

N72 28252

DISTRIBUTION OF FINAL REPORT

NASA Contract NAS 9-7926

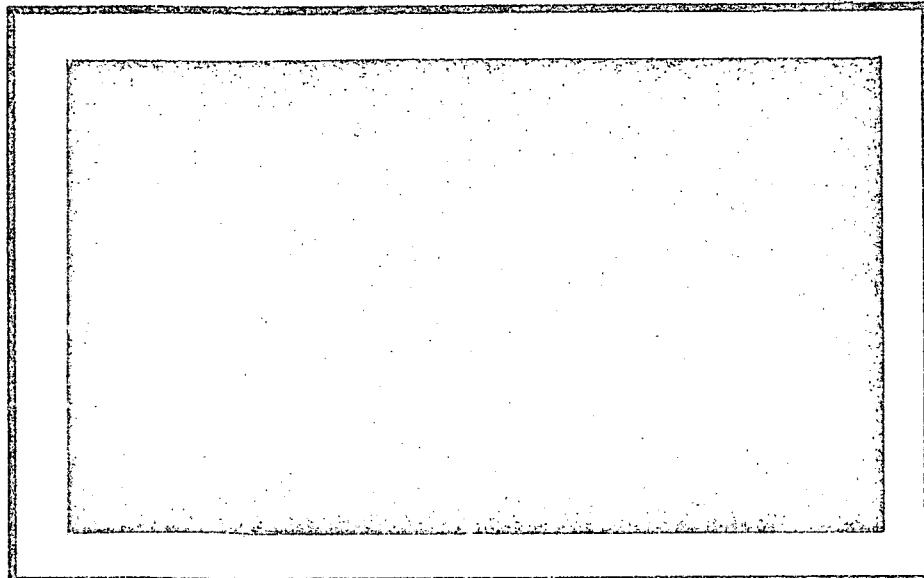
Mr. E. E. Smith, Jr. 6 copies
Systems Analysis Branch
NASA Manned Spacecraft Center
Houston, Texas 77058

Mr. Robert Luddy/BB 321 (80) 1 copy
Facilities and Laboratory Support Branch
NASA Manned Spacecraft Center
Houston, Texas 77058

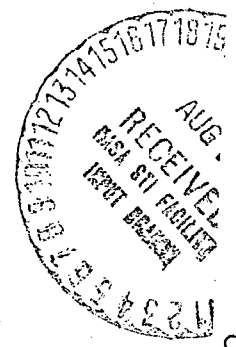
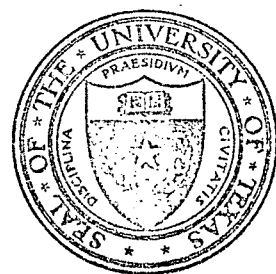
Ms. Retha Shirkey/BM 6 4 copies
Technical Information Dissemination Branch
NASA Manned Spacecraft Center
Houston, Texas 77058

Mr. John T. Wheeler/BM 6 1 copy
Management Services Division
NASA Manned Spacecraft Center
Houston, Texas 77058

CR-115741



Reproduced from
best available copy.



THE UNIVERSITY OF TEXAS
COLLEGE OF ENGINEERING
AUSTIN

(NASA-CR-115741) UTILIZATION OF A CRT
DISPLAY LIGHT PEN IN THE DESIGN OF FEEDBACK
CONTROL SYSTEMS Final Report J.G.
Thompson, et al (Texas Univ.) May 1972

CSCI 09C G3/10

Unclas
37590

N72-28252

FINAL REPORT

UTILIZATION OF A CRT DISPLAY LIGHT PEN
IN THE DESIGN OF
FEEDBACK CONTROL SYSTEMS

NASA MANNED SPACECRAFT CENTER
SYSTEMS ANALYSIS BRANCH
HOUSTON, TEXAS 77058

MAY, 1972

NASA CONTRACT -- NAS 9-7926

TO: DR. KENNETH J. COX AND
MR. E. E. SMITH, Jr.

FROM: DR. J. GARTH THOMPSON AND
MR. KENNETH R. YOUNG

DEPARTMENT OF MECHANICAL ENGINEERING
THE UNIVERSITY OF TEXAS AT AUSTIN
AUSTIN, TEXAS 78712



THE UNIVERSITY OF TEXAS AT AUSTIN
COLLEGE OF ENGINEERING
AUSTIN, TEXAS 78712

Department of Mechanical Engineering
512 GR 1-1136

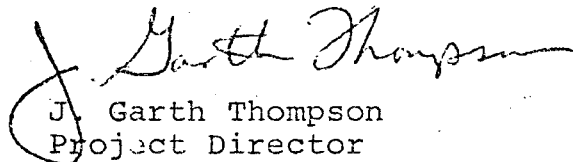
May, 1972

Mr. Robert Luddy/BB 321(80)
Facilities and Laboratory Support Branch
NASA Manned Spacecraft Center
Houston, Texas 77058

Dear Mr. Luddy:

I am pleased to send the attached final report for NASA Contract NAS 9-7926. It has been a great pleasure for me to supervise the performance of the work reported herein. Many very interesting results have been produced by this contract as reported herein. The University of Texas and many of my students have been the greatest benefactors of this work. I trust that the results will be of interest and use to NASA. I sincerely appreciate the contract which made this work possible.

Sincerely,


J. Garth Thompson
Project Director

JGT/pea

Attachment

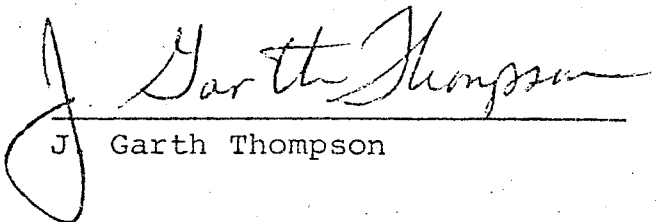
cc: J. P. Lamb
Bobby G. Cook
Jens Jacobsen
R. W. Prestridge
Jack Fuller

ACKNOWLEDGEMENTS

It is a pleasure to acknowledge the many valuable ideas and suggestions of Kenneth J. Cox, Emory E. Smith, Jr. and William H. Peters of the Systems Analysis Branch, Guidance and Control Division, NASA Manned Spacecraft Center, Houston, Texas. Their encouragement and support throughout the project has been invaluable. The project would not have been possible without many long and dedicated hours from my students. Special thanks goes to Kenneth R. Young whose brilliant analysis and programming skill has brought to reality a most ambitious project.

I appreciatively acknowledge the financial support of NASA Manned Spacecraft Center and the technical support and use of facilities of the Computation Center of The University of Texas.

I appreciate the secretarial assistance of Mrs. Patty Adams throughout the project and particularly in typing this report.


J Garth Thompson

ABSTRACT

In May, 1968, a project to develop the capability to perform computer-aided design of feedback control systems was initiated at The University of Texas at Austin with support from The National Aeronautics and Space Administration. The project has generated computer programs to accept feedback control system model descriptions through an interactive graphics terminal, to perform analyses and simulation of the described system and to present the results of the analyses and simulation on the graphics terminal. The programs are interactive and interlinked so that the user can at any point return to the model description routines to modify his system and obtain a new analysis or simulation to compare with the previous results.

The principal technical contributions of this work are:

1. The development of the hierarchical structure of the interlinked programs to provide a completely flexible computer-aided design tool.
2. The development of a graphical input technique and a data structure which provides the capability of entering the control system model description into the computer in block diagram form.

3. The development of an information storage and retrieval system to keep track of the system description and all the analyses and simulation results and provide them to the correct routines in the correct form for further manipulation or display.

4. The development of error analysis and diagnostic capabilities to prevent the unsuspecting generation of erroneous results.

5. The development of a technique to reduce a transfer function to a set of nested integrals suitable for digital simulation and

6. The development of a general, automated block diagram reduction procedure to prepare the system description for the analyses routines.

TABLE OF CONTENTS

	PAGE
Acknowledgements	2
Abstract	3
Table of Contents	5
I. Introduction	6
II. Operation of the Interactive Design System . .	8
III. Conclusions	24

APPENDICES

- A. ORIGINAL PROPOSAL
- B. HARDWARE
- C. SYSTEM SOFTWARE
- D. MASTER SELECTION MODE
- E. DRAW MODE
- F. EQUATION MODE
- G. DATA MODE
- H. SIMULATION MODE
- I. FREQUENCY RESPONSE MODE
- J. ROOT LOCUS MODE
- K. PLOT AND PLOT TITLES MODES
- L. READ AND WRITE TAPE MODES
- M. BLOCK DIAGRAM REDUCTION

I. INTRODUCTION

In May, 1968, a project to develop the capability to perform computer-aided design of feedback control systems was initiated at The University of Texas at Austin with support from The National Aeronautics and Space Administration. The project was completed in October, 1971. This document is the final report of the work performed on that project. The technical monitor for the project was the Systems Analysis Branch, Guidance and Control Division, NASA Manned Spacecraft Center, Houston, Texas. The goal of the project was to develop computer programs to accept a control system model description through an interactive graphics terminal, to perform analyses and simulation of the described system and to present the results of the analyses and simulation on the graphics terminal. The programs are interactive and interlinked so that the user can at any point return to the model description routines to modify his system and obtain a new analysis or simulation to compare with previous results. A control system engineer using this facility is able in a very short time to investigate a variety of system configurations and parameter variations to arrive at a suitable design for the system under consideration. In the initial proposal (see Appendix A) the project was outlined as a three year effort and the tasks to be accomplished were given as:

1. Develop the means of easily communicating the system structure to the computer utilizing conventional control system terminology.

2. Develop a program to determine and display the locus of the roots of the system characteristic equation as a function of any parameter in the system.

3. Develop a program to determine and display a Bode plot for either the open loop or closed loop transfer function.

4. Develop a program to determine and display a Nyquist plot for either the open loop or closed loop transfer function.

5. Develop a program to perform a transient response simulation of the system and display the results on the CRT.

6. Develop a supervisory program capable of communicating with all of the above programs and with the control engineer at the CRT.

All of the essential concepts to accomplish these tasks have been developed and successfully tested. It was hoped that by this time all of the routines would be integrated into the final working program. Four factors have contributed to this project taking longer than anticipated. They are:

1. The move of the Project Director to Kansas State University.

2. The loss of one graduate student who was writing an important part of the program.

3. The difficulty encountered in the development of a general purpose block diagram reduction procedure and

4. The difficulty of programming and debugging a program of this magnitude.

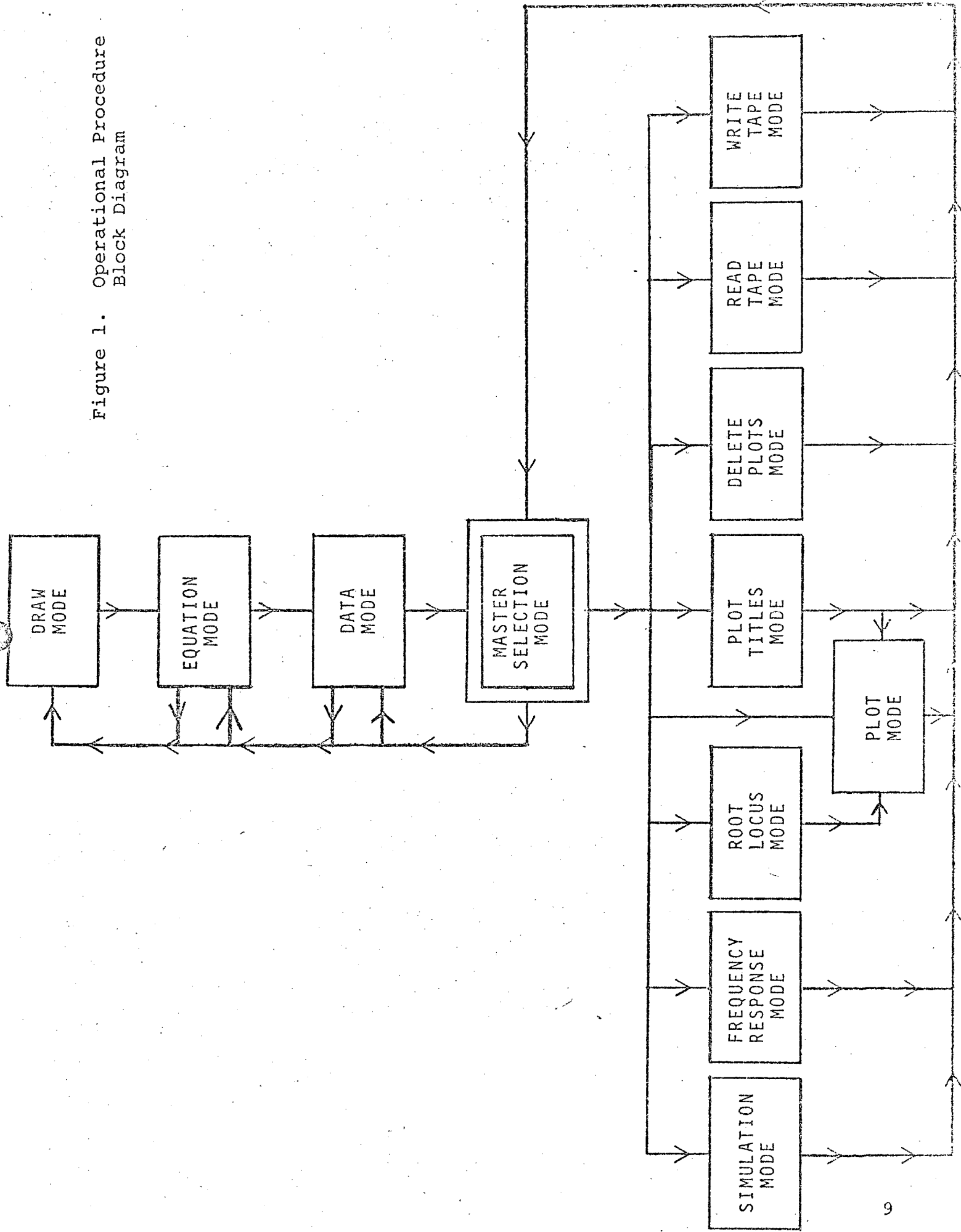
One Ph.D. student is finishing up the integration of the various parts of the project. A copy of his dissertation will be provided to the Systems Analysis Branch when he has completed it.

The remainder of this report gives an overview of the operation of the program. A detailed description of the various elements of the program is provided in the appendices.

II. OPERATION OF THE INTERACTIVE DESIGN SYSTEM

The operation of the interactive design system is divided into twelve modes. These and the branching operations between them are illustrated in Figure 1. The MASTER SELECTION MODE is the central operating point from which all other modes can be reached. The program comes to the MASTER SELECTION MODE when it is initially loaded at the beginning of operation. In MASTER SELECTION MODE the engineer is presented with a menu of operations from which he may select the next operation he wants to perform. Figure 2 is an illustration of the menu which is presented to the engineer. If a new control system is to be entered or the block diagram structure of the existing system modified the engineer would select the DRAW MODE by taking a light pen hit on the title "DRAW". The DRAW MODE may be reached from the MASTER SELECTION MODE, from the EQUATION MODE, or from the DATA MODE. Once the DRAW MODE is selected you cannot get back to the MASTER SELECTION MODE except through

Figure 1. Operational Procedure Block Diagram



MASTER SELECTION

DRAW			PLOT
EQUATION			TITLES
DATA			DELETE
SIMULATION		STORED	NEW
ROOT LOCUS	1	6	1
FREQUENCY RESPONSE	2	7	2
WRITE TAPE	3		3
READ TAPE	4		4
	5		

Figure 2 MASTER SELECTION MODE menu

the EQUATION MODE and the DATA MODE. Figure 3 illustrates the display during the DRAW MODE. Across the bottom of the screen is a menu of 7 elements which are used to create the block diagram of the system. These elements may be selected with the light pen and placed on the screen. Lines are also placed between elements by selecting the line element with the light pen and touching the light pen to the elements to be connected. While the block diagram is being formed the program generates a data structure which contains all the information necessary for all future operations on the block diagram. The four control commands FILM IT, COMPLETE, CLEAR, and SETBLOCK are in the upper left hand corner of the screen. Touching the light pen to the FILM IT command causes a microfilm record to be made of the current display. The COMPLETE command is touched with the light pen when the block diagram is complete and the engineer wants to proceed to the EQUATION MODE. Touching the light pen to the CLEAR command clears the block diagram from the screen and permits the engineer to start over. The SETBLOCK command is used in forming the block diagram. When an element has been selected from the menu and placed in the desired position on the screen the SETBLOCK command is touched with the light pen. This action "sets" the block in the selected position. The tracking cross is also on the screen. It is also used in placing elements on the screen. The tracking cross will follow the light pen around the screen. After an element is selected from the menu, the tracking cross

