

1230

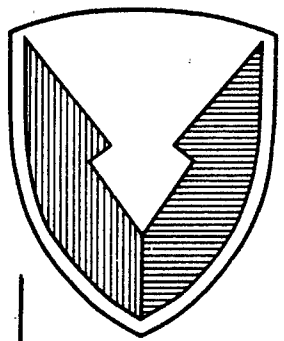
2268

ADA 204667

R D & E

C E N T E R

Technical Report



No. 13419

HYDRAULIC CONTROL DESIGN
AND MODELING TECHNIQUES
FEBRUARY 1989

By Arthur L. Helinski
U.S. Army Tank-Automotive Command
ATTN: AMSTA-RYA
Warren, MI 48397-5000

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION IS UNLIMITED

2003 1212 004

U.S. ARMY TANK-AUTOMOTIVE COMMAND
RESEARCH, DEVELOPMENT & ENGINEERING CENTER
Warren, Michigan 48397-5000

NOTICES

This report is not to be construed as an official Department of the Army position.

Mention of any trade names or manufacturers in this report shall not be construed as an official endorsement or approval of such products or companies by the U.S. Government.

Destroy this report when it is no longer needed. Do not return it to the originator.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS None		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release: Distribution is unlimited		
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE					
4. PERFORMING ORGANIZATION REPORT NUMBER(S)			5. MONITORING ORGANIZATION REPORT NUMBER(S)		
6a. NAME OF PERFORMING ORGANIZATION U.S. Army Tank-Automotive Command		6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION U.S. Army Tank-Automotive Command		
6c. ADDRESS (City, State, and ZIP Code) Warren, MI 48397-5000			7b. ADDRESS (City, State, and ZIP Code) Warren, MI 48397-5000		
8a. NAME OF FUNDING / SPONSORING ORGANIZATION		8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c. ADDRESS (City, State, and ZIP Code)			10. SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.
					WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) Hydraulic Control Design and Modeling Techniques					
12. PERSONAL AUTHOR(S) Arthur L. Helinski					
13a. TYPE OF REPORT Final		13b. TIME COVERED FROM 9/88 TO 2/89		14. DATE OF REPORT (Year, Month, Day) February 1989	
15. PAGE COUNT 98					
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	Classical Control Theory, Hydraulic Modeling, Turret Motion Base Simulator (TMBS)		
19. ABSTRACT (Continue on reverse if necessary and identify by block number) This report details the analysis of a hydraulic actuator system. The analysis consists of designing control compensation by means of deriving a frequency response from a non-linear mathematical model and using classical control theory. This hydraulic actuator system will be used on the Turret Motion Base Simulator (TMBS) which is a unique six degree of freedom simulator being developed.					
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL Arthur L. Helinski			22b. TELEPHONE (Include Area Code) (313) 574-6676		22c. OFFICE SYMBOL AMSTA-RYA

PREFACE

This report covers the analytical modeling of hydraulic actuator systems. It also details control design methodologies. This report may be difficult to read for the casual reader. It is written assuming the reader has some fundamental background in control theory.

I would like to thank Contraves Goerz Corporation for their notes describing the hydraulic system which will be used on the Turret Motion Base Simulator (TMBS). Contraves Goerz Corporation is the prime contractor for the TMBS.

TABLE OF CONTENTS

Section	Page
1.0. INTRODUCTION	9
2.0. OBJECTIVE	9
3.0. CONCLUSIONS.	10
4.0. RECOMMENDATIONS	10
5.0. DISCUSSION	11
5.1. <u>Hydraulic Model</u>	11
5.2. <u>Method For Determining A Frequency Response</u>	14
5.3. <u>Control Design Analysis</u>	19
5.3.1 <u>Pressure Loop Response</u>	19
5.3.2 <u>Rate Loop Response</u>	27
5.3.3 <u>Position Loop Response</u>	27
5.4 RESULTS	27
LIST OF REFERENCES	39
APPENDIX A	
List of Hydraulic Actuator Computer Model	A-1
APPENDIX B	
List of FORTRAN Program for Frequency Response Analysis	B-1
APPENDIX C	
Various States of a Step Response	C-1
APPENDIX D	
Varied Step Response Results	D-1
APPENDIX E	
Varied Mass Results	E-1
DISTRIBUTION LIST	Dist-1

LIST OF ILLUSTRATIONS

Figure	Title	Page
5-1.	Hydraulic Actuator System	12
5-2.	Method Used For Determining A Frequency Response	.17
5-3	Input Signal Used For Frequency Response	.18
5-4	Actuator System Block Diagram	20
5-5.	Control Design Procedure	.21
5-6.	Pressure Loop Block Diagram	.22
5-7.	Plant Frequency Response of Pressure Loop	23
5-8.	Compensation Frequency Response of Pressure Loop	.24
5-9.	Open Loop Frequency Response of Pressure Loop	25
5-10	Closed Loop Response of Pressure Loop	26
5-11	Rate Loop Block Diagram	28
5-12.	Plant Frequency Response of Rate Loop	.29
5-13.	Compensation Frequency Response of Rate Loop	.30
5-14.	Open Loop Frequency Response of Rate Loop	31
5-15.	Closed Loop Frequency Response of Rate Loop	.32
5-16.	Position Loop Block Diagram	33
5-17.	Plant Frequency Response of Position Loop	.34
5-18.	Open Loop Frequency Response of Position Loop	.35
5-19.	Closed Loop Response of Position Loop	.36

LIST OF TABLES

Table	Title	Page
5-1.	Hydraulic Equations	.13
5-2.	System Parameters Used in Modeling	15
5-3.	Results of Controller Design	.37

1.0 INTRODUCTION

This report, prepared by the Systems Simulation and Technology Division, of the U.S. Army Tank-Automotive Command (TACOM) describes an analytical study of hydraulic systems used for laboratory testing. Motion base simulators are used by the Army to test vehicles or vehicle subsystems for various forms of performance and structural integrity. This form of testing is performed by hydraulic systems which produce forces applied to the vehicle simulating operation over terrain profiles.

A considerable effort is being made to develop analytical models of laboratory simulators so that a complete assessment can be made before testing is conducted. Analytical studies may point to various problems with the system and control design considerations may be made early in the development stage. The results of selected command signals representing terrain profiles can be simulated with the analytical model before being applied to the actual system. As the level of complexity in laboratory testing is increased the more important becomes analytical studies.

The Turret Motion Base Simulator (TMBS) will be a very complex system used for laboratory simulation testing. The TMBS is a computer controlled 6 degree of freedom motion base simulator which is capable of driving an M1 turret. An effort is being made to model the entire TMBS system. The starting grounds is to develop a model of a hydraulic actuator system and a means of control design and evaluation of performance. Although the model used in this report describes the hydraulics, it only contains a single actuator while the TMBS consist of a six actuator system. Therefore, the results in this study may not describe the performance of the TMBS. At this initial stage of the TMBS development, the material presented in this report should only be considered a study of a general hydraulic system. The analytical study that more directly relates to the TMBS will be addressed at a later time when the entire system is modeled. A portion of a model describing the dynamics and kinematics for the TMBS has already been established. The results of this study will be incorporated in the model in the near future so that a model will be constructed that contains the six actuator systems coupled with the platform dynamics.

This is a first attempt in modeling hydraulic systems in detail. An earlier attempt was made in creating an empirical model based strickly from test data which is described in reference 1. This technique consisted of a curve fit model of a hydraulic system derived from a measured closed loop frequency response of the system. Although this technique had resulted in prediction of actuator motion to some degree, it was only good for a system with no modifications or complexity.

2.0 OBJECTIVE

This report contains the initial stage of an analytical study of the TMBS. The primary objective is to develop a hydraulic actuator model along with a control design which produces reasonable performance. It is expected that the actuator system will have a performance bandwidth of 10 Hz with an adequate stable transient response. The initial step in achieving this objective is to find a means of designing a control system by using the mathematical model. A methodology was developed which consisted of writing a FORTRAN program which is used to design control system compensation by supplementing it with the computer mathematical model.

3.0 CONCLUSIONS

The methodology and software developed to design control compensation seems to work very well. There is definitely a correlation between the stability margins obtained in the frequency domain and the transient response in the time domain. The model shows that the actuator system can perform a 10 Hz bandwidth. The step response simulated has very good results in terms of overshoot and settling time. It is concluded that a control design can be obtained analytically without making a linear reduction of the system.

The control design obtained through this analysis may or may not be adequate for the actual system. This will depend solely on how accurately the hydraulic model compares with the real system. The most serious difficulty encountered while modeling a hydraulic system is that several of the parameters required to describe the system are unknown. What is more important at this time is that the methodology is now available for control design if an adequate model can be obtained. The ground work established in this study can lead to the analysis of other hydraulic systems used for laboratory testing or for any other application. Deriving the frequency response from a non-linear mathematical model can assist in control design or improving existing control designs.

The study presented in this report is a starting point for more extensive analytical studies of hydraulic systems. A model of the TMBS has been established containing the dynamics and kinematics which consist of the configuration of the 6 actuator orientation and platform. However, this model required a working model of a hydraulic actuator with control. Now that a working hydraulic actuator model has been established, the next step is to incorporate the hydraulic model into the TMBS model. The study will lead to investigating the control of the platform by the six actuators. Various methodologies have been developed by Contraves Goerz Corporation which uncouples the effects of the six actuators driving the platform. These methodologies consist of a form of adaptive control which involve considerations of the inertias, mass and actuator/platform orientations in 3-dimensional space. This form of adaptive control, supplements with the single actuator control to complete a control design for the entire system. Future goals are to investigate these methods thoroughly by using analytical models which may lead to problem solving or possible improvements with the TMBS system.

4.0 RECOMMENDATIONS

It is important to validate the hydraulic model developed in this study. At this time it can be stated that the model exhibits behavior very similar to other known hydraulic systems. The pressures, flow rates and actuator/valve characteristics of the model are reasonable for this type of system. Nevertheless, there is a degree of uncertainty to the parameters used for the model. Contraves Goerz Corporation is scheduled to perform a single actuator test later this year. At this time the model can be validated and adjusted if necessary. The analysis and control design were conducted using a small mass for a load. This was done so that comparisons can be made with the single actuator testing using little to no load as a starting point.

The search for an optimal control design was not real extensive at this time. Once the model shows some validity then an extensive optimal control design can be studied. This is only when the unknown parameters can be determined and the proper configuration is known. (There are still uncertainties on the inner loop configuration). Once this is accomplished a further study can be made on optimizing the controller for a loaded case consisting of a larger mass.

5.0 DISCUSSION

The starting point for modeling hydraulics is to obtain a set of equations which best describes the system. These equations contain many parameters which reflect the characteristics of the system. These parameters are not always known and must be estimated or guessed at some way until measurements can be made on the system. It may not be possible to measure some of these parameters directly. Some parameters may require determination by a combination of measurements and other known mathematical relationships. Once a set of equations and parameters are obtained, a model can be developed. There are many forms of computer software available to simulate the model. The one used for this analysis is the Advanced Continuous Simulation Language (ACSL) which basically simulates equations in the time domain.

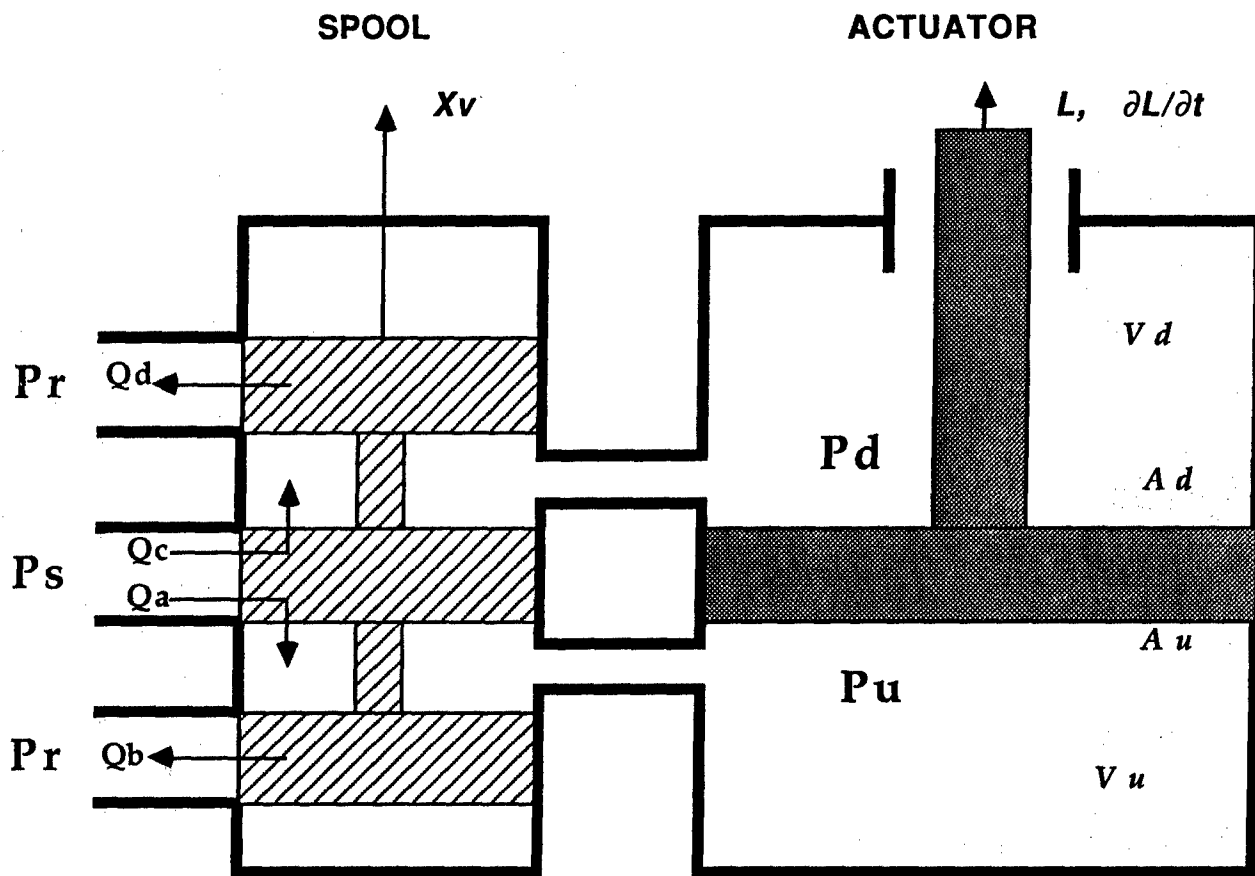
To achieve the results of a system with feedback control, a control design must be known or derived from the model. A control design is part of the system that is modeled and comprises the loop configuration and control compensations. For our case, the control design received was from early development and is not considered acceptable. A good part of this study is to find a means of deriving a control design from an analytical model.

To derive a control design, the proper method using classical control theory implies that various frequency responses must be determined where an open loop response can be evaluated and control compensation can be designed. Most fundamental courses in control theory stress the importance of linearizing the system equations (Plant equations). This is done so that the frequency response can be easily obtained because the equations reduce to transfer functions in simple Laplace Transform format. With a linearized set of equations it is also convenient to determine the eigenvalues (root locus analysis). It is usually questionable when linearizing a system if the end results are still truly representative of the system. An effort was made in this study to develop a software technique which will determine a frequency response from a non-linear mathematical model directly without linearizing. The primary philosophy was that no questionable linearizing techniques have to be applied and the procedure of algebraically reducing the system equations to a transfer function was not required. This method takes advantage of number crunching computer power and is similar to a frequency response technique which is measured in a laboratory utilizing a signal analyzer. The hydraulic model is then evaluated with this technique and control compensation is designed for each feedback loop. The rest of this report describes the details of the hydraulic model and the results of the analysis used for this technique.

5.1 HYDRAULIC MODEL

Shown in Figure 5-1 is a diagram illustrating the internal workings of a hydraulic spool/actuator system. The spool basically controls the fluid flow to the actuator and is considered the input state to these equations. As the spool is displaced up, the supply pressure creates flow to the actuator bottom chamber which creates a pressure (P_u) which in turn produces an upward force on the actuator. The negative actuator motion is produced the same way with a spool displacement down. The equations describing the spool/actuator system are shown in Table 5-1. The model is a simplified case where no overlaps are considered for the spool or the actuator.

HYDRAULIC ACTUATOR SYSTEM



INPUT: X_v Spool Position

ACTUATOR: L Actuator Position
 $\partial L / \partial t$ Actuator Velocity
 A_u, A_d Effective Actuator Area
 V_u, V_d Entrained Volume

PRESSURES: P_s Supply Pressure
 P_r Return Pressure
 P_u Actuator Pressure Up
 P_d Actuator Pressure Down

FLOWS: Q_a Flow In from Supply (Actuator Up)
 Q_b Flow Out to Return (Actuator Up)
 Q_c Flow In from Supply (Actuator Down)
 Q_d Flow Out to Return (Actuator Down)

FIGURE 5-1

HYDRAULIC EQUATIONS

Assume Ideal Valve with Zero Overlap

$$\text{Flow Area} = \pi D X_v = W X_v$$

D Spool Diameter

$$\text{Flow } Q = C X_v \sqrt{\Delta P}$$

Xv Spool displacement from null

Q Flow

Where C is hydraulic constant

ΔP Pressure across valve/actuator

$$C = C_d W \sqrt{SQRO}$$

W Valve Spool Circumference

SQRO Fluid Constant

Cd Hydraulic Constant

Valve Displacement

Valve Up Xvu

Valve Down Xvd

$$\begin{aligned} X_{vu} &= X_v \text{ if } X_v > 0 \\ &= 0 \text{ if } X_v < 0 \end{aligned}$$

$$\begin{aligned} X_{vd} &= |X_v| \text{ if } X_v < 0 \\ &= 0 \text{ if } X_v > 0 \end{aligned}$$

Flow Equations for Valve/Spool/Actuator System

$$Q_a = C X_{vu} \sqrt{|P_s - P_u|} \text{ Sign } (P_s - P_u)$$

Pu Actuator Pressure Up

$$Q_b = C X_{vd} \sqrt{|P_u - P_r|} \text{ Sign } (P_u - P_r)$$

Pd Actuator Pressure Down

$$Q_c = C X_{vu} \sqrt{|P_s - P_d|} \text{ Sign } (P_s - P_d)$$

Ps Supply Pressure

$$Q_d = C X_{vd} \sqrt{|P_d - P_r|} \text{ Sign } (P_d - P_r)$$

Pr Return Pressure

Qa, Qc Flow In from Supply

Qb, Qd Flow Out to Return

(Actuator Up Down Respectively)

Actuator Pressure

β Bulk Modulus of Fluid

$$\partial(P_u)/\partial t = \beta / V_u (Q_a - Q_b - A_u \partial L/\partial t)$$

Au, Ad Actuator Piston Area

$$\partial(P_d)/\partial t = \beta / V_d (Q_c - Q_d - A_d \partial L/\partial t)$$

(for up and down respectively)

where

$$V_u = V_{u0} + A_u L$$

Vu0, Vd0 Initial Entrained

Vu, Vd Entrained Volume

(for up and down respectively)

$$V_d = V_{d0} - A_d L$$

Actuator Force

L Actuator Displacement

$\partial L/\partial t$ Actuator Rate

$$F = P_u A_u - P_d A_d$$

F Actuator Force

TABLE 5-1

The direction of the valve displacement is detected and separated by the variables X_{vu} & X_{vd} for up and down motion respectively. This was done so that the up and down cases can be handled separately since the two motions have different characteristics in terms of flows, effective areas and entrained volumes. Note the u & d subscripts for the remaining equations. There are four relative pressure drops between the supply & return (P_s & P_r) and the actuator pressures (P_u & P_d). The four relative pressure drops produce four flows which are described in the equations as Q_a , Q_b , Q_c and Q_d . (See Figure 5-1) The flow is assumed to be proportional to the spool displacement (X_v) times the square root of the corresponding pressure drop by a constant C . The equations take the absolute value of the pressure drops since complex numbers do not apply (square root of a negative value). The sign of the pressure drop is considered by the 'SIGN' function instead. Thus a negative relative pressure drop creates a negative flow. The spring characteristic of fluid is affected by the entrained volumes (V_u , V_d , V_{u0} and V_{d0}) and the bulk modulus (β) which is assumed to be constant (no temperature variation). The derivative of pressure is dependent on many quantities including the actuator motion. In a sense this can be considered a natural internal feedback in the system. Reducing these equations to a single transfer function is very difficult and is not required for the methodology presented in this analysis. The valve transfer function is shown in Table 5-2 which describes the valve dynamics by a second order. The S represents Laplace Transform notation throughout this report. This equation describes the response for a valve where the input would be current and the output is the displacement of the spool. Also shown in Table 5-2 are the various parameters used for the model. The most questionable and difficult to measure value is the constant C which is the hydraulic flow constant mentioned earlier. The various equations were obtained partially from a combination of reference 2, 3, and 4. The parameters used for this analysis are from reference 4. The key element of control is the feedback loops which can now be incorporated with these equations describing the system to complete the model. The model was computer simulated using Advance Continuous Simulation Language (ACSL) which is listed in Appendix A.

It should be mentioned that this hydraulic model is a simplified representation. There are many nonlinear hydraulic fluid flow properties which are assumed to be negligible in this model. In addition this model describes a two stage servo system even though it is believed the TMBS will have a three stage servo system. Since no details have been received on the pilot valve characteristics it will be assumed that the pilot valve is an ideal case. It is also proposed that the TMBS will have three valves driving each actuator which are all controlled in the same manner. This was primarily done for safety and backup considerations. It is also assumed that this will not change the actuator performance predicted by this model. Some of the assumptions made to simplify this model may not be valid. These modifications may be incorporated in the model at a later time.

5.2 METHOD FOR DETERMINING A FREQUENCY RESPONSE

The basis of designing a control system using classical control theory is to obtain a frequency response of the system (Plant Response). Once a frequency response is established, control compensation can be designed to desired stability margins and gain crossover (bandwidth) which will reflect the system's performance in the time domain. As mentioned earlier, an effort was made not to complicate the analysis with questionable linearizing techniques which simplify but may burden the accuracy of the analysis.

System Parameters Used In Modeling

Valve Transfer Function

$$\frac{\text{Spool Position}}{\text{Input Current}} = \frac{1.5 \text{ E-4}}{2.533 \text{ E-6 S}^2 + .002228 \text{ S} + 1} \frac{\text{IN}}{\text{mA}}$$

$$\text{Circumference } W = 3.78 \text{ IN}$$

Actuator

$$\begin{array}{l} \text{Effective Area} = 38.5 \text{ IN}^2 \\ \text{Initial Entrained Volume} \\ \text{Up } 1697 \text{ IN}^3 \quad \text{Down } 1336 \text{ IN}^3 \end{array}$$

Hydraulic Fluid Constants

$$\text{SQRO} = 1. \quad \text{Bulk Modulus } (\beta) = 100,000. \quad \text{Cd} = 100.$$

Pressures

$$\text{Supply} = 3000. \text{ PSI} \quad \text{Return} = 100. \text{ PSI}$$

Mass

$$M = \frac{1}{6} (\text{Ring Mass}) = 32.37 \text{ Slugs}$$

Note; All frictions and hydraulic leakages are assumed to be negligible

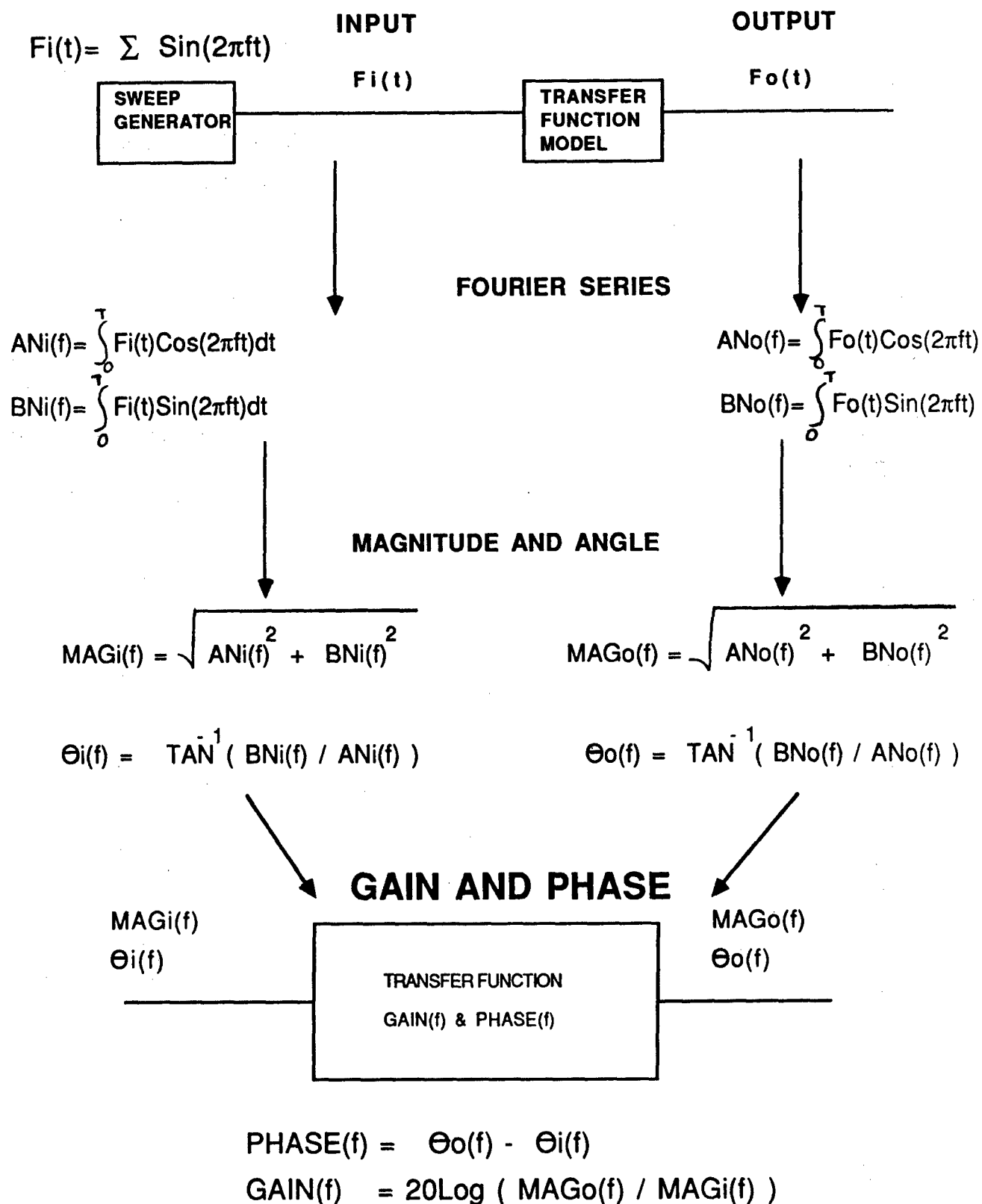
TABLE 5-2

The method of determining a frequency response from a non-linear mathematical model is shown in Figure 5-2. The method is based on taking the Fourier series of both the input and output time histories. This method can be interpreted as having the analysis force the model to behave much like a real system when measured by a Fourier analyzer in the laboratory. However the input signal for this case consists of a sum of sine waves at selected frequencies which span a desired range. The Fourier coefficients are then used to determine the appropriate gain and phase of the response. The end result of the analysis is only considered at the frequencies generated by the input signal. (A round off technique is applied to round off to the nearest milli-Hz) The technique used in the analysis was derived from experimentation with a known linear model describing a transfer function. This was done so that the correct results can be determined and compared with the results of this analysis. Various forms of input signals were tested using this technique. An impulse function was used in several cases. Although the results were close for gain there was an offset in the frequency domain which was affected by how the impulse function was approximated. The phase could not be determined properly using this technique. Random noise was also tested much like a signal analyzer applies the input. This form of input did produce accurate results but required a considerable amount of simulation time and data points to average to a smooth spectrum of frequencies (White Noise). The sum of sine waves was selected because it was a much more controlled input signal and gave accurate results. There were only slight inaccuracies at the low frequencies (first few points) which should not affect control analysis. One drawback with this technique is the limitation of data points generated in the frequency domain. This technique was used previously to validate a non-linear math model describing the M1 gun/turret tracking system. The results gave responses much closer to the measured test data than the linearized model at various operating points.

Figure 5-3 shows the input signal used for this analysis which consist of a sum of sine waves at the frequencies specified. For this particular example the sine waves have a magnitude (or amplitude) of unity. When the Fourier series is evaluated at the frequencies generated the results are a magnitude of unity and angle of 90° as would be expected for this case. The input in the time domain aproaches a function similar to an impulse train as more frequencies are added. One consideration when using this technique is determining what magnitude to use on the sine wave generation. This consideration is similar to determining what operating point to choose when linearizing a model. With some insight and experimentation one can select reasonable magnitudes to use for a particular part of the system. Avoiding saturation levels is the primary concern, however the results are magnitude dependent as would be expected for a non-linear system.

The frequency response analysis will be conducted at selected parts of the system model to determine various plant responses and closed loop responses for each loop configuration. The sine wave generated input will be applied to the model at the input of desired functions while the output is recorded for future use. Once the data file is created from the model, a frequency response analysis can be conducted using a FORTRAN program called Bode. This program is listed in Appendix B and has many applications. When the frequency response represents a plant response the program has the feature of designing compensation to the results. Since compensation is considered a cascade function with the plant response, the total open loop can be determined by simply adding the gain and phase of the functions. The program has the feature of doing this and observing stability margins. The program is menu driven and only takes seconds to vary the compensation to obtain a new open loop response. The following section will describe how this technique was applied to the hydraulic actuator model.

METHOD FOR DETERMINING A FREQUENCY RESPONSE



This method is evaluated at the frequencies used in the Sweep Generator.

f discrete frequency
t discrete time

FIGURE 5-2

INPUT SIGNAL USED FOR FREQUENCY RESPONSE

INPUT:

$$\text{INPUT}(t) = \sum A \sin (2 \pi F_n t)$$

Where : A desired magnitude
 F_n desired frequencies
 t simulated time

For our case F_n consist of 42 selected frequencies as follows:

$F_n = .2, .4, .6, .8,$
1., 1.4, 1.6, 2.0, 2.4, 3.0, 3.4, 4.0, 4.4, 5.0, 5.4, 6.0, 6.4,
7.0, 7.4, 8.0, 8.4, 9.0, 10.0,
14.0, 20.0, 24.0, 30.0, 34.0, 40.0, 44.0, 50.0, 54.0, 60.0
64.0, 70.0, 75.0, 80.0, 85.0, 90.0, 95.0, 99.0 Hz

Which gives the following time history:

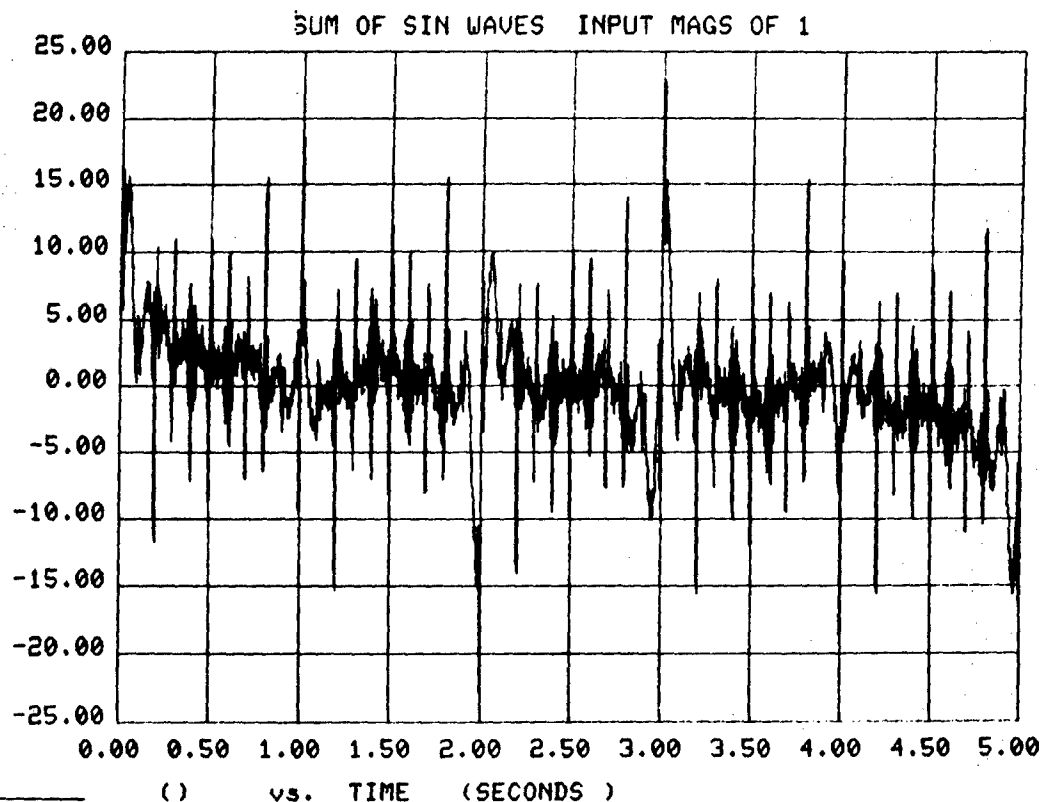
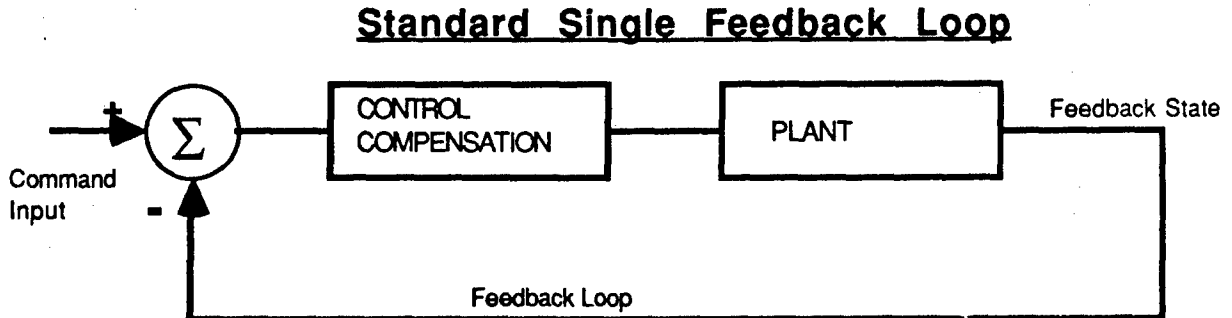


FIGURE 5-3

5.3 CONTROL DESIGN ANALYSIS

Shown in Figure 5-4 is a block diagram of the actuator system. The system comprises three feedback loops for control. The states containing feedback are position, rate and pressure. Some notes received show the inner loop to be force rather than pressure. This should only result in a difference in gain in the inner loop. When designing control compensation with classical control theory most fundamental text books describe the following simplified system:



Where the compensation is the only portion of the system which is subject to change. The plant is considered that portion of the system which is left unchanged. The design consist of deriving compensation which produces the end result or performance. For this analysis, the multi-loop system will be reduced to three systems with a single feedback, such as the system shown above. This procedure will be conducted one step at a time until the entire system is established. Figure 5-5 shows the procedure in more detail for each step. The frequency response mentioned earlier will be used as the tool to evaluate a plant response, closed loop response and design compensation by observing the open loop response.

