OUT OF OPERATE MODE)
;   ASSUME ODT BREAKPOINT AT 'HBRKPT'
;   IF NO BREAK POINT SET, LSI-11 IS STILL IN HOLD STATE
;   UNTIL USER PRESSES OPERATE.
;
HOLD:
;
HBRKPT:
    MOV     #1,HLDFLG      ; INDICATE IN HOLD MODE
    JMP     START
;
; **** SYSTEM IN OPERATION ****
;
OPERAT:
    CLR     HLDFLG        ; INDICATE NOT IN HOLD MODE
    CLR     TIFLAG        ; INIT TIMER INTERRUPT FLAG
;
; WAIT FOR TIMER INTERRUPT
;
TIMEWT:
    MOV     @#DIIOIN,ABAF  ; IC OR HOLD KEY PRESSED ?
    BIC     #177776,ABAF   ; A INDICATES IN I, C, OR HOLD
    BEQ     NOICH          ; NO I, C, OR HOLD MODE PRESSED
    JMP     START          ; YES, GOTO START
;
NOICH:
    CMP     #1,TIFLAG      ; TIMER INTERRUPT YET ?
    BNE
;
; MULTIPLY MATRIX HD (KALMAN FILTER GAIN MATRIX) WITH
;   VECTOR Z (SYSTEM MEASUREMENT VECTOR).
;   HD IS 5 ROWS X 5 COLS
;   Z IS 5 ROWS X 1 COL
;
; INITIALIZATION
;
    MOV     #VS1,R0        ; CLEAR MATRIX MULT. RESULT AREA
    MOV     #MMRA1,R2
;
INI1:
    CLR     (R2)+
    DEC     R0
    BNE     INI1
    MOV     #VS1,COLCNT    ; ESTABLISH VECTOR SIZE
    MOV     #DMA1,R1       ; DATA MATRIX START ADDR
;
COLN1:
    MOV     #MS1,R0        ; ESTABLISH MATRIX ROW SIZE
    MOV     #MMRA1,R2      ; INIT. RESULT ADDR.
    MOV     (R1)+,R3       ; GET OPERAND #1
;
ROWN1:
    MOV     (R1)+,R4       ; GET OPERAND # 2
    JSR     PC,MULT        ; 16 BIT SIGNED
;
; SCALE
;
    ADD     R4,(R2)+       ; SUM CORRESPONDING TERMS

```

```

CHECK FOR ADDITION OVERFLOW

    DEC    R0                ; THIS COLUMN ALL DONE ?
    BNE    ROWN1            ; NO
    DEC    COLCNT           ; ALL MULTIPLIES COMPLETE ?
    BNE    COLM1            ; NO

MULTIPLY MATRIX PHID (CLOSED LOOP SYSTEM MATRIX) WITH
VECTOR W (PAST STATE VECTOR).
PHID IS 5 ROWS X 5 COLS
W IS 5 ROWS X 1 COL

INITIALIZATION

    MOV    #MS2,R0          ; CLEAR MATRIX MULTI. RESULT AREA
    MOV    #MMR42,R2

INIT:
    CLR    (R2)+
    DEC    R0
    BNE    INIT
    MOV    #VS2,COLCNT     ; ESTABLISH VECTOR SIZE
    MOV    #DM42,R1        ; DATA MATRIX START ADDR.

COLM2:
    MOV    #MS2,R0          ; ESTABLISH MATRIX ROW SIZE
    MOV    #MMR42,R2        ; INIT. RESULT ADDR.
    MOV    (R1)+,R3        ; GET OPERAND #1

ROWM2:
    MOV    (R1)+,R4        ; GET OPERAND #2
    JSR    PC,MULT         ; 16 BIT SIGN 0

SCALE
    ADD    R4,(R2)+        ; SUM CORRESPONDING TERMS

CHECK FOR ADDITION OVERFLOW

    DEC    R0                ; THIS COLUMN ALL DONE ?
    BNE    ROWN2            ; NO
    DEC    COLCNT           ; ALL MULTIPLIES COMPLETE ?
    BNE    COLM2            ; NO