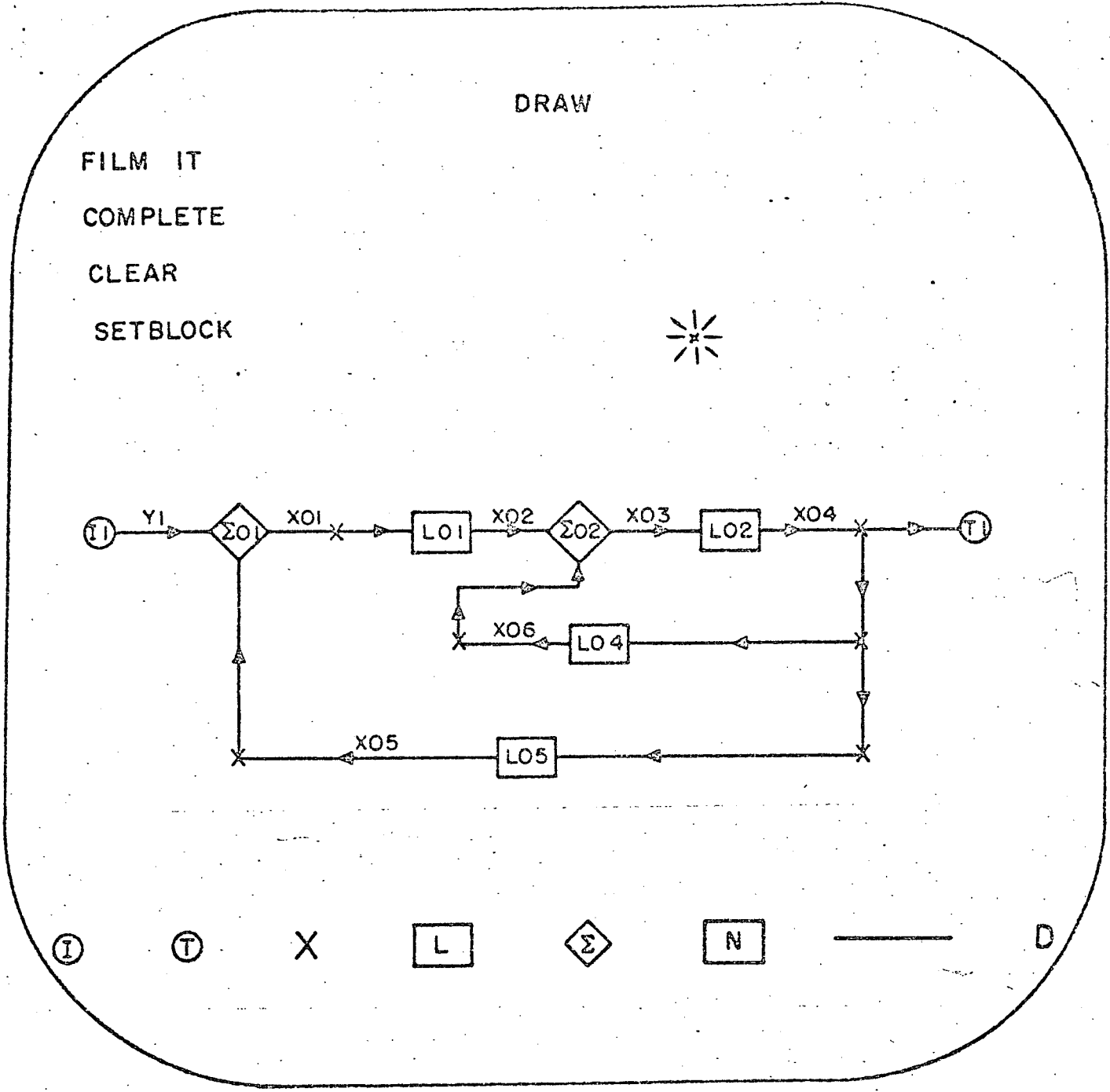
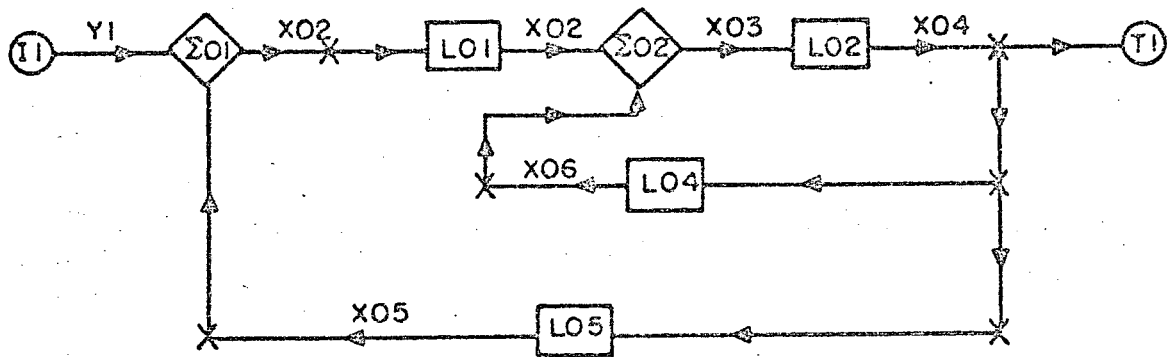


Figure 3 Display Presented During DRAW MODE

is placed with the light pen in the position where the selected element is to go. After the block diagram is completed, the engineer touches the light pen to the COMPLETE command to proceed to the EQUATION MODE. The program checks to be certain that a correct block diagram has been entered. If an error is detected an appropriate error message will be displayed, otherwise the program enters the EQUATION MODE.

The EQUATION MODE may be entered from the DRAW MODE, the MASTER SELECTION MODE or from the DATA MODE. Entrance to the EQUATION MODE from the MASTER SELECTION MODE would be made when the engineer wanted to alter the equations for his system without altering the basic form of the block diagram. Entrance of the EQUATION MODE from the DRAW MODE occurs in the normal procedure of developing the block diagram structure and then specifying the mathematical relationships for each block. Entrance to the EQUATION MODE from the DATA MODE provides the capability to cycle back to the EQUATION MODE if it becomes evident during the entry of data that the mathematical model should be changed. Figure 4 illustrates the form of the display in the EQUATION MODE. In this mode the engineer provides a mathematical model for each block in the block diagram. The display includes the block diagram for reference. The program sets one of the blocks winking and displays information cues which prompt the engineer on the form of model information he may enter for each block. As the engineer enters the model the program forms a data structure

EQUATION



DESCRIPTION OF WINKING LINEAR ELEMENT. COEFFICIENTS MUST BE CONSTANTS.
 FORM IS RATIO OF POLYNOMIALS $0 \leq N \leq M \leq 30$.
 M = DENOMINATOR ORDER N = NUMERATOR ORDER

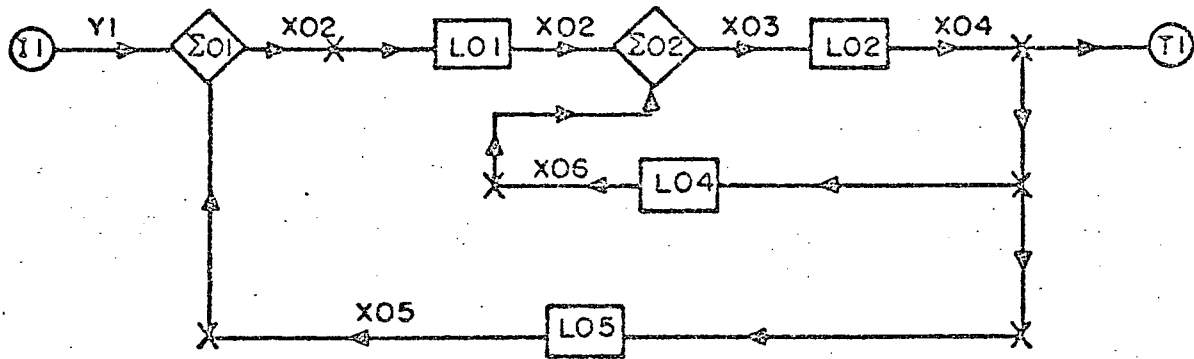
-Figure 4 Display Presented During EQUATION MODE

with all the information about the model which will be required for future operations. The program also monitors the model for errors and provides appropriate messages if any are detected. When the complete mathematical model for each element in the block diagram has been entered, the program passes to the DATA MODE.

The DATA MODE may be entered from either the EQUATION MODE or the MASTER SELECTION MODE. Entrance from the EQUATION MODE follows the normal procedure of building up the mathematical model and then specifying the values of the parameters. Entrance to the DATA MODE from the MASTER SELECTION MODE would be used when the engineer desired to modify the parameter values without altering the structure or mathematical description of his system. Figure 5 illustrates the display presented during the DATA MODE. The block diagram of the system is retained on the screen for reference. The parameter names are presented and the engineer is permitted to enter numerical values for the parameters. The entered values are displayed on the screen for verification. Upon completion of data entry the program passes to the MASTER SELECTION MODE where the engineer may specify the next operation he wishes to perform.

If the engineer selects the SIMULATION MODE the program retrieves all the required information about the block diagram, the individual block models and the parameter data and forms a digital simulation program. The engineer is asked to provide values for a few simulation parameters and the simulation

DATA



TAU = 1.5000000E-1
K01 = -3.1250000E-1
ENTER VALUE OF DISPLAYED CONSTANT

Figure 5 Display Presented During DATA MODE

program is executed. Figure 6 illustrates the display presented during the SIMULATION MODE. The message BUSY PROCESSING is displayed during the execution of the simulation program to assure the engineer that the program is operating. The program also provides a running display of the integration time and the total time the program has been specified to run. This display provides information on the progress of the program. If the program is not progressing satisfactorily the engineer may terminate execution of the simulation program. Upon completion of the simulation the program returns to the MASTER SELECTION MODE. During the simulation new plots were generated and their numbers are listed under the NEW PLOTS heading. These plots may be viewed by entering the PLOT MODE by touching the light pen to the number of the plot to be displayed. The form of the display presented in the PLOT MODE will be discussed following the discussion of the FREQUENCY RESPONSE MODE and the ROOT LOCUS MODE.

If the engineer selects the FREQUENCY RESPONSE MODE the program first checks to see that the block diagram does not include any nonlinear blocks. The program then presents a display of the block diagram and asks the engineer to specify where he wishes to break the loop (the location of the proposed compensator). The program then performs a block diagram reduction to obtain the open loop transfer function with the system opened at the specified point. The frequency response of this open loop transfer function is then obtained. The

SIMULATION

BUSY PROCESSING

T = 5.30000E+00
TFIR = 1.00000E+01

Figure.6 Display Presented During SIMULATION MODE

results of the frequency response are prepared for plotting in Bode, Nyquist and gain-phase form and the program returns to the MASTER SELECTION MODE. These plots may be viewed by entering the PLOT MODE by touching the light pen to the number under the NEW PLOTS heading of the plot to be displayed.

If the engineer selects the ROOT LOCUS MODE the program checks to see that the block diagram does not include any nonlinear blocks. The engineer is requested to specify the parameter which he wants to be varied to obtain the root locus. He is also asked to provide the range of values this parameter is to take. The program then repeatedly increments the specified parameter, performs a block diagram reduction to obtain the characteristic equation, and determines the roots of the characteristic equation. When the specified range of the root locus parameter has been covered, the root location data is prepared for plotting and the program progresses to the PLOT MODE where the results are displayed.

The PLOT MODE may be entered from the ROOT LOCUS MODE, from the PLOT TITLES MODE or from the MASTER SELECTION MODE. The form of the plot displayed depends on the analysis which generated the plot. Figures 7, 8, and 9 illustrate the form of the plots obtained from the SIMULATION MODE, the FREQUENCY RESPONSE MODE and the ROOT LOCUS MODE. Figure 7 shows that in the SIMULATION MODE the response of a system variable connected to a terminator block is plotted versus time. In addition to the plot the program presents the RESCALE, REDUCE,

