

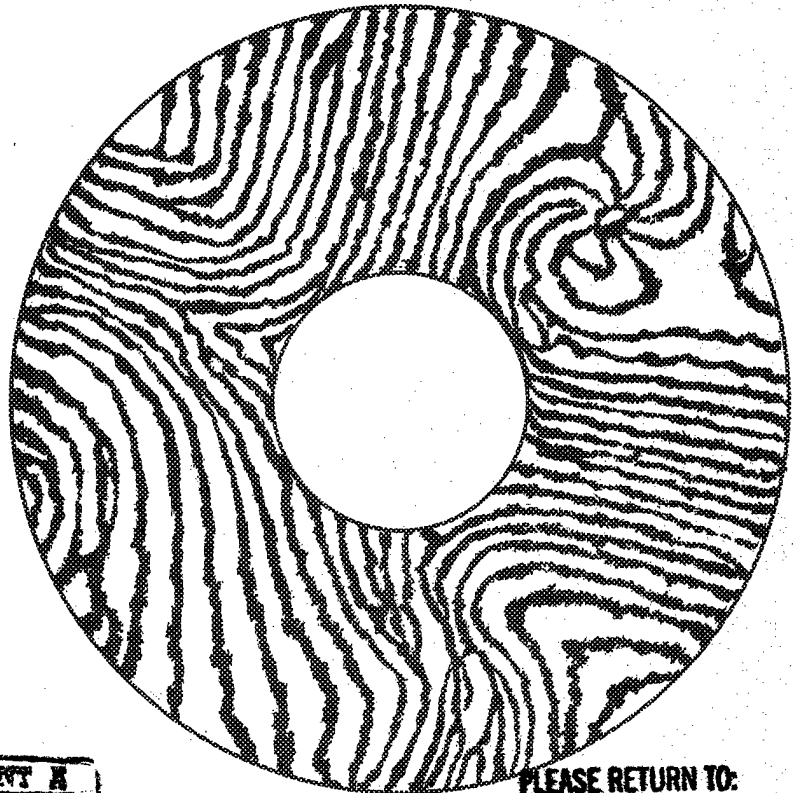
SLL 81 341

RRI/Spre Low Lib
AO,

R-704

CHARLES STARK DRAPER LABORATORY

A Division of Massachusetts Institute of Technology • Cambridge, Mass



DISTRIBUTION STATEMENT A
Approved for public release
Distribution Unlimited

PLEASE RETURN TO:

BMD TECHNICAL INFORMATION CENTER
BALLISTIC MISSILE DEFENSE ORGANIZATION
7100 DEFENSE PENTAGON
WASHINGTON D.C. 20301-7100

DTIC QUALITY INSPECTED 4

DIGITAL CONTROL SYSTEM DEVELOPMENT FOR OPTICAL MIRROR FIGURE CONTROL

DUNCAN MAC KINNON

19980309 348

u 4098

Accession Number: 4098

Publication Date: Dec 01, 1971

Title: Digital Control System Development for Optical Mirror Figure Control

Personal Author: MacKinnon, D.D.; Desai, M.N.; Freiburghouse, E.K.; et al.

Corporate Author Or Publisher: Charles Stark Draper Laboratory, Cambridge, MA 02139 Report Number: R-704

Report Prepared for: National Aeronautics and Space Administration, Marshall Space Flight Center, AL Report Number Assigned by Contract Monitor: SLL 81-341

Comments on Document: Archive, RRI, DEW

Descriptors, Keywords: Digital Control System Development Optical Mirror Figure Primary Environment Disturbance Diffraction Limit Telescope Algorithm Sensor Actuator Structure Simulation

Pages: 285

Cataloged Date: Dec 10, 1992

Contract Number: NAS 8-27620

Document Type: HC

Number of Copies In Library: 000001

Record ID: 25666

Source of Document: DEW

R-704

DIGITAL CONTROL SYSTEM DEVELOPMENT FOR
OPTICAL MIRROR FIGURE CONTROL

by

D. Duncan MacKinnon

and

Mukund N. Desai

Eugenia K. Freiburghouse

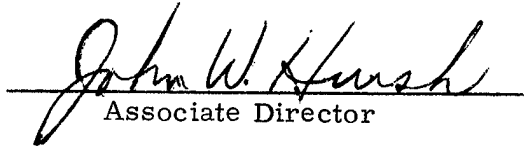
Paul A. Madden

Keto Soosaar

December, 1971

THE CHARLES STARK DRAPER LABORATORY
A DIVISION OF MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE, MASSACHUSETTS 02139

Approved:


Associate Director

Approved:


Deputy Director

ACKNOWLEDGMENT

The senior author wishes to acknowledge the sponsorship of Dr. Joseph Randall of the National Aeronautics and Space Administration, Marshall Space Flight Center (MSFC), whose interest in the application of modern control and structural theory to active primary mirror figure control led to the establishment of the project reported herein. The author also expresses his thanks to Dr. William Clarke of MSFC who monitored the project and contributed a number of helpful suggestions. The contribution of Howard Newberry of MSFC and Dick Bean and Pat Brown of the Computer Science Corporation to the development and checkout of the computer software is also gratefully acknowledged. The author also wishes to thank Charles Wyman and Charles Jones of MSFC for their current and previous support of this project.

Co-authors Mukund Desai, Paul Madden and Keto Soosaar contributed significantly to the analytical investigations described herein and Eugenia Freiburghouse assisted in the preparation of programs and program descriptions.

This report was prepared under DSR Project 55-46500 sponsored by the Marshall Space Flight Center of the National Aeronautics and Space Administration through contract NAS 8-27620.

The publication of this report does not constitute approval by the National Aeronautics and Space Administration of the findings or the conclusions contained therein. It is published only for the exchange and stimulation of ideas.

DIGITAL CONTROL SYSTEM DEVELOPMENT FOR OPTICAL MIRROR FIGURE CONTROL

ABSTRACT

The maintenance of accurate primary mirror figure in the face of environmental disturbances is the key to the achievement of diffraction-limited performance in a large space telescope. In order to develop the concepts of optical mirror figure control, an experimental program has been initiated at the Marshall Space Flight Center, Huntsville, Alabama. A major component in this experiment will be an XDS Sigma 5-2 multi-digital computer system which will realize the mirror figure control algorithm.

Development of the control system for the experimental active mirror was initially described in two earlier MIT/DL reports in this series.^{1,2} This report extends the previous work in several areas. Figure control laws, suitable for digital computer implementation, have been designed and incorporated in a very flexible software package. A figure control system simulation capability was achieved by including models of the figure sensor, figure actuators, mirror structure and a simulation control module in the software package. This permits the figure sensor control software to be completely checked out and evaluated using the simulation before interface with the actual hardware components is attempted.

TABLE OF CONTENTS

<u>Chapter</u>		<u>Page</u>
1.	AN EXPERIMENTAL ACTIVE MIRROR	1
1.1	Introduction.	1
1.2	Experiment Design Concept	3
1.3	Software Design Considerations	5
2.0	EXPERIMENTAL ACTIVE MIRROR COMPONENT MODELS	7
2.1	Introduction.	7
2.2	Optical Figure Sensor	7
2.3	Optical Figure Sensor Model	10
2.4	Ambiguity Sensor	12
2.5	Ambiguity Sensor Model	13
2.6	Mirror Figure Actuators	15
2.7	Force Actuators.	15
2.8	Position Actuators.	18
2.9	Actuator Models.	18
2.10	Deformable Mirror Model	20
2.10.1	Introduction.	20
2.10.2	Approach	24
2.10.3	Results.	25
2.10.4	Conclusions.	30
2.11	Segmented Mirror Model.	31
3.	EXPERIMENTAL ACTIVE MIRROR FIGURE CONTROL ALGORITHMS.	35
3.1	Introduction.	35
3.2	Deformable Mirror Control Laws	36
3.3	Segmented Mirror Control Laws	38
3.4	Discrete Control Algorithm for Mirror Figure Control .	39
3.5	Figure Sensor Data Processing Algorithm	44
3.6	Force Actuator Control System	50

TABLE OF CONTENTS (cont)

<u>Chapter</u>		<u>Page</u>
3.7	Position Actuator Control System	50
3.8	Initial Active Mirror Alignment	53
3.9	Tilt Alignment System	54
3.10	Slew Alignment Algorithm	56
4.	EXPERIMENTAL ACTIVE MIRROR SOFTWARE . . .	59
4.1	Introduction.	59
4.2	Supervisory Software	62
4.3	Cataloging Transfer Data.	67
4.4	Determination of the Transfer Address	70
4.5	Experimental Active Mirror Simulation	70
4.6	Experimental Active Mirror Experiment Operation . .	72
4.7	EAM Software Descriptions and Functional Block Diagrams.	74
4.7.1	ACTCAL: Actuator Calibration.	74
4.7.2	ACTCMD: Figure Actuator Control Systems	76
4.7.3	ACTMDL: Mirror Figure Actuator Models.	76
4.7.4	EAMCS: Hardware Component Control Software . . .	76
4.7.5	FIGSEN: Figure Sensor Control Module	77
4.7.6	FSMDL: Figure Sensor Model	77
4.7.7	MAINA: EAM System Computations.	77
4.7.8	MAINB: Remote Control Operations.	78
4.7.9	MAINC: Initial Alignment Computations	78
4.7.10	MFCS: Control System Data Input and Gain Matrix Computation	78
4.7.11	MIRCAL: Mirror Calibration.	78
4.7.12	MIRMDL: Mirror Model.	79
4.7.13	PINDX: Performance Index Generator.	79
4.7.14	PLRT: Plotted Data Storage and Scaling.	80
4.7.15	RESPON: Simulation Control Software.	80

TABLE OF CONTENTS (cont)

<u>Chapter</u>		<u>Page</u>
4.7.16	SIMSYS: Program Control Module	81
4.7.17	STORED: Plotting Control Software	81
4.7.18	SUPE2: Main Sigma 2 Program.	81
4.7.19	SUPE5: Main Sigma 5 Program.	81
4.7.20	TYPCON: Remote Terminal Control Software	147
5.	EXPERIMENTAL ACTIVE MIRROR INPUT DATA.	149
5.1	Introduction.	149
5.2	Input Data Deck Description	149
5.2.1	Output Data Heading	150
5.2.2	Operating Modes	150
5.2.3	Simulation Control Data	158
5.2.4	Run Set Identification and the Number of Runs	159
5.2.5	Device Assignment for Manual Simulation Control.	160
5.2.6	Plotting Data	161
5.2.7	Peripheral Device Assignment	161
5.2.8	Mirror Model Data	162
5.2.9	Figure Sensor Data	163
5.2.10	Actuator Position Data	164
5.2.11	Actuator Scale Factors.	164
5.2.12	Figure Control Algorithm	164
5.2.13	Actuator Test Data	165
5.2.14	Mirror Calibration Test Data	165
5.2.15	Real Time Control System Parameters	165
5.2.16	Figure Sensor Filter Parameters	166
5.2.17	Figure Sensor Model Data	166
5.2.18	Actuator Model Data	167
5.2.19	Mirror Model Data	167
5.2.20	Performance Index Data	168
5.2.21	Initial Alignment Control System Data	169

TABLE OF CONTENTS (cont)

<u>Chapter</u>		<u>Page</u>
6.	REMOTE CONTROL OF THE EXPERIMENT	170
6.1	Introduction	170
6.2	Experiment Control Commands	170
6.3	Experiment Monitoring Capability	173
7.	SUMMARY	175
7.1	General Features of the Digital Control System	175
Appendix		
A.	EXPERIMENTAL ACTIVE MIRROR SOFTWARE LISTINGS.	178
B.	EAM LIBRARY ROUTINE LISTINGS	235
Table of References		273
Table of Symbols.		275

CHAPTER 1

AN EXPERIMENTAL ACTIVE MIRROR

1.1 Introduction

Astronomical observations through a large earth-based telescope suffer from limitations placed on the resolving power of the telescope by fluctuations in the earth's atmosphere. As part of a space orbiting astronomical laboratory, however, a telescope would not be subject to these limitations. A large instrument which is diffraction-limited over a major part of its useful spectrum of observation is envisioned. Maximum resolving power requires extremely accurate maintenance of the figure of the primary mirror.¹⁻⁹

Although it is possible to polish a large mirror to the desired surface accuracy, stresses introduced by thermal variations in the mirror and fluctuations in support structure loads, structural instability and the elimination of gravity loading in orbit could create surface perturbations which would exceed the surface accuracy limits required for diffraction limited performance. As a result, investigators have attempted to develop techniques for actively correcting the mirror figure in a space environment and a number of promising control techniques have been developed.¹⁻⁹ The development and application of these control concepts is one of the key challenges facing the designer of the large space telescope.

The development of Mirror Figure Control Systems at the M. I. T. Draper Laboratory was initially on a theoretical level which defined hypothetical analytical models of the various figure control system components and the control algorithms necessary to achieve figure control.¹ An investigation was also made of the potential

improvement in rms deformable mirror figure accuracy as a function of the number of actuators used and their arrangement. These studies indicated the basic feasibility of figure control and the fact that substantial improvements in figure accuracy could be achieved even with a relatively modest figure control system realization. While the large scale digital computer is an extremely valuable tool for the analysis and simulation of complicated mirror figure control systems, the results obtained are only as reliable as the modelling accuracy of the physical components in the system. Accurate modelling requires a considerable amount of intuition if the trade-off between modelling accuracy and computation time and analytical difficulty are to be resolved satisfactorily. Often, terms neglected in the modelling process are of key importance to the overall system design.

To resolve these problems it is important to have some way of checking the results of numerical analysis against actual system behavior. Such checks are furnished by an experimental program.

Experimental work in the past has been largely conducted using analog devices to synthesize figure actuator commands from surface error measurements. As a result of the expense and time associated with programming a general purpose analog computer or constructing a special purpose analog system, it has been difficult to explore the full spectrum of control solutions or efficiently process experimental data.

In response to these problems, a more efficient experimental tool has evolved in the hybrid digital analog computer system. Spurred by declining cost hybrid computing systems are appearing in a wide variety of laboratory environments. Software has been developed at MIT/DL which permits a hybrid digital computer to provide the sensor signal processing and control computations for an experimental

active mirror figure control system.

1.2 Experiment Design Concept

The experimental system consists of the primary mirror fitted with actuators for figure modification, a mirror figure sensor and an "on line" control system processing the figure errors measured by the sensor to provide proper corrective signals to the actuators.

Following a modern approach a digital computer has been selected as the control processor of the experimental system. The utilization of a general purpose computer has a number of advantages, including:

1. Programmability permitting a large number of different primary mirror control configurations to be investigated without extensive hardware modification.
2. The ability to handle a number of auxiliary tasks such as experimental data processing and display.
3. Characteristics similar to the system computer which will be used in an orbiting astronomical observatory.

A simplified block diagram of an experimental active mirror is shown in Figure 1.2.1. The digital computer consists of a central processor, a random access core memory and an input-output processor. The central processor handles arithmetic operations, logic operations and some data transfer operations using instructions extracted from the core memory. The input-output processor controls the transfer of information from the central processor and core memory to computer peripherals which are part of the interface with the real world. The core memory holds two types of stored information, program instructions and program storage. The program instructions tell the computer what to do with information extracted from program

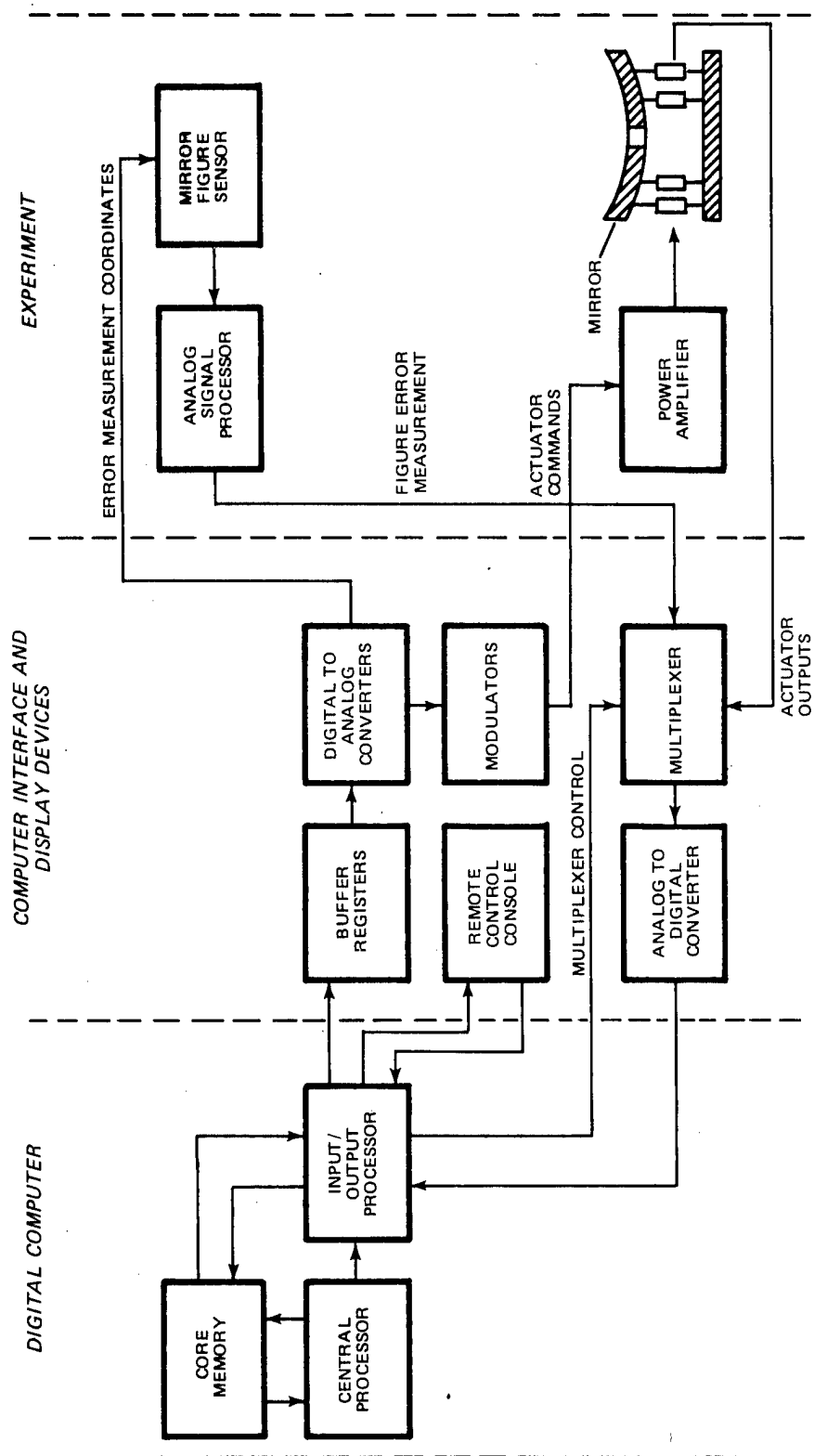


Fig. 1.2.1 A digital control system for an experimental active mirror.

storage and input devices.

The interface consists of a number of components. The digital output buffers are a set of addressable registers which temporarily store digital information, which is converted to an analog signal by the digital to analog converters. The output signals from the digital to analog converters provide measurement coordinate signals for the figure sensor image dissector and actuator command signals which are modulated by DC-AC modulators. The modulator outputs are amplified by the actuator power amplifiers to provide sufficient energy to drive the actuator motors. The state of the hardware is observed by the computer via means of the actuator output sensor and the figure sensor error measurements. The analog figure sensor and actuator output signals pass through a computer-controlled switch or multiplexer; the single output of the multiplexer is converted to a digital signal by the analog to digital converter. The digital signal is then transferred to the central processor or the correct location in the core memory by the input-output processor.

1.3 Software Design Considerations

The actual experimental system is complicated by the use of two computers rather than one as indicated in Fig. 1.2.1. The real time control computations are performed in an XDS Sigma 2 computer which is available to the experiment on a dedicated basis. The relatively small memory capacity of this computer severely limits the scope of the program which it can successfully execute. As a result, it was decided to use a second, more powerful, computer, the XDS Sigma 5 to perform most of the complex arithmetic and data processing operations on a time-shared basis. This decision necessitated the development of an elaborate software module to transfer program control and data from the 5 to the 2 and vice versa.

A digital simulation capability was also realized to provide numerical data for comparison with experimental results and to aid in the development of figure control strategies. The real time control software for the EAM is an integral part of the simulation which in essence substitutes mathematical models for the actual hardware components.* Selection of the simulation mode, hardware control mode, and various other operating configurations is accomplished via a set of mode selection variables.

* Using the techniques successfully employed during the Apollo program.

CHAPTER 2

EXPERIMENTAL ACTIVE MIRROR COMPONENT MODELS

2.1 Introduction

The hardware components of the experimental active mirror consist of devices to measure the figure error, figure sensors, figure actuators for effecting changes in the mirror figure and the primary mirror structure itself.

In order to analyze and to simulate the experimental active mirror it is necessary to develop mathematical models of the system hardware components. This chapter describes the major EAM components and presents simplified mathematical models which characterize their operation.

2.2 Optical Figure Sensor

The figure error measurement function is provided by the optical Figure Sensor.^{8,10} The Figure Sensor is a modified Twyman-Green two-beam interferometer. The interferometer utilizes a laser to produce a coherent beam of light. The plane laser wavefront enters a beamsplitter where it is divided into two beams. One, a reference beam, is allowed to illuminate a plane reference mirror, the second beam is passed through an aspheric decollimating lens which creates a spherical wavefront illuminating the unobscured aperture of the mirror under test. In the case of a spherical primary mirror, coincidence is maintained between the centers of curvature of the primary mirror figure and the spherical wavefront emanating from the decollimator. The reflected energy from the primary returns through the decollimator where collimation occurs. The collimated wavefront mixes with the reflected energy from the reference flat mirror at the beamsplitter producing an interference pattern which is imaged on the face of an image dissector tube.

Irregularities in the spherical primary mirror figure result in a nonspherical returning wavefront and ensuing fringes in the interference pattern. Since a one to one relationship exists between position in the interference pattern and position on the mirror surface it is possible to identify the location of figure errors.

The actual measurement of the magnitude of the figure error is accomplished electronically. Suppose that the reference mirror is mounted on a piezoelectric crystal arranged so that the introduction of an electric field produces an axial translation of the mirror varying the optical path difference between the two arms of the interferometer. The variation in path produces a sinusoidally related change in the interference pattern intensity. The current system utilizes a triangular driving signal to modify the path length which results in a number of cycles variation in the intensity level. As a result, interference pattern information provided by the image dissector is of a sinusoidal nature. The frequency associated with the intensity variation is the product of the optical path difference modulator drive frequency times the number of complete cycles of intensity variation during each complete path length modulation cycle. The sinusoidal variation in the interference pattern at a designated point, sampled by a photodiode, is used to provide a reference signal. The phase difference between the intensity variation at the reference point and a measurement point in the interferogram, observed by the image dissector, is proportional to the figure error at the measurement location. Measurement ambiguity arises from the inability of the phase detector to differentiate phase shifts which are multiples of 360 degrees.

The output of the figure sensor contains noise which arises from mechanical vibrations and internal sources within the optical and electronic components. The noise appears to be adequately modelled by a white noise superimposed on the phase detector input.

The most important characteristics of the Figure Sensor are summarized in Table 2.2.1.

TABLE 2.2.1
FIGURE SENSOR PARAMETERS

PARAMETER	VALUE
OPERATING WAVELENGTH	632.8 nm
PHASE DETECTOR CARRIER FREQUENCY	180 hz
PATH DIFFERENCE MODULATOR FREQUENCY	18 hz
PATH DIFFERENCE MODULATION AMPLITUDE	3164 nm
PHASE DETECTOR FILTER TIME CONSTANT	0.159 sec
ABSOLUTE FIGURE SENSOR ACCURACY	6.32 nm* (after calibration)
FIGURE SENSOR NOISE LEVEL	3.16 nm* rms

* Perkin Elmer performance goals

2.3 Optical Figure Sensor Model

A model of the Figure Sensor has been developed which provides a reasonably good approximation to the actual characteristics of the Figure Sensor. A block diagram of the Figure Sensor model is illustrated in Fig. 2.3.1. The model consists of a Gaussian white noise generator which superimposes noise on the actual figure error β_{xf} , a phase detector model which produces an output β_p equivalent to that provided by the Figure Sensor phase detector and a first-order filter which smooths the phase detector output.

Let the actual figure error at the measurement point coordinates be designated β_{xf} . If the noise superimposed on the figure error is β_{nf} , a suitable form for the phase angle generator is

$$\beta_p = \begin{cases} \beta_{xa}, & |\beta_{xa}| < \frac{\lambda}{4} \\ \beta_{xa} - \frac{\lambda}{2}, & \beta_{xa} \geq \frac{\lambda}{4} \\ \beta_{xa} + \frac{\lambda}{2}, & \beta_{xa} \leq -\frac{\lambda}{4} \end{cases} \quad (2.3.1)$$

where

$$\beta_{xa} = \beta_{xf} + \beta_{nf} \quad (2.3.2)$$

and β_p is the phase detector output. Note that this representation will only provide a useful phase output for figure errors in the range

$$|\beta_{xf}| < \frac{3\lambda}{4} \quad (2.3.3)$$

This restriction was introduced to simplify model computations while still permitting sufficient range to investigate the measurement ambiguity problems which arise at figure errors of $\frac{\lambda}{4}(1 + 2i)$ where i is an integer.

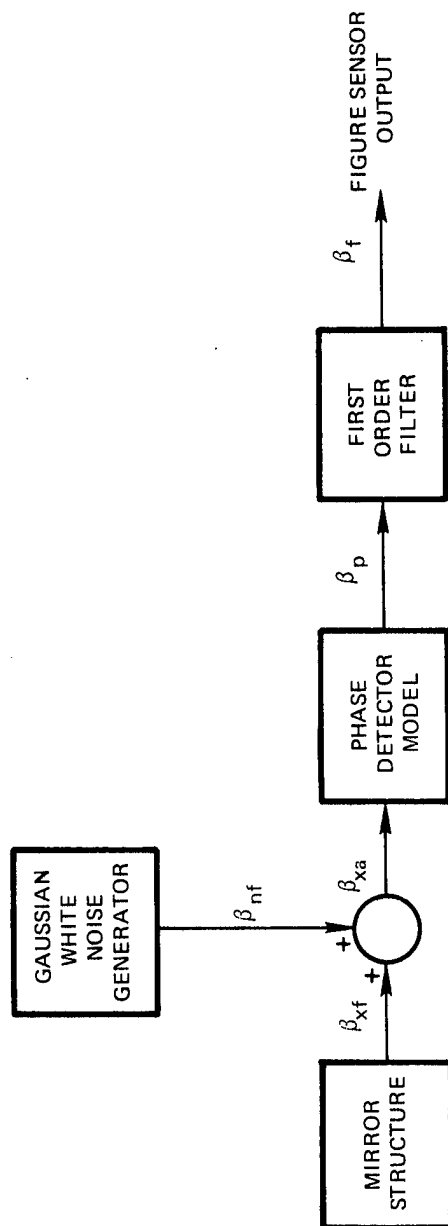


Fig. 2.3.3.1 Figure sensor model.

The noise input β_{nf} is a Gaussian white noise with zero mean and a standard deviation σ_f .

The figure sensor phase angle filter is modelled by a first-order time lag. A discrete representation of the filter was utilized to save computer time and improve accuracy. The filter has the form

$$\beta_f(i+1) = \phi_f \beta_f(i) + \gamma_f \beta_p(i) \quad (2.3.4)$$

$$\beta_f(1) = 0 \quad (2.3.5)$$

where $\beta_f(k)$ and $\beta_p(k)$ are the filter input and output at time t_k and

$$t_{k+1} = t_k + \Delta t \quad (2.3.6)$$

where Δt is the real time control loop cycle time. The state and input transition parameters ϕ_f and γ_f are given by

$$\phi_f = e^{-\frac{\Delta t}{t_f}} \quad (2.3.7)$$

$$\gamma_f = 1 - \phi_f \quad (2.3.8)$$

where t_f is the time constant of the first-order lag.

2.4 Ambiguity Sensor

Axial alignment of the individual segments to assure that the resultant figure lies on the surface of a sphere centered on the figure sensor decollimator requires measurement of the relative position of adjacent segment edges. While the Figure Sensor can provide accurate

figure measurements over a continuous surface it is unable to resolve multiple half-wavelength ambiguities at the discontinuity presented by the divisions between adjacent segments. To eliminate this problem an additional sensing device has been added to the segmented mirror system.

The ambiguity sensor is a modification of the Michaelson interferometer spectrometer in which the interferogram produced by varying the relative length of a two-beam interferometer may be analyzed to determine the spectral content of the excitation source.¹³ In the case of the segmented mirror the interferometer is mounted across the adjacent segment edges as indicated in Fig. 2.11.1. Light provided by a tungsten arc source is allowed to impinge on the adjacent segment reflecting surfaces. A Koester prism performs the functions of beam-splitting and recombination. The broad spectrum of the source results in a zero-order interference lobe which is readily recognizable by a peak in intensity. Signal processing is simplified by modulating the arc source with a mechanical chopper permitting the use of ac signal processing techniques. Two lead sulphide detectors are used to examine the interference pattern: one providing a reference signal while the other observes the interference pattern. The difference between the detector outputs is a measure of the intensity of the interference lobe.

2.5 Ambiguity Sensor Model

Assume that an array of measurement points x_f have been defined on the surface of the segmented mirror. Suppose that six of the points coincide with or are near the areas observed by the white light interferometers. The six point elements of x_f are conveniently identified by the elements of a 2×3 array L_f . Thus ambiguity sensor 1 observes measurement points ℓ_{11} and ℓ_{12} and so forth.

A suitable model for the i th ambiguity sensor may be obtained by expanding the ambiguity sensor output as a function of the difference between the ℓ_{i1} and ℓ_{i2} elements of x_f in a Taylor's series. Let the i th ambiguity sensor output be a_{si} . Then

TABLE 2.5.1

AMBIGUITY SENSOR MODEL PARAMETERS

β_{sm}	1.0
β_{sd}	$-\frac{4.0}{\lambda^2}$

$$a_{si} = \beta_{sm} + \beta_{sd} \left[(x_f)_{\beta i, 1} - (x_f)_{\beta i, 2} \right]^2 \quad (2.5.1)$$

where β_{sm} is the maximum output of the sensor and β_{sd} is a negative parameter. Note that this model is only capable of adequately representing the zero order node which has a width of approximately ± 316 nm ($\lambda/2$ at 632.4 nm). This is a reasonable restriction since errors in excess of 316 nm would result in axial alignment to the peak of the second node. Note that this implies that initial manual axial alignment must be performed to at least an accuracy of 316 nm. The values selected for β_{sm} and a_{si} are summarized in Table 2.5.1.

2.6 Mirror Figure Actuators

Modification of the primary mirror figure is induced by mechanically perturbing the surface of the mirror. In the case of the deformable mirror figure, modification is produced by an array of controllable loads which elastically deform the mirror structure. The loads act virtually parallel to the optical axis. The segmented active mirror, on the other hand, utilizes mechanical displacement of the individual segments to improve the overall mirror figure.

2.7 Force Actuators

A functional block diagram of the force actuator is shown in Fig. 2.7.1. The actuator servo accepts an angular velocity command from the actuator digital to analog converter channel. The commanded velocity modulates a 400-Hz signal to produce a velocity signal ω_c^* . The transfer function relating the rms modulator output to the dc input signal is assumed to be a constant k_{mod} . The output of the modulator is compared with a corresponding 400-Hz signal ω_a^* which is proportional in amplitude to the angular velocity ω_a of the servo-motor shaft output shaft. The servo motor produces a torque which is proportional to k_t

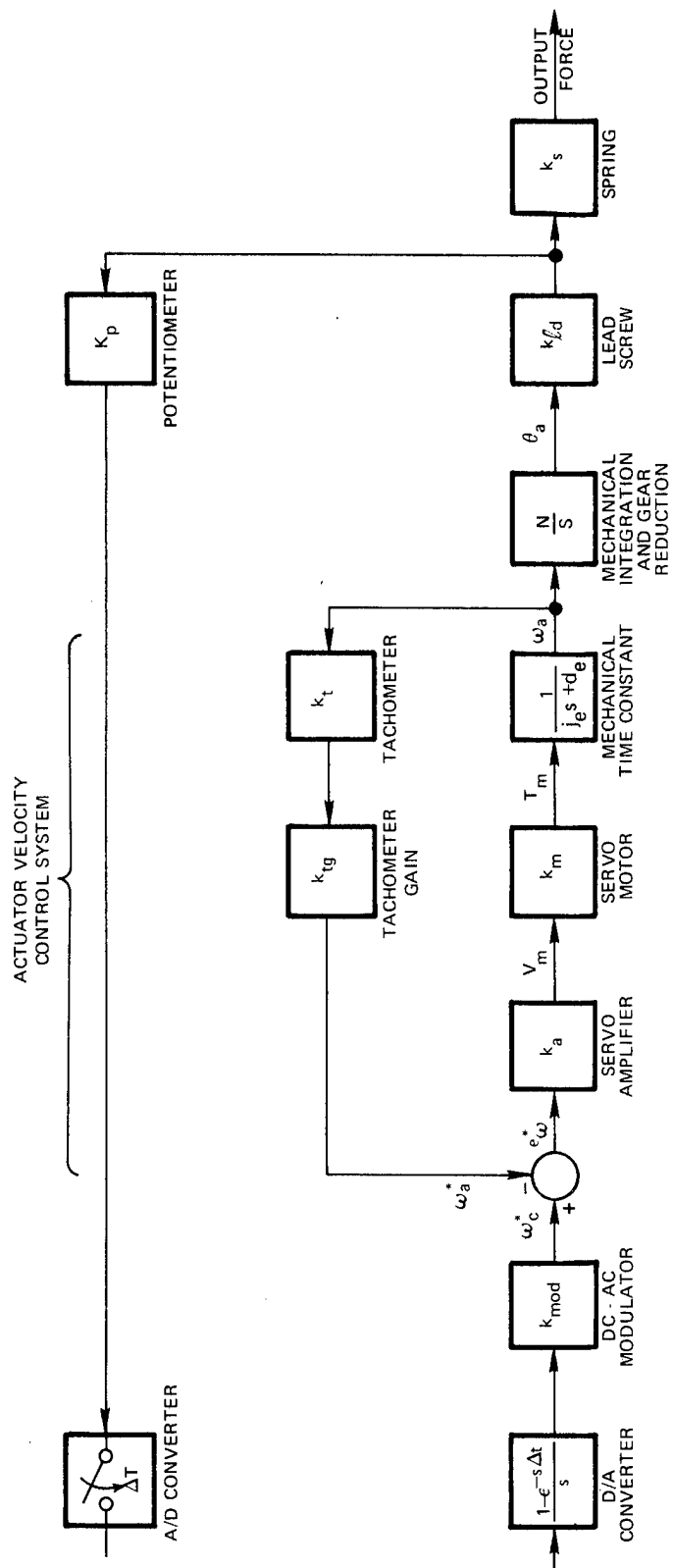


Fig. 2.7.1 Force actuator block diagram.

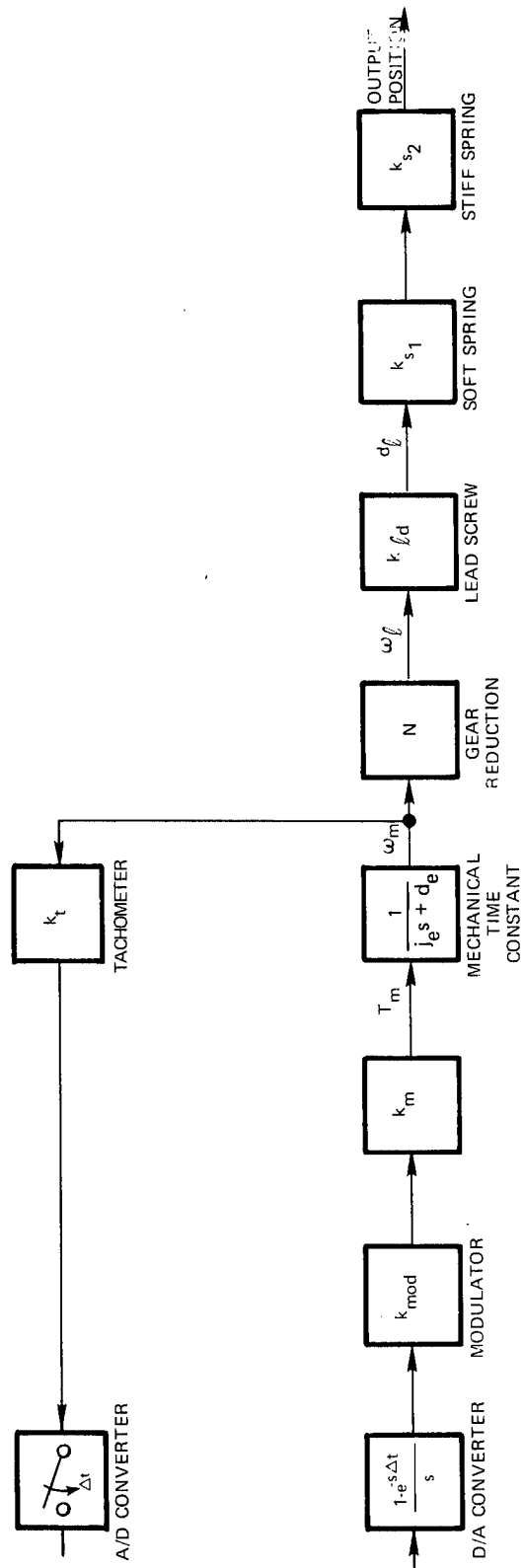


Fig. 2.8.1 Position actuator block diagram.

times the amplitude of the error signal e_{ω}^* . The direction of the torque T_m depends upon whether or not e_{ω}^* is in-or-out-of-phase with a 400-Hz reference signal. The torque accelerates the equivalent motor armature and load inertia j_e subject to the damping torque $d_e \omega$. The angular rotation of the motor shaft produces a reduced rotation at the output of a gear reduction. Conversion to linear motion is achieved using a lead screw. The linear movement of the threaded collar on the screw compresses a spring which provides the desired actuator output force. A linear potentiometer slider is connected to the collar to provide a position feedback signal which is proportional to the actuator output force.

2.8 Position Actuators

The position actuators are in some respects similar in design to the force actuators. A motor-tachometer drives a lead screw through a gear reduction. A threaded collar converts the rotation of the screw to linear motion compressing a soft spring. The change in load produced by the fine spring alters the length of a very stiff spring. The change in length of the stiff spring produces a corresponding displacement in the mirror segment through a kinematic support point. The main difference stems from the absence of an actuator output position feedback sensor, a change which alters the character of the control command required to drive the actuators. Actuator position control is achieved by commanding an output velocity pulse whose total area equals that of the desired change in position. The commanded velocity drives a velocity control loop which is closed in the Sigma 2. A block of the segment position actuator is shown in Fig. 2.8.1.

2.9 Actuator Models

The actuator control systems are characterized by a fast inner loop which controls the angular velocity of the tachometer-motor and a relatively slow outer loop which controls the position of the threaded

lead screw collar. As a result it was decided to model the transfer function between the desired, m_c , and actual, m_m , actuator outputs by a simple first-order system. For the i th actuator

$$\dot{m}_{mi} = \frac{1}{t_{ai}} [m_{ci} - m_{mi}] \quad (2.9.1)$$

The time constants $t_{ai} = 1/n_r$ are read in as elements of the array TACTV. In order to eliminate the need for numerical integration the actuator dynamics were represented by the equivalent discrete equation

$$m_{mi}(i+1) = \phi_{ai} m_{mi}(i) + \gamma_{ai} m_{ci}(i) \quad (2.9.2)$$

where $m_{mi}(k)$ and $m_{ci}(k)$ are the values of m_{mi} and m_{ci} at t_k and

$$t_{k+1} = t_k + \Delta t \quad (2.9.3)$$

where Δt is the actuator control system cycle time. The state and input transition parameters ϕ_{ai} and γ_{ai} satisfy the equations

$$\phi_{ai} = e^{-\frac{\Delta t}{t_{ai}}} \quad (2.9.4)$$

$$\gamma_{ai} = 1 - \phi_{ai} \quad (2.9.5)$$

where t_{ai} is the time constant associated with the i th actuator.

2.10 Deformable Mirror Model

2.10.1 Introduction

A linear model of the deformable mirror is desired which relates the displacements x_f sensed by the figure sensor at an array of measurement points to the loads m_m applied parallel to the optical axis by an array of force actuators. The linear transformation is conveniently expressed in the form

$$x_f = A m_m \quad (2.10.1)$$

where A is an $n \times n$ matrix.

A linear model of the form (2.10.1) is conveniently generated by representing the mirror by an approximate structural model consisting of a large number of finite elements as indicated in Fig. 2.10.1. The node points (joints) of the finite element representation are selected to coincide with the actuator and measurement point locations. (While the actuator locations are fixed in this particular example the measurement points may be reassigned to coincide with a desirable set of nodes.)

The i th column of the matrix A is generated by applying a unit load at actuator location i and calculating the resulting deformations at the measurement locations. The finite element algorithm provides deformations parallel and normal to the optical axis. The normal deflections may generally be neglected for a thin shallow shell.

The figure sensor detects differences in the length of radii joining the measurement point and a reference point to the center of curvature of the spherical wavefronts emerging from the decollimator. In the case of the deformable mirror the reference point is normally selected to coincide with a point on the mirror's surface corresponding to a rigid support location. As a result the reference radius may be considered constant and

the measurement provided by the figure sensor is the actual change in length of the measurement point radius.

Thus if the computed deflection parallel to the optical axis is $\hat{x}_{fk}$ at the k th measurement point at a distance d_k from the optical axis the deflection sensed by the figure sensor is

$$x_{fk} = \hat{x}_{fk} \cos \gamma_d \quad (2.10.2)$$

where

$$\gamma_d = \sin^{-1} \frac{d_k}{R} \quad (2.10.3)$$

The data generated by Rackley¹⁵ and the models presented in this section do not consider this transformation and (2.10.1) and (2.10.3) should be applied to the data presented in this section in order to achieve a more exact linear model for simulation.

A number of computations were performed to obtain the flexibility matrix for the 20" NASA/MSFC active mirror which is physically described in Figs. 2.10.1 and 2.10.2. The objective here was to determine the differences, if any, between results obtained by MSFC using the NASTRAN system and the results of analysis by the ICES-STRUDL II finite element analyzer.

The finite element approach is a very powerful numerical approximation technique, but at the same time the results can be quite sensitive to the way in which the element model is defined. Since some of the mirror control algorithms are highly dependent on the accuracy of the flexibility matrix, it is of great interest to

Mirror $\phi = 20.0^\circ$
 Mirror $t = 0.668$
 Radius of
 Curvature = 80.33"
 $R_1 = 5.07$
 $R_2 = 9.48$

FUSED SILICA
 $E = 10,600,000 \text{ PSI}$
 $\text{POISSON} = 0.17$

▣ SUPPORTS
 ▲ ACTUATORS

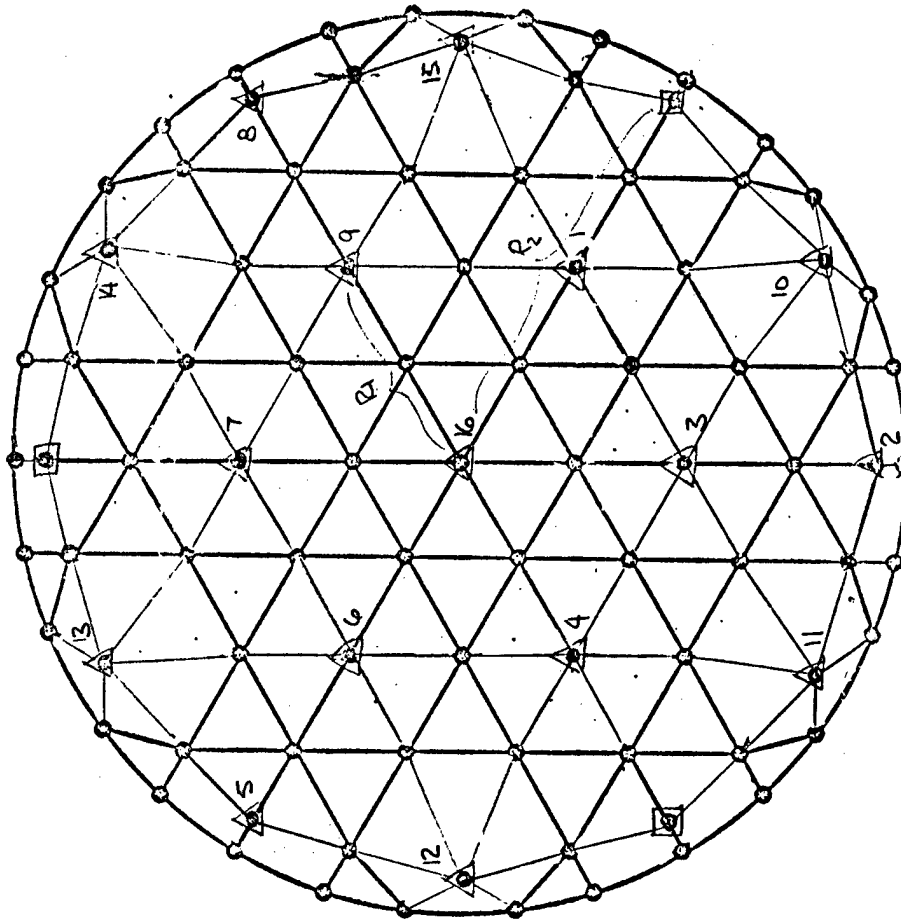


Fig. 2.10.1 Finite element deformable mirror model.

MIT Node #	MJFC Actuator #
62	1
46	2
44	3
25	4
5	5
23	6
41	7
75	8
60	9
64	10
27	11
7	12
21	13
58	14
77	15
85	16

" 10 - NODES
 " 11 - ELEMENTS
 " 12 - SUPPORTS
 " 13 - ACTUATORS

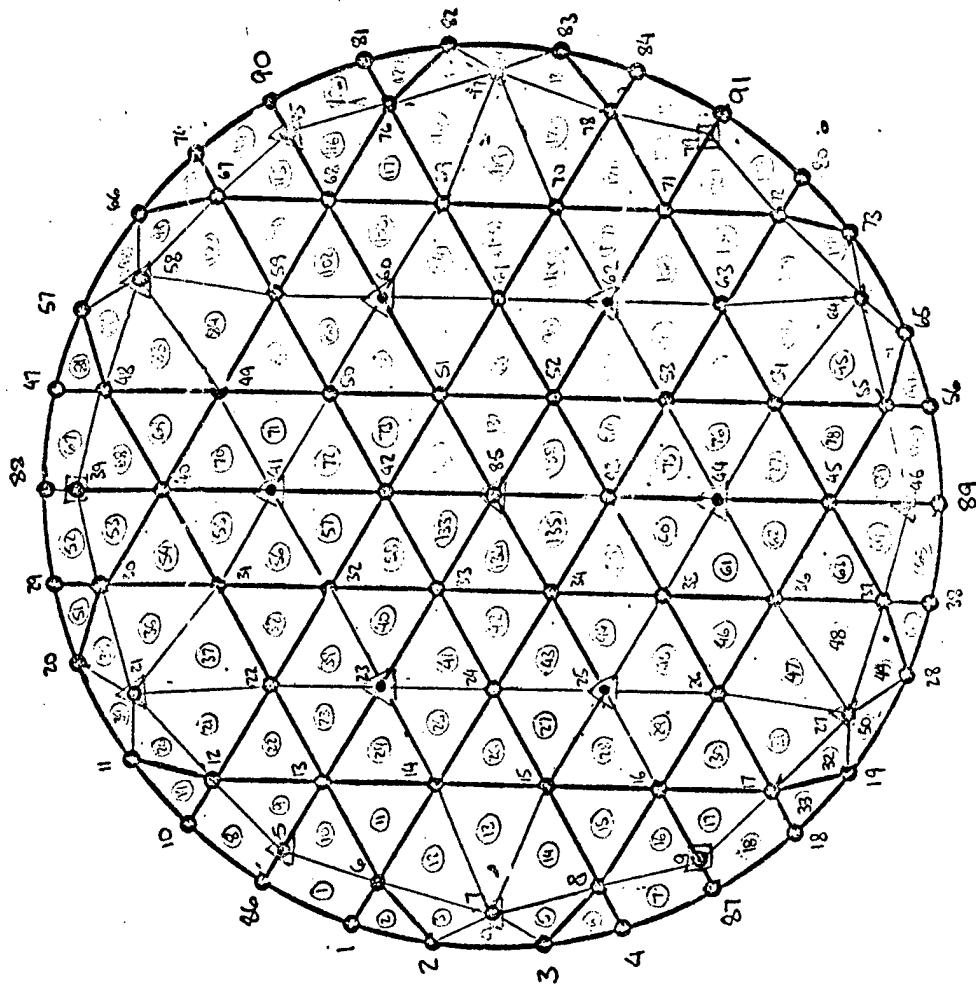


Fig. 2. 10.2 Finite element deformable mirror model.

measure the possible range of errors that may be expected in the use of the finite element technique here.

2.10.2 Approach

The ideal model for finite element shell analysis involves the use of individually curved elements where the bending and stretching actions are coupled. Such elements are still, however, in the highly experimental stage and are unavailable in any of the more common large-scale analyzers, including both NASTRAN and ICES-STRUDL II. The next best approach is to use flat plates where the bending and stretching are uncoupled, but as a result, more individual elements should be used. This latter approach was the one employed by MIT/CSDL as well as NASA/MSFC.¹⁵

The model chosen for the studies was one using primarily triangular elements in a configuration previously tested against limiting closed-form solutions and found to perform quite satisfactorily.¹⁶ "CPT"¹⁷ elements were chosen for the bending triangles, and "PBQ1" for the bending quadrilaterals. The latter element is made up of four "CPT's". For the stretching components "CSTG" and "PSQ1" elements were employed. All of these elements are identical in function to the "CQUAD2" and "CTRIA2" elements found in the NASTRAN model. The number of elements used for the STRUDL study was less than with NASTRAN, but previous studies with the STRUDL model has shown that the number was already adequate.

The mirror was analyzed first as a flat plate with bending action only, then the curvature and the stretching component were included to represent the actual shell. As significant differences appeared between the STRUDL and NASTRAN shell results, the element modelling in NASTRAN (as per NASA/MSFC memo referenced

above) was tested using the STRUDL system and elements. Due to funding limitations, this could be done for the bending case only.

2.10.3 Results

Table 2.10.1 summarizes the STRUDL finite element bending behavior results for the flexibility matrix. The complete matrix was recorded to simplify error detection. While there are a total of sixteen actuators, there are only five independent ones, the behavior of the rest may be found by various symmetry conditions. In the STRUDL analysis, all sixteen were analyzed so that any unsymmetries in the modelling that had occurred by chance error could be immediately detected. The results in Table 2.10.1 are symmetrical to about an average of 0.5%.

Table 2.10.2 summarizes the results of the shallow shell representation of the mirror. Again, good symmetry has been attained, but the differences between the plate and the shell are rather striking. In general, the shell is stiffer, with the typical decrease in stiffness near a free edge which affects shells more than plates. Subsequently, the differences at the center for the plate and shell are more noticeable than at the edges. The average difference between the plate and shell deformations is about 60%.

Table 2.10.3 summarizes the results of the NASTRAN model using STRUDL elements. Symmetry is very good again, and differences between the comparable Tables 2.10.1 and 2.10.3 are in the range of one to three percent. This is approaching the best numerical range that might be expected using finite element methods. Modelling configuration therefore does not seem to contribute significant error levels.