5.3.1 PRESSURE LOOP RESPONSE

Shown in Figure 5-6 is the inner loop block diagram which illustrates in more detail the inner loop configuration including the valve dynamics. The blocks after the compensation are considered the plant for this case. The plant response is shown in Figure 5-7 for gain and phase. The valley at .4 Hz may be a numerical problem and will not have any influence on stability. The peak at 10 Hz represents the hydraulic resonance due to compression of the fluid. Figure 5-8 shows the pressure compensation frequency response which was derived by experimentation by observing the open loop response which resulted in Figure 9 for the compensation described. The goal was to establish the highest gain cross frequency with still maintaining adequate stability margins (See Figure 5-9). The end result is the pressure loop response shown in Figure 5-10. The closed loop response indicates a bandwidth greater than 62 Hz and a very quick step response. This gives an indication to how quick the system will build up pressure to the actuator for a desired 1500 PSI command. The step response was obtained by running the computer model in closed loop form and compensation design intact with a step command as pressure command. This verifies the design which was derived in the frequency domain.

ACTUATOR SYSTEM BLOCK DIAGRAM

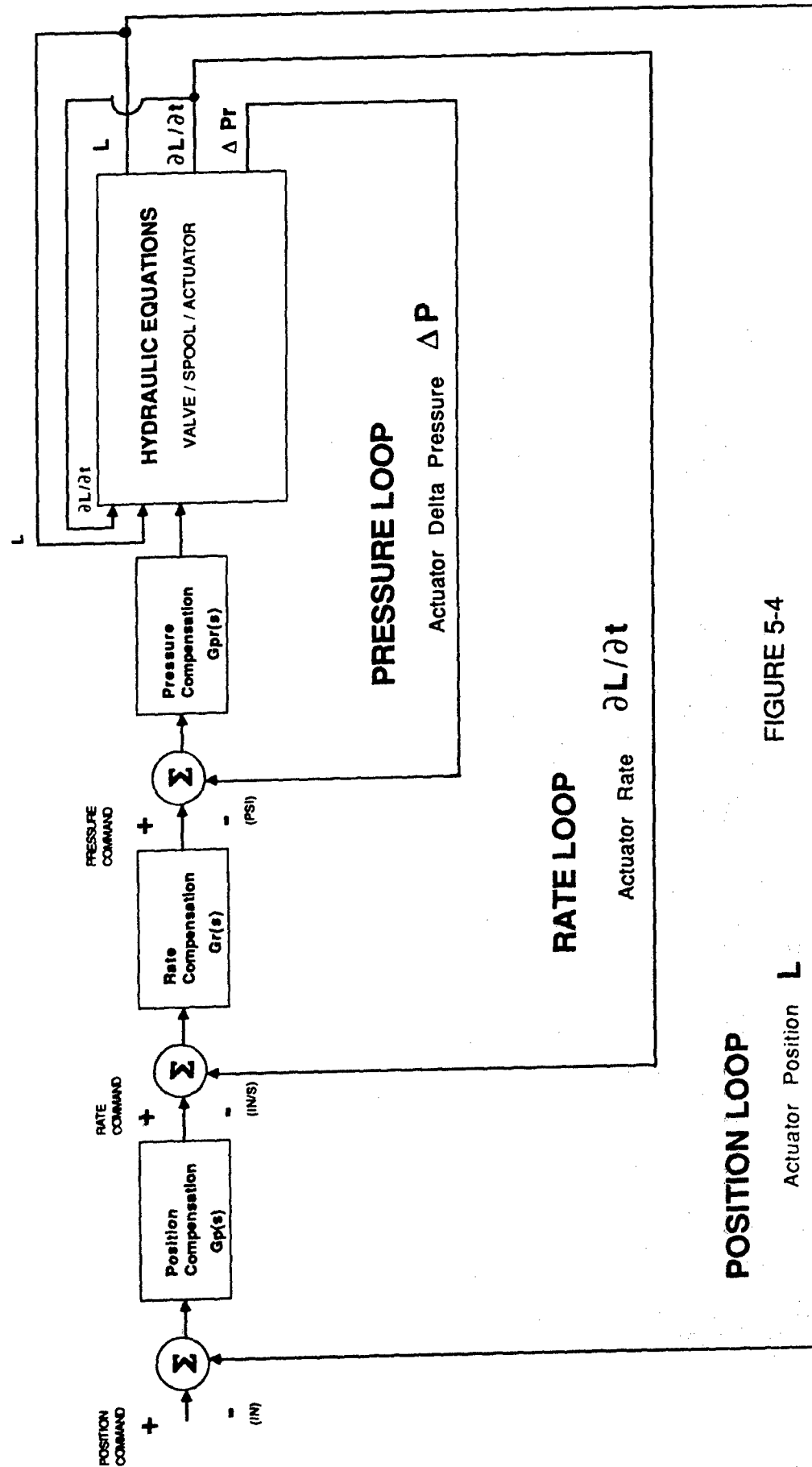


FIGURE 5-4

CONTROL DESIGN PROCEDURE

INNER LOOP

Obtain plant response of Inner Loop

Design Compensation with Open Loop Response

Close Loop and verify design with step response

SECONDARY LOOP

Obtain plant response of secondary loop with new compensated inner loop included

Design Compensation with Open Loop Response

Close Loop and verify design with step response

OUTER LOOP

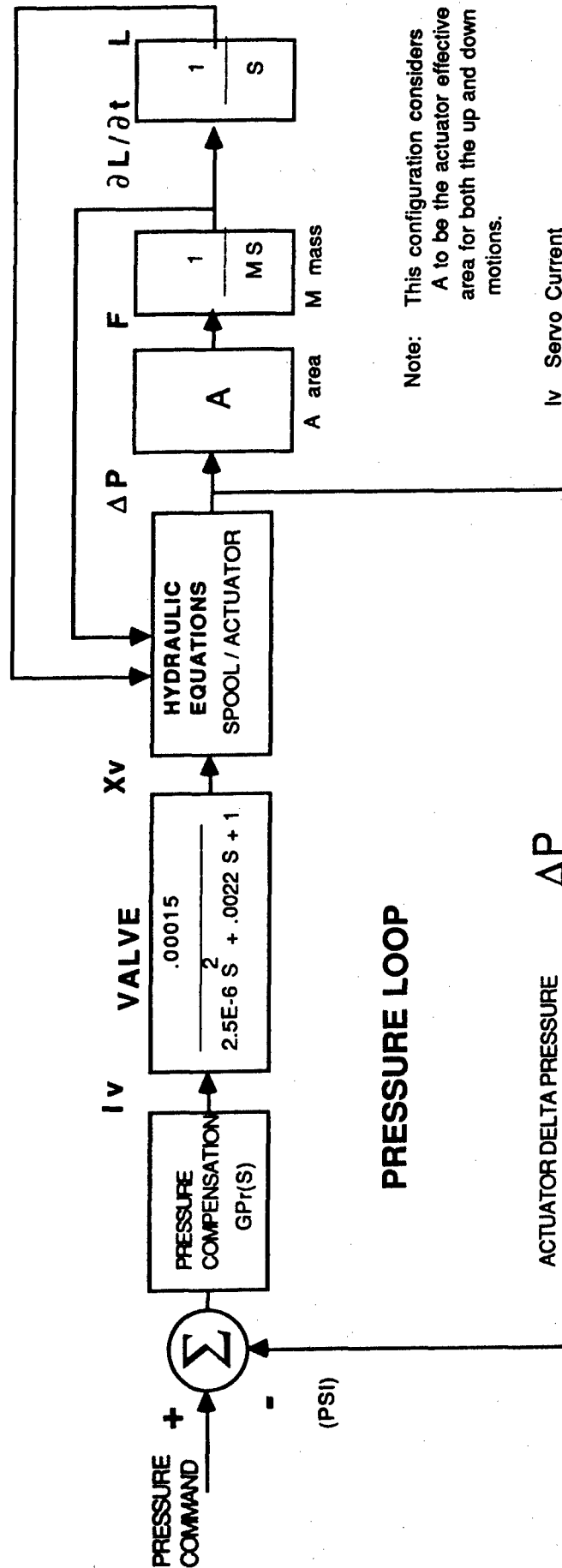
Obtain plant response of outer loop with new secondary and inner loops included

Design Compensation with Open Loop Response

Close Final Loop and verify design with step response

FIGURE 5-5

PRESSURE LOOP BLOCK DIAGRAM (INNER LOOP)



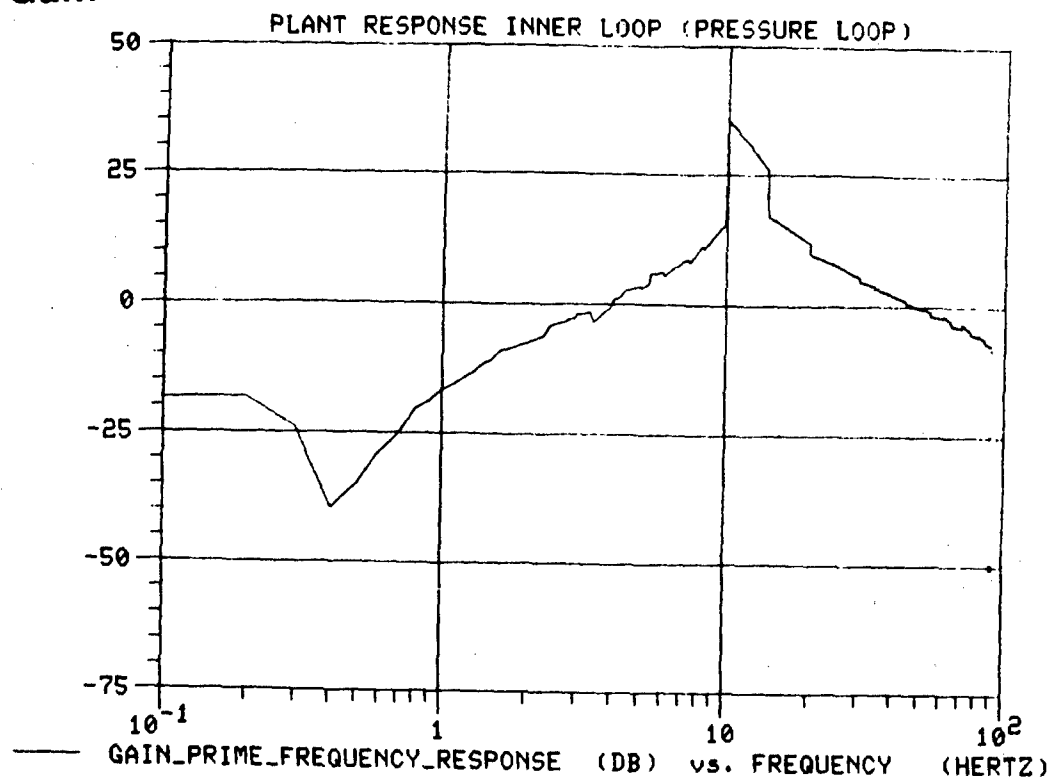
Note: This configuration considers
A to be the actuator effective
area for both the up and down
motions.

Iv Servo Current
Xv Spool Position
 ΔP Delta Actuator Pressure
F Actuator Force
 $\frac{\partial L}{\partial t}$ Actuator Rate
L Actuator Position

FIGURE 5-6

Plant Frequency Response of Pressure Loop

Gain



Phase

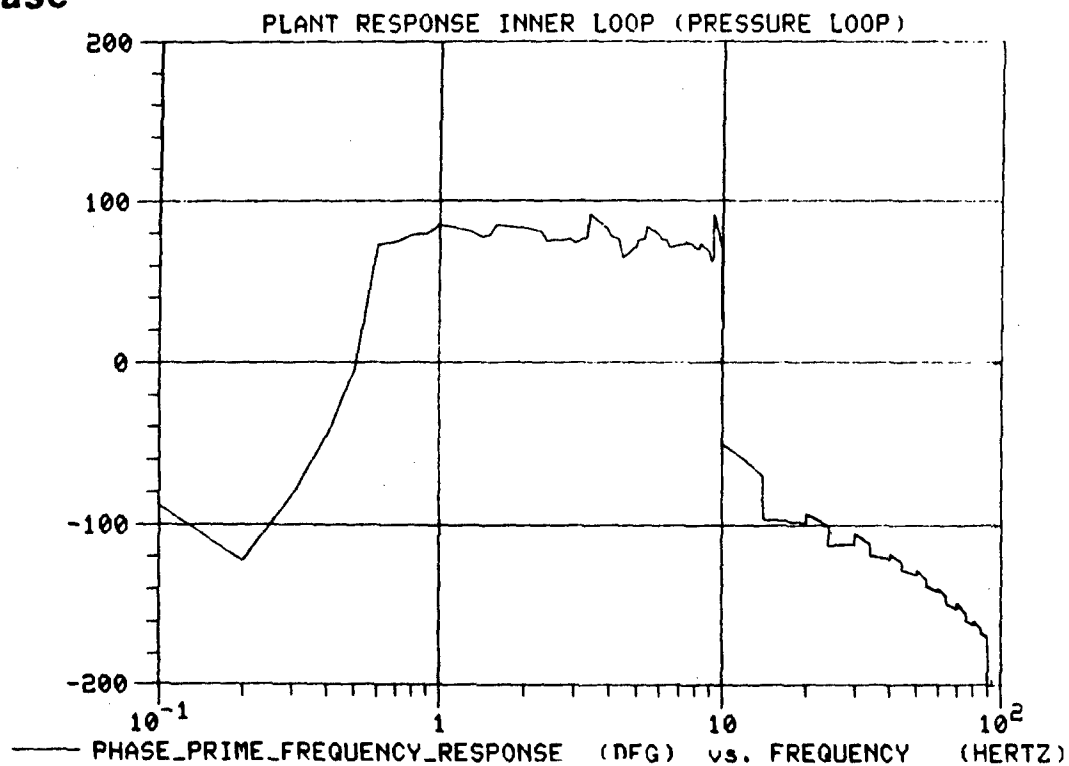
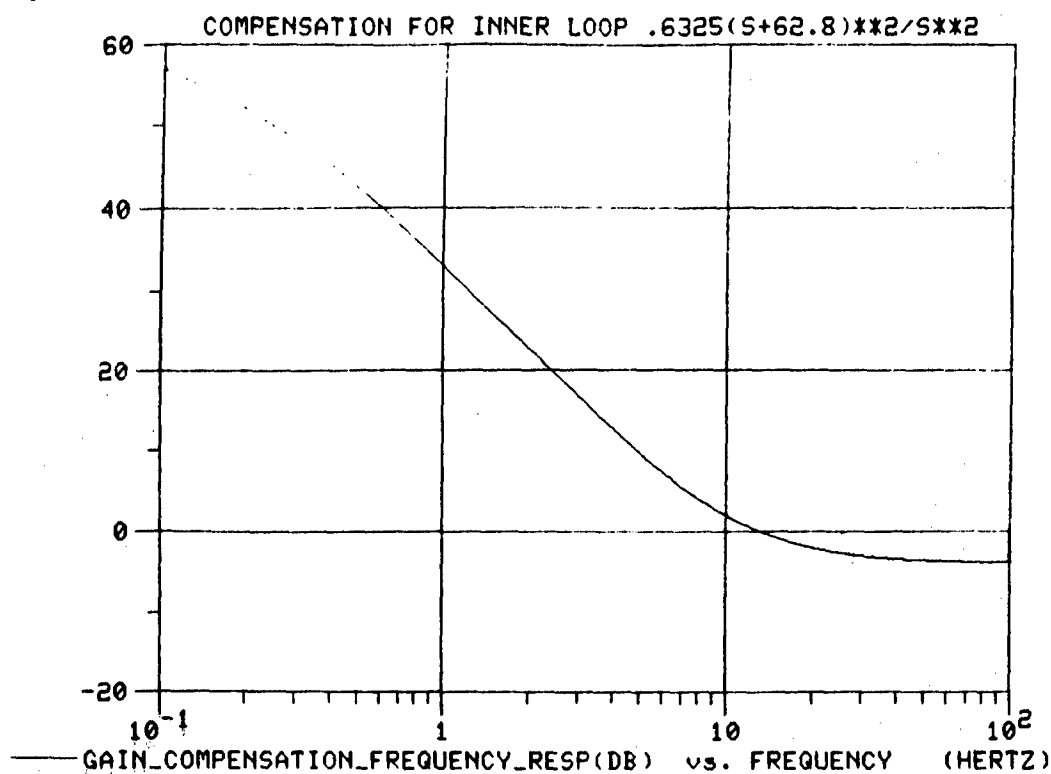


FIGURE 5-7

Compensation Frequency Response of Pressure Loop

Gain



Phase

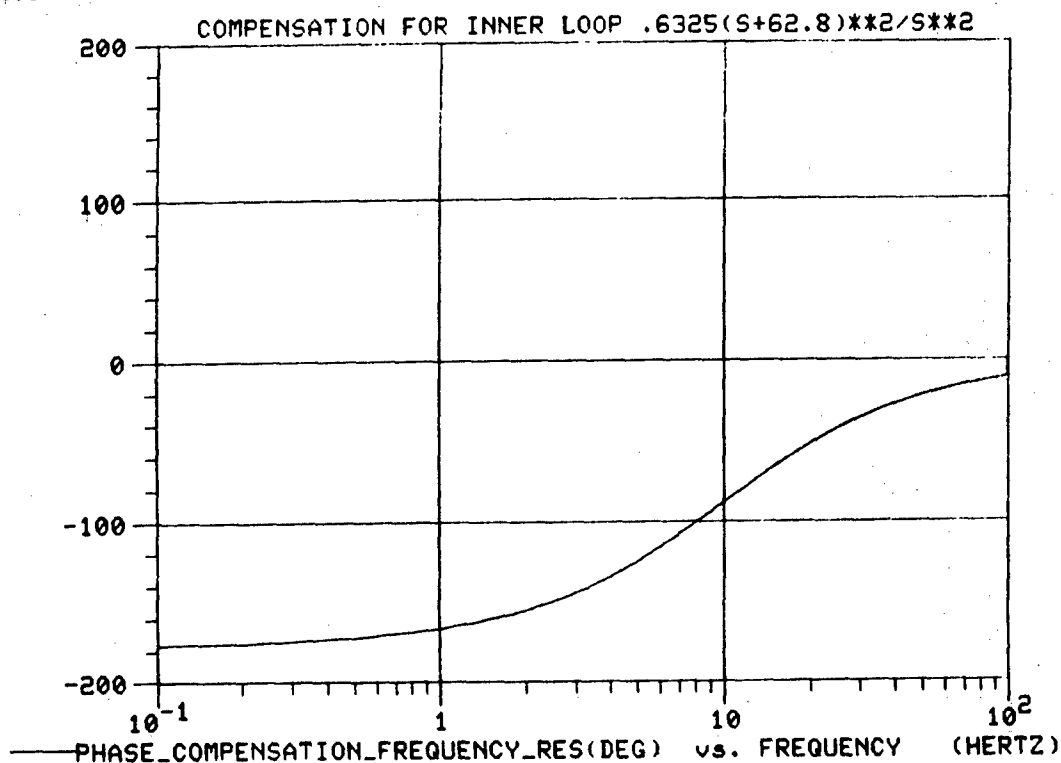


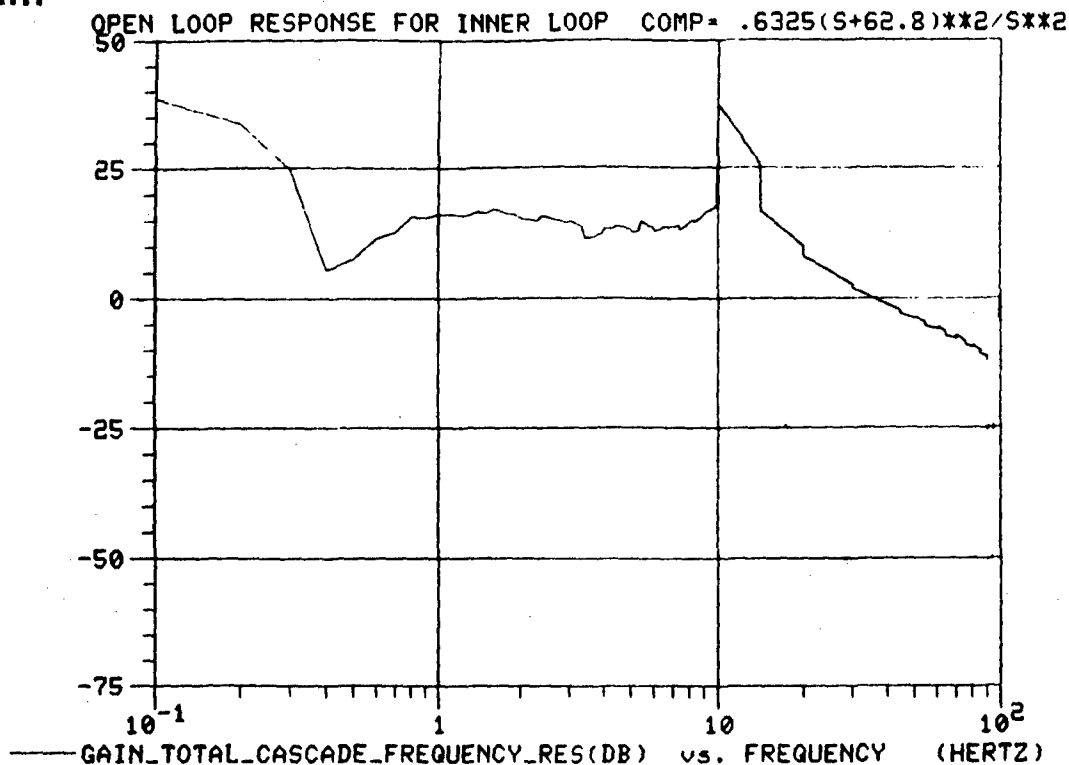
FIGURE 5-8

Open Loop Frequency Response of Pressure Loop

Gain Margin = 10.2 dB

Gain Cross 35.3 Hz

Gain



Phase Margin = 28.9°

Phase -180° Cross 85.1 Hz

Phase

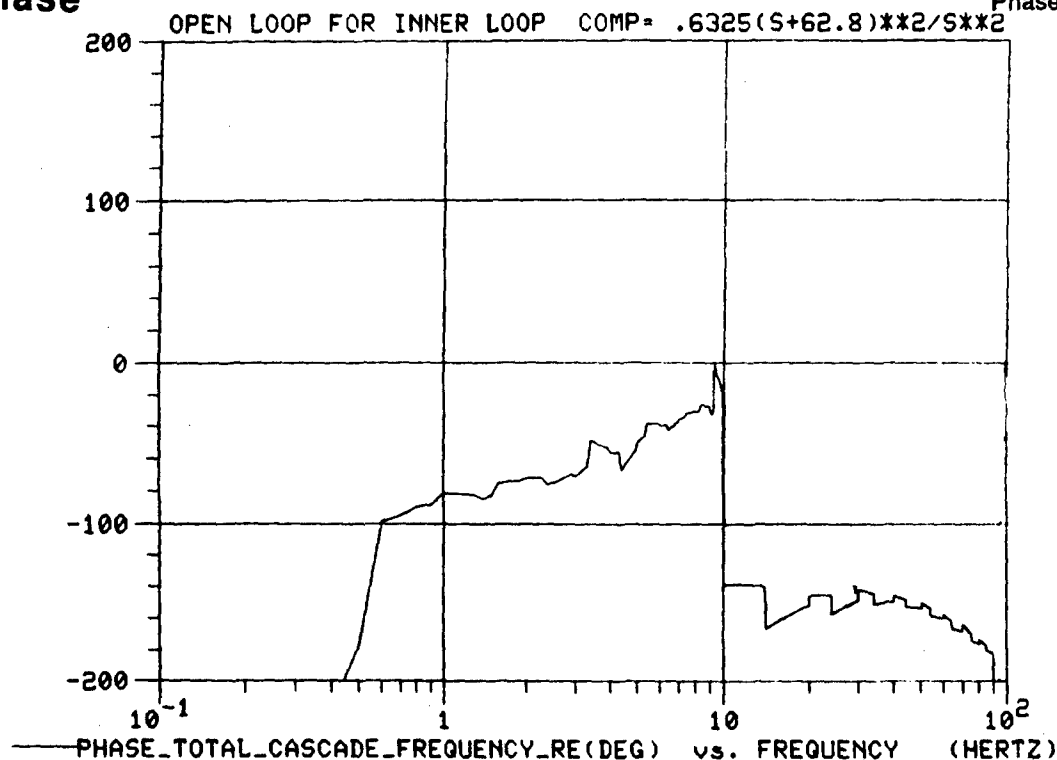
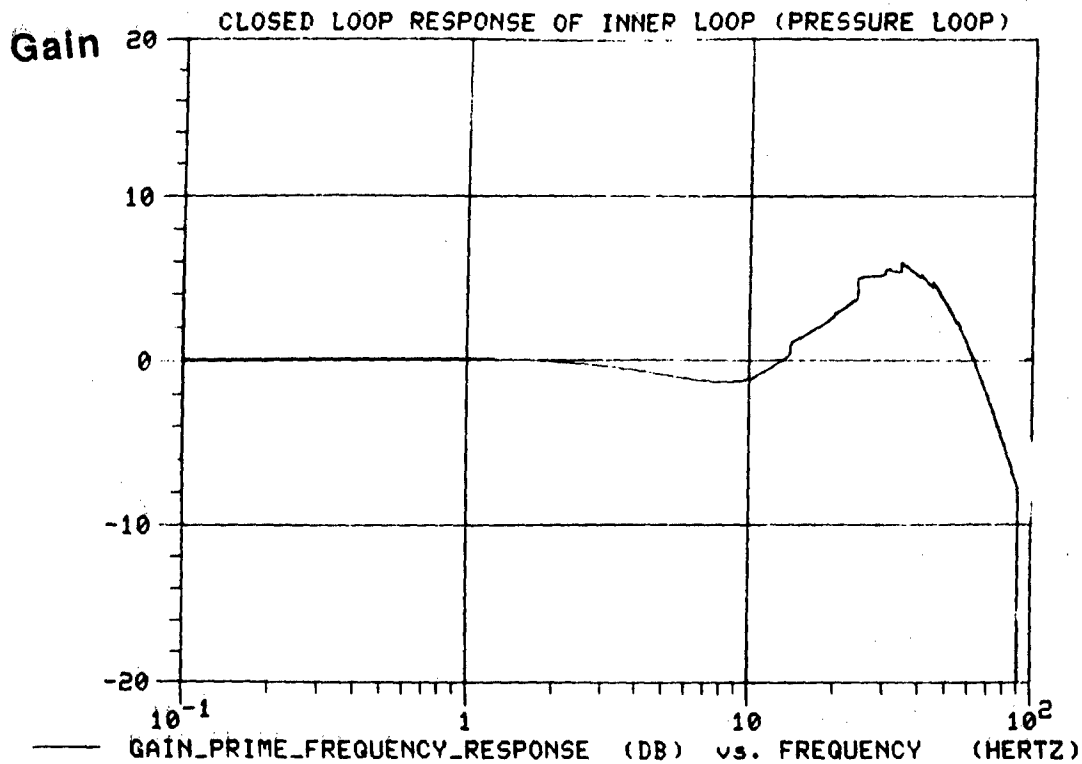


FIGURE 5-9

Closed Loop Response of Pressure Loop

Gain Cross 62. Hz



Step Response

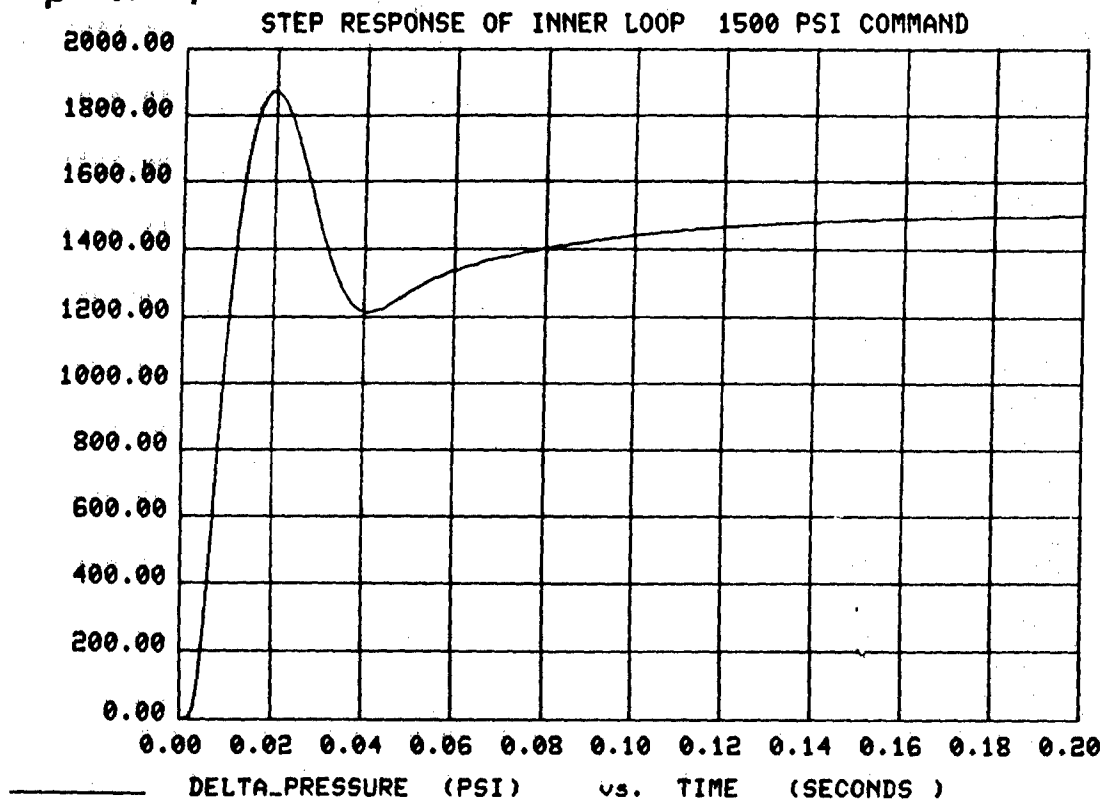


FIGURE 5-10

The compensation for the inner loop was determined to be a second order transfer function. A first order transfer function resulted in a steady state error. It was decided to increase the type number of the system (Number of pure integrators). The result is using two PI (Proportional + Integral) controls in cascade instead of a single one.