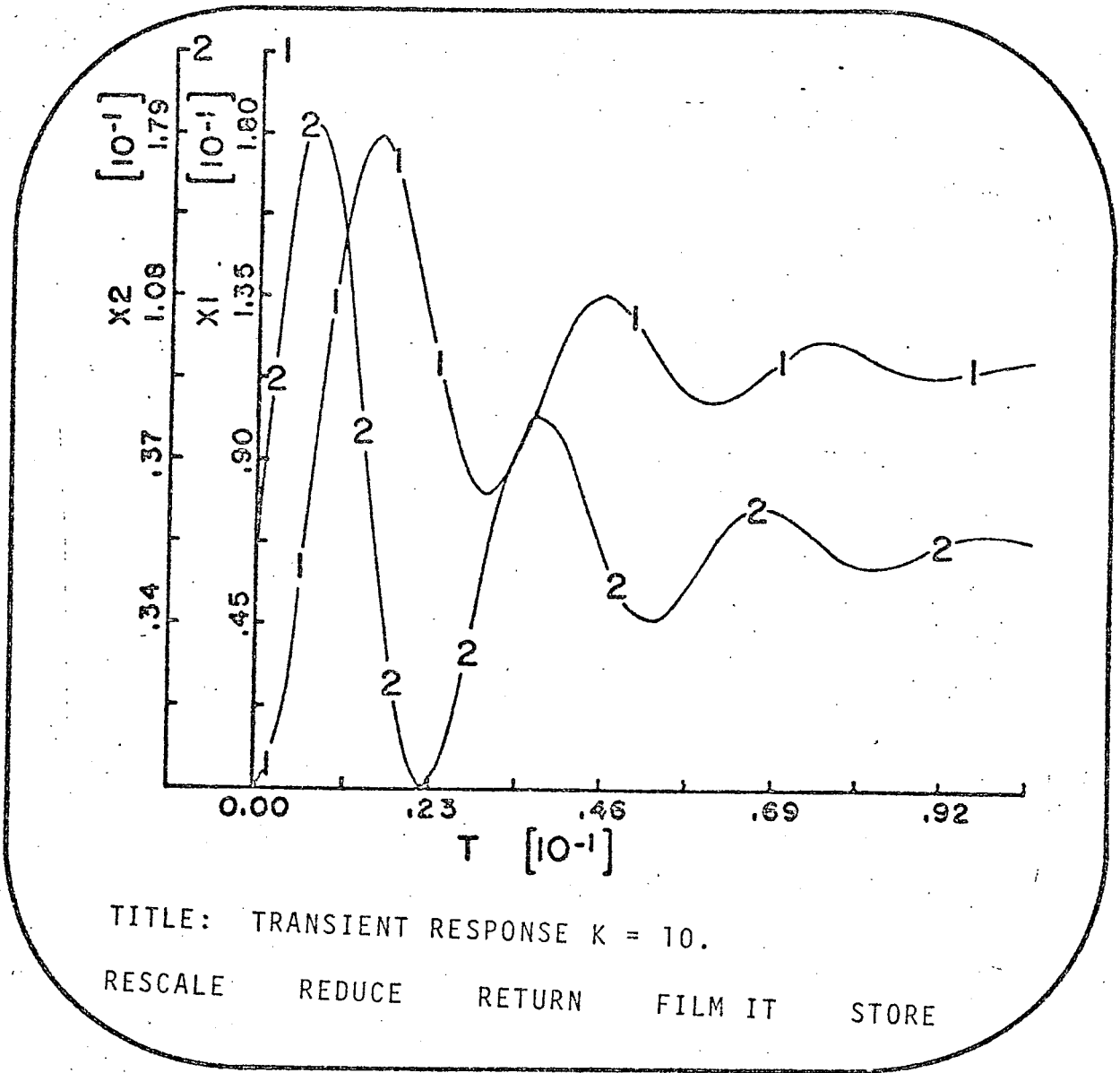


Figure 7 Simulation Response Plot

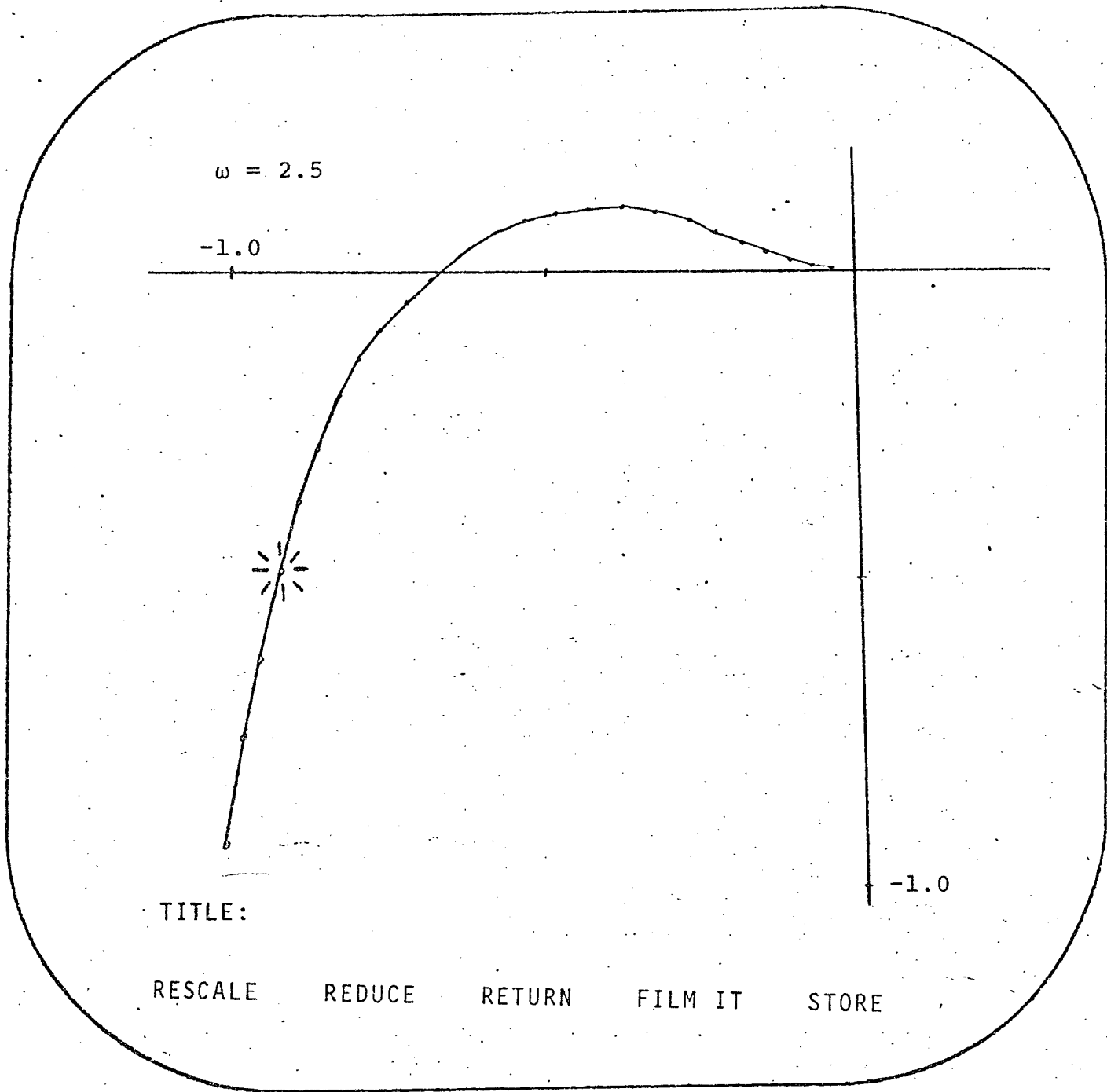
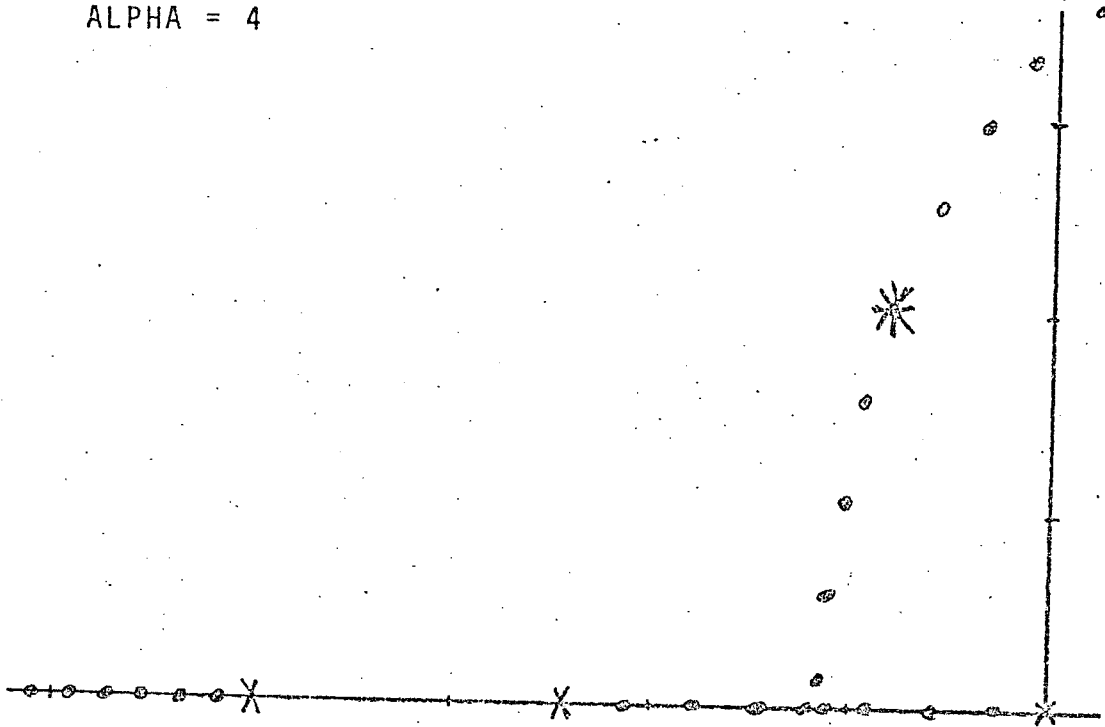


Figure 8 Frequency Response (Nyquist) Plot

ALPHA = 4



TITLE: ROOT LOCUS ALPHA = 1 TO 6

RESCALE REDUCE RETURN FILM IT STORE

Figure 9 CRT Presentation in PLOT MODE

RETURN, FILM IT, and STORE commands. The RESCALE and REDUCE commands are used to change the coordinates against which the plots are displayed. The RETURN command is used to return to the MASTER SELECTION MODE. The FILM IT command causes a microfilm copy of the display to be made. The STORE command is used to place the plot on the stored plot file so it won't be lost when the engineer selects a mode in which a new set of NEW PLOTS is generated.

Figure 8 illustrates the form of the plots generated by the FREQUENCY RESPONSE MODE. In this example the Nyquist plot of the open loop transfer function is illustrated. The tracking cross rests on one of the data points and the value of frequency indicated in the upper left corner of the display corresponds to the frequency at the point marked by the tracking cross. The tracking cross may be moved to determine the value of frequency at other points on the plot. The commands across the bottom of the display have the same function explained above.

Figure 9 illustrates the form of the plots generated by the ROOT LOCUS MODE. In addition to the plot and the control commands across the bottom of the display the name of the locus parameter and the value corresponding to the tracking cross position are presented. As before the tracking cross may be moved to determine values corresponding to other points on the plot. A space is provided at the bottom of all plots where the engineer may enter an identifying title. Titles are also

placed in the plot titles file and may be viewed collectively in the PLOT TITLES MODE.

The PLOT TITLES MODE may be entered from the MASTER SELECTION MODE. Figure 10 illustrates the display in the PLOT TITLES MODE. Titles to all "stored" and "new" plots are displayed adjacent to the plot identification numbers. If a title has not been entered for a particular plot the identification number will appear, but no title. The engineer may return to the MASTER SELECTION MODE by touching the light pen to the RETURN command or he may view a plot in the PLOT MODE by touching the light pen to the corresponding plot number.

Plots may be deleted while in the MASTER SELECTION MODE by taking consecutive hits on DELETE PLOTS and on the plot number.

The WRITE TAPE MODE provides the capability of writing all files onto a tape for future use. The READ TAPE MODE provides the capability of reading files previously stored on tape into the active program files. These provisions make it possible to describe a system and perform a partial analysis of it and return at a future time to complete the analysis without having to re-enter the entire description of the system.

A more detailed description of the operation of the programs in each mode is given in the attached appendices.

III. CONCLUSIONS

An interactive and interlinked set of programs has been developed which provides the capability to perform

PLOT TITLES

STORED

- 1 ROOT LOCUS $K = 0$ to 50 ..
- 2 LINEAR SYSTEM X AND \dot{X} VS T
- 3 ROOT LOCUS $\text{ALPHA} = 1$ TO 6
- 4
- 5 FREQUENCY RESPONSE NYQUIST $\text{ALPHA} = 5$ $K = 25$
- 6 NONLINEAR SPRING X VS T
- 7

NEW

- 1
 - 2 NONLINEAR SPRING X VS T
 - 3
- RETURN

Figure 10 CRT Presentation in PLOT
TITLES MODE

computer aided design of feedback control systems. These programs give a control systems design engineer an extremely versatile and powerful tool. In developing and working with the programs many things have been learned about how a design engineer may interact effectively with a computer. It was learned, for instance, that it is necessary to provide a large number of information cues to guide the engineer through his task, but the cues must be kept short and simple. It was learned that everything that is entered must be displayed for verification. It was learned that a great deal of flexibility in changing and correcting the model must be provided and that the engineer cannot be permitted to make errors that are uncorrectable without starting over. It was learned that it is necessary to inform the engineer of the status of his program if a period of computation more than a few seconds duration is encountered. It is helpful to provide the capability to interrupt the program during these long computation periods if the engineer suspects the computation is not proceeding properly. Finally, it was learned that both the light pen and the keyboard are effective communication devices. A new user is more at ease with the light pen, but some operations are not easily adapted to the light pen's capabilities.

In addition to the things learned about building an interactive system several new and significant technical capabilities were developed. The technique of automatically reducing a general block diagram is felt to be the most

significant technical development. The technique of automatically reducing a transfer operator to a set of nested integrals for digital simulation is also interesting. The hierarchical structure of the interlinked programs and the extremely complex data structure which resulted, though not unique, certainly provided an interesting and challenging application.

Finally, the project provided a very stimulating and rewarding educational experience for many students and several faculty members. Besides the opportunity to work on the project, the opportunity to interact with engineers at the Manned Spacecraft Center was most satisfying.

APPENDIX A

PROPOSAL

A copy of the original proposal by which the NASA Contract was secured for this project is included as Appendix A for reference purposes.



THE UNIVERSITY OF TEXAS

COLLEGE OF ENGINEERING

AUSTIN, TEXAS 78712

Department of Mechanical Engineering
512—GR 1-1501

Code Y
Mr. Kirby McCollum
Office of University Affairs
NASA Headquarters
Washington, D. C.

Dear Mr. McCollum:

The objectives of the National Aeronautics and Space Administration include:

- (1) The expansion of human knowledge of phenomena in the atmosphere and space;
- (2) The improvement of the usefulness, performance, speed, safety, and efficiency of aeronautical and space vehicles; and
- (3) The development and operation of vehicles capable of carrying instruments, equipment, supplies, and living organisms through space.

To accomplish these objectives NASA has developed and continues to develop a large number of missile, space vehicle and related systems. Feedback control systems constitute an important, frequently a critical, part of each of these projects. To design these control systems engineers at NASA and elsewhere should utilize the most powerful techniques available. This proposal describes

Mr. Kirby McCollum

Page 2

a program to provide engineers with an extremely powerful tool for the design of feedback control systems.

This project has been discussed informally with Dr. Kenneth J. Cox of the Guidance and Control Division of the Manned Spacecraft Center in Houston, Texas, and he has expressed an interest in it.

It is estimated that it will require three years to complete and fully document this project in the university environment. I would be happy to answer any questions concerning this proposal or to discuss it with any of your officials.

Sincerely yours,

J. Garth Thompson

JGT/ch

Enclosure

cc: Kenneth J. Cox

PROPOSAL

to the

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

for research in

UTILIZATION OF A CRT DISPLAY-LIGHT PEN
IN THE DESIGN OF FEEDBACK CONTROL SYSTEMS

Principal Investigator: J. Garth Thompson, Assistant Professor
Department of Mechanical Engineering
The University of Texas
Austin, Texas 78712
512 GR1-1333

Budget Requested: Amount \$78,740.
Duration 36 months
Starting Date January 1968

Submitted:

Original Signed By

J. Garth Thompson

Approved:

Original Signed By

W. R. Upthegrove, Chairman
Department of Mechanical Engineering

ORIGINAL SIGNED BY
JENS M. JACOBSEN

Jens Jacobsen, Executive Director
Office of Sponsored Projects

I certify that the distribution of costs between the direct and indirect categories as shown in the proposal conforms to the usual accounting practices of the institution for all Federally supported or sponsored research, and to the distribution used by the cognizant Federal audit agency.

Original Signed By

James H. Colvin, Business Manager
Main University

ABSTRACT

The digital computer is used in many aspects of feedback control system design, from the computation of data for Nyquist, Bode and root locus plots to the dynamic simulation of systems. With the development of the CRT (cathode ray tube) display - light pen system, the digital computer can become a much more powerful tool for control system design. The usefulness of this tool depends on the development of a set of programs by which the controls engineer may communicate with the computer. These programs should permit the controls engineer to describe his system to the computer using standard control terminology. The computer programs should then provide, on the CRT-display, any one of a variety of control system design plots including Nyquist, Bode and root locus plots and transient response plots. The engineer should be able to communicate easily a change in the value of any system parameter and to observe the change in the plot on the CRT-display.

This proposal solicites funds to support a project to develop such a set of programs. A more detailed description of the programs to be developed is given in the body of the proposal.

PROPOSED BUDGET - THREE YEARS

	<u>First Year</u>	<u>Second Year</u>	<u>Third Year</u>
I. Salaries			
A. Principal Investigator ½ time - 3 months	\$ 2,000	\$ 2,250	\$ 2,500
B. Research Assistants			
#1 - ½ time 9 months full time 3 months	4,125	4,313	4,500
#2 - ½ time 9 months full time 3 months	3,600	4,125	4,313
#3 - ½ time 9 months full time 3 months	3,600	3,600	3,600
#4 - ½ time 9 months	1,305	1,305	1,305
B. Stenographic, Clerical and Key-Punch Assistant ½ time 12 months	1,830	1,830	1,830
TOTAL SALARIES	\$16,460	\$17,423	\$18,048
II. Fringe Benefits	725	767	794
III. Expendable Supplies (Computer Cards, Pulp Charges, Etc.)	700	700	700
IV. Domestic Travel	400	400	400
V. Other (Publication and Miscellaneous Costs)	200	200	200
TOTAL DIRECT COSTS	\$18,485	\$19,490	\$20,142

	<u>First Year</u>	<u>Second Year</u>	<u>Third Year</u>
VI. Indirect Costs			
NASA Contribution @ 20% of Total Direct Costs	\$ 3,697	\$ 3,898	\$ 4,028
The University of Texas Contribu- tion @ 40.5% of Total Salaries minus NASA Contributions	2,969	3,158	3,281
VII. Computer Time	3,000	3,000	3,000
Central Processor @ \$200./hr. Peripheral Processor @ \$30./hr.			
TOTAL COST	\$28,151	\$29,546	\$30,451
TOTAL U of T CONTRIBUTION	\$ 2,969	\$ 3,158	\$ 3,281
TOTAL NASA CONTRIBUTION	\$25,182	\$26,388	\$27,170
TOTAL NASA CONTRIBUTION FOR THREE YEARS			<u>\$78,740</u>

BUDGET NOTES

- 1 Salaries paid from Government grant funds at The University of Texas conform to the rates approved by the Board of Regents for salaries paid from regular University Funds.
- 2 Federal Social Security (4.4%); Workmen's Compensation Insurance (0.50% of applicable salaries), as may be required by University regulations under the University's self-insuring plan. Under University accounting procedures these items are direct costs.
- 3 The distribution of costs between the direct and indirect categories as shown in the proposal conforms to the usual accounting practices of the institution for all Federally supported or sponsored research, and to the distribution used by the cognizant Federal audit agency.
- 4 The current provisional overhead paid by Government agencies is 40.5% of Salaries and Wages. In this proposal the NASA is requested to pay overhead at the rate of 20% of costs. Thus, the difference between 40.5% of Salaries and Wages and 20% of costs represents a University contribution.

BIOGRAPHICAL SKETCH

August 1967

1. J. Garth Thompson, Assistant Professor
Department of Mechanical Engineering
The University of Texas
Austin, Texas 78712

2. Birth: August 15, 1935; Logan, Utah

3. Education:

B.E.S., M.E., Brigham Young University, June, 1960

M.S., M.E., Purdue University, January, 1962

Ph.D., Purdue University, January, 1967

4. Experience and Employment:

Assistant Professor of Mechanical Engineering, The University of Texas, September, 1966 to present.

Instructor of Mechanical Engineering, Purdue University, September, 1962 to September, 1966.

Research and Design of Space Craft Attitude Control Systems, Space Technology Laboratories, Redondo Beach, California, September, 1961 to September, 1962.

Research Assistant, Automatic Control Group, Department of Mechanical Engineering, Purdue University, September, 1960 to September, 1961.

5. Fields of Specialization:

Design of Dynamic Systems, Nonlinear Systems, Feedback Control, Optimization, Analog and Digital Simulation.

6. Memberships in Professional Societies:

The American Society of Mechanical Engineers
Simulation Councils, Inc.

Sigma Xi

Pi Tau Sigma

7. Publications:

"Introduction to Time Optimal Control of Stationary Linear Systems," Automatica, 1963, Vol. 1, pp. 177-205.

"Time Optimal, Discontinuous Control," Master's Thesis, Purdue University, January, 1962.

"A Method for Modeling and Compensating Nonlinear Dynamic Systems," Ph.D. Thesis, Purdue University, January, 1967.