Table 2.10.1 FINITE ELEMENT RESULTS - BENDING MODEL $(\Delta i_j) \times 10^4$ IN/LB = DEFLECTION AT i FROM LOAD AT j

1. ELEMENT #	(2) ACTUATOR 10-NO. 195FC															
	42	46	44	75	5	23	41	78	60	64	27	7	21	58	17	85
2. ELEMENT #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1 (6)	-0.139	-0.099	-0.129	-0.077	+0.033	-0.053	-0.077	-0.099	-0.129	-0.061	-0.064	+0.021	+0.021	-0.064	-0.062	-0.133
2 (6)	-0.099	-0.844	-0.379	-0.098	+0.207	+0.062	+0.023	+0.206	+0.061	-0.523	-0.524	+0.225	+0.084	+0.023	+0.154	-0.071
3 (6)	-0.129	-0.379	-0.280	-0.129	+0.061	-0.047	-0.053	+0.061	-0.047	-0.246	-0.246	+0.081	+0.020	+0.020	+0.221	-0.146
4 (5)	-0.077	-0.098	-0.129	-0.139	-0.099	-0.129	-0.077	+0.023	-0.053	-0.063	-0.061	-0.062	-0.062	+0.021	+0.021	-0.133
5 (6)	+0.023	+0.207	+0.061	-0.099	-0.844	-0.380	-0.099	+0.206	+0.061	+0.084	+0.226	-0.525	-0.525	+0.15	+0.083	-0.072
6 (5)	-0.053	+0.062	-0.047	-0.129	-0.380	-0.281	-0.129	+0.061	-0.047	+0.020	+0.082	-0.247	-0.247	+0.081	+0.020	-0.147
7 (6)	-0.077	+0.023	-0.053	-0.077	-0.099	-0.129	-0.077	-0.099	-0.129	+0.041	+0.021	-0.064	-0.064	-0.061	-0.064	-0.183
8 (5)	-0.099	+0.206	+0.061	+0.023	+0.206	+0.061	-0.099	-0.844	-0.378	+0.224	+0.083	+0.083	+0.15	-0.523	-0.522	-0.072
9 (6)	-0.129	+0.061	+0.047	-0.053	+0.061	-0.047	-0.129	-0.378	-0.280	+0.081	+0.020	+0.020	+0.08	-0.246	-0.246	-0.147
10 (5)	-0.061	-0.823	-0.246	-0.063	+0.084	+0.020	+0.021	+0.224	+0.081	-0.465	-0.284	+0.108	+0.017	+0.107	+0.230	-0.042
11 (5)	-0.064	-0.524	-0.246	-0.061	+0.226	+0.082	+0.021	+0.083	+0.020	-0.284	-0.465	+0.231	+0.108	+0.016	+0.107	-0.042
12 (7)	+0.021	+0.225	+0.081	-0.062	-0.525	-0.247	-0.064	+0.083	+0.020	+0.108	+0.231	-0.466	-0.285	+0.107	+0.017	-0.042
13 (5)	+0.021	+0.084	+0.020	-0.064	-0.525	-0.247	-0.062	+0.224	+0.081	+0.017	+0.108	-0.285	-0.467	+0.230	+0.107	-0.042
14 (5)	-0.064	+0.083	+0.020	+0.021	+0.225	+0.081	-0.061	-0.523	-0.246	+0.107	+0.016	+0.107	+0.230	-0.465	-0.284	-0.042
15 (7)	-0.062	+0.224	+0.081	+0.021	+0.083	+0.020	-0.064	-0.522	-0.246	+0.230	+0.107	+0.017	+0.107	-0.284	-0.464	-0.042
16 (5)	-0.133	-0.071	-0.146	-0.133	-0.072	-0.147	-0.133	-0.072	-0.147	-0.042	-0.042	-0.043	-0.043	-0.042	-0.042	-0.212

Table 2.10.2 FINITE ELEMENT RESULTS - BENDING & STRETCHING (SHELL) MODEL $(\Delta_3) \times 10^4 \text{ W/L3}$

MIT JCT. NO.	62	46	44	25	5	23	41	75	60	64	27	7	21	58	77	85
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1 (4)	-0.076	-0.047	-0.058	-0.026	+0.011	-0.015	-0.026	-0.047	-0.058	-0.033	-0.027	+0.010	+0.010	-0.027	-0.033	-0.054
2 (4)	-0.047	-0.042	-0.217	-0.047	+0.102	+0.034	+0.011	+0.102	+0.034	-0.318	-0.317	+0.120	+0.036	+0.037	+0.121	-0.031
3 (4)	-0.058	-0.217	-0.153	-0.058	+0.034	-0.011	-0.015	+0.034	-0.011	-0.138	-0.137	+0.046	+0.011	+0.011	+0.046	-0.060
4 (5)	-0.026	-0.047	-0.058	-0.026	-0.047	-0.058	-0.026	+0.011	-0.015	-0.027	-0.033	-0.033	-0.027	+0.010	+0.010	-0.034
5 (5)	+0.011	+0.102	+0.034	-0.047	-0.541	-0.217	-0.047	+0.101	+0.034	+0.036	+0.120	-0.317	-0.317	+0.110	+0.036	-0.031
6 (5)	-0.015	+0.034	-0.011	-0.058	-0.217	-0.164	-0.058	+0.034	-0.011	+0.011	+0.046	-0.138	-0.137	+0.045	+0.011	-0.060
7 (4)	-0.026	+0.011	-0.015	-0.026	-0.047	-0.058	-0.026	-0.048	-0.058	+0.010	+0.010	-0.027	-0.033	-0.033	-0.027	-0.054
8 (5)	-0.047	+0.102	+0.034	+0.011	+0.101	+0.034	-0.048	-0.540	-0.217	+0.120	+0.036	+0.036	+0.120	-0.317	-0.317	-0.031
9 (4)	-0.058	+0.034	-0.011	-0.015	+0.034	-0.011	-0.058	-0.217	-0.153	+0.046	+0.011	+0.011	+0.046	-0.138	-0.137	-0.060
10 (4)	-0.033	-0.318	-0.138	-0.027	+0.036	+0.011	+0.010	+0.120	+0.046	-0.319	-0.148	+0.051	+0.004	+0.051	+0.138	-0.017
11 (7)	-0.027	-0.217	-0.137	-0.033	+0.120	+0.046	+0.010	+0.036	+0.011	-0.148	-0.318	+0.137	+0.051	+0.004	+0.051	-0.017
12 (7)	+0.010	+0.120	+0.046	-0.033	-0.317	-0.138	-0.027	+0.036	+0.011	+0.051	+0.137	-0.319	-0.148	+0.051	+0.004	-0.017
13 (3)	+0.010	+0.036	+0.011	-0.027	-0.317	-0.137	-0.033	+0.120	+0.046	+0.004	+0.051	-0.148	-0.319	+0.137	+0.051	-0.017
14 (5)	-0.027	+0.037	+0.011	+0.010	+0.120	+0.046	-0.033	-0.317	-0.138	+0.051	+0.004	+0.051	+0.137	-0.319	-0.148	-0.017
15 (7)	-0.033	+0.121	+0.046	+0.010	+0.036	+0.011	-0.027	-0.317	-0.137	+0.138	+0.051	+0.004	+0.051	-0.148	-0.318	-0.017
16 (5)	-0.054	-0.031	-0.060	-0.054	-0.031	-0.060	-0.054	-0.031	-0.060	-0.017	-0.017	-0.017	-0.018	-0.018	-0.018	-0.017

Table 2.10.3 - FINITE ELEMENT RESULTS - MSFC MODEL ON ICES-STEADY II - BENDING ACTION ONLY $(\Delta z) \times 10^{-3}$ in/us

ACTUATOR #	86	7	65	116	231	42	213	185	162	11	28	169	266	249	109	137
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1 (16)	-0.139	-0.096	-0.127	-0.075	+0.033	-0.052	-0.075	-0.096	-0.127	-0.061	-0.062	+0.021	+0.021	-0.062	-0.061	-0.131
2 (1)	-0.066	-0.832	-0.374	-0.097	+0.205	+0.061	+0.023	+0.205	+0.061	-0.515	-0.515	+0.221	+0.083	+0.083	+0.221	-0.070
3 (65)	-0.121	-0.374	-0.218	-0.127	+0.061	-0.062	-0.052	+0.062	-0.045	-0.243	-0.243	+0.080	+0.070	+0.021	+0.060	-0.144
4 (116)	-0.075	-0.097	-0.127	-0.139	-0.096	-0.127	-0.075	+0.023	-0.052	-0.063	-0.061	-0.060	-0.062	+0.021	+0.021	-0.130
5 (231)	+0.023	+0.205	+0.061	-0.096	-0.832	-0.373	-0.096	+0.205	+0.061	+0.083	+0.221	-0.515	-0.514	+0.221	+0.083	-0.080
6 (42)	-0.052	+0.061	-0.066	-0.127	-0.373	-0.218	-0.127	+0.061	-0.046	+0.020	+0.080	-0.243	-0.243	+0.080	+0.070	-0.144
7 (185)	-0.075	+0.023	-0.052	-0.075	-0.096	-0.127	-0.139	-0.097	-0.127	+0.021	+0.021	-0.062	-0.060	-0.061	-0.063	-0.130
8 (162)	-0.066	+0.205	+0.062	+0.023	+0.205	+0.061	-0.097	-0.832	-0.374	+0.221	+0.083	+0.083	+0.221	-0.515	-0.515	-0.070
9 (266)	-0.127	+0.062	-0.045	-0.052	+0.061	-0.066	-0.127	-0.374	-0.218	+0.080	+0.021	+0.020	+0.080	-0.243	-0.243	-0.144
10 (11)	-0.061	-0.515	-0.243	-0.063	+0.083	+0.020	+0.021	+0.221	+0.020	-0.458	-0.219	+0.105	+0.018	+0.105	+0.215	-0.094
11 (28)	-0.062	-0.515	-0.243	-0.061	+0.221	+0.080	+0.021	+0.083	+0.021	-0.279	-0.458	+0.225	+0.105	+0.103	+0.105	-0.094
12 (169)	+0.021	+0.221	+0.080	-0.061	-0.514	-0.243	-0.062	+0.083	+0.020	+0.105	+0.215	-0.458	-0.279	+0.105	+0.103	-0.094
13 (266)	+0.021	+0.083	+0.020	-0.062	-0.514	-0.243	-0.060	+0.221	+0.080	+0.018	+0.105	-0.279	-0.458	+0.215	+0.105	-0.094
14 (249)	-0.062	+0.083	+0.021	+0.021	+0.221	+0.080	-0.061	-0.515	-0.243	+0.105	+0.018	+0.105	+0.225	-0.458	-0.279	-0.094
15 (109)	-0.061	+0.221	+0.080	+0.021	+0.083	+0.020	-0.063	-0.515	-0.243	+0.215	+0.105	+0.018	+0.105	-0.279	-0.458	-0.094
16 (137)	-0.131	-0.070	-0.144	-0.130	-0.070	-0.144	-0.130	-0.070	-0.144	-0.094	-0.094	-0.094	-0.341	-0.341	-0.341	-0.210

(ΔU) · 10⁴ in/lb
SIGN CHANGED

Table 2.10.4 NISA /MSFC FINITE ELEMENT RESULTS ON NASTRANS - SAME AS TABLE I (S/E-ASTR-MA-71-104)

ACTUATOR #	86	7	65	116	237	192	213	185	162	11	28	169	266	249	109	137
1	1 (44)	- .110														
2	2 (7)	- .102	- .863													
3	3 (46)	- .108	- .387	- .264												
4	4 (14)	- .055	- .101	- .108												
5	5 (11)	+ .024	+ .212	+ .063	- .101											
6	6 (42)	- .032	+ .063	- .075	- .107	- .264										
7	7 (13)	- .055	+ .025	- .032	- .055	- .107	- .120									
8	8 (45)	- .101	+ .211	+ .063	+ .074	+ .063	- .101	- .863								
9	9 (10)	- .103	+ .063	- .075	- .032	- .075	- .108	- .387	- .264							
10	10 (11)	- .064	- .534	- .253	- .066	+ .021	+ .022	+ .227	+ .082	- .477						
11	11 (26)	- .066	- .534	- .253	- .064	+ .082	+ .022	+ .086	+ .021	- .793	- .477					
12	12 (44)	+ .022	+ .228	+ .082	- .064	- .252	- .066	+ .087	+ .021	+ .108	+ .231	- .478				
13	13 (24)	+ .022	+ .087	+ .021	- .066	- .252	- .064	+ .228	+ .082	+ .019	+ .109	- .291	- .478			
14	14 (24)	- .066	+ .087	+ .021	+ .021	+ .082	- .064	- .534	- .252	+ .109	+ .019	+ .109	+ .231	- .478		
15	15 (13)	- .064	+ .228	+ .082	+ .021	+ .086	- .066	- .534	- .252	+ .231	+ .109	+ .019	+ .109	- .280	- .477	
16	16 (10)	- .026	- .073	- .112	- .098	- .112	- .098	- .073	- .112	- .043	- .043	- .043	- .043	- .043	- .243	- .160

- SYMMETRIC -

Table 2.10.4 is the NASTRAN results obtained by NASA/MSFC. This matrix should be identical to that in Table 2.10.2, but considerable differences are discernable, to a degree not covered by numerical errors. Some matrix components in Table 2.10.4 are easily twice those in Table 2.10.2, with the mean deviation in the order of about 70%. Table 2.10.4 appears to correlate much better with Tables 2.10.1 and 2.10.3 which represent the bending behavior alone, usually within 15 to 20%. It is possible that if the STRUDL shell study had been performed with the stretching component suppressed, but with a z component (as well as x and y) for the nodes, results in Tables 2.10.1, 2.10.3 and 2.10.4 might all be very close indeed.

A copy of the NASTRAN input used to generate Table 2.10.4 was obtained and examined, but no errors could be discovered that could have caused this. We are less familiar with the subtleties of NASTRAN, however, than with STRUDL. The possibility remains open, therefore, of a fundamental "bug" inside the NASTRAN system that prevents stretching action from occurring in this example.

A final confirming study should be performed, that of testing the NASTRAN shell model on STRUDL and the STRUDL models on NASTRAN, but because of funding limitations, this has not been done here.

2.10.4 Conclusions

Based on these results, it is obvious that rather large errors of presently unknown origin can creep into the finite-element modelling of shallow shells. These are especially difficult to detect if no closed-form solutions are available for reference. The configurational modelling on the other hand appeared to have relatively small effect, but the reliability of the results in absolute terms is in question.

If it is anticipated that the control algorithms are going to be sensitive to errors in the flexibility matrix, thorough, confirming studies to establish the reliability of the structural analysis results will be absolutely mandatory.

2.11 Segmented Mirror Model

A linearized model of the segmented mirror relating the figure error measurements x_f to the displacement actuator outputs m_m in the form

$$x_f = Am_m \quad (2.11.1)$$

was desired where A is an n by n_r measurement-position matrix. If the actuators associated with each segment are grouped in the form $m_m^{(1)}, m_m^{(2)}, \dots, m_m^{(l)}$ and the corresponding surface deflections are identified as $x_f^{(1)}, x_f^{(2)}, \dots, x_f^{(l)}$, it is apparent that (2.11.1) may be partitioned in the form:

$$\begin{bmatrix} x_f^{(1)} \\ x_f^{(2)} \\ \vdots \\ x_f^{(l)} \end{bmatrix} = \begin{bmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ & & \ddots \\ 0 & & A_{ll} \end{bmatrix} \begin{bmatrix} m_m^{(1)} \\ m_m^{(2)} \\ \vdots \\ m_m^{(l)} \end{bmatrix} \quad (2.11.2)$$

The linear model for the k th segment, for example, relates the k th mirror segment measurements provided by the figure sensor at $n^{(k)}$ measurement points on the surface of the k th segment to the displacement actuations $m_m^{(k)}$ at $n_r^{(k)}$ segment actuator locations. The value of $n_r^{(k)}$ is three, providing three degrees of freedom for each segment.*

The figure sensor detects the difference in the length of radii joining the figure sensor decollimator to the desired measurement point and a fixed reference location. Since the entire segment is capable of motion,

* Note that $n_r^{(k)}$ cannot be greater than three if segment deformation is to be avoided.

the reference radius cannot be considered constant, as in the case of the deformable mirror; and the effects of perturbations in the reference radius must be considered.

Consider the k th segment shown in Fig. 2.11.1. In order to simplify the analysis it is assumed that the points p_1 , p_2 and p_3 lie on a sphere centered on the figure sensor decollimator. This condition is satisfied by the initial tilt alignment control system.

Suppose that the radii joining the measurement point $x_{fi}^{(k)}$ and the reference point to the decollimator are identified by $R_i^{(k)}$ and $R_r^{(k)}$, respectively. Suppose that a perturbation Δm_{mj} is introduced in the j th displacement actuator. The perturbation in the i th figure error measurement is

$$\Delta x_{fi}^{(k)} = \Delta R_i^{(k)} - \Delta R_r^{(k)} \quad (2.11.3)$$

The elements of A_{kk} may then be obtained by passing to the limit $\Delta m_{mj}^{(k)} \rightarrow 0$.

$$a_{ij}^{(k)} = \lim_{\Delta m_{mj}^{(k)} \rightarrow 0} \frac{\Delta R_i^{(k)} - \Delta R_r^{(k)}}{\Delta m_{mj}^{(k)}} \quad (2.11.4)$$

The perturbations $\Delta R_i^{(k)}$ and $\Delta R_r^{(k)}$ may be computed by considering the segment rotation about axes joining the segment actuator locations. Translation in this philosophy are the result of the superposition of rotational effects. Since the analysis procedures involved in computing $\Delta R_i^{(k)}$ and $\Delta R_r^{(k)}$ are identical, it is sufficient to illustrate the procedure by generating $\Delta R_i^{(k)}$.

Consider the k th segment illustrated in Fig. 2.11.1 and suppose that $j = 1$. The perturbation $\Delta m_m^{(k)}$ will cause a rotation about points p_2 and p_3 . Consider a point $x_{fi}^{(k)}$ on the surface of the mirror which is d_a distant from p_2p_3 . If d_b is the distance from the actuator $m_{m1}^{(k)}$ to p_2p_3 , the change in radius $R_i^{(k)}$:

$$\Delta R_i^{(k)} = \frac{d_a}{d_b} \frac{1}{\cos \gamma_1 \cos \gamma_2} \Delta m_{m1}^{(k)} \quad (2.11.5)$$

if $\Delta m_{m1}^{(k)}$ is sufficiently small. The angles γ_1 and γ_2 are given by the expressions:

$$\gamma_1 = \sin^{-1} \frac{d_a}{2R} \quad (2.11.6)$$

$$\gamma_2 = \sin^{-1} \frac{d_c}{R} \quad (2.11.7)$$

where d_c is the distance between $x_{fi}^{(k)}$ and a plane perpendicular to $p_2 p_3$ containing the center of curvature of the mirror. Computation of $\Delta R_r^{(k)}$ leads to an expression similar to (2.11.5).

$$\Delta R_r^{(k)} = \frac{d_e}{d_f} \frac{1}{\cos \gamma_3 \cos \gamma_4} \quad (2.11.8)$$

where d_e , d_f , γ_3 and γ_4 correspond to d_a , d_b , γ_1 and γ_2 respectively. Expressions (2.11.5) and (2.11.8) may then be substituted in (2.11.4) to yield.

$$a_{i1}^{(k)} = \frac{d_a}{d_b} \frac{1}{\cos \gamma_1 \cos \gamma_2} - \frac{d_e}{d_f} \frac{1}{\cos \gamma_3 \cos \gamma_4} \quad (2.11.9)$$

the desired element of $A^{(k)}$. A similar procedure may be used to construct the elements relating measurement errors to actuator perturbations at locations p_2 and p_3 .

The computations outlined above are easily mechanized. Note that the effect of neglecting γ_1 to γ_4 is quite small for segments of aperture less than $f/4$. The effect of neglecting the curvature of the mirror in the d_a , d_e computation is also quite small. Thus it is probably possible to construct an adequate model of the segmented mirror from the projected $x - y$ coordinates obtained from the segmented mirror drawings.

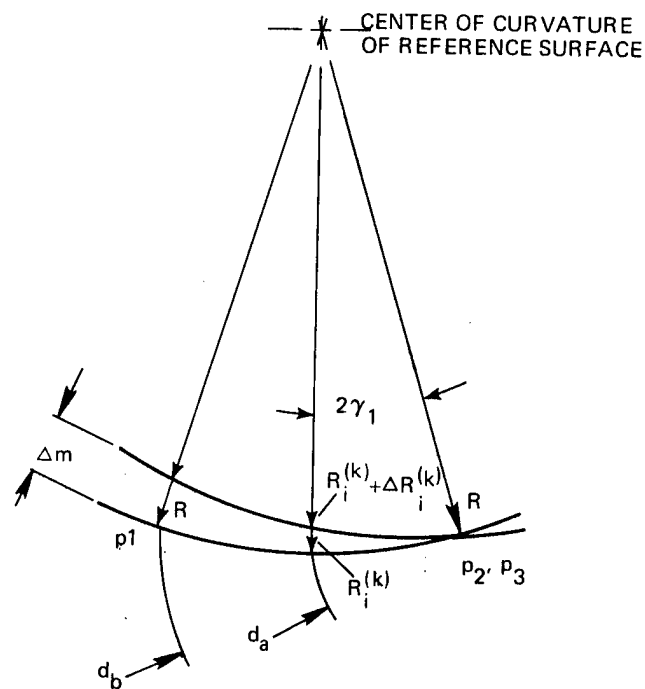
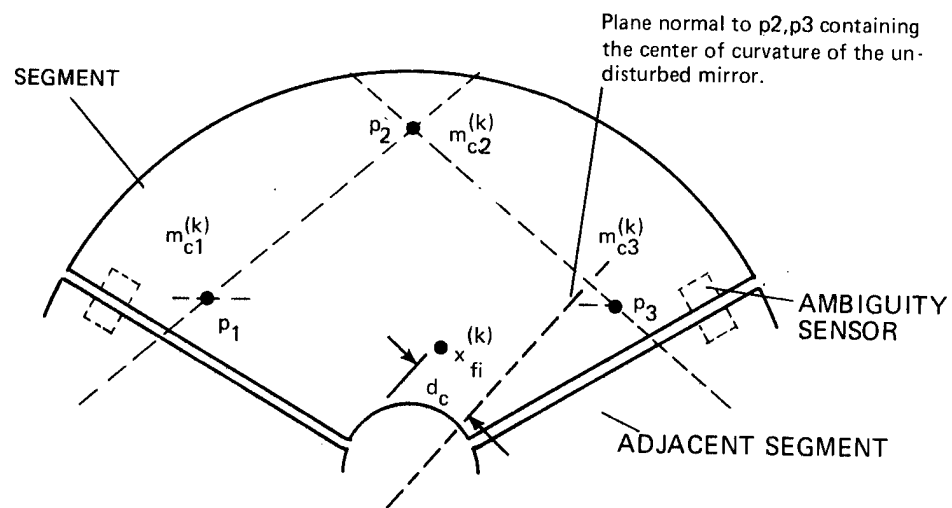


Fig. 2.11.1 Typical mirror segment geometry.

CHAPTER 3

EXPERIMENTAL ACTIVE MIRROR FIGURE CONTROL ALGORITHMS

3.1 Introduction

Figure control systems have been developed by previous investigators for two types of primary mirror structures -- segmented mirrors and deformable mirrors. Figure control in the former case is achieved by translating and rotating rigid individual mirror segments. Deformable mirror figure control is realized by elastically deforming the reflecting surface of the mirror to improve figure accuracy.

The surface accuracy achieved by figure control systems is determined by the number and arrangement of the actuators and the measurement points, the accuracy of the figure sensor, and the type of control algorithm.

Segmented mirror accuracy is ultimately limited by the figure accuracy of the individual segments which can be quite high as a result of the relatively small size of each segment. The development of large active segmented mirrors is hampered, however, by the problems associated with the accurate fabrication of off-axis surfaces of rotation.

Active deformable mirror figure accuracy is ultimately limited by the number and the geometric arrangement of the figure actuators.

The goal of current figure control systems is to achieve an rms figure accuracy of 30 nm ($\approx \lambda / 20$ at 632.8 nm) which would provide diffraction-limited performance throughout most of the visible spectrum.

Two classes of figure control algorithms have emerged. The first treats the mirror as a static body -- depending on inherent mechanical damping to eliminate vibrations induced by disturbances and actuator motion. The static representation has been used by previous investigators^{1, 6, 7, 13} to control deformable and segmented mirrors and is probably an adequate approach for the space telescope as a result of the low frequencies associated with the disturbances acting on the mirror.¹ Theoretical studies have also been performed to develop algorithms which provide active control of the dynamical bending modes of the mirror.¹⁴ The modal control approach places severe bandwidth requirements on the control system and complicates the problem of actuator placement. The following systems describe algorithms which have been developed at MIT/DL to control the mirror figure in the static sense.

3.2 Deformable Mirror Control Laws

Suppose that the error between that actual figure and the ideal figure is evaluated on the surface of the mirror at n discrete points. The errors may be conveniently expressed as elements of an array x_f .

The figure of the primary mirror is controlled by elastic deformation achieved by applying an array of n_r loads m_m to the rear of the mirror which is rigidly supported at three points. If

the initial figure errors ($m_m = 0$) at the n measurement points are associated with the array x_d , the net figure errors at the n points may be written:

$$x_f = x_d + A_r m_m \quad (3.2.1)$$

where A_r is a reduced deformation-force matrix.

In order to develop a control strategy it is useful to define a performance index. A useful index is the unbiased root mean square figure error:

$$J_m = \left[\frac{1}{n} x_f' x_f \right]^{1/2} \quad (3.2.2)$$

This performance index is minimized if the control force m_m is of the form:

$$m_m = - \left[A_r' A_r \right]^{-1} A_r' x_d \quad (3.2.3)$$

Such a control is the linear optimal control for the system (3.2.1) with the performance index (3.2.2). The resulting figure is a best least squares fit to the ideal reflecting surface. Note that the control in (3.2.3) requires the measurement of n errors ($n > n_r$) in order to compute the n_r figure controls.

A special case of (3.2.3) occurs if $n = n_r$ in which case the figure errors at all n_r locations may be reduced to zero. The

required control in this case is:

$$m_m = -A_{rr}^{-1} I_r x_d \quad (3.2.4)$$

where I_r is a reduced identity matrix which maps the n figure measurements into n_r measurements. This control strategy has been used by other investigators and is referred to here as the simplified linear control.

The above controls are special cases of the general linear control law of the form:

$$m_m = -K_g x_d \quad (3.2.5)$$

The linear optimal and simplified linear gain matrices are differentiated by subscripts:

$$K_o = [A_r' A_r]^{-1} A_r' \quad (3.2.6)$$

$$K_\ell = A_{rr}^{-1} I_r \quad (3.2.7)$$

3.3 Segmented Mirror Control Laws

Segmented mirror figure control is achieved by translating and rotating each segment by means of three position actuators. It is convenient to represent the position controls as elements of an array m_m in which case the figure error x_f may be written:

$$x_f = x_d + A_r m_m \quad (3.3.1)$$

where x_d is the initial figure error and A_r is a linear transformation relating the actuator position changes to a corresponding change in the monitored figure errors.

In light of the similarity between equations (3.2.1) and (3.3.1) it is apparent that identical control laws are applicable to the segmented and deformable mirrors. Thus the simplified linear K_l and linear optimal K_o control gain matrices for the segmented system are:

$$K_o = \begin{bmatrix} A_r' & A_r \end{bmatrix}^{-1} A_r' \quad (3.3.2)$$

$$K_l = A_{rr}^{-1} \quad (3.3.3)$$

where A_{rr} is the doubly reduced model matrix for the segmented mirror.

3.4 Discrete Control Algorithm for Mirror Figure Control

Sections 3.2 and 3.3 have described the mathematical properties of the figure control laws. In order to achieve this control it is necessary to develop a discrete algorithm for digital computer realization. A simplified block diagram of the digital figure control system is shown in Fig. 3.4.1. A complete set of figure error data is sampled every t_s seconds. The figure control algorithm operates on the figure error to produce a set of desired actuator outputs m_c . The actuator commands m_c provide inputs to a digital figure actuator control system with a cycle time Δt . The figure control computation cycle time t_s is an integral multiple of Δt . The actuator control system operates to assure that outputs m_m are approximately equal to the actuator commands m_c . This

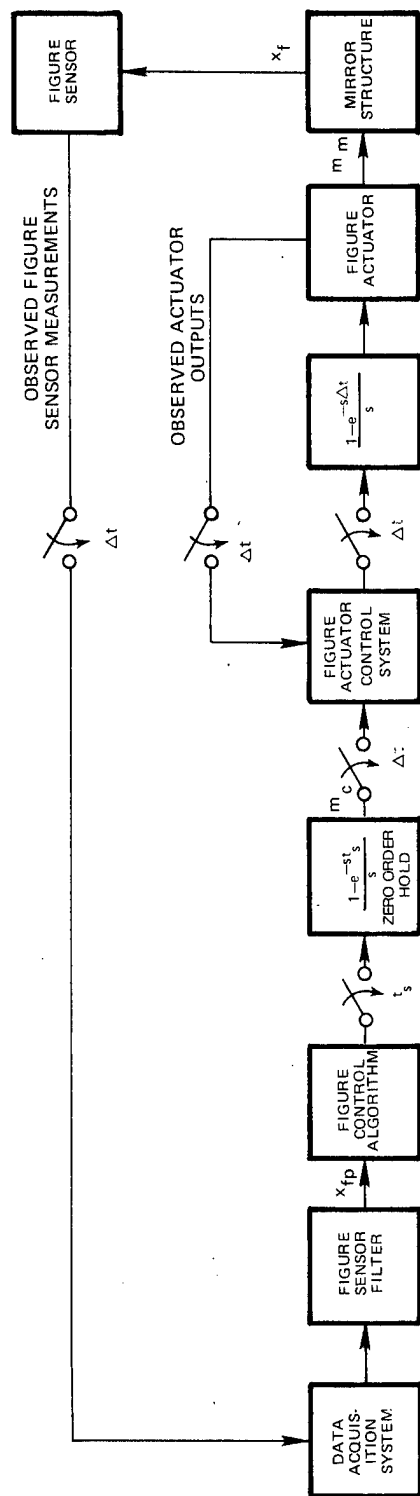


Fig. 3.4.1 Simplified diagram of the digital active mirror control system.

implementation of the EAM figure control is classified as a multirate sampled data system.

The time interval t_s between successive computations of m_c is determined by a number of considerations including the response time of the figure sensor and the dynamic response characteristics of the mirror structure and figure actuators.

The figure sensor currently incorporated in the EAM is a serial measuring device inasmuch as the figure error can only be measured one point at a time.* Limitations in the design of the piezoelectric interferometer path difference modulator and the phase detector filter time constant limit the minimum observation time at each measurement point, as indicated in Table 2.2.1, to approximately 0.2 seconds.

Operation of the figure actuators excites a damped vibration in the mirror structure. As a result of the serial nature of the figure sensor it is necessary to wait until the vibration has decayed below an acceptable level before measuring the figure error.

The dynamic response time of the actuators is determined by the bandwidth of the actuator control systems. The digital character of the actuator control systems will mean that the choice of Δt will play a strong role in determining minimum response time.

The EAM digital control system accounts for the dynamical characteristics of the EAM components by realizing the operation

* Other versions of the figure sensor utilize a photodiode array to permit parallel processing of figure error data.

sequence shown in Fig. 3.4.2. A control cycle is initiated by a period n_{wait} cycles long during which the actuators respond to the actuator commands. The number of cycles n_{wait} should be large enough to permit the actuators to reach an essentially steady state. At the end of the actuator control interval the actuator outputs are frozen, and the image dissector is positioned to the first measurement location. The measurement sequence consists of positioning the image dissector, a pause of n_{pos} cycles during which transients in the phase detector output are allowed to settle followed by a sequence of n_{meas} phase detector filter output measurements at intervals of n_{mint} cycles. The position, n_{pos} cycle wait, n_{mint} wait, measure, n_{mint} wait ... sequence is repeated at each of the n measurement points. At the completion of measurements the figure sensor data are processed to reduce noise and eliminate ambiguities and a new set of actuator commands m_c computed. The actuators are then activated to initiate another control cycle.

The discrete figure control algorithm is currently implemented in the form:

$$m_c(i+1) = m_c(i) + \beta_k K x_{fp}(i) \quad (3.4.1)$$

where $m_c(k)$ and $x_{fp}(k)$ are the values of actuator commands m_c and the processed figure error measurements x_{fp} at time t_k where:

$$t_{k+1} = t_k + t_s \quad (3.4.2)$$

A description of the figure sensor filter is given in Section 3.5.

The scalar factor β_k is given by the relation:

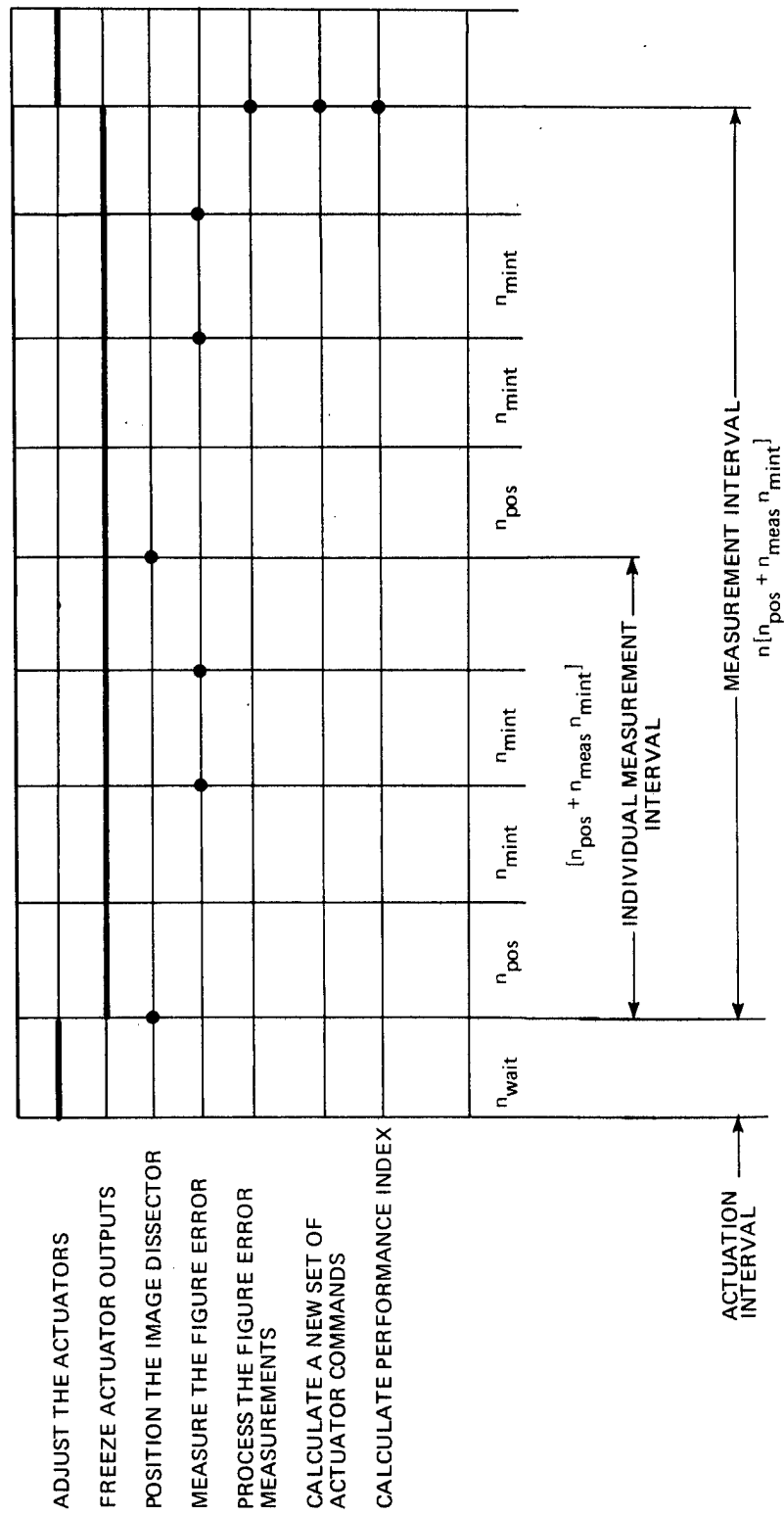


Fig. 3.4.2 Figure control system timing diagram.

$$\beta_k = \beta_g t_s \quad (3.4.3)$$

where β_g is a scalar constant. This choice for β_g assures that the dynamic response characteristics of the control system will be relatively independent of changes in t_s .

The feedback gain matrix K may be K_o , K_ℓ or a general gain matrix K_g . If $\beta_k = -1$ and $m_c(0) = 0$, the computed value m_c will equal the desired control:

$$m_c(1) = -K_g x_d \quad (3.4.4)$$

after one iteration. Values of $-1 < \beta_k < 0$ result in a solution which slowly converges to the desired control.¹

3.5 Figure Sensor Data Processing Algorithm

Ambiguities in figure measurement which occur whenever the magnitude of the figure error equals $\frac{\lambda}{4}[1 + 2i]$ where i is an integer limit the effective measurement range of the figure sensor to $\pm \frac{\lambda}{4}$. Since the initial figure magnitude could easily exceed $\frac{\lambda}{4}$, a digital signal processor was developed which would extend the effective operating range. The resulting algorithm reduces sensor noise in addition to eliminating measurement ambiguity.

The inspiration behind the figure sensor filter design may be obtained by observing certain statistics associated with the figure sensor filter outputs. Suppose that a sequence of points is defined on the surface of the mirror in such a way that the error at each

* If aspheric figure control is desired, the difference between the reference sphere and the desired figure can be many wavelengths. Measurement of an aspheric surface may be accomplished using the technique presented here or an optical data processing procedure such as the moire' fringe technique.⁸

point satisfies the relationship:

$$x_{fi} = i\Delta_f \quad (3.5.1)$$

where Δ_f is a positive number $\ll \frac{\lambda}{4}$. That is, the error at the points increases in a linear fashion. Such a situation may be achieved, in practice, by defining a set of measurement points equidistantly spaced or on a straight line and then slightly tilting the mirror about an axis perpendicular to the optical axis and the straight line. If the figure control system is required to perform a sequence of measurements on the points (3.5.1), the outputs of the figure sensor as a function of time will appear as shown in Fig. 3.5.1 which delineates the figure error input β_{xf} , figure sensor noise β_{nf} , phase detector input β_{xa} , output β_p filtered output β_f and rms value β_{mrf} calculated at each measurement location where:

$$\beta_{mf} = \frac{1}{n_{meas}} \sum_{i=1}^{n_{meas}} \beta_{fi} \quad (3.5.2)$$

$$\beta_{mrf} = \left[\frac{1}{n_{meas}} \sum_{i=1}^{n_{meas}} (\beta_{fi}^2 - \beta_{mf}^2) \right]^{1/2} \quad (3.5.3)$$

and β_{fi} is the i th filter output sample at each position. Note that β_{mrf} increases whenever a switching boundary ($\beta_{xf} = \frac{\lambda}{4}(1 + 2k)$; k an integer) is approached. This property is used advantageously to detect an ambiguous measurement range by defining a decision threshold β_{ft} on β_{mrf} .

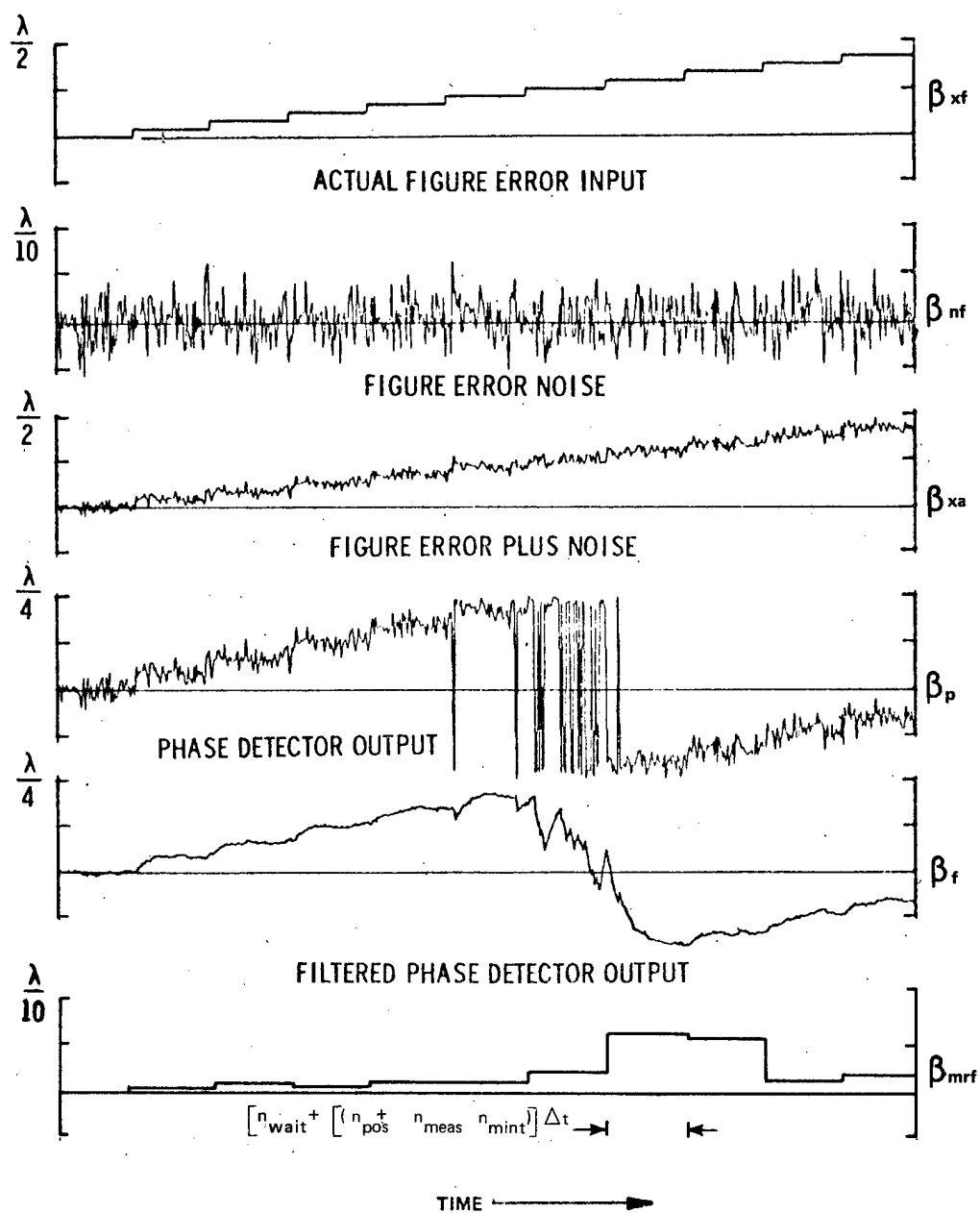


Fig. 3.5.1 Typical figure sensor simulation results.

A flow diagram of the figure sensor data processing algorithm is shown in Fig. 3.5.2. This algorithm is valid for measurement errors in the range $|\beta_{xa}| < \frac{3\lambda}{4}$ where λ is the laser wavelength (632.4 nm). An extension in range is easily accomplished by modifying the switching boundary and ambiguity factor computations. The limited range was adopted to simplify coding and to save computation time during simulation.

The initial measurement point should be selected so that $\beta_{mrf} < \beta_{ft}$. Subsequent computations provide values of the processed figure error relative to the first location.

If $\beta_{mrf} < \beta_{ft}$ at a measurement point, the figure error is calculated by adding the mean value β_{mf} of the n_{meas} measurements to the ambiguity factor β_{ab} which is initially zero. The computed value x_{fpi} is stored in β_{lf} for future reference. The nearest switching boundary is also calculated for use when an ambiguous measurement is perceived.

If the rms value of the measurements exceeds β_{ft} at the i th location, an ambiguous measurement problem is identified and the processed figure error is calculated by linear extrapolation about the closest switching boundary β_{sw} . The extrapolation constant β_z is read in by the program. The ambiguity factor is calculated if the value of β_{mf} has changed sign and the magnitude of the product of the old value of β_{mf} and its current value is greater than β_{ts} . This test prevents the generation of spurious ambiguity factor values which would arise if $\beta_{mf} \approx 0$.

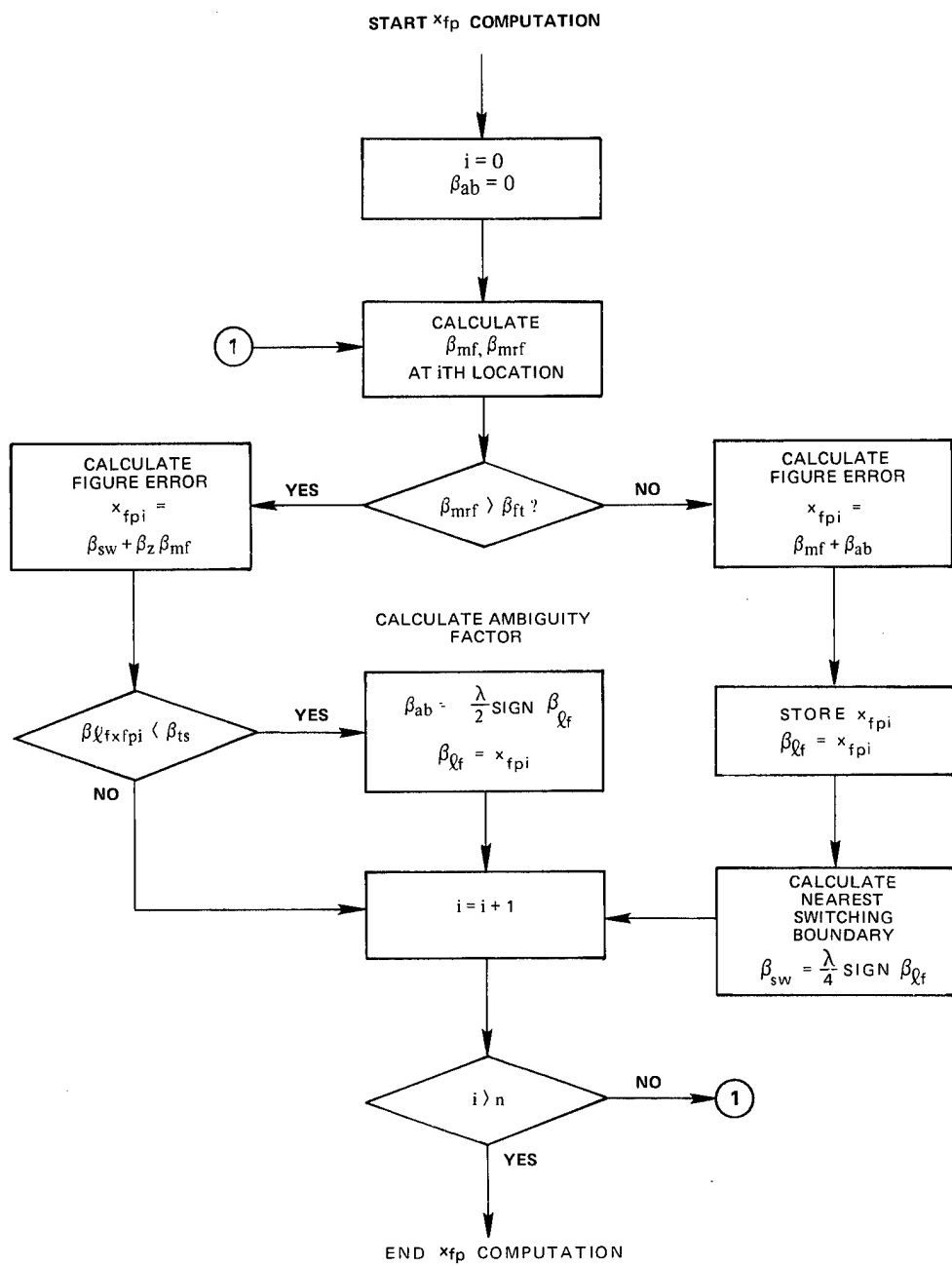


Fig. 3.5.2. Figure sensor data processing algorithm.

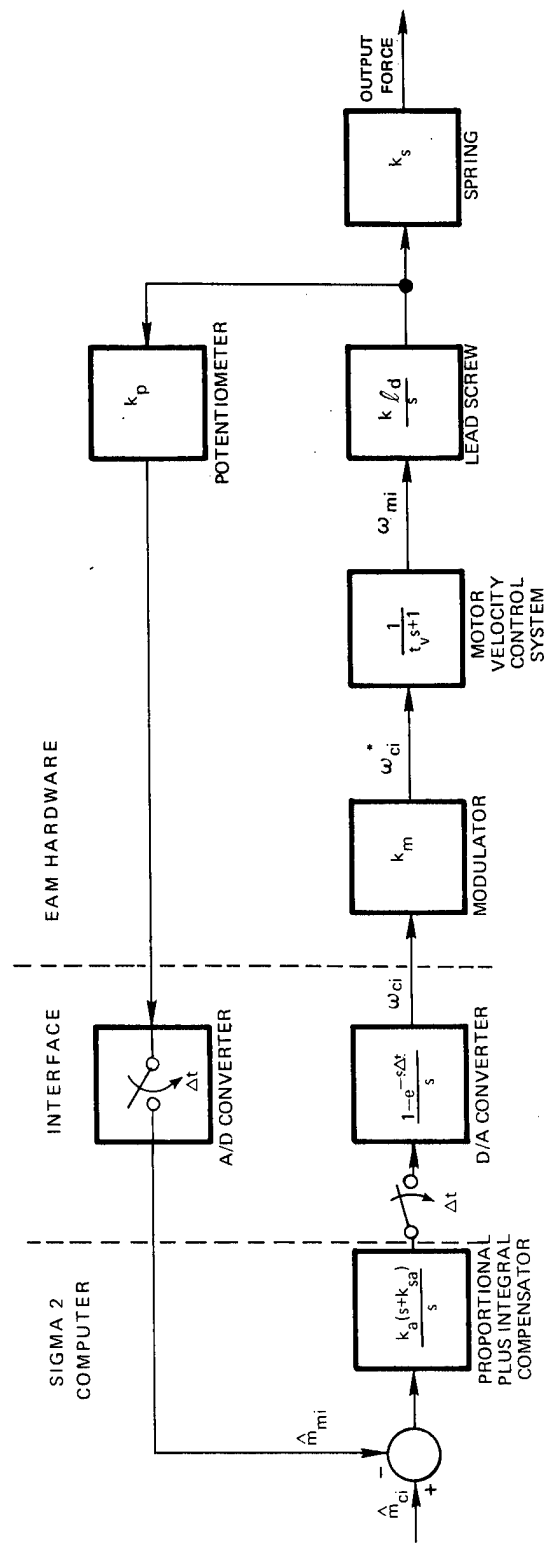


Fig. 3.6.1 Force actuator control algorithm.

Sensor noise is reduced by the averaging process performed during the computation of β_{mf} . If the sample period is long compared to the figure sensor time constant and the noise source is uncolored, the rms noise level is reduced by the reciprocal of the square root of n_{meas} .

3.6 Force Actuator Control System

A block diagram of the i th force actuator control algorithm is shown in Fig. 3.6.1. The actuator output force m_m is regulated by controlling the extension of a spring. The sensed i th spring extension $\hat{m}_{mi}$ is compared with the desired spring extension $\hat{m}_c$ and the resulting error signal processed to provide the motor velocity command ω_c . Proportional plus integral compensation is used to eliminate steady state position errors. The position command $\hat{m}_{ci}$ is obtained by operating on m_{ci} which is calculated by the discrete control law (3.4.1)

$$\hat{m}_{ci} = m_{ci} m_{gi} \quad (3.6.1)$$

where m_g is an input vector of scale factors.

3.7 Position Actuator Control System

The position actuator control system is shown diagrammatically in Fig. 3.7.1. The actuator position command is scaled using equation (3.6.1) to produce $\hat{m}_c$. The pulse modulator produces a pulse every time a new control is calculated. The area of the pulse is selected so that the resulting change in actuator position equals the desired change in position δm_{ci} :

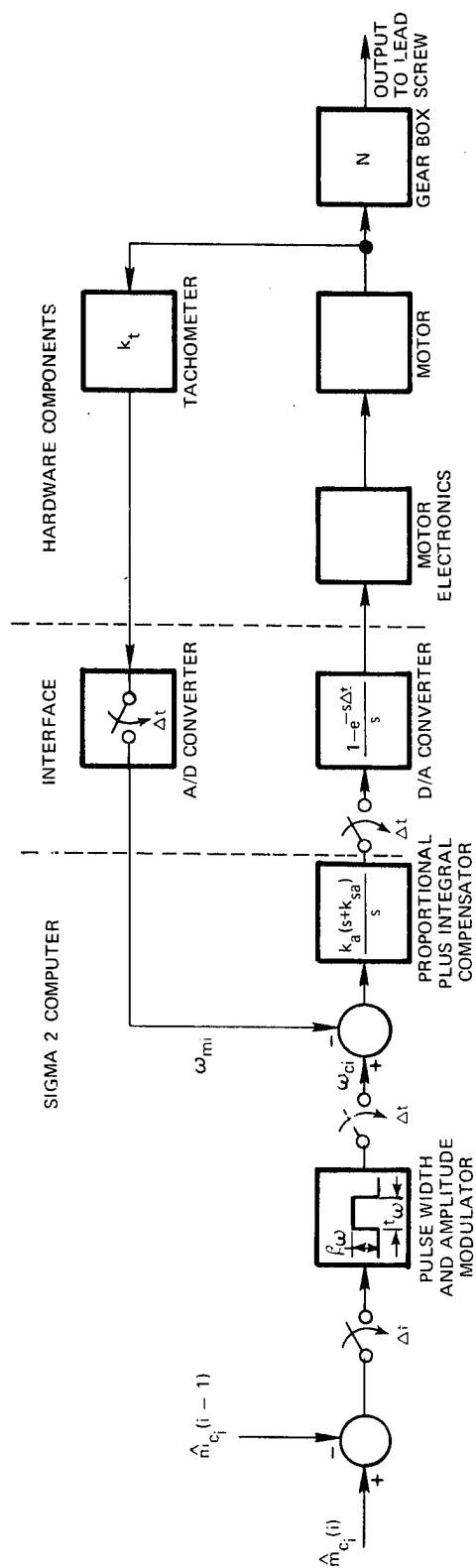


Fig. 3.7.1 Position actuator control system.

$$\delta m_{ci}^{(i)} = m_{ci}^{(i)} \quad (3.7.1)$$

thus β_ω and t_ω must be selected so that

$$\frac{NK_{ld} K_{s_1} K_{s_2}}{K_t} \beta_\omega t_\omega = \delta m_{ci} m_{gi} \quad (3.7.2)$$

Further constraints are imposed by requiring t_ω to be an integral multiple of Δt :

$$t_\omega = n_\omega \Delta t \quad (3.7.3)$$

With this restriction β_ω becomes:

$$\beta_\omega = \frac{\delta m_{ci} m_{gi} K_t}{NK_{ld} K_{s_1} K_{s_2} n_\omega \Delta t} \quad (3.7.4)$$

If m_{gi} is selected so that:

$$m_{gi} = \frac{NK_{ld} K_{s_1} K_{s_2}}{K_t} \quad (3.7.5)$$

the computation for β_ω becomes:

$$\beta_\omega = \frac{\delta m_{ci}}{n_\omega \Delta t} \quad (3.7.6)$$

n_{ω} is read in as part of the input data and must satisfy the relationship:

$$n_{\omega} \leq n_{\text{wait}} \quad (3.7.7)$$

Small values of n_{ω} lead to large commanded rates.

Note that this implementation does not include the motor velocity feedback loop signal processing filter included in the original position actuator control system.¹³ This omission was made because the digital control system does not impose severe response requirements on the velocity servomechanism.

3.8 Initial Active Mirror Alignment

The final alignment procedures described above are only applicable after the mirror figure has been approximately aligned relative to the figure sensor. Initial alignment is performed in two stages. The first stage consists of tilting the mirror or mirror segment until three designated non-collinear points on its surface lie on the surface of a sphere centered on the image sensor decollimator. The second stage of adjustment moves the mirror in an axial fashion until the distance to the decollimator focus equals the radius of curvature of the mirror. The latter adjustment is less sensitive than the former. Tilt and axial control adjustments can be repeated a number of times until satisfactory alignment is achieved.

The initial alignment control systems for the deformable mirror are included in the deformable mirror electronics and are

not discussed here. The alignment algorithms for the segmented mirror were implemented as part of the software package and are discussed in the following sections.