5.3.2 RATE LOOP RESPONSE

The same procedure is now used for the rate stage of the system model which is illustrated in Figure 5-11. The plant is now considered to be the entire new inner (pressure) loop which includes the compensation which was just created. Figure 5-12 shows the frequency plant response for the rate loop. The compensation was a single PI control shown in Figure 5-13. The resulting open loop response is shown in Figure 5-14. The results are shown in Figure 5-15 with the closed loop response and the step response. The gain cross is over 38 Hz for the closed loop response. The step response indicates some ringing but settles within .12 seconds. Perhaps the stability margins could have been improved for this stage. However, at this time the primary concern was to establish an adequate response for the overall system which is in terms of a position command response.

5.3.3 POSITION LOOP RESPONSE

With the new pressure and rate loops having compensation design, the final stage can be evaluated, completing the system. Figure 5-16 shows the position loop configuration. The plant now includes the new pressure and rate loops, which has a response shown in Figure 5-17. The compensation is not shown because it simply has a gain of 31.6 to establish good stability margins. The total open loop response is shown in Figure 5-18. The step response is well damped and has a settling time of about .2 seconds as shown in figure 5-18. The bandwidth is 10.1 Hz as shown in closed loop frequency response shown in Figure 5-19. There is always a compromise between stability margins and bandwidth performance. The bandwidth can be easily raised for this system if desired. A simple increase in compensation gain will result in a quicker response but produce more overshoot and ringing. The general consideration for this application is to have a quick response with minimal overshoot.

5.4 RESULTS

Shown in Table 5-3 are the results of the compensation design. These results are considered adequate at this time. It is illustrated that there is a correlation between the stability margins and step response characteristics. Control compensation can be designed with this technique and may prove to be of value if the model is accurate or is modified for improving its accuracy. Some of the frequency responses derived here for control design are not always possible to measure on the actual system. This stresses the importance of establishing a design analytically before doing the fine tuning or "tweaking" of the controls in the laboratory. Analytical design gives insight into which direction to go with various gains of the controller.

RATE LOOP BLOCK DIAGRAM

(SECONDARY LOOP)

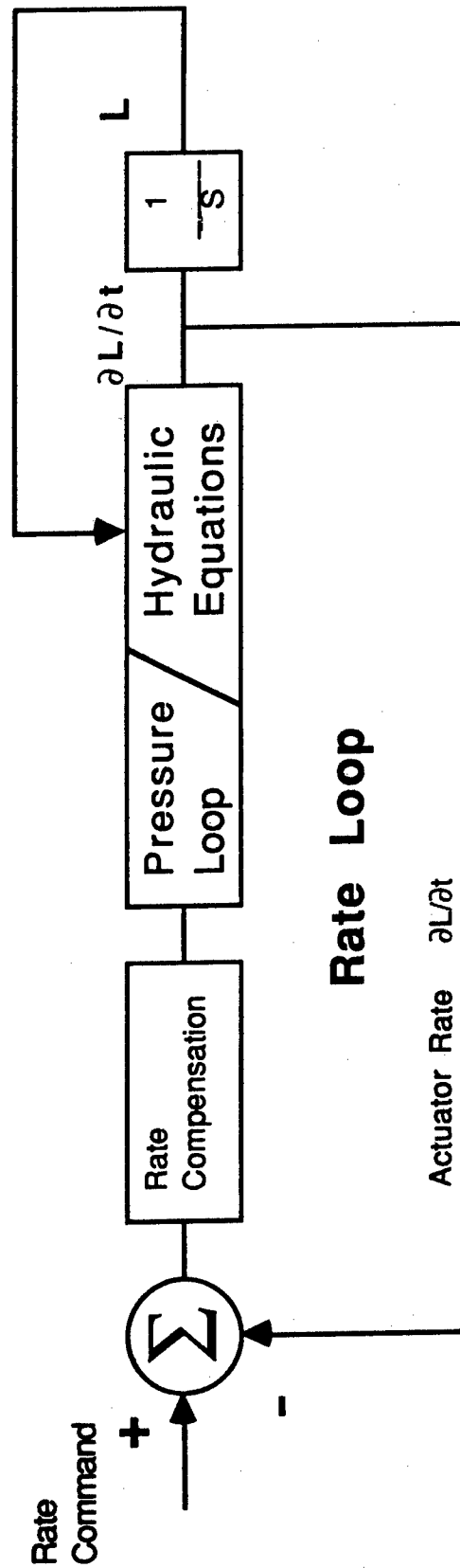
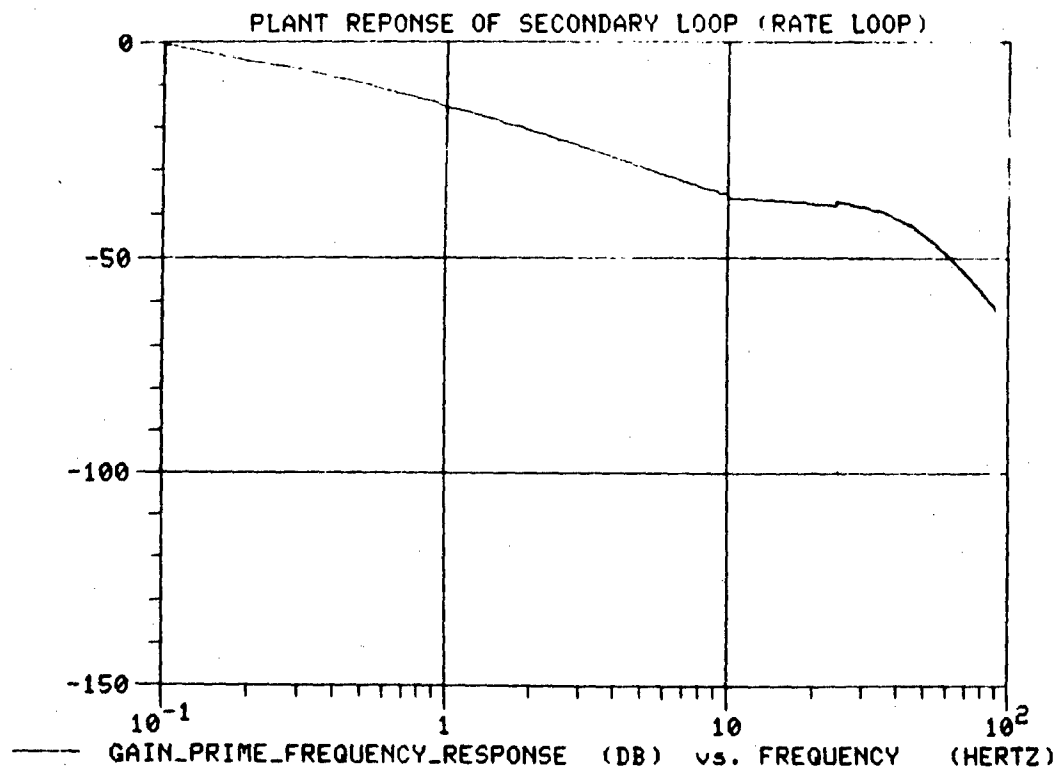


FIGURE 5-11

Plant Frequency Response of Rate Loop

Gain



Phase

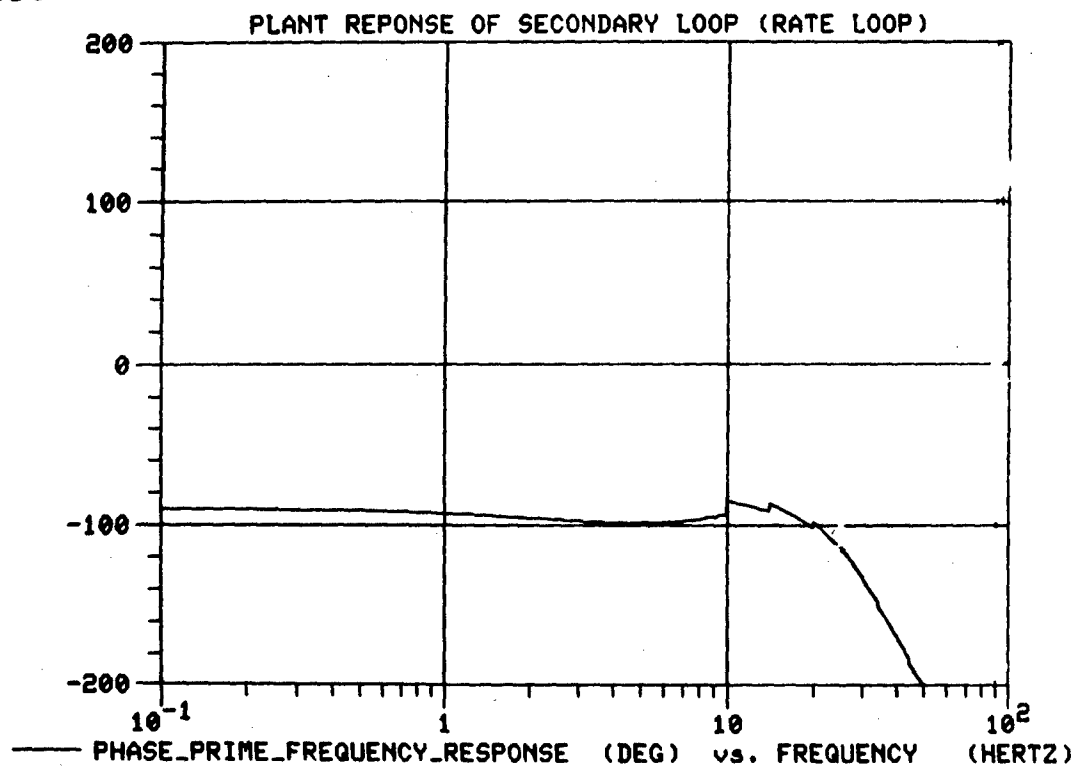
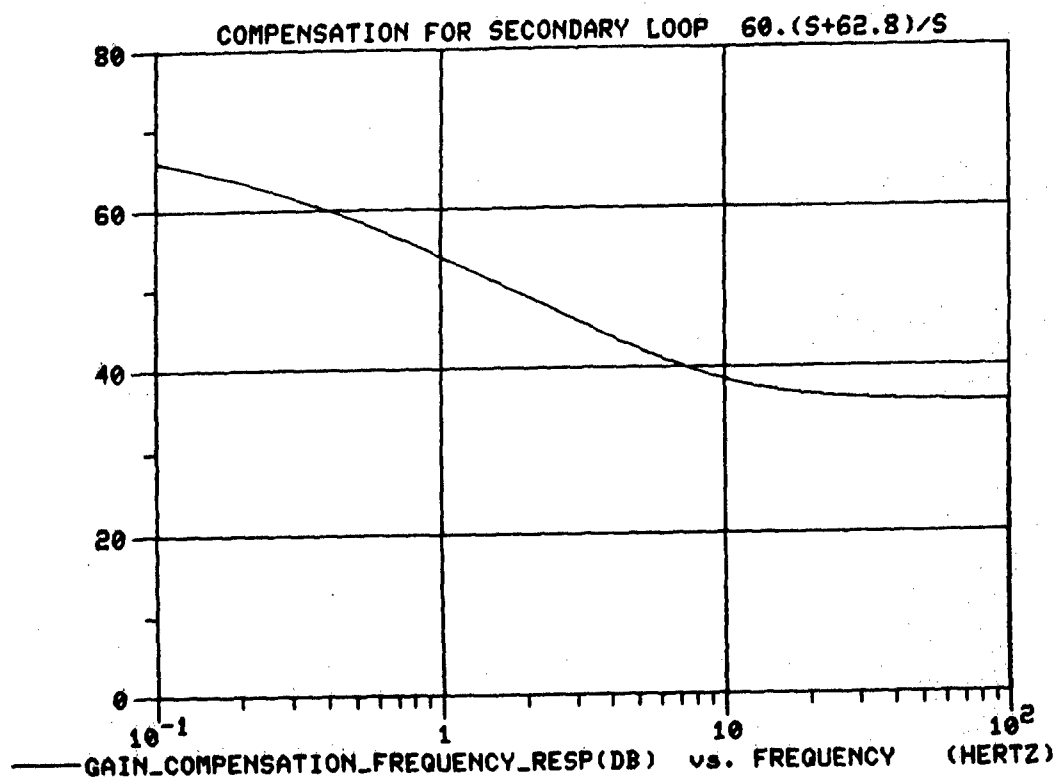


FIGURE 5-12

Compensation Frequency Response of Rate Loop

Gain



Phase

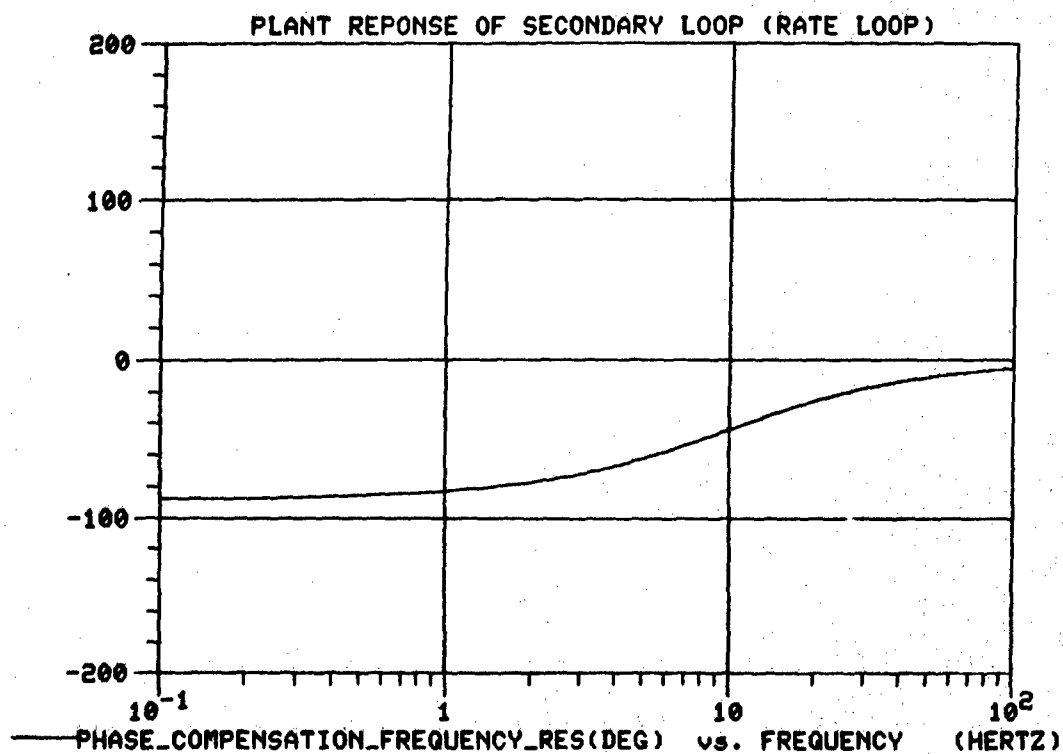


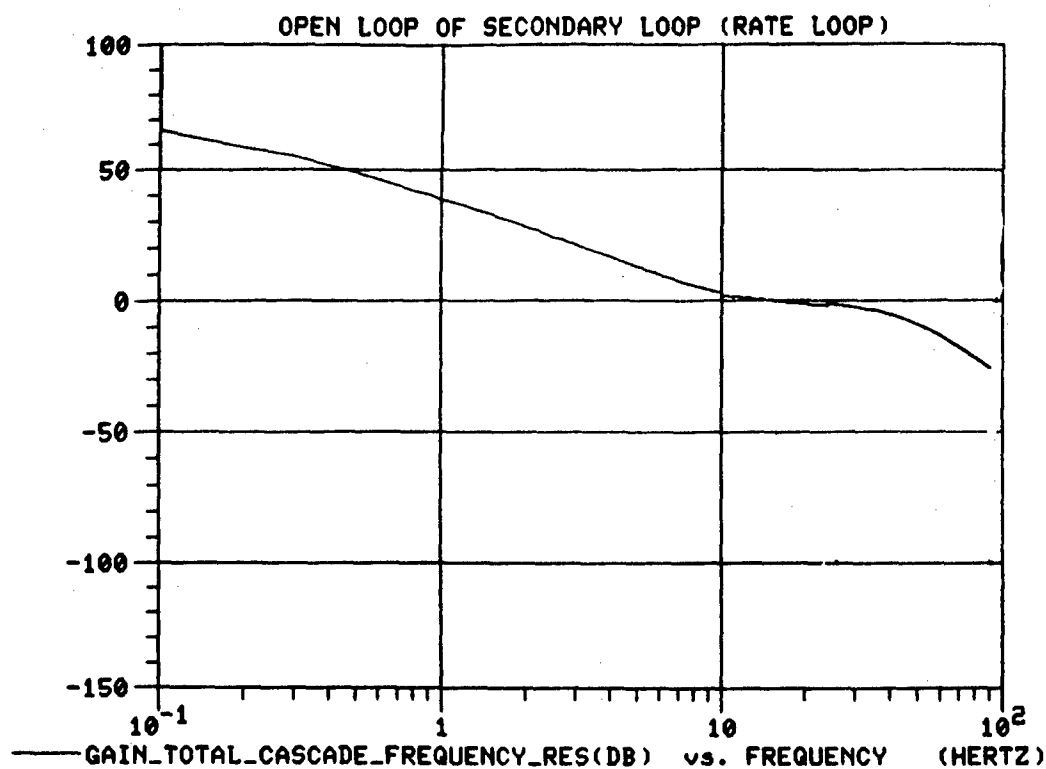
FIGURE 5-13

Open Loop Frequency Response of Rate Loop

Gain Margin = 5. dB

Gain Cross 14.7 Hz

Gain



Phase

Phase Margin = 58°

Phase -180° Cross 38.9 Hz

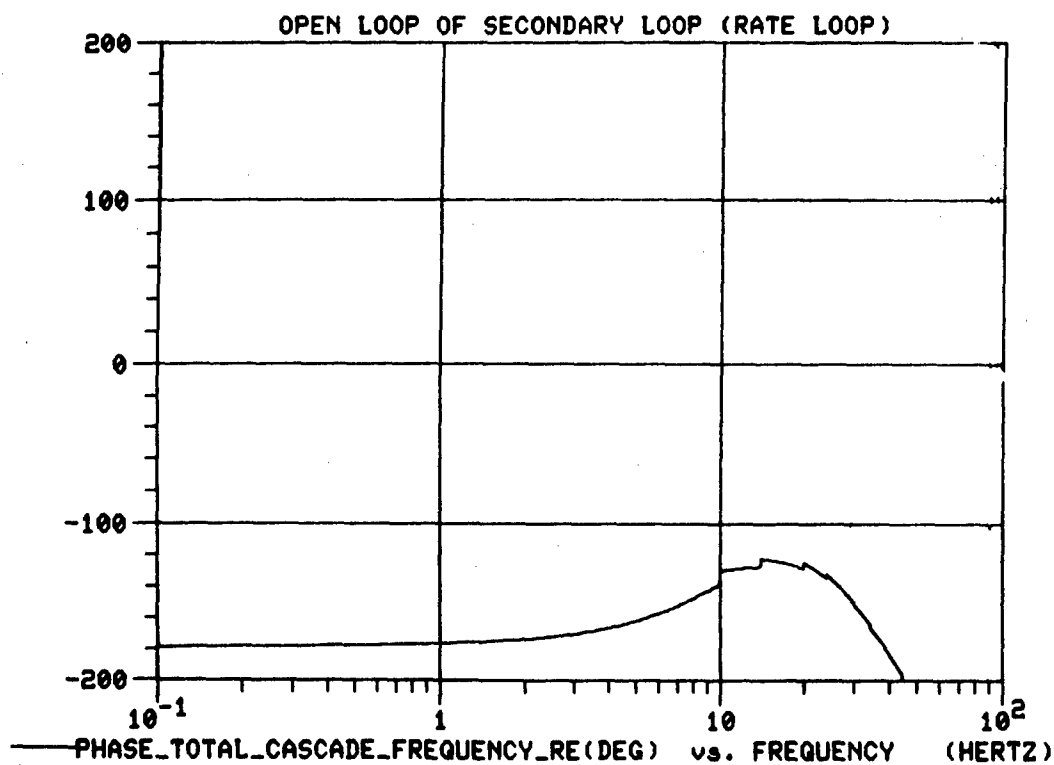
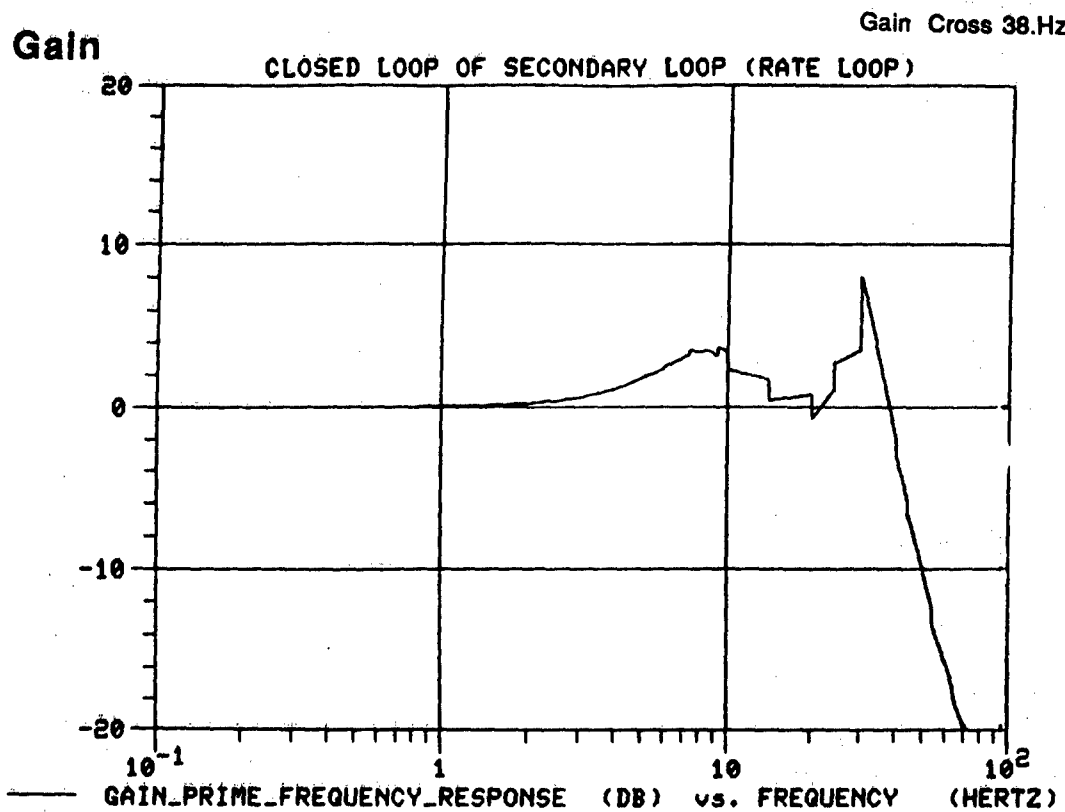


FIGURE 5-14

Closed Loop Response of Rate Loop



Step Response

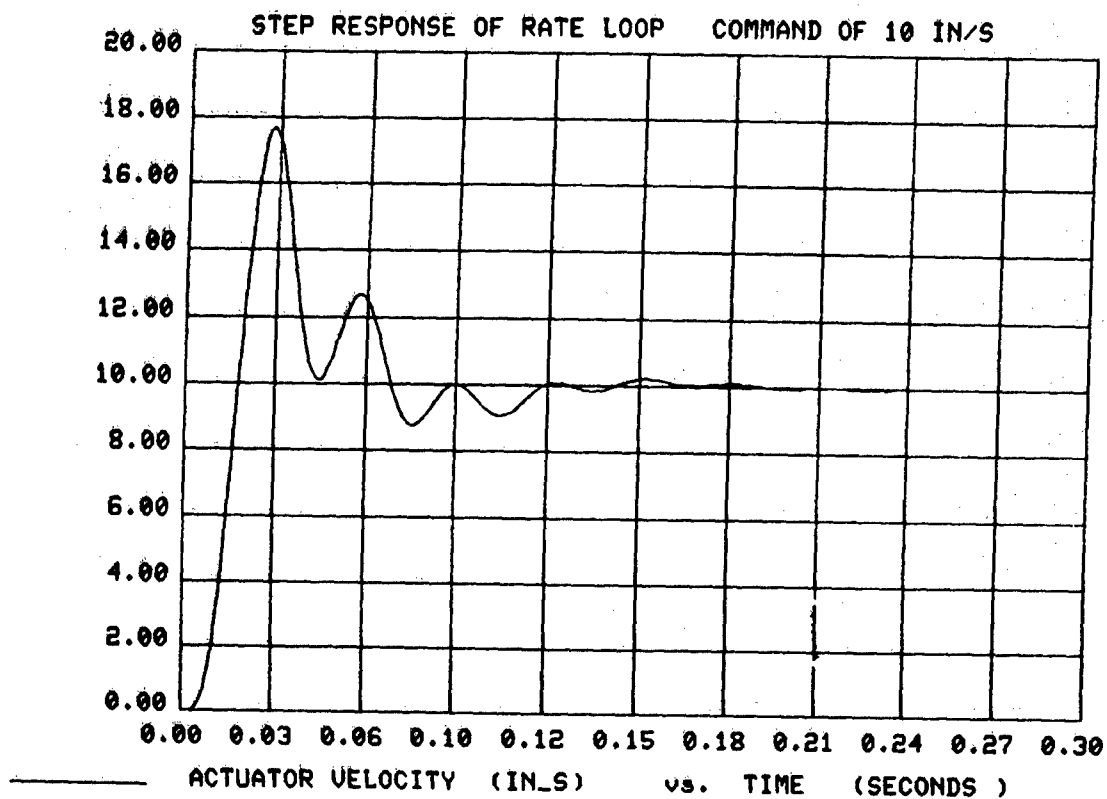


FIGURE 5-15

Position Loop Block Diagram

(Outer Loop)

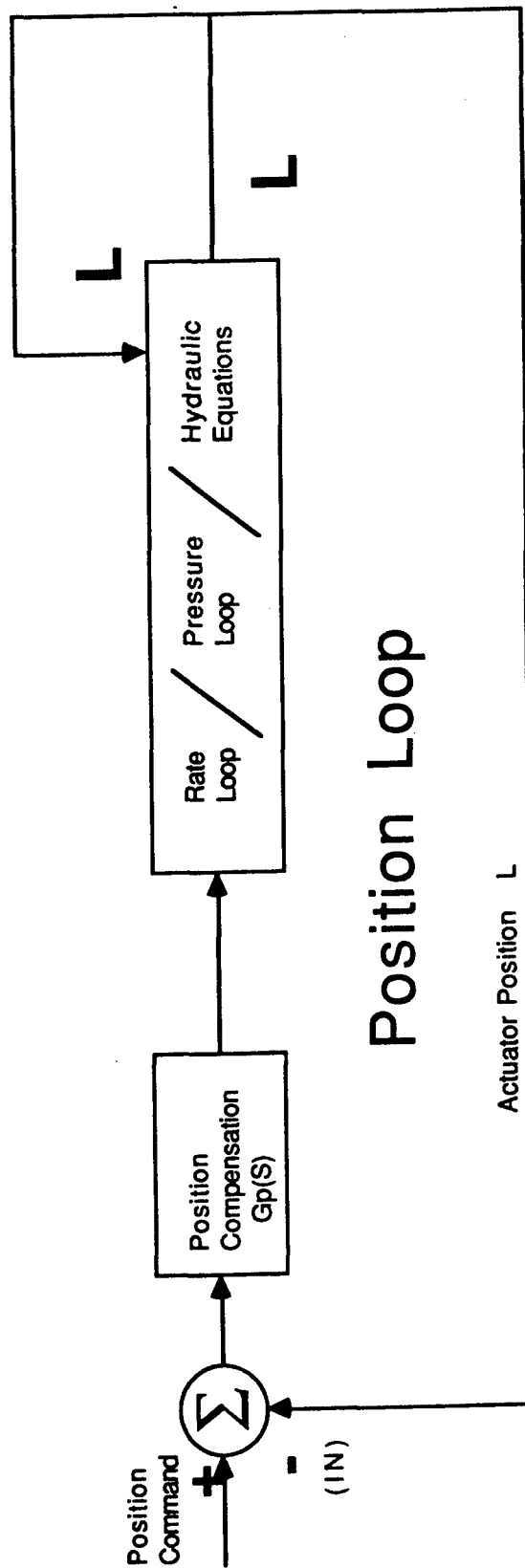
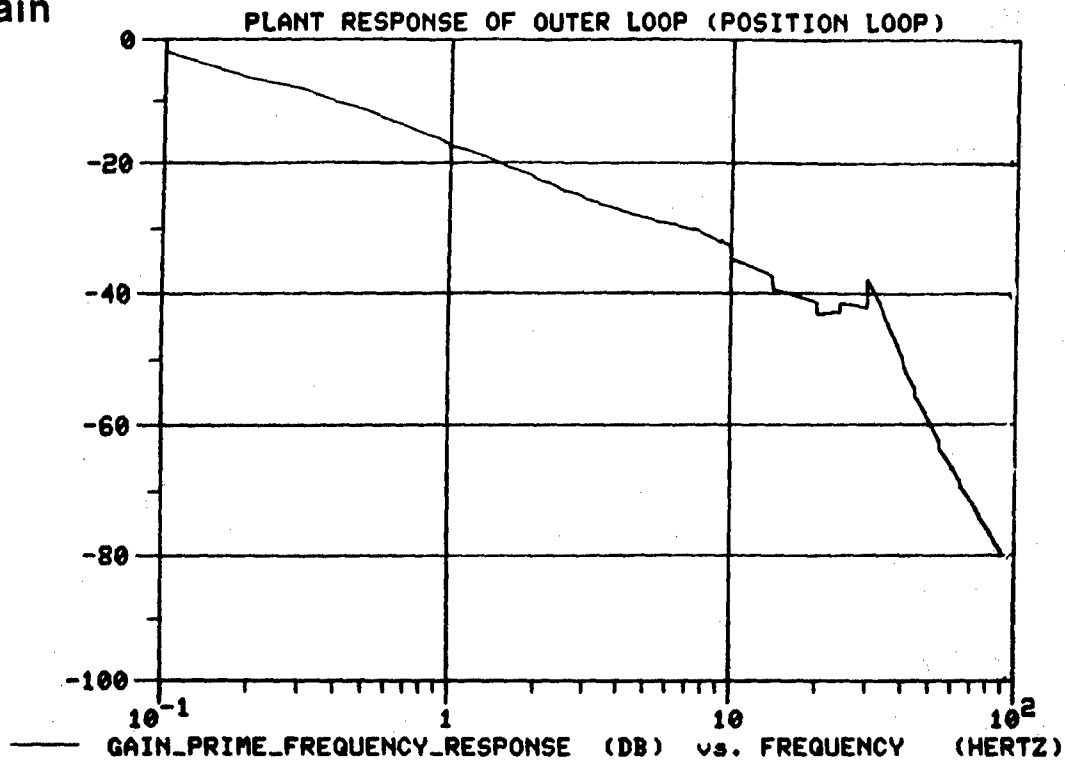