"Modeling and Compensation of Nonlinear Systems using Sensitivity Analysis," 1967 Joint Automatic Control Conference Preprints and Journal of Basic Engineering (ASME).

8. Invited Lectures:

"An Analytical Design Technique for Feedback Control Systems," presented at The University of Texas, March 18, 1966.

"Modeling and Compensation of Nonlinear Systems Using Sensitivity Analysis," presented at the 1967 Joint Automatic Control Conference, June 29, 1967.

DESCRIPTION OF PROPOSED PROJECT

I. Introduction

The design of feedback control systems is usually carried out in two phases. In the first phase the system is considered to be linear and the form of the system and rough values of the system parameters to give adequate performance are determined using classical linear control theory. In the second phase of the design the model of the system is extended to include the significant nonlinearities in the system and refined values of system parameters to give satisfactory performance are determined using system simulations and nonlinear control theory. Analog, digital, and hybrid computers are used extensively in the design of feedback control systems. For example, suitable digital computer routines for calculating standard root locus data have been in use for several years. The use of the analog computer and more recently the digital and hybrid computers for system simulation is an indispensable part of the design of nonlinear systems.

Current developments in digital computer technology suggest the possibilities of new design aids if appropriate programs for communication with the computer are developed. In particular, the availability of the CRT (cathode-ray-tube) display-light pen combination with the digital computer should permit the development of extremely powerful tools which have not been possible in the past. For example, with the best programs presently available a controls engineer will typically spend several weeks running and analyzing root locus

plots to determine an acceptable set of values of the system parameters. Even then he may have obtained only a superficial knowledge of the effect of varying parameters other than the system gain. To achieve this result the engineer must: obtain and analyze a set of plots, change the system parameters in the direction indicated by the analysis, and reiterate the process with the new parameter values. If the number of parameters to be considered is large or the system complicated, the number of times this process must be repeated to obtain satisfactory results, including a minimal understanding of the effect of varying each parameter, may be quite large. The total elapsed time is frequently excessive.

On the other hand, a controls engineer equipped with a CRT display-light pen system and appropriate programs may be able to perform a more complete analysis in a matter of hours. A better understanding about the effect of varying system parameters may be obtained with this system in a few hours than is now obtained in a matter of weeks. The increase in speed and understanding depends not only on the availability of the CRT display-light pen system but also on the availability of an appropriate set of computer programs.

This proposal describes a project to develop these programs. The task to be accomplished by this project may be divided into the following six parts.

1. Develop the means of easily communicating the system structure to the computer utilizing conventional control system terminology.

2. Develop a program to determine and display the locus of the roots of the system characteristic equation as a function of any parameter in the system.
3. Develop a program to determine and display a Bode plot for either the open loop or closed loop transfer function.
4. Develop a program to determine and display a Nyquist plot for either the open loop or closed loop transfer function.
5. Develop a program to perform a transient response simulation of the system and display the results on the CRT.
6. Develop a supervisory program capable of communicating with all of the above programs and with the control engineer at the CRT.

Each of these parts is described in more detail in the following sections of this proposal.

II. Communicating the System Structure to the Computer

It is recognized to begin with that the ultimate usefulness of the programs described in this proposal will depend on the ease with which the engineer can communicate his system description to the computer and can obtain results from it. The communication should be in a form already familiar to the majority of prospective users. The two ways in which most controls engineers describe their systems are block diagrams and sets of differential equations. There are three ways that the engineer might communicate his system description to the computer. They are by reading cards in the card reader, by typing characters on the keyboard or by means of the light pen. The card reader or keyboard would be used to input the system description in the form of a set of differential equations. The light pen and keyboard would be used together to input the system description in the form of a block diagram.

The use of the card reader or keyboard to input the system description in the form of a set of differential equations does not require the development of any new techniques. For linear systems the coefficients may be input as an array of variables while nonlinear systems may be described by a set of Fortran-like statements.

The use of the light pen and keyboard to input a system description in block diagram form will require the development of techniques not presently in common usage. The following approach is planned. The engineer will be presented with an assortment of "standard elements" displayed across the bottom of the display tube. These might include summers,

linear transfer elements, nonlinear transfer elements, connecting links and delete. The engineer would "pick up" an element and "place" it at some point on the display tube. The "pick up" and "place" would be done by touching the light pen to the "standard element" displayed at the bottom of the tube and opening the shutter momentarily. Then, the pen is placed at the point where the element is to be placed and the shutter is again opened momentarily. The appropriate standard element will appear on the display tube at the tip of the light pen. After two or more elements are placed inter-connections may be made by "picking up" a connecting link and "placing" it by indicating the element from which it originates and into which it terminates. Any element could be removed from the block diagram by "picking up" delete and "placing" it on the element to be removed. Each element as it is placed in the diagram is identified, the summers by S1, S2, etc., the linear transfer elements by L1, L2, etc., and the nonlinear transfer elements by N1, N2, etc. as each is placed. The transfer characteristics of the linear and nonlinear transfer elements will be communicated to the system through the keyboard after the block diagram form is completed. At any time the block diagram may be recalled to the display tube and altered by addition or deletion of elements.

After the block diagram form is completed and the transfer characteristics of each element entered the computer program, on command from the engineer, will translate the system description into a set of machine codes which will be used in the subsequent analysis programs.

III. Root Locus Plotting Routine

One of the most useful tools in the design of linear control systems is a root locus plot. In its original form the root locus method involved plotting the loci of the roots of the system characteristic equation as the open loop gain varied over a range of values; usually zero to infinity. Conceptually there is no reason to limit the method to studying the effect of varying the open loop gain. The loci of the roots of the characteristic equation as any system parameter varies over a range of values would be equally interesting. The approach taken in the development of this routine will be as general as possible. It is anticipated that a technique can be developed whereby the root loci may be plotted as a function of any open or closed loop system parameter. The routine should allow the user to change from one locus parameter to another locus parameter easily.

There are other features which should be included in this routine. The scale of either axis of the locus should be easily changed with the scale of the other axis automatically adjusted. The engineer should be able to look at any portion of the root plane automatically scaled to fill the face of the display tube. The engineer should be able to determine the value of the locus parameter at any point along the locus by simply touching the tip of the light pen to the point in question. The location of the corresponding roots on the other loci for that value of the locus parameter should be indicated.

Another feature which would be desirable and which it

is anticipated might be developed is an indication of the sensitivity of a particular root location as a function of a parameter other than the locus parameter. For example, a particular system might include the parameters x and y . If the loci of roots versus x were plotted and a particular root location corresponding to a particular value of x were being considered, it would be desirable to know in which direction and by how much these roots would move if the parameter y were varied. These sensitivities to changes in y might be indicated on the display tube by vectors originating on the root location corresponding to the value of x indicated and directed in the directions the roots would move for increasing y . The lengths of the vectors would indicate the amounts by which each root would move for a prescribed increase in y .

The techniques for the root sensitivity feature are not fully developed. Research toward this end will be carried on as part of this project.

IV. Bode Plotting Routine

Another very useful tool for the design of linear control systems is the Bode plot. The standard Bode plot consists of the plot of the frequency response magnitude and phase angle of the open loop transfer function plotted with respect to frequency. Usually the magnitude plot is made on log-log coordinates and the phase angle plot is made on semi-log coordinates. The following features should be included in this routine:

1. The ability to easily change the segment of the frequency coordinate with automatic adjustment of the magnitude and phase angle coordinates to utilize the entire face of the display tube.
2. The ability to read off magnitude, phase angle and frequency values by indicating the point on the plot in question with the light pen.
3. The ability to change any system parameter and observe the resulting change in the Bode plot.
4. In addition to the usual Bode plot this routine should include the capability of plotting the closed loop frequency response magnitude and phase angle curves as a function of frequency. These plots should have the same features as mentioned for the open loop Bode plots.

V: Nyquist Plotting Routine

The third linear system design tool to be included in the project is a Nyquist plotting routine. This routine uses essentially the same data as the Bode plotting routine, but displays the results on a polar plot. Features to be included in this routine are as follows:

1. The ability to easily change the scale of the plot to observe a particular portion of the plot in greater detail.
2. The ability to read off values of frequency, magnitude and phase angle corresponding to points on the plot by placing the tip of the light pen at the point in question.
3. The ability to change system parameters, particularly system gain, and observe the resulting change in the Nyquist plot.
4. The ability to superimpose M-circles (contours of constant closed loop magnitude) on the plots.
5. In addition to the ability to plot the usual Nyquist diagrams this routine should include the capability of making polar plots of the closed loop frequency response. This feature should include the capabilities of changing the scale of the plot, reading off the coordinate values and changing the values of system parameters as mentioned above.

VI. Transient Response Simulation Routine

The fourth design tool to be included in the project is a transient response simulator. This routine will be built around MIMIC, a digital simulator program which was developed at Wright-Patterson Air Force Base. A well-developed version of the MIMIC program is presently available on the computer on which this project will be conducted. The features which must be added to the MIMIC program to make it a part of this project are as follows:

1. MIMIC must be modified to communicate with the engineer at the CRT display by presenting results on the display tube and receiving data from the keyboard and light pen.
2. MIMIC must be able to accept the system description not only from the card reader as is now possible, but also from the block diagram system structure as described in Section II of this proposal. This feature should include the ability to describe nonlinear functions in the block diagram and have them properly translated into the MIMIC program.
3. The response should be plotted on the display tube either as a function of time or as a phase plane plot.
4. The input function and system parameter values should be easily changed without requiring re-compilation of the entire program.

VII. Supervisory Program

In order to obtain the maximum use from the routines described in this proposal it will be important to unite them all under the direction of a main supervisory program. This program will take care of the details of keeping track of the storage location of parameter names and values and communicating the required information from one routine to the other. This feature will make it possible for the engineer to move from a root locus study, to a Nyquist study, to a transient response simulation, etc. without re-describing the system to the computer. This program should be written so that each of the routines under its control could also be called into use for batch processing jobs which require their capabilities.

VIII. Facilities Available for the Project

All of the principal facilities required for this project are available at The University of Texas Computation Center. The main computer of this center is a Control Data Corporation 6600 computer. This computer is used both as a batch processing system and as a time-sharing system with several kinds of peripheral devices. The CRT display-light pen system is one of the peripheral devices which communicates with the CDC 6600 in a time sharing fashion. The same peripheral controller which operates the display tube-light pen system also controls a microfilm recorder. This is a very desirable arrangement because it facilitates obtaining hard copies of any of the material developed in the process of the system design. Other peripheral devices which are available to support the project as needed include card readers, line printers, magnetic tape units, disk storage units, time sharing teletype terminals, a Calcomp plotter and in the near future remote computers, one of which will be a hybrid computer.

IX. Tentative Time Schedule and General Approach

It is estimated that the project described herein will require three years to complete and document. A tentative time schedule for accomplishing the various tasks follows:

1. The basic techniques for communicating the system structure in block diagram form with the light pen should be developed and tested the first year. Implementation and refinement will be done in the second and third years.
2. A basic root locus routine should be implemented and the techniques for extending the routine to the more general applications should be developed the first year. The implementation of the general application and the sensitivity feature will be done in the second and third years.
3. The Bode plotting routine should be implemented the first year.
4. The Nyquist plotting routine should be implemented the first year.
5. The basic transient response simulation routine should be implemented the first year but improvements will be made the second year.
6. Some of the basic concepts of the supervisory program will be developed the first year, but its implementation will take place during the second and third years.

At the end of the second year a two-week short course on the design of feedback control systems will be scheduled and

built around the programs developed in this project. This will provide an evaluation of the programs and will result in suggestions for their improvements. The programs will be completed and documented during the third year. At the end of the third year another two-week short course will be conducted at which time the completed programs should be ready for general distribution.

The general approach taken will be to write as much of the computer codes as possible in Fortran so that they will be readily usable in a majority of computer installations. All routines will be developed so that they can be used for the time sharing application described and for producing plots in a batch mode as well.

The end result of the project should be a set of very powerful programs to aid the control system designer in gaining insight into the operation of his system and in selecting a good set of values for the parameters of his system.

ADMINISTRATIVE STATEMENT

Additional information pertaining to the technical aspects of this proposal may be obtained from the Principal Investigator. Non-technical administrative and contractual matters should be referred to the University's Office of Sponsored Projects, The University of Texas, Austin, Texas 78712; Telephone Area Code 512, Greenwood 1-1353.

APPENDIX B

HARDWARE

The hardware used in this project includes the Control Data Corporation 6600 computer system and the 250 display system. The 6600 computer system includes the central processor, ten peripheral processors, a large central memory, an extended core memory, and a disk storage unit. The 250 display system includes the display controller, the display console, and a microfilm recorder.

The Control Data Corporation 6600 computer system is a large scale, solid state, general purpose digital computer. Advanced design techniques have been incorporated in this system to provide for extremely fast solutions to data processing, scientific and control center problems. The system also is capable of multi-processing, time-sharing, and management information applications. The 6600 computer system has eleven independent computers. Ten of these are peripheral and control processors which are used primarily for high speed information transfer in and out of the system and to provide operator control. Each computer has a separate memory and can execute programs independently of each other or the central processor. The eleventh computer,

the central processor, is an extremely high speed arithmetic device. The common element of the peripheral and control processors and the central processor is a large (131,072₁₀, sixty-bit word) central memory.

The ten peripheral and control processors have twelve-bit 4096 word random-access memories. They are identical and operate independently and simultaneously as stored program computers. Therefore, ten programs may be running at the same time. Also, a combination of processors could be involved in one problem. The solution of the problem might require a number of input or output tasks plus the use of central memory. Since the peripheral processors perform system control and input-output, the central processor can perform high-speed computations while the peripheral processors do slower input-output and supervisory operations.

In order to provide for the multi-processing, time sharing service required of the 6600 system, a large, high speed extended core storage unit and four very large magnetic disk drive units are used. The extended core storage unit consists of 500,000 (60-bit) words to and from which data may be transferred at the rate of 100 million characters per second. This capability permits

interactive and time sharing programs to be set aside while no computations are required for them, but they may be retrieved quickly when service is needed.

The disk storage unit consists of four magnetic disk storage units each with a capability of 84 million characters. The four units may be accessed simultaneously with an access time of 30 to 100 milliseconds per access. The disk units provide for large amounts of intermediate speed storage. They are used extensively by the programs described in the subsequent appendices.

The central processor consists of an arithmetic unit and a control unit. The arithmetic unit contains all the logic necessary to execute the arithmetic, manipulative and logical operations. The control unit directs the arithmetic operations and provides the interface between the arithmetic unit and central memory. It also performs instruction fetching, address preparation, memory protection, and data fetching and storing.

High speed in the central processor depends first on minimizing memory references and second on decreasing the number of sequentially executed instructions. The central processor contains twenty-four registers to lower central memory access requirements for arithmetic operands and results. These are divided into eight address registers, eight increment registers, and eight operand registers. The address and increment registers are eighteen bits in length, and the operand registers are sixty bits in length. An additional eight sixty-bit registers are provided to hold instructions. These registers limit the number of memory reads for repetitive instructions, especially in inner loops. Multiple banks of central memory are available to minimize memory reference time, due to the fact that references to different banks of memory may be handled without delay.

In order to decrease the number of sequentially executed instructions, the 6600 is provided with ten arithmetic units and a reservation control. The reservation control keeps a running account of the address, increment, and operand registers and of the arithmetic units in order to preserve the sequence that is established at the time each instruction is issued. Only operations which depend on previous steps prevent the issuing of instructions, and then only if the steps are incomplete. The ten units are the branch unit, the boolean unit, the shift unit, the add

unit, the multiply unit, the divide unit, the long add unit, and the increment unit. There are two multiply and increment units.

Just as the central processor has the peripheral processors to perform input and output, the peripheral processors have electronic devices which perform communication with basic hardware such as line printers, card punches, tape drives and display devices. These electronic devices are called peripheral controllers. The graphics terminal has a controller referred to as the display controller. The display controller is comprised of two parts, a logic unit and a memory unit.

The logic unit contains two networks. They are registers for manipulation and storage of data, and digital to analog circuitry for beam positioning and symbol formation. The registers accept incoming data in twelve-bit bytes and assemble it into twenty-four-bit words in the buffer register. These words are then transferred to memory or directly to the display registers. The digital to analog circuitry performs the conversion of digital codes in the display registers into X and Y beam movements for symbols and vectors.

The memory unit contains an expandable magnetic core memory. The present size of the memory is 4096 words. Each word contains twenty-four data bits and two parity bits.

Words arrive and leave memory through the memory register which is loaded directly from the buffer register in the logic unit. Memory is not used when the display equipment is in the direct mode with data going directly from the input circuits to the display circuits. When the equipment is in the normal display operation, the memory register is read into the display register. Each instruction in memory is executed fifty times per second.

A major element of the Control Data Corporation 250 display system is the display console which consists of a cathode ray tube (crt), a light pen, and a typewriter keyboard. The light pen is connected to the front panel of the display console, and enables the user to specify to the computer individual symbols on the crt. The light pen is able to detect the light from the symbols through an optic detection system at the instant the symbols are being displayed. The light pen detects the initial phosphor illumination and signals the display controller. Since the display controller just completed execution of the display instruction that caused the phosphor illumination detected by the light pen, it can notify the computer and indicate which display instruction was involved. With appropriate programming the machine is thus able to correlate light pen "hits" with the instruction generating the displayed information. Any action taken due to the light pen hit is determined by the user's program.