3.9 Tilt Alignment System

Each segment of the segmented active mirror is equipped with three actuators which permit segment motion in three degrees of freedom. The actuator deflections associated with the j th segment are conveniently identified by the elements m_{mj1} , m_{mj2} and m_{mj3} of the actuator output vector m_m . The corresponding x , y position coordinates on the figure surface are identified x_{j1} , y_{j1} , x_{j2} , y_{j2} , and x_{j3} , y_{j3} . Suppose that it is desired to tilt the mirror so that the figure errors at the actuator locations are all zeros. Tilt alignment is achieved by the following sequence of operations:

1. Drive the figure error at x_{j1} , y_{j1} to zero using actuator m_{mj1} .
2. Measure the figure error at position $(1 - i\Delta)x_{j1} + i\Delta x_{j2}$, $(1 - i\Delta)y_{j1} + i\Delta y_{j2}$ where $\Delta = n_{\text{tilt}}^{-1}$, and adjust actuator m_{mj2} to drive the error to zero.
3. Repeat step 2 for increasing values of $i = 1, 2, 3 \dots n_{\text{tilt}}$.
4. Simultaneously perform steps 2 and 3 for position x_{j3} , y_{j3} using actuator m_{mj3} .

At the end of step 4 the errors at all three actuator positions will be zero.

The tilt control algorithm was realized by modifying the program data so that the existing control structure could be utilized for both tilt

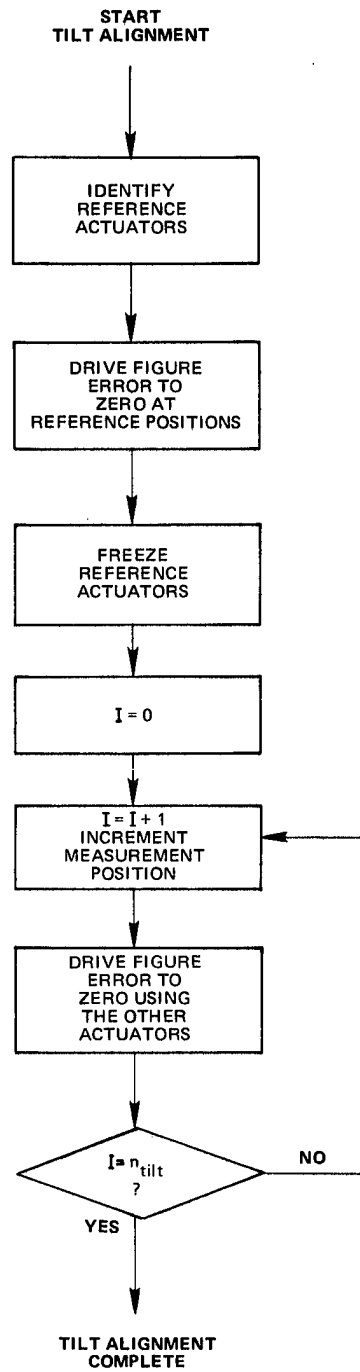


Fig. 3.9.1 Tilt alignment algorithm.

control and fine mirror figure adjustments. The actual tilt alignment program, illustrated in Fig. 3.9.1, performs steps 1-4 for all three segments simultaneously.

3.10 Slew Alignment Algorithm

Once the segments have been aligned in tilt, the three measurement points on each segment corresponding to the actuator locations will lie on a spherical surface centered on the center of curvature of the spherical wavefronts emerging from the decollimator. However, there is no assurance that all the measurement points will lie on a sphere of uniform curvature as a result of possible axial misalignment. In order to correct this problem the segmented mirror is equipped with an adjacent edge ambiguity sensor* which provides a measure of the relative axial segment position. The output of the ambiguity sensor is a maximum when the adjacent segment edges are at the same radius from the figure sensor. A disparity in edge alignment results in a reduction in sensor output monotonically related to the magnitude of the error in a useful range of ± 600 nm.

In order to correct errors in axial segment position a simple algorithm was developed. The control algorithm produces a sequence of axial position commands which converge to the position which maximizes the ambiguity sensor output.

Suppose that the output of the sensor between the reference segment j and segment k is identified as α_{jk} and the corresponding axial position command to the segment k actuators is α_{mk} . A simple algorithm which produces a sequence of α_{mk} which maximize

* white light interferometer

α_{jk} is shown in Fig. 3.10.1. This algorithm is based on the method of steepest descent. The variable α_d is a dummy variable. The parameters NHC and NIC keep track of the number of step size halvings and successful iterations respectively. An iteration is successful if the ambiguity sensor output is increased. The step size is halved if a zero or negative change in α_{sa} occurs. Computation is terminated if NHC equals NHM or NIC equals NIM where NHM and NIM are input variables.

The axial alignment control system is realized so that the axial positions of two segments are simultaneously adjusted with respect to the third segment which serves as a position reference.

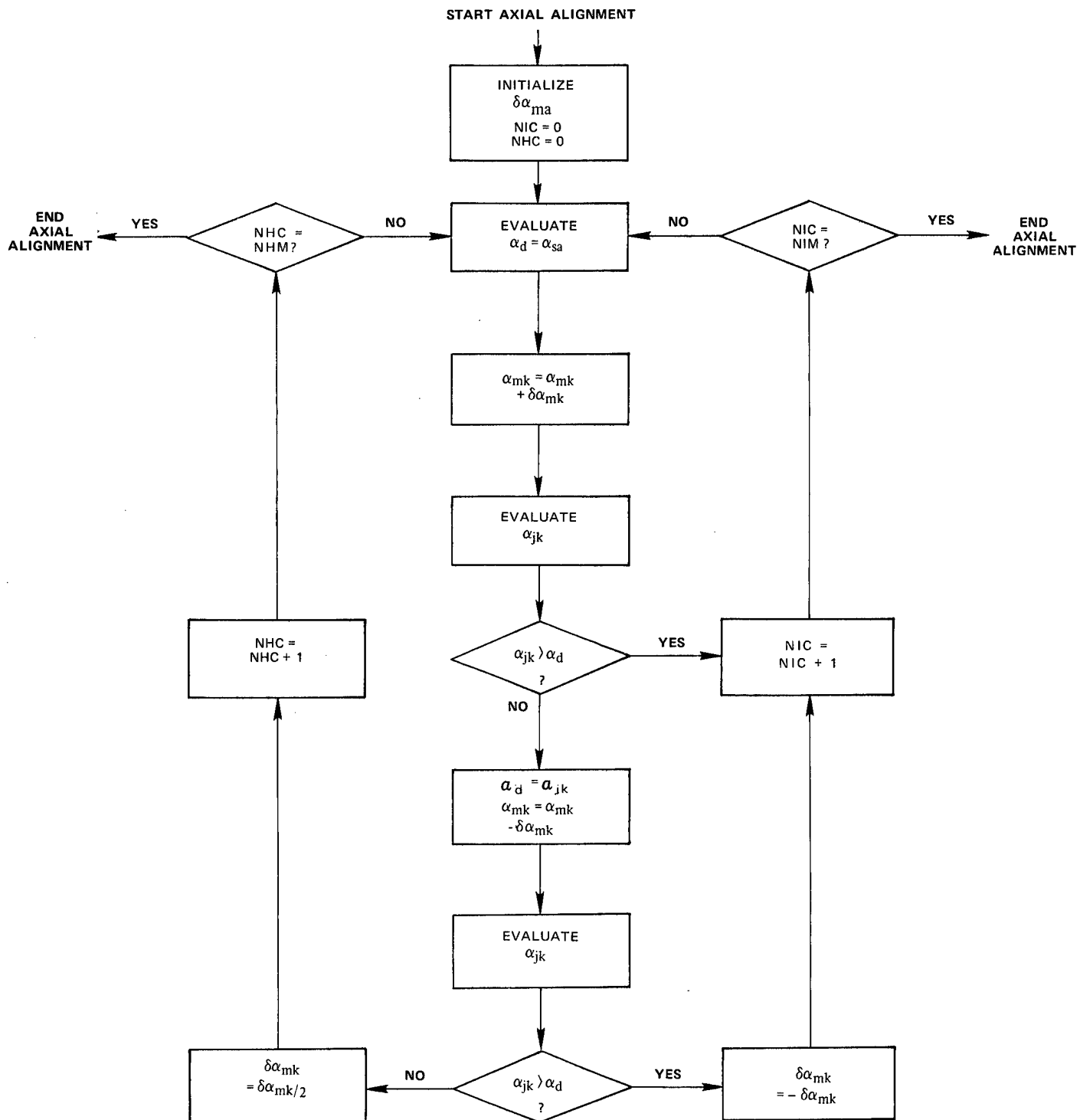


Fig. 3.10.1 Slew control system algorithm.

CHAPTER 4

EXPERIMENTAL ACTIVE MIRROR SOFTWARE

4.1 Introduction

The software of the EAM was written in FORTRAN, permitting execution on a wide variety of computers. Development and initial checkout were performed on an IBM 370/155, and final checkout was completed on an XDS Sigma 5/7 at MSFC. Software to be used in the XDS Sigma 2 was written in a simplified FORTRAN (370/155, Sigma 5/7 compatible) to accommodate the limitations of the Sigma 2 FORTRAN monitor.

The software consists of two sections; the first is designed for residence in the Sigma 5 where the complicated EAM control computations are performed on a time-shared basis; the second resides in the Sigma 2 and provides real-time control of the EAM hardware.

The architecture of the most important software elements is illustrated in Fig. 4.1.1. The main programs resident in the Sigma 5 and the Sigma 2 are designated SUPE5 and SUPE2 respectively.

The major hardware component control functions are performed by routine EAMCS via the figure sensor FIGSEN, actuator ACTCMD, and remote terminal TYPCON supervisory subroutines. While figure error measurement and actuator command execution are performed in the Sigma 2, figure data processing and figure control computation are performed in MAINA which is interrogated by EAMCS. The capabilities of MAINA are extended by subroutine MAINC.

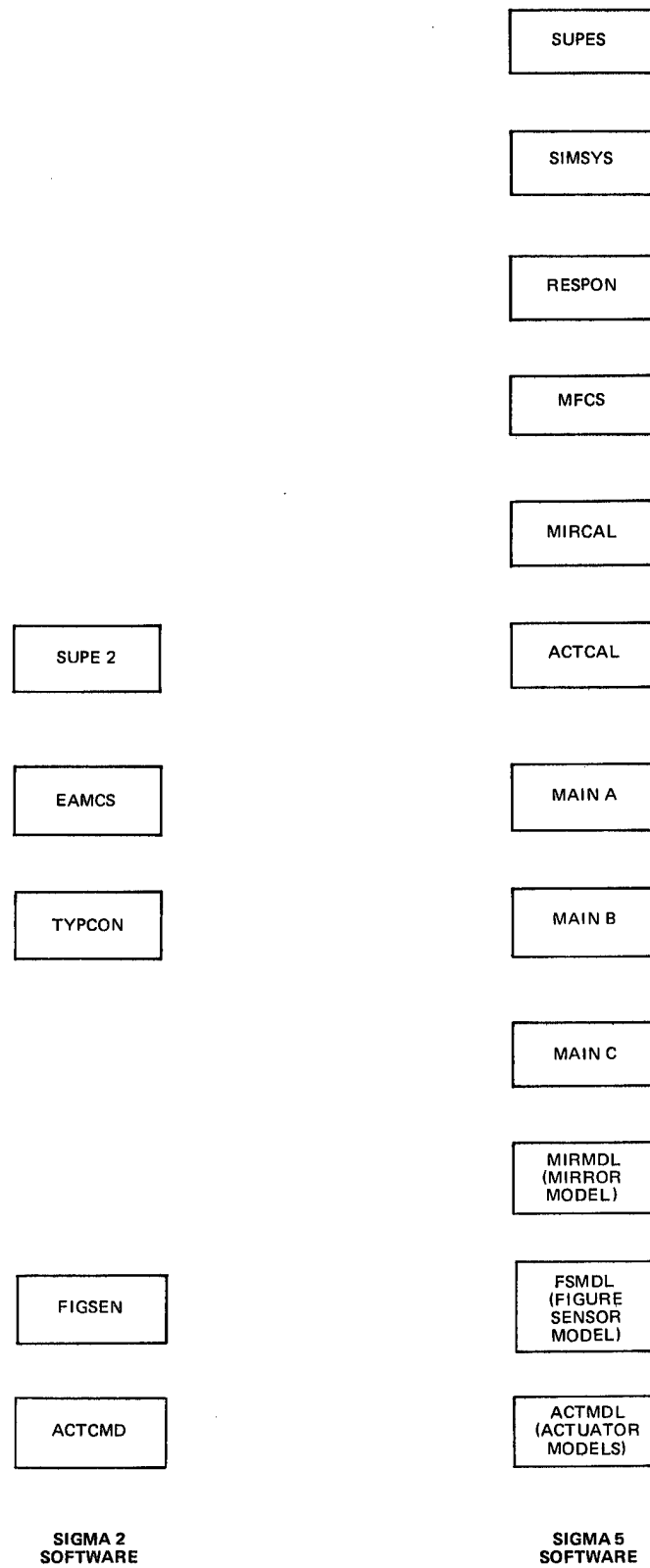


Fig. 4.1.1 Major EAM software components.

Data input control is primarily provided by routines SIMSYS and RESPON which, in addition, provide the basic software structure for active mirror simulation. Most of the data for the EAM control system and hardware models is read in by MFCS which also provides coding for the computation of the gain matrices K_o and K_ℓ .

Models for the mirror structure, figure sensor and actuator are provided by MIRMDL, FSMDL, and ACTMDL respectively.

Subroutines MIRCAL and ACTCAL provide software for experimentally evaluating the reduced structural model matrix of the mirror and checking the actuators for correct operation via actuator command perturbation and EAMCS.

Diagnostic and parameter modification functions using the remote terminal are provided by the combination of TYPCON in the 2 which operates the terminal and MAINB in the 5 which provides the coding required to perform the desired operations.

The software is designed to provide three major operating configurations determined by input operating mode control parameters (MODV, see section 5.2.1).

1. Experimental active mirror simulation using the real-time control system software and hardware models.
All program components are resident in the Sigma 5/7.
2. Experimental active mirror simulation using the real-time control system software resident in the Sigma 2.
3. Experiment operation using the Sigma 5 for complex control computations on a when-available-basis and a real-time hardware control system resident in the Sigma 2.

Operating configurations 1 and 2 are illustrated in Fig. 4.1.2 which shows the calling priorities^{*} in the simulation mode. In configuration 1 all the software modules are resident in the Sigma 5 as indicated by boundary "A." It is also possible to simulate the active mirror using both the Sigma 5 and Sigma 2 computers as indicated by boundary "B." In this case all transfers across boundary "B" are completed using SUPE5 and SUPE2 as explained in sections 5.2 - 5.4.

The calling priorities in the experiment operating mode are delineated in Fig. 4.1.3. Experiment control is transferred to the remote console via EAMCS and TYPCON. Once the experiment is started it will continue to operate for NTIMSQ cycles unless it is interrupted by a command from the remote terminal. Boundary "D" transfers are completed through SUPE5 and SUPE2 while communication across boundary "C" is accomplished by the A/D, D/A and D/D channels associated with the Sigma 2 interface.

Note that SUPE2 in configuration 1 is a subroutine whereas SUPE2 is a main program in operating modes 2 and 3. The software is set up so that versions of SUPE2, EAMCS, TYPCON, FIGSEN and ACTCMD can reside in the Sigma 5 and 2 simultaneously. This permits all three operating configurations to be tested without the necessity of reloading object programs.

4.2 Supervisory Software

Experimental active mirror software control is complicated by the dichotomization of the computer system into two essentially autonomous parts consisting of a large scale general purpose computer,

^{*} i.e., subroutine RESPON calls subrouting EAMCS.

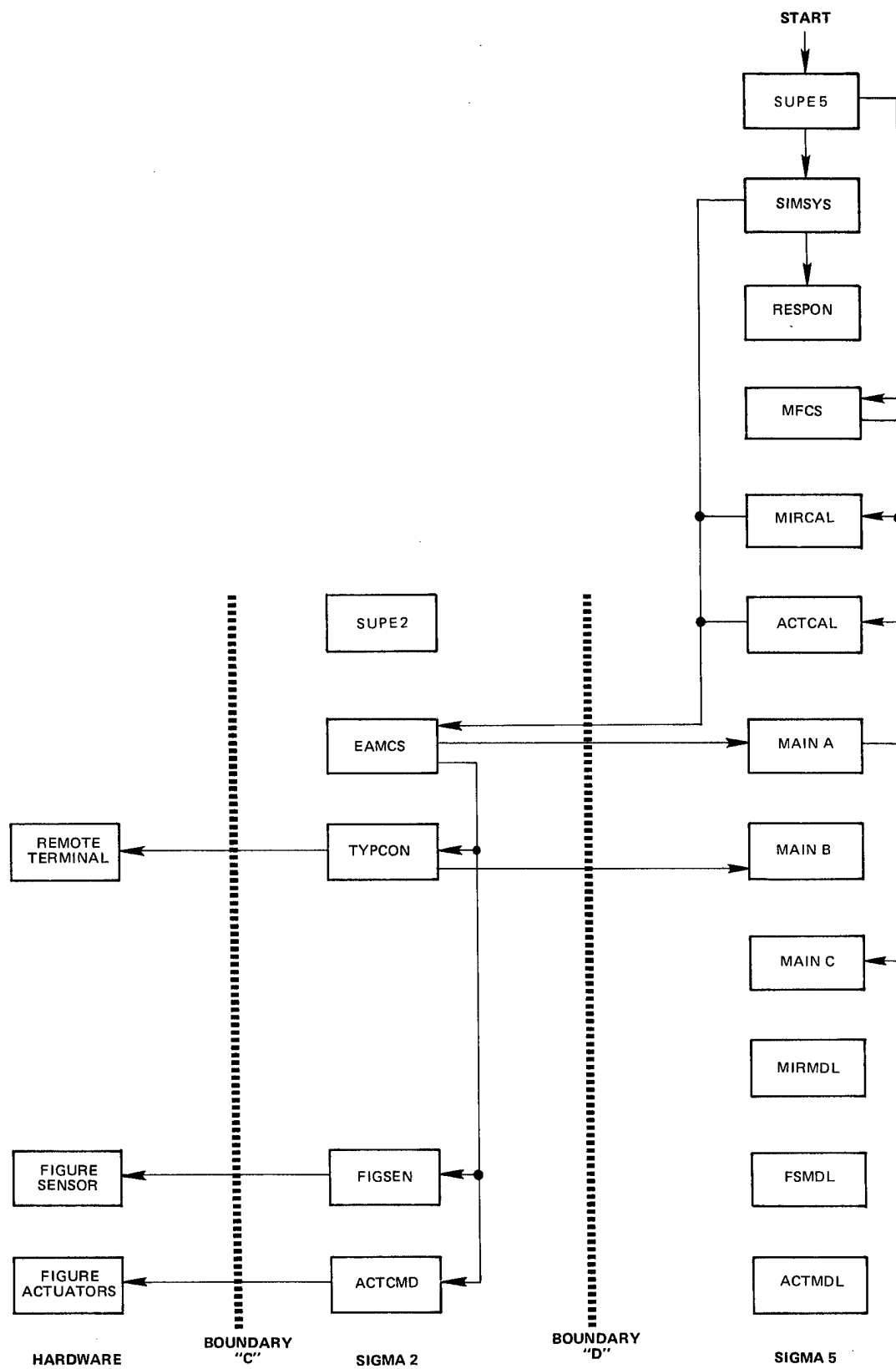


Fig. 4.1.2 Major subrouting calling priorities in the EAM simulation mode.

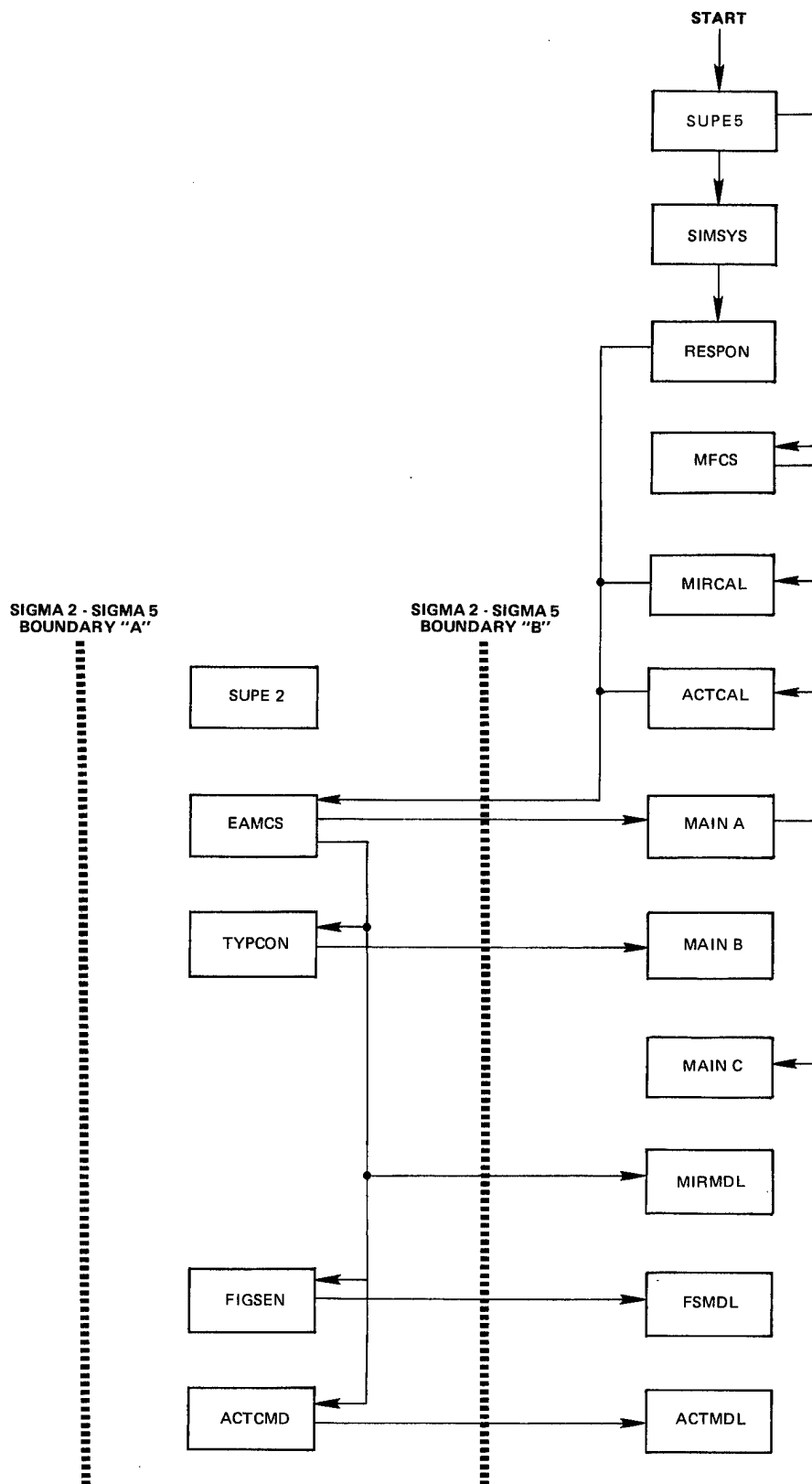


Fig. 4.1.3 Major subroutine calling priorities in the EAM experiment operating mode.

the XDS Sigma 5/7, and a small limited capacity machine, the XDS Sigma 2. Real-time control functions are restricted to the Sigma 2 which plays an essentially dedicated role in the experiment. When computation complexity exceeds the abilities of the Sigma 2, a data processing request is placed for the Sigma 5. Because of the limitations of the current Sigma 5-2 operating system, it was decided to provide the control structure in the form of two programs SUPE5 and SUPE2 which serve as main programs in the Sigma 5 and Sigma 2 respectively.

Transfer of computation responsibility between the computers presents a number of unusual problems. It was decided to treat each computer's software as an essentially independent program. Data communication between the computers is restricted to the transfer of a common data block. Transfers from the Sigma 2 to the Sigma 5 are accomplished by the following sequence of operations.

1. Catalog the destination and return subroutine identification numbers and NENTRY values.
2. Return Sigma 5 program control to SUPE5.
3. Transfer the Sigma 2 subroutine identification number and NENTRY value to the Sigma 2 as part of the common storage block via NFLGA and NFLGB.
4. Transmit an enable signal to the Sigma 2 to initiate execution of the Sigma 2 software. A computed "GO TO" statement in SUPE2 assures transfer to the 2 subroutine with the appropriate value of NENTRY.
5. Terminate computation in the Sigma 5.

The Sigma 2 computer will continue to perform the programmed operations in the selected section of the Sigma 2 software until they are complete. On completion return to the Sigma 5 is achieved by the following sequence.

6. The exit point from the Sigma 2 routine is identified by assigning a value to NFLGA.
7. Sigma 2 program control is returned to SUPE2.
8. SUPE2 enables an interrupt which initiates execution of SUPE5 and terminates Sigma 2 computation.
9. SUPE5 transfers the contents of the common data block from the Sigma 2 to the Sigma 5 memory.
10. SUPE5 uses its addressing structure to determine the return address in the Sigma 5 software.
11. Control is transferred to the identified Sigma 5 subroutine with the appropriate value of NENTRY.

A transfer of computational authority from the Sigma 2 to the Sigma 5 uses the following procedure.

1. The Sigma 2 software exit point is identified by a value of NFLGA.
2. Return Sigma 2 program control to SUPE2.
3. Enable an interrupt requesting execution of SUPE5 in the Sigma 5.
4. Terminate computation in the Sigma 2.

The Sigma 5 monitor will begin execution of SUPE5 as soon as it comes to the head of the job queue.

5. Begin execution of SUPE5.
6. SUPE5 transfers the contents of the common data base from the 2 to the 5.

7. Use the value of NFLGA to establish a new Sigma 5 destination Sigma 2 return address catalogue entry.
8. Extract the address and NENTRY value for the subroutine in the Sigma 5.
9. Transfer to the 5 subroutine to perform the desired computation.
10. Return control to SUPE5.
11. Extract the return address in the Sigma 2 software package.
12. Transmit the common data block to the Sigma 2.
13. Enable an interrupt requesting execution of SUPE2.
14. Terminate Sigma 5 computation.
15. Transfer to the return address in the Sigma 2.

Note that the above procedures permit most of the complex transfer control logic to reside in the Sigma 5, thus minimizing Sigma 2 memory storage requirements.

A readily apparent problem is that of initially starting program execution. This problem is circumvented by using an IF statement to compare the value of a program variable ISTART to 9999 as the first statement in SUPE5. Since the XDS Sigma 5 sets all program storage to zero during loading, the initial value of ISTART is zero. Thus equality is initially violated, and a branch occurs to a section of SUPE5 which performs initialization and sets ISTART = 9999. Thus subsequent entries to SUPE5 will skip the initialization step.

4.3 Cataloging Transfer Data

Each time a transfer is initiated between the Sigma 5 and Sigma 2 or vice versa it is necessary to store the following information:

1. The identification of the subprogram to which transfer is desired.
2. The value of NENTRY which appears in the destination subroutine parameter list when it is called.
3. The identification of the subroutine to which return is desired when the computations in the destination software are complete.
4. The value of NENTRY to be used in the subroutine parameter list when the return address subroutine is called.

In order to facilitate this process the subroutines' names are each associated with a distinct identifying number, as indicated in Table 4.3.1. Each time a transfer is initiated numbers corresponding to 1, 2, 3 and 4 above are stored by inserting them as parameters in subroutine MARK which is then called with NENTRY = 1. For example:

CALL MARK (1, 24, 4, 4, 3) (4.3.1)

indicates that a catalog entry is to be generated. A transfer to subroutine TYPCON (4) is desired. When the operations designated in TYPCON (4) are complete, MAINB (3) should be called. Instruction (4.3.1) stores the address information and identifies the new catalog entry by incrementing an integer ITRANS which is equal to the number of entries which have been generated.

The latest address data may be extracted from the catalog by calling MARK with NENTRY = 2 or 3. The instruction:

CALL MARK (2, NFLGA, NFLGB, IA, IA) (4.3.2)

sets NFLGA, NFLGB equal to the destination subroutine identification number and NENTRY value respectively. The return address is produced by the call:

CALL MARK (3, IA, IA, NFLGA, NFLGB) (4.3.3)

which also deletes the present address data by decrementing ITRANS.

Table 4.3.1

SUBROUTINE	SIGMA 5 IDENTIFICATION NUMBER (NFLGA)	SIGMA 2 IDENTIFICATION NUMBER (NFLGA)
SIMSYS	1	
MFCS	2	
MAINA	3	
MAINB	4	
FSMDL	5	
MIRMDL	6	
RESPON	7	
ACTCAL	8	
MIRCAL	9	
ACTMDL	10	
ACTCMD	21	1
EAMCS	22	2
FIGSEN	23	3
TYPCON	24	4

4.4 Determination of the Transfer Address

The addressing structure of SUPE5 provides the subroutine identification number NFLGA and the entry point NFLGB for each Sigma 5 - 2 or 2 - 5 transfer. Since SUPE5 directly controls transfers to any desired subroutine and entry point, it is necessary to provide some method of indicating whether or not the transfer is to a destination or a return address. This is accomplished by counting the number of catalog entries. The number of destination - return addresses is stored in ITRANS which is incremented every time a new catalog entry is generated. The value of ITRANS at the completion of the last use of the addressing structure is stored in ISTORE. ITRANS is compared with ISTORE at the beginning of the addressing structure. If ITRANS is greater than ISTORE, the current catalog entry is new and a destination address is generated by calling MARK with NENTRY = 2. If on the other hand, ITRANS equals ISTORE a return address is desired and MARK is called with NENTRY = 3. A request for a return address signifies the completion of a cataloged Sigma 2 - 5 or 5 - 2 transfer and automatically deletes the current catalog entry by decrementing ITRANS.

4.5 Experimental Active Mirror Simulation

Simulation of the experimental active mirror is accomplished by combining a simulation control structure (RESPON) with the actual mirror control software (ACTCMD, EAMCS, FIGSEN, and TYPCON) and software models of the hardware components (ACTMDL, FSMDL, MIRMDL).

A block diagram of the simulation is shown in Fig. 4.5.1. The control system is operated for one cycle by calling EAMCS with NENTRY = 4 and NTIMSQ = 1.

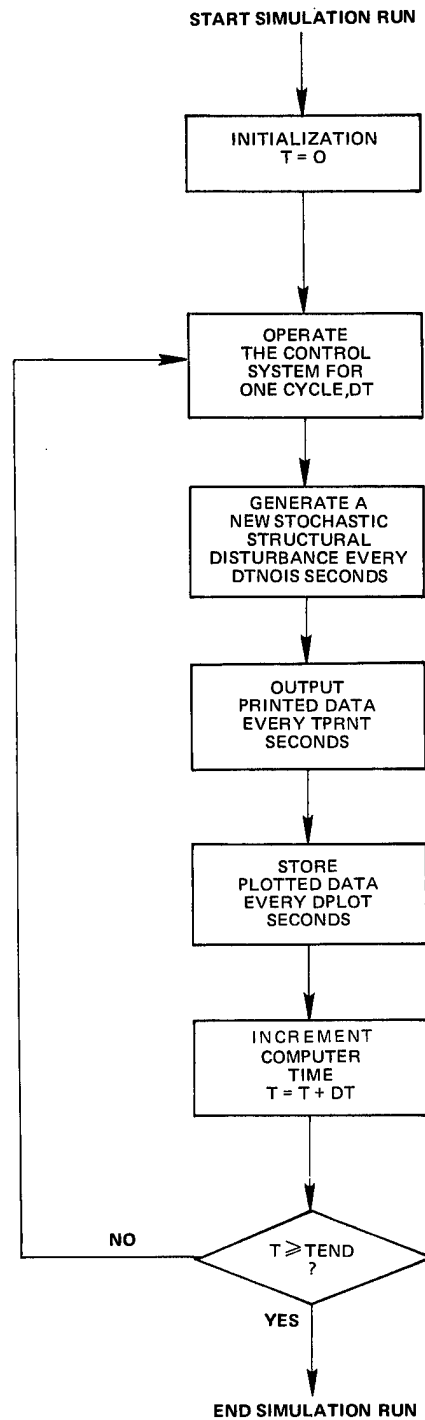


Fig. 4.5.1 Simplified block diagram of the EAM simulation.

Provision is made for the inclusion of a stochastic structural disturbance generator to simulate the effect of random orbital disturbances. A new random disturbance is requested every DTNOIS seconds.

The simulation is designed to output data every TPRNT seconds on the Sigma 5 line printer. The data illustrated in Fig. 4.5.2 includes the time, T, the performance index, PINDEX, actual figure error, XFAV, sensed figure error, XFV, commanded figure control, UFV, and the actual figure control, UFAV. Diagnostic information, useful for figure control system development, is also provided.

Data for online or offline plotting is processed and stored every DTPLOT seconds by subroutines PLRT and STORED.

The simulation run is terminated when the simulation time exceeds the input limit TEND. On termination program control is returned to SIMSYS.

4.6 Experimental Active Mirror Experiment Operation

Operation of the experiment is achieved by selecting the appropriate operating mode configuration using the mode descriptions in section 5.2.2. The OPERATE CONTROL SYSTEM, USE HARDWARE COMPONENTS and SIGMA 5 - 2 CONFIGURATION modes must be selected. Once all input has been read, TYPCON will request instructions from the remote control terminal for initializing and starting the experiment. A detailed description of the remote control functions is contained in Chapter 6. Once operation is achieved, the hardware components function under the control of EAMCS which realizes the control sequence illustrated in Fig. 3.4.2.

A simplified flow diagram of EAMCS is shown in Fig. 4.6.1. The major part of EAMCS is associated with the acquisition of figure error data. The collected measurements are processed by the Sigma 5 to eliminate measurement ambiguities and to produce a new figure control. The remaining portion of EAMCS operates the actuator control systems, controls the real-time control cycle duration, and provides an experiment interrupt provision which enables the operator to suspend operation for diagnostic or other purposes via the remote terminal. The experiment is automatically terminated when NTIMSQ* control cycles have elapsed.

4.7 EAM Software Descriptions and Functional Block Diagrams

The following subsections contain brief descriptions and functional block diagrams of the major components of the EAM software. The program descriptions and functional block diagrams are arranged in alphabetical order.

4.7.1 ACTCAL: Actuator Calibration

ACTCAL provides the software necessary to check the figure actuators for correct operation. The software perturbs each element of the actuator command vector m_{ci} by $\pm \delta_{aa}$ and observes the corresponding change in the measured output m_{mi} . The command perturbation - output measurement sequence is repeated n_{ma} times. The average ratio between the output and input perturbations is then calculated using the relationship:

$$\frac{\delta m_{mi}}{\delta m_{ci}} = \frac{1}{n_{ma}} \sum_{j=1}^{n_{ma}} \frac{1}{2\delta_{aa}} \left[\left(m_{mi} \right)_{m_{ci} = \delta_{aa}} - \left(m_{mi} \right)_{m_{ci} = -\delta_{aa}} \right]_j \quad (4.7.1)$$

* NTIMSQ = NTIMS

The actuator output command perturbations and output measurements are executed by EAMCS via ACTCMD. A flow diagram of ACTCAL is shown in Fig. 4.7.1.

4.7.2 ACTCMD: Figure Actuator Control Systems

ACTCMD provides the necessary closed loop control for the mirror figure actuators. ACTCMD also transfers the actuator commands to the actuator models and returns the actuator model outputs in the EAM simulation mode. Provisions are also made to freeze the actuator positions whenever computational authority is transferred to the Sigma 5. A detailed discussion of the actuator control algorithms is given in sections 3.6 and 3.7. Figure 4.7.2 shows a flow diagram of ACTCMD.

4.7.3 ACTMDL: Mirror Figure Actuator Models

ACTMDL provides software models for the force and position figure actuators. ACTMDL must be called for each figure actuator which is identified in the parameter list by the index IENTRY. A detailed description of the actuator models is given in section 2.9. A flow diagram of ACTMDL appears in Fig. 4.7.3.

4.7.4 EAMCS: Hardware Component Control Software

The Sigma 2 subroutine EAMCS provides the core of the real-time figure control system. In this routine the physical characteristics of the hardware components are considered. The primary functions of EAMCS are to provide: an actuator control time interval during which the actuators can reach a steady state; actuator freeze signals to inhibit actuator motion during figure error measurement and data processing; position commands to the figure sensor image dissector; figure sensor data acquisition and storage; and actuator control system commands. Further discussion of

the real-time control system is given in sections 3.4., 4.6 and Chapter 5. A flow diagram of EAMCS is given in Fig. 4.7.4.

4.7.5 FIGSEN: Figure Sensor Control Module

Subroutine FIGSEN provides the communications link between the EAM software and the mirror figure sensor. FIGSEN transfers position commands to the image dissector and interrogates the output of the figure sensor phase detector fitter. In the simulation mode FIGSEN obtains figure sensor output data from the figure sensor model FSMDL. Figure 4.7.5 shows a flow diagram of FIGSEN.

4.7.6 FSMDL: Figure Sensor Model

FSMDL provides a software model of the figure sensor. FSMDL also provides software for reading the figure sensor model data, calculating performance indices, and constructing the plot data transfer vector XV. A flow diagram of FSMDL is presented in Fig. 4.7.6.

4.7.7 MAINA: EAM System Computations

MAINA inputs all the sequence timing data for the digital control system. MAINA also reads in the gains for the figure and actuator control algorithms, the saturation limit $m_{\max}$ on the actuator commands and the figure sensor data processing algorithm parameter values. MAINA provides calls to the EAM component models and performance index generator data input. In the initialization mode MAINA provides coding and/or subroutine calls to initialize the entire EAM system. During EAM operation or simulation MAINA provides software for figure sensor data processing and actuator command computation. A flow diagram of MAINA appears in Fig. 4.7.7.

4.7.8 MAINB: Remote Control Operations

MAINB provides service operations on EAM program variables for the remote terminal. The service operations include the interpretation of an input variable identification, variable display, and variable modification. MAINB also provides computations required to process data for display on the remote terminal during experiment operation. A flow diagram of MAINB is shown in Fig. 4.7.8.

4.7.9 MAINC: Initial Alignment Computations

MAINC provides coding for the mirror segment tilt and axial alignment control systems. MAINC reads in all the required data and provides coding for the segment actuator command computations using figure and ambiguity sensor data. A flow diagram of MAINC is shown in Fig. 4.7.9.

4.7.10 MFCS: Control System Data Input and Gain Matrix Computation

Subroutine MFCS reads all the basic information for the experimental active mirror control system and simulation. If MODOP = 1 or 2 MFCS computes the simplified linear or linear optimal feedback gain matrix respectively. A value of MODOP = 3 results in the input of a general n_r by n gain matrix as part of the input data deck. MFCS also provides calls to ACTCAL and MIRCAL for actuator and mirror tests. Figure 4.7.10 shows a flow diagram of MFCS.

4.7.11 MIRCAL: Mirror Calibration

Subroutine MIRCAL provides the software required to experimentally evaluate the reduced deformation-force matrix A_r of the mirror which

relates deformations at selected points on the mirror surface to perturbations δ_{af} in the actuator outputs. The reduced matrix is measured n_{mf} times and the results averaged. The relationship between the deformation at the i th measurement point x_{fmi} due to a change in the j th actuator output m_{cj} is:

$$a_{ij} \approx \frac{1}{n_{mf}} \sum_{k=1}^{n_{mf}} \frac{1}{2\delta_{af}} \left[(x_{fmi})_{m_{ci} = \delta_{af}} - (x_{fmi})_{m_{ci} = -\delta_{af}} \right]_k \quad (4.7.2)$$

The resulting matrix is displayed on the Sigma 5 line printer. MIRCAL is shown in flow diagram form in Fig. 4.7.11.

4.7.12 MIRMDL: Mirror Model

MIRMDL provides a linear representation of the deformable or segmented mirror for use in the EAM simulation. The equation of the mirror model has the form:

$$x_f = x_d + A_r m_m \quad (4.7.3)$$

where x_f is the figure error; x_d the initial figure disturbance; A_r the reduced position-position or deformation-force matrix and m_m the position or force actuator outputs. A flow diagram of MIRCAL appears in Fig. 4.7.12.

4.7.13 PINDX: Performance Index Generator

The subroutine PINDX accepts the n dimensional array x and returns a performance index evaluated using the equation:

$$J = \left[\sum_{i=1}^n w_i x_i^2 \right]^{1/2} \quad (4.7.4)$$

where w is an input array of positive weights and x is a parameter array. For an unbiased index $w_i = n^{-1}$ $i = 1, n$. Figure 4.7.13 shows a flow diagram of PINDX.

4.7.14 PLRT: Plotted Data Storage and Scaling

Subroutine PLRT provides the software required to store, scale and plot simulation data. Potential data for plotting must be stored in XV. The NPLOTV elements of XV to be stored every DTPLOT seconds are identified by the elements of IPLOTV. The scales to be used when each element is plotted are stored in the array SCALV. MODV contains elements which indicate whether or not the corresponding element of XV is to be plotted using the input scale factor or a scale factor produced automatically. If IMODV (I) is 2, a scale factor is automatically generated for the data corresponding to XV (IPLOTV (I)). Automatic scale factor generation is omitted if IMODV (I) = 1. A flow diagram of PLRT is delineated in Fig. 4.7.14.

4.7.15 RESPON: Simulation Control Software

Subroutine RESPON provides the control structure for the EAM Simulation. RESPON initializes the EAM models and control software and the simulation data collection and processing programs. During simulation RESPON provides calls to EAMCS every control cycle, artificial real-time generation, stochastic structural disturbance inputs, as well as outputs for the line printer and the data plotting routine. See Fig. 4.7.15 for a flow diagram of RESPON.

4.7.16 SIMSYS: Program Control Module

SIMSYS provides the computations and/or calls required required to read in all EAM system data. Provisions are also included to permit a number of simulation runs to be performed automatically, each with data modifications provided by a data editing capability. Thus it is possible to make up to ten simulation runs with different values of β_g for example without the necessity of reloading the data deck. In the hardware operating mode SIMSYS allows the experiment to select any of the possible edited data configurations via remote terminal commands. SIMSYS is flow charted in Fig. 4.7.16.

4.7.17 STORED: Plotting Control Software

Subroutine STORED provides the coding required to plot the data prepared by PLRT using Calcomp plotting routines. Figure 4.7.17 shows a flow chart of STORED.

4.7.18 SUPE2: Main Sigma 2 Program

SUPE2 is the main program resident in the Sigma 2. SUPE2 provides a computed transfer to the Sigma 2 subroutine identified by the value of NFLGA.

4.7.19 SUPE5: Main Sigma 5 Program

The main program in the Sigma 5 provides the software required to store Sigma 2 transfer data; to extract subroutine address data from the transfer file and to transfer to subroutines in the Sigma 5. Refer to Fig. 4.7.19 for a functional block diagram of SUPE5.

READ DATA

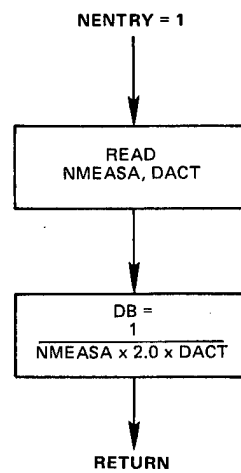


Fig. 4.7.1 ACTCAL flow diagram.

TEST ACTUATORS

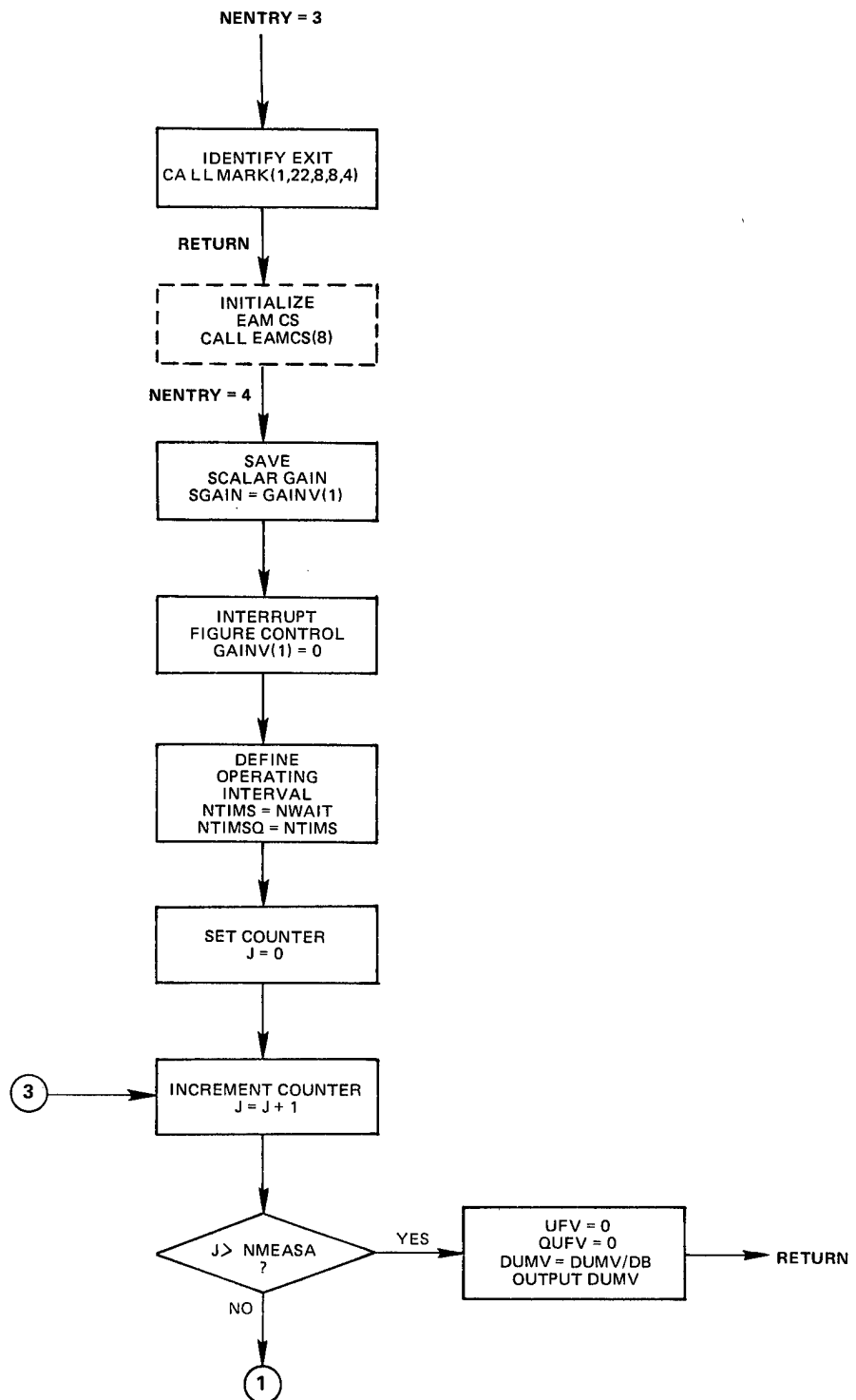


Fig. 4.7.1 Cont.

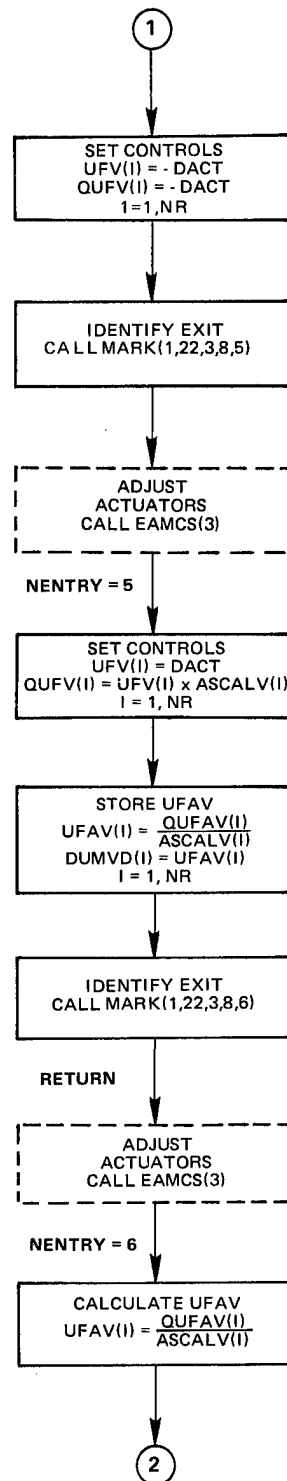


Fig. 4.7.1 Cont.

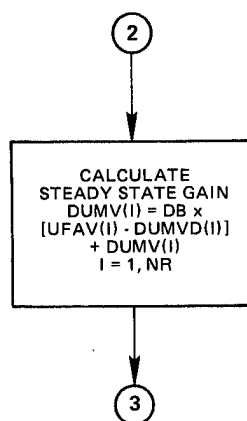


Fig. 4.7.1 Cont.

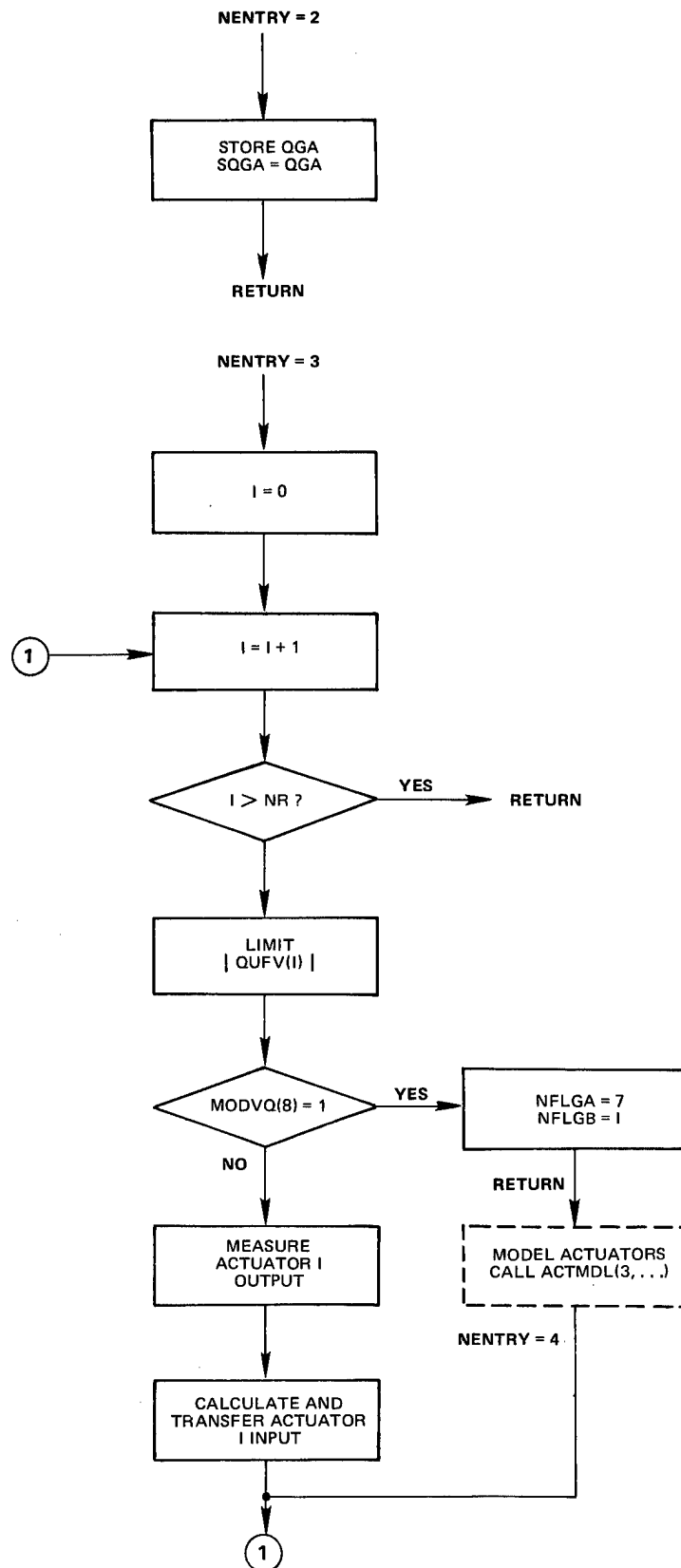


Fig. 4.7.2 ACTCMD flow diagram.

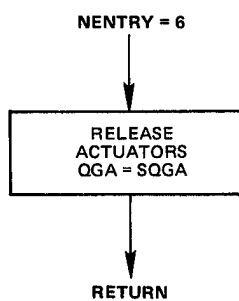
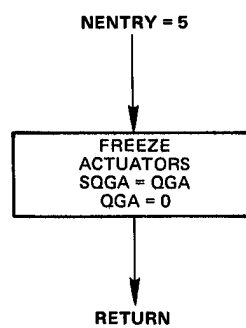


Fig. 4.7.2 Cont.