DO VECTOR ADDS TO COMPUTE W(N+1) I.E.
W(N+1)1 = PHID*W(N)1 + HD*Z(N)1
      .
      .
W(N+1)N = PHID*W(N)N + HD*Z(N)N

AND

UPDATE W(N) WITH NEWLY CALCULATED W(N+1)

    MOV    #DM42,R1        ; W(N+1) STORAGE AREA ADDR.
    MOV    #MMR41,R2        ; LD#2 RESULT START ADDR.
    MOV    #MMR42,R3        ; PHID*W(N)1 STORAGE ADDR.

```


SUBROUTINE 'DIVIDE' --- 16 BIT SIGNED DIVIDE

DIVIDE:

```

MOV    R3,RS          ; SAVE R3 VALUE
CLR    SGNFLG         ; INIT SIGN FLAG
TST    R5              ; CHK SIGN OF 1ST OPERAND
RFL    DIVD1          ; T1 POSITIVE
NEG    R5              ; 2'S COMPLIMENT T1
COM    SGNFLG         ; COMPLIMENT SIGN FLG

```

DIVD1:

```

TST    R1              ; CHK SIGN OF 2ND OPERAND
RFL    DIVD2          ; T2 POSITIVE
NEG    R1              ; 2'S COMPLIMENT T2
COM    SGNFLG         ; COMPLIMENT SIGN FLAG

```

DIVD2:

```

DIV    RS,R0          ; DIVIDE OPERANDS
TST    SGNFLG         ; CHK FOR PROPER SIGN
BEQ    DIVD3          ; NEGATIVE SIGN, TAKE 2'S COMPLIMENT
NEG    R0

```

DIVD3:

```

RTE    R0

```

SGNFLG: WORD 0 ; SIGN FLAG FOR MULT ROUT.

INTERMIT SERVICE ROUTINE

FUNCTIONAL DESCRIPTION:

THIS ROUTINE IS ENTERED WHEN THE CLOCK (DELTA T) GOES OFF. DELTA T IS A SQUARE WAVE CLOCK INPUT FROM THE ANALOG CHANNEL. HENCE IT IS PREDICTABLE AND IT PROVIDES A DELTA T TIME STEP.

AT THIS TIME ALL A/D CHANNELS ARE READY AND APPROPRIATELY PLACED INTO THE DATA MATRIX.

TIME1:

```

MOV    #ADCH0,#ADCSR ; START A/D CONVERSION
ADD    #1,CNT         ; RAMP INTERRUPT COUNTER
MOV    R0,-RFP       ; PUSH R0 ON STACK
MOV    R3,-RFP       ; PUSH R3 ON STACK
CLR    R0             ; TABLE INDEX POINTER
MOV    #1,TIFLAG     ; SET TIME INTERRUPT FLAG TRUE

```

TIME2:

```

TST    #ADCSF        ; CONVERSION COMPLETE ?
RFL    NDEG10        ; NO
MOV    #ADCH0,R0     ; YES, READ IN
RFL    #ADCH1,#ADCSF ; START NEXT A/D CONVERSION
MOV    R0,SCALAD     ; SCALE THIS A/D INPUT

```

TIME3:

```

TST    #ADCSF        ; CONVERSION COMPLETE ?
RFL    NDEG11        ; NO
MOV    #1,R0         ; YES, READ A/D
MOV    #ADCH0,#ADCSR ; START NEXT A/D CONVERSION
MOV    R0,SCALAD     ; SCALE THIS A/D INPUT

```

TIME4:

```

TST    #ADCSF        ; CONVERSION COMPLETE ?
RFL    NDEG12        ; NO
MOV    #1,R0         ; YES, READ A/D
MOV    #ADCH0,#ADCSR ; START NEXT A/D CONVERSION

```