FIGURE 5-16

Plant Frequency Response of Position Loop

Gain



Phase

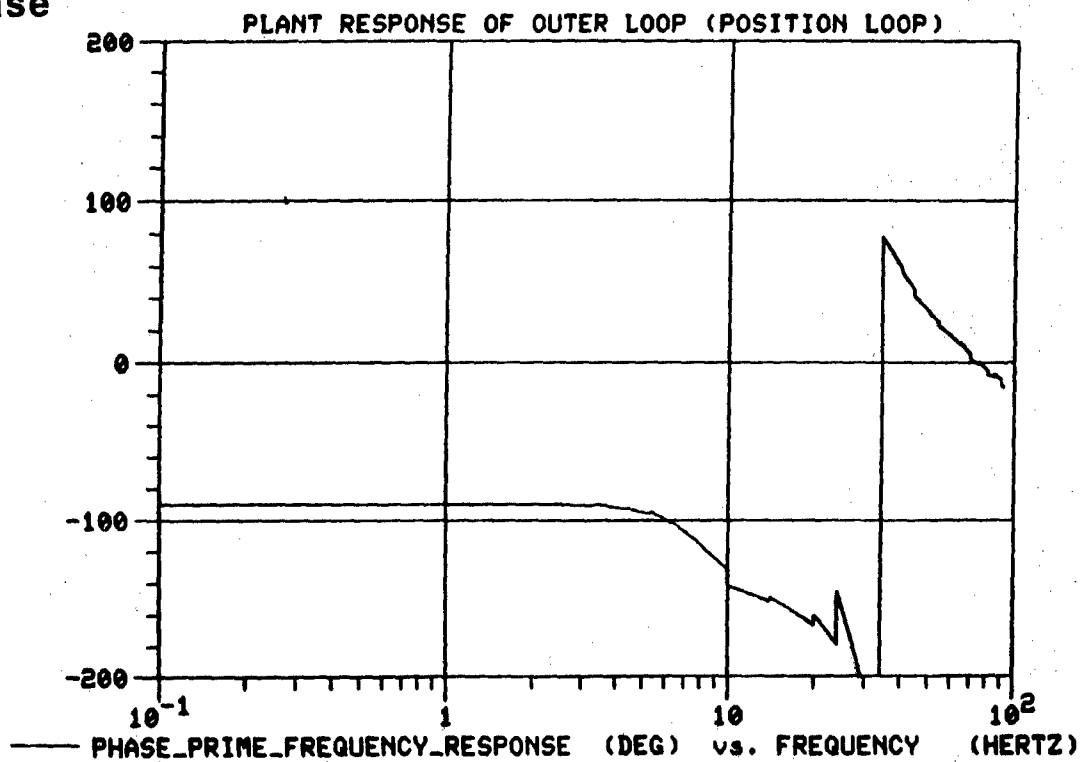
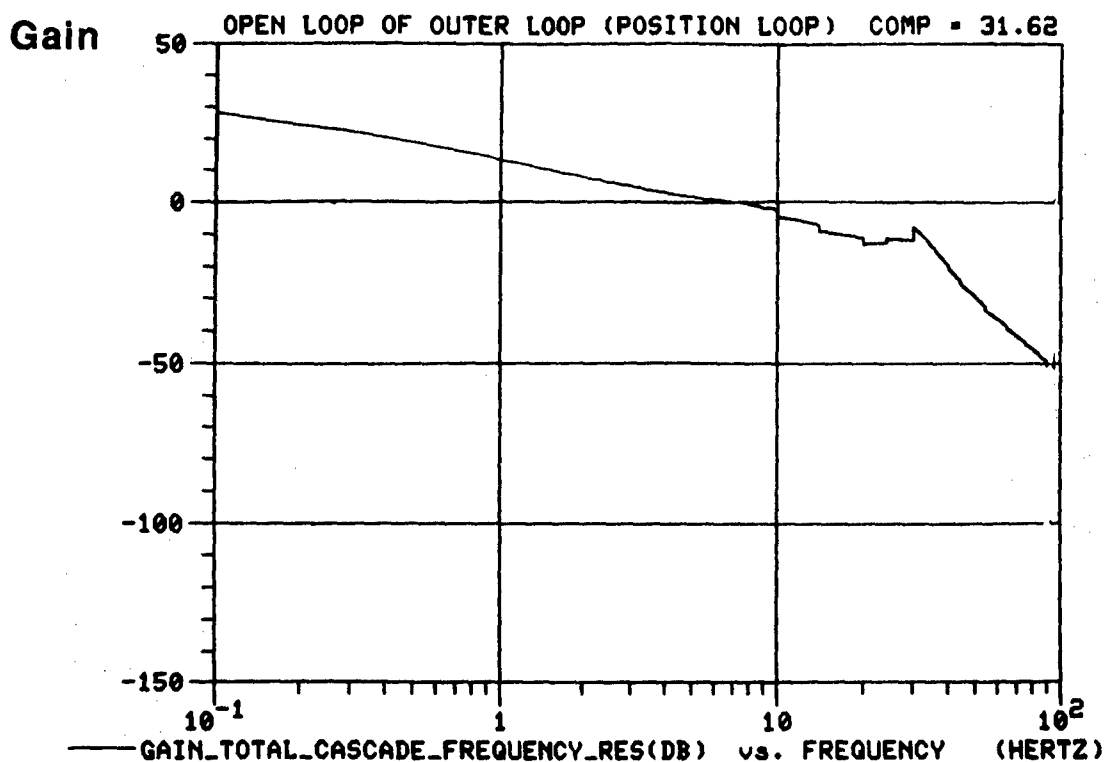


FIGURE 5-17

Open Loop Frequency Response of Position Loop

Gain Margin = 11.7 dB

Gain Cross 7.1Hz



Phase Margin = 75°

Phase -180° Cross 27.5 Hz

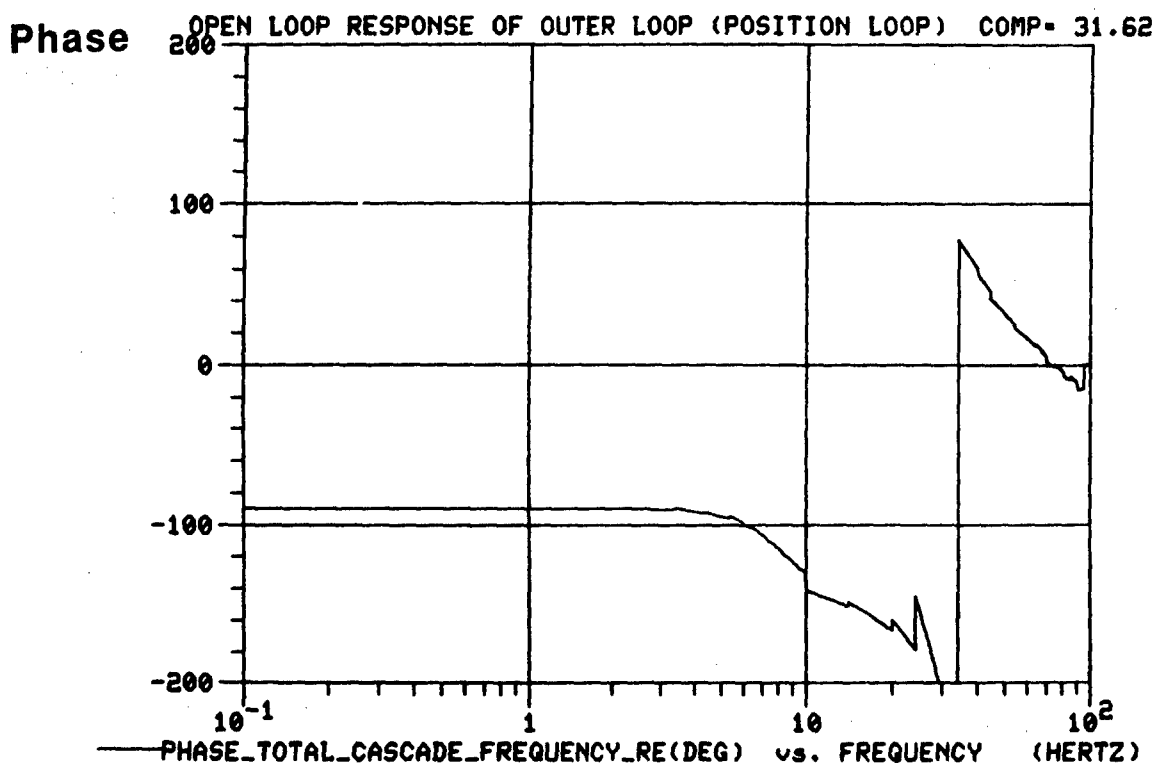
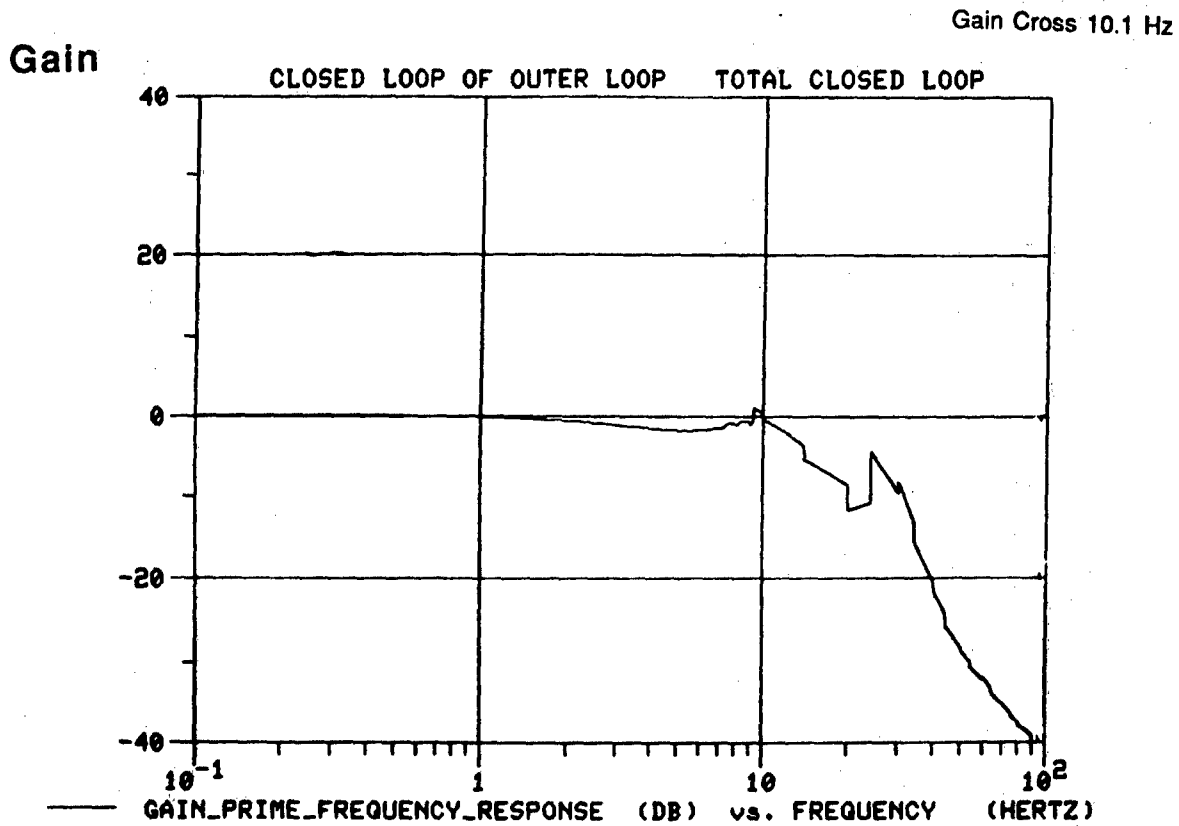


FIGURE 5-18

Closed Loop Response of Position Loop



Step Response

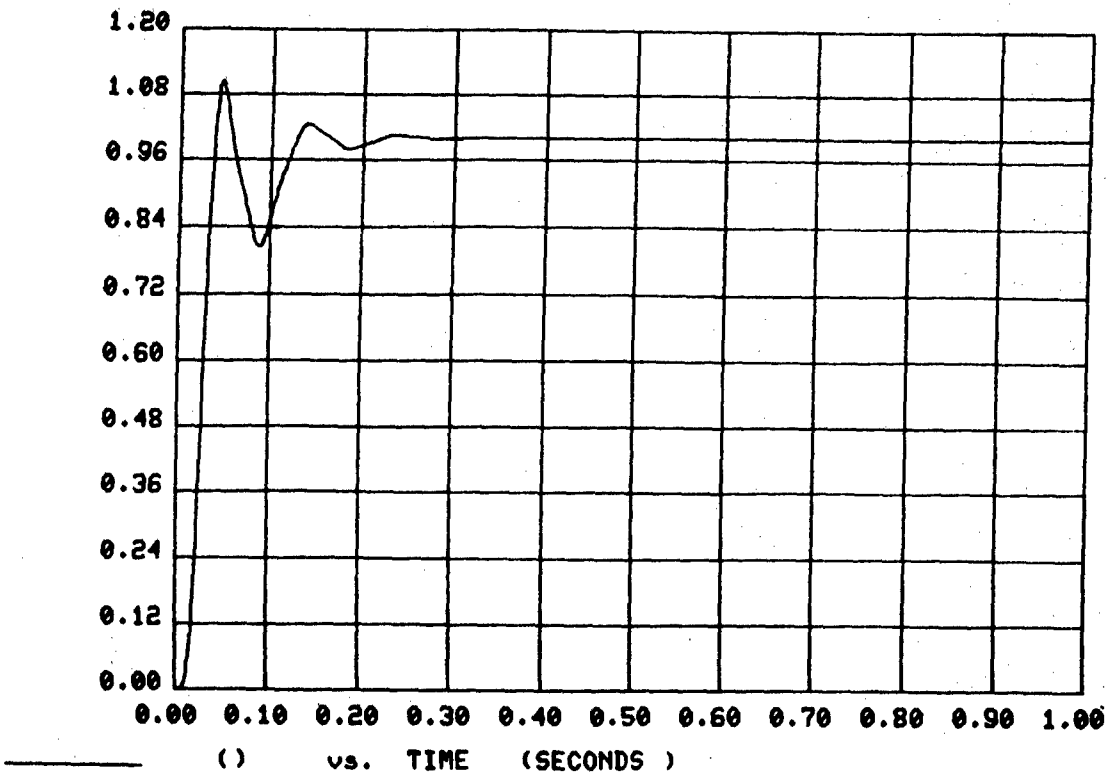


FIGURE 5-19

Results Of Controller Design

	Compensation	Stability Phase	Margins Gain	Closed Loop Cross (0 dB)	Step Response		
					Command	Settling Time	Overshoot
Pressure Loop	$G_p(s) = \frac{.633 (s + 62.8)^2}{s^2}$	28.9 °	10.2 dB	62 Hz.	1500 PSI	.1 S	26 %
Rate Loop	$G_r(s) = \frac{60. (s + 62.8)}{s}$	58.0 °	5.0 dB	38 Hz	10 IN/S	.12 S	75 %
Position Loop (Total Loop)	$G_p(s) = 31.6$	75.0 °	11.7 dB	10.1 Hz	1. IN	.2 S	10 %

TABLE 5-3

Shown in Appendix C are the results of a simulated step response. This Appendix includes various simulated states of the system such as pressures, flows, entrained volumes, spool displacement etc. A step response is obtained by applying a position step command to the system. A step response is used in the analysis to verify control design and give an indication to the performance of the system. This type of command signal may not always be practical to directly apply to real actuator systems due to harshness. This is especially true for a position command signal. As can be seen by the plots that the states are driven to large values the first few milli-seconds of the simulation. These values point to additional limits which should be incorporated in the model. The servo current has a spike value of 2000 milliamps (2 amps) which is an excessive value.

Investigations are being made on filtering techniques which will filter command signals before they are applied to the system. Previous laboratory testing consisted of detrend and filtering processes which modify command signals as described in some detail in Reference 5. In addition low pass filters are used in the laboratory which essentially smooth command signals. This would round off command signals representing step functions (Filter high frequency content). A command signal which has been filtered would reduce some of the enormous overshoots revealed by these simulations.

Appendix D includes the simulated actuator response to various levels of step command. The response changes as the level of command is varied, illustrating the non-linear characteristics of the system. The control design procedure introduced in this report could be conducted at various signal levels to produce a compensation design for a given signal level. This form of adaptive control could be incorporated in software to interactively select the appropriate control compensation for a given signal level. With this design a control design could be made which gives good results for all operating regions of the system.

Appendix E shows the actuator responses to a variation of mass. These results show that as the actuator is loaded with mass the transient response is drastically changed. In fact the system becomes unstable as the mass is increased beyond 60 slugs. This is due to the influence mass has on the plant response characteristics of the system which in turn has a direct impact on the stability margins. To some extent, a change in gain of the rate loop should compensate (adjust) to variation in mass. However, this does not completely improve the system response as bandwidth performance is expected to decrease. These characteristics stress the importance of adaptive control for the TMBS where variations in inertia are expected to reflect on each of the actuator's stability and performance. Additional concern is the coupling effects that each actuator will induce on each other. An analytical investigation will cover these problems in the near future.

LIST OF REFERENCES

1. Helinski, A. L. , "Deriving an Empirical Model of an Electrohydraulic Actuator System from Frequency Response Data", RDE Center Technical Report No. 13368, U. S. Army Tank-Automotive Command, Warren, MI (June, 1988)
2. Kaminski, A. P., "A Mathematical Representation of the M60A1 Azimuth and Elevation Control", General Dynamics (Chrysler Corporation- Defense Division) System Analysis Technical Report (November 1971)
3. CADS Inc., "DADS Rev. 5.0 Supplement Manual", DADS Control/Hydraulic Element Theoretical Manual. Computer Aided Kinematics and Dynamics of Mechanical Systems. 1988
4. Various unpublished notes from Contraves Goertz Corporation related to Turret Motion Base Simulator (TMBS) hydraulic system. Includes an AD10 code and DADS model listings. (1987)
5. Helinski, A. L. , "Simulation Test of the MK19 MOD3 Grenade Machine Gun Support Kit", RDE Center Technical Report No. 13297, U. S. Army Tank-Automotive Command, Warren, MI (October 1987)

APPENDIX A
LIST OF HYDRAULIC ACTUATOR COMPUTER MODEL

```

PROGRAM SA_CLOSED_LOOP
" "
" SINGLE UNCOUPLED ACTUATOR SYSTEM MODEL for TMBS
" "
cinterval cint=0.001
ARRAY GFREQ(50)
REAL DOUT(9000)
INTEGER INDEX,NTABLE "$USED FOR SAMPLING INDEX"
" The following arrays "
" are used to describe the transfer function coefficient"
ARRAY SERVN,SERVD(3)
" SERVO TRANSFER FUNCTION COEFFICIENTS"
CONSTANT SERVN = 1.5E-4
CONSTANT SERVD = 2.533E-6 , .002228 , 1.
" INITIAL ENTRAINED VOLUME and RING MASS "
CONSTANT V10=1697. , V20=1336. , RMASS=194.26
" MECHANICAL PARAMETERS FOR VALVE"
" BULK - OIL BULK MODULAS"
CONSTANT CD=100. , SQRO=1. , BULK = 100000.
" VALVE SPOOL CIRCUMFERENCE"
CONSTANT W1=3.787 , W2=3.787
" UP - LOW CYLINDER AREA"
CONSTANT AREA1=38.5 , AREA2=38.5
" SUPPLY & RETURN PRESSURE"
CONSTANT PS=3000. , PR=100.
" "
INITIAL
INDEX=0
CALL READFREQ(GFREQ,GLEVEL,NTABLE=)
END "$" OF INITIAL "
*****
DERIVATIVE
" ***** LEG COMMAND SIGNAL *****"
" LE1= LEG ERROR"
" LC1= LEG COMMAND"
" L1= LEG DISPLACEMENT"
" "
" GENERATE SIGNAL FOR FREQUENCY ANALYSIS "
" LC1 POSITION COMMAND "
" "
PROCEDURAL(LC1=GFREQ,GLEVEL,NTABLE,T)
CALL GENERATE(LC1,GFREQ,GLEVEL,NTABLE,T)
END"$" OF PROCEDURAL-GENERATE"
" "
LE1 = LC1 - L1
" "
" ***** POSITION LOOP COMPENSATION *****"
" RC1= COMPENSATED LEG ERROR SIGNAL"
RC1 = KP*(LE1+OMEGP*INTEG(LE1,0.))
" "
" ***** VELOCITY LOOP COMPENSATION *****"
" LDE1= LEG VELOCITY COMMAND"
" LD1= LEG VELOCITY SIGNAL"
" PRC1= PRESSURE COMMAND ??????"
LDE1 = RC1 - LD1
" "
KT = 60.
OMEGT = 62.8
PRC1 = KT*(LDE1+OMEGT*INTEG(LDE1,0.))
" "
" ***** PRESSURE LOOP COMPENSATION *****"

```



```

ACTUATOR:      Ln   POSITION (IN)
               LDn   RATE    (IN)

***** CONDITION THE SPOOL POSITION XV *****
***** FOR UPPER AND LOWER CONSIDERATION *****
*****

XVD1 for spool down
XVU1 for spool up

XVD1= RSW(XV1 .LT. 0.,-XV1,0.)$ " SPOOL DOWN"
XVU1= RSW(XV1 .GT. 0.,XV1,0.)$ " SPOOL UP "

FLOW COEFFICIENT Cd
Q = C*X*SQR(T(DELTA P))
where C=Cd*W1*SQR(2/DENS) W1=PI*D

***** FLOW FROM PRESSURE DROPS *****
QA1 = CD*SORO*W1*XVU1*SQR(ABS(P5-PUI))*SIGN(1.0,P5-PUI)
QB1 = CD*SORO*W1*XVD1*SQR(ABS(PUI-PR))*SIGN(1.0,PUI-PR)
QC1 = CD*SORO*W1*XVD1*SQR(ABS(P5-PD1))*SIGN(1.0,P5-PD1)
QD1 = CD*SORO*W1*XVU1*SQR(ABS(PDI-PR))*SIGN(1.0,PDI-PR)

***** CYLINDER VOLUME VOLUn & VOLDn *****
VOLU1 = V10 + L1*AREA1
VOLD1 = V20 - L1*AREA2
***** DERIVATIVE OF PRESSURE PDOTUn & PDOTDn*****
PDOTU1 = (BULK/VOLU1)*(QA1 - QB1 - AREA1*LD1)
PDOTD1 = (BULK/VOLD1)*(QC1 - QD1 + AREA2*LD1)

***** INTEGRATE FOR PRESSURE PUI & PD1 *****
PUI = INTEG(PDOTU1,0.)
PD1 = INTEG(PDOTD1,0.)

***** DELTA PRESSURE ACROSS PISTON *****
DELTPI = PUI - PD1

***** ACTUATOR FORCE Fn *****
F1=AREA1*PUI - AREA2*PD1 - FRICT $ " MUST LOGIC THE FRICTION"

**** SIMPLIFY FOR NOW ACTUATOR VELOCITY AND POSITION ****
ACTUATOR FORCE Qn"

MASS=RMASS/6. $ " Assume a single unconstrained mass for now"
LD1 = (1./MASS)*INTEG(F1,0.) $ "actuator velocity inches"
L1 = INTEG(LD1,0.)

END $"OF DERIVATIVE"
*****
*** DISCRETIZE THE OUTPUT INTO ARRAY FOR WRITING OUTPUT ***
*****
DISCRETE SAMPLE
INTERVAL DTSMIP=.0012207
STEP=.0012207
INDEX=INDEX+1
DOUT(INDEX2)=L1

```

```

END $"OF DISCRETE SAMP 2"
* *****
DERIVATIVE
  termt(INDEX2 .GE. 4098)
END $"OF DERIVATIVE"
*
TERMINAL
  CALL WRITEERD(=STEP,INDEX2,DOUT)
END $"OF TERMINAL"
END $"OF PROGRAM"

SUBROUTINE READFREQ(GFREQ,GLEVEL,NTABLE)
  REAL GFREQ(50)
  CHARACTER*30 FILEN
*
  WRITE(5,100)
100  FORMAT('//////////.
*      ' Enter The File with the frequency table?'.//,
*      ' Example: freqs.dat')
  READ(5,140)FILEN
140  FORMAT(A30)
  OPEN(10,FILE=FILEN,FORM='FORMATTED',SHARED,
*  STATUS='OLD',ERR=210)
  READ(10,400)GLEVEL
400  FORMAT(/,G10.4)
  NTABLE=1
800  READ(10,600)GFREQ(NTABLE)
600  FORMAT(G10.4)
  IF(GFREQ(NTABLE) .LT. 0.)GOTO 300
  NTABLE=NTABLE+1
  GOTO 800
210  WRITE(5,170)
170  FORMAT(' ERROR OPENING FILE')
300  CLOSE(10)
  RETURN
  END

SUBROUTINE GENERATE(SINPUT,GFREQ,GLEVEL,NTABLE,T)
*
*  This subroutine creates the sweep wave (Sum of sine waves)
*  for creating the input signal to a frequency response
*  analysis to determine transfer function characteristics.
*
  REAL GFREQ(50)
  SINPUT=0.
  PI=3.141592654
  DO 100 II=1,NTABLE
    SINPUT=SINPUT + GLEVEL*SIN(2*PI*T*GFREQ(II))
100  CONTINUE
  RETURN
  END

SUBROUTINE WRITEERD(STEP,NSAMP,OUTPUT)
*
  CHARACTER*80 ERD_TITLE, LONG_TITLE, DUMMY80
  CHARACTER*64 ERD_FILE, HDR_FILE, ERD_FILE_O, HDR_FILE_O
  CHARACTER*32 LONG_NAME(16), DUMMY32
  CHARACTER*12 DUMMY
  CHARACTER*8 SHORT_NAME(16), UNIT_NAME(16), XUNIT, DUMMY8
  CHARACTER*4 ERD, HDR
  CHARACTER*1 COMMA, REPLY

```

```

CHARACTER*2 IOPERATE(20)
DIMENSION SMAX(18), SMIN(18)
REAL*4 SCALE(16), OFFSET(16), DATA(16,30000)
REAL RMS(16), SMEAN(16), OUTPUT(7000)
INTEGER*4 START SAMP ELIM, END SAMP ELIM
INTEGER*2 ERD_UNIT, HDR_UNIT, IDATA(16)
LOGICAL*4 TIME
LOGICAL*1 NEWCHAN, RECHAN
DATA ERD_UNIT, HDR_UNIT/10,11/
ERD = '.ERD'
HDR = '.HDR'
*****
*
WRITE(5,1881)
1881 FORMAT(///, ' ENTER how many channels are output')
READ(5,*)NCHAN
*
DO 1998 J=1,NCHAN
WRITE(5,1888)J
1888 FORMAT(///, ' ENTER the LONG NAME for channel ',I2)
READ(5,1889)LONG_NAME(J)
1889 FORMAT(A32)
WRITE(5,1992)J
1992 FORMAT(///, ' ENTER the SHORT NAME for channel ',I2)
READ(5,1920)SHORT_NAME(J)
1920 FORMAT(A8)
WRITE(5,1993)J
1993 FORMAT(///, ' ENTER the UNIT NAME for channel ',I2)
READ(5,1920)UNIT_NAME(J)
*
OFFSET(J)=0.
SCALE(J)=1.
*
1998 CONTINUE
*
DO 1766 ISMP=1,NSAMP
DATA(1,ISMP)=OUTPUT(ISMP)
1766 CONTINUE
*
WRITE(5,201)
201 FORMAT(///, ' Indicate how new data file is to be stored.',/,
+ ' 0 = 2 byte integer (binary)',/,
+ ' 1 = 4 byte floating point (binary)',/,
+ ' 2 = 8 byte floating point (binary)',/,
+ ' 3 = 8 byte complex (binary)',/,
+ ' 4 = 16 byte complex (binary)',/,
+ ' 5 = formatted floating point. The format is (Nchannels)E13.',
+6',/, '$Enter selection (0-5):(1 CHOSEN MOST COMMONLY) ')
READ(5,*) KEYNUM
*
* do not let the user choose complex numbers
*
IF (KEYNUM .EQ. 3 .OR. KEYNUM .EQ. 4) THEN
TYPE*, '
TYPE*, 'Choose another format besides complex numbers.'
ELSE IF (KEYNUM .LT. 0 .OR. KEYNUM .GT. 5) THEN
TYPE*, '
TYPE*, 'Selection out of range.'
ENDIF
*
* open and create files

```

```

*
*
* begin writing header information
*
  WRITE(5,265)
265  FORMAT(//,'Enter name of the data file to write to: ',
+/, ' ERD FORMAT ASSUMED')
  READ(5,267) ERD_FILE_O
267  FORMAT(A32)
* Create the two file names
*
  WRITE(5,4446)
4446  FORMAT('Enter ERD title?')
  READ(5,4447) ERD_TITLE
4447  FORMAT(A80)
  HDR_FILE_O = ERD_FILE_O
  CALL STRTRIM(HDR_FILE_O, ERD_FILE_O, LENGTH)
  HDR_FILE_O(LENGTH+1:LENGTH+4) = HDR
  ERD_FILE_O(LENGTH+1:LENGTH+4) = ERD
  OPEN(HDR_UNIT, FILE=HDR_FILE_O, STATUS='UNKNOWN',
+ FORM='FORMATTED', RECL=256)
*
* WRITE OUT HEADER DATA
*
*
* UNKNOWN KNOWN
*
  DUMMY = 'ERDFILEV1.00'
  KEYOPT=0
  NLines=0
  NBIN=-1
  NBin=-1
  NByte=-1
  COMMA=', '
  WRITE(HDR_UNIT,270) DUMMY
270  FORMAT(A12)
  WRITE(HDR_UNIT,280) ERD_TITLE
280  FORMAT(A80)
  WRITE(HDR_UNIT,290) NCHAN, COMMA, NSAMP, COMMA, NLines, COMMA, NBIN,
& COMMA, NByte, COMMA, KEYNUM, COMMA, STEP, COMMA, KEYOPT
290  FORMAT(6(I7,A),E13.6,A,I7)
  WRITE(HDR_UNIT,300) SCALE(1),((COMMA,SCALE(J)),J=2,NCHAN)
300  FORMAT(18(E13.6,A))
  WRITE(HDR_UNIT,300) OFFSET(1),((COMMA,OFFSET(J)),J=2,NCHAN)
  WRITE(HDR_UNIT,310) (SHORT_NAME(J),J=1,NCHAN)
310  FORMAT(31(A8))
  WRITE(HDR_UNIT,320) (LONG_NAME(J),J=1,NCHAN)
320  FORMAT(7(A32))
  WRITE(HDR_UNIT,310) (UNIT_NAME(J),J=1,NCHAN)
*
* write 9+ lines to file
*
C  TYPE*, ' '
  IF(NLines .EQ. 0) GOTO 330
*  TYPE*, 'Enter additional descriptor lines'
  DO 330 J=1,NLines
*    READ(5,280) LONG
    WRITE(HDR_UNIT,6560) J, SHORT_NAME(J), RMS(J), SMEAN(J)
6560  FORMAT('CHAN ',I2,3X,A8,3X,' RMS= ',E15.3,4X,' MEAN= ',E15.3)
330  CONTINUE
*

```

```

CLOSE(HDR_UNIT)
* write data to file
*
  IF (KEYNUM .EQ. 5) THEN
    OPEN(ERD_UNIT, FILE=ERD_FILE_0, FORM='FORMATTED',
    + STATUS='UNKNOWN', RECL=256)
  ELSE
    OPEN(ERD_UNIT, FILE=ERD_FILE_0, FORM='UNFORMATTED',
    + STATUS='UNKNOWN', RECL=256)
  ENDIF
* 2 byte integer
*
  IF (KEYNUM .EQ. 0) THEN
    DO 350 J=1, NSAMP
      WRITE(ERD_UNIT, ERR=406)
    + (IIFIX(DATA(L,J)), L=1, NCHAN)
350    CONTINUE
*
* 4 byte floation - binary
*
    ELSE IF (KEYNUM .EQ. 1) THEN
      DO 370 J=1, NSAMP
        WRITE(ERD_UNIT, ERR=406)
    + (DATA(L,J), L=1, NCHAN)
370    CONTINUE
*
* 8 byte floating - binary
*
    ELSE IF (KEYNUM .EQ. 2) THEN
      DO 390 J=1, NSAMP
        WRITE(ERD_UNIT, ERR=406)
    + (DBLE(DATA(L,J)), L=1, NCHAN)
390    CONTINUE
*
* formatted output
*
    ELSE
      DO 410 J=1, NSAMP
        WRITE(ERD_UNIT, 405, ERR=406)
    + (DATA(L,J), L=1, NCHAN)
410    CONTINUE
405    FORMAT(19(E13.6))
      GOTO 35
406    WRITE(5, 407) J-1
407    FORMAT(/, 'There were ', I10, ' records written out before the f1
+1e filled up.', /, ' Change NSAMP in the header file accordingly.')
      ENDIF
35    CLOSE(ERD_UNIT)
*
    RETURN
*
    END

```

APPENDIX B
LIST OF FORTRAN PROGRAM FOR FREQUENCY RESPONSE ANALYSIS

PROGRAM BODE

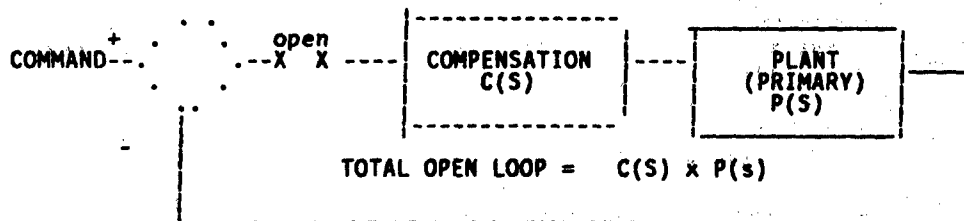
***** Written by A. L. HELINSKI, US ARMY TACOM AMSTA-RV *****

 ***** FOURIER ROUTINES were developed by IEEE "Programs *****
 ***** for Digital signal processing." Although any Fast *****
 ***** Fourier program which determines the Real and *****
 ***** Imaginary Fourier coefficients from a time *****
 ***** history could be used. *****

PURPOSE

This program is used to plot the frequency response of either the results of a simulated model or by directly putting in the polynomial transfer function coefficients. The simulated model option must have the ERD file of the results. Simulation must involve using the sum of sine waves as input. The resulting file of the output of the transfer function time history must be in ERD format. In addition the sum of sine waves (input of transfer function) will be re-generated by giving the same table of frequencies used in the simulation.