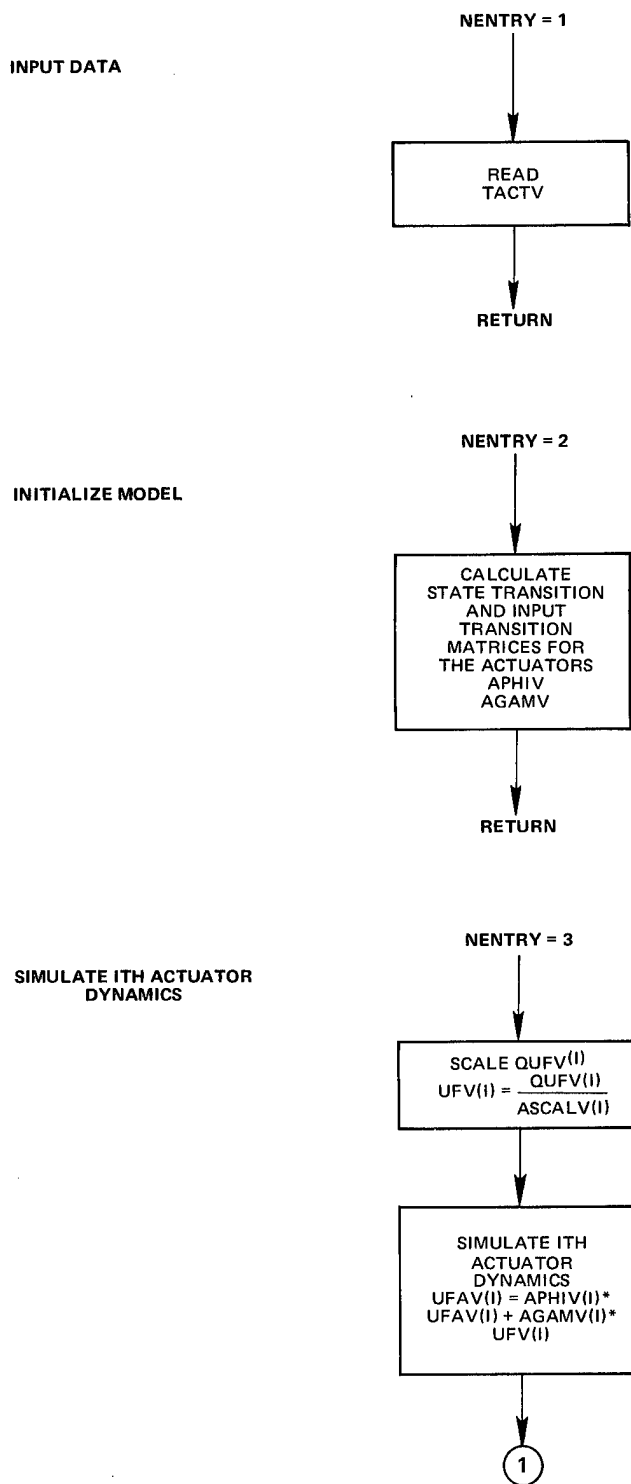


Fig. 4.7.3 ACTMDL flow diagram.

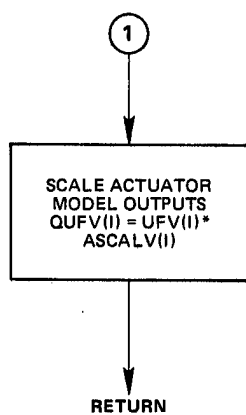


Fig. 4.7.3 Cont.

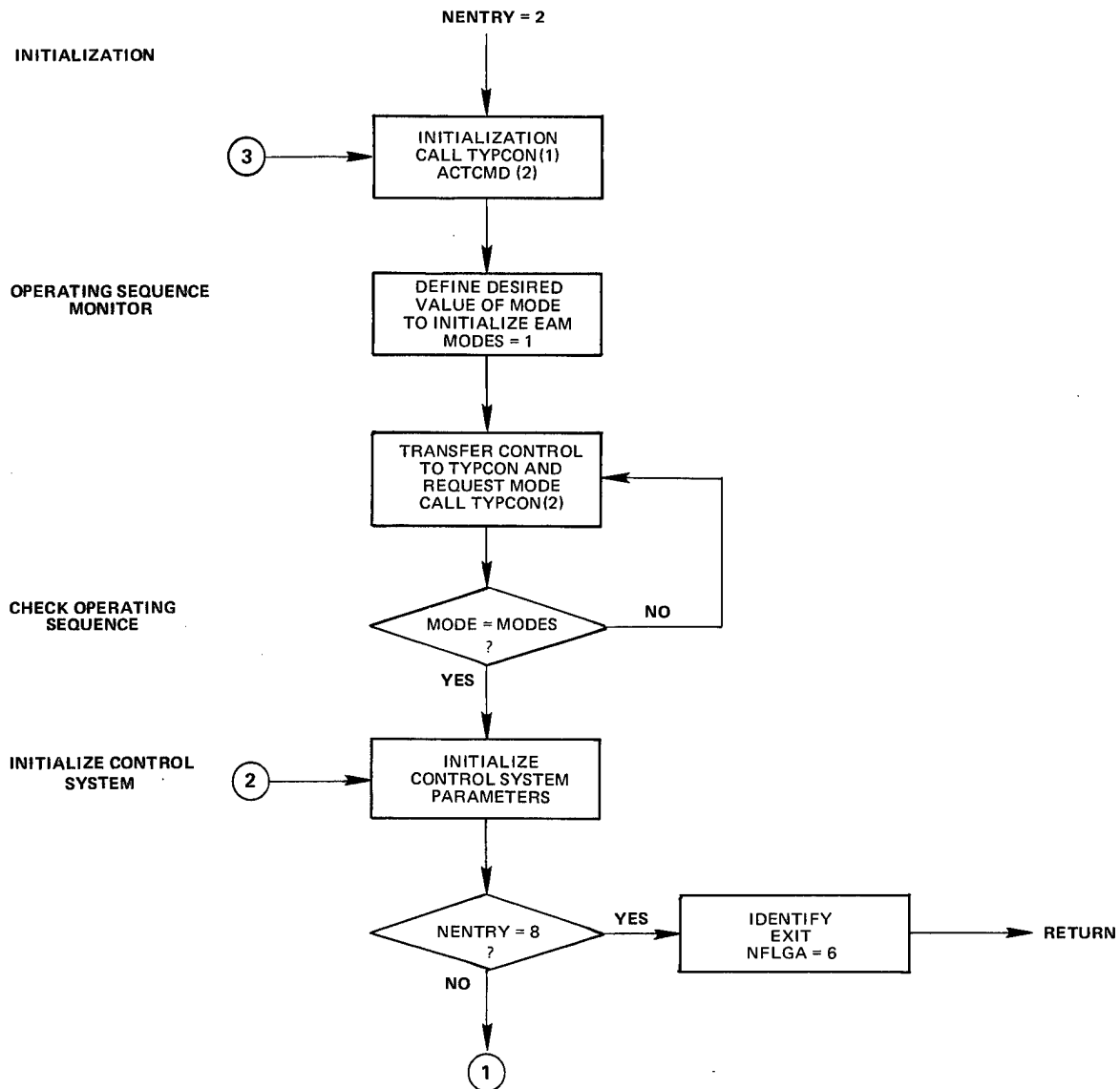


Fig. 4.7.4 EAMCS flow diagram.

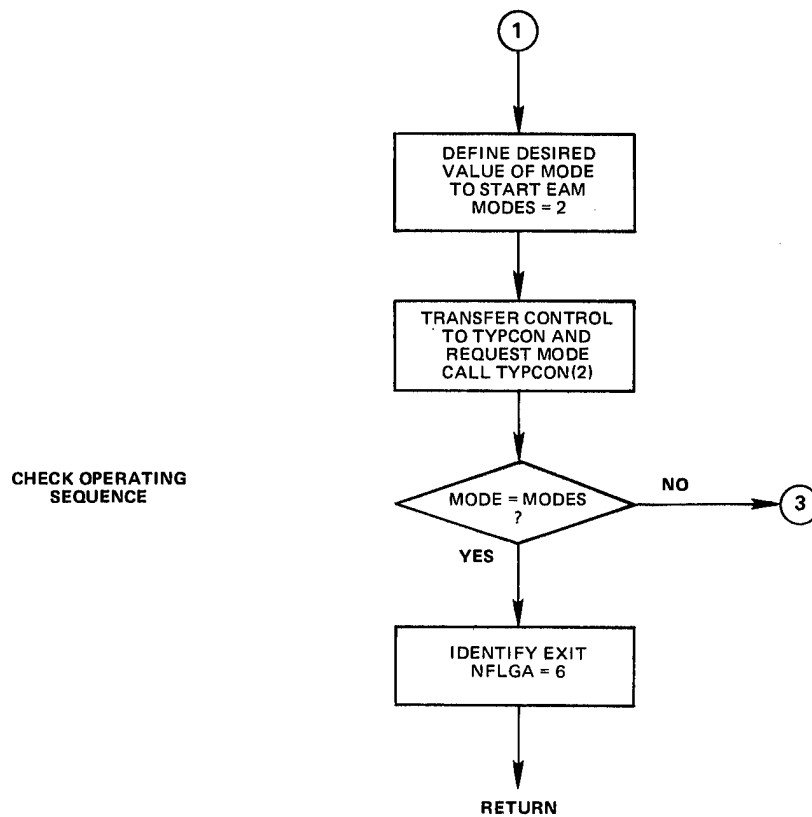


Fig. 4.7.4 Cont.

EXECUTE CONTROL
SYSTEM FOR NTIMSQ
CYCLES

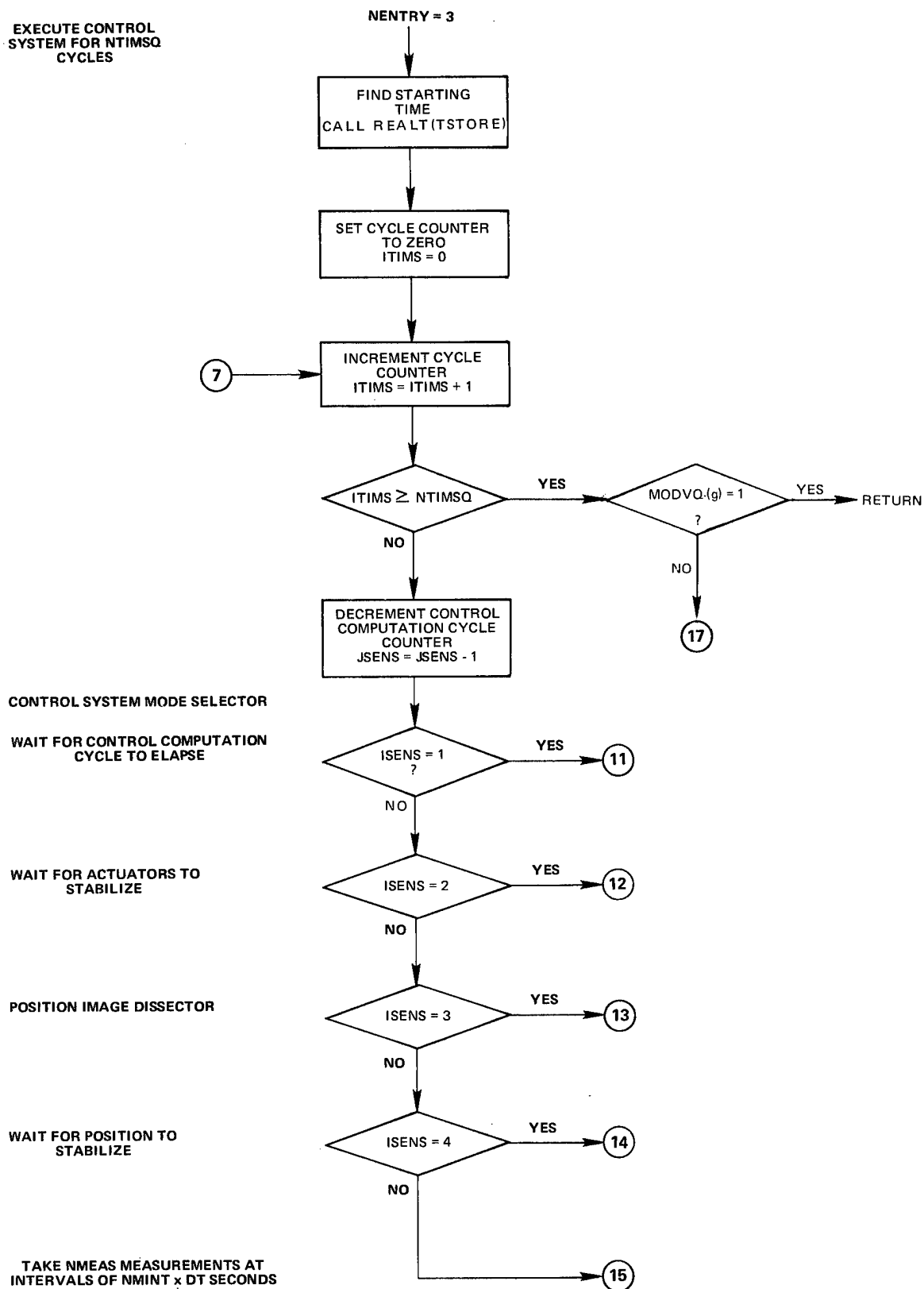
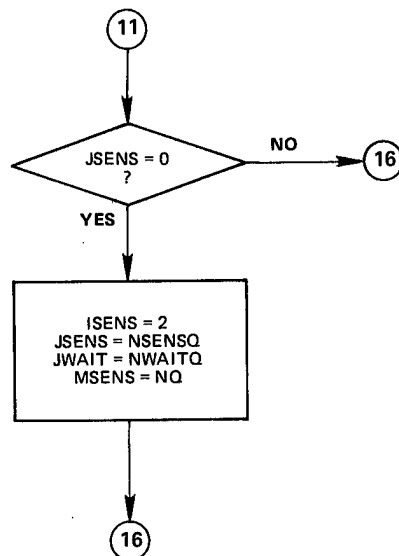


Fig. 4.7.4 Cont.

WAIT FOR CONTROL COMPUTATION
CYCLE TO ELAPSE



WAIT NWAIT CYCLES FOR
ACTUATORS TO STABILIZE

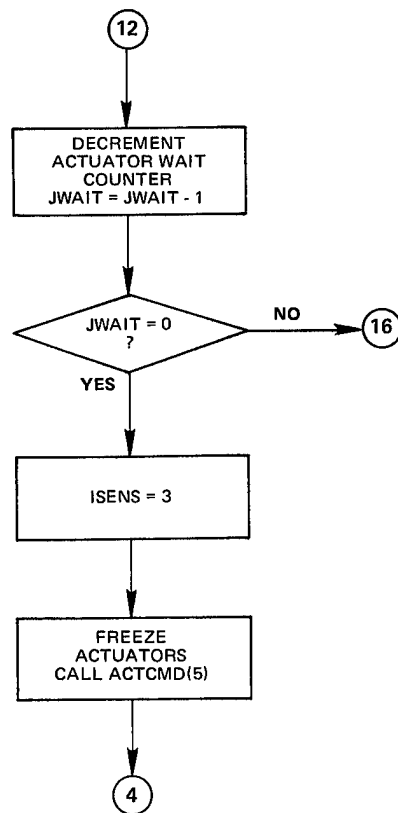


Fig. 4.7.4 Cont.

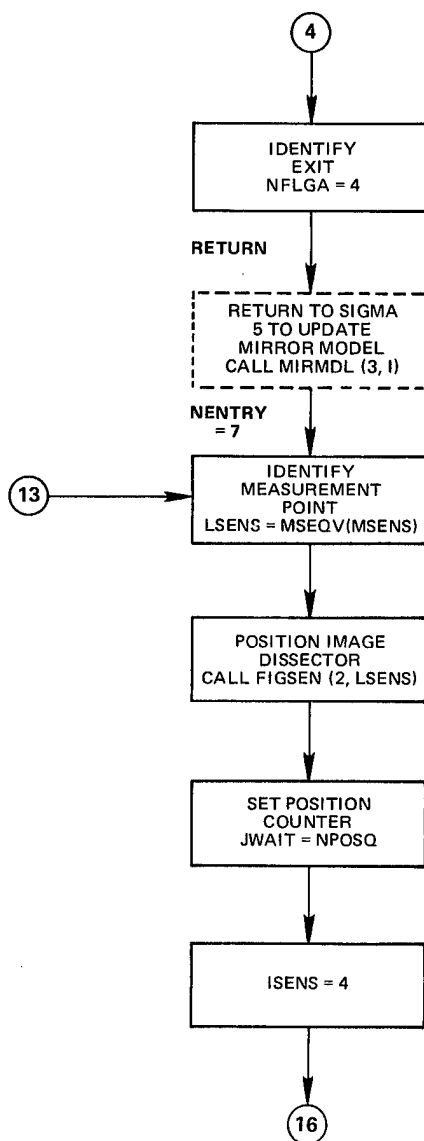


Fig. 4.7.4 Cont.

WAIT FOR MEASUREMENT POSITION
TO STABILIZE

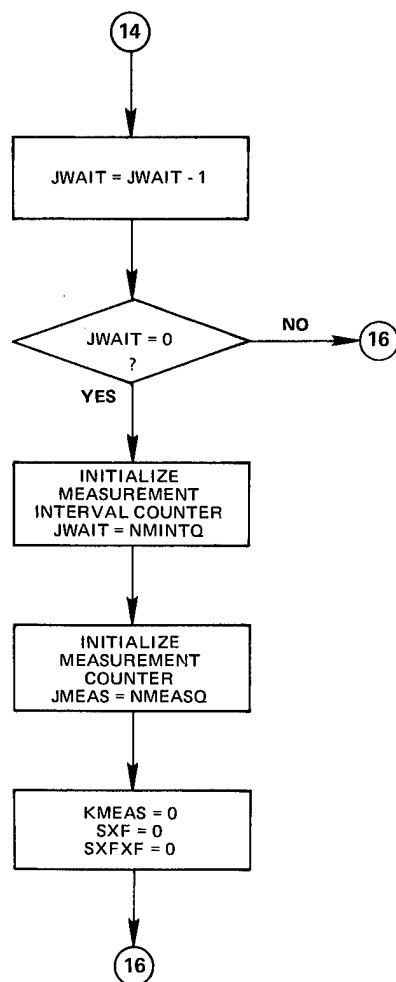


Fig. 4.7.4 Cont.

WAIT FOR NMINTQ CYCLES
BEFORE MEASUREMENT

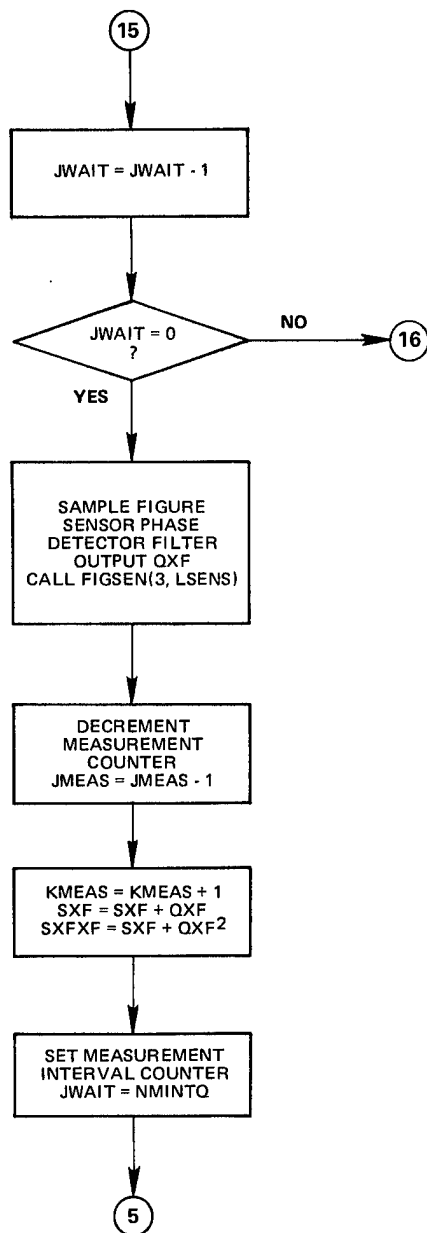


Fig. 4.7.4 Cont.

HAVE NMEAS MEASUREMENTS
BEEN MADE

HAVE ALL N POINTS
BEEN EXAMINED

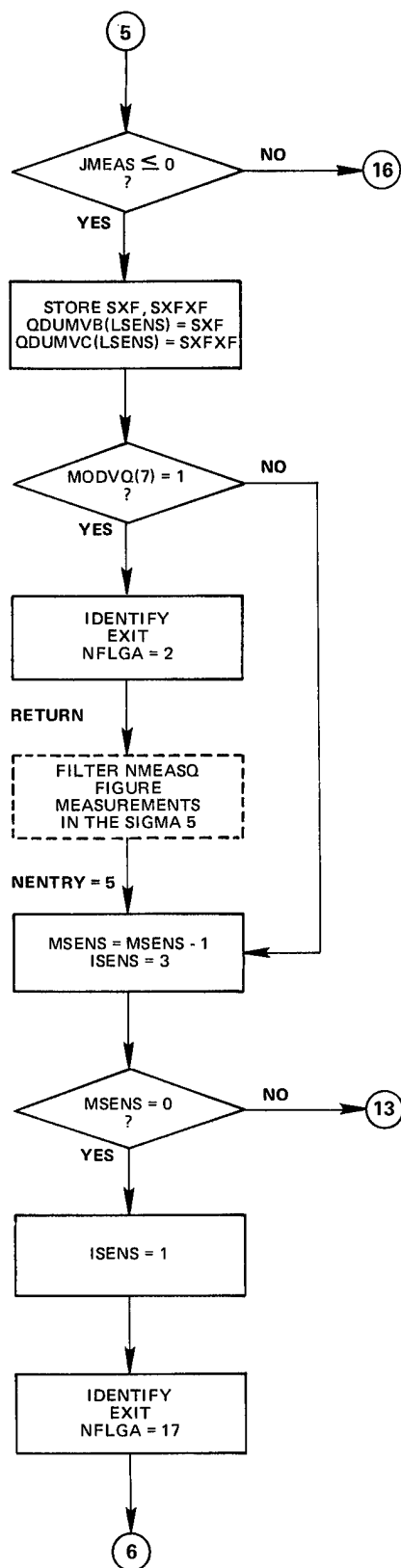


Fig. 4.7.4 Cont.

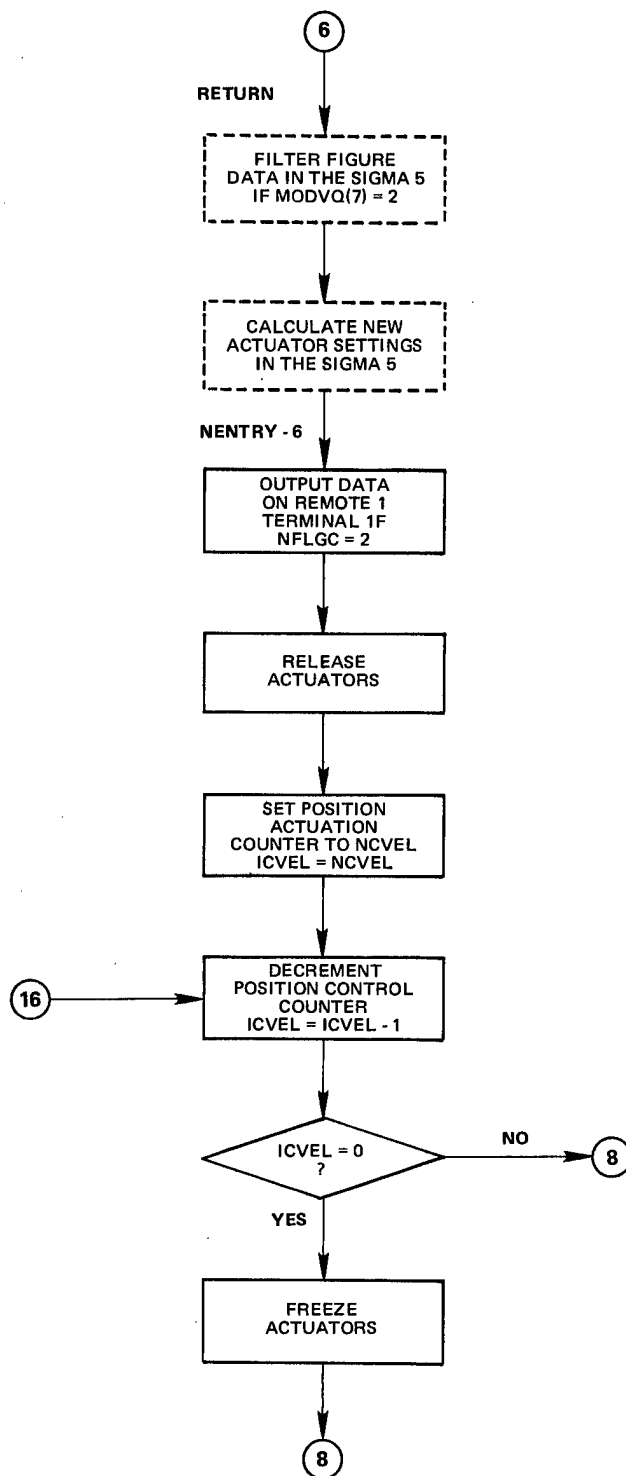


Fig. 4.7.4 Cont.

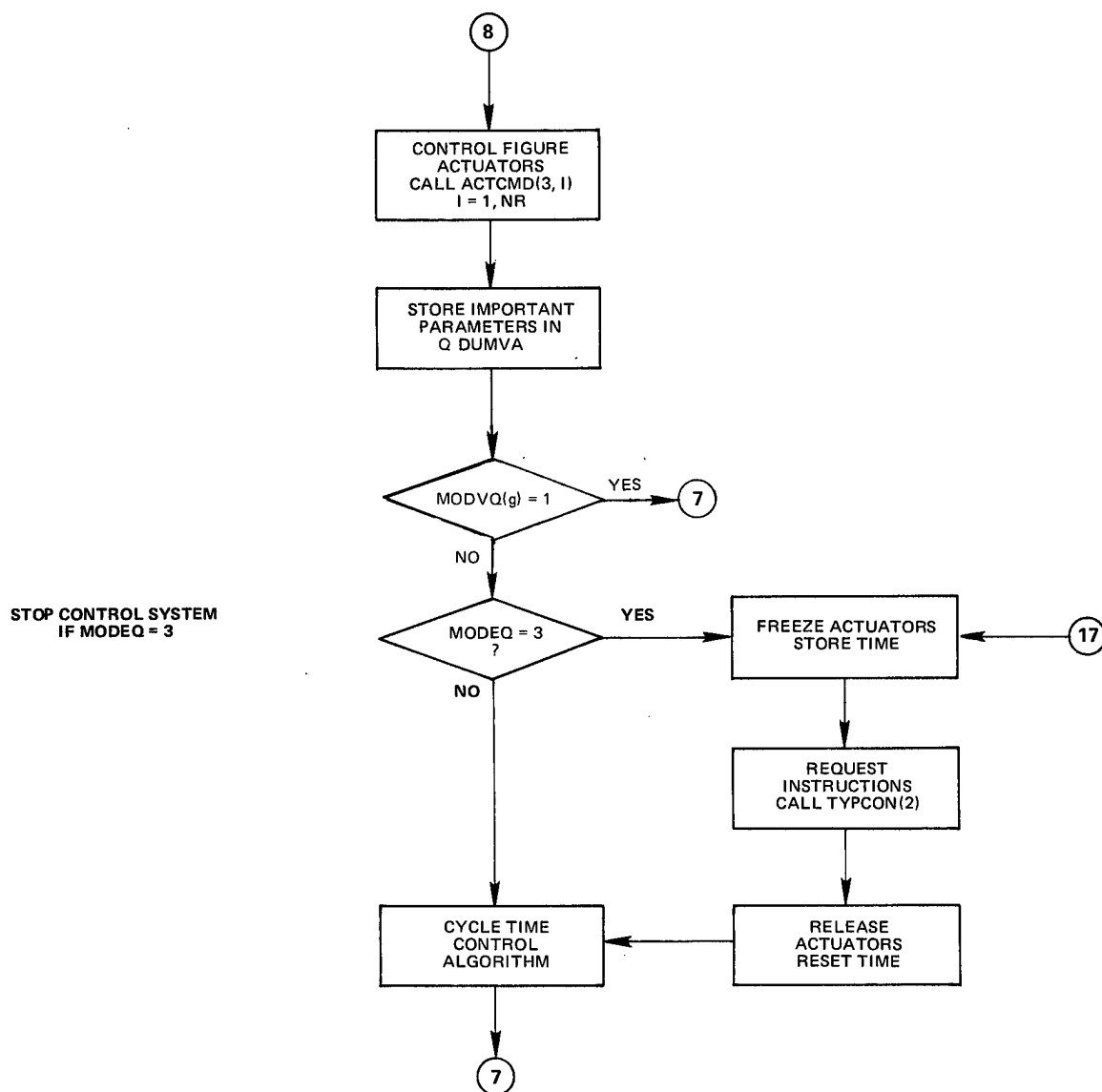


Fig. 4.7.4 Cont.

POSITION IMAGE DISSECTOR

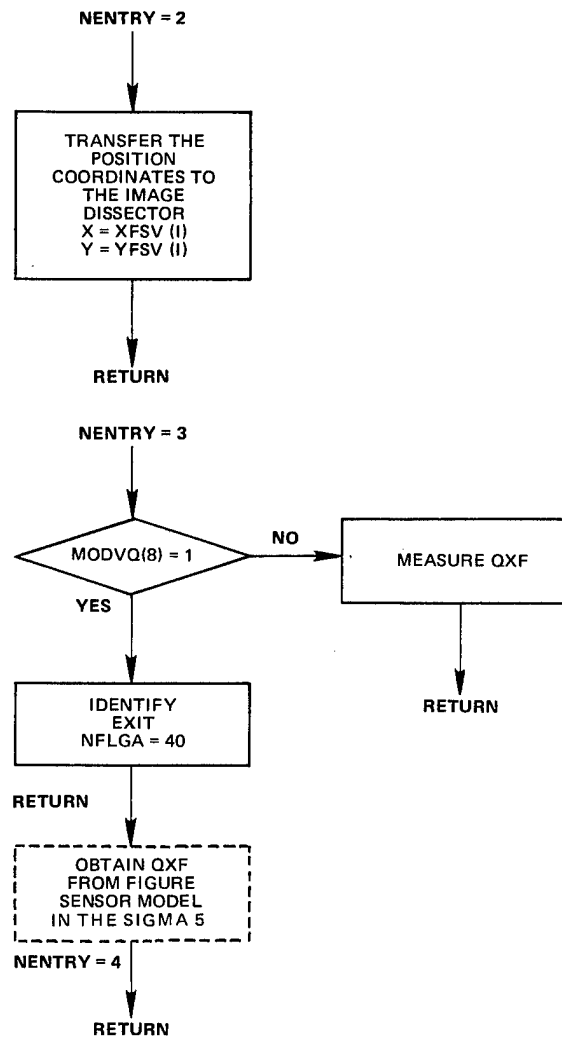
MEASURE FIGURE SENSOR
PHASE DETECTOR FILTER
OUTPUT

Fig. 4.7.5 FIGSEN flow diagram.

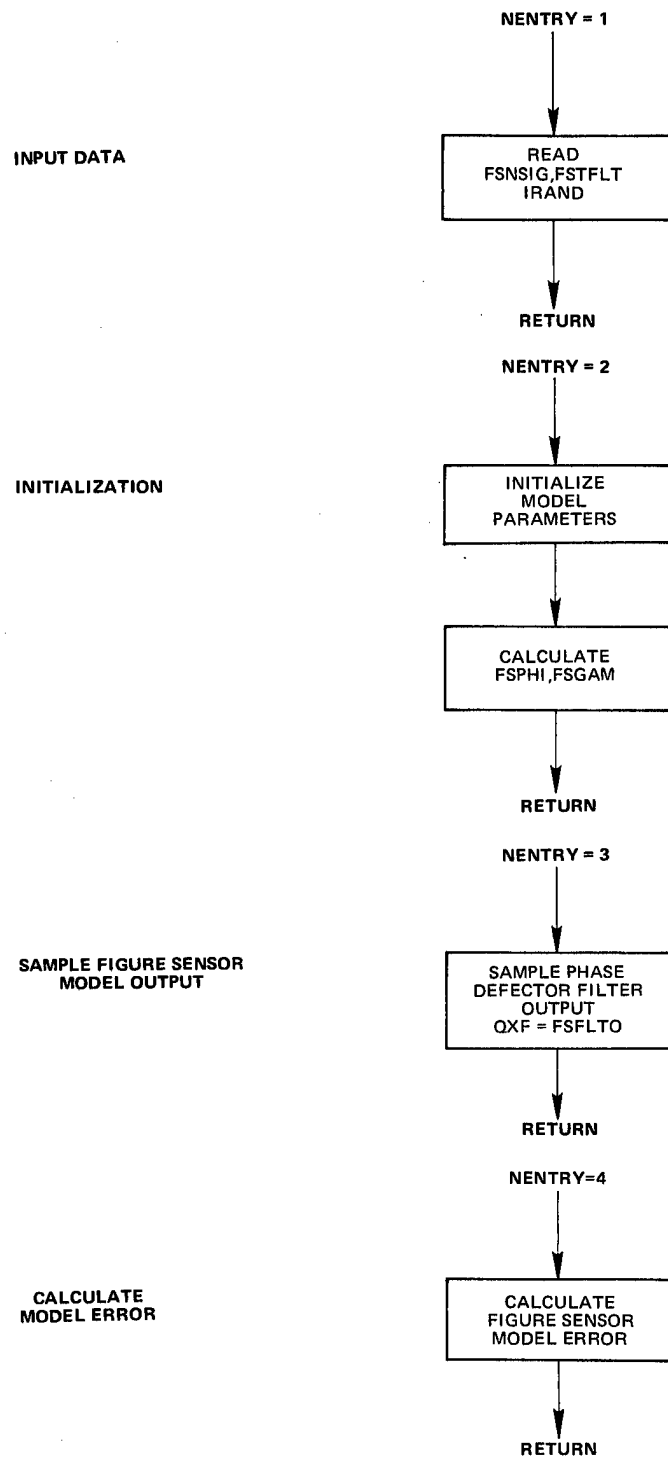


Fig. 4.7.6 FSMDL flow diagram.

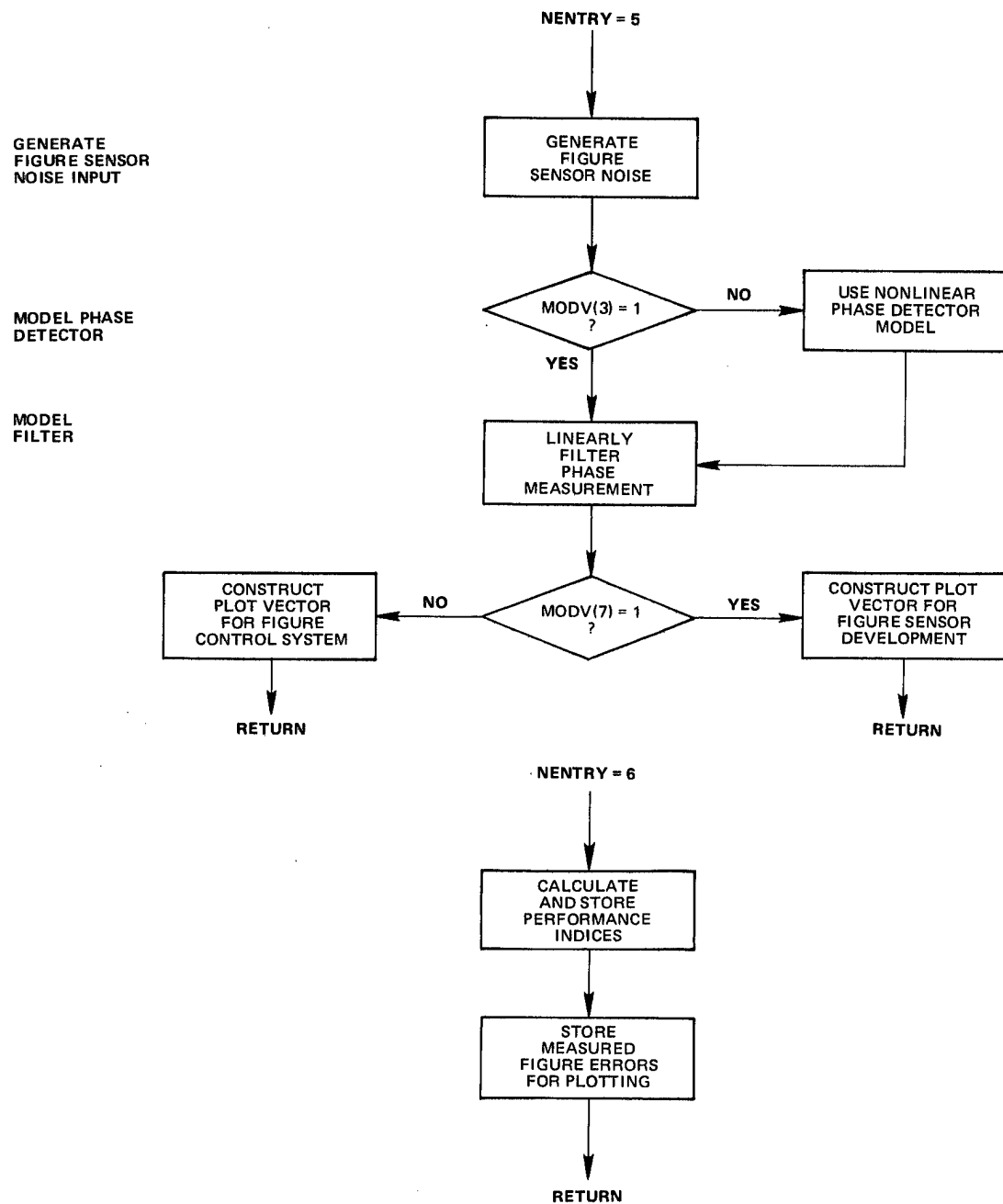


Fig. 4.7.6 Cont.

INPUT DATA

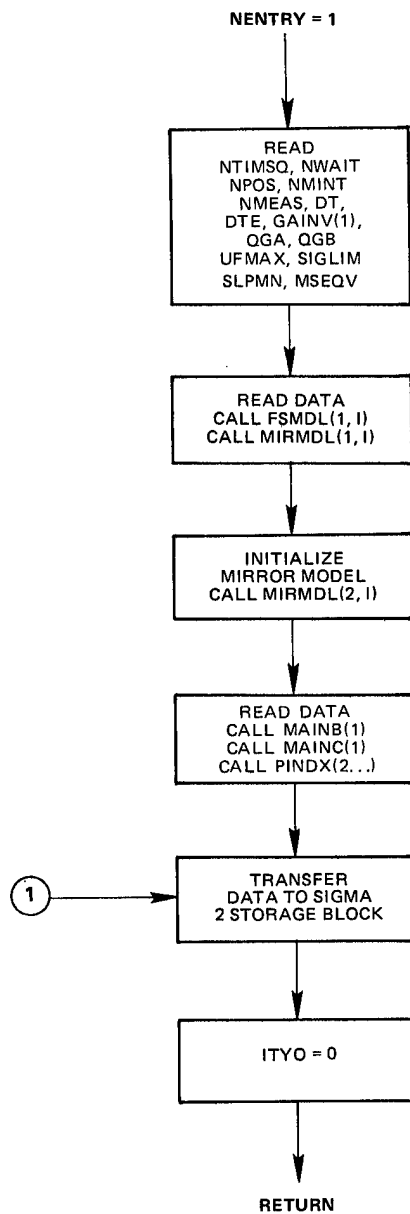


Fig. 4.7.7 MAINA flow diagram.

INITIALIZATION

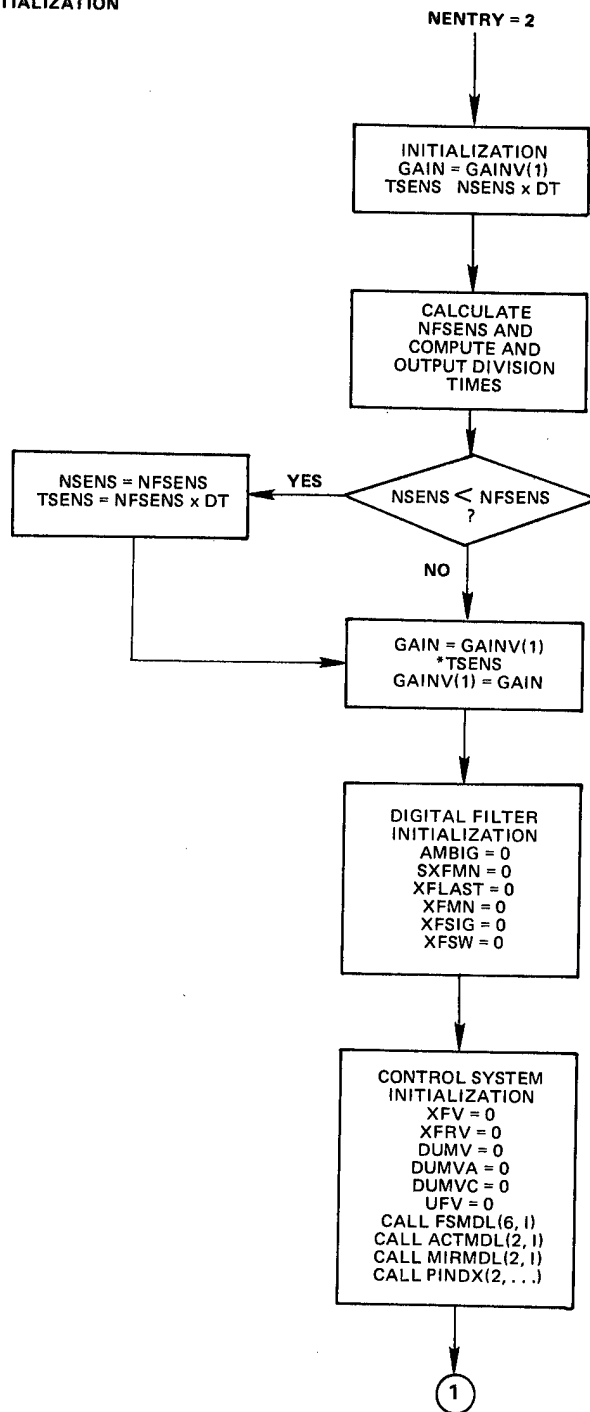


Fig. 4.7.7 Cont.

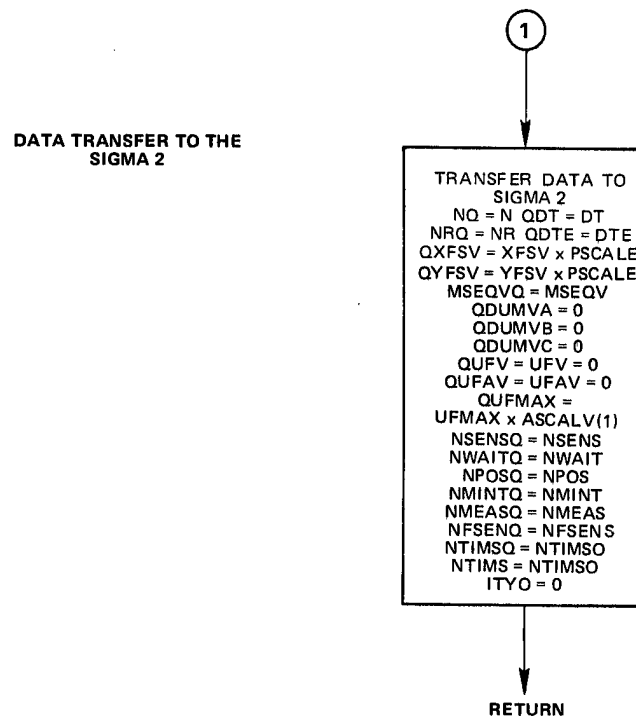


Fig. 4.7.7 Cont.

CONTROL CALCULATION

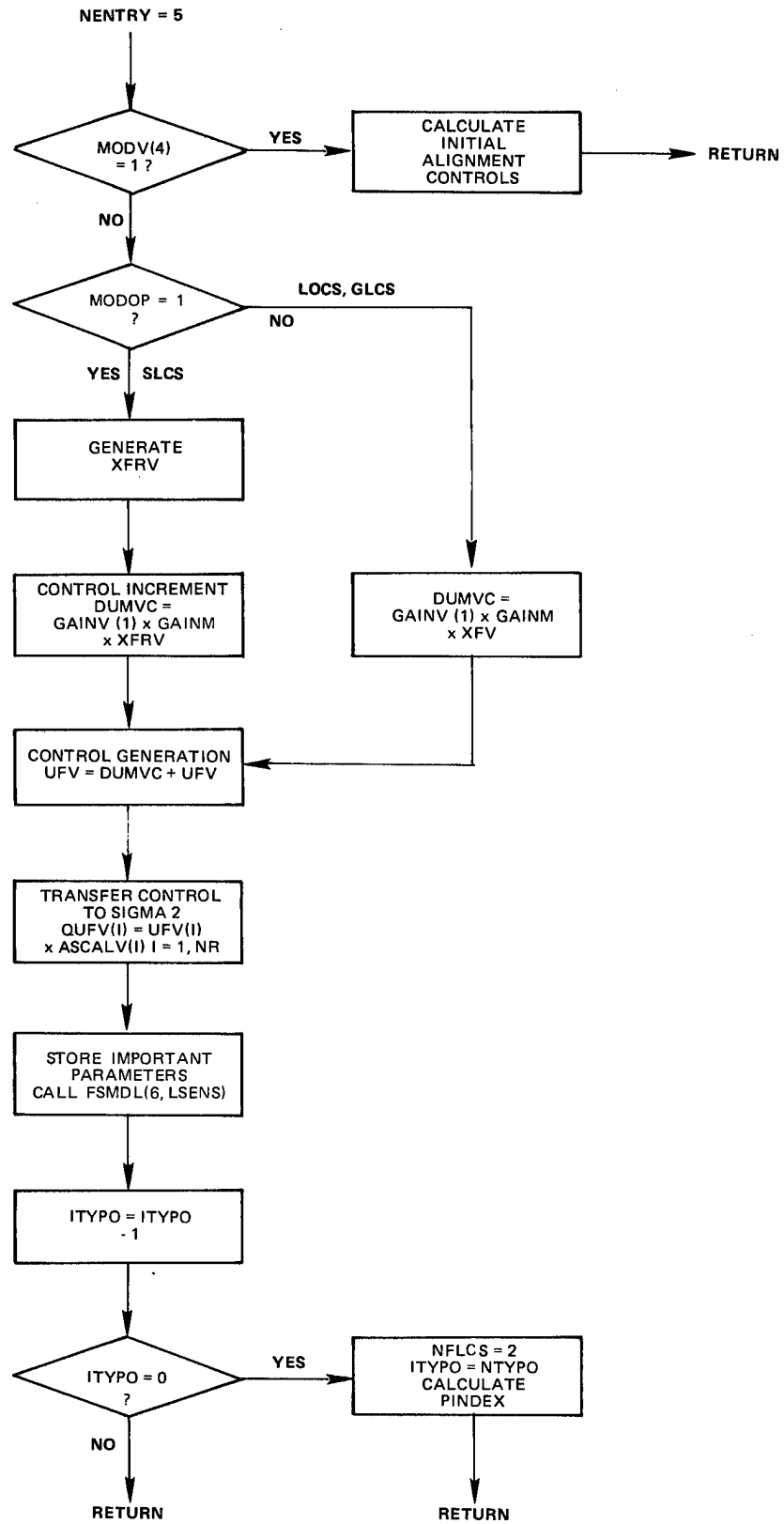


Fig. 4.7.7 Cont.

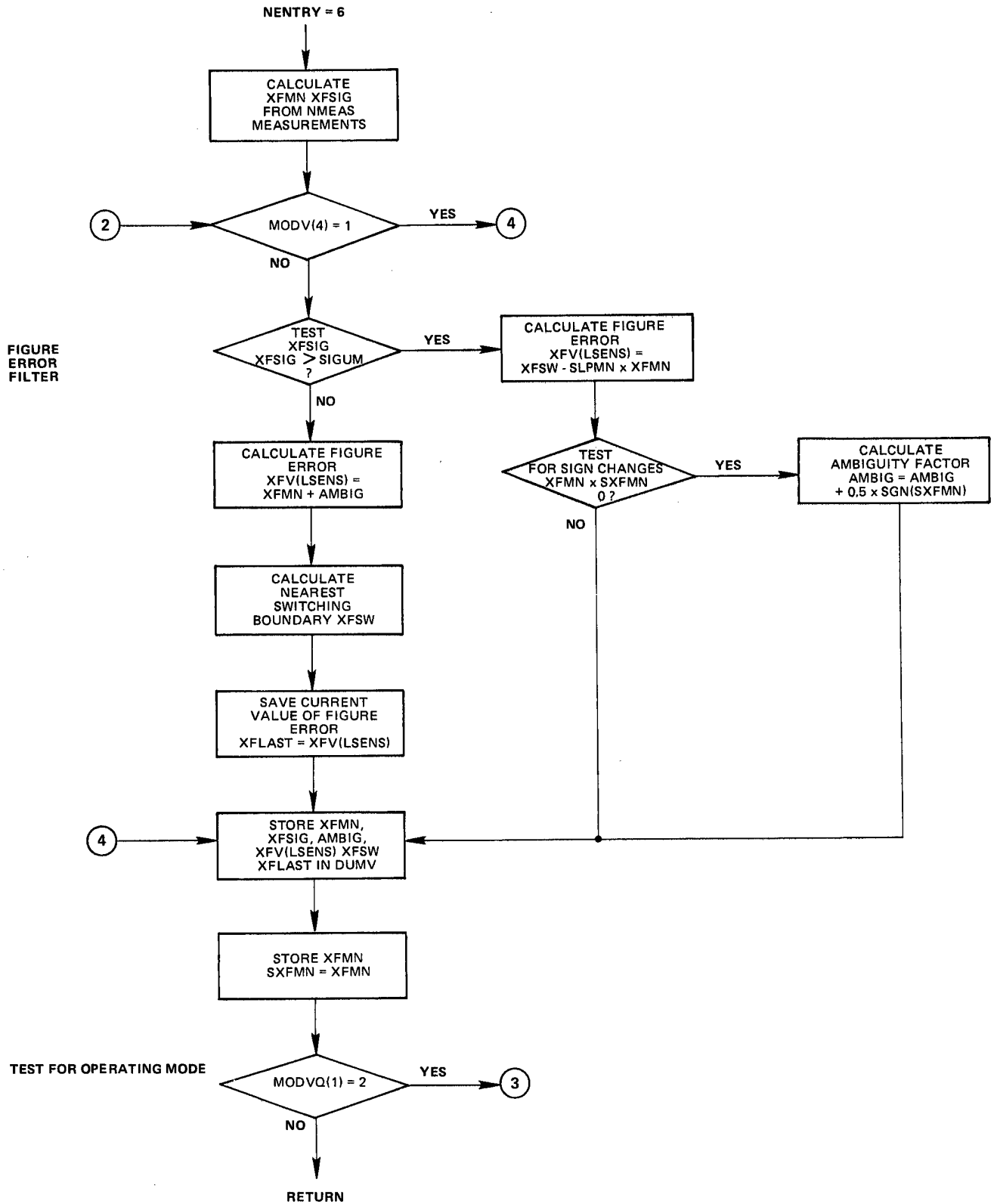


Fig. 4.7.7 Cont.

CONTROLS FOR ONE ENTRY
GENERATION OF Xfv AND Ufv

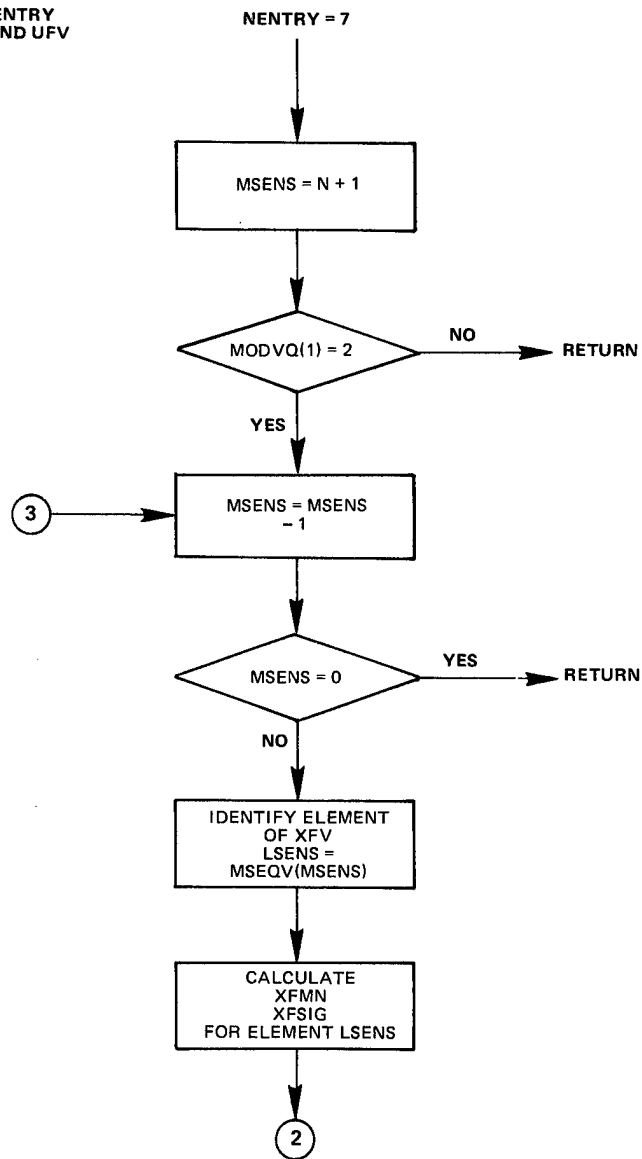


Fig. 4.7.7 Cont.