```

NOE003: JSR      PC,SCALAD      ; SCALE THIS A/D INPUT
        TSTB   @#ADCSF      ; CONVERSION COMPLETE ?
        BPL    NOE003        ; NO
        MOV    @#ADBUF,R3    ; YES, READ A/D
        MOV    #ADCH4,@#ADCSF ; START NEXT A/D CONVERSION
        JSR    PC,SCALAD      ; SCALE THIS A/D INPUT

NOE004: JSR      PC,SCALAD      ; SCALE THIS A/D INPUT
        TSTB   @#ADCSF      ; CONVERSION COMPLETE ?
        BPL    NOE004        ; NO
        MOV    @#ADBUF,R3    ; YES, READ A/D
        JSR    PC,SCALAD      ; SCALE THIS A/D INPUT
        MOV    (SP)+,R3      ; RESTORE REG 3
        MOV    (SP)+,R2      ; RESTORE REG 2
        RTI

```

SCALAD -- SCALE A/D INPUT ROUTINE

```

SCALAD:
        ROL    R3            ; GET A/D SIGN BIT INTO COMP. SIGN BIT
        ROL    R3
        ROL    R3
        ROL    R3
        ROL    R3
        CMF    #1,R3        ; CLEAR EXTRANEIOUSLY ROTATED BITS
        ROL    #20480,R3    ; A/D INPUT HAS +/- 1 VOLT RANGE
        ADERR  #20480,R3    ; 20480 = (10/50)*10240TS*50*E
        ROL    R3
        ROL    R3
        MOV    R3,DMAC(R2)  ; STORE RESULT
        ADD    #14,R2      ; ADJUST POINTER IN TABLE

SCALRET:
        RTS                PC

ADERR:
        NOP
        JMP    SCALRET      ; SET UP BKPT HERE

```

STORAGE AREA

```

ABAP1: .WORD 0      ; ABAP A FLAG INDICATOR
HOLD1: .WORD 0      ; ANALOG SIGNAL HOLD FLAG
TIFLAG: .WORD 0     ; TIMER INTERRUPT INDICATOR FLAG
ICNT: .WORD 0       ; INTERRUPT COUNTER
COLCNT: .WORD 0     ; COLUMN COUNTER

```

DATA MATRIX AREA #1 (HDVZ)

```

DELTA T = 0.025 SEC
I. 1.1 = 0.1

```

SCALING:
X = FRAC * 32768

```

IMA1: .WORD 6715 ; HI 1.1 = 0.1
      .WORD 31614 ; HI 1.1 = 0.1
      .WORD 5261  ; HI 1.1 = 0.1
      .WORD 12243  ; HI 1.1 = 0.1
      .WORD 66    ; HI 1.1 = 0.1
      .WORD 45    ; HI 1.1 = 0.1
      .WORD 8311  ; YES
      .WORD 3003  ; HI 1.1 = 0.1
      .WORD 1521  ; HI 1.1 = 0.1
      .WORD 340   ; HI 1.1 = 0.1
      .WORD 45    ; HI 1.1 = 0.1
      .WORD 16    ; HI 1.1 = 0.1

```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM GSA FPMR (41 CFR) 101-11.6

```

.WORD 6315 : Y(3) = Y
.WORD 40711 : HD(1,3) = Y
.WORD 23344 : HD(2,3) = Y
.WORD 54555 : HD(3,3) = Y
.WORD 33170 : HD(4,3) = Y
.WORD 34 : HD(5,3) = Y
.WORD 65515 : Y(4) = Y
.WORD 33744 : HD(1,4) = Y
.WORD 77777 : HD(2,4) = Y
.WORD 40 : HD(3,4) = Y
.WORD 47 : HD(4,4) = Y
.WORD 17774 : HD(5,4) = Y
.WORD 6551 : Y(5) = Y
.WORD 36 : HD(1,5) = Y
.WORD 16 : HD(2,5) = Y
.WORD 55 : HD(3,5) = Y
.WORD 1 : HD(4,5) = Y
.WORD 1 : HD(5,5) = Y

```