In addition the option is available to design cascade compensation if the prime frequency response is describing the plant response as shown in the configuration below. The compensation frequency response can also be plotted separately or combined with the Primary (Plant) to generate a total open loop response for this option. An option is available to determine crossover points representing stability margins



OBJECTIVE

The main objective to developing this program is to have a simple means of observing a frequency response with easy access to changes in the transfer function. (Compensation) It is also beneficial in designing compensation for a non-linear plant model by supplementing this program with a time domain simulation software package like ACSL or MIMIC.

INPUT SIGNAL VARIABLES

DIMENSION SINP(10000)	! TIME HISTORY TO BE GENERATED
DIMENSION AMPI(50)	! AMPLITUDE
DIMENSION DBI(50)	! DB VALUE
DIMENSION PHASEI(50)	! PHASE
DIMENSION REALI(5000)	! REAL PART

```

        DIMENSION AIMAGI(5000) ! IMAGINARY PART
        DIMENSION GFREQ(50) ! TABLE OF FREQUENCIES USED
*   OUTPUT SIGNAL VARIABLES
        DIMENSION SOUT(10000) ! TIME HISTORY RECORDED FROM ACSL
        DIMENSION AMPO(50) ! AMPLITUDE
        DIMENSION DBO(50) ! DB VALUE
        DIMENSION PHASEO(50) ! PHASE
        DIMENSION REALO(5000) ! REAL PART
        DIMENSION AIMAGO(5000) ! IMAGINARY PART
*   TIME & FREQUENCY VALUES
        DIMENSION FREQ TF(50),FREQ(5000),SIGTIM(10000)
*   TRANSFER FUNCTION VARIABLES
        DIMENSION DB TF(50),PHASE TF(50)
** DATA HDR INFO *****
        REAL*4 DATA(10000,6)
        CHARACTER*8 UNIT_NAME(6),SHORT_NAME(6)
        CHARACTER*32 LONG_NAME(6)
*
*   POLYNOMIAL INPUTS
*
        REAL N MAT(20,20)
        INTEGER NODR(20),NPLYS,N_DEG
        REAL POLY_N(40)
*
        REAL D MAT(20,20)
        INTEGER DODR(20),DPLYS,D_DEG
        REAL POLY_D(40)
*
*
** COMPENSATION POLYS
*
        REAL C N MAT(20,20)
        INTEGER C NODR(20),C_NPLYS,C_N_DEG
        REAL C_POLY_N(40)
*
        REAL C D MAT(20,20)
        INTEGER C DODR(20),C_DPLYS,C_D_DEG
        REAL C_POLY_D(40)
*
*****
        INTEGER*4 NPOINT,NSAMP
        CHARACTER*1 REPLY,NORD
        CHARACTER*8 UNITS
        CHARACTER*32 CHANNEL
        CHARACTER*80 NOTE
        CHARACTER*20 FILETIT
*****
*
        PI=3.141592654
        SCAL=180./PI
        EPSILON=10.E-30 ! Used for detection of 20LOG(small)
*
*
*****
*   INITIALIZE Set all channels of data to zero
*****
        DO 789 J=1,10000
            DO 790 K=1,6
                DATA(J,K)=0.
790         CONTINUE
789     CONTINUE

```

```

*
* Determine the users terminal type for later use.
*
5      WRITE(5,6)
6      FORMAT(///,' Enter the terminal identifier code:',/,
&      '      1 --> VT240',/,
&      '      2 --> TAB',/,
&      '      3 --> TEKTRONIX 40XX',/,
&      '      4 --> TEKTRONIX 41XX',/,
&      '$Enter id: ')
      READ(5,*) ITERM
*
5566   WRITE(5,7050)
7050   FORMAT(//////////,
+      ' This program evaluates transfer fucntion',
+      ' frequency response.',/, ' It is used to plot frequency',
+      ' response and design compensation.',/,/,
+      ' Choose the desired analysis for the PRIMARY frequency',
+      ' response (Plant):',/,/, ' (1) ACSL simulation-TIME HISTORY',/,
+      ' Interpolation will be included',/,
+      ' (Must know ERD file generated from ACSL and',/,
+      ' frequency table file used in ACSL)',/,/,
+      ' (2) Enter Frequency/Magnitude/Phase POINTS by hand',/,
+      ' Interpolation will be included',/,/,
+      ' (3) Enter transfer funtion in terms of POLYNOMIAL',/,
+      ' COEFFICIENTS',/,/, ' Enter 1, 2 or 3')
*
      READ(5,*) IMENU1
*
* Label channels to this program's channel
* configuration for later use.
*
      NCHAN=6
      LONG_NAME(1)='GAIN PRIME FREQUENCY RESPONSE'
      LONG_NAME(2)='PHASE PRIME FREQUENCY RESPONSE'
      LONG_NAME(3)='GAIN COMPENSATION FREQUENCY RESPONSE'
      LONG_NAME(4)='PHASE COMPENSATION FREQUENCY RESPONSE'
      LONG_NAME(5)='GAIN TOTAL CASCADE FREQUENCY RESPONSE'
      LONG_NAME(6)='PHASE_TOTAL_CASCADE_FREQUENCY_RESPONSE'
*
      SHORT_NAME(1)=' '
      SHORT_NAME(2)=' '
      SHORT_NAME(3)=' '
      SHORT_NAME(4)=' '
      SHORT_NAME(5)=' '
      SHORT_NAME(6)=' '
*
      UNIT_NAME(1)='DB'
      UNIT_NAME(2)='DEG'
      UNIT_NAME(3)='DB'
      UNIT_NAME(4)='DEG'
      UNIT_NAME(5)='DB'
      UNIT_NAME(6)='DEG'
*
* Enter a note (title)
*
5290   WRITE(5,5290)
      FORMAT(' Enter a NOTE for this run ')
      READ(5,5294) NOTE
5294   FORMAT(A80)
*

```

```

*****
*      OPTION 1  FREQUENCY RESPONSE FROM TIME HISTORY      *
*      This option used generally for a non-                *
*      linear model where a time history can                *
*      be generated.  Simulation must be from              *
*      sum of sine wave response                            *
*      Need:                                                *
*          1.  ERD file of output time history.            *
*              (Created from ACSL)                          *
*          2.  Table file which contains the                *
*              frequencies used to generate                 *
*              the input.                                    *
*****
*      IF(IMENU1 .EQ. 1)THEN
*
*      -----READ INPUT FILE
*      Read the data file containing the time history of the output
*      signal of the transfer function. The transfer function (model)
*      must be executed by a input containing a sum of sine waves for this
*      option.
*
*      CALL READERD(SIGTIM,SOUT,NPOINT,CHANNEL,UNITS)
*
*      ----- Determine Fourier Coefficients
*      Fourier subroutine computes the fourier coefficients of the output
*      signal in terms of Real and Imaginary terms (Cos & Sin) for each
*      frequency.
*
*      CALL FOURIER(SIGTIM,SOUT,NPOINT,REALO,AIMAGO,FREQ,MM)
*
**      READ FREQUENCIES AND GENERATE THE INPUT SIGNAL (The same
*      as it was done in ACSL.)
*
*      -----Read the same frequency table as ACSL
*      Re-generate the same input signal which was used to excite the model.
*      (Due to memory limitation it was easier to re-generate the signal then
*      to create it in the data file.)
*
*      CALL READFREQ(GFREQ,GLEVEL,NTABLE)
*
*      ----- Generate Sum of Sin Waves of corresponding frequencies.
*
*      DO 771 II=1,NPOINT
*          T=SIGTIM(II)
*          CALL GENERATE(SINPQ,GFREQ,GLEVEL,NTABLE,T)
*          SINP(II)=SINPQ
771      CONTINUE
*
*      ----- Call Fourier for input signal
*      Fourier subroutine determines the fourier coefficients of the input
*      signal.  Theoretically:
*          Real      (Cos Term) = 0.
*          Imaginary (Sin Term) = GVEVEL
*
*      CALL FOURIER(SIGTIM,SINP,NPOINT,REALI,AIMAGI,FREQ,MM)
**
**      DETERMINE AMPLITUDE AND PHASE FOR ONLY THE FREQUENCIES USED
*
*      INDEX=1

```

```

DO 8945 II=1,MM
*****
**** First round off frequency to the nearest mill-Hz *****
**** Note- For example it avoids the problem of 4.999 Hz *****
**** not being recognized as 5 Hz. *****
*****
      FREQ(II)=INT(FREQ(II)*1000+.5)/1000.
*
*****
* Now determine if the frequency is one included in the original *
* generated input signal to include in new array of data. *
*****
*
* ---- Determine if the frequency is included member of roster
* If the frequency is the same as one generated in the input signal
* (Included in roster- Freq-Table member used in input generation) then
* create in the record (Freq-Mag-Phase) otherwise go to next frequency.
*
      IF(FREQ(II) .EQ. GFREQ(INDEX))THEN
*
* Create new frequency array
*
      FREQ_TF(INDEX)=FREQ(II)
*
* Create amplitude array of input *
*
      AMPI(INDEX)=SQRT(REALI(II)**2+AIMAGI(II)**2)
*
* Create dB array.
* Avoid LOG10 of very small numbers
*
      IF(AMPI(INDEX) .LT. EPSILON)THEN
        DBI(INDEX)=-580.      ! -580 Db is small, make as limit
      ELSE
        DBI(INDEX)=20.*ALOG10(AMPI(INDEX))
      ENDIF
*
* Create amplitude and dB array of output signal *
*
      AMPO(INDEX)=SQRT(REALO(II)**2+AIMAGO(II)**2)
      IF(AMPO(INDEX) .LT. EPSILON)THEN
        DBO(INDEX)=-580.      ! -580 Db is small, make as limit
      ELSE
        DBO(INDEX)=20.*ALOG10(AMPO(INDEX))
      ENDIF
*
* Create Phase array of input and output signal
*
      IF(REALI(II) .EQ. 0.)THEN
        IF(AIMAGI(II) .LT. 0.)THEN
          PHASEI(INDEX)=-90.
        ELSE
          PHASEI(INDEX)=90.
        ENDIF
*
** DETERMINE WHICH QUADRANT THE INPUT PHASE IS IN
*
      ELSEIF(REALI(II) .GT. 0. .AND. AIMAGI(II) .GT. 0.)THEN
        RATIOI=ABS(AIMAGI(II)/REALI(II))
        PHASEI(INDEX)=      SCAL*ATAN(RATIOI)

```

```

ELSEIF(REALI(II) .LT. 0. .AND. AIMAGI(II) .GT. 0.) THEN
  RATIOI=ABS(AIMAGI(II)/REALI(II))
  PHASEI(INDEX)=180. - SCAL*ATAN(RATIOI)
ELSEIF(REALI(II) .LT. 0. .AND. AIMAGI(II) .LT. 0.) THEN
  RATIOI=ABS(AIMAGI(II)/REALI(II))
  PHASEI(INDEX)=180. + SCAL*ATAN(RATIOI)
ELSEIF(REALI(II) .GT. 0. .AND. AIMAGI(II) .LT. 0.) THEN
  RATIOI=ABS(AIMAGI(II)/REALI(II))
  PHASEI(INDEX)=360. - SCAL*ATAN(RATIOI)
*
*   ENDIF
*
*   OUTPUT PHASE
*
*   IF(REALO(II) .EQ. 0.) THEN
*     IF(AIMAGO(II) .LT. 0.) THEN
*       PHASEO(INDEX)=-90.
*     ELSE
*       PHASEO(INDEX)=90.
*     ENDIF
*
*   DETERMINE WHICH PHASE THE OUTPUT IS IN
*
*   ELSEIF(REALO(II) .GT. 0. .AND. AIMAGO(II) .GT. 0.) THEN
*     RATIOO=ABS(AIMAGO(II)/REALO(II))
*     PHASEO(INDEX)= SCAL*ATAN(RATIOO)
*   ELSEIF(REALO(II) .LT. 0. .AND. AIMAGO(II) .GT. 0.) THEN
*     RATIOO=ABS(AIMAGO(II)/REALO(II))
*     PHASEO(INDEX)= 180. - SCAL*ATAN(RATIOO)
*   ELSEIF(REALO(II) .LT. 0. .AND. AIMAGO(II) .LT. 0.) THEN
*     RATIOO=ABS(AIMAGO(II)/REALO(II))
*     PHASEO(INDEX)= 180. + SCAL*ATAN(RATIOO)
*   ELSEIF(REALO(II) .GT. 0. .AND. AIMAGO(II) .LT. 0.) THEN
*     RATIOO=ABS(AIMAGO(II)/REALO(II))
*     PHASEO(INDEX)= 360. - SCAL*ATAN(RATIOO)
*
*   ENDIF
*
*   Create dB and phase array of transfer function
*
*   DB_TF(INDEX)=DBO(INDEX)-DBI(INDEX)
*   PHASE_TF(INDEX)=PHASEO(INDEX)-PHASEI(INDEX)
*
*   INDEX=INDEX+1
*
*   ENDIF
*
8945   CONTINUE
*
*   WHICH DELTA F INTERPLOATE
*
*   WRITE(5,4659)FREQ_TF(1)
4659   FORMAT('////////',
+         'ENTER A DELTA-F(In Hz) FOR INTERPOLATION?',
+         'Choose smaller or equal to ',F10.4)
*   READ(5,*)STEP

```

```

*
* ---- Interpolate data
* Interpolate the frequency response data, more points will be generated
* for a constant step (Delta Hz). This is done for plotting purposes used
* later.
*
      F_END=FREQ_TF(INDEX-1)
      NSAMP=INDEX-1
      CALL INTEG(1,NSAMP,STEP,FREQ_TF,DB_TF,PHASE_TF,DATA)
*
      NSAMP=NSAMP-1 ! TRYING TO ELIMINATE ONE MYSTERIOUS BULLSHIT POINT
***
FORM TOTAL OPEN LOOP
*
      Add primary and Compensation
*
      DO 44333 J=1,NSAMP
        DATA(J,5)=DATA(J,1)+DATA(J,3)
        DATA(J,6)=DATA(J,2)+DATA(J,4)
44333      CONTINUE
*
*
      ENDIF ! OPTION IMENU1=1
*****
*      OPTION 2 ENTER DATA POINTS
*****
*      This option is available when only a few frequency data points are
*      known. (Usually from test data or extracted from a plot.) This option
*      will interpolate points in between the given ones. Be sure to include
*      enough points to describe the response. The more drastic the changes the
*      more points required.
*
      IF(IMENU1.EQ. 2)THEN
        WRITE(5,3000)
3000      FORMAT(' Enter how many points to enter by hand',
+          ' (This contains each Frequency, Gain and Phase as',
+          ' one point)')
        READ(5,*)NPTS
        DO 3002 J=1,NPTS
          WRITE(5,3003)J
3003      FORMAT(' POINT ',I2,' Enter FREQ(Hz),MAG(dB),PHASE(Deg)')
          READ(5,*)FREQ_TF(J),DB_TF(J),PHASE_TF(J)
3002      CONTINUE
*
        WRITE(5,4669)FREQ_TF(1)
4669      FORMAT('////////',
+          ' ENTER A DELTA-F(In Hz) FOR INTERPOLATION?',
+          ' Choose smaller or equal to ',F10.4)
        READ(5,*)STEP
*
        F_END=FREQ_TF(INDEX)
*
        CALL INTEG(1,J,STEP,FREQ_TF,DB_TF,PHASE_TF,DATA)
*
***
FORM TOTAL OPEN LOOP
*
      DO 99333 J=1,NSAMP
        DATA(J,5)=DATA(J,1)+DATA(J,3)
        DATA(J,6)=DATA(J,2)+DATA(J,4)
99333      CONTINUE
*
      ENDIF ! OPTION 2 (POINTS BY HAND)

```

```

*****
**** OPTIION 3 ENTER PRIME FREQ BY POLYNOMIALS *****
* This option gives the frequency response for a transfer function
* model described by polynomials in the S (Laplace) domain.
*****
      IF(IMENU1.EQ. 3)THEN
        CALL ENTER_NPOLYS(N_MAT,NODR,NPLYS,N_DEG) ! ASK NUMERATOR
        CALL ENTER_DPOLYS(D_MAT,DODR,DPLYS,D_DEG) ! ASK DENOMINATOR
        CALL MULTIPLY_POLYS(N_DEG,NODR,NPLYS,N_MAT,POLY_N) ! MULTIPLY NUM
        CALL MULTIPLY_POLYS(D_DEG,DODR,DPLYS,D_MAT,POLY_D) ! MULTIPLY DEN
*
      WRITE(5,4689)
4689  FORMAT('/////////
+      ' ENTER A DELTA-F(In Hz) FOR INTERPOLATION?'/,
+      ' (.1 Generally Used)')
*      READ(5,*)STEP
*
      WRITE(5,5689)
5689  FORMAT('/////////
+      ' ENTER THE END FREQUENCY (SPAN) In Hz ?'/,
+      ' (100. Generally Used)')
*      READ(5,*)F_END
*
*      ----- Poly Response
*      Determine the frequency response for the Numerator and Denominator
*      Product Polynomial (Poly N & Poly D respectively). Find response for
*      frequency range STEP to F_END by STEP. The channels for the Primary
*      response are 1 & 2 for Gain and Phase respectively.
*
*      CALL POLY_RESPONSE(N_DEG,POLY_N,D_DEG,POLY_D,STEP,
+      F_END,1,2,DATA,NSAMP)
*
***      ----- FORM TOTAL OPEN LOOP
*      By adding the Primary and Compensation response. (Cascade)
*
*      DO 7333 J=1,NSAMP
*          DATA(J,5)=DATA(J,1)+DATA(J,3)
*          DATA(J,6)=DATA(J,2)+DATA(J,4)
7333  CONTINUE
*
*      ENDIF 1 OF OPTION 3
*****
*      2ND MENU
*****
901  WRITE(5,900)
900  FORMAT('/////////
+      '***** SECOND MENU *****',/,
+      ' (1) PLOT frequency response',/,
+      ' (2) Enter COMPENSATION by polynomials',/,
+      ' (3) CHECK OR EDIT polynomials',/,
+      ' (4) Determine CROSSEOVERS for total response',/,
+      ' (5) Enter NEW PRIME FREQUENCY RESPONSE',/,
+      ' (6) STOP / QUIT',/,
+      ' Select 1,2,3,4,5 or 6')
*      READ(5,*)IMENU2
*****
*      OPTION 1 (2ND MENU) PLOT FREQUENCY RESPONSE
*****
      IF(IMENU2.EQ. 1)THEN

```

```

      CALL TF_PLOT(ITERM,NOTE,NSAMP,STEP,NCHAN,DATA,LONG_NAME,
+      UNIT_NAME)
      GOTO 901
    ENDIF 1 IMENU2 1ST OPTION
*****
*   OPTION 2 (2ND MENU)      ENTER COMPENSATION POLYS      *
*****
*
*   Enter compensation by means of transfer function polynomials
*   in S (Laplace) domain, same technique used above for option 3
*   of entering primary response.
*
    IF(IMENU2 .EQ. 2)THEN
      CALL ENTER_NPOLYS(C_N_MAT,C_NODR,C_NPLYS,C_N_DEG)
      CALL ENTER_DPOLYS(C_D_MAT,C_DODR,C_DPLYS,C_D_DEG)
      CALL MULTIPLY_POLYS(C_N_DEG,C_NODR,C_NPLYS,C_N_MAT,C_POLY_N)
      CALL MULTIPLY_POLYS(C_D_DEG,C_DODR,C_DPLYS,C_D_MAT,C_POLY_D)
      CALL POLY_RESPONSE(C_N_DEG,C_POLY_N,C_D_DEG,C_POLY_D,
+      STEP,F_END,3,4,DATA,NSAMP)
*
***   ----- FORM TOTAL OPEN LOOP
*   by adding primary and compensation response (Cascade)
*
      DO 9333 J=1,NSAMP
        DATA(J,5)=DATA(J,1)+DATA(J,3)
        DATA(J,6)=DATA(J,2)+DATA(J,4)
9333    CONTINUE
*
      GOTO 901
    ENDIF 1 IMENU2 2ND OPTION
*****
*   OPTION 3 (2ND MENU)      EDIT/LIST POLYNOMIALS      *
*****
*
    IF(IMENU2 .EQ. 3)THEN
*
      IF(IMENU1 .EQ. 3)THEN
        WRITE(5,3335)
        FORMAT(//////////,
3335      +      ' CHECK/EDIT: ',//, '(1) PRIMARY TRANSFER FUNCTION',//,
+      +      ' (2) COMPENSATION TRANSFER FUNCTION',//,
+      +      ' ENTER 1 or 2')
        READ(5,*)ICOMP
        ELSE
          ICOMP=2
        ENDIF
*****
*
*   LIST PRIMARY POLYS
*
*****
    IF(ICOMP .EQ. 1)THEN
*
      --- List individual polynomials
*
      WRITE(5,5550)
      FORMAT(//////////,
5550      +      '***** PRIMARY NUMERATOR POLYS *****',//)
      CALL LIST_POLY(1,N_MAT,POLY_N,NODR,NPLYS,N_DEG)
*

```

```

      WRITE(5,5551)
      FORMAT(//////////,
      5551 + '***** PRIMARY DENOMINATOR POLYS *****',//)
      CALL LIST_POLY(1,D_MAT,POLY_D,DODR,DPLYS,D_DEG)
*
* LIST PRODUCT POLYS, unless the same as above
*
      IF(NPLYS .GT. 1)THEN
      WRITE(5,4441)
      4441 + FORMAT(//////////,
      + '***** PRIMARY PRODUCT NUMERATOR POLYNMIAL *',
      + '*****',//)
      CALL LIST_POLY(2,N_MAT,POLY_N,NODR,1,N_DEG)
      ENDIF
      IF(DPLYS .GT. 1)THEN
      WRITE(5,4442)
      4442 + FORMAT(//////////,
      + '***** PRIMARY PRODUCT DENOMINATOR POLYNOMIAL',
      + '*****',//)
      CALL LIST_POLY(2,D_MAT,POLY_D,DODR,1,D_DEG)
      ENDIF
*
      ELSEIF(ICOMP .EQ. 2)THEN
      *****
      * LIST COMPENSATION POLYS *
      *****
      WRITE(5,5530)
      5530 + FORMAT(//////////,
      + '***** COMPENSATION NUMERATOR POLYS *****',//)
      CALL LIST_POLY(1,C_N_MAT,C_POLY_N,
      + C_NODR,C_NPLYS,C_N_DEG)
      WRITE(5,5531)
      5531 + FORMAT(//////////,
      + '***** COMPENSATION DENOMINATOR POLYS *****',//)
      CALL LIST_POLY(1,C_D_MAT,C_POLY_D,
      + C_DODR,C_DPLYS,C_D_DEG)
*
* LIST PRODUCT POLYS, unless the same as above
*
      IF(NPLYS .GT. 1)THEN
      WRITE(5,5534)
      5534 + FORMAT(//////////,
      + '***** COMPENSATION PRODUCT NUMERATOR *****',//)
      CALL LIST_POLY(2,C_N_MAT,C_POLY_N,C_NODR,1,C_N_DEG)
      ENDIF
      IF(DPLYS .GT. 1)THEN
      WRITE(5,5537)
      5537 + FORMAT(//////////,
      + '***** COMPENSATION PRODUCT DENOMINATOR ****',//)
      CALL LIST_POLY(2,C_D_MAT,C_POLY_D,C_DODR,1,C_D_DEG)
      ENDIF ! DPLYS .GT. 1
      *****
      *****
      *****
*
      ENDIF ! ICOMP 1 or 2
*
      *****
      ** EDIT POLYNOMIAL **
      *****

```

```

*
5555 WRITE(5,5555)
      FORMAT(//////////.
+      ' Do you want to make a change? (Y or N)')
77777 READ(5,77777)REPLY
      FORMAT(A)
*
      IF(REPLY .EQ. 'Y' .OR. REPLY .EQ. 'y')THEN
*
**      NUMERATOR OR DENOMINATOR *****
*
*.....
*
66666 WRITE(5,66666)
      FORMAT(//////////.
+      ' Changes in the NUMERATOR or DENOMINATOR ? (N or D)')
18    READ(5,18)NORD
      FORMAT(A)
*
      IF(ICOMP .EQ. 1)THEN
        IF(NORD .EQ. 'N' .OR. NORD .EQ. 'n')THEN
*****
* CHANGE PRIME POLY & RESPONSE *
*****
*----- CHANGE PRIME NUMERATOR -----
          CALL EDIT_POLY(1,N_MAT,NODR,NPLYS,N_DEG)
          ELSEIF(NORD .EQ. 'D' .OR. NORD .EQ. 'd')THEN
*----- CHANGE PRIME DENOMINATOR -----
          CALL EDIT_POLY(2,D_MAT,DODR,DPLYS,D_DEG)
          ENDIF
*
* RECREATE FREQUENCY RESPONSE FOR PRIME AND TOTAL
*
          CALL MULTIPLY_POLYS(N_DEG,NODR,NPLYS,N_MAT,POLY_N)
          CALL MULTIPLY_POLYS(D_DEG,DODR,DPLYS,D_MAT,POLY_D)
          CALL POLY_RESPONSE(N_DEG,POLY_N,D_DEG,POLY_D,STEP,
+          F_END,1,2,DATA,NSAMP)
          DO 3333 J=1,NSAMP
            DATA(J,5)=DATA(J,1)+DATA(J,3)
            DATA(J,6)=DATA(J,2)+DATA(J,4)
3333    CONTINUE
*****
*
        ELSE ! ICOMP MUST EQ 2
*****
* CHANGE COMPENSATION POLY & RESPONSE *
*****
*----- CHANGE COMPENSATION NUMERATOR -----
          IF(NORD .EQ. 'N' .OR. NORD .EQ. 'n')THEN
            CALL EDIT_POLY(1,C_N_MAT,C_NODR,C_NPLYS,C_N_DEG)
*----- CHANGE COMPENSATION DENOMINATOR -----
          ELSEIF(NORD .EQ. 'D' .OR. NORD .EQ. 'd')THEN
            CALL EDIT_POLY(2,C_D_MAT,C_DODR,C_DPLYS,C_D_DEG)
          ENDIF
*
* RECREATE FREQUENCY RESPONSE FOR COMPENSATION AND TOTAL
*
          CALL MULTIPLY_POLYS(C_N_DEG,C_NODR,C_NPLYS,
+          C_N_MAT,C_POLY_N)

```

```

*
*      CALL MULTIPLY_POLYS(C_D DEG,C DODR,C DPLYS,
*      +                  C_D_MAT,C_POLY_D)
*
*      CALL POLY_RESPONSE(C N DEG,C POLY N,C D DEG,
*      +                  C_POLY_D,STEP,F_END,3,4,DATA,NSAMP)
*
*      ---- CREATE TOTAL OPEN LOOP
*      by adding primary and compensation response (Cascade)
*
*      DO 45333 J=1,NSAMP
*          DATA(J,5)=DATA(J,1)+DATA(J,3)
*          DATA(J,6)=DATA(J,2)+DATA(J,4)
45333      CONTINUE
*****
*.....
*      ENDIF
*
*      ENDIF      ! MAKE CHANGES
*
*      GOTO 901
*
*      ENDIF      ! END OF LIST & EDIT POLYNOMIAL
*
*****
*      OPTION 4 (2ND MENU)      DETERMINE CROSS OVERS FOR CASCADE RESPONSE *
*****
*      IF(IMENU2.EQ. 4)THEN
*          CALL CROSS_OVER(5,NSAMP,STEP,DATA)
*          GOTO 901
*      ENDIF
*****
*      OPTION 5 (2ND MENU)      RETURN TO MAIN MENU
*****
*      IF(IMENU2.EQ. 5)THEN
*          GO TO 5566
*      ENDIF
*****
*
*      STOP
*      END      !      END OF MAIN PROGRAM 'BODE'
*
-----
*****
*****
*****      SUBROUTINE SECTION      *****
*****
-----
*****
*****
*****      SUBROUTINE READFREQ(GFREQ,GLEVEL,NTABLE)
*
*      Written by A. L. Helinski  US TACOM  AMSTA-RV
*
*      This subroutine reads the frequency table used by the time domain
*      simulation (ACSL) to create the input signal to excite the transfer
*      function.
*
*      INPUT:  None (File Table Name)
*      OUTPUT: GFREQ(Index)  Table Frequencies (Hz) (Frequency of each sin wave)

```

```

*           GLEVEL           Amplitude of each sine wave (1 value for all)
*           NTABLE           Number of Sin Waves (Frequencies)
*

```

```

REAL GFREQ(50)
CHARACTER*30 FILEN
*
WRITE(5,100)
100  FORMAT('Enter The File with the frequency table?',/,
+         'Example: freqs.dat')
READ(5,140)FILEN
140  FORMAT(A30)
OPEN(10,FILE=FILEN,FORM='FORMATTED',SHARED,
+ STATUS='OLD',ERR=210)
READ(10,400)GLEVEL
400  FORMAT(/,G10.4)
NTABLE=1
800  READ(10,600)GFREQ(NTABLE)
600  FORMAT(G10.4)
IF(GFREQ(NTABLE) .LT. 0.)GOTO 300
NTABLE=NTABLE+1
GOTO 800
210  WRITE(5,170)
170  FORMAT(' ERROR OPENING FILE')
300  CLOSE(10)
RETURN
END

```

```

SUBROUTINE GENERATE(SINPUT,GFREQ,GLEVEL,NTABLE,T)
*
*   Written by A. L. Helinski  US TACOM  AMSTA-RY
*
*   This subroutine creates the sweep wave (Sum of sine waves)
*   for creating the input signal to a frequency response
*   analysis to determine transfer function characteristics.
*
*   INPUT:  GFREQ  Table of Frequencies
*           GLEVEL  Amplitude for all
*           NTABLE  Number of Sine waves (Frequencies)
*   OUTPUT: SINPUT  Sum of sine wave signal (Signal Generated)
*           T       Corresponding Time (Seconds)
*

```

```

*
*   DIMENSION GFREQ(50)
*   SINPUT=0.
*   PI=3.141592654
*   DO 100 II=1,NTABLE
*       SINPUT=SINPUT + GLEVEL*SIN(2*PI*T*GFREQ(II))
100  CONTINUE
RETURN
END

```

```

*****
*****
*****
*****

```

```

SUBROUTINE INTEP(ICHAN,NSAMP,DELTA_F,FREQ,DB,PHASE,DATA)

```

```

*
*   Written by A. L. Helinski  US TACOM  AMSTA-RY
*
***** INTERPOLATION *****
*   This subroutine interpolates points from the frequency response results

```

```

* (Gain and Phase) to re-establish data with a constant step
* (Delta Freq. Hz) so that the further processes will be adaptable.
* In other words this program basically converts a set of raw data
* of varied steps (Varied Sampling Rates) to a set of data with constant
* step. If you think that sounds hairy, wait until you see the data!
* Warning must be a sufficient number of points to begin with, enter
* at your own risk. The more drastic the changes in the data the
* more points will be required.
*****
* INPUT; ICHAN      Index for channel      ICHAN will be gain
*                                     and ICHAN+1 will be phase
*      NSAMP        Number of samples
*      DELTA_F      Desired Step frequency (Hz)
*      FREQ         Actual frequency of points for Phase & DB
*                   before interpolation. (Un-interpolated points)
*      DB           dB values of un-interpolated points
*      PHASE        Phase (Deg) values of un-interpolated points
*
* OUTPUT; NSAMP      New number of samples for interpolated
*                   results (Changed from input)
*      DATA(ISAMP,ICH) New interpolated data with constant step
*
* Generate slopes and intercepts from the frequency selected
* points.
*
* DIMENSION PHASE(50),DB(50),FREQ(50)
* DIMENSION DB_SL(50),DB_IT(50)
* DIMENSION PHASE_SL(50),PHASE_IT(50)
* REAL*4 DATA(10000,6)
* INTEGER*4 NSAMP
*
* DO 1999 J=1,NSAMP-1
*   PHASE_SL(J) =
* + (PHASE(J+1)-PHASE(J))/(FREQ(J+1)-FREQ(J))
*   PHASE_IT(J) =
* + PHASE(J)-PHASE_SL(J)*FREQ(J)
*   DB_SL(J) =
* + (DB(J+1)-DB(J))/(FREQ(J+1)-FREQ(J))
*   DB_IT(J) =
* + DB(J)-DB_SL(J)*FREQ(J)
* 1999 CONTINUE
*   FREQFIN=FREQ(NSAMP)
*
* DETERMINE NEW INTERPLOATED DATA
*
*   NSAMP=INT(FREQFIN/DELTA_F-1)
*   STEP=DELTA_F
*
*   J=1
*   DO 7510 K=1,NSAMP
*     FREQN=DELTA_F*(K-1)
*     FREQO=FREQ(J)
*     IF(FREQN.GT. FREQO)THEN
*       J=J+1
*     ENDIF
*     DATA(K,ICHAN)=DB_SL(J)*FREQN+DB_IT(J)
*     DATA(K,ICHAN+1)=PHASE_SL(J)*FREQN+PHASE_IT(J)
* 7510 CONTINUE
*   RETURN
*   END
*****