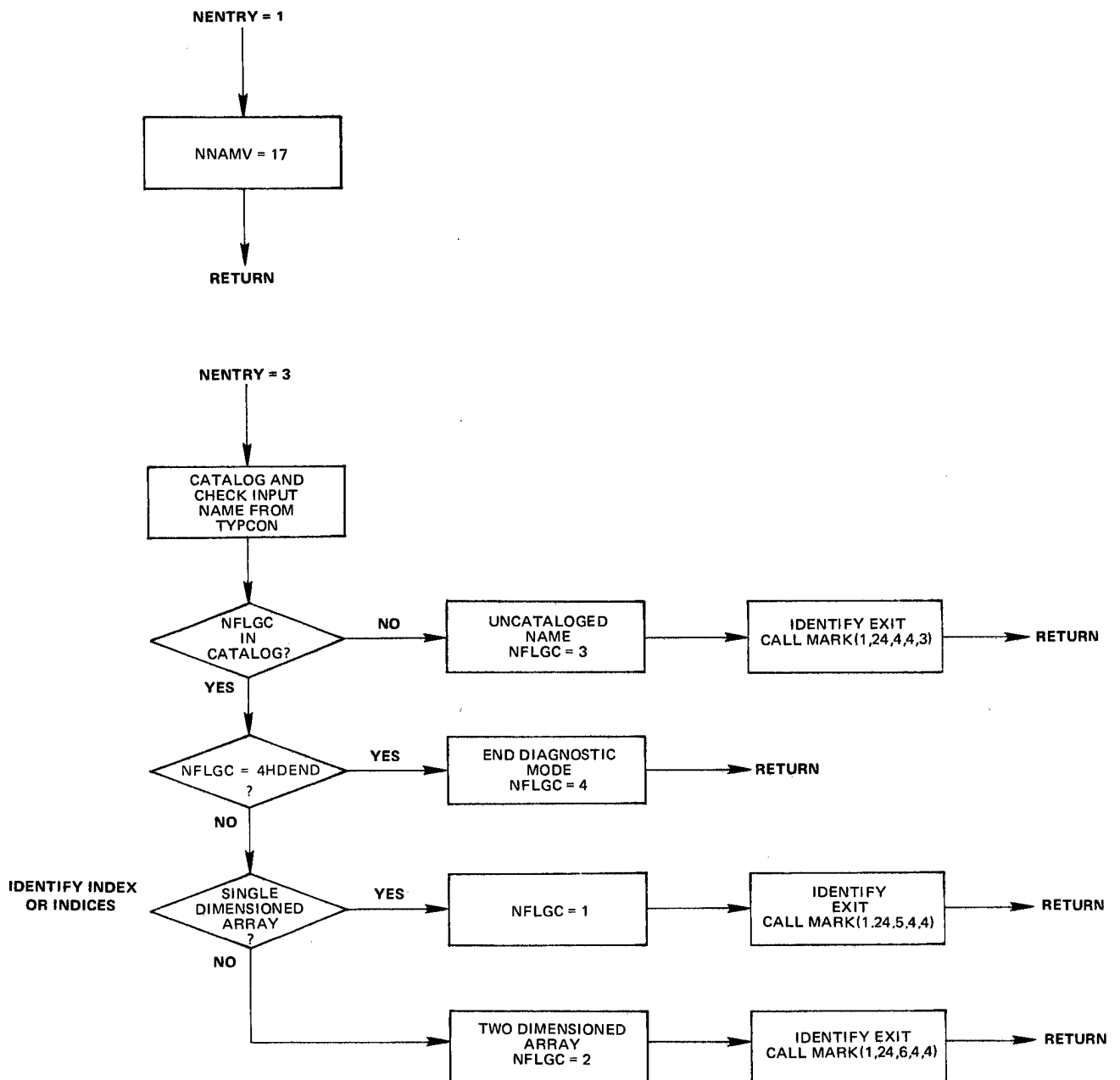


Fig. 4.7.8 MAINB flow diagram.

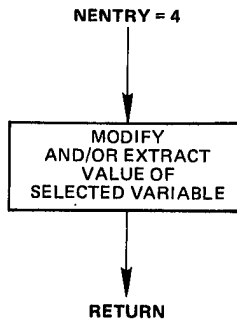


Fig. 4.7.8 Cont.

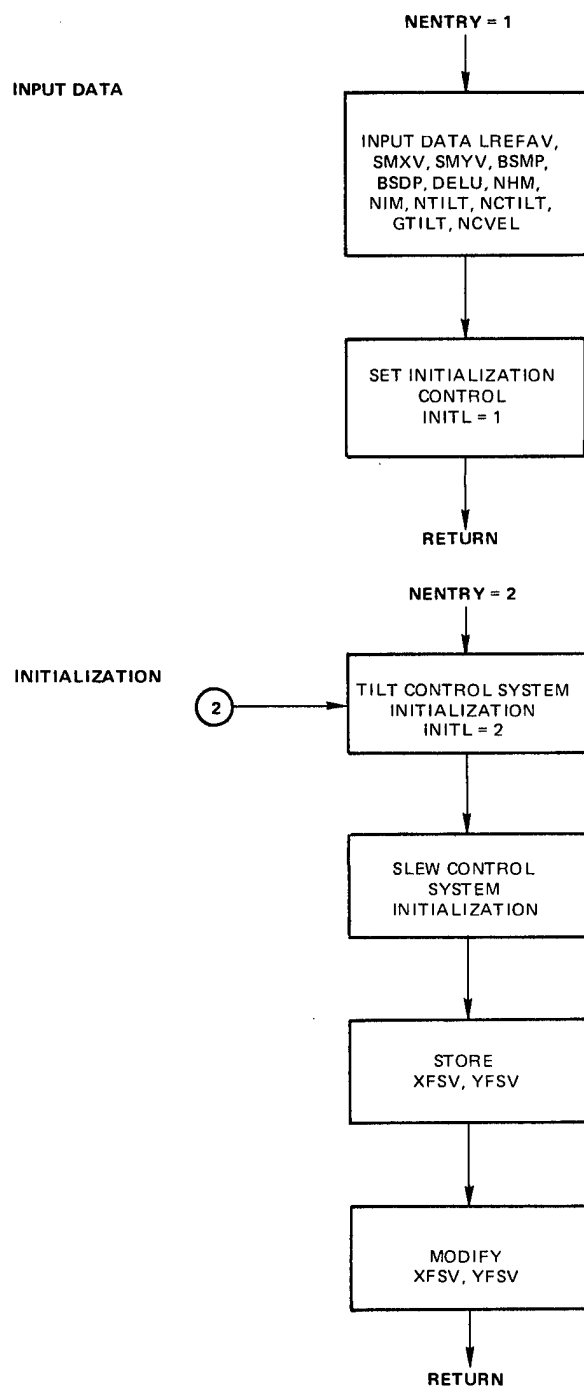


Fig. 4.7.9 MAINC flow diagram.

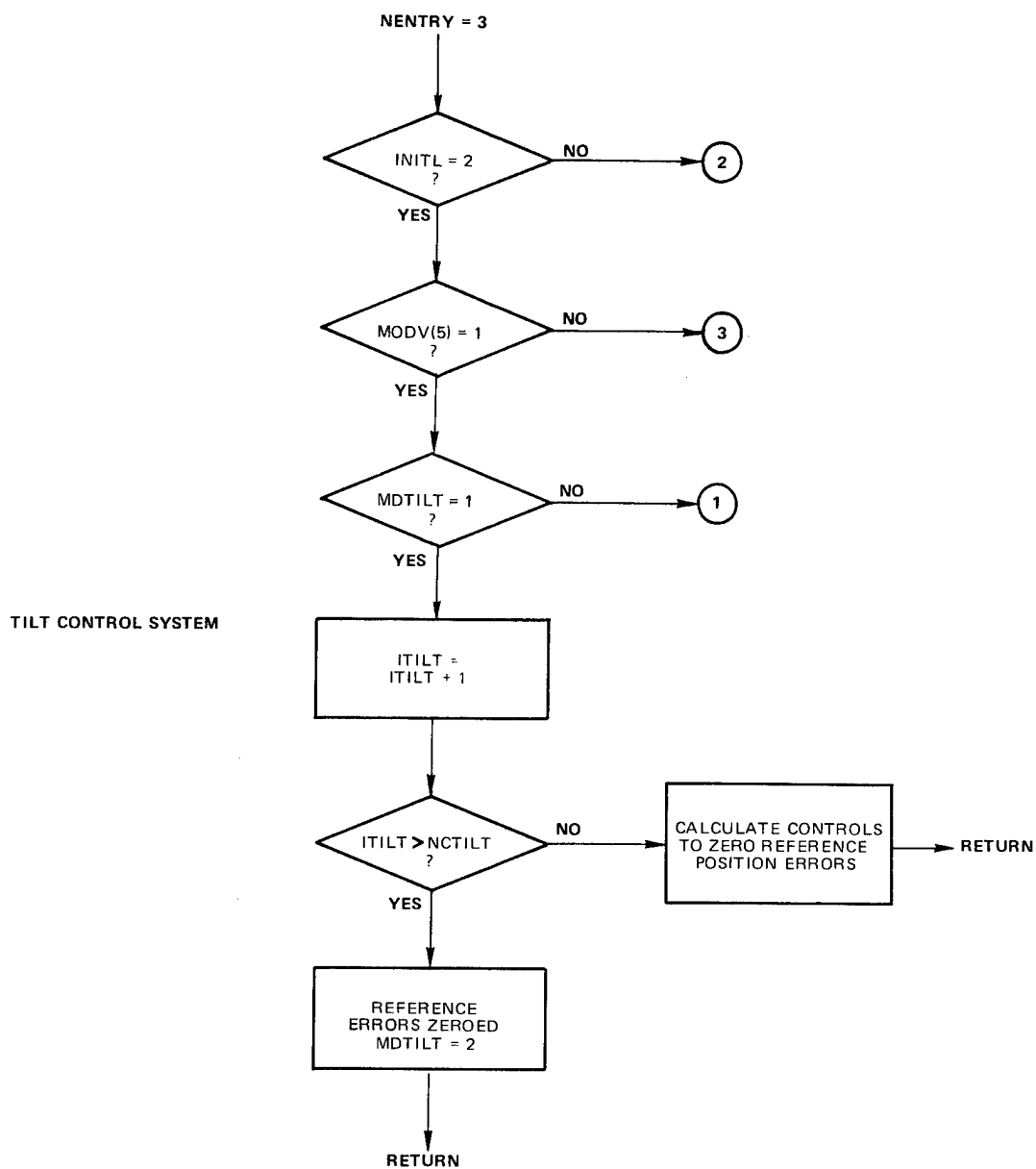


Fig. 4.7.9 Cont.

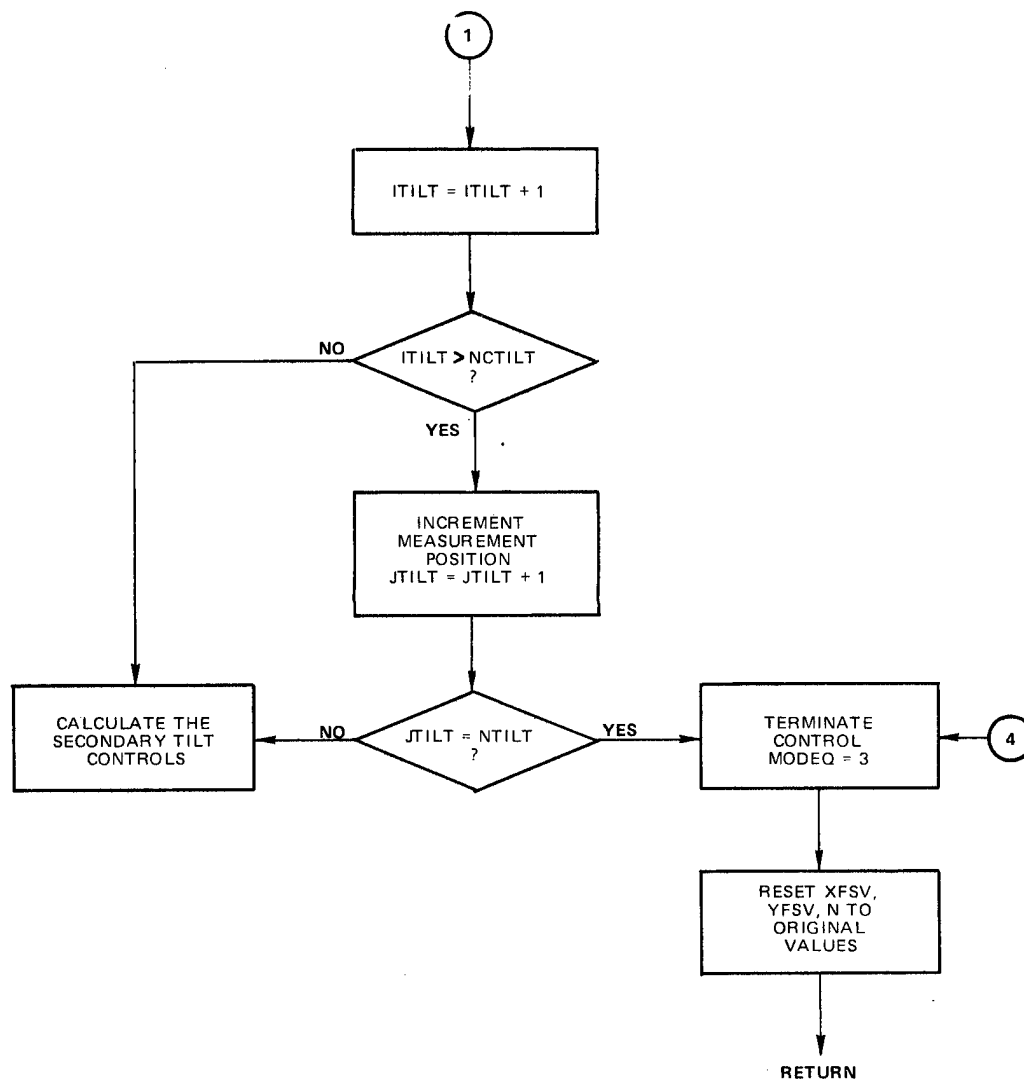


Fig. 4.7.9 Cont.

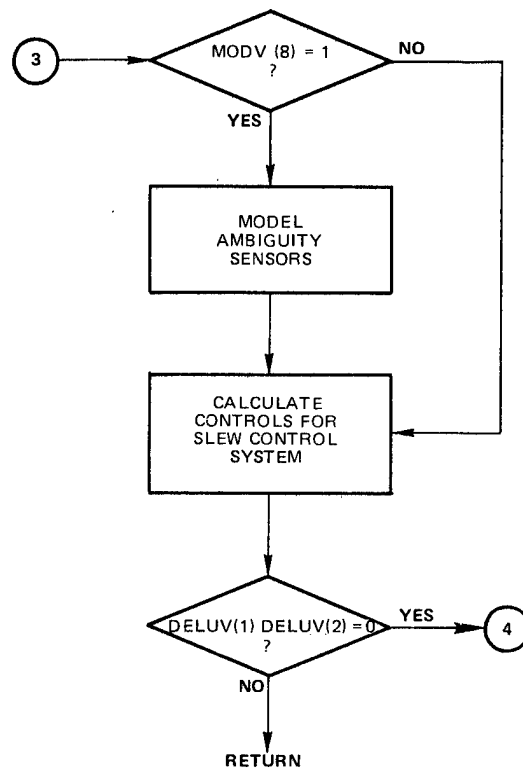


Fig. 4.7.9 Cont.

INPUT DATA

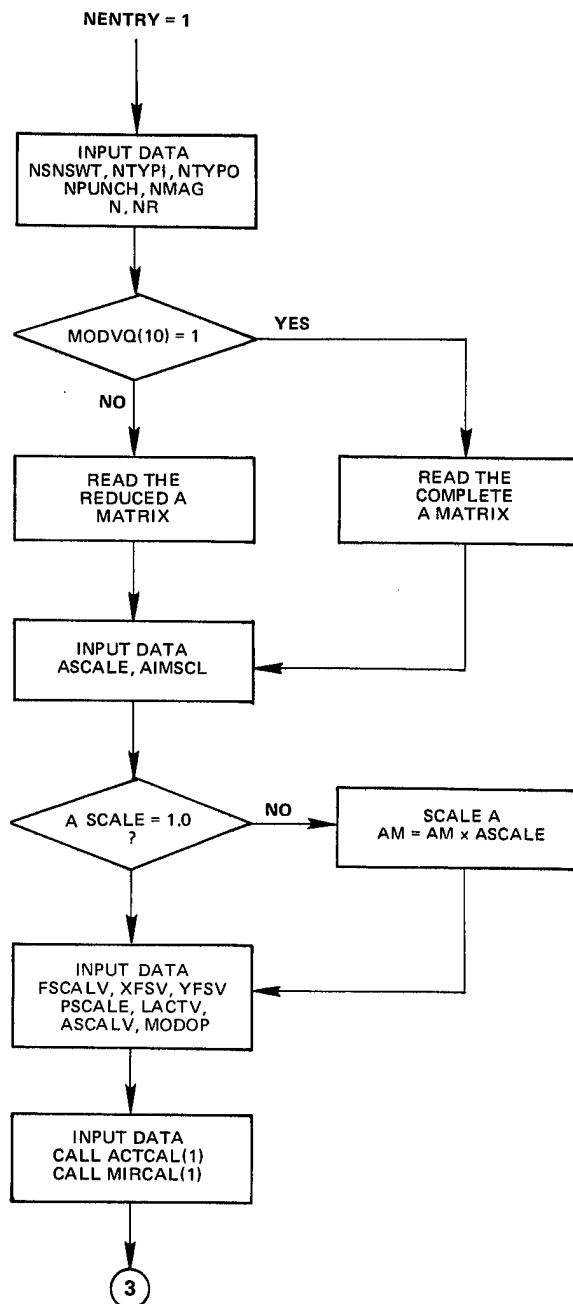


Fig. 4.7.10 MFCS flow diagram.

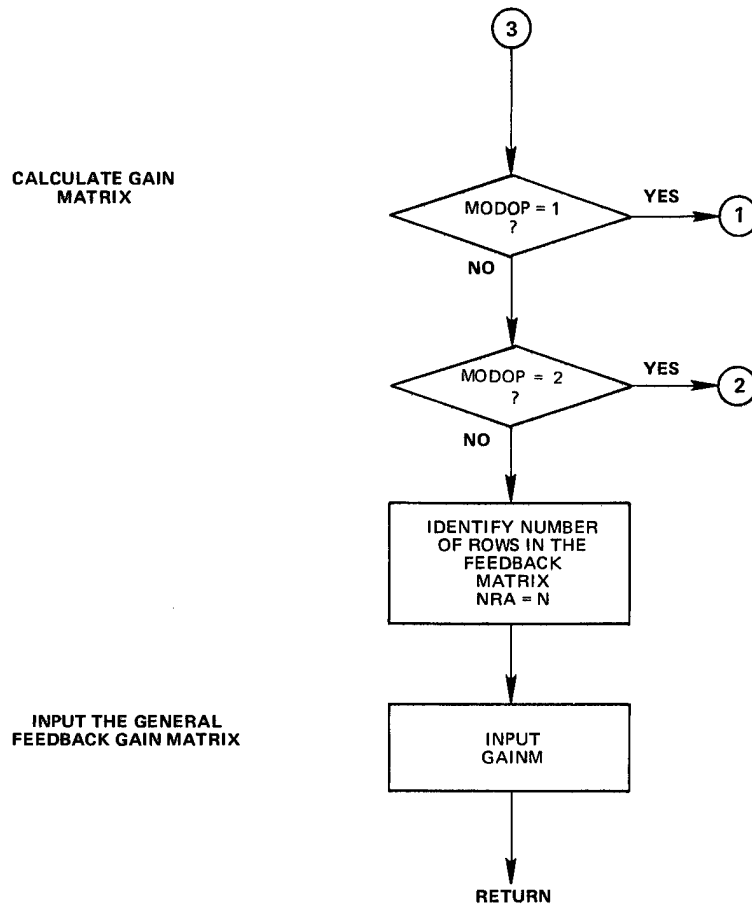


Fig. 4.7.10 Cont.

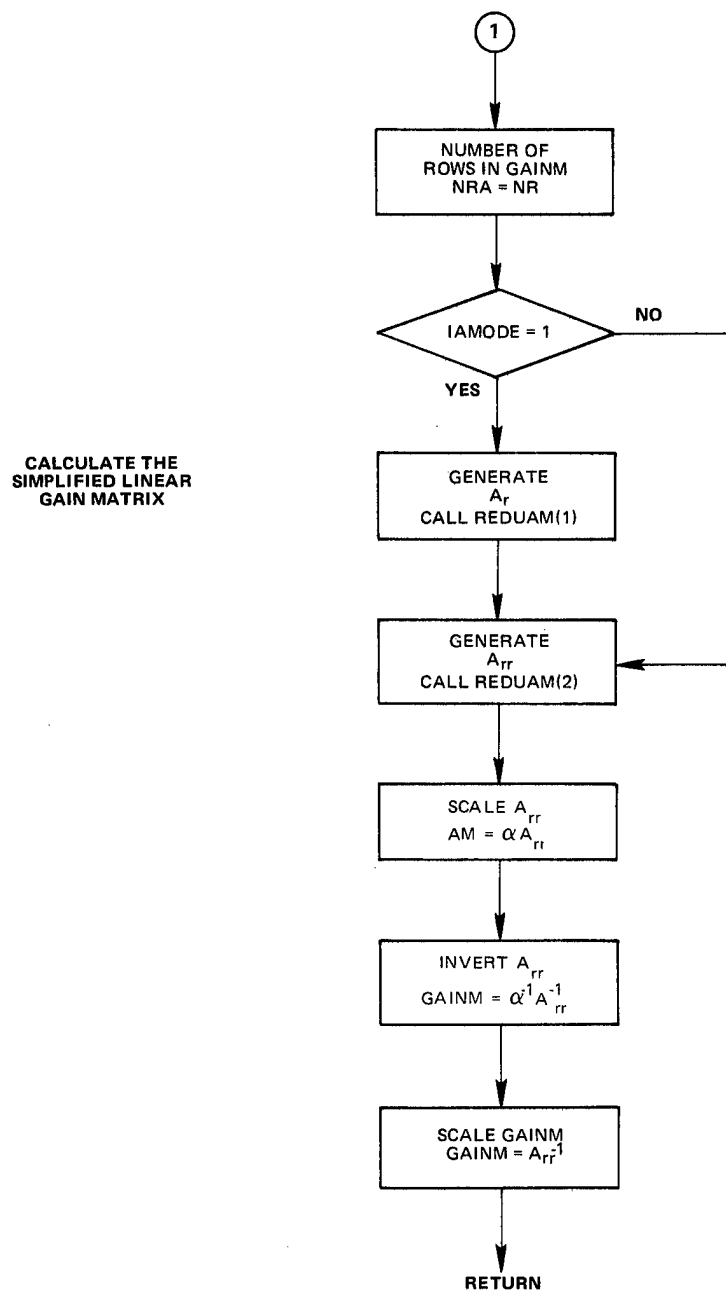


Fig. 4.7.10 Cont.

CALCULATE THE LINEAR
OPTIMAL GAIN MATRIX

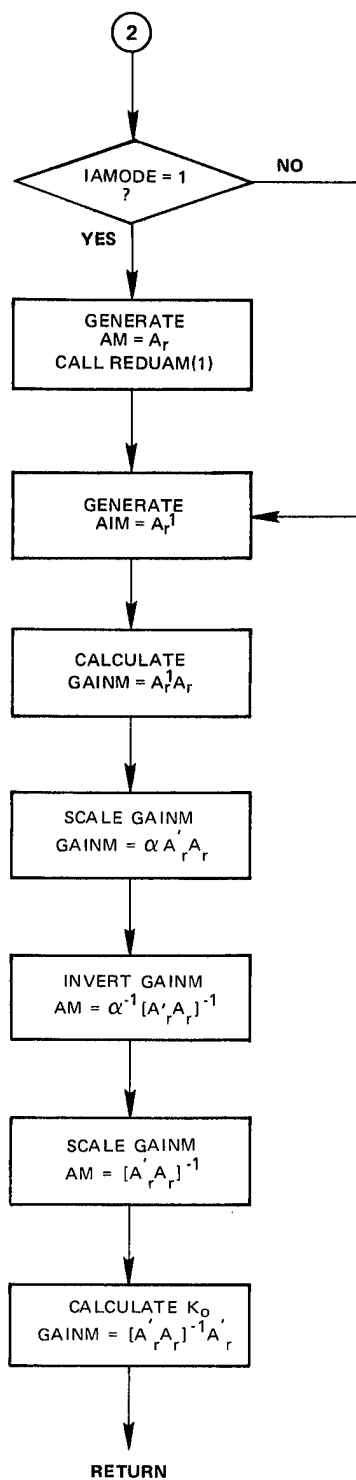


Fig. 4.7.10 Cont.

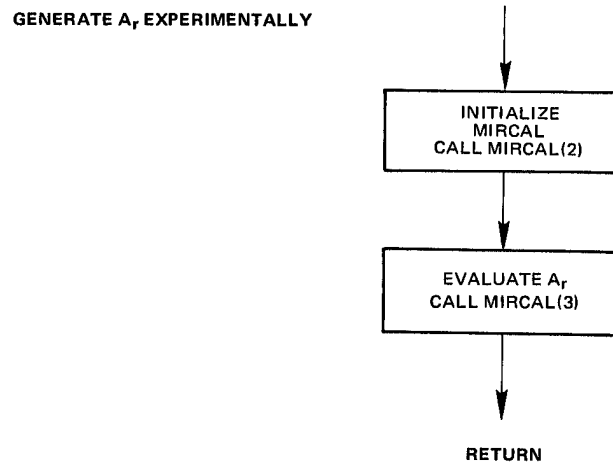
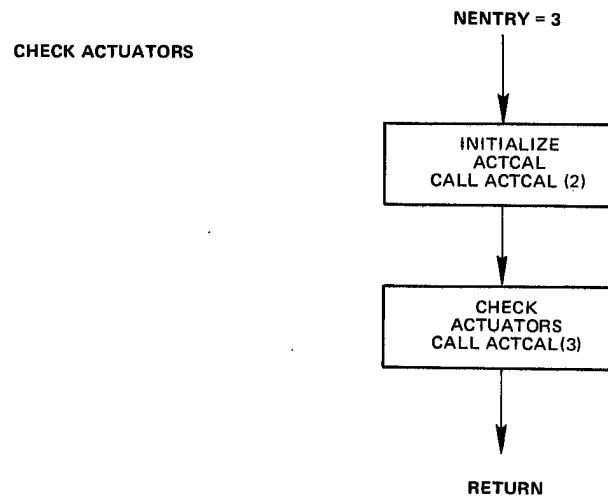


Fig. 4.7.10 Cont.

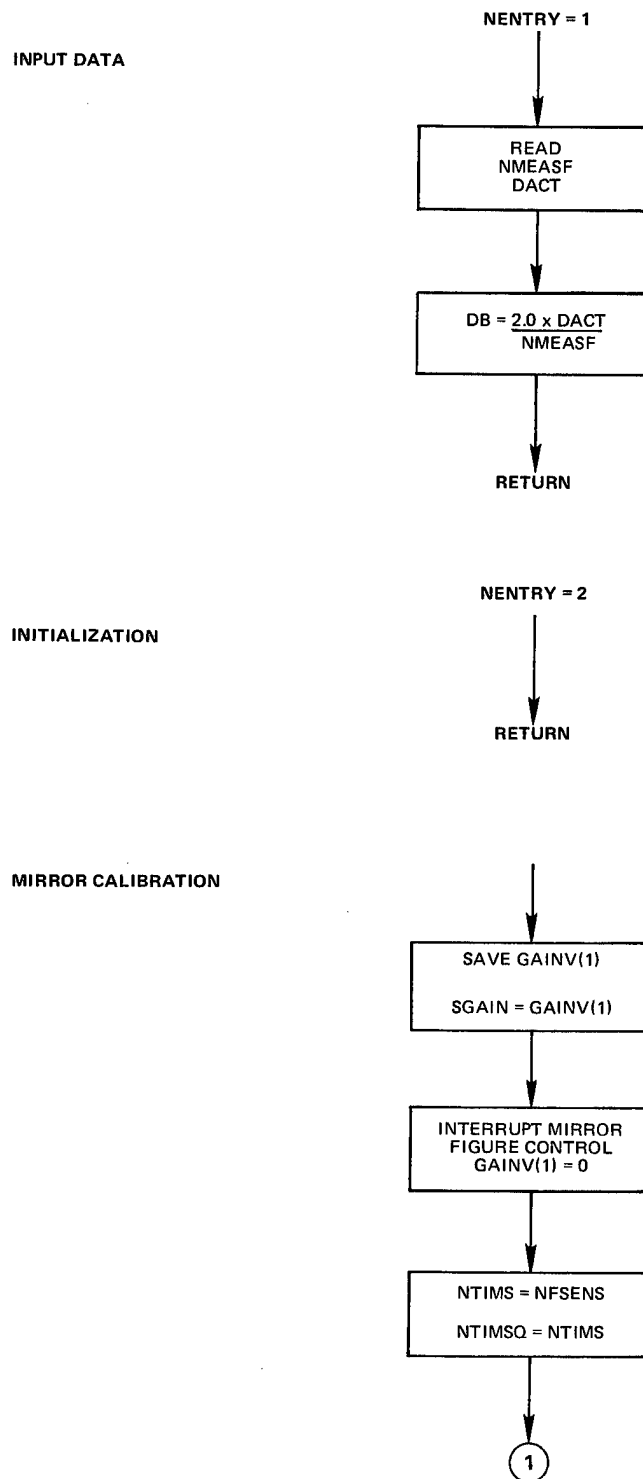


Fig. 4.7.11 MIRCAL flow diagram.

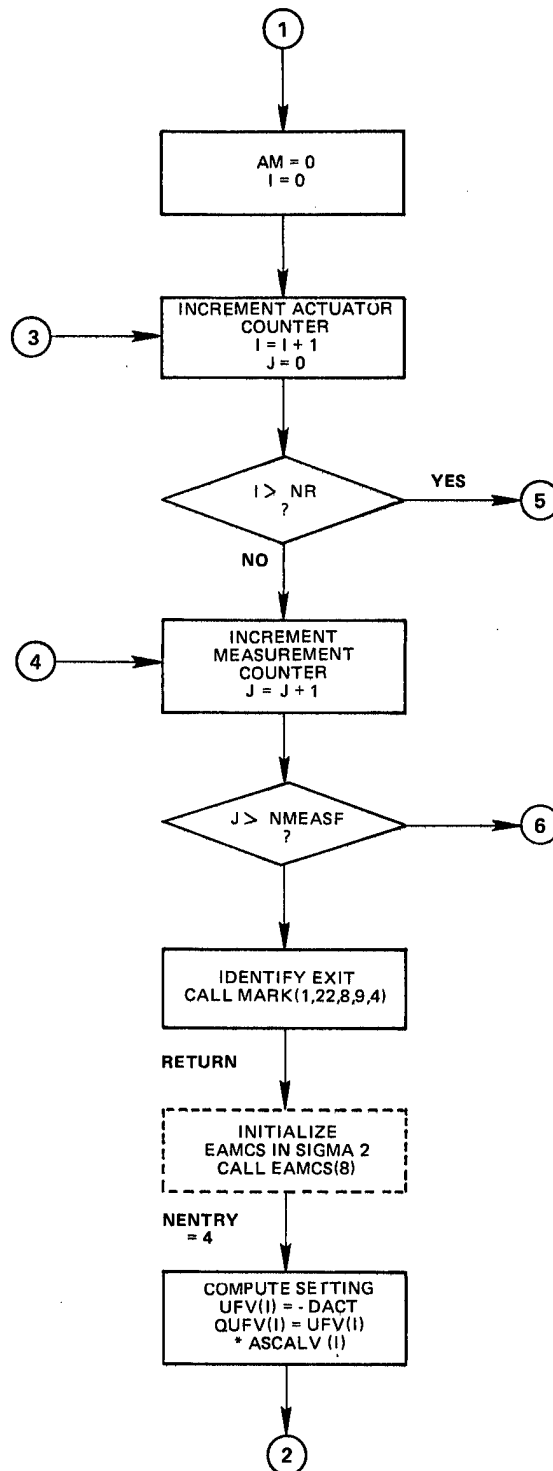


Fig. 4.7.11 Cont.

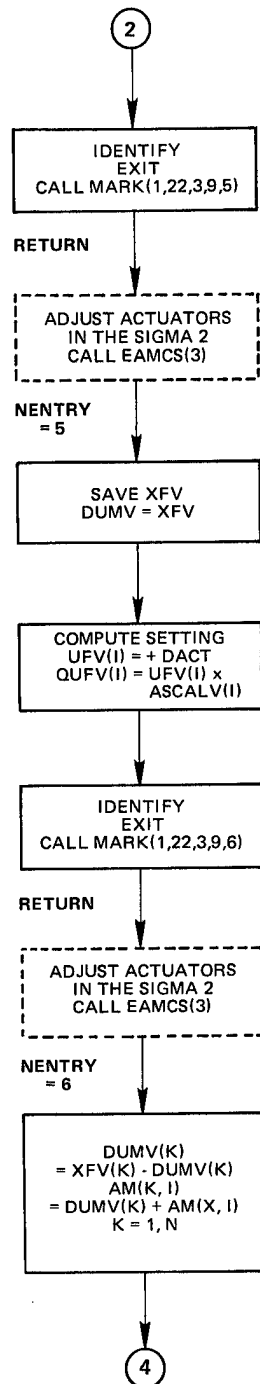
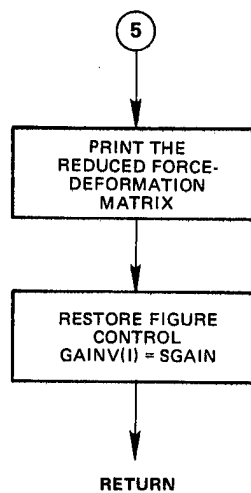


Fig. 4.7.11 Cont.

OUTPUT PRINT



AVERAGE MEASUREMENTS

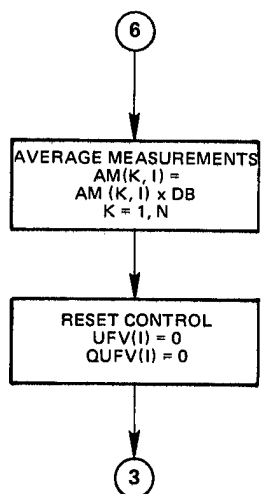


Fig. 4.7.11 Cont.

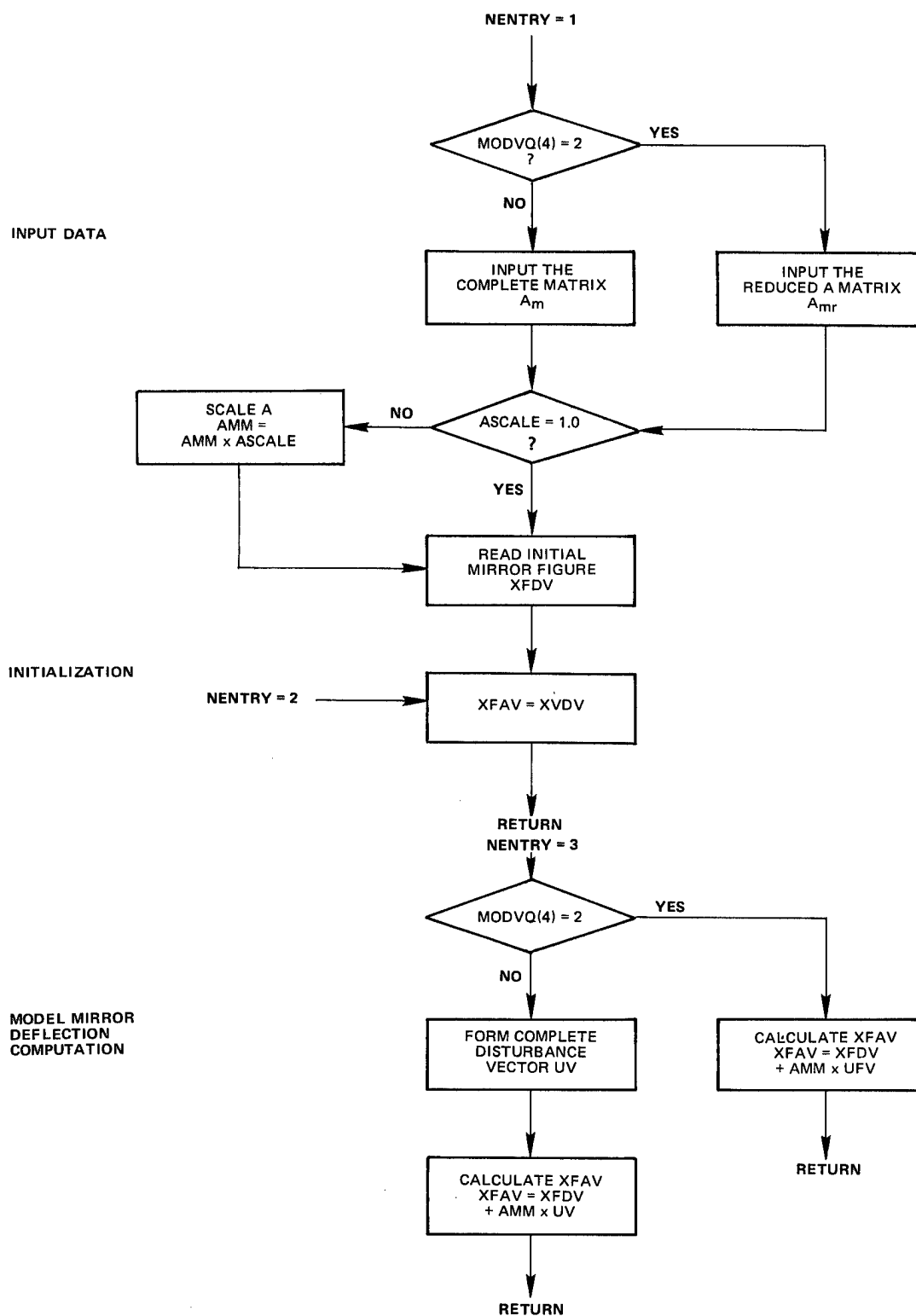


Fig. 4.7.12 MIRMDL flow diagram.

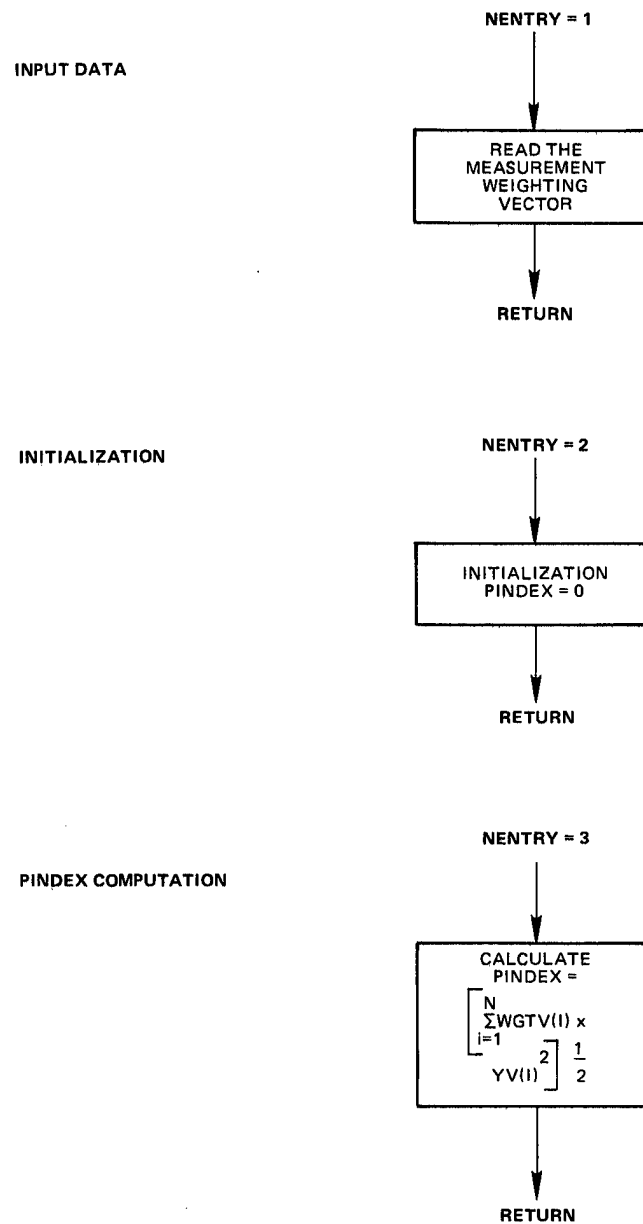


Fig. 4.7.13 PINDX flow diagram.

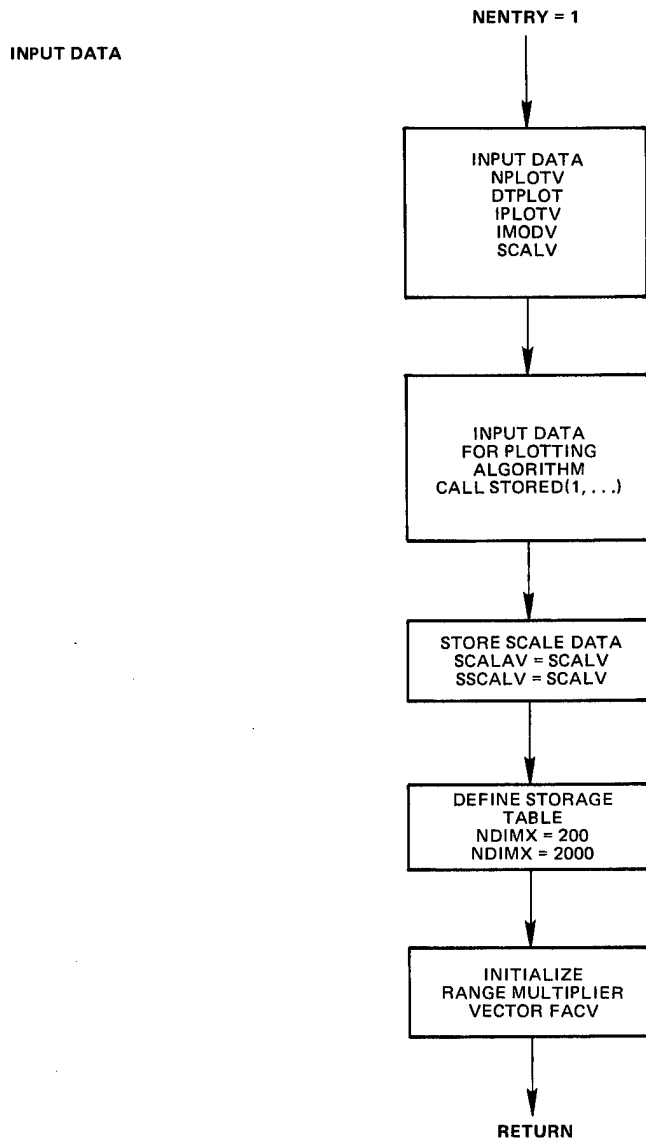
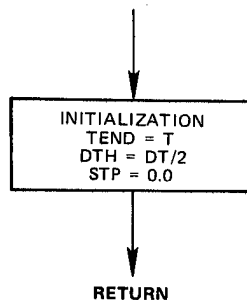


Fig. 4.7.14 PLRT flow diagram.

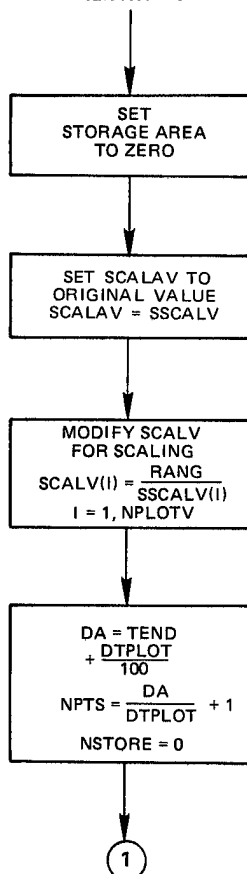
INITIALIZATION

NENTRY = 2



START OF PLOT RUN

NENTRY = 3

DEFINE AND PROTECT THE
DATA SET

SIZE X STORAGE

Fig. 4.7.14 Cont.

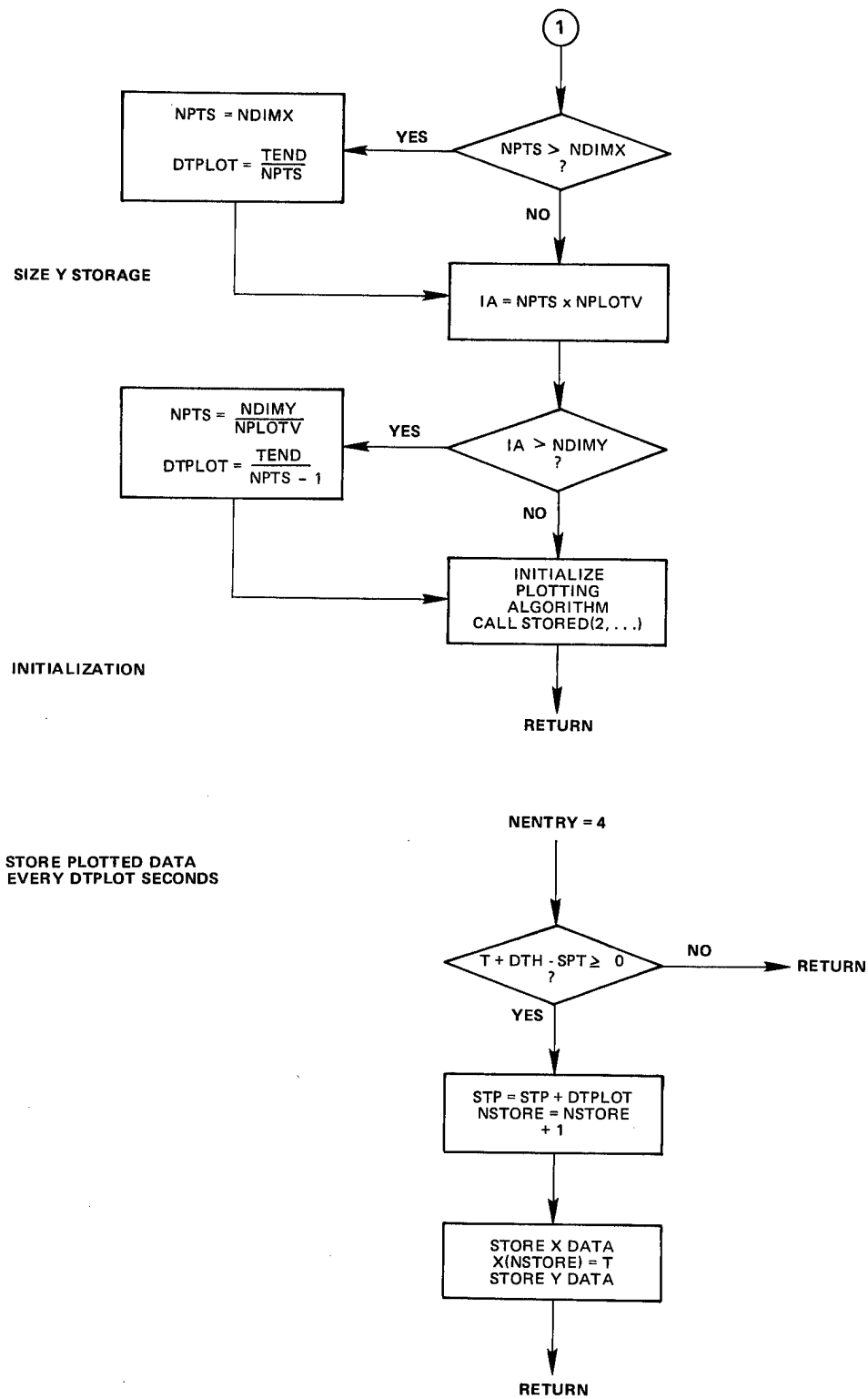


Fig. 4.7.14 Cont.

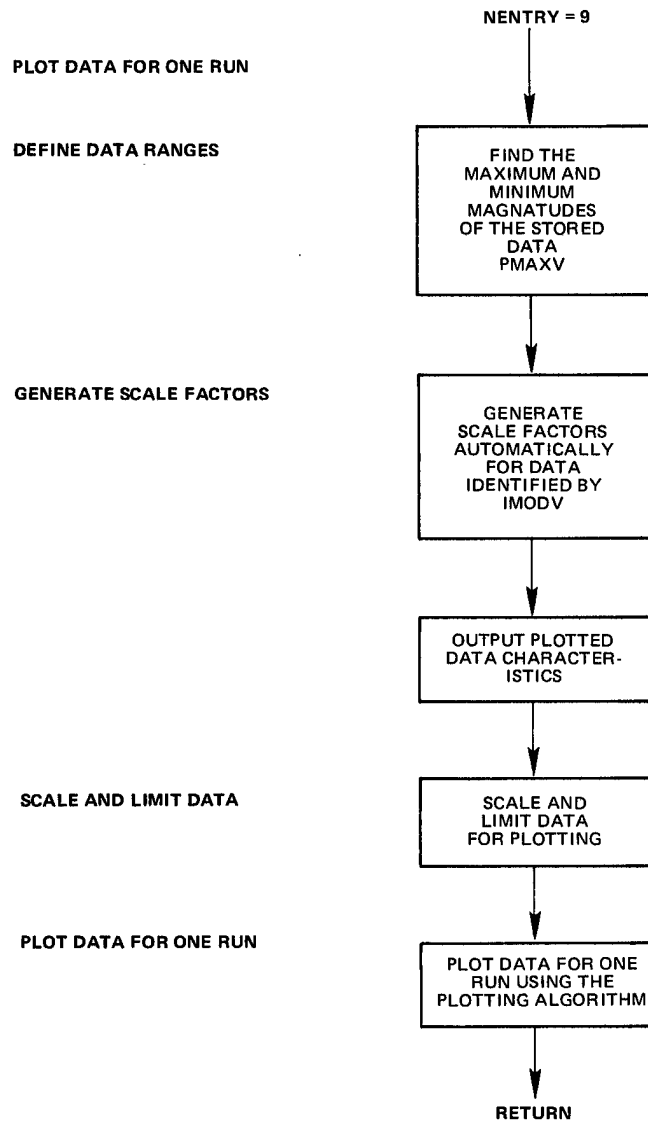


Fig. 4.7.14 Cont.

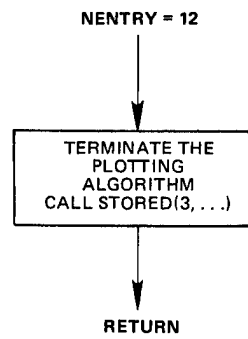


Fig. 4.7.14 Cont.

INPUT DATA

NENTRY = 1

READ
DT, TPRNT, TEND
DTNOIS, NSSRUN

RETURN

INITIALIZATION

NENTRY = 2

T = 0 ST = 0
STN = 0 STP = 0
DTH = DT/2
NTIMSQ = 1

INITIALIZE
FSMDL, MIRMDL
MAINA

TEST
REMOTE CONTROL
IA = 2

YES

MODV(11) = 2
?

NO

IA = 8

IDENTIFY EXIT
CALL
MARK(1,22,IA,7,4)

RETURN

INITIALIZE
EAMCS IN
SIGMA 2
CALL EAMCS(8)

NENTRY
= 4

INITIALIZE
PLOTING ROUTINES
CALL PLRT(2, ...)

RETURN

Fig. 4.7.15 RESPON flow diagram.

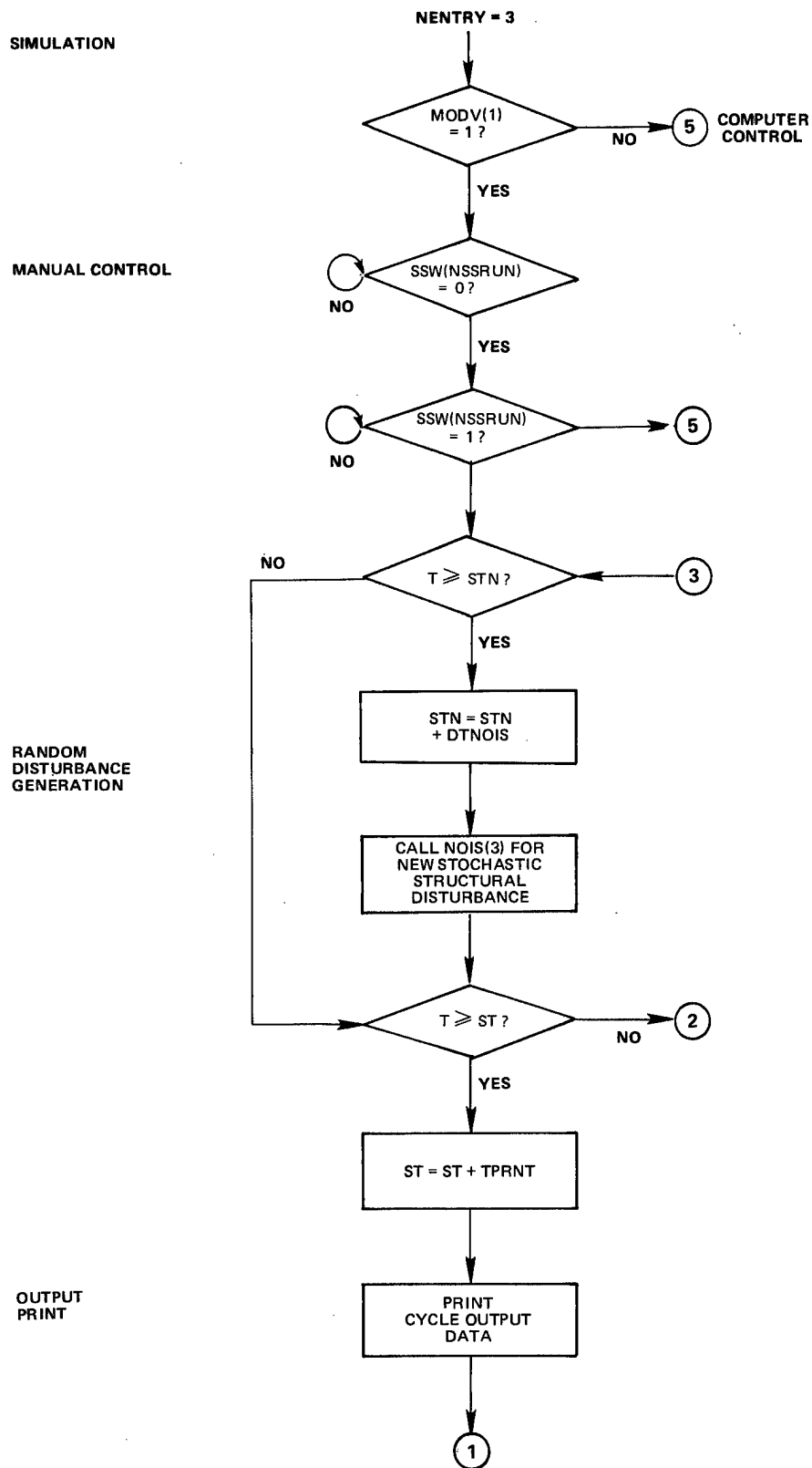


Fig. 4.7.15 Cont.