The major component of the display console is the crt for user monitoring and control of displays. The console contains a nineteen inch crt with a square display area of twelve by twelve inches. The display area represents the first quadrant of a Cartesian coordinate system as seen by the user, with 1024 raster units in each direction. Allowable coordinate values lie between zero and 1023 (1777_8). The position of symbols is controlled by twenty bits, ten bits for horizontal (X) and ten bits for vertical (Y), of the twenty-four bit display controller word. The remaining four bits are used for the display controller operation codes.

The alphanumeric keyboard is a standard typewriter arrangement attached to the display console shelf. Depression of a key causes an interrupt signal to be sent to the computer and the symbol code to be stored in the display controller memory.

The system also includes a microfilm recorder which is a physically separate unit connected to the logic unit of the display controller. It is a separate crt presentation and recording camera combination which produces high-speed recordings of displays directly on thirty-five millimeter microfilm. The camera holds magazines with a film capacity of up to one thousand feet. The recorder allows direct

on-line recording of up to four-hundred-thousand symbols per second and up to thirty frames of data per second.

This equipment constitutes the system being used to meet the objectives of the contract. The software and an explanation of its operation will be presented in Appendix C.

APPENDIX C

SYSTEM SOFTWARE

In order to provide an interface between the display system hardware described in the previous appendix and a program written by the user, a general software support package has been written by the computation center staff at The University of Texas. The package consists of two programs called DDI and CRT. DDI is a peripheral processor program which handles the communication between the display hardware and CRT. CRT is a central processor program and is used as a subroutine to the user's program. CRT with the user's program occupies central memory continuously but executes only when called by DDI or the user's program.

Information flows from the user's program to the display system hardware in four steps. First, the user's program calls CRT when it needs to display something. Second, CRT creates a display controller instruction from the information sent by the user's program and transmits it to DDI. Third, DDI receives the instruction, checks it, and then sends it to the display controller. Fourth, the display controller stores the display instruction in its memory and begins to execute the instruction fifty times per second to produce the display on the crt.

The information flow from the user at the console to his program in the central processor is more complex and requires seven steps. First, the user may point to some displayed information with the light pen. Second, the light pen signals the display controller which records the memory address of the display instruction that produced the light seen by the light pen. Third, the display controller sets a signal indicating that it has received a light pen hit. Fourth, when DDI discovers that a light pen hit has occurred, it reads the address of the display instruction from the display controller memory (buffer) and assembles it with other information and stores it in central memory. Fifth, then DDI signals the monitor to start executing the central processor program if it is idle. If the program is already running, it will detect the light pen information sent by DDI. Sixth, after CRT notices the light pen hit, it calls a subroutine name LITEPEN to process the hit. Seven, LITEPEN is a user supplied subroutine which processes the hit as directed by the user. If in Step One the user signaled the display controller with a keyboard entry, the steps are essentially the same except that CRT calls an entry point within the user's program instead of calling LITEPEN. All of these tasks are performed automatically by DDI and CRT

except for deciding what to display and what to do when a light pen hit occurs.

The user's program and CRT communicate by standard FORTRAN calling sequences. The argument list is variable and determines how CRT will react to the call. The argument list may have from one to six arguments depending on what CRT is requested to do. The first argument is always a word containing a string of characters, left-justified with blank fill (i.e. a Hollerith constant or variable). A Hollerith constant is of the form nHxxx, where n is the number of characters (x's) following the Hollerith indicator (H). If the other arguments are present, they are words containing left-justified character strings packed ten characters per word with blank fill, integer variables, or integer constants.

The first call to CRT must be of the form

```
CALL CRT(6HBUFFER, IARRAY, LENGTH).
```

LENGTH is the number of words in the array, IARRAY, which is used by CRT and DDI as a circular buffer for storing light pen hits and keyboard characters before processing. The entry also initializes the display by calling the display driver DDI. If this is not the first call to CRT, then the job will be aborted.

Each of the following FORTRAN calls to CRT, cause only one display instruction to be generated. Each instruction sent to the display controller is stored in the next available buffer location.

The entry

```
CALL CRT(5HABASE, IX, IY)
```

causes the generation of a display instruction to move the beam of the crt to an absolute base position. When executed by the controller, this instruction moves the beam to raster position, (IX, IY). The beam is not visible during its motion. The entry

```
CALL CRT(5HRBASE, IDX, IDY)
```

is used to generate the display instruction to move the beam to a position relative to its current position. This moves the beam IDX raster units from its current X position and IDY raster units from its current Y position. CRT uses values of IX, IY, IDX, and IDY modulo 1024 (i.e. the remainder of the values divided by 1024). The hardware also uses modulo 1024 in positioning the beam. Therefore, if IX had a value of 1044, then the value sent to DDI would be IX modulo 1024 or twenty.

To display a single dot at raster position (IX, IY), the appropriate command is

```
CALL CRT(4HADOT, IX, IY).
```

The command

```
CALL CRT(4HRDOT, IDX, IDY)
```

displays a dot at a position relative to the current beam position. The dot is displayed IDX raster units from the beam's current X position and IDY units from its current Y position.

The entry

```
CALL CRT(5HAVECT, IX, IY)
```

draws a straight line from the current beam position to an absolute position (IX, IY). To draw a straight line from the current beam position to a point IDX units away in the X direction and IDY units in the Y direction, the command is

```
CALL CRT(5HRVECT, IDX, IDY).
```

The horizontal or vertical length of a straight line drawn by either command should not exceed 255 units. This restriction is caused by the display hardware which uses only the least significant eight bits of the IX and IY (or IDX and IDY) values sent by CRT. Therefore, the hardware uses the original values modulo 256.

The display controller hardware is capable of displaying characters, lines, and dots in several different "modes." Characters, lines and dots may be displayed brighter than normal or winking. Characters can be displayed in four sizes and may be displayed in italics or rotated.

The instruction is

```
CALL CRT(5HSETUP, MODE).
```

The instruction specifies the manner in which all the subsequent display instructions are to be executed. The display mode is in effect until the next SETUP command and is determined by MODE which is a character string, left-justified with blank fill. Each non-blank character of MODE has a meaning. The letter I means display all characters in italics; B means display characters, dots and lines brighter than normal or bold; O means rotate characters ninety degrees counterclockwise before displaying them or oriented; and W means blink characters, dots, and lines on and off in a mode called winking. T, S, M, and L are used to set the character size at tiny, small, medium or large, respectively. If MODE had a value of 4HWBOM, the display mode would be set to winking, bold, oriented, and medium.

Another command is available that is the same as the SETUP command, but in addition, it also displays one character at the current beam position. The entry is

```
CALL CRT(6HSYMSET, CHAR, MODE)
```

where MODE is optional and is the same as discussed with the SETUP command. The leftmost character of CHAR is displayed at the current beam position. If the mode argument is

not specified, then the display mode established by the last SETUP or SYMSET command is used.

The instruction

```
CALL CRT(7HSYMBOLS, CHARS)
```

causes the leftmost three characters of CHARS to be displayed, automatically spaced, and starting at the current beam position. The characters are displayed in the mode set by the last SETUP or SYMSET instruction. The beam moves horizontally to the right unless the mode is oriented and then the beam moves upward between the characters.

Besides the standard sixty-four symbol set, there exists an extended symbol set. The standard symbol set is shown in Table (C.1), and the extended symbol set is shown in Table (C.2). In order to display a character from the extended set, some symbol is used as an escape character. The up arrow (↑) is the extended character set escape character. If CHARS was 4HA↑BC then the symbols displayed would be AbC. The up arrow means that the extended symbol b, equivalent to B, is to be displayed instead of B itself. The equivalence is determined by observing which character in the extended table occupies the same position as the character from the standard character set.

:	1	2	3	4	5	6	7
8	9	0	=	≠	≤	%	[
SPACE	/	S	T	U	V	W	X
Y	Z]	,	(→	≡	∧
-	J	K	L	M	N	O	P
Q	R	V	S	*	↑	↓	>
+	A	B	C	D	E	F	G
H	I	<	.)	≥	¬	;

Table C.1
Standard Symbol Set

β	1	2	3	ψ	ρ	γ	ϕ
ω	α	Δ	δ	Σ	σ	μ	π
ϵ	←	s	t	u	v	w	x
γ	z	~	□	⊙	△	◇	▽
\pm	i	k	l	m	n	o	p
q	r	f	∂	e	Δ	-	√
θ	a	b	c	d	e	f	g
h	i	λ	Ψ	"	'	ω	?

Table C.2
Expanded Symbol Set

Each of the commands described above causes the generation of only one display instruction. The commands to be described in the following paragraphs may not generate any instructions or they may generate one or more instructions.

The entry

```
CALL CRT(5HERASE)
```

causes the entire memory of the display controller to be cleared, or in other words, the entire current display is erased. Only no-op display instructions are generated.

If only part of the display is to be erased, the entry is

```
CALL CRT(4HBKSP, N).
```

This causes the last N display instructions to be erased or replaced by no-op words. This is called a backspace command.

Since lines of only 255 raster units in the horizontal or vertical direction can be drawn with RVECT or AVECT commands, a command is available which draws a straight, solid line from point (IXA, IYA) to point (IXB, IYB) in the display area. The entry is

```
CALL CRT(4HLINE, IXA, IYA, IXB, IYB).
```

An ABASE instruction is used to move the beam to (IXA, IYA) Zero or more RVECT instructions are generated to draw the remaining line segments, except the last segment, which is drawn with an AVECT instruction to (IXB, IYB).

Because the SYMSET or SYMBOLS instructions may display only one or three characters, a command is included which displays a character string of arbitrary length starting at the current beam position and using the last established mode. The entry is

```
CALL CRT(4HTEXT, STRING(1), NCHAR,  
        NWORDS, NBLANKS, CONTROL).
```

STRING is considered to be NWORDS consecutive words long, and it may contain blanks between the characters. NBLANKS is the number of trailing blank characters from STRING to be included in the display and character count. If NCHARS is zero, then CRT determines the exact length of STRING, displays the characters, and returns the length determined in NCHARS. If NCHARS is not zero, then CRT displays that number of characters from STRING. CONTROL controls the generation of the display instructions. If CONTROL is non-zero, the instructions are generated. If CONTROL is zero, no instructions are generated but NCHARS is determined.

When the light pen is to be used for selection of displayed data, LITEPEN must have some means of keeping track of where display instructions are stored in the display controller memory. This information is necessary because the controller tells the central processor which instruction caused a light pen hit rather than the position of the displayed data. CRT sends information to LITEPEN through a labelled,

102 word COMMON block whose name is DD250. LITEPEN uses only the first word (NEXT) and the third word (IHIT) to interpret light pen hits. The display instructions are stored in the controller memory in the order they are generated. NEXT contains current information about the next available buffer location and is incremented by one each time an instruction is generated. LITEPEN can determine the first and last buffer addresses of a group of instructions by using the information supplied by NEXT. IHIT is used to send light pen hits to LITEPEN. For light pen hits on characters, lines, or dots, IHIT is positive and the buffer address that caused the hit is stored in the rightmost twelve bits of IHIT. Therefore, by keeping track of where a certain instruction or group of instructions are stored, LITEPEN can interpret light pen hits by comparing known addresses, supplied by NEXT, with the address of the hit, supplied by the rightmost twelve bits of IHIT.

NEXT can also be used to help perform erasures of selected material from the middle of a display. The BACKSPACE command actually erases the last N display instructions starting with NEXT minus one. Therefore, to erase selected parts of a display, NEXT is set to indicate the proper display instruction, the BACKSPACE command is used and then NEXT is set to its original value.

The tracking cross can be used as a means for using the light pen for drawing purposes. DDI can follow the light pen across the crt by using the tracking cross. When the light pen is pointed to the tracking cross, DDI tries to keep the tracking cross centered under the light pen. Each time DDI receives a light pen hit on the tracking cross, LITEPEN is called and informed of the position of the tracking cross through the hit word (IHIT). The structure of IHIT is slightly different than for hits on displayed data. IHIT is negative and the Y coordinate is in the rightmost twelve bits of IHIT and the X coordinate is in the twelve bits to the left of the Y coordinate.

DDI, which monitors everything at the console, keeps track of every hit detected by the light pen. Therefore, LITEPEN will be called by CRT for every hit detected by DDI unless LITEPEN is written to ignore multiple hits on the same data. There is a CRT command that helps in this regard. The entry

```
CALL CRT(6HIGNORE)
```

causes all unprocessed hits currently in the light pen hit buffer created by the BUFFER command to be eliminated. The common procedure is to execute this command just before leaving LITEPEN. This does not completely eliminate the

problem of multiple hits as the light pen may be collecting hits on the same data after leaving LITEPEN. Therefore, some method of disabling hits on the same characters, lines, or dots must be employed.

The entry

```
CALL CRT(6HENABLE)
```

will in effect turn the light pen on. The command to turn the light pen off is

```
CALL CRT(7HDISABLE).
```

When the light pen is disabled, CRT will not call LITEPEN when a hit is detected by DDI. These commands do not generate any display instructions.

To remove the tracking cross from the display, the command

```
CALL CRT(8HCROSSOFF)
```

is used. To display the tracking cross, provided the light pen is enabled, the entry

```
CALL CRT(7HCROSSON)
```

is used. The command

```
CALL CRT(6HSETPEN,IX, IY)
```

is used to position the tracking cross at the coordinates (IX, IY). These commands do not generate any display instructions.

The primary functions of the keyboard are as a control device and as a data input device. CRT contains an option for allowing subroutines or entry points to be called as a result of keyboard entries. In addition, any of the CALL CRT commands may be entered from the keyboard. As the characters are typed on the keyboard, they are displayed across the bottom of the scope from left to right so that the user may verify his typing. In case of a typing error, the user may delete the entire entry by typing %, or he may backspace a character at a time by using \$. The end-of-line symbol is the semi-colon.

The user must equate a subroutine name with a character string if he wants the subroutine to be entered as a result of a keyboard entry. This may be done with the command

```
CALL CRT(7HCOMMAND, NAME, ISYM),
```

where NAME contains a left-justified character string of the subroutine name and ISYM contains the same type of character string with one to ten characters which contains the synonym. After this command has been executed, the user may type in the synonym at the keyboard and CRT will call the subroutine named in the variable NAME.

The command

```
CALL CRT(6HTEXTIN, N)
```

will allow the user to type in an arbitrary character string.

CRT waits until the user types in N characters on the keyboard, types the end-of-line symbol, or enters ninety characters before returning from the call. The characters typed in are available in the user's program through a ninety word labelled COMMON block called KBIC. The characters are right-adjusted with zero fill.

The entry to clear the keyboard display and set the KBIC COMMON block to zero is

```
CALL CRT(5HCLEAR).
```

The command should be used after each TEXTIN command.

The command

```
CALL CRT(7HMONITOR)
```

causes CRT to monitor light pen hits and keyboard entries.

If a light pen hit occurs under this mode, CRT will call LITEPEN. When LITEPEN returns to CRT, the MONITOR loop continues. To turn off the display of keyboard entries, the entry is

```
CALL CRT(9HCOLUMNNOFF).
```

In order to turn it back on,

```
CALL CRT(8HCOLUMNON)
```

is used.

Anytime the user would like a microfilm copy of the current display, he may obtain one by using the command

```
CALL CRT(6HRECORD).
```

The command does not automatically advance the microfilm to the next frame; and if the user does not want multiple exposures, he must use the statement

```
CALL CRT(7HADVANCE)
```

to advance to the next frame.

CRT has the capability of saving for future use a digital description of the display instructions generated to create a display. The descriptions may be retrieved and redisplayed during the same job or copied on cards or magnetic tape for future jobs. Before any input-output is done, a buffer area must be set aside for display reading and writing. The appropriate command is

```
CALL CRT(2HIO, 6HBUFFER, IARRAY, LENGTH).
```

LENGTH is the length of the input-output buffer area IARRAY. IARRAY must be at least sixty-four words long, and if it is longer, it should be a multiple of sixty-four words in length.

The entry

```
CALL CRT(2HIO, 5HWRITE, FILE)
```

will write the current display on the specified file. FILE is the display code name of the specified file. In order to read the next display from a specified file and display it, the command is

```
CALL CRT(2HIO, 4HREAD, FILE).
```

CRT clears the current display before the display of the display that is read. Thus, a saved display may not be superimposed on an existing display. The command

```
CALL CRT(2HIO, 6HREWIND, FILE)
```

rewinds the file specified in display code of the word FILE.

The following appendices contain a discussion of the programs that provide the engineer with the capability of describing a control system to the computer in block diagram form. The programs are highly dependent on the software described in this appendix and refers to subroutine CRT frequently. This discussion of CRT did not include all of the commands available. It was meant to be just thorough enough to allow the reader to understand the following appendices without difficulty.