```

```

*****
*****
SUBROUTINE TEK_DELAY
*
*   Written by AL Reid   US TACOM AMSTA-RV   for plotting routine
*
*   This subroutine will delay 2 seconds to allow the tektronix screen ample
*   time to clear itself.
*
*   INPUT;  NONE
*   OUTPUT; NONE
*
*   TNOW = SECNDS(0.0)
10  DELTA = SECNDS(TNOW)
*   IF (DELTA .LT. 2.0) GOTO 10
*
*   RETURN
*   END
*
*
SUBROUTINE CROSS_OVER(ICH,NSAMP,DELTA_F,DATA)
*****
***** DETERMINE ALL CROSSOVER POINTS *****
*****
*
*   Written by A. L. Helinski  US TACOM  AMSTA-RV
*
*   This subroutine determines the crossovers of a frequency
*   response data for stability checks for the case when the response
*   is a total open loop. The process simply scans the DATA array
*   for magnitude (DB) and Phase and detects any crossover points of
*   0 dB or +/- 180 Deg respectively.
*
*   INPUT;  ICH           Index channel as
*                       ICH would be Gain
*                       then ICH1 would be Phase
*
*           NSAMP          Number of samples
*           DELTA_F        Step frequency in Hz
*           DATA(SAMP,Channel Number)  Data as Gain or Phase
*
*   OUTPUT; NONE (Results are printed on screen)
*
*   INTERNAL;  SIGN_DB,SIGN_PH  Change of sign indicators
*             JCRP,JCRG        Index for crossovers
*
*   INTEGER*4 NSAMP
*   DIMENSION SIGN_DB(10000),SIGN_PH(10000)
*   DIMENSION JCRP(10000),JCRG(10000)
*   REAL*4 DATA(10000,6)
*
*   DETERMINE ALL POINTS AROUND 0 dB
*
*   DO 8199 J=1,NSAMP
*
*   DETERMINE ALL POINTS AROUND 0 dB
*
*   IF(DATA(J,ICH) .EQ. 0.)THEN
*     SIGN_DB(J)=0.
*   ELSE

```

```

SIGN_DB(J)=DATA(J,ICH)/ABS(DATA(J,ICH))
ENDIF
*
*
*
* DETERMINE ALL POINTS AROUND -180 DEG
*
IF(DATA(J,ICH+1) .EQ. -180.)THEN
SIGN_PH(J)=0.
ELSE
IF(DATA(J,ICH+1) .LT. 0.)THEN
SIGN_PH(J)=(-180.-DATA(J,ICH+1))/ABS(-180.-DATA(J,ICH+1))
ELSEIF(DATA(J,ICH+1) .GT. 0.)THEN
SIGN_PH(J)=(-180.+DATA(J,ICH+1))/ABS(-180.+DATA(J,ICH+1))
ENDIF
ENDIF
*
8199 CONTINUE
*
* DETERMINE SIGN CHANGES
*
ICG=1.
ICP=1.
DO 8198 J=1,NSAMP-1
IF(SIGN_DB(J+1) .NE. SIGN_DB(J))THEN
JCRG(ICG)=J+1
ICG=ICG+1
ENDIF
IF(SIGN_PH(J+1) .NE. SIGN_PH(J))THEN
JCRP(ICP)=J+1
ICP=ICP+1
ENDIF
8198 CONTINUE
*
*
WRITE(5,2005)
2005 + FORMAT(23X,'***** GAIN CROSSOVERS ***** ',/,
10X,' GAIN(dB)',13X,' PHASE(Deg)',13X,' FREQ(Hz)')
IF(ICG .EQ. 1)THEN
WRITE(5,9373)
9373 + FORMAT('** NONE **')
ELSE
DO 1666 J=1,ICG-1
FHZH=(JCRG(J))*DELTA_F
FHZL=(JCRG(J)-1)*DELTA_F
WRITE(5,4502)DATA(JCRG(J)-1,ICH),DATA(JCRG(J)-1,ICH+1),FHZL
+ WRITE(5,4502)DATA(JCRG(J),ICH),DATA(JCRG(J),ICH+1),FHZH
4502 + FORMAT(6X,F10.4,13X,F10.4,13X,F10.4)
WRITE(5,4903)
4903 + FORMAT(/)
1666 CONTINUE
ENDIF
*
WRITE(5,447)
READ(5,*)
*
WRITE(5,1667)
1667 + FORMAT(/,23X,'***** PHASE +/- 180DEG CROSSOVERS ***** ',
/,10X,' PHASE(DEG)',13X,' GAIN(DB)',13X,' FREQ(Hz)')
IF(ICP .EQ. 1)THEN

```

```

9365      WRITE(5,9365)
          FORMAT(/,' ** NONE ** ')
        ELSE
          DO 1234 J=1,ICP-1
            FHZH=(JCRP(J))*DELTA_F
            FHZL=(JCRP(J)-1)*DELTA_F
            WRITE(5,4502)DATA(JCRP(J)-1,ICH+1),DATA(JCRP(J)-1,ICH),FHZL
            WRITE(5,4502)DATA(JCRP(J),ICH+1),DATA(JCRP(J),ICH),FHZH
            WRITE(5,4903)
          CONTINUE
1234      ENDOIF
        *
        WRITE(5,447)
        447      FORMAT(///,' HIT RETURN TO CONTINUE')
        READ(5,*)
        *
        RETURN
        END
*****
*****
SUBROUTINE ENTER_NPOLYS(N_MAT,NODR,NPLYS,N_ORDER)
*
*   Written by A. L. Helinski US TACOM AMSTA-RV
*
*   This subroutine will let you enter the polynomials representing
*   the numerator of a transfer function. It's no different then
*   Subroutine ENTER_DPOLYS.
*
*   INPUT:  NONE (Enter all output parameters by hand)
*   OUTPUT: N_MAT(POLY#,ORDER#) Polynomial Coefficients
*           NODR(POLY#)          Order of each polynomial
*           NPLYS                Number of Polynomial
*           N_Order              Total Order (Degree of product)
*                               Sum of all DODR(POLY#)
*
      REAL N_MAT(20,20)      I NUM. MATRIX COEFFICIENTS(POLY#,ORDER)
      INTEGER NODR(20),NPLYS I NUM. ORDER (POLY#) , NUM. # OF POLYS
*
      WRITE(5,10)
      10      FORMAT(//////////,
+             ' Enter number of polynomials in the numerator?')
      READ(5,*)NPLYS
      WRITE(5,20)
      20      FORMAT(//)
      N_ORDER=0
      DO 90 I=1,NPLYS
        WRITE(5,30)I
        30      FORMAT(' Enter order of poly #',I2)
        READ(5,*)NODR(I)
        N_ORDER=N_ORDER+NODR(I) I ADD TOTAL ORDERS (DEGREE)
        WRITE(5,40)
        40      FORMAT(/)
        DO 50 J=NODR(I)+1,1,-1
          WRITE(5,60)J-1
          60      FORMAT(' S**',I2,' TERM IS ? ',:)
          READ(5,*) N_MAT(I,J) I NUMERATOR MATRIX (POLY#,ORDER+1)
        50      CONTINUE
        WRITE(5,70)
        70      FORMAT(/)
      90      CONTINUE

```

```

*
      RETURN
      END
*****
*****
      SUBROUTINE ENTER_DPOLYS(D_MAT,DODR,DPLYS,D_ORDER)
*
*      Written by A. L. Helinski US TACOM AMSTA-RY
*
*      This subroutine will let you enter the polynomials representing
*      the denominator of a transfer function. It's no different then
*      Subroutine ENTER_NPOLYS.
*
*      INPUT:  NONE (Enter all output parameters by hand)
*      OUPUT:  D_MAT(POLY#,ORDER#) Polynomial Coefficients
*              DODR(POLY#)         Order of each polynomial
*              DPLYS                Number of Polynomial
*              D_Order              Total Order (Degree of product)
*                                  Sum of all DODR(POLY#)
*
      REAL D_MAT(20,20)
      INTEGER DODR(20),DPLYS,D_ORDER
*
      WRITE(5,10)
10      FORMAT('//////////',)
      +      'Enter number of polynomials in the denominator?')
      READ(5,*)DPLYS
      WRITE(5,20)
20      FORMAT('//')
      D_ORDER=0
      DO 90 I=1,DPLYS
      WRITE(5,30)I
30      FORMAT('Enter order of poly # ',I2)
      READ(5,*)DODR(I)
      D_ORDER=D_ORDER+DODR(I)
      WRITE(5,40)
40      FORMAT('/')
      DO 50 J=DODR(I)+1,1,-1
      WRITE(5,60)J-1
60      FORMAT('S**',I2,' TERM IS ?',:)
      READ(5,*) D_MAT(I,J)  ! NUMERATOR MATRIX (POLY#,ORDER+1)
50      CONTINUE
      WRITE(5,70)
70      FORMAT('/')
90      CONTINUE
*
      RETURN
      END
*****
*****
      SUBROUTINE MULTIPLY_POLYS
      (TOTAL_ORDER,POLY_ORDER,NPOLYS,MATRIX,PRODUCT)
*
*      +
*      Written by A. L. Helinski US TACOM AMSTA-RY
*
*      This subroutine multiplies polynomials for resulting product of
*      a single resulting polynomial. The procedure is much the same way
*      as one would do by hand. (Term by term)
*
*      INPUT:  TOTAL_ORDER      Total degree of product was done before hand
*                                  by adding the degree of each polynomial

```

```

*          POLY ORDER          Degree of each individual polynomial
*          NPOLYS              Number polynomials
*          MATRIX(POLY#,Order#) Matrix of Polynomial coefficients
*                               (Simply a 2-D array)
*
* OUTPUT:  PRODUCT(Order #)    Final product polynomial
*
*          INTEGER TOTAL ORDER  ! ORDER OF PRODUCT (OR TOTAL SUM degrees)
*          INTEGER POLY ORDER(20) ! ORDER OF EACH POLY
*          INTEGER NPOLYS       ! NUMBER OF POLYS
*          REAL MATRIX(20,20)   ! COEFFICIENTS OF (POLY#,ORDER)
*          REAL SM(50)          ! USED AS SUMMER DURING OPERATION
*          REAL PRODUCT(40)     ! PRODUCT RESULTS
*
*          PRODUCT must have a initial value. In this program the
*          initial product will be 1.
*
*          PRODUCT(1)=1.      ! INITIAL PRODUCT WILL BE 1.
*
*          LEVL=0
*          DO 10 M=1,TOTAL_ORDER+1
*             SM(M)=0.        ! SUMMER USED IN MULIPLICATION
10          CONTINUE
*
*          DO 20 I=1,NPOLYS
*             DO 30 K=1,LEVL+1
*                DO 40 L=1,POLY_ORDER(I)+1
*                   SM(K+L-1)=SM(K+L-1) + MATRIX(I,L) * PRODUCT(K)
40          CONTINUE
30          CONTINUE
*          DO 50 M=1,TOTAL_ORDER+1
*             PRODUCT(M)=SM(M)
*             SM(M)=0.
50          CONTINUE
*          LEVL=LEVL+POLY_ORDER(I)
20          CONTINUE
*
*          RETURN
*          END
*****
*****
*****
SUBROUTINE POLY_RESPONSE(N DEG,POLY N,D DEG,POLY_D,
+ F_STEP,F_END,NCHAN_MAG,NCHAN_PHASE,DATA,NPTS)
*
*          Written by A. L. Helinski US TACOM AMSTA-RY
*
*          Determine frequency response for a single polynomial describing the
*          numerator and one describing the denominator of a transfer function for
*          a given frequency range.
*
*          INPUT;  N DEG          Degree of numerator
*                  POLY N(Order#) Coefficients describing numerator poly
*                  D DEG          Degree of Denominator
*                  POLY D(Order#) Coefficients describing denominator poly
*                  F_STEP         Desired delta frequency of range
*                               Also first frequency point to evaluate
*                  F_END          End frequency of range (Span)
*                  NCHAN_MAG      Desired channel number for Magnitude (dB)
*                  NCHAN_PHASE    Desired channel number for Phase (Deg)

```

```

*
*
*      OUTPUT;  NPTS      Number of frequency pts made
*               DATA(NSAMP,NCHAN)  DATA made
*               where Gain (dB) is DATA(NSAMP,NCHAN_MAG)
*               Phase (Deg) is DATA(NSAMP,NCHAN_PHASE)
*               @ frequencies = NSAMP*F_STEP Hz
*
REAL DREL,DIMG,NREL,NIMG
REAL PHASEN,PHASED
REAL*4 DATA(10000,6)
REAL POLY_D(40),POLY_N(40)
INTEGER D_DEG,N_DEG
*
PI=3.141592654
SCAL=180./PI
EPSILON=10.E-30
*
NPTS=INT(F_END/F_STEP)
*
DETERMINE GAIN AND PHASE FOR NUMERATOR AND DENOMINATOR
*
DO 90 JJ=1,NPTS
  FR=F_STEP*JJ
  WR=FR*2.*PI
* Determine Real and Imaginary components for NUM and DEN
  CALL POLY_REAL_IMAG(NREL,NIMG,WR,N_DEG,POLY_N)
  CALL POLY_REAL_IMAG(DREL,DIMG,WR,D_DEG,POLY_D)
*
* Determine dB Gain
*
  GAIN=SQRT( (NREL**2+NIMG**2) / (DREL**2+DIMG**2) )
*
  IF(GAIN .LE. EPSILON)THEN
    DATA(JJ,NCHAN_MAG) = -580.
  ELSE
    DATA(JJ,NCHAN_MAG) = 20.* ALOG10(GAIN)
  ENDIF
*
* Determine PHASE for NUM and DEN
*
* Numerator
*
  IF(NREL .EQ. 0. .AND. NIMG .LT. 0.)THEN
    PHASEN = -90.
  ELSEIF(NREL .EQ. 0. .AND. NIMG .GT. 0.)THEN
    PHASEN = 90.
  ELSEIF(NIMG .EQ. 0. .AND. NREL .GT. 0.)THEN
    PHASEN = 0.
  ELSEIF(NIMG .EQ. 0. .AND. NREL .LT. 0.)THEN
    PHASEN = 180.
*
  ELSEIF(NREL .GT. 0. .AND. NIMG .GT. 0.)THEN
    RATION=ABS(NIMG/NREL)
    PHASEN= SCAL*ATAN(RATION)
  ELSEIF(NREL .LT. 0. .AND. NIMG .GT. 0.)THEN
    RATION=ABS(NIMG/NREL)
    PHASEN=180. - SCAL*ATAN(RATION)
  ELSEIF(NREL .LT. 0. .AND. NIMG .LT. 0.)THEN
    RATION=ABS(NIMG/NREL)
    PHASEN=180. + SCAL*ATAN(RATION)
  ELSEIF(NREL .GT. 0. .AND. NIMG .LT. 0.)THEN

```

```

      RATION=ABS(NIMG/NREL)
      PHASEN=360. - SCAL*ATAN(RATION)
*
*   ENDIF
*
*   DENOMINATOR PHASE
*
      IF(DREL .EQ. 0. .AND. DIMG .LT. 0.)THEN
        PHASED = -90.
      ELSEIF(DREL .EQ. 0. .AND. DIMG .GT. 0.)THEN
        PHASED = 90.
      ELSEIF(DIMG .EQ. 0. .AND. DREL .GT. 0.)THEN
        PHASED = 0.
      ELSEIF(DIMG .EQ. 0. .AND. DREL .LT. 0.)THEN
        PHASED = 180.
*
      ELSEIF(DREL .GT. 0. .AND. DIMG .GT. 0.)THEN
        RATIOD=ABS(DIMG/DREL)
        PHASED=SCAL*ATAN(RATIOD)
      ELSEIF(DREL .LT. 0. .AND. DIMG .GT. 0.)THEN
        RATIOD=ABS(DIMG/DREL)
        PHASED=180. - SCAL*ATAN(RATIOD)
      ELSEIF(DREL .LT. 0. .AND. DIMG .LT. 0.)THEN
        RATIOD=ABS(DIMG/DREL)
        PHASED=180. + SCAL*ATAN(RATIOD)
      ELSEIF(DREL .GT. 0. .AND. DIMG .LT. 0.)THEN
        RATIOD=ABS(DIMG/DREL)
        PHASED=360. - SCAL*ATAN(RATIOD)
*
*   ENDIF
*
*   Determine Total Phase
*
      DATA(JJ,NCHAN_PHASE)=PHASEN-PHASED
*
90    CONTINUE
*
      RETURN
      END
*****
*****
*****
      SUBROUTINE POLY_REAL_IMAG(XREAL,XIMAG,OMEG,NORDER,POLY)
*
*   Written by A. L. Helinski  US TACOM  AMSTA-RY
*
*   DETERMINE REAL AND IMAGINARY COMPONENTS FOR A
*   GIVEN FREQUENCY(OMEG) AND POLYNOMIAL
*
*   This subroutine called upon by sub-poly-response to determine the
*   Real and Imaginary components for a given polynomial and frequency(W).
*   It simply plugs in S=jW and resolves the complex algebra. ( W omega in
*   rad/sec and j is complex where j=SQRT [ -1 ]. )
*
*   INPUT:  POLY      (Array of POLY coefficients)
*           NORDER    Order of polynomial
*           OMEG       Frequency in Rad/Sec
*   OUPUT:  XREAL     Real Component
*           XIMAG     Imaginary Component

```

```

*
*      REAL POLY(40)
*
*      XREAL=0.
*      XIMAG=0.
*
*      DO 10 II=1,NORDER+1
*
*          IORD=II-1
*
*          INDEX=IORD
20      IF(INDEX .GE. 4) THEN
*          INDEX=INDEX-4
*          GO TO 20
*      ELSE
*
*          IF(INDEX .EQ. 0) THEN
*              XREAL=XREAL + POLY(IORD+1)*OMEG**IORD
*          ELSEIF(INDEX .EQ. 1) THEN
*              XIMAG=XIMAG + POLY(IORD+1)*OMEG**IORD
*          ELSEIF(INDEX .EQ. 2) THEN
*              XREAL=XREAL - POLY(IORD+1)*OMEG**IORD
*          ELSEIF(INDEX .EQ. 3) THEN
*              XIMAG=XIMAG - POLY(IORD+1)*OMEG**IORD
*          ENDIF
*
*      ENDIF
*
10      CONTINUE
*
*      RETURN
*      END
*****
*****
*****
+      SUBROUTINE LIST_POLY(ID,POLY_MATRIX,PRODUCT_POLY,
*      ORDER_MATRIX,NUM_POLYS,DEGREE)
*
*      Written by A. L. Helinski   US TACOM   AMSTA-RV
*
*      This subroutine list the polynomial coefficients.
*
*      INPUT;      ID  Identify individual polynomials or single product
*                  If ID = 1  Number of individual Polynomials
*                  ID = 2  Single Product Polynomial
*      POLY_MATRIX(POLY#,Order #)
*                  Actually describes the coefficients in this
*                  2-D array fashion. (When ID=1 otherwise a dummy)
*
*      PRODUCT_POLY(Order #)
*                  Describes a single polynomial in coefficient terms.
*                  (When ID=2 otherwise a dummy variable.)
*
*      ORDER_MATRIX(POLY #)
*                  Describes the degree for each polynomial
*
*      NUM_POLYS  Number of polynomials (=1 for ID=2)
*
*      DEGREE  Total degree of polynomial (or sum of orders)
*
*      OUTPUT;      None (Just simply lists the polynomials on the screen)

```

```

REAL POLY_MATRIX(20,20),PRODUCT_POLY(40)
INTEGER ID,ORDER_MATRIX(20),NUM_POLYS,DEGREE,POWER
INTEGER NUM_POLS
*
IF(ID.EQ. 2)THEN
  NUM_POLS=1
ELSE
  NUM_POLS=NUM_POLYS
ENDIF
*
      WRITE(5,8888)
      FORMAT('          COEFFICIENT TERMS ',/)
*
      DO 10 IPOLY=1,NUM_POLS
      WRITE(5,20)IPOLY,ORDER_MATRIX(IPOLY)
      FORMAT(' POLY ',I2,5X,' ORDER =',I2,/,
*
      '-----')
*
      IF(ID.EQ. 1)THEN
        POWER=ORDER_MATRIX(IPOLY)
      ELSEIF(ID.EQ. 2)THEN
        POWER=DEGREE
      ENDIF
*
      DO 40 IORD=POWER+1,1,-1
*
      IF(ID.EQ. 1)THEN
        WRITE(5,70)POLY_MATRIX(IPOLY,IORD),IORD-1
        FORMAT(5X,F15.8,' S**',I2)
      ELSE
        WRITE(5,80)PRODUCT_POLY(IORD),IORD-1
        FORMAT(5X,F15.8,' S**',I2)
      ENDIF
*
      CONTINUE
*
      CONTINUE
*
      WRITE(5,66)
      FORMAT(//,' HIT RETURN TO CONTINUE')
      READ(5,*)
*
      RETURN
      END
*****
*****
SUBROUTINE EDIT_POLY(ID,MATRIX,ORDER,NUM_POLYS,DEG)
*
*   Written by A. L. Helinski  US TACOM  AMSTA-RV
*
*   This subroutine edits the polynomials by changing the
*   coefficient terms individually.
*
*   INPUT;  ID;      IF ID = 1  NUMERATOR TERMS
*              ID = 2  DENOMINATOR TERMS
*              MATRIX(POLY#,ORDER #);  2-D ARRAY OF POLYNOMIAL COEFFICIENTS
*              ORDER(POLY #);          DEGREE OF EACH POLY#
*              NUM_POLYS                NUMBER OF POLYNOMIALS
*              DEG;                     TOTAL DEGREE
*   OUTPUT; Same as input with modifications
*

```

```

      REAL MATRIX(20,20)
      INTEGER ID,NUM_POLYS,DEG,ORDER(20)
      CHARACTER*1 REPLY
*
      IF(ID.EQ. 1)THEN
        WRITE(5,10)
10      FORMAT(//'/')
        + ' Do you desire to re-enter the entire NUMERATOR ? (Y or N)')
        READ(5,20)REPLY
20      FORMAT(A)
        IF(REPLY.EQ. 'Y' .OR. REPLY.EQ. 'y')THEN
          CALL ENTER_NPOLYS(MATRIX,ORDER,NUM_POLYS,DEG)
          RETURN
*
        ENDIF
      ELSE
        WRITE(5,30)
30      FORMAT(//'/')
        + ' Do you desire to re-enter the entire DENOMINATOR ? (Y or N)')
        READ(5,20)REPLY
        IF(REPLY.EQ. 'Y' .OR. REPLY.EQ. 'y')THEN
          CALL ENTER_DPOLYS(MATRIX,ORDER,NUM_POLYS,DEG)
          RETURN
*
        ENDIF
      ENDIF
*
      Edit Polynomial by terms
*
      IF(NUM_POLYS.GT. 1)THEN
        WRITE(5,70)
70      FORMAT(//'/')
        + ' Which Polynomial do you desire to change ?',/,
        + ' ( Give the number representing the polynomial sequence )')
        READ(5,*)MPOLY
      ELSE
        MPOLY=1
      ENDIF
*
      IF(ORDER(MPOLY).GT. 1)THEN
        WRITE(5,80)MPOLY
80      FORMAT(//'/')
        + ' Which ORDER coefficient of S do you desire to ',/,
        + ' change in POLY # ',I2,' ?')
        READ(5,*)MCOEFF
      ELSE
        MCOEFF=1
      ENDIF
*
      WRITE(5,90)MATRIX(MPOLY,MCOEFF+1)
90      FORMAT(//'/')
        + ' Change ',F10.4,' TO ?')
        READ(5,*)MATRIX(MPOLY,MCOEFF+1)
*
      RETURN
      END
*****
*****
      SUBROUTINE READERD(TIME,SIGNAL,NSAMP,
        + CHANNEL,UNITS,NCH)
*

```

```

*      Written by AL Reid  US TACOM  AMSTA-RV
*      Modified by A. L. Helinski  US TACOM  AMSTA-RV
*
*      This subroutine originally written by Al Reid and modified by A. L.
*      Helinski.  It simply reads a data file of a time history in ERD file
*      format.  For this application ACSL creates the time data file
*      simulating the output of a transfer function this routine reads the
*      time history into the desired variables by calling it.
*
*      INPUT;      None (Data File Name)
*      OUTPUT;     SIGNAL      Amplitude of history
*                  TIME        Corresponding Time in Seconds
*
*      DIMENSION TIME(10000), SIGNAL(10000)
*      CHARACTER*80 ERD_TITLE, LONG_TITLE, DUMMY80
*      CHARACTER*64 ERD_FILE, HDR_FILE, ERD_FILE_0, HDR_FILE_0
*      CHARACTER*32 LONG_NAME(6), DUMMY32, CHANNEL
*      CHARACTER*12 DUMMY
*      CHARACTER*8 SHORT_NAME(6), UNIT_NAME(6), XUNIT, DUMMY8,
*      + UNITS
*      CHARACTER*4 ERD, HDR
*      CHARACTER*1 COMMA, REPLY
*      REAL*4 SCALE(6), OFFSET(6), RDATA(6, 10000)
*      INTEGER*4 NSAMP
*      INTEGER*4 START_SAMP_ELIM, END_SAMP_ELIM
*      INTEGER*2 ERD_UNIT, HDR_UNIT
*      DIMENSION IARRAY(400), IDATA(4)
*      DATA ERD_UNIT, HDR_UNIT / 10, 11 /
*      ERD = '.ERD'
*      HDR = '.HDR'
*
*      * Determine the name of the input data file
*
10  WRITE(5, 20)
20  FORMAT(//, ' Enter name of ERD data file to analyze?')
  READ(5, 30) ERD_FILE
30  FORMAT(A32)
  CALL STRTRIM(HDR_FILE, ERD_FILE, LENGTH)
  HDR_FILE(LENGTH+1:LENGTH+4) = HDR
  ERD_FILE(LENGTH+1:LENGTH+4) = ERD
*
*      * Open the data file, print the header characteristics, and determine if
*      * this is the correct data file
*
  OPEN(HDR_UNIT, FILE=HDR_FILE, FORM='FORMATTED',
  +     SHARED, STATUS='OLD', ERR=210)
*
*      * Read the header data
*
  READ(HDR_UNIT, 60) DUMMY
60  FORMAT(A12)
  READ(HDR_UNIT, 70) ERD_TITLE
70  FORMAT(A80)
  READ(HDR_UNIT, 80) NCHAN, COMMA, NSAMP, COMMA, NLINES, COMMA, NBIN,
  + COMMA, NBYTE, COMMA, KEYNUM, COMMA, STEP, COMMA, KEYOPT
80  FORMAT(6(I7, A), E13.6, A, I7)
  READ(HDR_UNIT, 90) SCALE(1), (COMMA, SCALE(L), L=2, NCHAN)
  READ(HDR_UNIT, 90) OFFSET(1), (COMMA, OFFSET(L), L=2, NCHAN)
  READ(HDR_UNIT, 100) (SHORT_NAME(L), L=1, NCHAN)
  READ(HDR_UNIT, 110) (LONG_NAME(L), L=1, NCHAN)
  READ(HDR_UNIT, 100) (UNIT_NAME(L), L=1, NCHAN)

```

```

90  FORMAT(18(E13.6,A))
100 FORMAT(31(A8))
110 FORMAT(7(A32))
*
* Write out the header information
*
  WRITE(5,120) ERD_TITLE,NCHAN,NSAMP,STEP
120  FORMAT(//,' The Title for this file is:',/,',',A80,/,', There are
+ ',12,' channels of data.',/,', There are ',17,' samples for each da
+ta channel.',/,', The step size is ',F8.5,' seconds.',//)
*
* See if there are more than 16 channels to plot
*
  IF (NCHAN .GT. 16) THEN
    TYPE*, ' There are more than 16 channels to read.'
    TYPE*, ' Please FORGET ABOUT IT'
    CLOSE (HDR_UNIT)
    STOP
  ENDIF
*
* Write out the additional descriptor lines
*
  IF (NLINES .GT. 0) THEN
    TYPE*, 'The following are the optional descriptor lines:'
    DO 130 L=1,NLINES
      READ(HDR_UNIT,70) LONG
      WRITE(5,125) LONG
125    FORMAT(' ',A80)
130    CONTINUE
  ENDIF
*
* Is this the correct data file
*
  WRITE(5,140)
140  FORMAT(//,'$Is this the correct data file to analyze (y or n)?
+ ')
  READ(5,150) REPLY
150  FORMAT(A)
  IF (REPLY .EQ. 'N' .OR. REPLY .EQ. 'n') THEN
    CLOSE(HDR_UNIT)
    WRITE(5,160)
160    FORMAT(/,'$Do you wish to look at another file (y or n)? ')
    READ(5,150) REPLY
    IF (REPLY .EQ. 'N' .OR. REPLY .EQ. 'n') STOP
    GOTO 10
  ENDIF
*
* Open the data part of the file
*
  IF (KEYNUM .EQ. 5) THEN
    OPEN(ERD_UNIT,FILE=ERD_FILE,STATUS='OLD'
+ ,SHARED,FORM='FORMATTED')
  ELSE
    OPEN(ERD_UNIT,FILE=ERD_FILE,STATUS='OLD'
+ ,SHARED,FORM='UNFORMATTED')
  ENDIF
  CLOSE(HDR_UNIT)
*
* Read the data
*
  J = 0

```

```

170   J=J+1
      IF (KEYNUM .EQ. 5) THEN
180         READ(ERD_UNIT,180,ERR=220,END=230) (RDATA(I,J),I=1,NCHAN)
          FORMAT(19(E13.6))
      ELSE
          IF (KEYNUM .EQ. 0) THEN
              READ(ERD_UNIT,ERR=220,END=230) (IDATA(I),I=1,NCHAN)
              DO 190 K=1,NCHAN
                  RDATA(K,J) = FLOATJ(IDATA(K))
190          CONTINUE
          ELSE
              READ(ERD_UNIT,ERR=220,END=230) (RDATA(I,J),I=1,NCHAN)
          ENDIF
      ENDIF
      GOTO 170

*
*
*
210   TYPE*, 'Error opening data file'
      STOP
*
220   TYPE*, 'Error reading data in file'
230   CLOSE(ERD_UNIT)
*
*
***** DATA IS READ IN, NOW START EVALUATION *****
*
*** Convert UNScaled and UNbiased data to proper values
      DO 6002 I=1,NCHAN
          DO 6003 J=1,NSAMP
              RDATA(I,J) = RDATA(I,J) / SCALE(I) + OFFSET(I)
6003          CONTINUE
              SCALE(I)=1.
              OFFSET(I)=0.
6002          CONTINUE
*
567   WRITE(5,1018)
1018   FORMAT(//, '***** CHANNELS *****', //)
      DO 1050 JI=1,NCHAN
          IF (JI .EQ. 6 .OR. JI .EQ. 12 .OR. JI .EQ. 18 .OR.
+          JI .EQ. 24 .OR. JI .EQ. 30 .OR. JI .EQ. 36) THEN
              WRITE(5,1032)
              READ(5,*)
          ENDIF
          WRITE(5,1060) JI, LONG_NAME(JI), UNIT_NAME(JI)
          FORMAT(/, ' CHANNEL ', I2, /, 1X, A32, /, 1X, A8)
1060          CONTINUE
1050          WRITE(5,1032)
1032          FORMAT(/, '***** HIT RETURN *****')
          READ(5,*)
***
      WRITE(5,1081)
1081   FORMAT(' ENTER CHANNEL NUMBER TO ANALYZE')
      READ(5,*) NCHANPSD
***
      UNITS=UNIT_NAME(NCHANPSD)