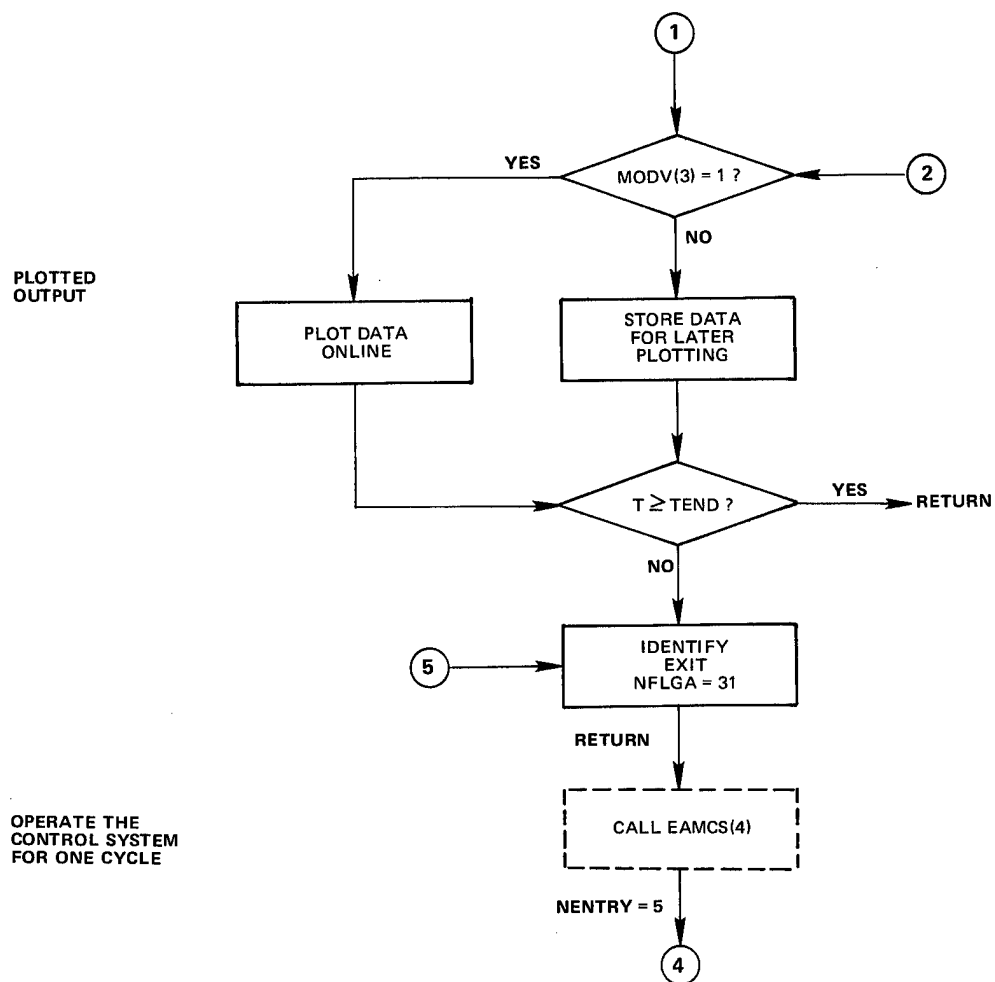


Fig. 4.7.15 Cont.

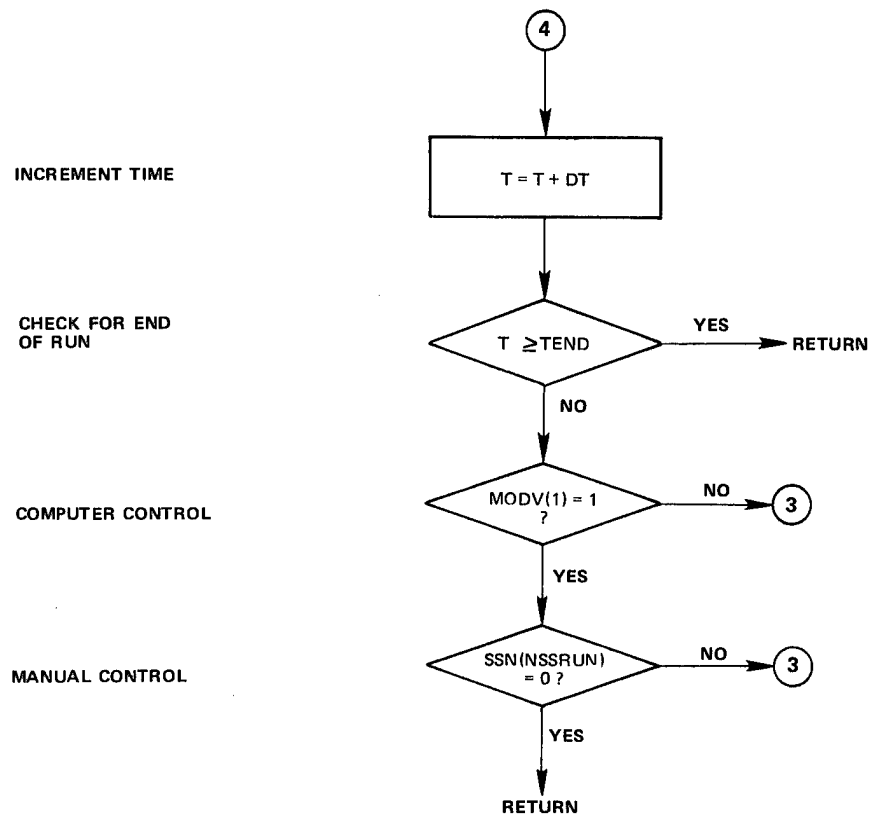


Fig. 4.7.15 Cont.

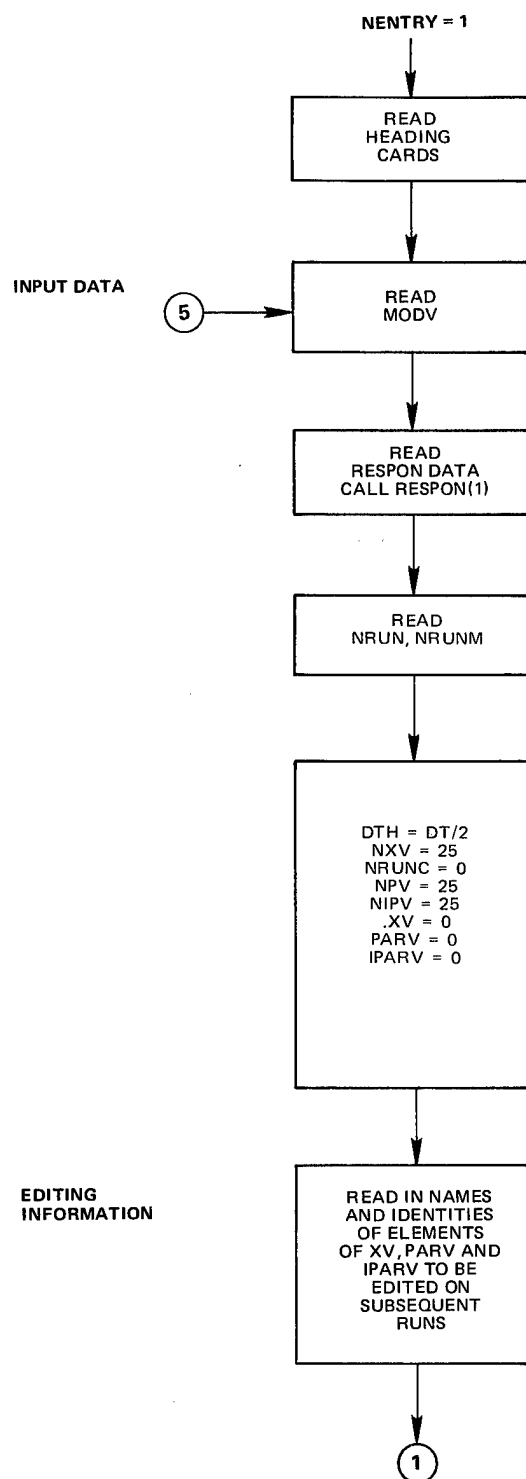


Fig. 4.7.16 SIMSYS flow diagram.

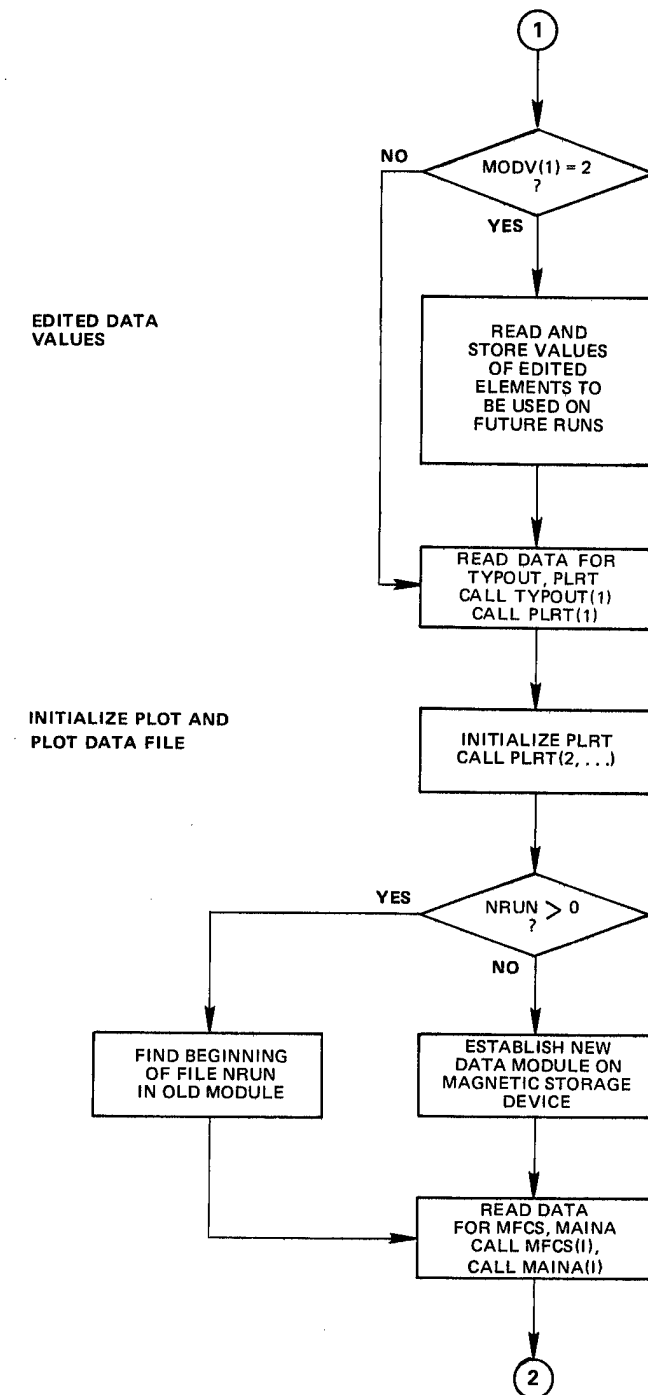


Fig. 4.7.16 Cont.

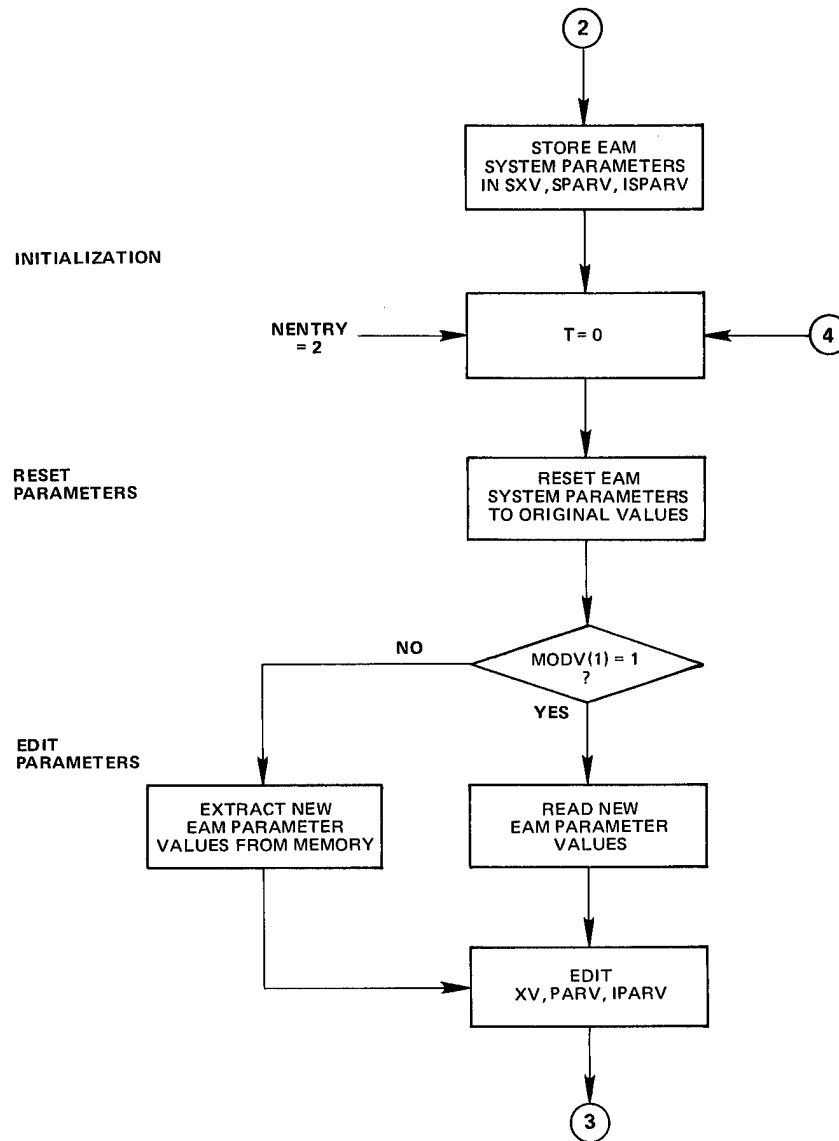


Fig. 4.7.16 Cont.

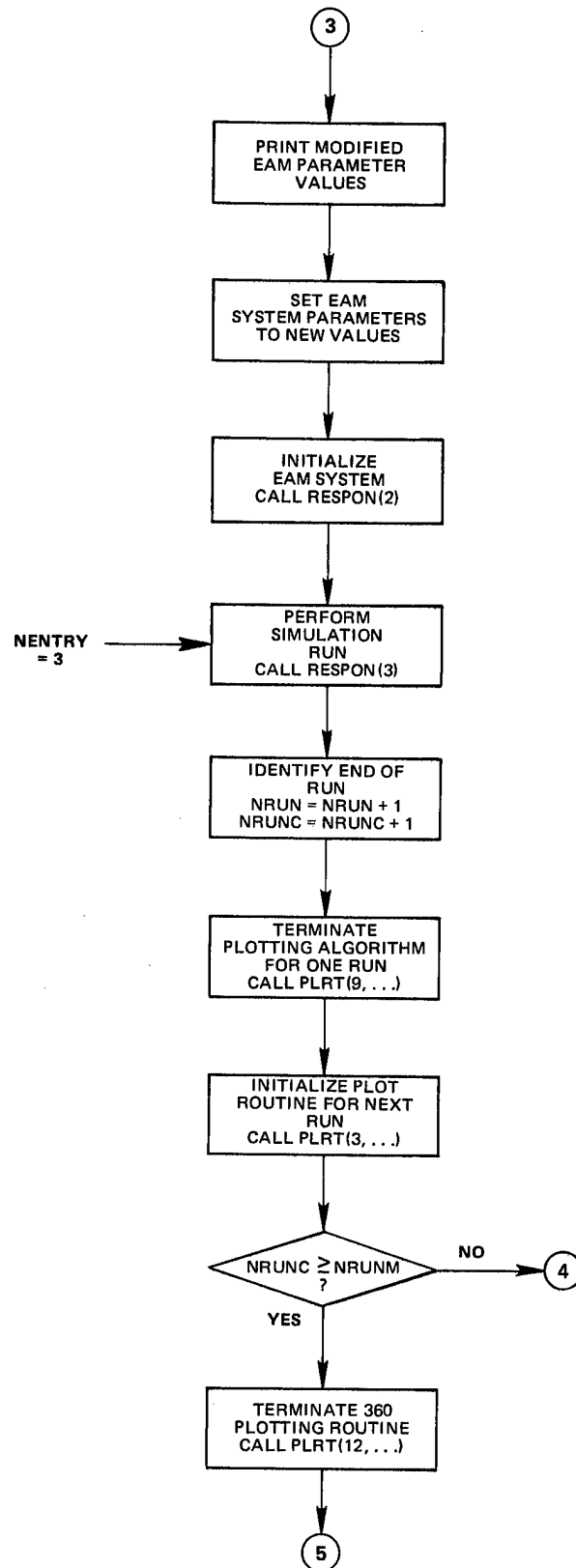


Fig. 4.7.16 Cont.

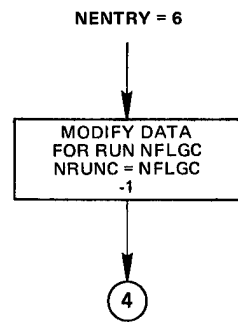


Fig. 4.7.16 Cont.

INPUT DATA

NENTRY = 1

READ
RANG, WIDTH
SPAC

TRANSFER DATA
TO 360 ROUTINE
CALL NEWPLT (.....)

INITIALIZE PLOT
VARIABLES
XBEGIN = 0
XEND = 0

RETURN

INITIALIZATION

NENTRY = 2

COMPUTE
PLOT
PARAMETERS

RETURN

NENTRY = 3

CONVERT
X VECTOR TO
INCHES

DEFINE
NEW REFERENCE
POINT

1

Fig. 4.7.17 STORED flow diagram.

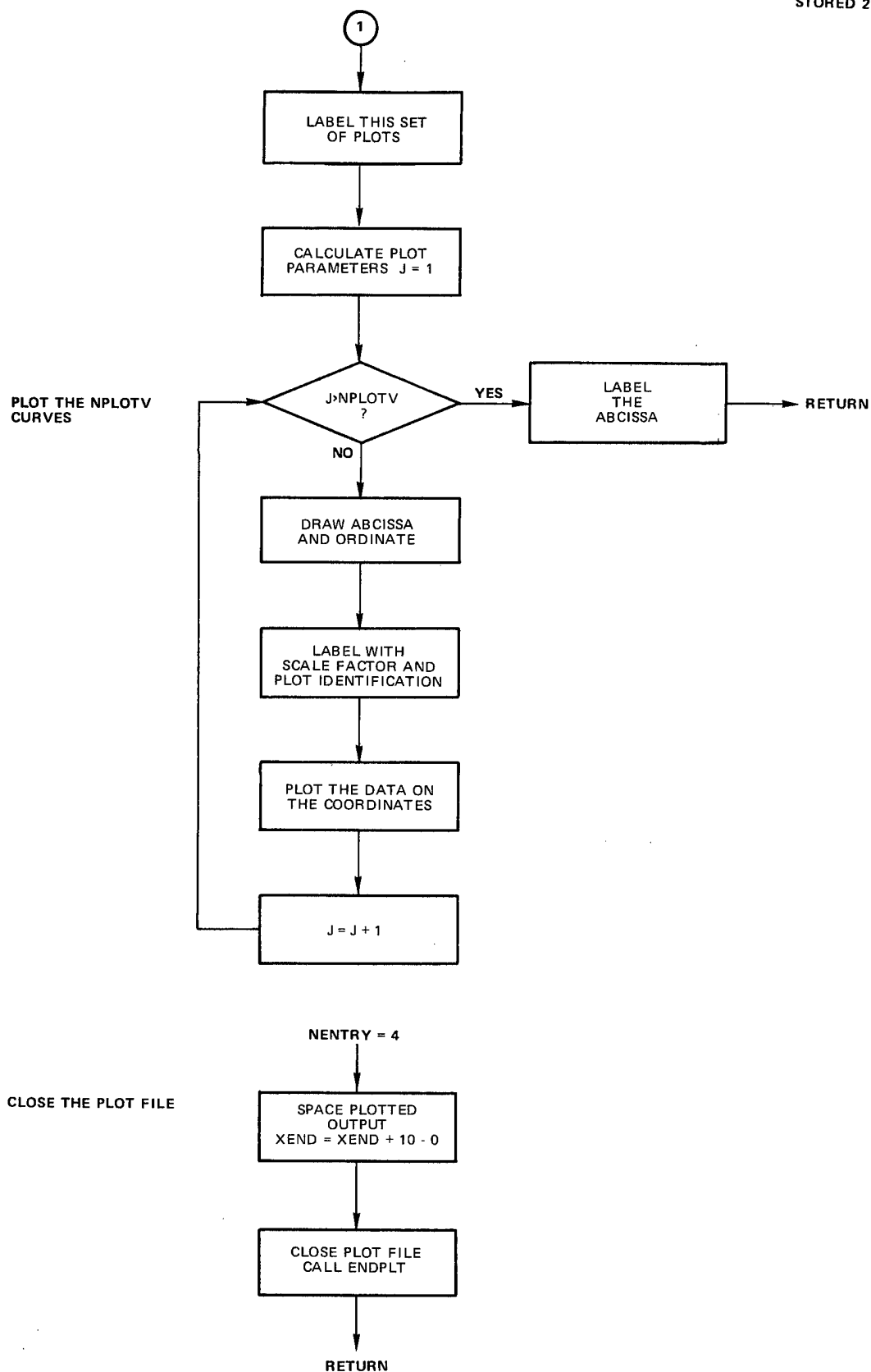


Fig. 4.7.17 Cont.

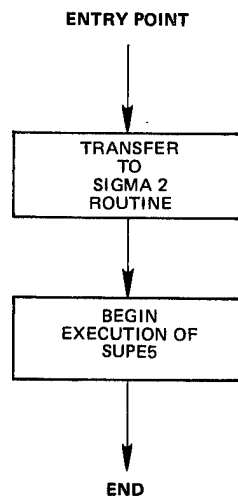


Fig. 4.7.18 SUPE2 flow diagram.

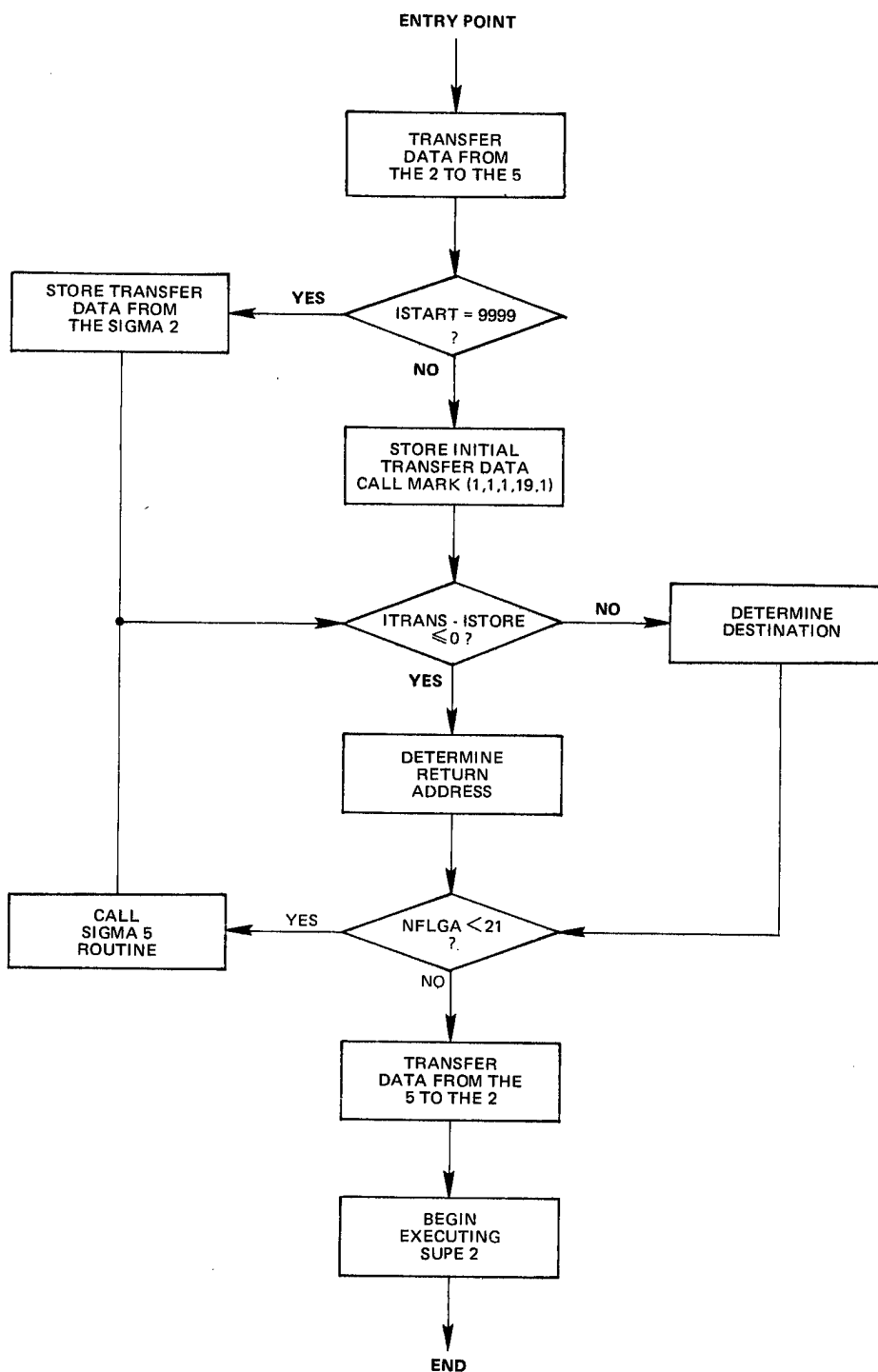
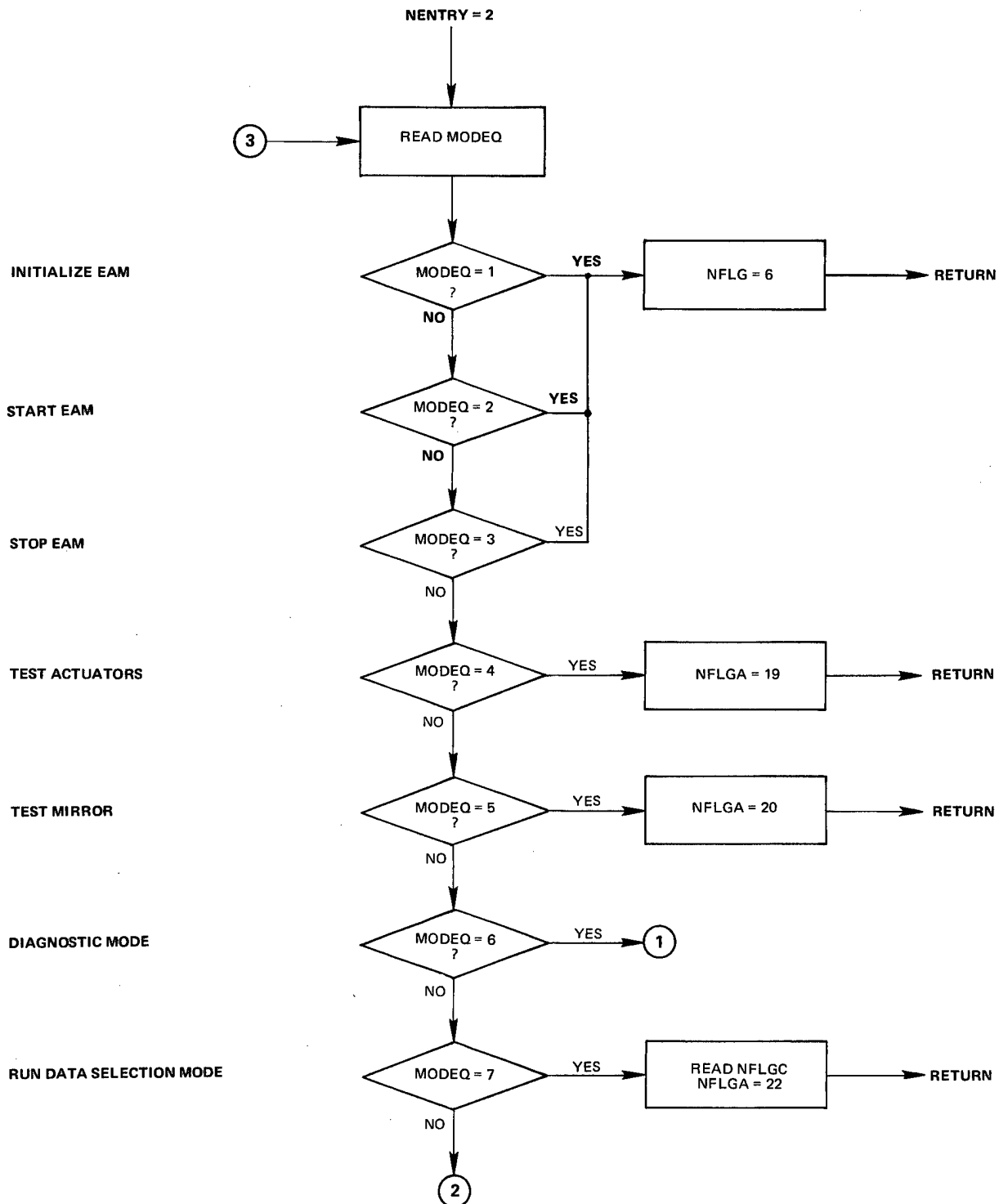


Fig. 4.7.19 SUPE5 flow diagram.



TYPCON 1/3

Fig. 4.7.20 TYPCON flow diagram.

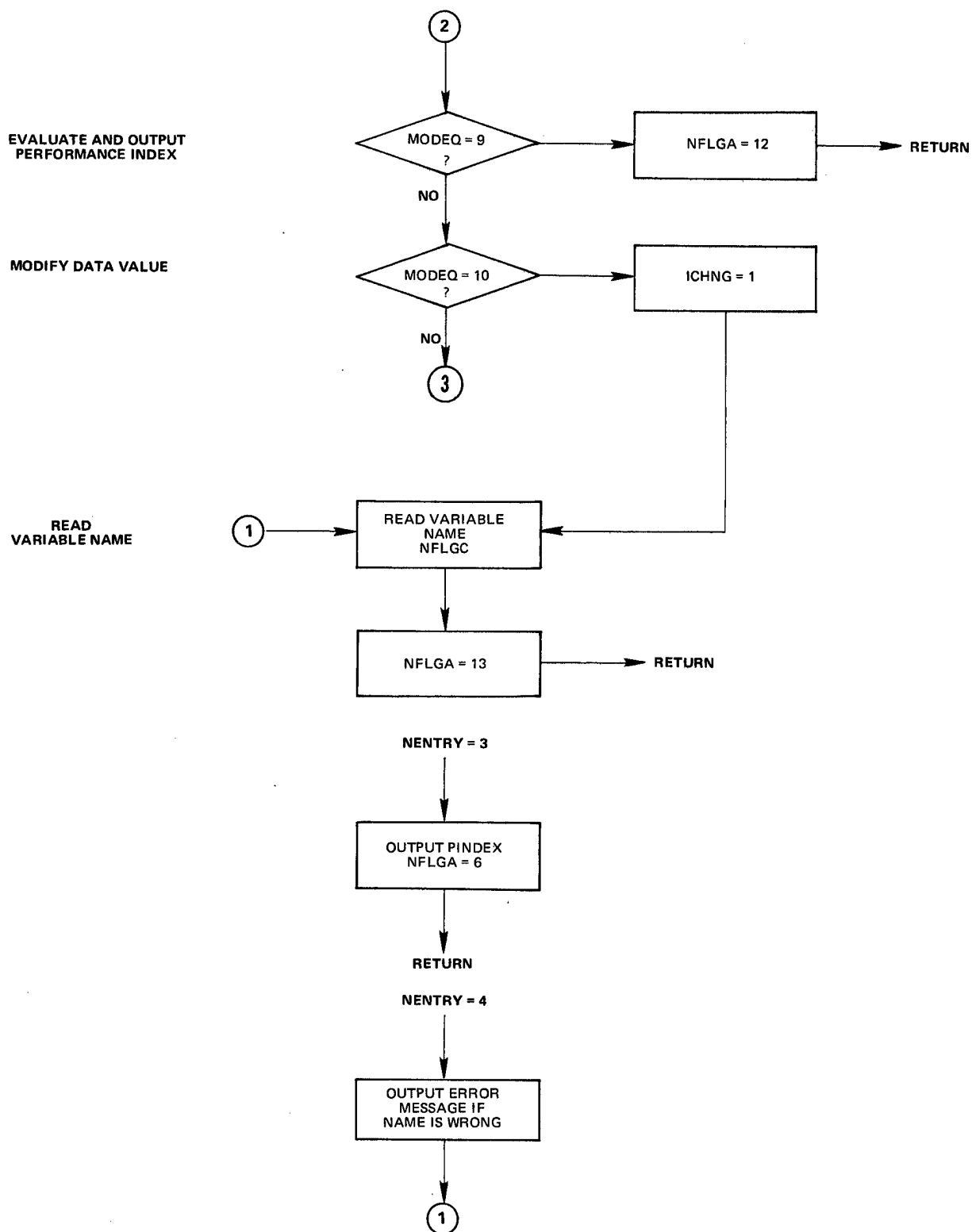


Fig. 4.7.20 Cont.

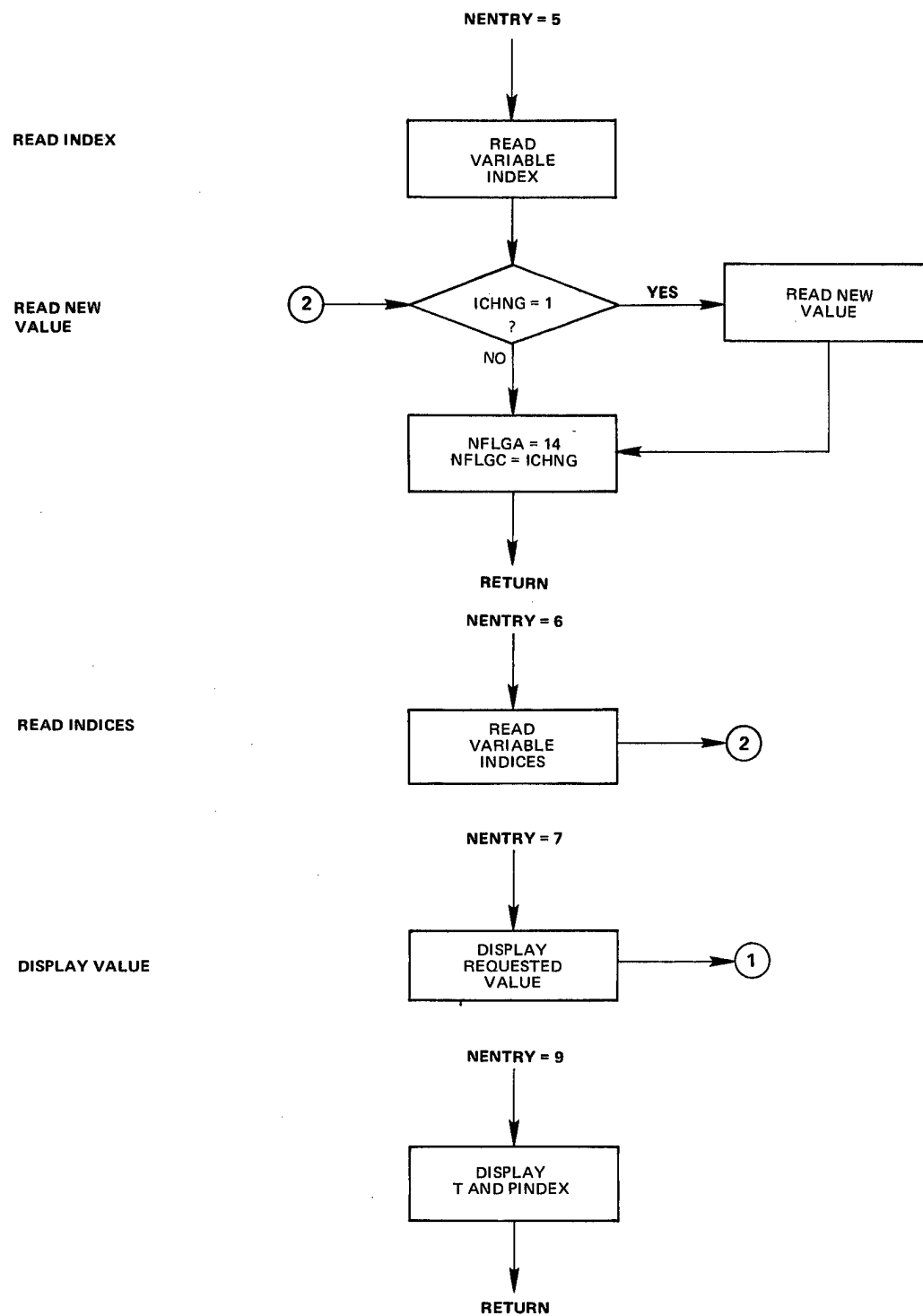


Fig. 4.7.20 Cont.

4.7.20 TYPCON: Remote Terminal Control Software

TYPCON provides the software required to transfer information to and from the remote terminal.

CHAPTER 5

EXPERIMENTAL ACTIVE MIRROR INPUT DATA

5.1 Introduction

Data for the EAM software is read in card format by the Sigma 5 card reader. This procedure was adopted to minimize the use of the rather limited input-output capability of the Sigma 2 computer.

Most of the input data operations are performed by subroutine members of the Input-Output Operations Package (IOP) which is described in Appendix B. Utilization of IOP subroutines results in large savings in core memory by reducing compiler generated in line code compared to that produced by READ and WRITE statements.

Considerable memory space is saved and a high level of convenience achieved by combining data heading cards with the data deck. This obviates the need for format statements and simplifies the identification of the data elements in the data deck.

Fixed point data is generally read in a 7I10 Format while floating point data is read in a 7E10.0 Format which automatically identifies the decimal point location. Single dimension arrays are read in transposed form while two-dimensional arrays are read in row by row.

Data is never read in columns 73-80 to allow space for data deck identification letters and card sequence numbers.

5.2 Input Data Deck Description

The following sections provide a sequential description of the input data deck. Heading cards are indicated by their contents, e. g. ,

NHEADING. Headings for numerical data are read in 7(2X,A8) format. Numerical data is indicated by its input format, e.g., (I10), or (E10.0). If the input data does not correspond in name to the heading card, the name is included in the format description, e.g., (N, I10). A listing of the input data is shown in Fig. 5.2.1.

5.2.1 Output Data Heading

The Sigma 5 reads a heading card NHEADING and an integer N which defines the number of heading cards to be read in and printed in A format. The card defining the value of N is followed by the heading cards as indicated in Fig. 5.2.1.

NHEADING
(N, I10)

N Heading cards (18A4)

5.2.2 Operating Modes

The operating modes of the system are read in as a one-dimensional array. The computer reads a heading card followed by the MODV array values followed by a set of cards which contain the mode identification names in 2(18A4) format. The computer reads

MODV
(7I10)

CONTROL EACH RUN MANUALLY
PLOT RESULTS DURING RUN
ELIMINATE MODEL NONLINEARITIES
INITIAL ALIGNMENT
TILT CONTROL
FORCE ACTUATORS
FIGURE SENSOR TEST
USE MODELS OF HARDWARE
SIMULATE CONTROL SYSTEM
READ IN COMPLETE A MATRIX
NORMAL OPERATION
SIGMA 5 CONFIGURATION

CONTROL RUNS AUTOMATICALLY
STORE RESULTS FOR LATER PLOTTING
INCLUDE MODEL NONLINEARITIES
FINAL ALIGNMENT
SLEW CONTROL
POSITION ACTUATORS
FIGURE CONTROL SYSTEM TEST
USE HARDWARE COMPONENTS
OPERATE CONTROL SYSTEM
READ IN REDUCED A MATRIX
TYPCON TEST MODE
SIGMA 5-2 CONFIGURATION

NHEADING 44

```

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

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

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

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

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

```

DEVELOPED BY THE CHARLES STARK DRAPER LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY, CAMBRIDGE, MASSACHUSETTS 02139

MODV

2	2	1
1	1	1

CONTROL EACH RUN MANUALLY
PLOT RESULTS DURING RUN
ELIMINATE MODEL NONLINEARITIES
INITIAL ALIGNMENT
TILT CONTROL
FORCE ACTUATORS
FIGURE SENSOR TEST
USE MODELS OF HARDWARE
SIMULATE CONTROL SYSTEM
READ IN THE COMPLETE A MATRIX
NORMAL OPERATION

2	1	1	2
1	1		

CONTROL RUNS AUTOMATICALLY
STORE RESULTS FOR LATER PLOTTING
INCLUDE MODEL NONLINEARITIES
FINAL ALIGNMENT
SLEW CONTROL
POSITION ACTUATORS
FIGURE CONTROL SYSTEM TEST
USE HARDWARE COMPONENTS
OPERATE CONTROL SYSTEM
READ IN THE REDUCED A MATRIX
TYPCON TEST MODE

1/5

Fig. 5.2.1 Input data deck.

SIGMA 5 CONFIGURATION

SIGMA 5-2 CONFIGURATION

DT	TPRNT	TEND	DTNOIS			
0.100	6.50	200.0	10000.0			
NSSRUN						
1						
NPUN	NRUNM					
1104710	1					
NCXV	NCPV	NICPV				
1	1	1				
X1						
36						
GAIN	DT	TEND	FSTFLT			
1	2	3	4			
IP1						
11						
CXM						
CPM						
-0.050						
CIPM						
NTYPE						
6						
NPLOTV						
24						
DTPLDT						
6.50						
IPLOTV						
1	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26				
IMODV						
2	2	2	2	2	2	2
2	2	2	2	2	2	2
2	2	2	2	2	2	2
2	2	2				
SCALV						
1.0	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0				
NSNSWT	NTYPI	NTYPO	NPUNCH	NMAG		
1	5	6	6	6		
N	NR					
16	7					
AM						
12.02290	10.12680	10.75020	5.52466	-2.41734	3.20453	5.52466
10.12680	10.75020	6.38214	6.56174	-2.15830	-2.15830	6.56174
6.38214	9.83788					
10.12680	86.33018	38.73888	10.12680	-21.20290	-6.26523	-2.41734
-21.20290	-6.26523	53.41199	53.41199	-22.75899	-8.63721	-8.63721
-22.75899	7.31642					
10.75020	38.73888	26.41588	10.75020	-6.26523	2.50271	3.20453
-6.26523	2.50271	25.24458	25.24458	-8.18677	-2.07520	-2.07520
-8.18677	11.23680					
5.52466	10.12680	10.75020	12.02290	10.12680	10.75020	5.52466
-2.41734	3.20453	6.56174	6.38214	6.38214	6.56174	-2.1583
-2.41734	9.83788					
-2.41734	-21.20290	-6.26523	10.12680	86.33018	38.73888	10.12680

2/5

Fig. 5.2.1 Cont.

-21.20290	-6.26523	-8.63721	-22.75899	53.41199	53.41199	-22.75899
-8.63721	7.31642					
3.20453	-6.26523	2.50271	10.75020	38.73888	26.41588	10.75020
-6.26523	2.50271	-2.07520	-8.18677	25.24458	25.24458	-8.18677
-2.07520	11.23680					
5.52466	-2.41734	3.20453	5.52466	10.12680	10.75020	12.02290
10.12680	10.75020	-2.15830	-2.15830	6.56174	6.38214	6.38214
6.56174	9.83788					
10.12680	-21.20290	-6.26523	-2.41734	-21.20290	-6.26523	10.12680
86.33018	38.73888	-22.75899	-8.63721	-8.63721	-22.75899	53.41199
53.41199	7.31642					
10.75020	-6.26523	2.50271	3.20453	-6.26523	2.50271	10.75020
38.73888	26.41588	-8.18677	-2.07520	-2.07520	-8.18677	25.24458
25.24458	11.23680					
6.38214	53.41199	25.24458	6.56174	-8.63721	-2.07520	-2.15830
-22.75899	-8.18677	47.71928	28.99750	-10.87630	-1.90107	-10.87630
-23.13948	4.33471					
6.56174	53.41199	25.24458	6.38214	-22.75899	-8.18677	-2.15830
-8.63721	-2.07520	23.99750	47.71928	-23.13948	-10.87630	-1.90107
-10.87630	4.33471					
-2.15830	-22.75899	-8.18677	6.38214	53.41199	25.24458	6.56174
-8.63721	-2.07520	-10.87630	-23.13948	47.71928	28.99750	-10.87630
-1.90107	4.33471					
-2.15830	-8.63721	-2.07520	6.56174	53.41199	25.24458	6.38214
-22.75899	-8.18677	-1.90107	-10.87630	28.99750	47.71928	-23.13948
-10.87630	4.33471					
6.56174	-8.63721	-2.07520	-2.15830	-22.75899	-8.18677	6.38214
53.41199	25.24458	-10.87630	-1.90107	-10.87630	-23.13948	47.71928
28.99750	4.33471					
6.38214	-22.75899	-8.18677	-2.15830	-8.63721	-2.07520	6.56174
53.41199	25.24458	-23.13948	-10.87630	-1.90107	-10.87630	28.99750
47.71928	4.33471					
9.83788	7.31642	11.23680	9.83788	7.31642	11.23680	9.83788
7.31642	11.23680	4.33471	4.33471	4.33471	4.33471	4.33471
4.33471	15.99379					
ASCALE	ATMSCL					
0.0931	10.00					
FSCALE						
1.0						
XFSV						

YFSV

PSCALE

1.0

LACTV

0

1

ASCALV

1.0

MODOP

1

NMFASA

1

1

1

1.0

1.0

1.0

1.0

1.0

3/5

Fig. 5.2.1 Cont.

```

1
DACT
0.10
NMFASF
1
DACT
0.10
NTIMSO      NWAIT      NPOS      NMINT      NMFAS      NTYO
1           1           1           3           1           1
DT          DTE      GAINV(1)      QGA      QGH      UFMAX
0.100      0.00001      -0.250      1.0      1.0      20.0
SIGLIM      SLDPMV
1000.0      0.0
MSEQV
2           11           4           12           6           5           13
7           14           8           9           15           1           10
3           15
FSMSIG      FSTFLT
0.0          0.0
IRAND      IPLDT
1           2
TACTV

AMM
12.02290    10.12680    10.75020    5.52466    -2.41734    3.20453    5.52466
10.12680    10.75020    6.38214    6.56174    -2.15830    -2.15830    6.56174
6.38214     9.83788
10.12680    86.33018    38.73888    10.12680    -21.20290    -6.26523    -2.41734
-21.20290    -6.26523    53.41199    53.41199    -22.75899    -8.63721    -8.63721
-22.75899    7.31642
10.75020    38.73888    26.41588    10.75020    -6.26523    2.50271    3.20453
-6.26523    2.50271    25.24458    25.24458    -8.18677    -2.07520    -2.07520
-8.18677    11.23680
5.52466     10.12680    10.75020    12.02290    10.12680    10.75020    5.52466
-2.41734    3.20453    6.56174    6.38214    6.38214    6.56174    -2.15830
-2.15830    9.83788
-2.41734    -21.20290    -6.26523    10.12680    86.33018    38.73888    10.12680
-21.20290    -6.26523    -8.63721    -22.75899    53.41199    53.41199    -22.75899
-8.63721    7.31642
3.20453     -6.26523    2.50271    10.75020    38.73888    26.41588    10.75020
-6.26523    2.50271    -2.07520    -8.18677    25.24458    25.24458    -8.18677
-2.07520    11.23680
5.52466     -2.41734    3.20453    5.52466    10.12680    10.75020    12.02290
10.12680    10.75020    -2.15830    -2.15830    6.56174    6.38214    6.38214
6.56174     9.83788
10.12680    -21.20290    -6.26523    -2.41734    -21.20290    -6.26523    10.12680
86.33018    38.73888    -22.75899    -8.63721    -8.63721    -22.75899    53.41199
53.41199    7.31642
10.75020    -6.26523    2.50271    3.20453    -6.26523    2.50271    10.75020
38.73888    26.41588    -8.18677    -2.07520    -2.07520    -8.18677    25.24458
25.24458    11.23680
6.38214     53.41199    25.24458    6.56174    -8.63721    -2.07520    -2.15830
-22.75899    -8.18677    47.71928    28.99750    -10.87630    -1.90107    -10.87630
-23.13948    4.33471
6.56174     53.41199    25.24458    6.38214    -22.75899    -8.18677    -2.15830
-8.63721    -2.07520    28.99750    47.71928    -23.13948    -10.87630    -1.90107
-10.87630    4.33471
-2.15830    -22.75899    -8.18677    6.38214    53.41199    25.24458    6.56174
-8.63721    -2.07520    -10.87630    -23.13948    47.71928    28.99750    -10.87630
-1.90107    4.33471

```

Fig. 5.2.1 Cont.

-2.15830	-8.63721	-2.07520	6.56174	53.41199	25.24458	6.38214
-22.75899	-8.18677	-1.90107	-10.87630	28.99750	47.71928	-23.13948
-10.87630	4.33471					
6.56174	-8.63721	-2.07520	-2.15830	-22.75899	-8.18677	6.38214
53.41199	25.24458	-10.87630	-1.90107	-10.87630	-23.13948	47.71928
28.99750	4.33471					
6.38214	-22.75899	-8.18677	-2.15830	-8.63721	-2.07520	6.56174
53.41199	25.24458	-23.13948	-10.87630	-1.90107	-10.87630	28.99750
47.71928	4.33471					
9.83788	7.31642	11.23680	9.83788	7.31642	11.23680	9.83788
7.31642	11.23680	4.33471	4.33471	4.33471	4.33471	4.33471
4.33471	15.99079					

XFDV

0.075	0.0	0.075	0.075	0.0	0.075	0.075
0.0	0.075					
0.0	0.10					

LRFFAV

1	2	3	4	5	6	7
8	9					

SMXV

SMYV

BSMP	BSDP	DELU
1.0	-1.0	0.10
NHM	NIM	
10	10	
NTILT	NCTILT	
10	10	
GTILT		
-1.0		
NCVEL		
10		
WGTV		
0.0625	0.0625	0.0625
0.0625	0.0625	0.0625
0.0625	0.0625	0.0625

Fig. 5.2.1 Cont.

and prints the heading card and the array values and then identifies the selected modes by displaying the contents of the mode identification card with an asterisk to indicate the selected mode. For example, if MODV(7) was 1, the printer would display

* FIGURE SENSOR TEST FIGURE CONTROL SYSTEM TEST

while MODV(7) = 2 would produce

FIGURE SENSOR TEST * FIGURE CONTROL SYSTEM TEST

MODV(1) permits the user to select between manual operation or completely automatic simulation run control. Manual control is accomplished by operating a sense switch on the computer console which is interrogated by the software. Manual control permits the operator to start and terminate the simulation run manually, a necessary feature if on line plotting of results using a strip chart or xy plotter is desired.

MODV(2) permits the user to select between online data display and offline plotting mode. In the online mode plot data is transferred directly to the display device. The offline plotting capability permits the user to store data on magnetic tape for later display. The current EAM package does not include an online plotting capability; however, online plotting is easily added with few minor modifications.

MODV(3) permits the user to eliminate all nonlinearities which may have been incorporated in hardware component models and the figure control algorithm. This capability is extremely useful for verifying the results of a linear analysis.

MODV(4) permits the operator to select either the initial alignment mode for coarse orientation of the mirror segments in tilt or axial position (Sections 3.9 and 3.10) or the final figure control mode (Sections 3.2 and 3.3).

MODV(5) determines whether the tilt or axial alignment control systems are operated if the INITIAL ALIGNMENT mode is selected.

MODV(6) is utilized to select the actuator control algorithm appropriate for force or position figure actuation.

MODV(7) provides two control system modes of operation. If the FIGURE SENSOR TEST mode is requested, EAMCS calls MAINA(6) to process the figure sensor data each time a set of NMEASQ figure measurements have been generated at a measurement position. Thus the figure error vector will be generated in a sequential fashion. This is a particularly useful display mode for the development of figure sensor data processing algorithms. If FIGURE CONTROL SYSTEM TEST is selected the sum of the squares of the NMEASQ figure measurements at each position are stored until all the measurement locations have been scanned. At this point MAINB is called to process the entire set of figure measurements. The latter approach results in a desirable reduction in the number of Sigma 2 to Sigma 5 interrogations.

MODV(8) enables the user to operate the experimental mirror control system with the actual hardware components or with software models of the figure sensor, actuators and mirror. If MODV(12) = 2, for example, the control system will use the real time control software in the Sigma 2 and the component models which always reside in the Sigma 5.

MODV(9) permits the user to operate the software in the simulation or in the actual experiment operating mode. MODV(9) permits most of the complicated operating sequence control logic which is essential for experiment operation to be skipped.

Computer memory may be saved by utilizing the reduced mirror model matrix A_r rather than the full matrix A .^{*} The reduced matrix may be used by setting MODV(10) equal to 2.

MODV(11) provides the capability for testing the remote terminal control capability provided by the subroutine TYPCON when the EAM software is operated in the simulation mode. If MODV(11) is set to 2, the operator must provide the control commands to initialize, start and stop the figure control system. The communications may be simulated by card input-printer output or by the Sigma 5 typewriter console with appropriate device assignment through NTYPO and NTYPI.

The computer configuration may be changed using MODV(12). If MODV(12) is one all simulation computations are performed in the Sigma 5. A value of MODV(12) = 2 permits the simulation of the Sigma 5-2 operating configuration with all inter computer routine transfers performed via SUPE2 and SUPE5. This permits a complete check of the supervisory software in the Sigma 5.

5.2.3 Simulation Control Data

Basic control data for the simulation consists of the time step size, DT, the interval between output print data, TPRNT, the duration of each simulation run TEND, and the time interval, DTNOIS, between the generation of new stochastic structural disturbances. The sense switch assignment, NSSRUN, for manual control of the simulation is also input at this time.