APPENDIX D

MASTER SELECTION MODE

The master selection mode is the central operating point from which all other modes can be reached. The program enters the master selection mode initially. Figure D.1 is an illustration of the display in this mode. When the engineer wishes to select one of the modes, he simply touches its title with the light pen.

Figure D.2 is a block diagram of the branching options implemented in the system. The first selection when the program is initially loaded must be DRAW since no block diagram has been previously entered. On completion of the draw mode the program passes to the equation mode. From the equation mode the engineer may return to the draw mode or he will pass to the data mode when all equations have been entered. From the data mode he may return to the draw or equation modes and he will pass to the master selection mode when all parameter values have been entered. From the master selection mode the engineer may choose the root locus, frequency response, or simulation modes or return to the draw, equation, or data modes to alter the information entered there.

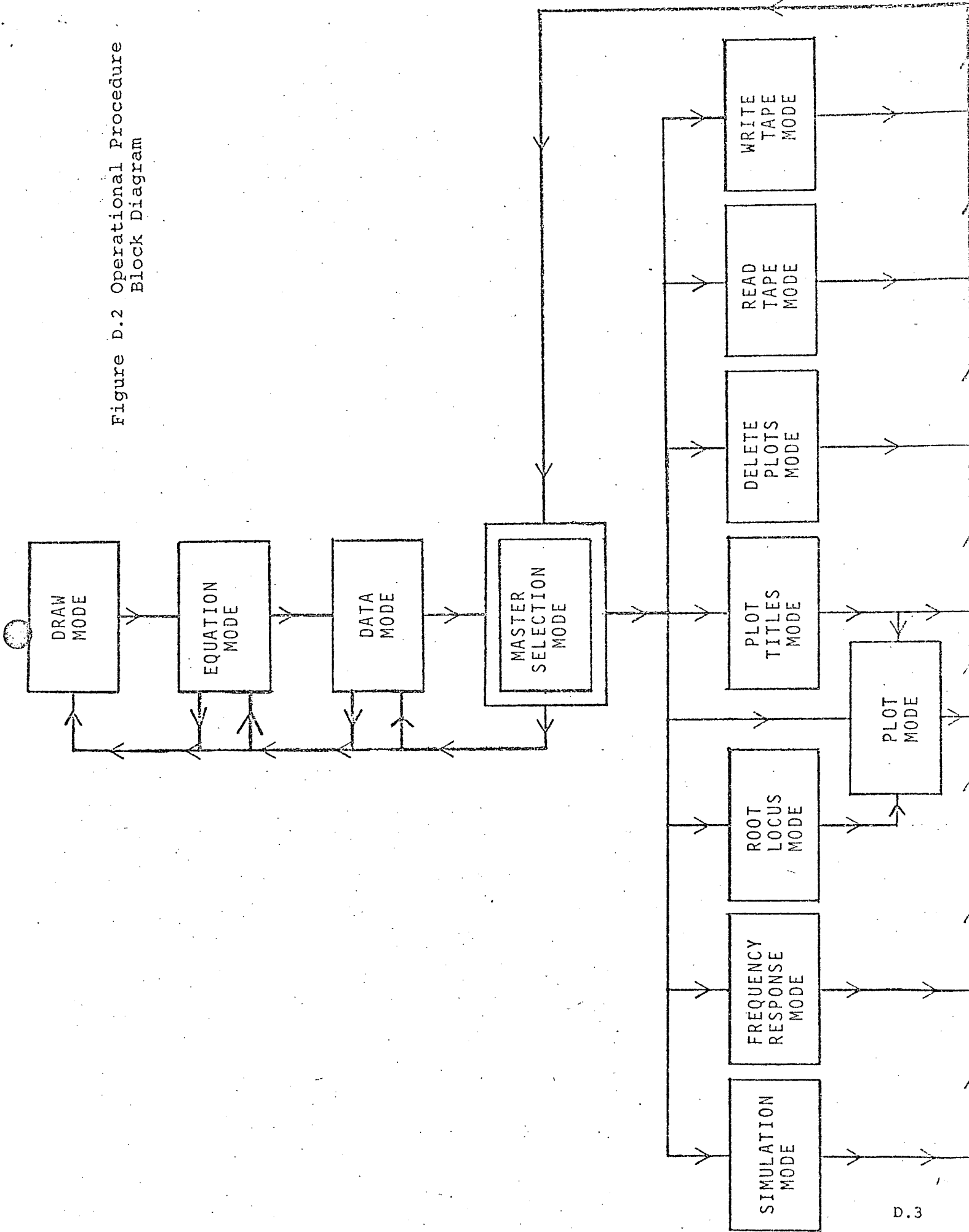
When the program returns to the master selection mode from an analysis program, several numbers are displayed under NEW corresponding to the plots generated in the analysis program. One of these plots may be selected with the light

MASTER SELECTION

DRAW			PLOT
EQUATION			TITLES
DATA			DELETE
SIMULATION		STORED	NEW
ROOT LOCUS	1	6	1
FREQUENCY RESPONSE	2	7	2
WRITE TAPE	3		3
READ TAPE	4		4
	5		

Figure D.1 MASTER SELECTION MODE menu

Figure D.2 Operational Procedure
Block Diagram



pen and the program enters the plot mode. The data previously stored is retrieved and displayed in the plot mode. In the plot mode the engineer has the ability to enter a title to describe the plot. All of the titles may be viewed collectively in the plot titles mode. The engineer has the opportunity to store the plot he is viewing so he may redisplay it for future reference. Plots may be removed from storage by taking consecutive light pen hits on DELETE and on the plot number.

The write tape mode provides the capability of writing all disk files and necessary central memory storage onto magnetic tape for future use. The read tape mode provides the capability of reading files previously stored on magnetic tape into the disk files and central memory storage for continued processing.

The functions of the root locus, frequency response, and simulation modes are self evident. A detailed description of the operation of the programs in all the modes is given in the following appendices.

When a mode is selected in the master selection mode, the program sets and inspects a number of flags to insure that erroneous operations are not performed. For example, if the engineer selects the frequency response or root locus mode the program checks to be certain the block diagram does not contain nonlinear blocks.

APPENDIX E

DRAW MODE

The draw mode uses the interactive capabilities of the graphic console to aid the engineer in constructing the block diagram. The block diagram configuration is set in this mode and it is not possible to change the configuration in any other mode.

When the display for the draw mode appears on the crt, the light pen is on, and program CRT is monitoring the console for light pen or keyboard interrupts. The display is a "menu" of items which the engineer may select and use in constructing a block diagram. The menu includes an initiator block for input signals, a terminator block for output signals, a distribution point for joining lines, a linear transfer function block, a summer block, a nonlinear block, a line, and a large D used as a delete command. In addition, there are four commands available for program control in the upper left-hand corner of the scope. These commands are SETBLOCK, FILM IT, COMPLETE, and CLEAR. The SETBLOCK command is used in the construction of the block diagram and is described in detail in the following paragraph. When the FILM IT command is "touched" with the light pen, the current display is saved on microfilm. When the engineer has completed construction of the block diagram, he touches the COMPLETE command with the

light pen to enter the equation mode. Touching the CLEAR command causes the entire block diagram to be erased. Figure (E.1) shows the location of each of these items in the initial display.

To place a block at any location on the crt, the user must first touch the desired block in the menu with the light pen. The light pen hit causes the block to begin winking. The tracking cross is touched with the light pen and a block similar to the winking block appears under the tracking cross. As the tracking cross follows the light pen across the screen, the program keeps the block under the tracking cross. When the engineer is satisfied with the location of the block, he may set the block by touching the word SETBLOCK with the light pen. The SETBLOCK command permanently positions the block at its last location, resets the block in the menu to non-winking, and moves the tracking cross one hundred raster units to the right of its last position. The engineer may also set the block by touching any non-winking block in the menu, but if he wishes to display the same type of block twice in a row, he must first touch SETBLOCK and then touch the block in the menu. This block will begin to wink and he may place it on the crt. The engineer may delete any block by touching the D in the menu and then touching the block to be deleted. If he makes a mistake and touches the wrong block in the menu, he simply touches the correct block with the light pen and it will begin to wink and the former block will be set to non-winking.

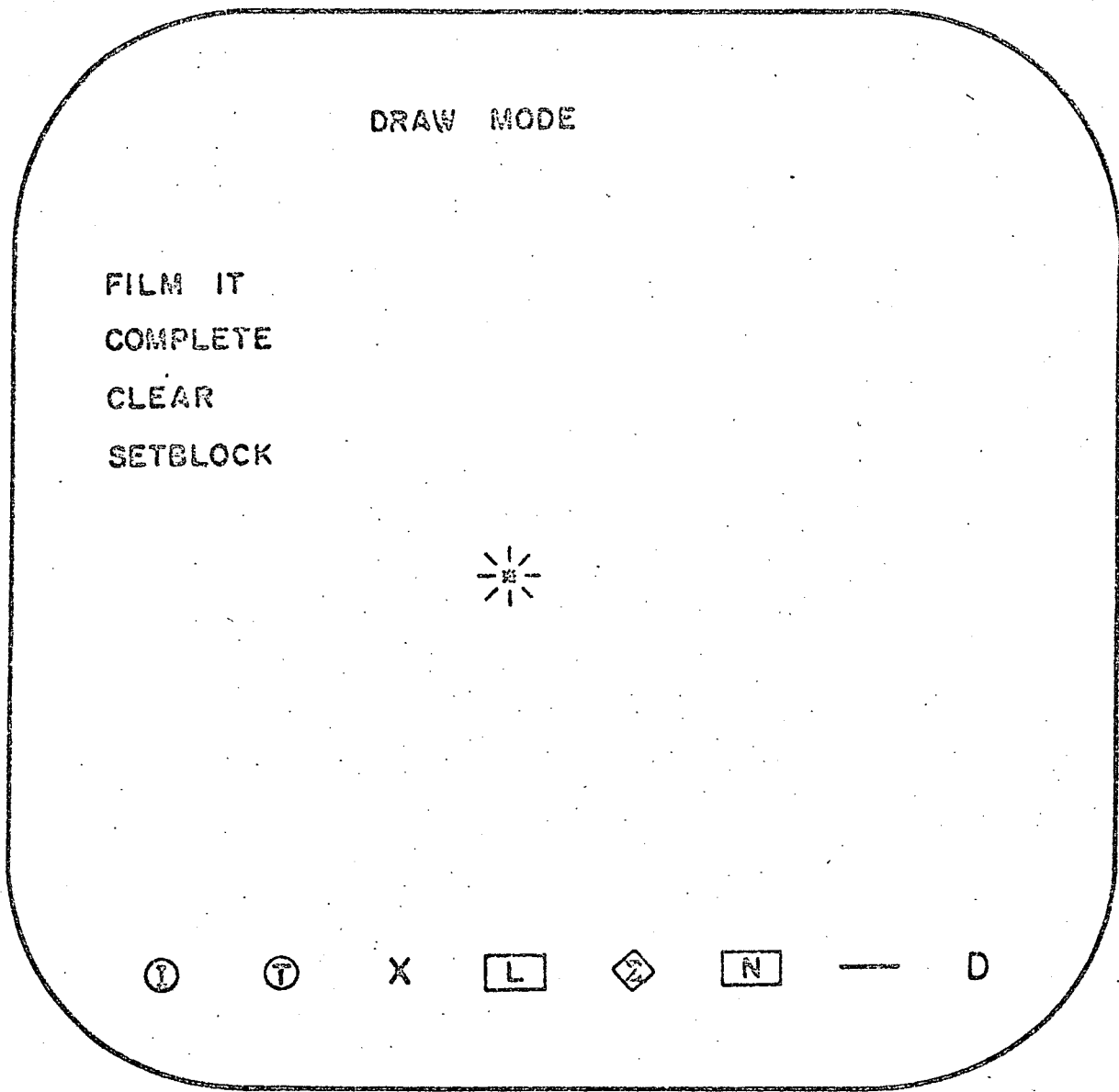


Figure (E.1) Initial Display

The blocks are assigned an identifier symbol as they are set on the screen so the engineer may distinguish between two blocks of the same type. The identifier symbol is of the form X-nn where X is a character which designates the type of block. The character corresponds to the one displayed inside the same type of block in the menu (L = linear block, etc.). nn is the number of that type of block that has been displayed since the last CLEAR command. A maximum of ten initiator or terminator blocks can be displayed between CLEAR commands. A maximum of 100 blocks of the remaining types is allowed. The total number of elements displayed between CLEAR commands can not exceed 150.

The blocks are automatically spaced 100 raster units apart in the vertical direction to make the block diagram more orderly. This spacing is accomplished by building within the program a series of vertical coordinates when the first block is set and finding the nearest vertical coordinate to the other blocks when they are set.

The engineer must touch the "line" in the menu with the light pen when he wants to connect the blocks. The line begins to wink and then he may touch two blocks in the diagram and a line appears between the blocks. The output is assumed to come from the block touched first, and the signal flow goes to the block touched second. The lines are displayed with an arrow denoting the direction of signal flow and are labelled consecutively, provided they do not originate at a distribution

point. Lines connecting initiator blocks to any other blocks are identified by a label of the form Y-n. Other lines are labelled X-nn. A maximum of 100 lines can be displayed since the last CLEAR command. A line may be deleted with the delete command.

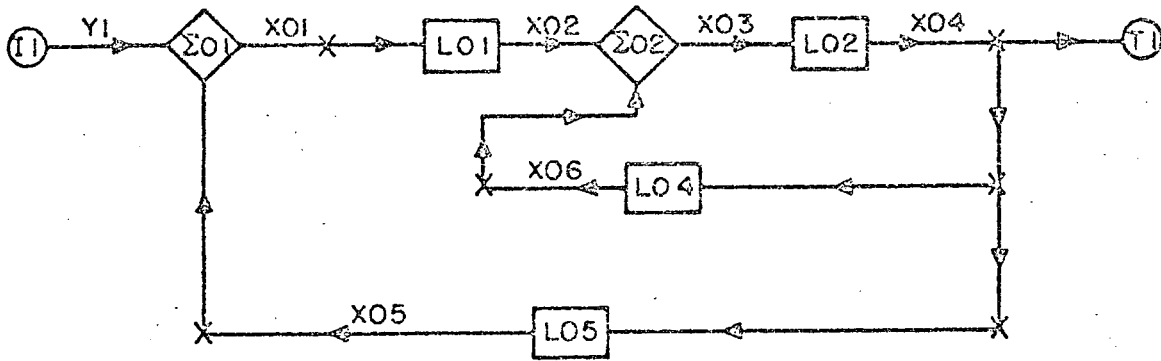
In order to provide a more orderly looking block diagram the program attempts to make the block diagram square by moving the block in the horizontal direction when a vertical line is drawn to it. If the program can not move the block because other blocks are in the way, it constructs a right-angled line between the blocks. An example of the right-angled line is shown in Figure (E.2).

If the engineer attempts to make an illegal connection, an error message appears at the bottom of the screen, and the line in the menu is set non-winking. The error message may be deleted with the delete command, or the engineer may try to draw the line again. If he draws the line correctly, the program automatically deletes the error message.

As the engineer builds the block diagram model of his system, the program must set up a data structure which contains all the pertinent information about each element (blocks and lines) used. The data structure consists of eight 150 word arrays. Each element (block or line) uses one word in each array. As each element of the block diagram is set on the screen, the pertinent information about it is entered in the next consecutive word of all eight arrays. Thus each element

DRAW

FILM IT
COMPLETE
CLEAR
SETBLOCK



I

T

X

L

Σ

N

—

D

Figure E.2 Display Presented During DRAW MODE

in the block diagram has associated with it an array index number which identifies its order in the array.

The first array contains the addresses of the first display instructions associated with each element. The second array contains a numeric code which identifies the type of element (0 = deleted element, 1 = initiator block, 2 = terminator block, etc.). The third and fourth arrays contain the x and y coordinate positions of blocks or the initial ends of lines. The fifth and sixth arrays contain the coordinate positions of the terminal ends of lines. For blocks the fifth array contains the identifier symbol of the block. For distribution points the fifth array contains zero. For blocks the sixth array contains three numbers packed in 12-bit bytes. These three numbers are the total number of lines, the number of inputs and the number of outputs associated with the block. The sixth array also contains a flag which indicates whether the mathematical model of that block has been specified. The mathematical model is specified (and this flag set to one) in the equation mode. The seventh array contains information about which lines are connected to which blocks. For a block the seventh array contains the array index numbers of all the lines connected to it (packed in 12-bit bytes). For a line the seventh array contains the array index numbers of the blocks which it connects (packed in 12-bit bytes). The eighth array contains miscellaneous information. For a line it contains the identifier symbol associated with the line. For a termination block it contains no informa-

tion. For other blocks it contains a code which identifies which sides of the block have lines connected to them. The lowest eighteen bits of the eighth array contains the disk sector address where the mathematical model of the block is stored. A more complete description of the mathematical model data format will be given in Appendix F.

The seventh array of this data structure contains information about which blocks are connected to which lines and which lines are connected to which blocks. This type of data structure is called a ring data structure with both forward and backward pointers.