```

```

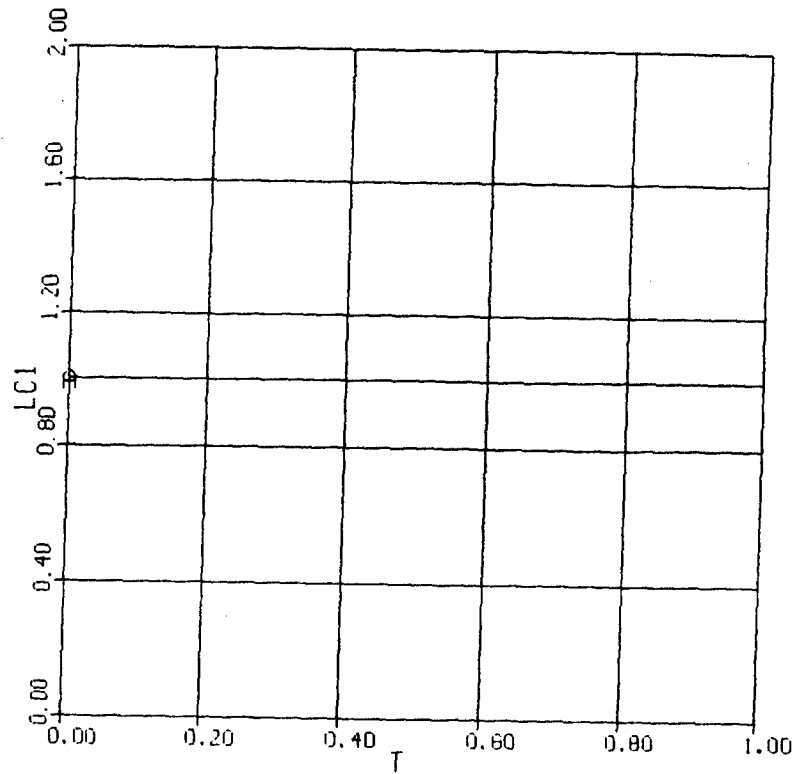
      CHANNEL=LONG_NAME(NCHANPSD)
*
*   DETERMINE TIME ARRAY FROM STEP
*
      DO 1061 I=1,NSAMP
        TIME(I)=STEP*(I-1)
        SIGNAL(I)=RDATA(NCHANPSD,I)
1061    CONTINUE
*
      RETURN
      END
*****
*****
*****
      SUBROUTINE TF_PLOT(ITERM,ERD_TITLE,NSAMP,STEP,NCHAN,DATA,
+      LONG_NAME,UNIT_NAME)
*
*
*   Must insert a subroutine for plotting ERD data format.
*   This particular application uses a PLOT10 Library routine
*   (Not Included). It is advised to use a subroutine which has
*   a graphics mode capable of plotting semi-log graphs.
*   The following format of input variables was used.
*
*   INPUT:  ITERM          ID for terminal
*           ERD_TITLE      Title for plot ('NOTE' from main)
*           NSAMP          Number of samples
*           STEP           Delta F (Hz)
*           NCHAN          Number of Channels (6 for this program)
*           DATA(NSAMP,Channel Number) Data in amplitude
*           LONG_NAME(Channel Number) Name for channel
*           UNIT_NAME(Channel Number) Unit for channel
*   OUTPUT; None (A plot on the screen! what else do you expect!!!)
*
*   TF_PLOT Subroutine...
*
      RETURN
      END
*****
*****
*****
      FOURIER SECTION *****
*****
      SUBROUTINE FOURIER(X,Y,NPT,REAL,AIMAG,FREQ,MM)
*
*   Must insert a fourier analysis subroutine.
*   The one used for this analysis was developed by IEEE for
*   digital processing. (Not included)
*   It determines the fourier coefficients for a time
*   history. The essential requirement is that the time history be
*   described by a Power-Of-2-Number-Of-Points.
*
*   INPUT;      Y (Nsamp)  Signal in Amplitude
*               X (Nsamp)  Corresponding Time (Seconds)
*               NPT        Corresponding Number Of points of signal
*                       (Program will help you choose closest
*                       power of 2)
*   OUPUT:      REAL       Real Coefficient      (Cos Term)
*               AIMAG      Imaginary Coefficient (Sin Term)
*
*               FREQ       Corresponding Frequency (Hz)
*               MM         Number of points in frequency domain.
*
*   FOURIER Subroutine...
*
      RETURN
      END

```

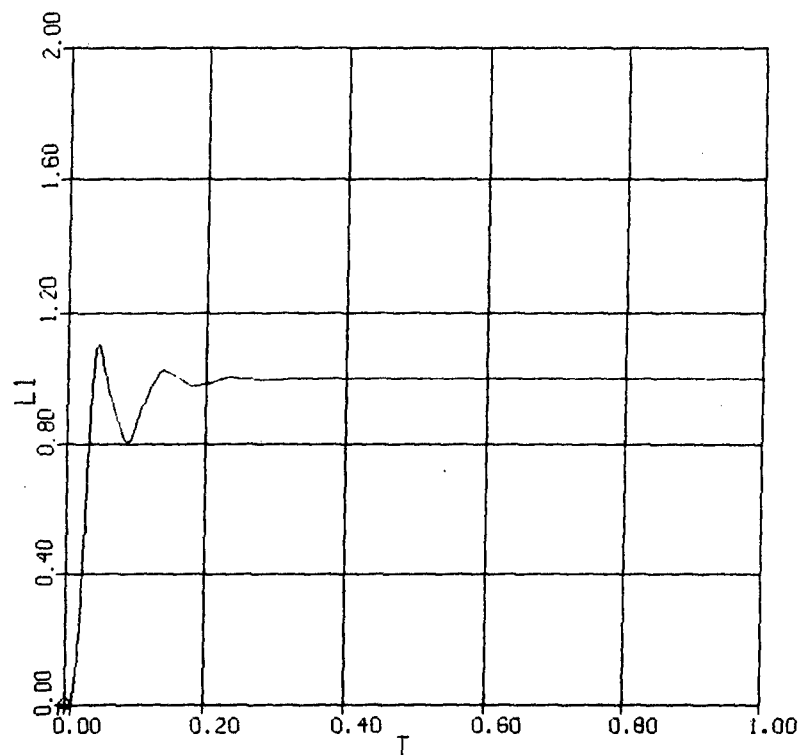
APPENDIX C
VARIOUS STATES OF A STEP RESPONSE

STEP RESPONSE 1 IN

POSITION COMMAND

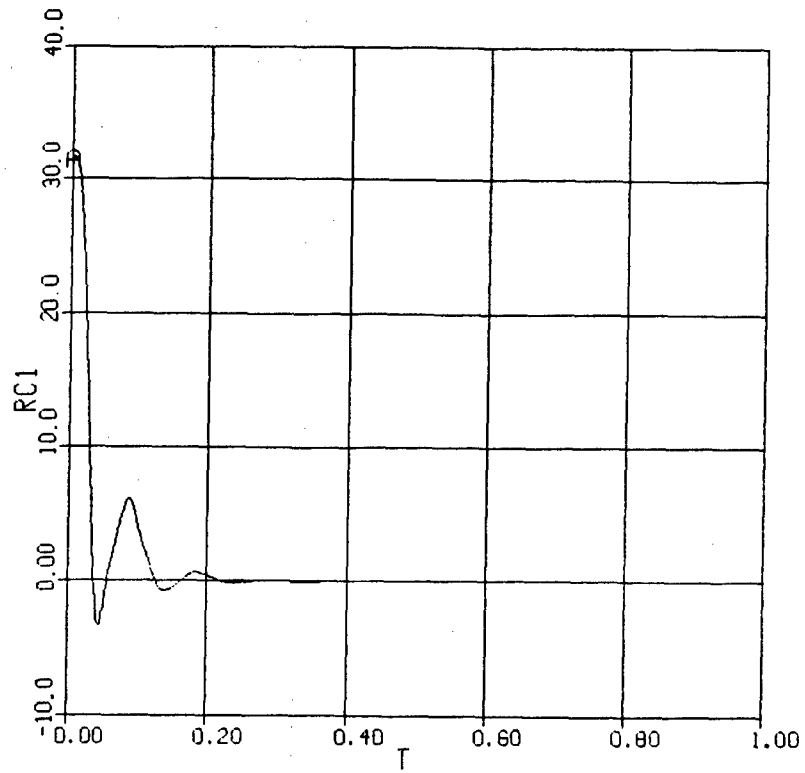


POSITION RESPONSE

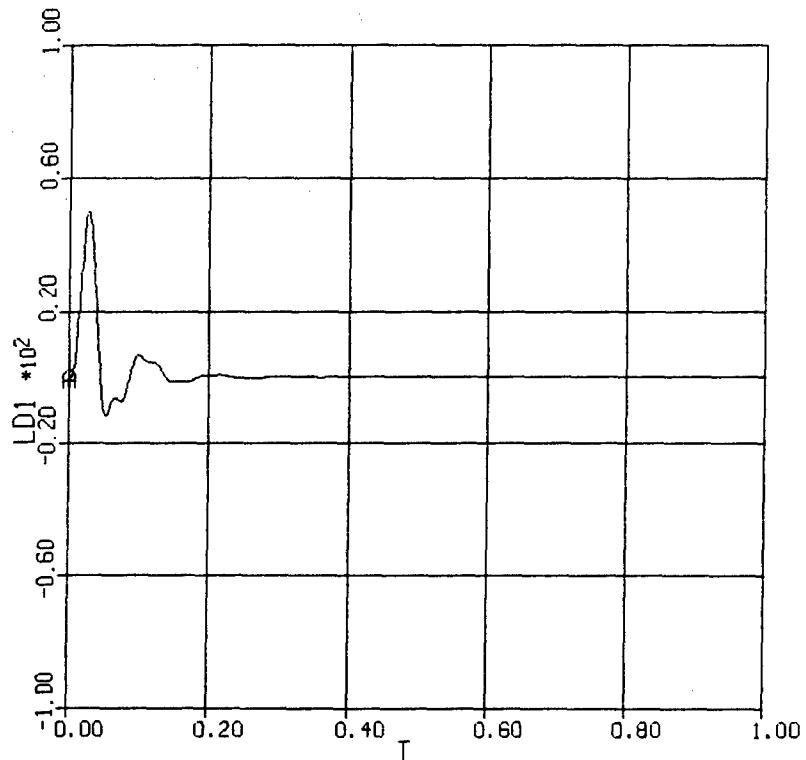


STEP RESPONSE 1 IN

RATE COMMAND

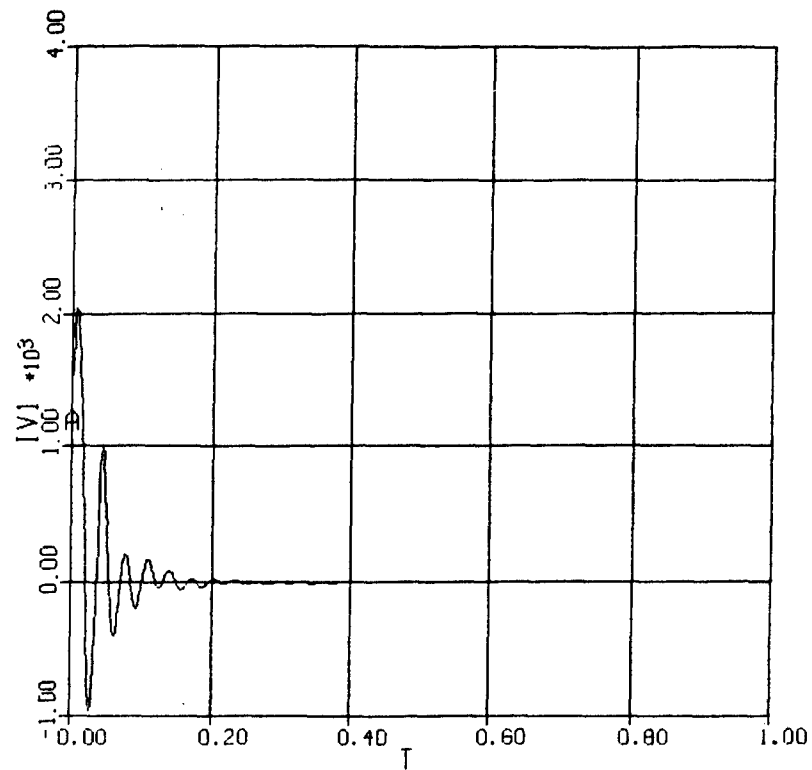


RATE RESPONSE

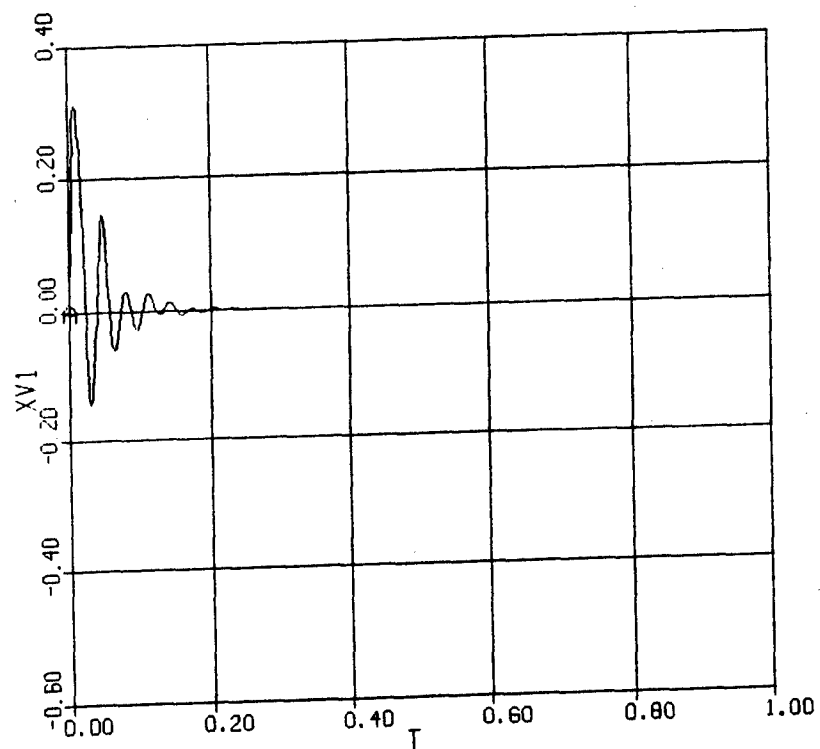


STEP RESPONSE 1 IN

SERVO INPUT CURRENT

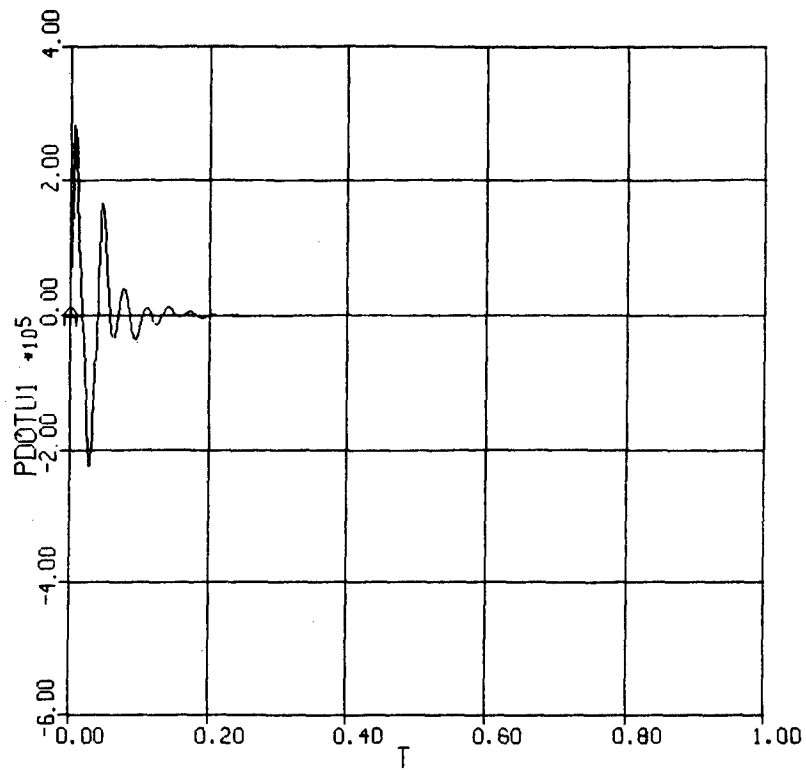


SPOOL POSITION

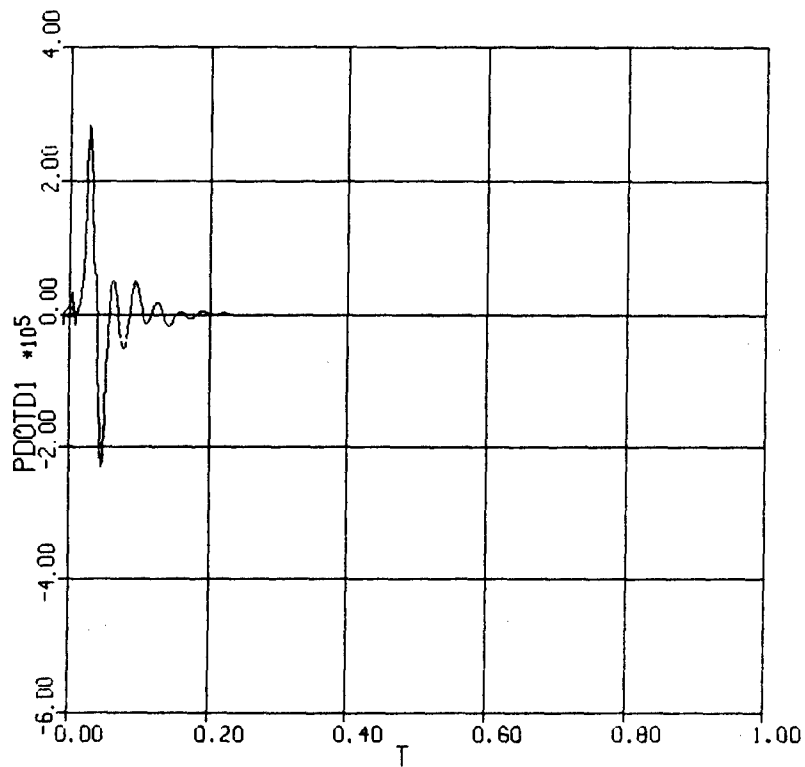


STEP RESPONSE 1 IN

∂ (PRESSURE) / ∂t (for up)

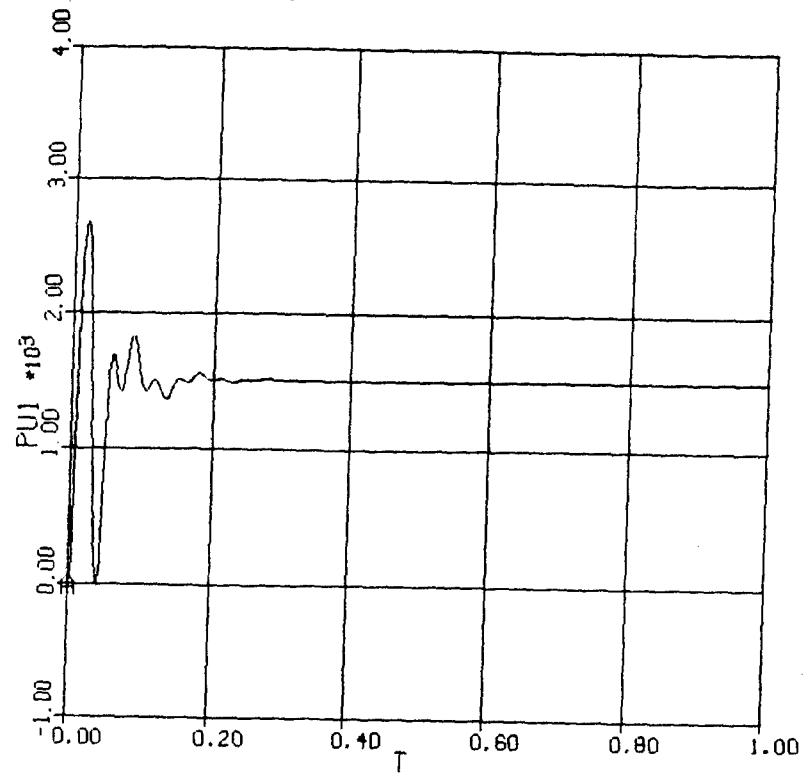


∂ (PRESSURE) / ∂t (for down)

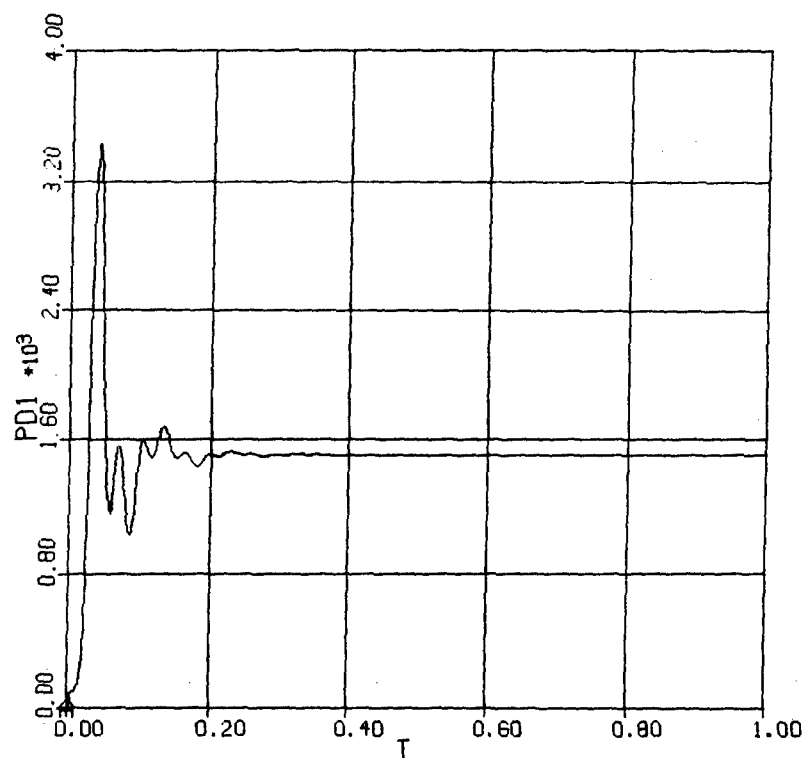


STEP RESPONSE 1 IN

ACTUATOR PRESSURE (for up)

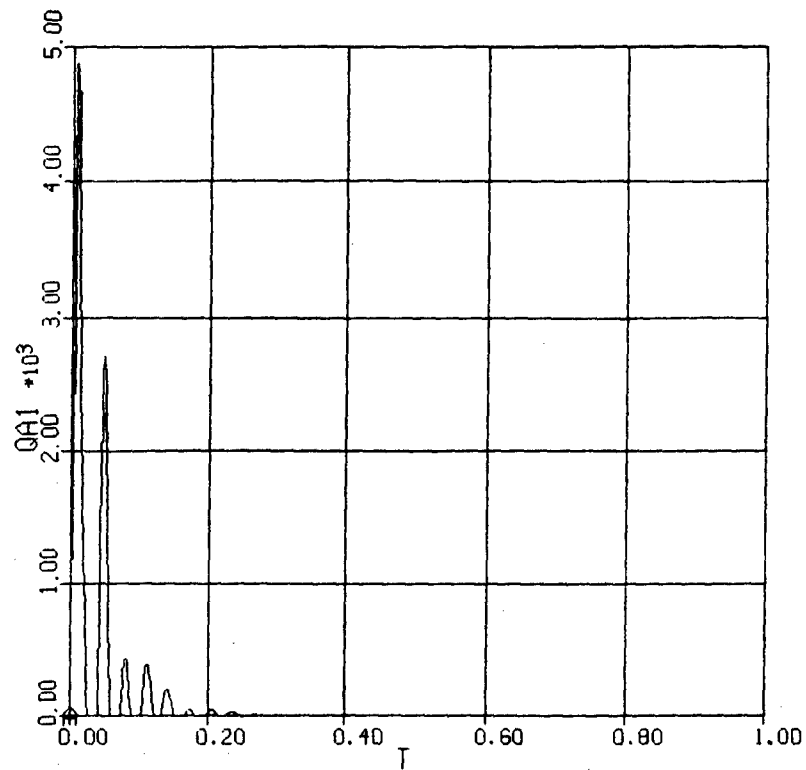


ACTUATOR PRESSURE (for down)

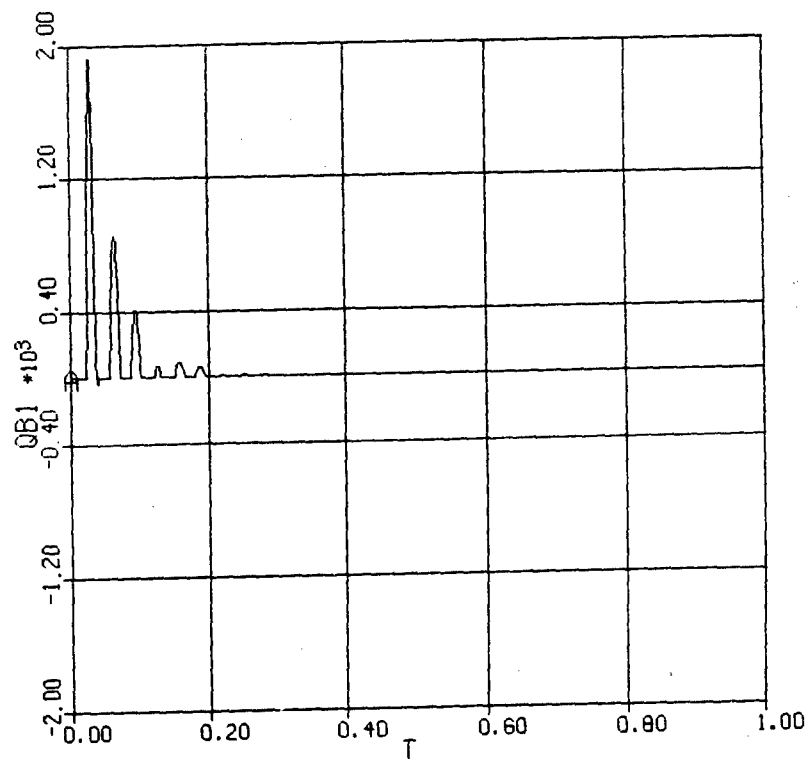


STEP RESPONSE 1 IN

FLOW IN (for up)

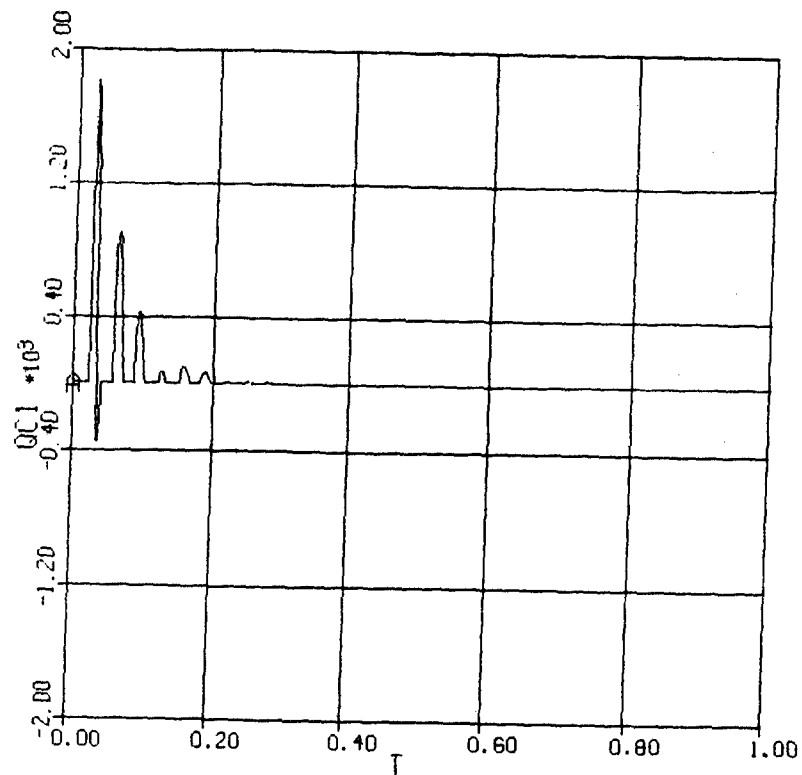


FLOW OUT (for up)

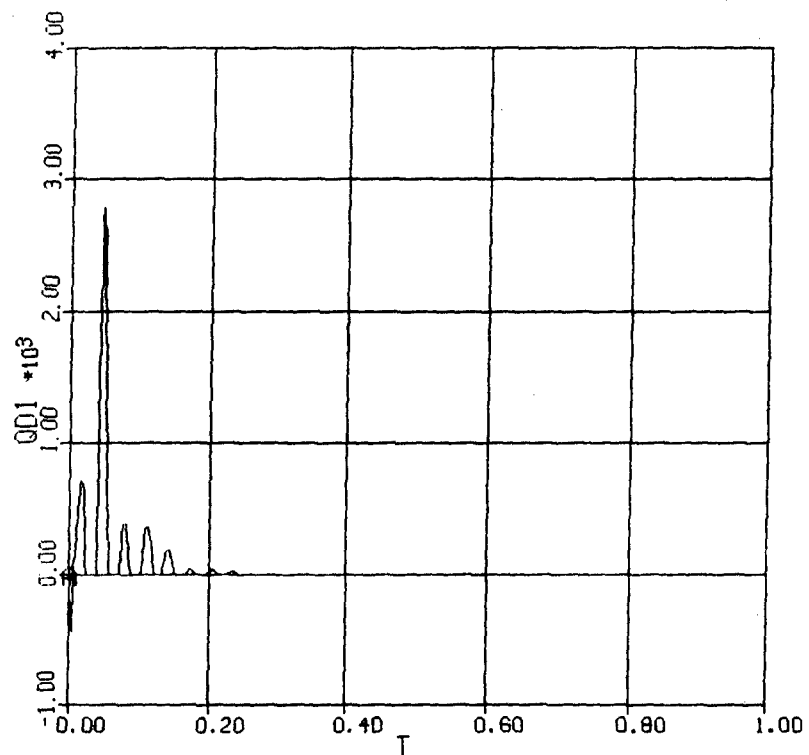


STEP RESPONSE 1 IN

FLOW IN (for down)

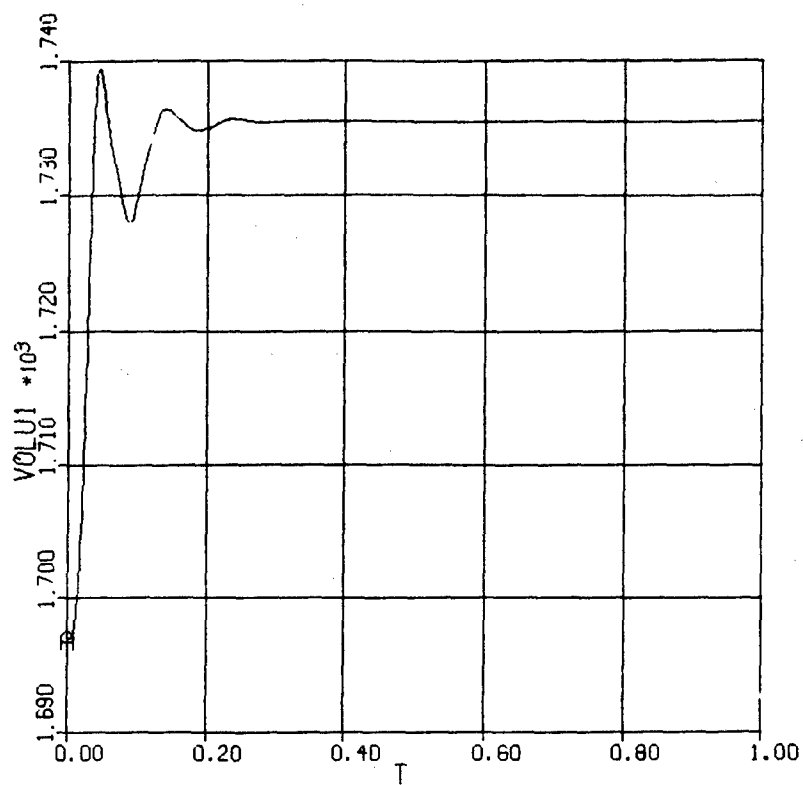


FLOW OUT (for down)