* It is only necessary to use the complete matrix if the effects of a time-varying distributed disturbance, not adequately modelled by a set of loads or displacements applied at the actuator locations, is under investigation.

DT	TPRNT	TEND	DTNOIS
(E10.0)	(E10.0)	(E10.0)	(E10.0)
NSSRUN			
(I10)			

The value of DT should equal the real time control system cycle time Δt . TPRNT, TEND and DTNOIS should be integer multiples of Δt .

5.2.4 Run Set Identification and the Number of Runs

NRUN	NRUNM	
(I10)	(I10)	
NCXV	NCPV	NICPV
(I10)	(I10)	(I10)
(7(6X,A4))		
(7I10)		
(7(6X,A4))		
(7I10)		
(7(6X,A4))		
(7I10)		
CXM		
(7E10.0)		
CPM		
(7E10.0)		
CIPM		
(7E10.0)		

The EAM program structure permits a series of different runs to be performed without the necessity of reloading input data. This capability is provided by incorporating data editing routines EDITA, IEDITA, and arrays which store the edited data values for each run. A series of runs are assigned a run identification number NRUN which identifies the first run in the set. Subsequent runs are identified by NRUN+1, NRUN+2, etc. The number of runs is identified by the variable NRUNM.

Provisions are also included to operate the experiment with any preselected set of data. The remote terminal can be used, via TYPCON, to select the desired set of data. The required data modifications are then performed in SIMSYS.

The edited data must be associated with the elements of three arrays XV, PARV and IPARV by the addition of appropriate code in SIMSYS. The original values of XV, PARV and IPARV are stored. XV, PARV and IPARV, are reset to the original values before editing. The editing routine extracts the new element values from memory and produces the modified arrays.

The elements of XV, PARV, and IPARV which are to be modified are identified by the integer arrays JCXV, JCPV and JICPV of dimension NCXV, NCPV and NICPV, respectively. These arrays are inputted together with identifying names which are stored in the arrays NMCXV, NMCPV, and NMICPV.

The values to be used in each of the NRUNM runs are stored in the arrays CXM, CPM and ICPM.

5.2.5 Device Assignment for Manual Simulation Control

NTYPE
(I10)

Sense switch operation cues are generated for the simulation user in the manual control mode (MODV(1)=1). The cues are displayed on the device identified by the assignment associated with the value of NTYPE. The normal display device is the console typewriter.

5.2.6 Plotting Data

NPLOTV

(I10)

DTPLOT

(E10.0)

IPLOTV

(7I10)

IMODV

(7I10)

SCALV

(7E10.0)

Data to be plotted must be transferred to the XV. In general, a larger number of variables are transferred to XV than are actually plotted. The number of XV elements to be plotted is defined by NPLOTV. The information to be plotted is stored every DTPLOT seconds. The elements of XV to be plotted are identified by the IPLOTV.

An automatic scaling provision has been added to the software. The IPLOTV(K) element of XV may be automatically scaled for plotting by setting IMODV(K) = 2. If IMODV(K) = 1, the IPLOTV(K)th element of XV is plotted using the scale defined by SCALV(K).

5.2.7 Peripheral Device Assignment

NSNSWT	NTYPE	NTYPE	NPUNCH	NMAG
(I10)	(I10)	(I10)	(I10)	(I10)

The EAM software is designed to permit a variety of computer peripheral device configurations. For example, the remote peripheral utilized to control the experiment may be assigned to card reader input-line printer output for test purposes. Similar assignments may

be made for the punch and a magnetic storage device by reading appropriate values of NPUNCH and NMAG.

NSNSWT defines the interrupt assignment which is used to transfer control to the typewriter during experiment operation. Control transfer to the typewriter must be accompanied by an orderly termination of actuator motion.

5.2.8 Mirror Model Data

N	NR
(I10)	(I10)
AM	
(7E10.0)	
ASCALE	AIMSCL
(E10.0)	(E10.0)

The dimension N of the mirror model matrix AM and the number of actuators NR are read in at this point. If MODV(10) = 1, the complete NxN AM is inputted. The reduced NxNR AM is read in if MODV(10) = 2.

If the mirror matrix is not in the desired units (wavelengths/kilogram), it may be scaled by assigning a non unity value to ASCALE. If ASCALE $\neq$ 1 the computer multiplies AM by ASCALE and prints the resulting scaled matrix.

Computation of the gain matrices K_l or K_o involves matrix inversion of A_{rr} or $A_r^T A_r$. If numerical problems arise as a result of the limited dynamic range of the computer a scale factor $\beta_{ms}(AIMSCL)$ may be introduced to improve numerical accuracy. The matrix inversions are then performed on the matrices.

$$D = \beta_{ms} A_{rr} \quad (5.2.1)$$

or

$$D = \beta_{ms} A_r^T A_r \quad (5.2.2)$$

The inverse matrix is reconstructed from D^{-1} by

$$A_{rr}^{-1} = \beta_{ms} D^{-1} \quad (5.2.3)$$

or

$$\left[A_r^T A_r \right]^{-1} = \beta_{ms} D^{-1} \quad (5.2.4)$$

5.2.9 Figure Sensor Data

FSCALE

(E10.0)

XFSV

(7E10.0)

YFSV

(7E10.0)

PSCALE

(E10.0)

The output of the figure sensor phase detector is converted to wavelengths of figure error by the scale factor FSCALE.

The x and y coordinates of the N figure measurement positions* are read in as elements of the arrays XFSV and YFSV, respectively.

* corresponding to joint locations in the finite element model.

The scale factor PSCALE converts the coordinate data to a set of values suitable for input to the figure sensor image dissector.

5.2.10 Actuator Position Data

LACTV
(7I10)

The actuators are assigned to joint positions by setting NR elements of the LACTV array to one. The other elements should be set to zero.

5.2.11 Actuator Scale Factors

ASCALV
(7E10.0)

The actuator commands m_c are converted to values suitable for processing by the Sigma 2 and the actuator hardware components by multiplying each element of m_c by the corresponding element of ASCALV. ASCALV may also be utilized to correct differences in the actuator scale factors.

5.2.12 Figure Control Algorithm

MODOP
(I10)

The figure control algorithm type may be selected by assigning an appropriate value to MODOP. If MODOP = 1, the gain matrix GAINM for the simplified linear control system is calculated by MFCS. MODOP = 2 results in computation of the linear optimal gain matrix. A value of MODOP = 3 inputs a NR by N gain matrix.

GAINM
(7E10.0)

5.2.13 Actuator Test Data

NMEASA
(I10)
DACT
(E10.0)

Test data for routine ACTCAL consists of the number of measurements to be performed and averaged NMEASA and the magnitude of the actuator command perturbation DACT.

5.2.14 Mirror Calibration Test Data

NMEASF
(I10)
DACT
(E10.0)

MIRCAL reads the number of calibration tests NMEASF to be performed and averaged and the size of the mirror test actuator command perturbation DACT.

5.2.15 Real Time Control System Parameters

NTIMSO	NWAIT	NPOS	NMINT	NMEAS	NTYO
(I10)	(I10)	(I10)	(I10)	(I10)	(I10)
DT	DTE	GAINV(1)	QGA	QGB	UFMAX
(E10.0)	(E10.0)	(E10.0)	(E10.0)	(E10.0)	(E10.0)

The real time control schedule parameters define the number of control cycles for actuator manipulation, NWAIT, the number of cycles allowed for the figure sensor output to settle after a dissector position change, NPOS, the number of control cycles between measurements, NMINT, and the number of measurements at each location, NMEAS.

The control system status is displayed on the typewriter after every NTYO complete sets of mirror figure error measurements. Currently typed data includes the elapsed operating time and the rms value of the figure measurements, PINDEX.

DT is the control system cycle time and DTE is a term which is added to T in the cycle time control loop (EAMCS) to compensate for roundoff error.

The scalar gain matrix multiplier β_g is read in as the first element of GAINV. QGA and QGB are the actuator control system gains. UFMAX is a limit imposed on the actuator command signals to prevent possible damage to the actuators or the mirror structure.

5.2.16 Figure Sensor Filter Parameters

SIGLIM SLP MN
(E10.0) (E10.0)
MSEQV
(7I10)

The digital figure error data processor required the limit on the rms measurement error SIGLIM, the extrapolation factor SLP MN for ambiguous measurements and the scanning sequence MSEQV for the N measurement points. The figure sensor looks at the point defined by MSEQV(N) first; i. e., XFSV(MSEQV(N)), YFSV(MSEQV(N)). Subsequent measurements are made at MSEQV(N-1), MSEQV(N-2), etc.

5.2.17 Figure Sensor Model Data

FSNSIG FSTFLT
(E10.0) (E10.0)
IRAND
(I10)

The figure sensor model data consists of the rms measurement noise FSNSIG, phase detector filter time constant FSTFLT, and the initial starting value IRAND for the random number generator which should be an odd integer.

5.2.18 Actuator Model Data

TACTV
(7E10.0)

The time constants for the first order actuator models are read in as NR elements of TACTV.

5.2.19 Mirror Model Data

AMM
(7E10.0)
XFDV
(7E10.0)

Data for the mirror model consists of the mirror model matrix AMM and the initial figure error XFDV. The mirror model matrix is read in in the same form as AM and scaled, if necessary, using ASCALE.

5.2.20 Initial Alignment Control System Data

LREFAV
(7I10)
SMXV
(7E10.0)
SMYV
(7E10.0)
BSMP BSDP DELU
(E10.0) (E10.0) (E10.0)
NHM NIM
(I10) (I10)
NTILT NCTILT
(I10) (I10)
GTILT
(E10.0)

The initial alignment control system data is read in by MAINC. The vector LREFAV of dimension 9 identifies the actuator allocation during initial alignment. The first three elements of LREFAV identify the elements of UFV associated with segment one. The first element provides the reference actuator for initial tilt alignment. The second and third elements identify the actuators used in the secondary tilt adjustments. The next three, and last three elements of LREFAV identify the actuators associated with the second and third segments in a similar fashion. The X and Y coordinates of the actuators are read in as the arrays SMXV and SMYV, respectively.

The peak ambiguity sensor model output BSMP and the second order coefficient BSDP are read in next. Note that BSDP must be less than zero.

The slew control system algorithm requires the actuator output

perturbation DELU, the maximum number of successful control algorithm iterations NIM and the maximum number of step-size halvings NHM as input variables.

Tilt control requires the number of control computations at each computed measurement position NCTILT and the number of scan path divisions NTILT (between the reference and secondary actuator positions). The control loop gain GTILT is also required.

5.2.21 Performance Index Data

WGTV
(7E10.0)

The weighting factors for the performance index are read in as N elements of the WGTV array.

CHAPTER 6

REMOTE CONTROL OF THE EXPERIMENT

6.1 Introduction

Since the Sigma 5 - 2 computers are some distance from the experimental hardware, it is desirable to incorporate a capability for controlling the experiment from a point remote from the computation facility.

The experimental active mirror software has been designed to permit operation of the experiment from a remote location by means of a peripheral device such as a teletype, for example. Instructions required to initiate, start and stop the experiment, display and modify control system parameters and more complicated functions such as actuator test and calibration are incorporated.

Coding has also been generated to permit the automatic periodic display of important control system parameters, for monitoring purposes, on the remote terminal.

6.2 Experiment Control Commands

The remote terminal is designated by the device assignments NTYPI for input and NTYPO for output (NTYPIQ and NTYPOQ in the Sigma 2 software). The split assignment enables the card reader to be used to simulate remote input while the line printer is used for remote output during initial checkout, for example.

Conversation with the remote terminal is initiated by calling TYPCON with NENTRY = 2. TYPCON (2) is called automatically by EAMCS to request control system initialization and start instructions. During experiment operation TYPCON (2) may be called by enabling a Sigma 2 interrupt or sense switch which sets MODEQ = 3 stopping the experiment and generating a request for instructions.

TYPCON (2) requests a value for MODEQ by typing MODE = ? and waiting for the operator to respond by typing an integer in I3 format. The integer value must be greater than zero and less than 13. The functions associated with each mode value are:

- MODEQ = 1 Initializes the mirror figure control system.
- MODEQ = 2 Starts the mirror figure control system. EAMCS checks to see if MODEQ = 2 was preceded by MODEQ = 1 to assure that the control system is ready to start.
- MODEQ = 3 Stops the mirror figure control system, freezes the figure actuators and returns control to TYPCON (2) for further instructions.
- MODEQ = 4 Provides a calling sequence to test the actuators for correct operation via MFCS and ACTCAL.
- MODEQ = 5 Provides a calling sequence to calibrate the mirror structure via calls to MFCS and MIRCAL.
- MODEQ = 6 Initiates the diagnostic mode of operation of TYPCON. TYPCON will then request a variable name and index

(indices) from the experiment operator. The operator responds by typing a variable name in (1X, A4) format. The name is checked and associated with a numerical identifier by MAINC. If the name is not a member of a stored list, an error is registered and a new name requested. The value of the numerical identifier determines whether or not an index or indices are requested of the operator. The member of the data, thus identified, is displayed on the remote terminal. The system remains in the diagnostic mode until the name DEND is submitted, resulting in a request for a new value of MODEQ.

MODEQ = 7 Provides the steps required to modify the input data to correspond to run NRUN using the stored information and editing capability of SIMSYS and a value for NRUN provided on request by the operator.

MODEQ = 8 Resumes operation of the experiment if termination has occurred during operation as a result of MODEQ = 3.

MODEQ = 9 Evaluates and types the rms figure error using PINDX.

MODEQ = 10 Initiates a parameter modification mode. The variable identification sequence for MODEQ = 6 is followed by a request for a new variable value

which is accepted in (I4) or (F12.6) format. The terminal responds by retyping the new variable value.

MODEQ = 11 Unused mode.

If MODEQ is greater than 11, the terminal types MODE TOO BIG and requests a new value for MODEQ.

6.3 Experiment Monitoring Capability

The software is designed to output data useful for monitoring the active mirror experiment. The data currently consists of the rms value of the N elements of the figure error vector XFV calculated by PINDX and the total operating time of the current experiment. The operating time is the product of the number of control cycles ITIMS and the control cycle time DT.

CHAPTER 7

SUMMARY

7.1 General Features of the Digital Control System

This report describes the current version of a versatile multi-computer software system developed at MIT/DL to simulate a mirror figure control system and by simple modification to provide the control software for an experimental active mirror at the Marshall Space Flight Center.

The internally generated control laws provide for initial alignment of the segmented mirror, and simplified linear and linear optimal algorithms for fine alignment of the segmented and deformable mirror figures. The control algorithm is also designed to accept a general gain matrix for the fine figure alignment algorithm.

The software includes a digital filter to process figure sensor data. The filter operates to remove measurement ambiguities and signal noise which arise as a result of the inherent characteristics of the interferometric figure sensor.

The software is written in FORTRAN, permitting execution of the simulation on a broad spectrum of different computers.

The control software for the experiment is an integral part of the simulation. Thus, it is possible to completely check out and evaluate a programmed control algorithm by simulation before execution with the hardware is attempted.

The software is arranged so that the simulation may be operated entirely within the boundaries of one computer. In this mode it is

possible to simulate operation in the two computer configuration. The simulation may also be operated in a two computer mode with the Sigma 5 simulation system utilizing the real time control in the Sigma 2.

Operation of the software with the hardware components is achieved by a simple modification of an array of input operating mode variables. Thus, it is not necessary to juggle program modules to convert from a simulation to an experiment operation mode and vice versa.

Simplified software models are provided for all components of the segmented and deformable mirror control systems. Models are included for the

1. mirror structure
2. mirror figure sensor
3. segment axial alignment (ambiguity) sensors
4. position figure actuators
5. force figure actuators

The software has been partitioned to permit the simple substitution of more elaborate models of each of the component models if desired.

An extensive remote control, diagnosis and parameter modification capability has been incorporated in the software to permit operation of the experiment from a terminal distant from the computer facility.

Data modification may be accomplished by rereading a modified input data deck, or by utilizing an automatic data editing capability and prestored values, or by directly modifying system parameters from the remote terminal.

The sequence of operator commands to the experiment is monitored to prevent errors. Thus the start command must be preceded by the initialization command, for example.

Actuator command signals to the figure actuators are limited in magnitude to prevent damage to the experiment in the event of an operator error.

The figure actuators are automatically "frozen" during figure error measurement, figure error processing and control generation and in the event of an operator or program induced termination of experiment operation. Freezing the actuator positions minimizes the effects of actuator motion on figure measurement and prevents actuator divergence during periods when the Sigma 2 computer is not available for real time component control.

Evaluation of the experimental or simulation results is facilitated by the inclusion of software to compute a number of performance indices.

Provisions have also been made for incorporation of online or offline plotting of experimental data. The plotting routine includes data acquisition, storage, and preparation for plotting. Automatic scale generation features are also included.

The software is designed to periodically display experiment status information on the operator's console for monitoring purposes. The status data currently includes the rms figure error and the experiment operating time.

APPENDIX A

EXPERIMENTAL ACTIVE MIRROR SOFTWARE LISTINGS

A.1 Introduction

This appendix contains software listings of each EAM software module with the exception of minor subroutines which appear in the library packages in Appendix B. The subroutines are presented in alphabetical order. The following general procedures were adhered to during software construction.

1. The names of one dimensional arrays end in V in general.
2. The names of two dimensional arrays end in M in general.
3. Data is generally read in and initial computations performed when a subroutine is called with
NENTRY = 1.
4. Initializations are performed when NENTRY = 2
5. Values of NENTRY > 2 generally signify that a system computation or data transfer is being performed.

All arrays appear in one dimensional form in the software package. This convention, which follows the practice of IBM in the development of their subroutines for one and two dimensional array manipulation, has a number of important advantages including savings in memory space by the elimination of compiler generated code and the elimination of many of the problems which arise when variables are transferred via subroutine parameter lists.

For example, the array AM which represents an $n \times n$ matrix may be simply set to zero by

```

      I = N * N
      DO 2000 J = 1, I
2000  AM(J) = 0.0

```

(A.1.1)

whereas the two dimensional $n \times n$ array BM requires the following code to be zeroed

```

      DO 2000 I = 1, N
      DO 2000 J = 1, N
2000  BM(I, J) = 0.0

```

(A.1.2)

Note that the computer, which treats all arrays as one dimensional, must compute each storage location L of the I, J th element in (A.1.2) by a computation similar to

$$L = (J - 1) * N + I + (L_r - 1) \quad (A.1.3)$$

where L_r is the necessary address of the array BM. In the case of (A.1.1) L is merely

$$L = I + (L_r - 1) \quad (A.1.4)$$

Suppose that the arrays AM and BM are to be passed to a subroutine as parameters. The array AM may be dimensioned

```

      DIMENSION AM(1)

```

(A.1.5)

in the subroutine regardless of the value of N in the calling program whereas BM must be dimensioned

```

      DIMENSION BM(N, N)

```

(A.1.6)

in both the main program and the subroutine. This leads to the possibility of errors and restricts the usefulness of the subroutine.

The data block utilized to transfer information to or from the Sigma 2 is labelled SIGTWO. Variables utilized by both the Sigma 2 and Sigma 5 software are identified in the data block by the suffix Q in the case of integer data or the prefix Q in the case of floating point variables. The SIGTWO block arrays QUFAV and MODVQ correspond to UFAV and MODV in the Sigma 5, for example.

```

SUBROUTINE ACTCAL(NENTRY)
SUBROUTINE ACTCMD(NENTRY)
SUBROUTINE ACTMDL(NENTRY,IENTRY)
SUBROUTINE EAMCS(NENTRY)
SUBROUTINE FIGSEN(NENTRY,I)
SUBROUTINE FSHDL(NENTRY,IENTRY)
SUBROUTINE MAINA(NENTRY)
SUBROUTINE MAINH(NENTRY)
SUBROUTINE MAINC(NENTRY)
SUBROUTINE MFCS(NENTRY)
SUBROUTINE MIRCAL(NENTRY)
SUBROUTINE MIRMDL(NENTRY,IACT)
SUBROUTINE PINDX(NENTRY,PINDEX,YV)
SUBROUTINE PLT(NENTRY,XV,T,DT,NRUN)
SUBROUTINE RESPON(NENTRY)
SUBROUTINE SIMSYS(NENTRY)
SUBROUTINE STORED(NENTRY,RANG,WIDTH,SPAC,SCALV,X,Y,XSPRED,NPLOTV,
1 NPTS,NRUN)
SUBROUTINE SUPE2
MAIN PROG SUPE5
SUBROUTINE TYPCON(NENTRY)

```

	SUBROUTINE ACTCAL(NENTRY)	EAM10000
C		EAM10010
C	SUBROUTINE TO TEST FIGURE ACTUATORS	EAM10020
C		EAM10030
C	SIGMA 5 TYPE B DIMENSION STATEMENTS START	EAM10040
	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM10050
	1 YFSV(20),XFRV(20),DUMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM10060
	2 GAINM(1600),ASV(3)	EAM10070
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DUMV,UFAV,	EAM10080
	1 DUMVA,GAINV,GAINM,ASV	EAM10090
C		EAM10100
	DIMENSION LACTV(20)	EAM10110
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM10120
	1 NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM10130
C		EAM10140
	DIMENSION AM(400),AIM(400)	EAM10150
	COMMON/BLKMFC/AM,AIM	EAM10160
C		EAM10170
	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM10180
	1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM10190
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM10200
	3 MODV(20)	EAM10210
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM10220
	1 NCXV,NCPV,NICPV,JCXV, JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM10230
	2 NMCXV,NMCPV,NMICPV,MODV	EAM10240
C		EAM10250
	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM10260
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM10270
C		EAM10280
C	SIGMA 5 TYPE B DIMENSION STATEMENTS END	EAM10290
C		EAM10300
	SIGMA 2 DIMENSION STATEMENTS START	EAM10310
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM10320
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM10330
	2 QASV(3)	EAM10340
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM10350
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENQ,NTIMSQ,LSENS,	EAM10360
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM10370
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM10380
C		EAM10390
C	SIGMA 2 DIMENSION STATEMENTS END	EAM10400
	DIMENSION DUMVD(20)	EAM10410
C		EAM10420
	1000 FORMAT(7H ACTCAL)	EAM10430
	1002 FORMAT(/,3X,12HACTOUT/ACTIN)	EAM10440
C		EAM10450
	GO TO (1,2,3,4,5,6),NENTRY	EAM10460
C		EAM10470
C	INPUT DATA	EAM10480
1	PRINT 1000	EAM10490
	CALL IRANDP(1,NMEASA,IA,IA,IA,IA,IA,IA,4)	EAM10500
	CALL RANDPD(1,DACT,DA,DA,DA,DA,DA,DA,4)	EAM10510
	DB=DACT*2.0	EAM10520
	DB=1.0/(NMEASA*DB)	EAM10530
	RETURN	EAM10540
C		EAM10550
C	INITIALIZATION	EAM10560
2	RETURN	

C		EAM10570
C	ACTUATOR CALIBRATION	EAM10580
C	RETURN TO SIGMA 2 TO INITIALIZE EAMCS	EAM10590
3	IGOA=MODV(12)	EAM10600
	GO TO (2213,2200),IGOA	EAM10610
C*****	EAM SOFTWARE TEST CODING*****	EAM10620
2213	CALL EAMCS(8)	EAM10630
	GO TO 4	EAM10640
C*****	EAM SOFTWARE TEST CODING*****	EAM10650
2200	CALL MARK(1,22,8,8,4)	EAM10660
	RETURN	EAM10670
C		EAM10680
4	SGAIN=GAINV(1)	EAM10690
	GAINV(1)=0.0	EAM10700
	NTIMS=NWAIT	EAM10710
	NTIMSQ=NTIMS	EAM10720
	DO 2202 I=1,NR	EAM10730
2202	DUMV(I)=0.0	EAM10740
	J=0	EAM10750
C	DETERMINE THE STEADY STATE ACTUATOR GAINS NMEASA TIMES AND	EAM10760
C	AVERAGE THE RESULTS	EAM10770
2208	J=J+1	EAM10780
	IF(J-NMEASA)2209,2209,2210	EAM10790
2209	DO 2203 K=1,NR	EAM10800
	UFV(K)=-DACT	EAM10810
2203	QUFV(K)=UFV(K)*ASCALV(K)	EAM10820
C	RETURN TO SIGMA 2 TO ADJUST THE ACTUATORS	EAM10830
	GO TO (2212,2204),IGOA	EAM10840
C*****	EAM SOFTWARE TEST CODING*****	EAM10850
2212	CALL EAMCS(3)	EAM10860
	GO TO 5	EAM10870
C*****	EAM SOFTWARE TEST CODING*****	EAM10880
2204	CALL MARK(1,22,3,8,5)	EAM10890
	RETURN	EAM10900
C		EAM10910
5	DO 2205 K=1,NR	EAM10920
	UFAV(K)=QUFAV(K)/ASCALV(K)	EAM10930
	DUMVD(K)=UFAV(K)	EAM10940
	UFV(K)=DACT	EAM10950
2205	QUFV(K)=UFV(K)*ASCALV(K)	EAM10960
C	RETURN TO SIGMA 2 TO ADJUST THE ACTUATORS	EAM10970
	GO TO (2211,2206),IGOA	EAM10980
C*****	EAM SOFTWARE TEST CODING*****	EAM10990
2211	CALL EAMCS(3)	EAM11000
	GO TO 6	EAM11010
C*****	EAM SOFTWARE TEST CODING*****	EAM11020
2206	CALL MARK(1,22,3,8,6)	EAM11030
	RETURN	EAM11040
C		EAM11050
6	DO 2201 K=1,NR	EAM11060
	UFAV(K)=QUFAV(K)/ASCALV(K)	EAM11070
2201	DUMV(K)=(UFAV(K)-DUMVD(K))+DUMV(K)	EAM11080
	GO TO 2208	EAM11090
2210	CONTINUE	EAM11100
	DO 2207 I=1,NR	EAM11110
	UFV(I)=0.0	EAM11120
	QUFV(I)=0.0	EAM11130

2207	DUMV(I)=DUMV(I)*DB	EAM11140
C		EAM11150
C	PRINT OUT ACTUATOR SCALE VECTOR	EAM11160
	PRINT 1002	EAM11170
	CALL MXRNP(DUMV,1,NR,3)	EAM11180
	GAINV(1)=SGAIN	EAM11190
	RETURN	EAM11200
C		EAM11210
	END	EAM11220
	SUBROUTINE ACTCMD(NENTRY)	EAM11230
C		EAM11240
C	SUBROUTINE TO SCALE AND TRANSFER ACTUATOR COMMANDS AND ACTUATOR MEASUREMENTS.	EAM11250
C		EAM11260
C		EAM11270
C	SIGMA 2 DIMENSION STATEMENTS START	EAM11280
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM11290
1	QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM11300
2	QASV(3)	EAM11310
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM11320
1	MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSENS,	EAM11330
2	NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM11340
3	QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELO	EAM11350
C	SIGMA 2 DIMENSION STATEMENTS END	EAM11360
C		EAM11370
	GO TO(1,2,3,4,5),NENTRY	EAM11380
C		EAM11390
C	INPUT DATA	EAM11400
1	RETURN	EAM11410
C		EAM11420
C	INITIALIZATION	EAM11430
2	SQGA=QGA	EAM11440
	RETURN	EAM11450
C		EAM11460
C	TRANSFER ACTUATOR COMMANDS AND MEASURE ACTUATER OUTPUTS	EAM11470
C		EAM11480
3	I=0	EAM11490
2309	I=I+1	EAM11500
	IF(I-NRQ)2308,2308,2303	EAM11510
C	LIMIT QUFV(I)	EAM11520
2308	IF(QUFV(I)-QUFMAX)2304,2304,2305	EAM11530
2304	IF(QUFV(I)+QUFMAX)2306,2307,2307	EAM11540
2305	QUFV(I)=QUFMAX	EAM11550
	GO TO 2307	EAM11560
2306	QUFV(I)=-QUFMAX	EAM11570
C	RETURN TO SIGMA 5 TO MODEL ACTUATORS IF MODVQ(8)=1	EAM11580
2307	IGO=MODVQ(8)	EAM11590
	GO TO(2301,2311),IGO	EAM11600
2301	NFLGA=7	EAM11610
	NFLGB=I	EAM11620
	IGO=MODVQ(12)	EAM11630
	GO TO (2302,2310),IGO	EAM11640
C*****	EAM SOFTWARE TEST CODING*****	EAM11650
2302	CALL ACTMDL(3,I)	EAM11660
	GO TO 4	EAM11670

C*****EAM SOFTWARE TEST CODING*****	EAM11680
2310 RETURN	EAM11690
C	EAM11700
4 GO TO 2309	EAM11710
C	EAM11720
2311 CONTINUE	EAM11730
C INSERT SOFTWARE TO MEASURE ACTUATOR POSITION	EAM11740
C QUFAV(I)=ACTUATOR POSITION	EAM11750
C CALCULATE THE ACTUATOR ERROR	EAM11760
DA=QUFV(I)-QUFAV(I)	EAM11770
C ACCUMULATE THE ERROR IN QUFERV	EAM11780
QUFERV(I)=QUFERV(I)+QDT*QGB*DA	EAM11790
C FORM THE ACTUATOR CONTROL	EAM11800
DA=QGA*(DA+QUFERV(I))	EAM11810
C SET ACTUATOR CONTROL EQUAL TO 0 IF NENTRY=5	EAM11820
GO TO(2402,2402,2402,2402,2401),NENTRY	EAM11830
2401 DA=0.0	EAM11840
2402 CONTINUE	EAM11850
C INSERT SOFTWARE TO TRANSFER ACTUATOR COMMANDS TO ACTUATORS	EAM11860
C DA IS THE INPUT FOT THE ITH ACTUATOR	EAM11870
GO TO 2309	EAM11880
2303 NFLGA=6	EAM11890
RETURN	EAM11900
C	EAM11910
C FREEZE ACTUATORS	EAM11920
5 QGA=0.0	EAM11930
NFLGA=6	EAM11940
RETURN	EAM11950
C	EAM11960
C RELEASE ACTUATOR OUTPUTS	EAM11970
C	EAM11980
6 QGA=SQGA	EAM11990
NFLGA=6	EAM12000
RETURN	EAM12010
C	EAM12020
C	EAM12030
END	EAM12040

C	SUBROUTINE ACTMDL(NENTRY,IENTRY)	EAM12050
C		EAM12060
C	SIMPLIFIED ACTUATOR MODEL FOR TESTING THE EAM SOFTWARE PACKAGE	EAM12070
C		EAM12080
C	SIGMA 5 TYPE B DIMENSION STATEMENTS START	EAM12090
	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM12100
1	YFSV(20),XFRV(20),DUMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM12110
2	GAINM(1600),ASV(3)	EAM12120
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DUMV,UFAV,	EAM12130
1	DUMVA,GAINV,GAINM,ASV	EAM12140
C		EAM12150
	DIMENSION LACTV(20)	EAM12160
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM12170
1	NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSSENS,NTIMS	EAM12180

C	DIMENSION AM(400),AIM(400)	EAM12190
	COMMON/BLKMFCA/AM,AIM	EAM12200
C	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM12210
	1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM12220
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM12230
	3 MODV(20)	EAM12240
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM12250
	1 NCXV,NCPV,NICPV,JCXV, JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM12260
	2 NMCXV,NMCPV,NMICPV,MODV	EAM12270
C	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM12280
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM12290
C	SIGMA 5 TYPE B DIMENSION STATEMENTS END	EAM12300
C		EAM12310
C	SIGMA 2 DIMENSION STATEMENTS START	EAM12320
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM12330
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM12340
	2 QASV(3)	EAM12350
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM12360
	1 MSEQVQ,NSSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSSENS,	EAM12370
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM12380
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM12390
C	SIGMA 2 DIMENSION STATEMENTS END	EAM12400
C		EAM12410
C	DIMENSION TACTV(20),AGAMV(20),APHIV(20)	EAM12420
		EAM12430
C	1000 FORMAT(7H ACTMDL)	EAM12440
	1001 FORMAT(10X,5HAPHIV)	EAM12450
	1002 FORMAT(10X,5HAGAMV)	EAM12460
C		EAM12470
	I=IENTRY	EAM12480
	GO TO(1,2,3),NENTRY	EAM12490
C		EAM12500
C	INPUT DATA	EAM12510
1	PRINT 1000	EAM12520
	CALL MXRNP(TACTV,1,NR,4)	EAM12530
	RETURN	EAM12540
C		EAM12550
C	INITIALIZATION	EAM12560
C	CONSTRUCT THE ACTUATOR MODELS	EAM12570
2	DO 2001 J=1,NR	EAM12580
	APHIV(J)=0.0	EAM12590
	AGAMV(J)=1.0	EAM12600
	IF(TACTV(J))2001,2001,2002	EAM12610
2002	DA=-(DT/TACTV(J))	EAM12620
C	CALCULATE THE STATE TRANSITION MATRICES FOR THE ACTUATORS	EAM12630
	APHIV(J)=EXP(DA)	EAM12640
C	CALCULATE THE INPUT TRANSITION MATRICES FOR THE ACTUATORS	EAM12650
	AGAMV(J)=1.0-APHIV(J)	EAM12660
2001	CONTINUE	EAM12670
	PRINT 1001	EAM12680
	CALL MXRNP(APHIV,1,NR,3)	EAM12690
	PRINT 1002	EAM12700
	CALL MXRNP(AGAMV,1,NR,3)	EAM12710
	RETURN	EAM12720
C		EAM12730
		EAM12740
		EAM12750
		EAM12760

C	SIMULATION	EAM12770
C	OBTAIN UFV FROM THE SIGMA 2	EAM12780
3	UFV(I)=QUFV(I)/ASCALV(I)	EAM12790
C	SIMULATE THE ACTUATOR DYNAMICS IN THE SIGMA 5	EAM12800
	UFAV(I)=APHIV(I)*UFAV(I)+AGAMV(I)*UFV(I)	EAM12810
C	TRANSFER NEW VALUE OF UFV TO SIGMA 2	EAM12820
	QUFAV(I)=UFAV(I)*ASCALV(I)	EAM12830
	RETURN	EAM12840
C		EAM12850
	END	EAM12860
	SUBROUTINE EAMCS(NENTRY)	EAM12870
C		EAM12880
C	SUBROUTINE TO REALIZE THE REAL TIME CONTROL SYSTEM FOR THE	EAM12890
C	EXPERIMENTAL ACTIVE MIRROR	EAM12900
C		EAM12910
C	SIGMA 2 DIMENSION STATEMENTS START	EAM12920
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM12930
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM12940
	2 QASV(3)	EAM12950
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM12960
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENQ,NTIMSQ,LSENS,	EAM12970
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM12980
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM12990
C	SIGMA 2 DIMENSION STATEMENTS END	EAM13000
C		EAM13010
	1002 FORMAT(16H INITIALIZE MFCS)	EAM13020
	1005 FORMAT(11H START MFCS)	EAM13030
	1003 FORMAT(10H BEGIN RUN)	EAM13040
	1009 FORMAT(15H SEQUENCE ERROR)	EAM13050
	1010 FORMAT(11H MODE NOT =,I2)	EAM13060
C		EAM13070
	GO TO(1,2,3,3,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19),NENTRY	EAM13080
C		EAM13090
C	INPUT DATA	EAM13100
1	RETURN	EAM13110
C		EAM13120
C	INITIALIZATION	EAM13130
C		EAM13140
2	WRITE(NTYPOQ,1003)	EAM13150
	NFLGB=1	EAM13160
	CALL TYPCON(1)	EAM13170
	CALL ACTCMD(2)	EAM13180
C		EAM13190
C	INITIALIZE ACTIVE MIRROR	EAM13200
9	WRITE(NTYPOQ,1002)	EAM13210
	MODES=1	EAM13220
	NFLGA=1	EAM13230
	IGOA=MODVQ(12)	EAM13240
	GO TO (2205,2206),IGOA	EAM13250
	2206 RETURN	EAM13260

C		EAM13270
2205	CALL TYPCON(2)	EAM13280
10	IF(MODEQ-MODES)2229,2230,2229	EAM13290
2229	WRITE(NTYPOQ,1010) MODES	EAM13300
	GO TO 2	EAM13310
2230	JSENS=1	EAM13320
	IGOA=MODVQ(12)	EAM13330
	ISENS=1	EAM13340
	LSENS=MSEQVQ(NQ)	EAM13350
	MSENS=0	EAM13360
	QXF=0.0	EAM13370
	SXF=0.0	EAM13380
	SXFXF=0.0	EAM13390
C	SET ACTUATOR INTEGRAL COMPENSATORS TO ZERO	EAM13400
	DO 2232 I=1,NRQ	EAM13410
2232	QUFERV(I)=0.0	EAM13420
C	SET STORAGE VECTORS FOR SXF AND SXFXF=0	EAM13430
	DO 2134 I=1,NQ	EAM13440
	QDUMVB(I)=0.0	EAM13450
2134	QDUMVC(I)=0.0	EAM13460
	IF(NENTRY-8)2233,2231,2233	EAM13470
2231	NFLGA=6	EAM13480
	RETURN	EAM13490
C		EAM13500
C	START ACTIVE MIRROR	EAM13510
2233	MODES=2	EAM13520
	WRITE(NTYPOQ,1005)	EAM13530
	NFLGA=2	EAM13540
	GO TO(2235,2237),IGOA	EAM13550
2235	CALL TYPCON(2)	EAM13560
11	IF(MODEQ-MODES)2,2236,2	EAM13570
2236	NFLGA=6	EAM13580
2237	RETURN	EAM13590
C		EAM13600
C	CONTROL SYSTEM COMPUTATIONS	EAM13610
C	ESTABLISH STARTING TIME	EAM13620
3	CALL REALT(T)	EAM13630
	TSTORE=T	EAM13640
	ITIMS=0	EAM13650
C	PERFORM CONTROL LOOP OPERATIONS FOR NTIMSQ CYCLES	EAM13660
2400	ITIMS=ITIMS+1	EAM13670
	IF(ITIMS-NTIMSQ)2401,2401,2402	EAM13680
C	NORMAL TERMINATION TO SIGMA 5 FOR FURTHER INSTRUCTIONS	EAM13690
2402	IGO=MODVQ(9)	EAM13700
	GO TO(2236,2702),IGO	EAM13710
C		EAM13720
C	FIGURE SENSOR CONTROL STRUCTURE	EAM13730
C		EAM13740
C	MEASURE FIGURE ERRORS EVERY NSENSQ*DT SECONDS	EAM13750
2401	JSENS=JSENS-1	EAM13760
	GO TO(2301,2302,2303,2304,2305),ISENS	EAM13770
2301	IF(JSENS)2310,2310,2390	EAM13780
2310	ISENS=2	EAM13790
	JSENS=NSENSQ	EAM13800
	JWAIT=NWAITQ	EAM13810
	MSENS=NQ	EAM13820
12	GO TO 2390	EAM13830
C	WAIT NWAITQ*DT SECONDS FOR THE ACTUATOR OUTPUTS TO STABILIZE	EAM13840
2302	JWAIT=JWAIT-1	EAM13850
	IF(JWAIT)2320,2320,2390	EAM13860
2320	ISENS=3	EAM13870

C	FREEZE ACTUATOR POSITIONS	EAM13880
	CALL ACTCMD(5)	EAM13890
C	RETURN TO SIGMA 5 SOFTWARE TO UPDATE MIRROR MODEL OUTPUTS	EAM13900
C	IF MODVQ(8)=1	EAM13910
	IGO=MODVQ(8)	EAM13920
	GO TO(2329,7),IGO	EAM13930
2329	NFLGA=4	EAM13940
2322	GO TO (2323,2324),IGOA	EAM13950
2324	RETURN	EAM13960
C*****	EAM SOFTWARE TEST CODING*****	EAM13970
2323	CALL MIRMDL(3,1)	EAM13980
	GO TO 7	EAM13990
C*****	EAM SOFTWARE TEST CODING*****	EAM14000
C	TRANSFER POSITION COORDINATES TO IMAGE DISSECTOR	EAM14010
7	CONTINUE	EAM14020
2303	LSENS=MSEQVQ(MSENS)	EAM14030
C	POSITION FIGURE SENSOR IMAGE DISSECTOR	EAM14040
	NFLGA=5	EAM14050
	NFLGB=LSENS	EAM14060
	GO TO (2325,2326),IGOA	EAM14070
2326	RETURN	EAM14080
C		EAM14090
2325	CALL FIGSEN(2,LSENS)	EAM14100
13	JWAIT=NPOSQ	EAM14110
	ISENS=4	EAM14120
	GO TO 2390	EAM14130
C	WAIT NPOSQ*DT FOR THE MEASUREMENT POSITION TO STABILIZE	EAM14140
2304	JWAIT=JWAIT-1	EAM14150
	IF(JWAIT)2340,2340,2390	EAM14160
2340	ISENS=5	EAM14170
	JWAIT=NMINTQ	EAM14180
	JMEAS=NMEASQ	EAM14190
	KMEAS=0	EAM14200
	SXF=0.0	EAM14210
	SXFXF=0.0	EAM14220
	GO TO 2390	EAM14230
C	TAKE NMEASQ MEASUREMENTS AT INTERVALS OF NMINTQ*DT SECONDS AT EACH	EAM14240
C	MEASUREMENT POINT	EAM14250
2305	JWAIT=JWAIT-1	EAM14260
	IF(JWAIT)2350,2350,2390	EAM14270
C	TAKE FIGURE ERROR PHASE MEASUREMENT	EAM14280
2350	NFLGA=16	EAM14290
	NFLGB=LSENS	EAM14300
	GO TO(2327,2328),IGOA	EAM14310
2328	RETURN	EAM14320
C		EAM14330
2327	CALL FIGSEN(3,LSENS)	EAM14340
14	JMEAS=JMEAS-1	EAM14350
	KMEAS=KMEAS+1	EAM14360
	SXF=SXF+QXF	EAM14370
	SXFXF=SXFXF+QXF*QXF	EAM14380
	JWAIT=NMINTQ	EAM14390
C	CHECK TO SEE IF NMEASQ MEASUREMENTS HAVE BEEN MADE	EAM14400
	IF(JMEAS)2351,2351,2390	EAM14410
C		EAM14420
C	STORE THE SUM OF THE FIGURE MEASUREMENTS IN QDUMVB	EAM14430
2351	QDUMVB(LSENS)=SXF	EAM14440
C	STORE THE SUM OF THE SQUARES OF THE FIGURE MEASUREMENTS IN QDUMVC	EAM14450
	QDUMVC(LSENS)=SXFXF	EAM14460
C	RETURN TO SIGMA 5 TO FILTER FIGURE ERROR DATA	EAM14470
C	IF MODVQ(7)=1 CALCULATE THE FIGURE ERROR AFTER EVERY NMEASQ	EAM14480

C	MEASUREMENTS	EAM14490
	IGO=MODVQ(7)	EAM14500
	GO TO(2356,5),IGO	EAM14510
2356	CONTINUE	EAM14520
	NFLGA=8	EAM14530
	GO TO(2357,2354),IGOA	EAM14540
2354	RETURN	EAM14550
C		EAM14560
C*****EAM SOFTWARE TEST CODING*****		EAM14570
2357	CALL MAINA(6)	EAM14580
	GO TO 5	EAM14590
C*****EAM SOFTWARE TEST CODING*****		EAM14600
C	ENTRY POINT TO EAMCS AT COMPLETION OF FIGURE ERROR COMPUTATIONS	EAM14610
5	CONTINUE	EAM14620
	MSENS=MSENS-1	EAM14630
	ISENS=3	EAM14640
C	TERMINATE THE MEASUREMENT MODE IF THE FIGURE ERROR HAS BEEN	EAM14650
C	OBTAINED FOR ALL NO POSITIONS	EAM14660
	IF(MSENS)2352,2352,2303	EAM14670
2352	ISENS=1	EAM14680
C		EAM14690
C	RETURN TO SIGMA 5 TO CALCULATE NEW FIGURE CONTROL	EAM14700
15	NFLGA=17	EAM14710
	NFLGC=1	EAM14720
	GO TO (2358,2359),IGOA	EAM14730
2359	RETURN	EAM14740
C		EAM14750
18	NFLGA=18	EAM14760
	RETURN	EAM14770
C		EAM14780
C*****EAM SOFTWARE TEST CODING*****		EAM14790
2358	CALL MAINA(7)	EAM14800
	CALL MAINA(5)	EAM14810
C*****EAM SOFTWARE TEST CODING*****		EAM14820
C		EAM14830
C	ENTRY POINT TO EAMCS AT COMPLETION OF CONTROL COMPUTATION	EAM14840
6	CONTINUE	EAM14850
C		EAM14860
C	PRINT OUTPUT DATA ON THE REMOTE IO DEVICE EVERY NTYO TIMES THE	EAM14870
C	CONTROL IS CALCULATED	EAM14880
	GO TO(2361,2362),NFLGC	EAM14890
2362	QDUMVA(2)=ITIMS*QDT	EAM14900
C	CALCULATE CONTROL SYSTEM OPERATING TIME	EAM14910
	CALL TYPCON(9)	EAM14920
2361	CONTINUE	EAM14930
C		EAM14940
C	RELEASE ACTUATORS	EAM14950
	CALL ACTCMD(6)	EAM14960
C		EAM14970
C	SET POSITION ACTUATOR COUNTER TO NCVELO	EAM14980
	ICVEL=NCVELO	EAM14990
C	FREEZE ACTUATORS AFTER NCVELO CONTROL CYCLES	EAM15000
2390	IGO=MODVQ(6)	EAM15010
	GO TO(2396,2397),IGO	EAM15020
2397	ICVEL=ICVEL-1	EAM15030
	IF(ICVEL)2399,2399,2396	EAM15040
C	FREEZE ACTUATORS	EAM15050
2399	CALL ACTCMD(5)	EAM15060
C		EAM15070
C	TRANSFER COMMANDS TO ACTUATOR CONTROL SYSTEM	EAM15080
2396	NFLGA=9	EAM15090
	GO TO (2393,2394),IGOA	EAM15100

2394 RETURN	EAM15110
C	EAM15120
2393 CALL ACTCMD(3)	EAM15130
16 CONTINUE	EAM15140
C	EAM15150
C STORE IMPORTANT PARAMETERS IN DUMVA FOR DIAGNOSTIC PURPOSES	EAM15160
C EVERY CYCLE	EAM15170
QDUMVA(7)=LSENS	EAM15180
QDUMVA(8)=JMEAS	EAM15190
QDUMVA(9)=JWAIT	EAM15200
QDUMVA(10)=JSSENS	EAM15210
QDUMVA(11)=ISENS	EAM15220
C	EAM15230
C TRANSFER EAM SYSTEM VARIABLES TO SIMULATION SYSTEM ROUTINES FOR	EAM15240
C FOR DISPLAY AND FURTHER PROCESSING IF MODVQ(8)=1	EAM15250
IGO=MODVQ(8)	EAM15260
GO TO(2391,2392),IGO	EAM15270
2391 NFLGA=10	EAM15280
NFLGB=LSENS	EAM15290
GO TO(2392,2395),IGO	EAM15300
2395 RETURN	EAM15310
C*****EAM SOFTWARE TEST CODING*****EAM15320	EAM15320
2392 CALL FSMDL(5,LSENS)	EAM15330
C*****EAM SOFTWARE TEST CODING*****EAM15340	EAM15340
C	EAM15350
17 IGO=MODVQ(9)	EAM15360
GO TO(2400,2700),IGO	EAM15370
C	EAM15380
C INTERRUPT FIGURE CONTROL, FREEZE ACTUATORS AND TRANSFER TO	EAM15390
C TO TYPCON FOR INSTRUCTIONS IF MODEQ=3	EAM15400
2700 GO TO(2703,2703,2702),MODEQ	EAM15410
C STORE CURRENT TIME	EAM15420
2702 CALL REALT(STA)	EAM15430
C FREEZE ACTUATOR OUTPUTS	EAM15440
CALL ACTCMD(5)	EAM15450
CALL ACTCMD(3)	EAM15460
C TRANSFER TO TYPCON FOR FURTHER INSTRUCTIONS	EAM15470
NFLGA=21	EAM15480
CALL TYPCON(2)	EAM15490
C RESET TIME	EAM15500
19 CALL REALT(T)	EAM15510
TSTORE=T-STA+TSTORE	EAM15520
C	EAM15530
C CYCLE TIME CONTROL	EAM15540
2703 CALL REALT(T)	EAM15550
IF(T+QDTE-TSTORE)2703,2701,2701	EAM15560
2701 TSTORE=TSTORE+QDT	EAM15570
GO TO 2400	EAM15580
C	EAM15590
C INITIALIZE EAMCS WITHOUT INTERROGATING TYPCON	EAM15600
8 GO TO 2230	EAM15610
C	EAM15620
END	EAM15630

	SUBROUTINE FIGSEN(NENTRY,I)	EAM15640
C		EAM15650
C	SUBROUTINE TO MEASURE THE FIGURE ERROR XFV(I) AT A DISCRETE POINT	EAM15660
C	COORDINATES XFSV(I),YFSV(I) ON THE REFLECTING SURFACE OF THE MIRROR	EAM15670
C		EAM15680
C	SIGMA 2 DIMENSION STATEMENTS START	EAM15690
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM15700
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM15710
	2 QASV(3)	EAM15720
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM15730
	1 MSEQVQ,NSSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSENS,	EAM15740
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM15750
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM15760
C	SIGMA 2 DIMENSION STATEMENTS END	EAM15770
C		EAM15780
	GO TO(1,2,3,4),NENTRY	EAM15790
C		EAM15800
C	INPUT DATA AND INITIALIZATION	EAM15810
1	RETURN	EAM15820
C		EAM15830
C	TRANSFER THE MEASUREMENT POSITION COORDINATES TO THE IMAGE	EAM15840
C	DISSECTOR	EAM15850
C	SKIP POSITION COORDINATE TRANSFER IF MODV(8)=1	EAM15860
2	IGO=MODVQ(8)	EAM15870
	GO TO(2201,2202),IGO	EAM15880
2202	X=QXFSV(I)	EAM15890
	Y=QYFSV(I)	EAM15900
C	X AND Y ARE THE COORDINATES OF THE MEASUREMENT POSITION	EAM15910
C	INSERT DTOA SOFTWARE HERE TO POSITION IMAGE DISSECTOR	EAM15920
2201	NFLGA=6	EAM15930
	RETURN	EAM15940
C		EAM15950
C	SAMPLE THE FIGURE SENSOR PHASE DETECTOR FILTER OUTPUT	EAM15960
C	RETURN TO SIGMA 5 TO MODEL FIGURE SENSOR IF MODVQ(8)=1	EAM15970
3	IGO=MODVQ(8)	EAM15980
	GO TO(2301,2302),IGO	EAM15990
2301	NFLGA=11	EAM16000
	NFLGB=1	EAM16010
	IGO=MODVQ(12)	EAM16020
	GO TO (2304,2303),IGO	EAM16030
2303	RETURN	EAM16040
C*****	EAM SOFTWARE TEST CODING*****	EAM16050
2304	CALL FSMDL(3,I)	EAM16060
	GO TO 4	EAM16070
C*****	EAM SOFTWARE TEST CODING*****	EAM16080
C		EAM16090
2302	CONTINUE	EAM16100
C	QXF IS THE FIGURE SENSOR PHASE DETECTOR FILTER OUTPUT	EAM16110
C	INSERT ATOD SOFTWARE HERE TO INTERROGATE FIGURE SENSOR	EAM16120
4	CONTINUE	EAM16130
	NFLGA=6	EAM16140
	RETURN	EAM16150
C		EAM16160
	END	EAM16170

	SUBROUTINE FSMDL(NENTRY,IENTRY)	EAM16180
C		EAM16190
C	SIMPLIFIED FIGURE SENSOR MODEL TO TEST THE EAM SOFTWARE PACKAGE	EAM16200
C	MODEL SIMULATES THE FIGURE SENSOR FOR INPUT ERRORS IN THE RANGE	EAM16210
C	FROM -3/4 TO +3/4 WAVELENGTHS	EAM16220
C		EAM16230
C	SIGMA 5 TYPE A DIMENSION STATEMENTS START	EAM16240
	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM16250
	1 YFSV(20),XFRV(20),DOMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM16260
	2 GAINM(1600),ASV(3)	EAM16270
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DOMV,UFAV,	EAM16280
	1 DUMVA,GAINV,GAINM,ASV	EAM16290
C		EAM16300
	DIMENSION LACTV(20)	EAM16310
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM16320
	1 NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM16330
C		EAM16340
	DIMENSION AM(400),AIM(400)	EAM16350
	COMMON/BLKMFC/AM,AIM	EAM16360
C		EAM16370
	COMMON/BLKT/T,DT,DTH,DTPLT,DTNOIS,TPhi,TPRNT,TEND	EAM16380
C		EAM16390
	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM16400
	1 JCV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM16410
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM16420
	3 MODV(20)	EAM16430
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM16440
	1 NCXV,NCPV,NICPV,JCV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM16450
	2 NMCXV,NMCPV,NMICPV,MODV	EAM16460
C		EAM16470
	DIMENSION XV(50),NAMV(50),DUMV(20),DUMM(400),PARV(50),IPARV(50),	EAM16480
	1 SXV(50),SPARV(50),ISPARV(50),IDUMV(20)	EAM16490
	COMMON/BLKSIM/XV,NAMV,DUMV,DUMM,PARV,IPARV,SXV,SPARV,ISPARV,	EAM16500
	1 IDUMV	EAM16510
C		EAM16520
	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM16530
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM16540
C		EAM16550
C	SIGMA 5 TYPE A DIMENSION STATEMENTS END	EAM16560
C		EAM16570
	SIGMA 2 DIMENSION STATEMENTS START	EAM16580
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM16590
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM16600
	2 QASV(3)	EAM16610
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM16620
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENQ,NTIMSQ,LSENS,	EAM16630
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM16640
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM16650
C		EAM16660
C	SIGMA 2 DIMENSION STATEMENTS END	EAM16670
	1000 FORMAT(6H FSMDL)	EAM16680
	1001 FORMAT(10X,5HFSPHI,10X,5HFSGAM,/,2F15.6)	EAM16690
C		EAM16700
	I=IENTRY	EAM16710
	GO TO(1,2,3,4,5,6),NENTRY	EAM16720
C		EAM16730
	INPUT DATA	EAM16740
C		
1	PRINT 1000	