The data stored in the eight arrays described above provides all the information necessary to reproduce a block diagram, to perform an error analysis to identify erroneous connections, and to convey the block diagram structure to the simulation and analysis routines. More details of the use of this data for these functions will be given in subsequent appendices.

APPENDIX F
EQUATION MODE

In the equation mode the mathematical model for each element in the block diagram is entered. The engineer may enter the equation mode from the draw mode by touching the COMPLETE command or from the master selection mode by touching the EQUATION command. If the engineer attempts to enter the equation mode before a complete and correct block diagram has been entered, the program will display an error message which identifies the action necessary to complete or correct the block diagram. The program which controls the equation mode is written so the engineer is required to provide only the minimum information necessary to correctly describe the model of each element. Cues are provided to prompt the engineer as to what information is required and what format is acceptable. If an error is made (which the program can diagnose) the engineer is notified by an error message and permitted to correct the error. If the engineer recognizes he has entered erroneous data, he may backspace (BKSP) to the error, correct it, and proceed on again. The first time the engineer enters the equation mode he is required to provide a model description for each element in the block diagram. On subsequent entries to the equation mode he is only required to provide model descriptions for those elements which have been altered by

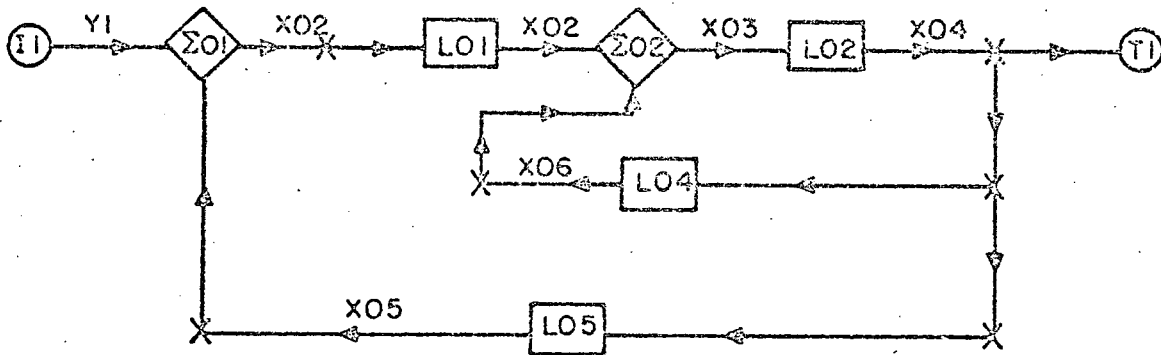
modifying the block diagram since the previous pass through the equation mode. He may, however, review any of the model descriptions previously entered or modify any portion of any previously entered model.

When the engineer enters the equation mode, the block diagram remains on the screen while the lower portion of the screen is used to display cue information and to display the model information so the engineer can verify his entries. Figure F.1 illustrates the form of this display. When the program is ready to receive the model for a particular block, it sets that block winking; traces the points in the ring data structure to determine the input and output variable names for that block; and provides model format cues. The program requests the engineer to enter the mathematical models of the elements by type in the order they were set in the block diagram; all of linear blocks followed by the summer blocks, the nonlinear blocks and the initiator blocks. The model entry procedure for each of these types of blocks will be described in the subsequent sections of this appendix.

Description of Linear Blocks:

The description of the model of the linear blocks is given in terms of a ratio of polynomials in the differential operator S . The engineer is requested to provide the order and coefficients of the numerator polynomial followed by the order and coefficients of the denominator polynomial. The

EQUATION MODE



DESCRIPTION OF WINKING LINEAR ELEMENT. COEFFICIENTS MUST BE CONSTANTS.
 FORM IS RATIO OF POLYNOMIALS $0 \leq N \leq M \leq 30$
 M = DENOMINATOR ORDER N = NUMERATOR ORDER

Figure (F.1) Linear Block Description
 in Equation Mode

order of the denominator must be less than or equal to thirty and the order of the numerator must be less than or equal to the order of the denominator. The coefficients are entered in order from the keyboard with the coefficient of the highest order term first. The coefficients of the polynomials may be algebraic expressions containing constants and variable names. Each expression is displayed as it is entered for verification. Errors made in entering these expressions may be corrected before proceeding to the next coefficient description. Certain errors are detectable by the program. These will produce an error message and provide an opportunity for the engineer to correct his description. Errors may be corrected or changes made in previously entered coefficients by entering the command BKSP on the keyboard. This command steps the program back one coefficient each time it is entered. The program will not back step beyond the point of specifying the order of a polynomial. Alternatively, changes in earlier specified coefficients can be made by recycling back through the equation mode from the beginning. In order to recycle back to the beginning of the equation mode the engineer enters the command DRAW on the keyboard. The program returns to the draw mode where the engineer may modify the block diagram structure or select the COMPLETE command and enter the equation mode. Upon re-entering the equation mode it is not necessary to re-enter all of the previously described models. If no changes have been made in the block diagram structure which effect the input or output

variables of a particular block, the instruction cue "ENTER YES IF YOU WISH TO CHANGE THE MODEL OF THE WINKING BLOCK" appears. If the engineer enters a semicolon symbol without entering YES, the program uses the previously defined model for the indicated block. If the engineer wishes to change (or review) the model of the specified block, he enters YES; whereupon the instruction cue "ENTER YES IF YOU WISH TO CHANGE THE NUMERATOR" is displayed. If the engineer enters a semicolon symbol without entering YES, the program uses the previously defined model for the numerator and the instruction cue "ENTER YES IF YOU WISH TO CHANGE THE DENOMINATOR" is displayed. If YES is entered following either of these cues the instruction cue "ENTER THE COEFFICIENT YOU WISH TO CHANGE OR ↓ TO VIEW - NUMERATOR (or DENOMINATOR)" is displayed. The engineer may enter the coefficient name to change a coefficient or ↓ to "page through" the coefficients. Paging through the coefficients is accomplished by repeatedly entering the semicolon symbol. As this is done the algebraic expression for each coefficient is displayed successively on the screen. While paging through the coefficients changes may be entered by simply entering the new coefficient expression over the displayed expression. The engineer may escape from viewing the coefficients by paging through to the end of the polynomial being viewed or by entering the ↑ (escape) symbol. If the ↑ symbol is used the engineer is returned to the instruction cue "ENTER THE COEFFICIENT YOU WISH TO CHANGE OR ↓ TO VIEW - NUMERATOR (or

DENOMINATOR)". To escape to the next cue the engineer enters the ↑ symbol again. Using these procedures the engineer may view, modify, correct or re-enter any part or all of the mathematical model of any linear block. Similar procedures allow the engineer to view or change the models of other types of blocks.

As the description of a linear element is entered from the keyboard it is stored in an array. When the description is completed and checked for errors the array is stored in a disk file and the address of the disk file is stored in the appropriate word of the eighth array of the ring data structure associated with that block as described in Appendix E. The flag bit which indicates that the model has been entered is also set to one.

Description of Summer Blocks:

When the mathematical models of all the linear blocks have been entered, the program sets the first summer block winking, traces the forward and backward pointers in the ring data structure to determine the input and output variable names of the summer and asks the engineer to enter the sign associated with each input variable. As these signs are entered in response to the information cues, the equation of the summer is formed and displayed for verification. During the entry of the summer variable signs, the commands BKSP and DRAW are available and cause the same actions described previously. When the model of the summer is complete its description is

stored on a disk file, the address of the disk file is placed in the appropriate word of the eighth array of the ring data structure, and the flag bit which indicates that the model has been defined is set to one.

If the engineer subsequently enters the equation mode and comes to a summer block which has previously been defined and no changes in the block diagram have occurred which effect the inputs or outputs of the summer, the previously stored description is retrieved from the disk and displayed. The engineer is then asked if he wants to change the description of the summer (sign of any input). If he enters YES he is required to input the sign of all the inputs as before. If he enters NO the previous description is retained.

Description of Nonlinear Blocks:

After all the summer blocks have been described, the program sets the first nonlinear block winking, traces the forward and backward points in the ring data structure to ascertain the names of the input and output variables associated with that nonlinear block, and displays the information cue "DESCRIPTION OF THE WINKING NONLINEAR ELEMENT. USE ONLY STANDARD VARIABLES, list of input and output variable names, AND CONSTANTS." The engineer enters the mathematical model of the nonlinear block in the form of algebraic and integral equations which may include the nonlinear functions; sine, cosine, tangent, arc sine, arc cosine, arc tangent, hyperbolic sine, hyperbolic cosine, hyperbolic tangent, absolute value,

square root, exponential, logarithm, power, switch, limiter, limited integrator, dead space, time delay and an arbitrary segmented function of one or two variables. Any algebraic or integral expressions of time and the input and output variables of the nonlinear block may be used to describe the block. Each expression is limited to ninety characters and each nonlinear block is limited to one hundred expressions. Each expression is displayed as it is entered for verification. The engineer may use the DRAW and BKSP commands as before to correct errors or modify the description of the nonlinear element. When the engineer has completed the description of the nonlinear block he enters the command NEXT. When the NEXT command is received the program stores the nonlinear block description on a disk file, enters the disk file address in the ring data structure and sets the flag which indicates that the nonlinear block has been described.

If the engineer re-enters the equation mode and comes to a nonlinear block which has been previously defined, the program checks to determine if any changes have been made in the block diagram which effect any of the input or output variables of that block. If any changes have occurred the engineer may not bypass reviewing the nonlinear block description. If no changes have been made he may elect to view or bypass the nonlinear block description. If the engineer elects to bypass viewing the nonlinear model, the program retains the previously stored model. If the engineer elects to view the

model he may "page through" the model by entering the semicolon symbol repeatedly. He may at any point modify the model by entering new expressions, escape from the viewing process (\uparrow , which retains the modified model), or enter NEXT which retains the modified model up to the point where NEXT was entered. The remainder of the previously stored nonlinear model is deleted. When the engineer has completed the modifications on the nonlinear model, it is again stored on the disk file.

Description of Input Blocks:

The description of the mathematical model of the input blocks has all the freedom of the description of nonlinear blocks except it is limited to one expression. As the expression is entered it is displayed for verification. When the description of the input block is complete, it is stored on a disk file with appropriate entries made in the ring data structure. When an input block is encountered which has previously been described, the engineer is given the opportunity to modify the description, review the description or retain the previously stored description.

After all the models of all the linear, nonlinear, summer and input blocks have been entered, the program performs several other tasks before proceeding to the data mode. First, it collects the names of all the input variables to terminator blocks and stores these on a disk file. These names are later used by the digital simulator program as

described in Appendix H. As the mathematical models of the system elements are entered, the program identifies and places in an array all parameter names which are used to describe the system, but which are not standard variable names (like time) or the names of the output variables of blocks in the block diagram. This array of parameter names must be assigned values in the data mode. If the data mode has not been previously completed, these parameters are assigned the value of zero and the program enters the data mode. If the data mode has been previously completed, the program checks to see if each of these parameter names has been assigned a value or not. If a parameter value has been previously assigned that value is assigned to the parameter name; otherwise the value zero is assigned. When all parameter names have been checked the program passes to the data mode. The operations performed in the data mode are described in Appendix G.

APPENDIX G

DATA MODE

The data mode is entered automatically upon completion of the equation mode or it may be entered from the master selection mode. In the data mode the engineer assigns a value to all of the parameters specified in the equation mode. The initial value assigned to all the parameters is zero.

If the engineer has entered the data mode for the first time, each parameter and its preassigned value is displayed on the screen. The engineer may enter a new value and a semicolon on the keyboard and the program will assign that value to the parameter. If the engineer enters a semicolon without a value, the previously assigned value will be retained. As the engineer enters values they are displayed on the screen for verification. If an error is made while entering a parameter value, it may be corrected. Changes in previously entered parameter values may be made by use of the BKSP command. The engineer may return to the draw or equation modes by entering the commands DRAW or EQN. After definition of all the parameters the values are stored on the disk for later use. If, however, the DRAW or EQN commands are used the newly assigned parameters are not saved on the disk file and must be re-entered.

When the data mode is re-entered the information cue "ENTER YES IF YOU WISH TO CHANGE A CONSTANT" is displayed.

If the engineer does not enter YES, the program passes to the master selection mode. If he enters YES the cue "ENTER THE NAME OF THE CONSTANT YOU WISH TO CHANGE OR ↓ TO VIEW" is presented. The engineer may either name a parameter and enter a new value for it or he may "page through" the parameter values by entering ↓ followed by the semicolon symbol. Each time the semicolon symbol is entered a new parameter with its value is displayed. The engineer may enter new parameter values as he "pages through" the parameters. To escape the paging process the engineer enters ↑. This command returns the engineer to the cue "ENTER THE NAME OF THE CONSTANT YOU WISH TO CHANGE OR ↓ TO VIEW". Entering a second ↑ symbol advances the program to the master selection mode after storing newly entered parameter values on the disk file.

APPENDIX H
SIMULATION MODE

The purpose of the simulation mode is to obtain the response in time of all the variables connected to terminator blocks when the model is subjected to the inputs specified for the initiator blocks. To accomplish this the program uses a variable step size fourth order Runge-Kutta numerical integration procedure patterned after the MIMIC digital simulator program. Before the numerical integration can proceed the mathematical model of each element in the block diagram must be retrieved from the disk files, translated into a set of algebraic and integral equations, sorted into an executable order, and assembled into a machine code program compatible with the numerical integration routine. In addition, model parameter values specified in the data mode must be retrieved and placed in an appropriate array, simulation parameters must be specified and provisions for collecting the response data for display must be provided. Once the machine code program is assembled it is stored on a disk file and may be re-executed with as many different parameter values as desired. If the engineer changes the mathematical model of the system in any way, other than changing parameter values, the entire machine code program must be reassembled.

The program which transforms the mathematical model of each block into a set of algebraic and integral equations

which can subsequently be assembled into a compatible, executable machine code program is quite complex. Its operations will be described in the remaining paragraphs of this appendix. The mathematical model of a linear block is a transfer operator in the form of a ratio of polynomials in the differential operator S

$$T(S) = \frac{A_m S^m + A_{m-1} S^{m-1} + \dots + A_1 S + A_0}{B_n S^n + B_{n-1} S^{n-1} + \dots + B_1 S + B_0} \quad (H.1)$$

If X02 is the output variable of the linear block and X01 the input variable, the transfer operator represents a differential equation of the form

$$\begin{aligned} & (B_n S^n + B_{n-1} S^{n-1} + \dots + B_1 S + B_0) X02 \\ = & (A_m S^m + A_{m-1} S^{m-1} + \dots + A_1 S + A_0) X01 . \end{aligned} \quad (H.2)$$

Equation (H.2) can be converted into the integral form

$$\begin{aligned} X02 = & \frac{1}{B_n} \left[- \frac{B_{n-1} X02}{S} - \frac{B_{n-2} X02}{S^2} - \dots \right. \\ & + \frac{A_m X01 - B_m X02}{S^{n-m}} + \frac{A_{m-1} X01 - B_{m-1} X02}{S^{n-m-1}} + \dots \\ & \left. + \frac{A_1 X01 - B_1 X02}{S^{n-1}} + \frac{A_0 X01 - B_0 X02}{S^n} \right] \end{aligned} \quad (H.3)$$

by dividing through by $B_n S^n$ and collecting like powers of S. In Equation (H.3) we recognize $1/S$ as an integral operator, $1/S^2$ as a double integral operator, etc. Thus Equation (H.3)

is the sum of a set of multiple integrals which can be solved directly by a numerical integration procedure. A much more efficient numerical integration procedure can be obtained by transforming the set of multiple integrals into a set of nested integrals. Equation (H.3) may be written

$$\begin{aligned}
 X_{02} = & - \frac{1}{B_n} \left[\frac{1}{S} (B_{n-1} X_{02} + \frac{1}{S} (B_{n-2} X_{02} + \dots \right. \\
 & + \frac{1}{S} (B_m X_{02} - A_m X_{01} + \frac{1}{S} (B_{m-1} X_{02} - A_{m-1} X_{01} + \dots \dots \\
 & \left. + \frac{1}{S} (B_1 X_{02} - A_1 X_{01} + \frac{1}{S} (B_0 X_{02} - A_0 X_{01})) \dots) \dots) \right]
 \end{aligned}
 \tag{H.4}$$

Defining a new set of variables Z_k for $k = 0$ to $n-1$ we may rewrite Equation (H.4) as the set of equations

$$\begin{aligned}
 Z_0 &= \int (B_0 X_{02} - A_0 X_{01}) dt \\
 Z_1 &= \int (B_1 X_{02} - A_1 X_{01} + Z_0) dt \\
 &\vdots \\
 Z_m &= \int (B_m X_{02} - A_m X_{01} + Z_{m-1}) dt \\
 &\vdots \\
 Z_{n-1} &= \int (B_{n-1} X_{02} + Z_{n-2}) dt \\
 X_{02} &= - \frac{1}{B_n} Z_{n-1}
 \end{aligned}
 \tag{H.5}$$

The program which controls the simulation mode uses a set of

algebraic, nonlinear, and integral functions (ADD, SUB, MUL, DIV, EQL, SIN, COS, SQR, ABS, INT for integrate, etc.) to form a set of function statements for the equations which describe each block. Equations (H.5) may be expressed directly in this functional form. This same procedure is used to form function statements corresponding to the models of all the linear blocks. It was pointed out in Appendix F that the coefficients of the transfer operators for linear blocks may be algebraic expressions. As these expressions are entered from the keyboard the program develops a set of corresponding function statements. These function statements are combined with the statements corresponding to Equations (H.5) to complete the description of the linear block. When the function statements have been formed for all the linear blocks, the models for the summer blocks are retrieved from the disk and a set of function statements for each is formed.