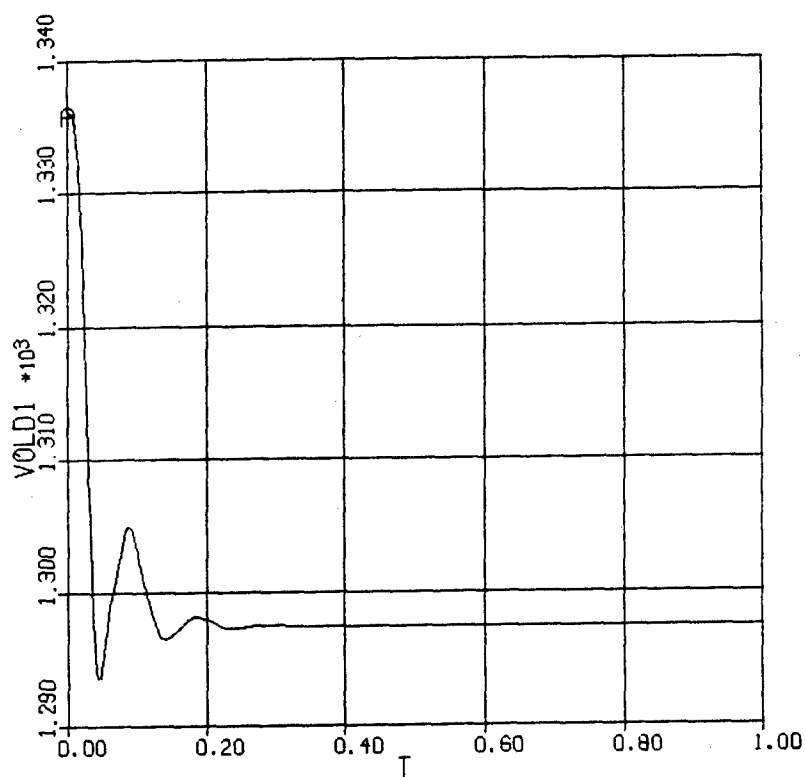


STEP RESPONSE 1 IN

ENTRAINED VOLUME (for up)

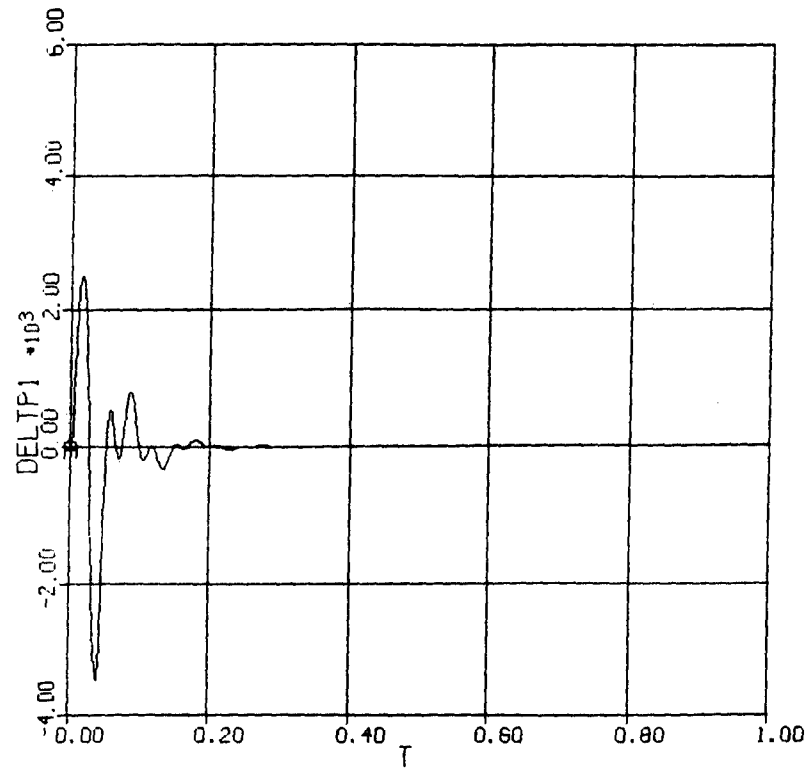


ENTRAINED VOLUME (for down)

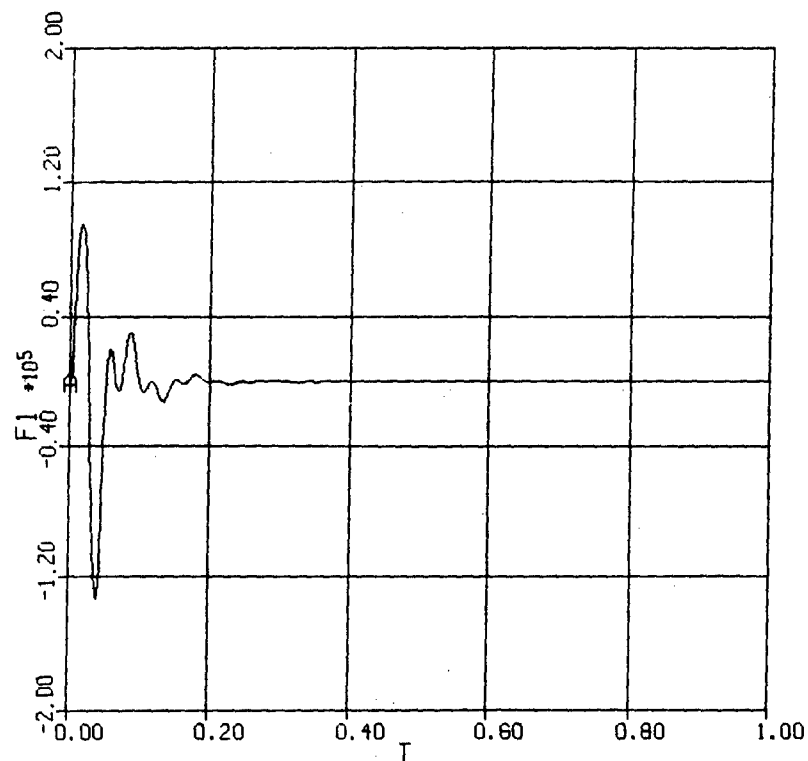


STEP RESPONSE 1 IN

ACTUATOR Δ Pressure



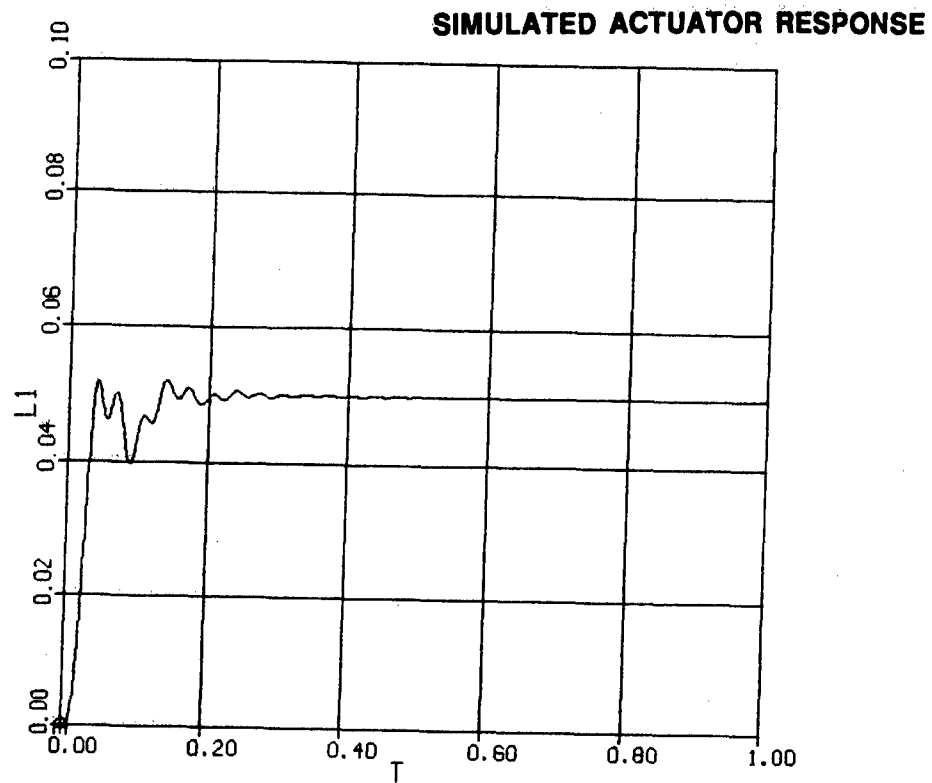
ACTUATOR FORCE



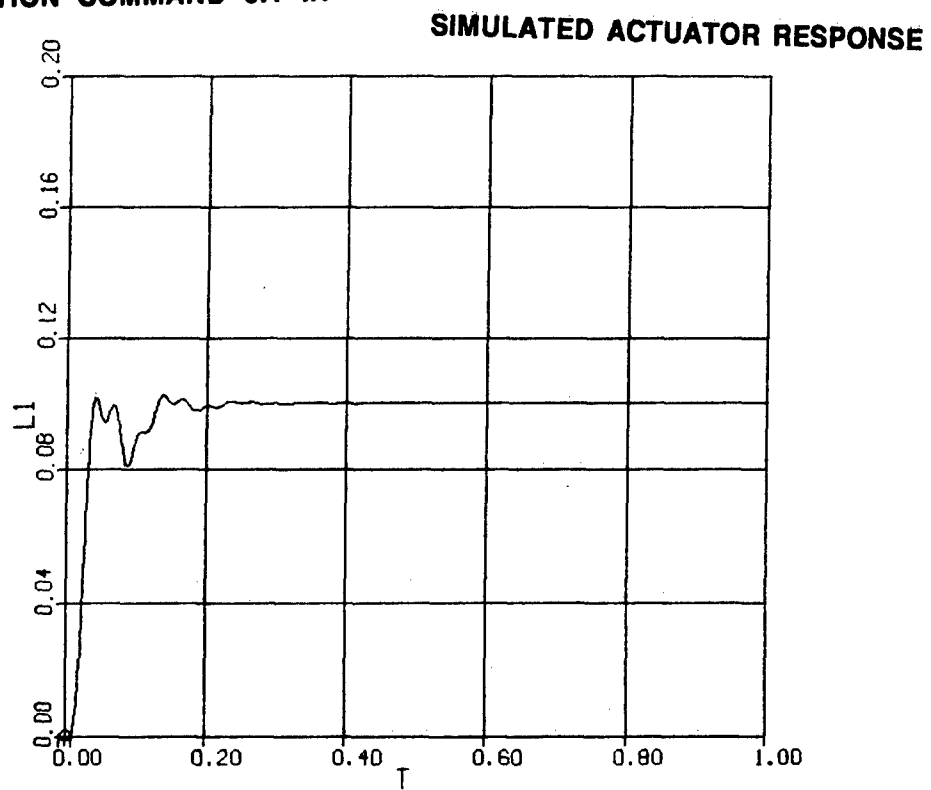
APPENDIX D
VARIED STEP RESPONSE RESULTS

VARIED STEP RESPONSE

POSITION COMMAND .05 IN



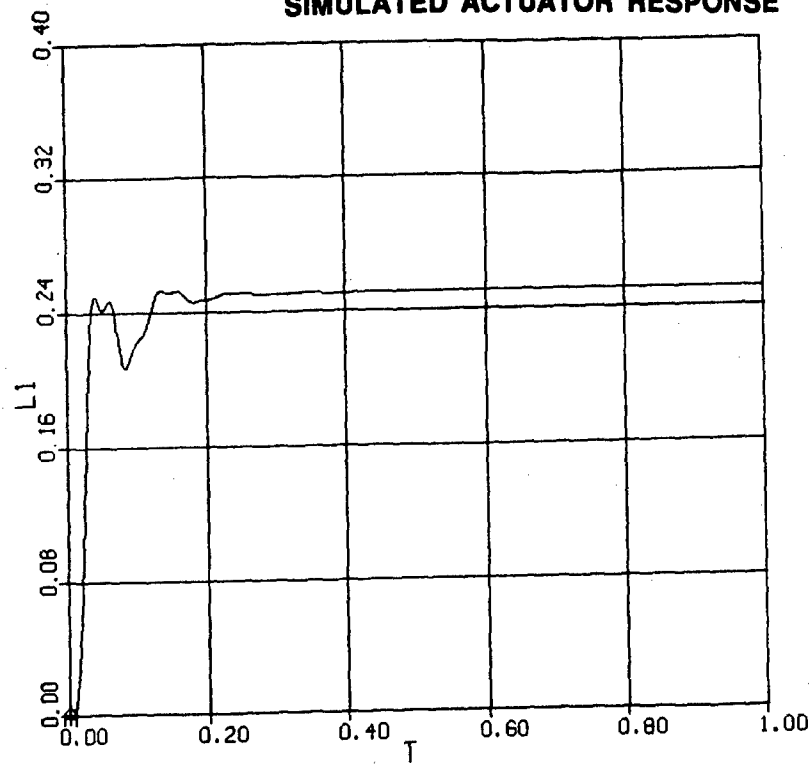
POSITION COMMAND 0.1 IN



VARIED STEP RESPONSE

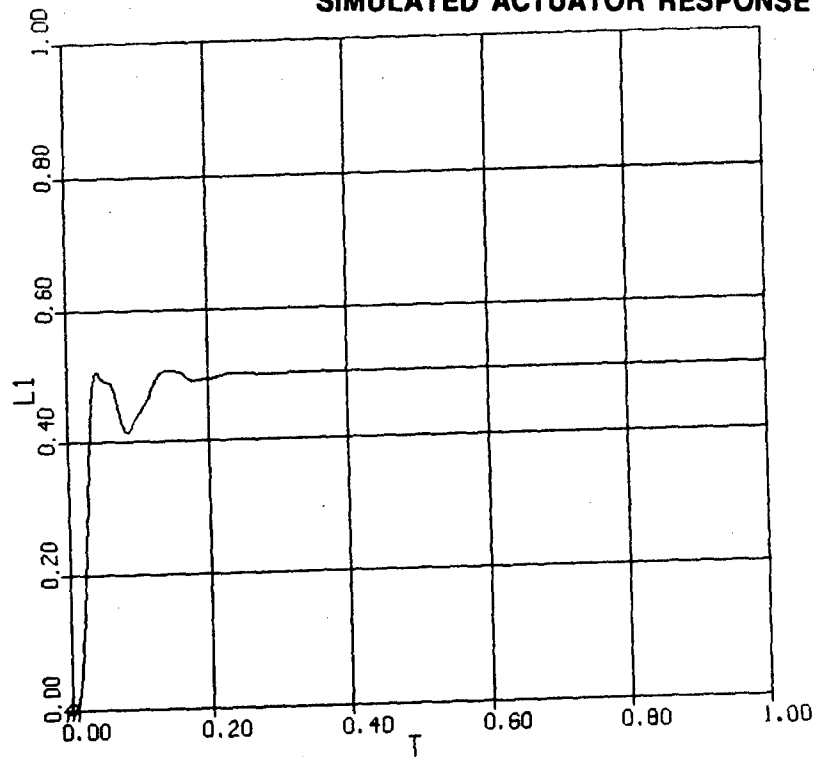
POSITION COMMAND 0.25 IN

SIMULATED ACTUATOR RESPONSE



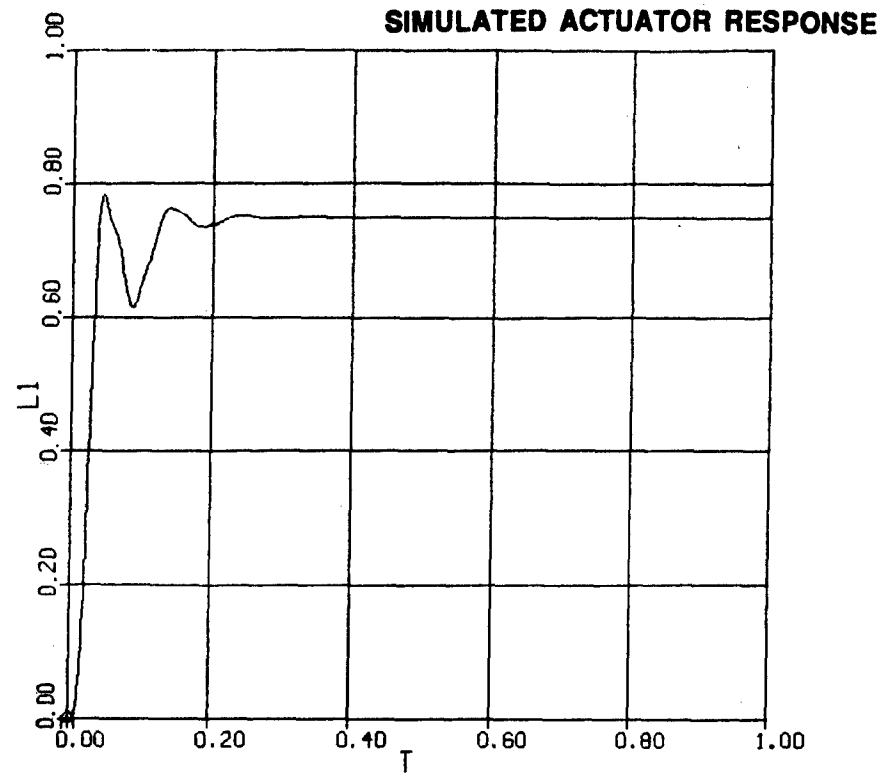
POSITION COMMAND 0.5 IN

SIMULATED ACTUATOR RESPONSE

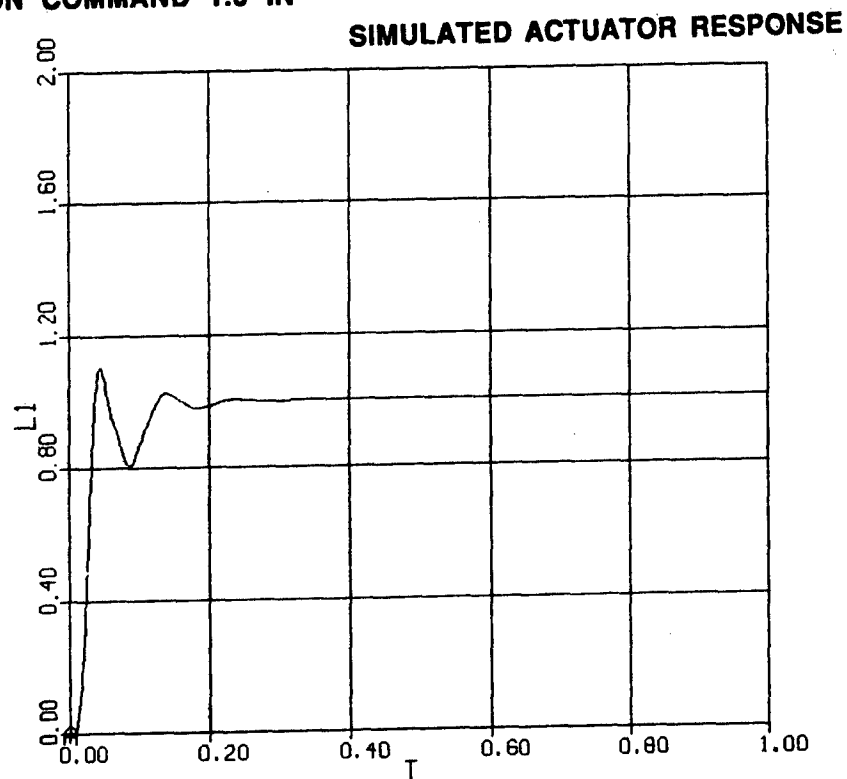


VARIED STEP RESPONSE

POSITION COMMAND 0.75 IN



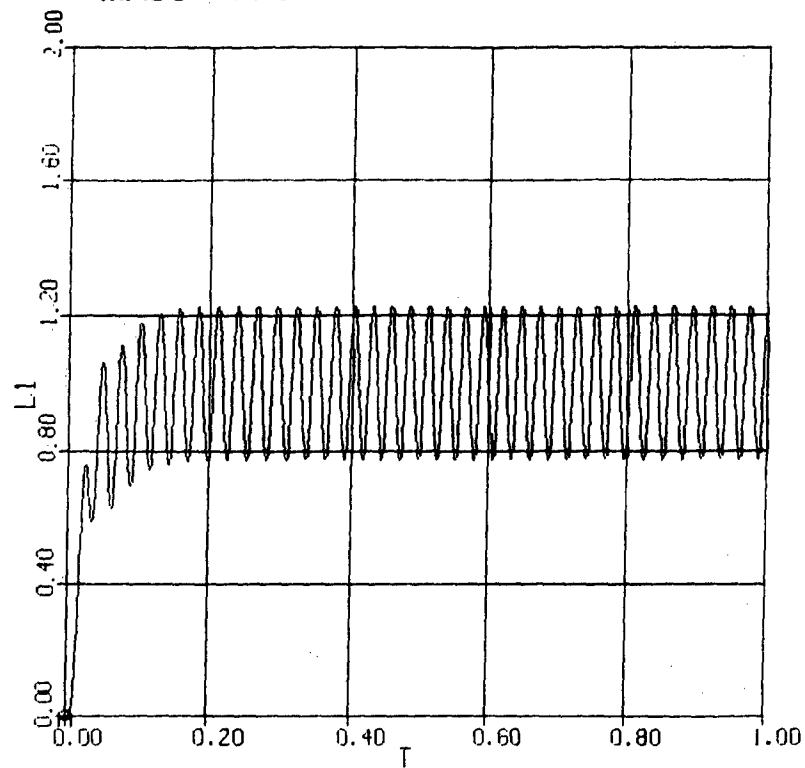
POSITION COMMAND 1.0 IN



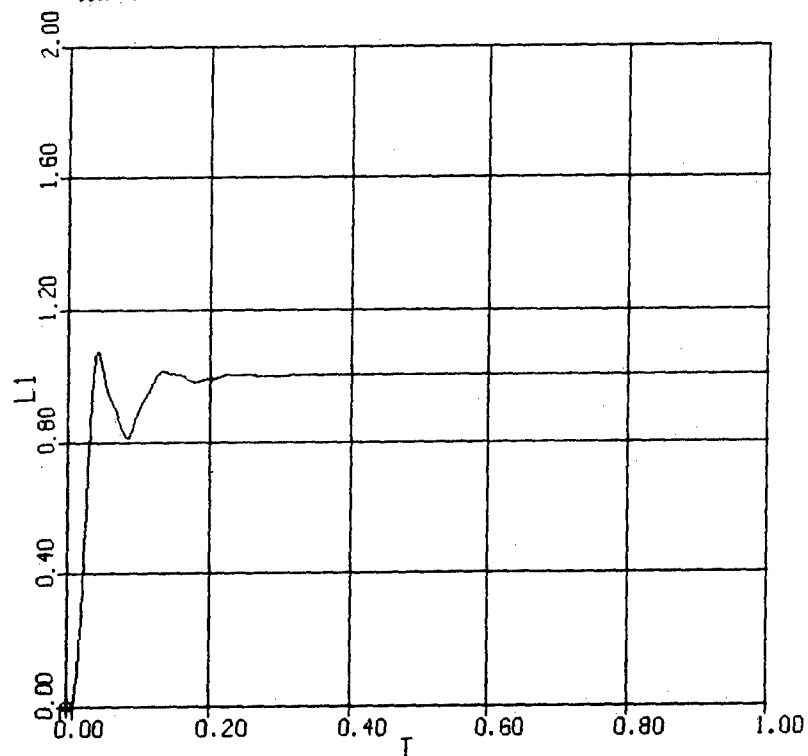
APPENDIX E
VARIED MASS RESULTS

VARIED MASS 1 IN STEP RESPONSE

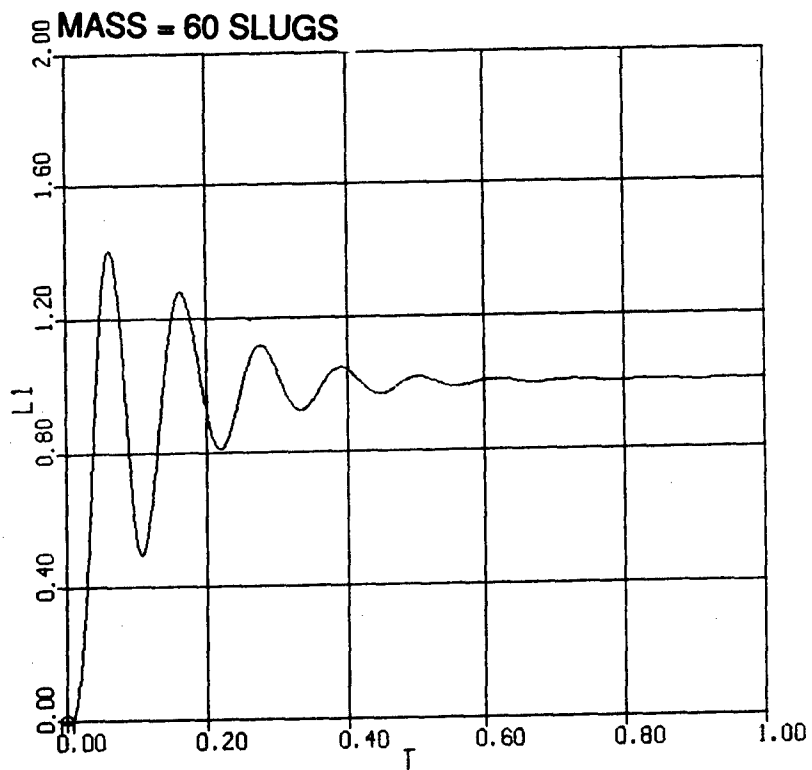
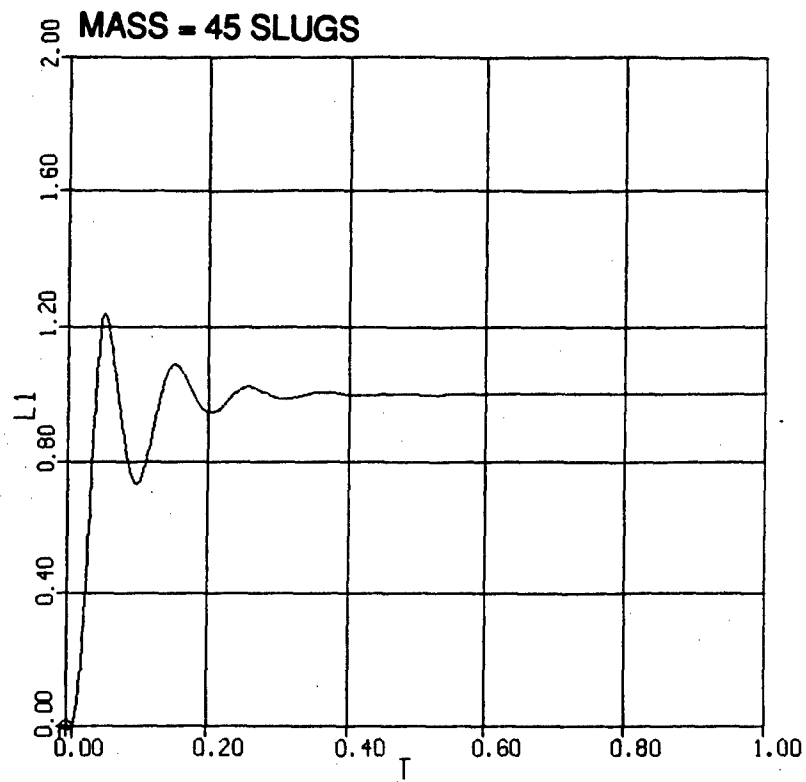
MASS = 10 SLUGS



MASS = 30 SLUGS



VARIED MASS 1 IN STEP RESPONSE



DISTRIBUTION LIST

	Copies
Commander	
U. S. Army Tank-Automotive Command	
ATTN: ASNC-TAC-DIT (Technical Library)	2
AMSTA-CF (Mr. Orlicki)	1
AMSTA-CR (Mr. Wheelock)	1
AMSTA-R (Mr. Jackovich)	1
AMSTA-RR (Robotics Division)	1
AMSTA-RTS (Dr. Hoogterp)	1
AMSTA-RV (Mr. Sirna)	1
AMSTA-RY (Systems Simulation and Tech. Div.)	22
AMSTA-TB	1
AMSTA-U (Dir. for Systems Eng.)	1
AMSTA-Z (Concepts & Tech. Demonstration Div.)	1
Warren, MI 48397-5000	
Commander	12
Defense Technical Information Center	
Bldg. 5, Cameron Station	
ATTN: DDAC	
Alexandria, VA 22304-9990	
Manager	
Defense Logistics Studies Information Exchange	2
ATTN: AMXMC-D	
Fort Lee, VA 23801-6044	
Commander	
U. S. Army Foreign Science & Tech Center	1
220 Seventh Street NE	
ATTN: AIAST-RA (Research and Analysis Dir.)	
Charlottesville, VA 22901-5396	
U. S. Army Laboratory Command	1
Army Research Office	
P. O. Box 12211	
ATTN: SLCRO-EG (Engineering Division)	
SLCRO-TS (Library Services)	
Research Triangle Park, NC 27709-2211	
Director	1
Harry Diamond Laboratories	
ATTN: SLCHD-IT (Eng. & Tech Support Div.)	
SLCHD-TA (Tech. Applications Div.)	
2800 Powder Mill Road	
Adelphi, MD 20783-1197	
Commander	2
U. S. Army Material Command	
ATTN: AMCDE (Development, Eng, & Acquisition)	
AMCDMA-ML (Library)	
5001 Eisenhower Avenue	
Alexandria, VA 22333-001	
Director	1
CECOM Research Development & Engineering Center	
ATTN: AMSEL-RD (Director)	
Fort Monmouth, NJ 07703-5001	

Dist-1

Commander	2
U. S. Army Missile Command	
ATTN: AMSMI-CG (Reference Library)	
AMSMI-RD (Research, Dev. and Eng. Center)	
Redstone Arsenal, AL 35898-5000	
Commander	1
U. S. Army Combined Arms Center and Fort Leavenworth	
ATTN: ATOR-CT (Scientific and Tech. Support Dir.)	
Fort Leavenworth, KS 66027-5130	
Commander	2
TRADOC Combined Arms Test Activity	
ATTN: ATCT-MA (Methodology & Analysis Dir.)	
ATCT-SPT (Technical Library)	
Fort Hood, TX 76544-5065	
President	
U.S. Army Communications-Electronics Board	1
ATTN: ATZH-BDS (Analysis & Tech. Support Div.)	
Fort Gordan, GA 30905-5350	
President	1
U.S. Army Field Artillery Board	
ATTN: ATZR-BDS (Analysis and Tech. Support Div.)	
Fort Sill, OK 73503-6100	
Commander	1
U. S. Army Belvoir Research, Development, & Engineering Center	
ATTN: STRBE-BT (Technical Library Div.)	
Fort Belvoir, VA 22060-5606	
Commander	1
U. S. Army Natick	
Research, Development, and Engineering Center	
ATTN: STRNC-ML (Technical Library)	
Natick, MA 01760-5000	
HQDA	2
Office of Dep Chief of Staff for Rsch Dev & Acquisition	
ATTN: ARZ-A Dr. Lasser - Dir. of Army Research	
DAMA-AR	
Washington, D. C. 20310	
Director	4
U. S. Army Material Systems Analysis Agency	
ATTN: AMXSY-DD	
AMXSY-C (Mr. Harold Burke)	
AMXSY-CM (Mr. Fordyce)	
AMXSY-MP (Mr. Cohen)	
Aberdeen Proving Grounds, MD 21005-5071	
Director	1
Keweenaw Research Center	
Michigan Technological University	
Houghton, MI 49931	

Director Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22209	1
Engineering Society Library 345 East 47th Street New York, NY 10017	1
Director TRADOC Systems Analysis Activity ATTN: ATOR-TF White Sands Missile Range, MN 88002-5502	1
General Dynamics Land Systems Division Analytical Engineering ATTN: R. J. Thompson ATTN: Glenn Socks ATTN: Kate Bush MZ 436-21-19 P.O. Box 2045 Warren, MI 48090	3
TRW Steering & Suspension Division Research & Development ATTN: Dr. Dave J. D'Onofrio ATTN: Roy Harper 34201 Van Dyke Ave. Sterling Heights, MI 48077	2
Duke University Mechanical Engineering ATTN: Dr. D. P. Garg Durham, NC 27706	1
Contraves Goerz Corporation ATTN: TMBS Manager (J. Smith) 610 Epsilon Drive Pittsburg, PA 15238	2
Oakland University S.E.C.S. / Dept. of Elect. & Syst. Engr. ATTN: Dr. K. C. Cheok Rochester, MI 48309-4441	1
University of Michigan Transportation Research Institute ATTN: Library (Ann Grimm) 2901 Baxter Road Ann Arbor, MI 48109-2150	1