```

DATA MATRIX AREA #1 (PHI#0000)
MULTI RESULT AREA #1

```

```

.WORD 6315 : PH(1) = 0.1
.WORD 40711 : PH(2) = 0.1
.WORD 23344 : PH(3) = 0.1
.WORD 54555 : PH(4) = 0.1
.WORD 33170 : PH(5) = 0.1
.WORD 34 : PH(6) = 0.2
.WORD 65515 : PH(7) = 0.2
.WORD 33744 : PH(8) = 0.1
.WORD 77777 : PH(9) = 0.1
.WORD 40 : PH(10) = 0.1
.WORD 47 : PH(11) = 0.1
.WORD 17774 : PH(12) = 0.1
.WORD 6551 : PH(13) = 0.1
.WORD 36 : PH(14) = 0.1
.WORD 16 : PH(15) = 0.1
.WORD 55 : PH(16) = 0.1
.WORD 1 : PH(17) = 0.1
.WORD 1 : PH(18) = 0.1

```

```

DATA MATRIX AREA #2 (PHI#0001)
MULTI RESULT AREA #2

```

```

.WORD 0 : WH(1,1) = 1
.WORD 6315 : CP(1,1) = 1
.WORD 231 : CP(2,1) = 1
.WORD 16417 : CP(3,1) = 1
.WORD 100434 : CP(4,1) = 1

```


Distribution List
Report R80-944590-1
Contract F49620-79-C-0078

Director of Mathematical and Information
Sciences
Bldg. 410
Bolling AFB, Washington, D. C. 20332
Attn: Lt. Col. George W. McKemie (16)

ASD/XRT
Wright-Patterson AFB, OH 45433
Attn: Lt Col W. Othling

OASD/PA&E
Pentagon
Washington, DC 20360
Attn: Mr. T. P. Christie

NASA-Flight Research Center
Edwards AFB, CA 93523
Attn: Mr. H. A. Rediess

SAMSO/YAD
Technology-Guidance & Control
Los Angeles AFS, CA 90045

NASA-Ames Research Center
Moffett Field, CA 94035
Attn: Mr. Elwood C. Stewart

Naval Ocean Systems Center
Tactical Command Control Division
San Diego, CA 92152
Attn: Dr. Robert Kolb, Code 824

NASA-Langely Research Ctr
Hampton, VA 23365
Attn: Mr. Larry W. Taylor, Jr.

AFIT/ENE
Wright-Patterson AFB, OH 45433
Attn: Capt. Gary Reid

Office of Naval Research
Technology Group
800 North Quincy St.
Arlington, VA 22217
Attn: Mr. David Siegel

AFIT/ENE
Wright-Patterson AFB, OH 45433
Attn: Capt James Negro

AFFDL/FGL
Wright-Patterson AFB, OH 45433
Attn: Paul Blatt

RADC/ISCP
Griffiss AFB, NY 13441
Attn: Heywood Webb

ARO
P.O. Box 12,211
Research Triangle Park, NC 27709
Attn: Dr. J. Chandra

Office of Naval Research
Mathematics Group
800 North Quincy Street
Arlington, VA 22217
Attn: Dr. Stewart Brodsky

AFAPL/TBC
Wright-Patterson AFB, OH 45433
Attn: Mr. Charles Skira

AFATL/DLY
Eglin AFB, FL 32542
Attn: Dr. Jesse Gonzales

AFWL/LRO
Kirtland AFB, NM 87117
Attn: Lt Col Dale Neal

David Taylor Naval Ship R&D Center
Code 2730
Annapolis, MD 21404
Attn: Mr. Walter J. Blumberg

AFWL/AL
Kirtland AFB, NM 87117
Attn: Maj Kenneth Herring

DATE
L MED
— 8