While the nonlinear and initiator block models were being entered in the equation mode, the program developed sets of function statements for each. These function statements are now retrieved to complete the function statement model of the system. This very large set of function statements is now sorted into an executable order, eg. the function statements are ordered so that no parameter is used as an input to a function until it has been defined as a constant (in data mode) or as the output of a previously defined function. This sorted array of function statements is assembled into the machine code program which operates with the numerical integration routine.

All of the block diagram variables that are connected to terminator blocks are specified as system output variables. As the numerical integration proceeds the values of these variables are collected at regular intervals of the integration variable and stored on disk files for later display. After the integration has proceeded for the specified integration time, the integration routine is discontinued and the output variable values are retrieved from the disk files, rearranged, and restored on disk files. Since the values of all the output variables are collected periodically and stored on the disk file, the values of a particular variable are interwoven among the values of the other variables. In order to form a plot to display the values of a single variable those values must be separated from the values of the other variables. This is the purpose of rearranging the results and restoring them on the disk file.

After these functions have been completed, the program returns to the master selection mode where the engineer may select the next operation he wants to perform.

APPENDIX I

FREQUENCY RESPONSE MODE

In the frequency response mode the engineer may obtain frequency response plots of the open loop transfer function for linear systems. The frequency response plots are presented in Bode, Nyquist, and gain-phase form. When the engineer selects the frequency response mode, the program checks to be certain that the block diagram contains only linear elements, displays the block diagram, and instructs the engineer to specify the point at which he wants the block diagram opened. The engineer may specify any point in any loop of the block diagram. The program then utilizes the general purpose block diagram reduction routine described in Appendix M to obtain the transfer function of the open loop system. The engineer is asked to specify the range of frequencies to be covered by the plots and the program produces the plot data and returns to the master selection mode.

In the classical design of feedback control systems the system is regarded as a single loop system with a compensator element in series with the plant element and the frequency response characteristic of the plant is obtained. Using classical control theory the engineer may then specify the phase margin, gain margin, bandwidth, M_m , etc. characteristics he desires in the compensated system. Comparing these specifi-

cations with the frequency response characteristic of the plant the engineer may determine the gain and phase characteristics required in the compensator.

In the frequency response mode when the engineer specifies the point at which he wants the block diagram opened, he is in effect specifying the point at which he wishes to insert a compensation element. The block diagram reduction procedure then obtains the open loop transfer function of the system broken at that point which is equivalent to the plant transfer function in classical control theory. The engineer may, therefore, examine the frequency response plots produced in the frequency response mode to determine the gain and phase characteristics required of a compensator inserted at the point he opened the system to obtain desired gain margin, phase margin, bandwidth, M_m , etc. specifications.

Upon initial selection of the frequency response mode the program determines the loops and paths of the block diagram. If the frequency response mode is entered at some subsequent time and the block diagram structure has not been altered the loop and path identification obtained previously may be utilized. The program stores this information and sets appropriate flags to provide most efficient use of previously generated data. Similarly, in the equation mode the coefficients of the transfer operator polynomials were specified in terms of algebraic functions of system parameters. In the data mode numerical values of the parameters were specified.

Before the transfer functions can be reduced numerical values of the polynomial coefficients must be determined. Upon initial selection of the frequency response mode the program assembles a machine code routine to compute the values of the polynomial coefficients from parameters. If the frequency response mode is re-entered and no changes have been made in the model, other than changing the values of the parameters, this machine code routine need not be reassembled, simply re-executed with the new parameter values. The program maintains the required flags to utilize this efficiency. Finally, provision is made to re-enter the frequency response mode and change only the range of frequencies for which the plots are obtained. In this case the complete block diagram reduction previously done is utilized.

APPENDIX J

ROOT LOCUS MODE

The root locus mode provides the capability to determine the locus of the roots of the characteristic equation of the system as any specified parameter is varied over a specified range of values. Upon entering the root locus mode the engineer is asked to specify the parameter with respect to which he wishes to see the root locus, the initial value of the parameter, the increment in the parameter value between subsequent root locations, and the number of root locus points to be computed. The program performs the required computations and passes to the plot mode where the root locus plot is displayed. The engineer may return to the root locus mode via the master selection mode, respecify the parameter, initial value, increment, and number of points and obtain the new locus plot.

In order to obtain the root locus plots it is necessary to compute numerical values of the coefficients of the transfer operators, perform a block diagram reduction to determine numerical values of the coefficients of the characteristic equation, and compute the roots of the characteristic equation. These three operations must be performed for each value of the locus parameter.

When the root locus mode is selected for the first time, the program assembles one machine code routine to

compute the values of the coefficients of the transfer operators and another machine code routine to perform the block diagram reduction to compute the values of the coefficients of the characteristic equation. These machine code routines are executed along with a general purpose root finding program for each value of the locus parameter to produce the root values. If the engineer subsequently selects the root locus mode and no changes have been made in the block diagram structure the machine code routine which was previously assembled may be used for computing the values of the coefficients of the characteristic equation. If no changes have been made in the mathematical model of the blocks, except for changes in parameter values, the machine code routine which was previously assembled may be used to compute values for the transfer operator coefficients. The program maintains proper flags to take advantage of these efficiencies.

APPENDIX K

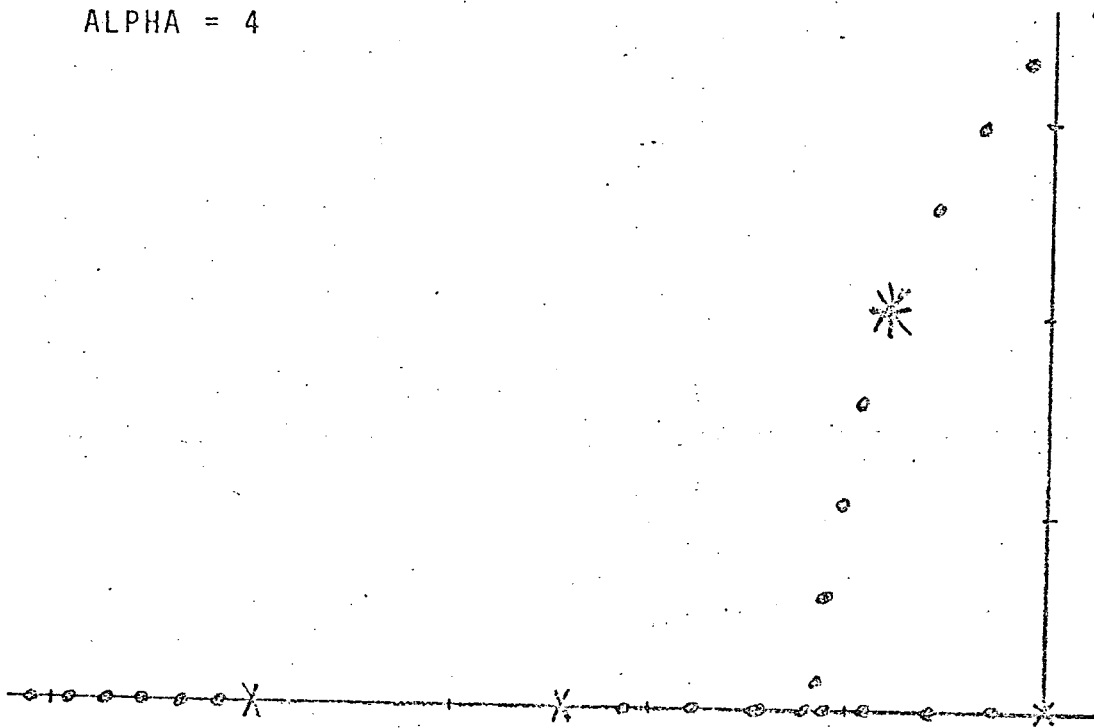
PLOT AND PLOT TITLES MODES

The purpose of the plot mode is to display the data generated in the simulation, frequency response, and root locus modes. In the plot mode the engineer has the capability to enter a title with each plot he displays. This provides him with a reminder of the source of the data which is displayed. In the plot titles mode all of the titles may be viewed collectively.

The plot mode may be entered from the master selection mode or the plot titles mode. The plot mode is also entered automatically after completion of the root locus mode. A typical plot mode display is shown in Figure (K.1). The engineer has the options of storing the plot for future reference, entering a descriptive title, microfilming the display for documentation purposes, rescaling plots which have more than one y-axis, returning to the master selection mode. Rescaling a plot causes all y-axes to have the same scale. Storing the displayed plot causes an entry to be made under the STORED PLOTS title in the master selection mode, and saves the plot data for future display. If a plot is not stored it is lost the next time the engineer selects the simulation, frequency response or root locus modes.

A representative display of the plot titles mode is shown in Figure (K.2). The engineer may enter the plot mode

ALPHA = 4



TITLE: ROOT LOCUS ALPHA = 1 TO 6

RESCALE REDUCE RETURN FILM IT STORE

Figure K.1 CRT Presentation in PLOT MODE

PLOT TITLES

STORED

- 1 ROOT LOCUS K = 0 to 50 ..
- 2 LINEAR SYSTEM X AND XDOT VS T
- 3 ROOT LOCUS ALPHA = 1 TO 6
- 4
- 5 FREQUENCY RESPONSE NYQUIST ALPHA = 5 K = 25
- 6 NONLINEAR SPRING X VS T
- 7

NEW

- 1
 - 2 NONLINEAR SPRING X VS T
 - 3
- RETURN

Figure K.2 CRT Presentation in PLOT
TITLES MODE

from the plot titles mode by selecting with the light pen the plot number or title associated with the plot he wishes to see. The engineer may also return to the master selection mode by selecting RETURN with the light pen.

Plots may be deleted from either the stored plots or new plots files by taking successive light pen hits on DELETE PLOT and the plot number while in the master selection mode.

When plots are viewed which were generated in the frequency response or root locus modes, the value of the frequency or locus parameter at points along the plot is needed. This value is provided by placing the tracking cross over the point in question. The program finds the value of the frequency or locus parameter corresponding to the point and displays the value in the upper left corner of the screen. The tracking cross may be moved to find the value at any point on the plot.

APPENDIX L

READ AND WRITE TAPE MODES

The purpose of the read and write tape modes is to allow the engineer to store and retrieve complete problem statements. This permits the engineer to continue the analysis of a system at a future session without re-entering the problem description and to study two or more systems at one session without losing the description of one when he goes to the other.

The write tape mode writes the data structure for the block diagram and all flags and counters necessary to the program on a magnetic tape. It then retrieves the descriptions of the linear, nonlinear, summer and input blocks from the disk files and writes them on the magnetic tape. The write mode then gives the engineer the option of choosing with the light pen which stored plots he would like to save on the magnetic tape. The program retrieves the plot data stored on the disk file and stores the data on the magnetic tape. When the engineer indicates with the light pen that he is through saving plot data, the program returns to the master selection mode.

The read tape mode performs the same operations in reverse order. The major difference is that as the plot data is read from the magnetic tape and written on a disk file,

the corresponding plots are assigned numbers consecutively in the order they were stored on the tape. When the program has completed this task, it returns to the master selection mode. The engineer may choose any program mode which was available to him when he stored the information on the magnetic tape.

APPENDIX M

BLOCK DIAGRAM REDUCTION

The block diagram reduction technique is highly dependent upon Mason's loop rule for signal flow graphs.

A signal flow graph, like a block diagram, is a diagram which graphically represents a set of linear relations. It is a weighted directed graph (i.e. a network) in which the nodes, also called vertices, are connected by branches, also called edges. The nodes represent each of the system variables and perform two functions. One, the addition of the signals on all incoming branches. Two, the transmission of the total node signal (the sum of all incoming signals) to all outgoing branches. A branch connected between two nodes relates the dependency of an input and an output variable in the manner equivalent to a block of a block diagram. A path is a branch or a continuous sequence of branches which are transversed in the same direction. A forward path between two nodes is one which follows the direction of successive branches and in which any node appears only once. A loop is a closed path which originates and terminates on the same node, and along the path no node is encountered more than once. A loop is also called a cycle or circuit.

In general, the overall transmittance or linear dependency between an independent variable at node i and a

dependent variable at node j is given by Mason's loop rule.

The formula is:

$$T_{ij} = \frac{\sum_k P_{ijk} \Delta_{ijk}}{\Delta}$$

where

1. P_{ijk} is the transmittance of the k^{th} forward path from node i to node j .

2. Δ is the graph determinant and is found from

$$\Delta = 1 - \Sigma L_1 + \Sigma L_2 - \Sigma L_3 + \dots$$

in which

a. L_1 is the transmittance of all closed paths in the graph.

b. L_2 is the product of the transmittances of two nontouching loops. Loops are nontouching if they do not have common nodes. ΣL_2 is the sum of the product of transmittances of all possible combinations of nontouching loops taken two at a time.

c. L_3 is the product of the transmittances of three nontouching loops. ΣL_3 is the sum of the product of transmittance of all possible combinations of nontouching loops taken three at a time.

3. Δ_{ijk} is the cofactor of P_{ijk} . It is the determinate of the remaining subgraph when the path which produces P_{ijk} is removed.

The gain formula is often used to relate the output variable $C(s)$ of a system to the input variable $R(s)$. The formula reduces to a somewhat simplified form as

$$T = \frac{\sum_k P_k \Delta_k}{\Delta}$$

where $T(s) = C(s)/R(s)$.

Obtaining the transfer function between two points in the system from the system topology and the definition of the elements requires four steps. Step one is to use the linear relations obtained from the data structure to obtain the directed graph and weights on each arc. Two, use the directed graph to find the loops and the forward feed paths between the two points. Three, using the loops generate the cofactors for each forward path and the system determinate. Step four, use the forward paths, cofactors, determinate, and weights in Mason's gain formula.

From the information supplied by the linear relationships between system variables a representation of a digraph is constructed. The representation of the digraph is called an adjacency matrix. The adjacency matrix of a digraph D is the $m \times m$ matrix $[a_{ij}]$ with $a_{ij} = 1$ if D contains an arc from node i to node j , and 0 otherwise. The size of the matrix is determined by the number of nodes m . Since a node represents a system variable, there is a one to one correspondence of system variables to nodes. As the arcs of the graph are located and recorded another record of the weights or transfer functions is kept.

A theorem stated by Berge¹ and Harary² is used. The theorem as stated by Harary: the i, j entry $a_{ij}^{(n)}$ of A^n is the number of walks of length n from v_i to v_j . A^n denotes the adjacency matrix raised to the n^{th} power and v_i and v_j represent node i and node j respectively. A walk, as defined by Harary, is an alternating sequence of nodes and arcs in which each arc is crossed in the positive direction. Application of this theorem implies that a non-zero diagonal element a_{ii} of A^n indicates that the graph contains an n arc cycle beginning and ending at node i . At first glance application of this theorem appeared to solve the problem, but several other problems arose. One problem is that a four arc cycle may contain two two-arc cycles. Another manifestation of this problem is that a four arc path can contain two arc or three arc cycles. The problem was solved by the development of an algorithm to remove the imbedded cycles. Another algorithm was developed to discover the order of the nodes in the cycles. All forward paths were also located.

When the forward paths and loops were located, another algorithm was implemented which identified which loops have common nodes with other loops or the forward paths. With these algorithms the cofactors and determinates can be formed.

¹C. Berge, The Theory of Graphs and Its Applications, John Wiley, New York, 1962.

²Frank Harary, Graph Theory, Addison-Wesley Publishing Co., Reading, Mass., 1969.

The calculation of the forward paths, loop gains, cofactors, and determinates are done by multiple state code so that computational time is saved.

Completion of step three allows the application of Mason's loop rule. The closed loop function given by

$$T(s) = \frac{\sum_k P_k \Delta_k}{\Delta}$$

can also be calculated with multiple state code.

After all the loops of the closed loop system are found the open loop transfer function is easily calculated using these loops and the variable upon which the loop is to be opened. By opening the signal flow graph and inserting a source node and a sink node, it is possible to show that the loops of the closed loop graph that include the opening become the forward paths and the remaining loops are used to form the determinate and cofactors. The use of Mason's loop rule results in the open loop transfer function. Since the characteristic equation is the determinate of the closed loop transfer function, it is also easily calculated.

The above techniques are used to produce machine code routines which compute the coefficients of the open loop transfer function for frequency response analysis and the coefficients of the characteristic equation for the root locus analysis.