C	READ IN DATA FOR FIGURE SENSOR NOISE MODEL	EAM16750
	CALL RANDPD(2,FSNSIG,FSTFLT,DA,DA,DA,DA,DA,4)	EAM16760
	CALL IRANDP(1,IRAND,IA,IA,IA,IA,IA,IA,4)	EAM16770
	IB=N+3	EAM16780
	IRANDS=IRAND	EAM16790
	GAINV(4)=FSTFLT	EAM16800
	GAINV(6)=FSNSIG	EAM16810
	RETURN	EAM16820
C		EAM16830
C	INITIALIZATION	EAM16840
2	CONTINUE	EAM16850
	IRAND=IRANDS	EAM16860
	FSNSIG=GAINV(6)	EAM16870
	FSTFLT=GAINV(4)	EAM16880
	FSFLT0=0.0	EAM16890
	FSNOIS=0.0	EAM16900
	FSPDO=0.0	EAM16910
	NNOIS=0	EAM16920
	SSNOIS=0.0	EAM16930
C	CALCULATE FIGURE SENSOR FILTER PARAMETERS	EAM16940
	IF(FSTFLT)2005,2005,2004	EAM16950
2004	DA=-DT/FSTFLT	EAM16960
	FSPHI=EXP(DA)	EAM16970
	GO TO 2006	EAM16980
2005	FSPHI=0.0	EAM16990
2006	FSGAM=1.0-FSPHI	EAM17000
	PRINT 1001,FSPHI,FSGAM	EAM17010
	RETURN	EAM17020
C		EAM17030
C	SAMPLE FIGURE SENSOR FILTER OUTPUT	EAM17040
C	TRANSFER FIGURE ERROR TO SIGMA 2 SOFTWARE	EAM17050
3	QXF=FSFLT0	EAM17060
	DOMV(14)=FSFLT0	EAM17070
	DOMV(12)=XFAV(I)	EAM17080
	RETURN	EAM17090
C		EAM17100
C	CALL TO FSMDL AT THE END OF EACH FIGURE MEASUREMENT	EAM17110
4	CONTINUE	EAM17120
C	CALCULATE THE FIGURE SENSOR ERROR	EAM17130
	DOMV(15)=DOMV(4)-XFAV(I)	EAM17140
	RETURN	EAM17150
C		EAM17160
C	CALL TO FSMDL EVERY DT	EAM17170
5	CONTINUE	EAM17180
C	CALCULATE NEW NOISE INPUT	EAM17190
	CALL GAUSS(IRAND,FSNSIG,0.0,FSNOIS)	EAM17200
C	ADD NOISE TO THE ACTUAL VALUE OF FIGURE ERROR	EAM17210
	DA=XFAV(I)+FSNOIS	EAM17220
C	CALCULATE THE STANDARD DEVIATION OF THE FIGURE SENSOR NOISE	EAM17230
	SSNOIS=SSNOIS+FSNOIS*FSNOIS	EAM17240
	NNOIS=NNOIS+1	EAM17250
	DOMV(17)=SQRT(SSNOIS/NNOIS)	EAM17260
	DOMV(18)=FSNOIS	EAM17270
C	USE A LINEAR FIGURE SENSOR MODEL IF MODV(3)=1	EAM17280
	IGO=MODV(3)	EAM17290
	GO TO(2000,2003),IGO	EAM17300
C	CALCULATE PHASE DETECTOR OUTPUT	EAM17310

2003	IF(DA*SGN(DA)-0.25)2000,2000,2001	EAM17320
2001	FSPDO=DA-0.50*SGN(DA)	EAM17330
	GO TO 2002	EAM17340
2000	FSPDO=DA	EAM17350
C	FILTER THE PHASE DETECTOR OUTPUT	EAM17360
2002	FSFLTO=FSPHI*FSFLTO+FSGAM*FSPDO	EAM17370
C	GENERATE XV FOR PLOTTING RESULTS	EAM17380
	IGO=MODV(7)	EAM17390
	GO TO(2200,2201),IGO	EAM17400
C	FIGURE SENSOR DEVELOPMENT	EAM17410
C	XFACT	EAM17420
2200	XV(1)=DOMV(12)	EAM17430
C	XFMEAS	EAM17440
	XV(2)=DOMV(4)	EAM17450
C	FSERR	EAM17460
	XV(3)=DOMV(15)	EAM17470
C	FSOUT	EAM17480
	XV(4)=DOMV(14)	EAM17490
C	XFMN	EAM17500
	XV(5)=DOMV(1)	EAM17510
C	XFSIG	EAM17520
	XV(6)=DOMV(2)	EAM17530
C	AMBIG	EAM17540
	XV(7)=DOMV(3)	EAM17550
C	XFSW	EAM17560
	XV(8)=DOMV(5)	EAM17570
C	FSNOIS	EAM17580
	XV(9)=DOMV(18)	EAM17590
	RETURN	EAM17600
C	EAM CONTROL SYSTEM DEVELOPMENT	EAM17610
C	PINDEX	EAM17620
2201	XV(1)=DOMV(13)	EAM17630
C	RPINDEX	EAM17640
	XV(2)=DOMV(19)	EAM17650
C	FSPINDEX	EAM17660
	XV(3)=DOMV(16)	EAM17670
C	XFV(SEE STATEMENT 2301)	EAM17680
C	UFAV,UFV	EAM17690
	DO 2202 J=1,NR	EAM17700
	K=J+IR	EAM17710
	XV(K)=UFAV(J)	EAM17720
	L=K+NR	EAM17730
2202	XV(L)=UFV(J)	EAM17740
	DO 2203 J=7,11	EAM17750
2203	DOMV(J)=QDUMVA(J)	EAM17760
	RETURN	EAM17770
C		EAM17780
C	CALL TO FSMSL AT THE END OF EACH COMPLETE SET OF MEASUREMENTS	EAM17790
6	CONTINUE	EAM17800
C	CALCULATE AND STORE THE PERFORMANCE INDEX	EAM17810
	CALL PINDX(3,PINDEX,XFV)	EAM17820
	DOMV(13)=PINDEX	EAM17830
C	CALCULATE THE FIGURE SENSOR PERFORMANCE INDEX	EAM17840
	DOMV(16)=0.0	EAM17850
	DO 2300 J=1,N	EAM17860
	DA=XFV(J)-XFAV(J)	EAM17870
2300	DOMV(16)=DOMV(16)+DA*DA	EAM17880

	DOMV(16)=DOMV(16)/N	EAM17890
	DOMV(16)=SQRT(DOMV(16))	EAM17900
C	CALCULATE THE TRUE VALUE OF THE PERFORMANCE INDEX	EAM17910
	CALL PINDX(3,DOMV(19),XFAV)	EAM17920
	GO TO(2302,2301),IPLOT	EAM17930
	GO TO(2302,2301),IPLOT	EAM17930
C	STORE XFV FOR PLOTTING	EAM17940
2301	DO 2303 J=1,N	EAM17950
	K=J+3	EAM17960
2303	XV(K)=XFV(J)	EAM17970
2302	RETURN	EAM17980
C		EAM17990
	END	EAM18000

	SUBROUTINE MAINA(NENTRY)	EAM18010
C		EAM18020
C	SUPERVISORY PROGRAM FOR THE EXPERIMENTAL ACTIVE MIRROR	EAM18030
C	RESIDENT IN THE SIGMA 5 COMPUTER	EAM18040
C		EAM18050
C	SIGMA 5 TYPE B DIMENSION STATEMENTS START	EAM18060
	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM18070
1	YFSV(20),XFRV(20),DUMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM18080
2	GAINM(1600),ASV(3)	EAM18090
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DUMV,UFAV,	EAM18100
1	DUMVA,GAINV,GAINM,ASV	EAM18110
C		EAM18120
	DIMENSION LACTV(20)	EAM18130
	COMMON/BLKEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM18140
1	NTYPD,NPUNCH,NMAG,NSFNS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM18150
C		EAM18160
	DIMENSION AM(400),AIM(400)	EAM18170
	COMMON/BLKMFC/AM,AIM	EAM18180
C		EAM18190
	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM18200
1	JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM18210
2	CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM18220
3	MODV(20)	EAM18230
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM18240
1	NCXV,NCPV,NICPV,JCXV, JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM18250
2	NMCXV,NMCPV,NMICPV,MODV	EAM18260
C		EAM18270
	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM18280
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM18290
C		EAM18300
C	SIGMA 5 TYPE B DIMENSION STATEMENTS END	EAM18310
C		EAM18320
C	SIGMA 2 DIMENSION STATEMENTS START	EAM18330
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM18340
1	QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFVQ(20),	EAM18350
2	QASV(3)	EAM18360
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM18370
1	MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSENS,	EAM18380
2	NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM18390
3	QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFVQ, IDUMVQ,QASV,NCVELQ	EAM18400
C		EAM18410
C	SIGMA 2 DIMENSION STATEMENTS END	EAM18420
	DIMENSION DUMVC(20),MSEQV(20)	

C		EAM18430
	1000 FORMAT(6H MAINA)	EAM18440
	1001 FORMAT(/,37H EAM FIGURE CONTROL SYSTEM PARAMETERS,/)	EAM18450
	1002 FORMAT(/,35H EAM FIGURE ERROR FILTER PARAMETERS,/)	EAM18460
	1003 FORMAT(/,20H HARDWARE MODEL DATA,/)	EAM18470
	1004 FORMAT(/,23H PERFORMANCE INDEX DATA,/)	EAM18480
	1011 FORMAT(1H1)	EAM18490
	1201 FORMAT(10X,5HTSENS,10X,5HTWAIT,11X,4HTPOS,10X,5HTMINT,10X,5HTMEAS,	EAM18500
	1 4X,11HTMEAS/TSENS)	EAM18510
	1400 FORMAT(10X,5HNSENS,10X,5HNWAIT,11X,4HNPOS,10X,5HNMINT,10X,5HNMEAS,	EAM18520
	1 9X,6HNFSENS,10X,5HNTIMS)	EAM18530
	1401 FORMAT(7I15)	EAM18540
	1402 FORMAT(11X,4HGAIN,/,F15.6)	EAM18550
C		EAM18560
	GO TO(1,2,3,3,5,6,7),NENTRY	EAM18570
C		EAM18580
C	INPUT DATA	EAM18590
1	PRINT 1000	EAM18600
C		EAM18610
C	READ DATA FOR THE SIGMA 2 SOFTWARE	EAM18620
	PRINT 1001	EAM18630
	CALL IRANDP(6,NTIMSO,NWAIT,NPOS,NMINT,NMEAS,NTYO,IA,4)	EAM18640
	CALL RANDPD(6,DT,DTE,GAINV(1),QGA,QGB,UFGMAX,DA,4)	EAM18650
	PRINT 1002	EAM18660
	CALL RANDPD(2,SIGLIM,SLPMN,DA,DA,DA,DA,DA,4)	EAM18670
	CALL IMXRNP(MSEQV,1,N,4)	EAM18680
	PRINT 1003	EAM18690
	CALL FSMDL(1,I)	EAM18700
	CALL ACTMDL(1,I)	EAM18710
	CALL MIRMDL(1,I)	EAM18720
	CALL MIRMDL(2,I)	EAM18730
C	READ DATA FOR MAINB AND MAINC	EAM18740
	CALL MAINB(1)	EAM18750
	CALL MAINC(1)	EAM18760
C		EAM18770
	PRINT 1004	EAM18780
	CALL PINDX(1,PINDEX,XFV)	EAM18790
	FSCALE=FSCALV(1)	EAM18800
	GO TO 2206	EAM18810
C		EAM18820
C	INITIALIZATION	EAM18830
2	CONTINUE	EAM18840
	DUMV(2)=NWAIT*DT	EAM18850
	DUMV(3)=NPOS*DT	EAM18860
	DUMV(4)=NMINT*DT	EAM18870
	DUMV(5)=DUMV(2)+N*(DUMV(3)+NMEAS*DUMV(4))	EAM18880
	NFSENS=NWAIT+N*(NPOS+NMEAS*NMINT)	EAM18890
	NSENS=NFSENS	EAM18900
	DUMV(1)=NSENS*DT	EAM18910
	TSENS=DUMV(1)	EAM18920
	DUMV(6)=DUMV(5)/DUMV(1)	EAM18930
C	PRINT OUT THE CONTROL SYSTEM TIMING CHARACTERISTICS	EAM18940
	PRINT 1201	EAM18950
	CALL MXRNP(DUMV,1,6,3)	EAM18960
C	MULTIPLY GAIN BY TSENS TO MAKE THE DYNAMIC RESPONSE INDEPENDENT	EAM18970
C	OF TSENS	EAM18980
	GAINV(1)=GAINV(1)*TSENS	EAM18990
C	DIGITAL FILTER INITIALIZATION	EAM19000
	AMBIG=0.0	EAM19010
	SXFMN=0.0	EAM19020
	XFLAST=0.0	EAM19030

	XFMN=0.0	EAM19040
	XFSIG=0.0	EAM19050
	XFSW=0.0	EAM19060
C		EAM19070
C	CONTROL SYSTEM INITIALIZATION	EAM19080
	DO 2201 I=1,N	EAM19090
	XFV(I)=0.0	EAM19100
	XFRV(I)=0.0	EAM19110
	DUMV(I)=0.0	EAM19120
	DUMVA(I)=0.0	EAM19130
2201	DUMVC(I)=0.0	EAM19140
	CALL FSMDL(6,I)	EAM19150
	DO 2203 I=1,20	EAM19160
2203	DUMV(I)=0.0	EAM19170
C		EAM19180
C	PRINT IMPORTANT PARAMETER VALUES FOR CURRENT RUN	EAM19190
	PRINT 1400	EAM19200
	PRINT 1401,NFSSENS,NWAIT,NPOS,NMINT,NMEAS,NFSSENS,NTIMS	EAM19210
	PRINT 1402,GAINV(1)	EAM19220
C		EAM19230
C		EAM19240
C	INITIALIZE HARDWARE MODELS	EAM19250
	CALL ACTMDL(2,I)	EAM19260
	CALL MIRMDL(2,I)	EAM19270
C	INITIALIZE THE PERFORMANCE INDEX GENERATOR	EAM19280
	CALL PINDX(2,PINDEX,XFV)	EAM19290
C		EAM19300
C	TRANSFER DATA TO THE SIGMA 2 SOFTWARE	EAM19310
2206	QDT=DT	EAM19320
	QDTE=DTE	EAM19330
	QUFMAX=UFMAX*ASCALV(1)	EAM19340
	NQ=N	EAM19350
	NRQ=NR	EAM19360
	DO 2204 I=1,N	EAM19370
	QXFSV(I)=XFSV(I)*PSCALE	EAM19380
	QYFSV(I)=YFSV(I)*PSCALE	EAM19390
	MSEQVQ(I)=MSEQV(I)	EAM19400
	QDUMVA(I)=0.0	EAM19410
	QDUMVB(I)=0.0	EAM19420
2204	QDUMVC(I)=0.0	EAM19430
	DO 2205 I=1,NR	EAM19440
	UFV(I)=0.0	EAM19450
	UFAV(I)=0.0	EAM19460
	QUFV(I)=0.0	EAM19470
2205	QUFAV(I)=0.0	EAM19480
	NSENSQ=NSENS	EAM19490
	NWAITQ=NWAIT	EAM19500
	NPOSQ=NPOS	EAM19510
	NMINTQ=NMINT	EAM19520
	NMEASQ=NMEAS	EAM19530
	NFSENQ=NFSSENS	EAM19540
	NTIMS=NTIMSO	EAM19550
	NTIMSQ=NTIMS	EAM19560
	ITYO=0	EAM19570
	FSCALE=FSCALV(1)	EAM19575
	RETURN	EAM19580
C		EAM19590
C	OPERATION	EAM19600
3	RETURN	EAM19610
C		EAM19620
C	CALCULATE FIGURE CONTROLS	EAM19630
C		EAM19640

C	CALCULATE THE INITIAL ALIGNMENT CONTROLS IF MODV(4)=1	EAM19650
5	IGO=MODV(4)	EAM19660
	GO TO(2501,2502),IGO	EAM19670
2501	CALL MAINC(3)	EAM19680
	RETURN	EAM19690
C		EAM19700
C	GENERATE XFRV	EAM19710
2502	CONTINUE	EAM19720
	J=0	EAM19730
	DO 2506 I=1,N	EAM19740
	GO TO (2508,2509,2509),MODOP	EAM19750
2508	IF(LACTV(I))2507,2506,2507	EAM19760
2507	J=J+1	EAM19770
	XFRV(J)=XFV(I)	EAM19780
	GO TO 2506	EAM19790
2509	XFRV(I)=XFV(I)	EAM19800
2506	CONTINUE	EAM19810
C		EAM19820
C	DUMVC=GAINM*XFRV	EAM19830
	CALL MPRD(GAINM,XFRV,DUMVC,NR,NRA,0,0,1)	EAM19840
C	DUMVC=DUMVC*GAINV(1)	EAM19850
	DO 2510 I=1,NR	EAM19860
2510	DUMVC(I)=DUMVC(I)*GAINV(1)	EAM19870
C	INTEGRAL COMPENSATION	EAM19880
C	UFV=DUMVC+UFV	EAM19890
	DO 2520 I=1,NR	EAM19900
	UFV(I)=DUMVC(I)+UFV(I)	EAM19910
2520	QUFV(I)=UFV(I)*ASCALV(I)	EAM19920
C*****	EAM SOFTWARE TEST CODING*****EAM19930	EAM19930
C	STORE IMPORTANT PARAMETERS IN FSMDL AT TERMINATION OF	EAM19940
C	MEASUREMENT SEQUENCE	EAM19950
	CALL FSMDL(6,LSENS)	EAM19960
C*****	EAM SOFTWARE TEST CODING*****EAM19970	EAM19970
C	PRINT OUTPUT DATA ON REMOTE IO DEVICE EVERY NTYO TIMES	EAM19980
C	THE CONTROL IS COMPUTED	EAM19990
	ITYO=ITYO-1	EAM20000
	IF(ITYO)2521,2521,2522	EAM20010
2521	ITYO=NTYO	EAM20020
	NFLGC=2	EAM20030
	CALL PINDX(3,QDUMVA(1),XFV)	EAM20040
2522	RETURN	EAM20050
C		EAM20060
C	DIGITAL FILTER FOR FIGURE SENSOR OUTPUTS	EAM20070
C		EAM20080
6	GO TO 2703	EAM20090
2702	IF(XFSIG-SIGLIM)2332,2332,2331	EAM20100
C	FIGURE ERROR COMPUTATION IF XFSIG IS LESS THAN SIGLIM	EAM20110
C	CORRECT FIGURE ERROR FOR AMBIGUITY	EAM20120
2332	XFV(LSENS)=XFMN+AMBIG	EAM20130
C	CALCULATE THE NEAREST SWITCHING BOUNDARY	EAM20140
	XFSW=0.25*SGN(XFLAST)	EAM20150
C	STORE CURRENT VALUE OF XF	EAM20160
	XFLAST=XFV(LSENS)	EAM20170
	GO TO 2360	EAM20180
C	FIGURE ERROR COMPUTATION IF XFSIG IS GREATER THAN SIGLIM	EAM20190
2331	XFV(LSENS)=XFSW-SLPMN*XFMN	EAM20200
C	CALCULATE AMBIGUITY FACTOR	EAM20210
	IF(SGN(XFMN*SXFMN))2353,2353,2360	EAM20220
2353	AMBIG=AMBIG+SGN(SXFMN)*0.50	EAM20230
2360	CONTINUE	EAM20240
C	STORE IMPORTANT PARAMETERS IN DUMV FOR FURTHER USE IN OTHER ROUTINE	EAM20250

DUMV(1)=XFMN	EAM20260
DUMV(2)=XFSIG	EAM20270
DUMV(3)=AMBIG	EAM20280
DUMV(4)=XFV(LSENS)	EAM20290
DUMV(5)=XFSW	EAM20300
DUMV(6)=XFLAST	EAM20310
C STORE LAST VALUE OF XFMN	EAM20320
SXFMN=XFMN	EAM20330
C*****EAM SOFTWARE TEST CODING*****	EAM20340
CALL FSMDL(4,LSENS)	EAM20350
C*****EAM SOFTWARE TEST CODING*****	EAM20360
C MODV(7)=1 FOR FIGURE SENSOR TEST, 2 FOR MIRROR FIGURE CONTROL	MEAM20370
IGO=MODV(7)	EAM20380
GO TO(2704,2705),IGO	EAM20390
2704 RETURN	EAM20400
C	EAM20410
C CALCULATE THE FIGURE ERRORS AT THE TERMINATION OF ALL MEASUREMENTS	EAM20420
C IF THE MIRROR FIGURE CONTROL MODE IS SELECTED I.E. MODV(7)=2	EAM20430
7 MSENS=N+1	EAM20440
IGO=MODV(7)	EAM20450
GO TO(2700,2705),IGO	EAM20460
2700 RETURN	EAM20470
C	EAM20480
2705 MSENS=MSENS-1	EAM20490
IF(MSENS)2700,2700,2701	EAM20500
2701 LSENS=MSEQV(MSENS)	EAM20510
C CALCULATE XFMN AND XFSIG AT ALL THE MEASUREMENT POINTS	EAM20520
2703 XFMN=QDUMVB(LSENS)*FSCALE	EAM20530
XFSIG=QDUMVC(LSENS)*FSCALE*FSCALE	EAM20540
XFSIG=XFSIG-XFMN*XFMN	EAM20550
XFMN=XFMN/NMEAS	EAM20560
XFSIG=XFSIG/NMEAS	EAM20570
XFSIG=SQRT(XFSIG)	EAM20580
C TRANSFER TO THE DIGITAL FIGURE ERROR FILTER	EAM20590
C SKIP FILTERING IF MODV(4)=1	EAM20600
IGO=MODV(4)	EAM20610
GO TO(2360,2702),IGO	EAM20620
C	EAM20630
END	EAM20640

	SUBROUTINE MAINB(NENTRY)	EAM20650
C		EAM20660
C	SUBROUTINE FOR TYPEWRITER CONTROL OF THE EXPERIMENTAL ACTIVE	EAM20670
C	MIRROR	EAM20680
C		EAM20690
C	SIGMA 5 TYPE C DIMENSION STATEMENTS START	EAM20700
	DIMENSION XV(1813)	EAM20710
	COMMON/BLKEAM/XV	EAM20720
	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM20730
	1 YFSV(20),XFRV(20),DUMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM20740
	2 GAINM(1600),ASV(3)	EAM20750
	EQUIVALENCE (XV(1),XFV(1)),(XV(21),UFV(1)),(XV(41),ASCALV(1)),	EAM20760
	1 (XV(61),FSCALV(1)),(XV(81),XFSV(1)),(XV(101),YFSV(1)),	EAM20770
	2 (XV(121),XFRV(1)),(XV(141),DUMV(1)),(XV(161),UFAV(1)),	EAM20780
	3 (XV(181),DUMVA(1)),(XV(201),GAINV(1)),(XV(211),GAINM(1)),	EAM20790
	4 (XV(1811),ASV(1))	EAM20800
C		EAM20810
	DIMENSION LACTV(20)	EAM20820
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM20830
	1 NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM20840
C		EAM20850
	DIMENSION AM(400),AIM(400)	EAM20860
	COMMON/BLKMFC/AM,AIM	EAM20870
C		EAM20880
	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM20890
	1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM20900
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM20910
	3 MODV(20)	EAM20920
	COMMON/BLKIV/IAV,NIAV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM20930
	1 NCXV,NCPV,NICPV,JCXV,JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM20940
	2 NMCXV,NMCPV,NMICPV,MODV	EAM20950
C		EAM20960
	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM20970
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM20980
C		EAM20990
C	SIGMA 5 TYPE C DIMENSION STATEMENTS END	EAM21000
C		EAM21010
	SIGMA 2 DIMENSION STATEMENTS START	EAM21020
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM21030
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM21040
	2 QASV(3)	EAM21050
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM21060
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSENS,	EAM21070
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM21080
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM21090
C		EAM21100
C	SIGMA 2 DIMENSION STATEMENTS END	EAM21110
		EAM21120
C	DIMENSION NAMV(17)	EAM21130
	DATA NAMV(1),NAMV(2),NAMV(3),NAMV(4),NAMV(5),NAMV(6),NAMV(7),	EAM21140
	1 NAMV(8),NAMV(9),NAMV(10),NAMV(11),NAMV(12),NAMV(13),NAMV(14),	EAM21150
	2 NAMV(15),NAMV(16),NAMV(17)	EAM21160
	3 /4HXV ,4HAM ,4HAIM ,4HXFV ,4HXFRV,4HUFV ,4HUFV,4HASCV,	EAM21170
	4 4HFSCV,4HXFSV,4HYFSV,4HDUMV,4HGANM,4HGANV,4HLACV,4HMDVQ,4HDEND/	EAM21180
C		EAM21190
	1000 FORMAT(6H MAINB)	EAM21200
C		EAM21210
	GO TO(1,2,3,4,5,6),NENTRY	

C		EAM21220
C	INPUT DATA	EAM21230
1	PRINT 1000	EAM21240
	NNAMV=17	EAM21250
	RETURN	EAM21260
C		EAM21270
C	INITIALIZATION	EAM21280
2	RETURN	EAM21290
C		EAM21300
C	CATALOG AND CHECK INPUT NAME FROM TYPCON	EAM21310
3	DO 2302 I=1,NNAMV	EAM21320
	IF(NFLGC-NAMV(I))2302,2301,2302	EAM21330
2301	LL=I	EAM21340
	GO TO 2303	EAM21350
2302	CONTINUE	EAM21360
C	SUBMITTED NAME NOT IN CATALOG	EAM21370
	NFLGC=3	EAM21380
	CALL MARK(1,24,4,4,3)	EAM21390
C	CALL TYPCON(4) AND RETURN THROUGH MAINB(3)	EAM21400
	RETURN	EAM21410
C		EAM21420
C	IDENTIFY THE NUMBER OF INDICES	EAM21430
2303	GO TO (2310,2320,2320,2310,2310,2310,2310,2310,2310,2310,	EAM21440
1	2310,2320,2310,2310,2310,2304),LL	EAM21450
C		EAM21460
C	REQUEST NEW MODE IF NFLGC=4HDEND	EAM21470
2304	NFLGC=4	EAM21480
	RETURN	EAM21490
C		EAM21500
C	RETURN TO SIGMA 2 TO REQUEST INDEX VALUES	EAM21510
2310	NFLGC=1	EAM21520
	CALL MARK(1,24,5,4,4)	EAM21530
	RETURN	EAM21540
C	CALL TYPCON(5) AND RETURN THROUGH MAINB(4)	EAM21550
C		EAM21560
2320	NFLGC=2	EAM21570
	CALL MARK(1,24,6,4,4)	EAM21580
	RETURN	EAM21590
C	CALL TYPCON(6) AND RETURN THROUGH MAINB(4)	EAM21600
C		EAM21610
C	MODIFY AND/OR EXTRACT THE VALUE OF THE INTERROGATED VARIABLE	EAM21620
4	V=QDUMVA(1)	EAM21630
	II=NFLGD	EAM21640
	JJ=NFLGE	EAM21650
	ICHNG=NFLGC	EAM21660
	GO TO (2401,2402,2403,2404,2405,2406,2407,2408,2409,2410,2411,	EAM21670
1	2412,2413,2414,2415,2416,2417),LL	EAM21680
2401	CALL CHNG(ICHNG,V,XV(II))	EAM21690
	V=XV(II)	EAM21700
	GO TO 2500	EAM21710
2402	CALL ELMA(ICHNG,AM,II,JJ,V,NR)	EAM21720
	CALL ELMA(2,AM,II,JJ,V,NR)	EAM21730
	GO TO 2500	EAM21740
2403	CALL ELMA(ICHNG,AIM,II,JJ,V,NR)	EAM21750
	CALL ELMA(2,AIM,II,JJ,V,NR)	EAM21760
	GO TO 2500	EAM21770
2404	CALL CHNG(ICHNG,V,XFV(II))	EAM21780

V=XFV(II)	EAM21790
GO TO 2500	EAM21800
2405 CALL CHNG(ICHNG,V,XFRV(II))	EAM21810
V=XFRV(II)	EAM21820
2406 CALL CHNG(ICHNG,V,UFV(II))	EAM21830
V=UFV(II)	EAM21840
GO TO 2500	EAM21850
2407 CALL CHNG(ICHNG,V,UFAV(II))	EAM21860
V=UFAV(II)	EAM21870
GO TO 2500	EAM21880
2408 CALL CHNG(ICHNG,V,ASCALV(II))	EAM21890
V=ASCALV(II)	EAM21900
GO TO 2500	EAM21910
2409 CALL CHNG(ICHNG,V,FSCALV(II))	EAM21920
V=FSCALV(II)	EAM21930
GO TO 2500	EAM21940
2410 CALL CHNG(ICHNG,V,XFSV(II))	EAM21950
V=XFSV(II)	EAM21960
GO TO 2500	EAM21970
2411 CALL CHNG(ICHNG,V,YFSV(II))	EAM21980
V=YFSV(II)	EAM21990
GO TO 2500	EAM22000
2412 CALL CHNG(ICHNG,V,DUMV(II))	EAM22010
V=DUMV(II)	EAM22020
GO TO 2500	EAM22030
2413 CALL ELMA(ICHNG,GAINM,II,JJ,V,NR)	EAM22040
CALL ELMA(2,GAINM,II,JJ,V,NR)	EAM22050
GO TO 2500	EAM22060
2414 CALL CHNG(ICHNG,V,GAINV(II))	EAM22070
V=GAINV(II)	EAM22080
GO TO 2500	EAM22090
2415 V=LACTV(II)	EAM22100
GO TO 2500	EAM22110
2416 V=MODVQ(II)	EAM22120
GO TO 2500	EAM22130
2417 RETURN	EAM22140
C	EAM22150
2500 QDUMVA(1)=V	EAM22160
RETURN	EAM22170
C	EAM22180
C CALCULATE AND TRANSFER THE VALUE OF THE PERFORMANCE INDEX	EAM22190
C TO THE SIGMA 2	EAM22200
5 CALL PINDX(3,QDUMVA(1),XFV)	EAM22210
RETURN	EAM22220
C RETURN TO TYPCON(3)	EAM22230
C	EAM22240
6 RETURN	EAM22250
END	EAM22260

	SUBROUTINE MAINC(NENTRY)	EAM22270
C		EAM22280
C	SUBROUTINE TO CALCULATE INITIAL ALIGNMENT CONTROLS	EAM22290
C		EAM22300
C	SIGMA 5 TYPE B DIMENSION STATEMENTS START	EAM22310
	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM22320
	1 YFSV(20),XFRV(20),DUMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM22330
	2 GAINM(1600),ASV(3)	EAM22340
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DUMV,UFAV,	EAM22350
	1 DUMVA,GAINV,GAINM,ASV	EAM22360
C		EAM22370
	DIMENSION LACTV(20)	EAM22380
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM22390
	1 NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM22400
C		EAM22410
	DIMENSION AM(400),AIM(400)	EAM22420
	COMMON/BLKMFC/AM,AIM	EAM22430
C		EAM22440
	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM22450
	1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM22460
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM22470
	3 MODV(20)	EAM22480
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM22490
	1 NCXV,NCPV,NICPV,JCXV,JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM22500
	2 NMCXV,NMCPV,NMICPV,MODV	EAM22510
C		EAM22520
	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM22530
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM22540
C		EAM22550
C	SIGMA 5 TYPE B DIMENSION STATEMENTS END	EAM22560
C		EAM22570
	SIGMA 2 DIMENSION STATEMENTS START	EAM22580
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM22590
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM22600
	2 QASV(3)	EAM22610
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM22620
	1 MSEQVQ,NSSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSENS,	EAM22630
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM22640
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM22650
C		EAM22660
C	SIGMA 2 DIMENSION STATEMENTS END	EAM22670
	DIMENSION NHCV(2),NICV(2),SXFSV(9),SYFSV(9),SUFV(9),DELUV(2),	EAM22680
	1 SASV(3),LREFAV(9),SMXV(9),SMYV(9)	EAM22690
C		EAM22700
	1000 FORMAT(' MAINC')	EAM22710
	1001 FORMAT(/,' INITIAL ALIGNMENT CONTROL SYSTEM DATA')	EAM22720
	1002 FORMAT(/,' AMBIGUITY SENSOR MODEL DATA')	EAM22730
	1003 FORMAT(/,' SLEW CONTROL SYSTEM DATA')	EAM22740
	1004 FORMAT(/,' TILT CONTROL SYSTEM DATA')	EAM22750
	1005 FORMAT(/,' POSITION ACTUATOR DATA')	EAM22760
C		EAM22770
	GO TO(1,2,3),NENTRY	EAM22780
C		EAM22790
	INPUT DATA	EAM22800
1	PRINT 1000	EAM22810
	PRINT 1001	EAM22820
C		EAM22830
	READ ACTUATOR INDICES	
	CALL IMXRNP(LREFAV,1,9,4)	

C	READ ACTUATOR POSITIONS	EAM22840
	CALL MXRNP(SMXV,1,9,4)	EAM22850
	CALL MXRNP(SMYV,1,9,4)	EAM22860
C	READ AMBIGUITY SENSOR MODEL DATA	EAM22870
	PRINT 1002	EAM22880
	CALL RANDPD(3,BSMP,BSDP,DELU,DA,DA,DA,4)	EAM22890
C	READ SLEW CONTROL SYSTEM DATA	EAM22900
	PRINT 1003	EAM22910
	CALL IRANDP(2,NHM,NIM,IA,IA,IA,IA,4)	EAM22920
C	READ TILT CONTROL SYSTEM DATA	EAM22930
	PRINT 1004	EAM22940
	CALL IRANDP(2,NTILT,NCTILT,IA,IA,IA,IA,4)	EAM22950
	CALL RANDPD(1,GTILT,DA,DA,DA,DA,DA,4)	EAM22960
C	READ POSITION ACTUATOR DATA	EAM22970
	PRINT 1005	EAM22980
	CALL IRANDP(1,NCVEL,IA,IA,IA,IA,IA,4)	EAM22990
	NCVELQ=NCVEL	EAM23000
	INITL=1	EAM23010
	RETURN	EAM23020
C		EAM23030
C	INITIALIZATION	EAM23040
2	INITL=2	EAM23050
C	SET ALL CONTROLS TO ZERO	EAM23060
	DO 2100 I=1,NR	EAM23070
	UFV(I)=0.0	EAM23080
	QUFV(I)=0.0	EAM23090
2100	SUFV(I)=0.0	EAM23100
C	TILT CONTROL SYSTEM INITIALIZATION	EAM23110
	ITILT=0	EAM23120
	JTILT=0	EAM23130
	MDTILT=1	EAM23140
C	CALCULATE TILT MEASUREMENT POSITION INCREMENT	EAM23150
	DC=1.0/NTILT	EAM23160
C	SLEW CONTROL SYSTEM INITIALIZATION	EAM23170
	DO 2101 I=1,2	EAM23180
	NICV(I)=0	EAM23190
	NHCV(I)=0	EAM23200
	SASV(I)=0.0	EAM23210
2101	DELUV(I)=DELU	EAM23220
C	STORE N AND REPLACE IT WITH NEW VALUE	EAM23230
	NS=N	EAM23240
	N=9	EAM23250
C	STORE XFSV AND YFSV IN SXFSV AND SYFSV	EAM23260
	CALL MCPY(XFSV,SXFSV,1,9,0)	EAM23270
	CALL MCPY(YFSV,SYFSV,1,9,0)	EAM23280
C	SET THE FIRST NINE ELEMENTS OF XFSV AND YFSV TO THE ACTUATOR	EAM23290
C	POSITIONS	EAM23300
	CALL MCPY(SMXV,XFSV,1,9,0)	EAM23310
	CALL MCPY(SMYV,YFSV,1,9,0)	EAM23320
C		EAM23330
3	GO TO(2,2409),INITL	EAM23340
2409	IGO=MODV(5)	EAM23350
	GO TO(2410,2420),IGO	EAM23360
C		EAM23370
C	TILT CONTROL SYSTEM	EAM23380
2410	GO TO(2310,2320),MDTILT	EAM23390
C		EAM23400

C	CALCULATE CONTROLS TO SET REFERENCE POSITION ERRORS TO ZERO	EAM23410
2310	ITILT=ITILT+1	EAM23420
	IF(ITILT-NCTILT)2311,2311,2313	EAM23430
2311	J=1	EAM23440
	DO 2312 I=1,3	EAM23450
	K=LREFAV(J)	EAM23460
	SUFV(K)=SUFV(K)+GTILT*(XFV(K))	EAM23470
	UFV(K)=SUFV(K)	EAM23480
	QUFV(K)=UFV(K)*ASCALV(K)	EAM23490
2312	J=J+3	EAM23500
	RETURN	EAM23510
2313	MDTILT=2	EAM23520
	ITILT=1	EAM23530
	GO TO 2482	EAM23540
C		EAM23550
C	CALCULATE CONTROLS TO SET SECONDARY POSITION ERRORS TO ZERO	EAM23560
2320	ITILT=ITILT+1	EAM23570
	IF(ITILT-NCTILT)2481,2481,2482	EAM23580
2482	JTILT=JTILT+1	EAM23590
	IF(JTILT-NTILT)2483,2483,2484	EAM23600
C	INCREMENT MEASUREMENT POSITIONS	EAM23610
2483	DB=DB+DC	EAM23620
	DA=1.0-DB	EAM23630
	ITILT=0	EAM23640
	J=1	EAM23650
	DO 2480 I=1,3	EAM23660
	K=J+1	EAM23670
	L=J+2	EAM23680
	XFSV(K)=XFSV(J)*DA+SMXV(K)*DB	EAM23690
	YFSV(K)=YFSV(J)*DA+SMYV(K)*DB	EAM23700
	XFSV(L)=XFSV(J)*DA+SMXV(L)*DB	EAM23710
	YFSV(L)=YFSV(J)*DA+SMYV(L)*DB	EAM23720
2480	J=J+3	EAM23730
C	CALCULATE THE CONTROLS TO SET THE TILT TO ZERO	EAM23740
2481	J=1	EAM23750
	DO 2490 I=1,3	EAM23760
	K=J+1	EAM23770
	L=J+2	EAM23780
	IA=LREFAV(K)	EAM23790
	IB=LREFAV(L)	EAM23800
	SUFV(IA)=SUFV(IA)+GTILT*XFV(K)	EAM23810
	UFV(IA)=SUFV(IA)	EAM23820
	QUFV(IA)=UFV(IA)*ASCALV(IA)	EAM23830
	SUFV(IB)=SUFV(IB)+GTILT*XFV(L)	EAM23840
	UFV(IB)=SUFV(IB)	EAM23850
	QUFV(IB)=UFV(IB)*ASCALV(IB)	EAM23860
2490	J=J+3	EAM23870
	RETURN	EAM23880
C		EAM23890
C	AMBIGUITY SENSOR MODEL	EAM23900
2420	IGO=MODV(8)	EAM23910
	GO TO(2506,2505),IGO	EAM23920
2506	J=2	EAM23930
	DO 2504 I=1,3	EAM23940
	GO TO(2501,2502,2503),I	EAM23950
2501	DA=XFV(4)-XFV(1)	EAM23960
	GO TO 2504	EAM23970

2502	DA=XFV(7)-XFV(1)	EAM23980
	GO TO 2504	EAM23990
2503	DA=XFV(7)-XFV(4)	EAM24000
2504	ASV(I)=BSMP-BSDP*DA*DA	EAM24010
	GO TO 2507	EAM24020
C		EAM24030
C	TRANSFER THE AMBIGUITY SENSOR OUTPUTS TO THE SIGMA 5	EAM24040
2505	CALL MCPY(QASV,ASV,1,3,0)	EAM24050
C		EAM24060
C	SLEW CONTROL SYSTEM	EAM24070
2507	J=4	EAM24080
	DO 2700 I=1,2	EAM24090
	L=J+2	EAM24100
	IF(ASV(I)-SASV(I))2702,2702,2703	EAM24110
C	SUCCESSFUL ITERATION	EAM24120
2703	NICV(I)=NICV(I)+1	EAM24130
	IGO=1	EAM24140
C	UNSUCCESSFUL ITERATION	EAM24150
	IF(NICV(I)-NIM)2708,2708,2717	EAM24160
2717	DELUV(I)=0.0	EAM24170
	GO TO 2708	EAM24180
2702	IGOA=IGOA+1	EAM24190
	GO TO(2706,2707),IGOA	EAM24200
C	CHANGE SIGN OF THE CONTROL PERTURBATION	EAM24210
2706	DELUV(I)=-DELUV(I)	EAM24220
	SASV(I)=ASV(I)	EAM24230
	GO TO 2708	EAM24240
C	REDUCE THE SIZE OF THE CONTROL PERTURBATION	EAM24250
2707	DELUV(I)=DELUV(I)/2.0	EAM24260
	NHCV(I)=NHCV(I)+1	EAM24270
	IF(NHCV(I)-NHM)2713,2713,2714	EAM24280
2714	DELUV(I)=0.0	EAM24290
	GO TO 2708	EAM24300
2713	IGOA=1	EAM24310
C	CALCULATE THE SLEW CONTROLS	EAM24320
2708	DO 2710 J=K,L	EAM24330
	IA=LREFAV(K)	EAM24340
	UFV(IA)=DELUV(I)	EAM24350
2710	QUFV(IA)=UFV(IA)*ASCALV(IA)	EAM24360
2700	J=J+3	EAM24370
C	CHECK TERMINATION CRITERION	EAM24380
5	IF(DELUV(1)*DELUV(2))2715,2484,2715	EAM24390
2715	RETURN	EAM24400
C		EAM24410
C	TERMINATE INITIAL ALIGNMENT	EAM24420
2484	MODEQ=3	EAM24430
	INITL=1	EAM24440
C	RESET N TO ORIGINAL VALUE	EAM24450
	N=NS	EAM24460
C	RESET XFSV AND YFSV TO ORIGINAL VALUES	EAM24470
	CALL MCPY(SXFSV,XFSV,1,9,0)	EAM24480
	CALL MCPY(SYFSV,YFSV,1,9,0)	EAM24490
	RETURN	EAM24500
C		EAM24510
	END	EAM24520

	SUBROUTINE MFCS(NENTRY)	EAM24530
C		EAM24540
C	SUBROUTINE TO CALCULATE MIRROR FIGURE CONTROL SYSTEM PARAMETERS	EAM24550
C		EAM24560
C	SIGMA 5 TYPE B DIMENSION STATEMENTS START	EAM24570
	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM24580
	1 YFSV(20),XFRV(20),DUMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM24590
	2 GAINM(1600),ASV(3)	EAM24600
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DUMV,UFAV,	EAM24610
	1 DUMVA,GAINV,GAINM,ASV	EAM24620
C		EAM24630
	DIMENSION LACTV(20)	EAM24640
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM24650
	1 NTPD,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM24660
C		EAM24670
	DIMENSION AM(400),AIM(400)	EAM24680
	COMMON/BLKMFC/AM,AIM	EAM24690
C		EAM24700
	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM24710
	1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM24720
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM24730
	3 MODV(20)	EAM24740
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM24750
	1 NCXV,NCPV,NICPV,JCXV, JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM24760
	2 NMCXV,NMCPV,NMICPV,MODV	EAM24770
C		EAM24780
	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM24790
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM24800
C		EAM24810
C	SIGMA 5 TYPE B DIMENSION STATEMENTS END	EAM24820
C		EAM24830
C	SIGMA 2 DIMENSION STATEMENTS START	EAM24840
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM24850
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM24860
	2 QASV(3)	EAM24870
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM24880
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSENS,	EAM24890
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM24900
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM24910
C		EAM24920
C	SIGMA 2 DIMENSION STATEMENTS END	EAM24930
		EAM24940
C	DIMENSION DUMM(400)	EAM24950
	1000 FORMAT(5H MFCS)	EAM24960
	1001 FORMAT(/,11X,4HA*AI)	EAM24970
	1002 FORMAT(/,10X,5HGAINM)	EAM24980
	1003 FORMAT(/,33H SIMPLIFIED LINEAR CONTROL SYSTEM,/))	EAM24990
	1004 FORMAT(/,30H LINEAR OPTIMAL CONTROL SYSTEM,/))	EAM25000
	1005 FORMAT(/,34H GENERALIZED LINEAR CONTROL SYSTEM,/))	EAM25010
	1006 FORMAT(/,7X,8HA*ASCALE)	EAM25020
	1007 FORMAT(/,9X,6HART*AR)	EAM25030
	1008 FORMAT(/,3X,12H(ART*AR)**-1)	EAM25040
	1009 FORMAT(/,19H FIGURE SENSOR DATA,/))	EAM25050
	1010 FORMAT(/,21H FIGURE ACTUATOR DATA,/))	EAM25060
	1011 FORMAT(/,29H CONTROL SYSTEM CONFIGURATION,/))	EAM25070
	1012 FORMAT(/,18H MIRROR MODEL DATA,/))	EAM25080
	1013 FORMAT(/,24H MIRROR CALIBRATION DATA,/))	EAM25090
	1014 FORMAT(/,19H ACTUATOR TEST DATA,/))	

1015	FORMAT(2E10.3)	EAM25100
C		EAM25110
	GO TO (1,2,3,4,5,6,7),NENTRY	EAM25120
C		EAM25130
C	MIRROR FIGURE CONTROL SYSTEM INPUT DATA	EAM25140
1	PRINT 1000	EAM25150
	CALL IRANDP(5,NSNSWT,NTYPI,NTYPO,NPUNCH,NMAG,IA,IA,4)	EAM25160
C		EAM25170
C	READ BASIC DATA FOR THE EXPERIMENTAL ACTIVE MIRROR	EAM25180
	PRINT 1012	EAM25190
C	N,NR	EAM25200
	CALL IRANDP(2,N,NR,IA,IA,IA,IA,4)	EAM25210
	NA=N	EAM25220
C	IF MODV(10)=2 READ IN THE REDUCED A MATRIX	EAM25230
	IGO=MODV(10)	EAM25240
	GO TO(2005,2006),IGO	EAM25250
2006	NA=NR	EAM25260
2005	CONTINUE	EAM25270
C	AM	EAM25280
	CALL MXRNP(AM,N,NA,4)	EAM25290
C	ASCALE	EAM25300
	CALL RANDPD(2,ASCALE,AIMSCL,DA,DA,DA,DA,4)	EAM25310
	ASCLB=1.0/AIMSCL	EAM25320
	GAINV(7)=ASCALE	EAM25330
C	SCALE THE A MATRIX	EAM25340
	IF(ASCALE-1.0)2002,2001,2002	EAM25350
2002	IA=N*NA	EAM25360
	DO 2003 I=1,IA	EAM25370
2003	AM(I)=AM(I)*ASCALE	EAM25380
	PRINT 1006	EAM25390
	CALL MXRNP(AM,N,NA,3)	EAM25400
2001	CONTINUE	EAM25410
	PRINT 1009	EAM25420
C	FSCALE	EAM25430
	CALL RANDPD(1,FSCALE,DA,DA,DA,DA,DA,4)	EAM25431
	FSCALV(1)=FSCALE	EAM25432
C	XFSV	EAM25450
	CALL MXRNP(XFSV,1,N,4)	EAM25460
C	YFSV	EAM25470
	CALL MXRNP(YFSV,1,N,4)	EAM25480
C	PSCALE	EAM25490
	CALL RANDPD(1,PSCALE,DA,DA,DA,DA,DA,4)	EAM25500
	PRINT 1010	EAM25510
C	LACTV	EAM25520
	CALL IMXRNP(LACTV,1,N,4)	EAM25530
C	ASCALV	EAM25540
	CALL MXRNP(ASCALV,1,NR,4)	EAM25550
C	GAINV	EAM25560
	PRINT 1011	EAM25570
C	MODOP	EAM25580
	CALL IRANDP(1,MODOP,IA,IA,IA,IA,IA,4)	EAM25590
	PRINT 1014	EAM25600
	CALL ACTCAL(1)	EAM25610
	PRINT 1013	EAM25620
	CALL MIRCAL(1)	EAM25630
C		EAM25640
C	CALCULATE FEEDBACK MATRIX	EAM25650
2	GO TO(2110,2120,2130),MODOP	EAM25660

C		EAM25670
C	SIMPLIFIED LINEAR CONTROL SYSTEM	EAM25680
C	GAINM=ARR**(-1)	EAM25690
C	GENERATE ARR	EAM25700
2110	NRA=NR	EAM25710
	PRINT 1003	EAM25720
	GO TO(2111,2112),IAMODE	EAM25730
2111	CALL REDUAM(1)	EAM25740
2112	CALL REDUAM(2)	EAM25750
C	GAINM=ARR**(-1)	EAM25760
C	SCALE A	EAM25770
	IA=NR*NR	EAM25780
	DO 2113 I=1,IA	EAM25790
2113	AM(I)=AM(I)*AIMSCL	EAM25800
	CALL SINV(NR,AM,GAINM,DUMM,DA)	EAM25810
C	SCALE AI	EAM25820
	DO 2114 I=1,IA	EAM25830
	AM(I)=AM(I)*ASCLB	EAM25840
2114	GAINM(I)=GAINM(I)*AIMSCL	EAM25850
C	CHECK A*AI=I?	EAM25860
	CALL MPRD(AM,GAINM,AIM,NR,NR,0,0,NR)	EAM25870
	PRINT 1001	EAM25880
	CALL MXRNP(AIM,NR,NR,3)	EAM25890
C		EAM25900
C	PRINT SLCS GAIN MATRIX	EAM25910
	PRINT 1002	EAM25920
	CALL MXRNP(GAINM,NR,NR,3)	EAM25930
	RETURN	EAM25940
C		EAM25950
C	LINEAR OPTIMAL CONTROL SYSTEM	EAM25960
C	GAINM=((ART*AR)**(-1))*ART	EAM25970
2120	NRA=N	EAM25980
	PRINT 1004	EAM25990
C	GENERATE AR	EAM26000
	GO TO(2121,2122),IAMODE	EAM26010
2121	CALL REDUAM(1)	EAM26020
C	GENERATE ART	EAM26030
2122	CALL MTRA(AM,AIM,N,NR,0)	EAM26040
	CALL MPRD(AIM,AM,GAINM,NR,N,0,0,NR)	EAM26050
	PRINT 1007	EAM26060
	CALL MXRNP(GAINM,NR,NR,3)	EAM26070
C	SCALE A	EAM26080
	IA=NR*NR	EAM26090
	DO 2123 I=1,IA	EAM26100
2123	GAINM(I)=GAINM(I)*AIMSCL	EAM26110
	CALL SINV(NR,GAINM,AM,DUMM,DA)	EAM26120
C	SCALE AI	EAM26130
	DO 2124 I=1,IA	EAM26140
	GAINM(I)=GAINM(I)*ASCLB	EAM26150
2124	AM(I)=AM(I)*AIMSCL	EAM26160
	PRINT 1008	EAM26170
	CALL MXRNP(AM,NR,NR,3)	EAM26180
	CALL MPRD(AM,GAINM,DUMM,NR,NR,0,0,NR)	EAM26190
	PRINT 1001	EAM26200
	CALL MXRNP(DUMM,NR,NR,3)	EAM26210
	CALL MPRD(AM,AIM,GAINM,NR,NR,0,0,N)	EAM26220
C		EAM26230

C	PRINT LOCS GAIN MATRIX	EAM26240
	PRINT 1002	EAM26250
	CALL MXRNP(GAINM,NR,N,3)	EAM26260
	RETURN	EAM26270
C		EAM26280
C	GENERALIZED LINEAR CONTROL SYSTEM	EAM26290
2130	NRA=N	EAM26300
	PRINT 1005	EAM26310
	CALL MXRNP(GAINM,NR,N,4)	EAM26320
	RETURN	EAM26330
C		EAM26340
C	EXPERIMENTALLY CHECK ACTUATOR OPERATION	EAM26350
3	IGO=MODV(12)	EAM26360
	GO TO (2201,2202),IGO	EAM26370
C		EAM26380
2202	CALL MARK(1,8,2,2,5)	EAM26390
	RETURN	EAM26400
C		EAM26410
5	CALL MARK(1,8,3,2,6)	EAM26420
	RETURN	EAM26430
C		EAM26440
2201	CALL ACTCAL(2)	EAM26450
	CALL ACTCAL(3)	EAM26460
6	RETURN	EAM26470
C		EAM26480
C	EXPERIMENTALLY DETERMINE AR	EAM26490
4	IGO=MODVQ(6)	EAM26500
	GO TO (2301,2302),IGO	EAM26510
C		EAM26520
2302	CALL MARK(1,9,2,2,7)	EAM26530
	RETURN	EAM26540
C		EAM26550
7	CALL MARK(1,9,3,2,6)	EAM26560
	RETURN	EAM26570
2301	CALL MIRCAL(2)	EAM26580
	CALL MIRCAL(3)	EAM26590
	RETURN	EAM26600
C		EAM26610
	END	EAM26620

C	SUBROUTINE MIRCAL(NENTRY)	EAM26630
C	SUBROUTINE TO CALIBRATE MIRROR	EAM26640
C	SIGMA 5 TYPE B DIMENSION STATEMENTS START	EAM26650
C	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM26660
	1 YFSV(20),XFRV(20),DUMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM26670
	2 GAINM(1600),ASV(3)	EAM26680
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DUMV,UFAV,	EAM26690
	1 DUMVA,GAINV,GAINM,ASV	EAM26700
C	DIMENSION LACTV(20)	EAM26710
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM26720
	1 NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM26730
C	DIMENSION AM(400),AIM(400)	EAM26740
	COMMON/BLKMFC/AM,AIM	EAM26750
C	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM26760
	1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM26770
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM26780
	3 MODV(3)	EAM26790
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM26800
	1 NCXV,NCPV,NICPV,JCXV,JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM26810
	2 NMCXV,NMCPV,NMICPV,MODV	EAM26820
C	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM26830
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM26840
C	SIGMA 5 TYPE B DIMENSION STATEMENTS END	EAM26850
C	SIGMA 2 DIMENSION STATEMENTS START	EAM26860
C	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM26870
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM26880
	2 QASV(3)	EAM26890
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM26900
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSENS,	EAM26910
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM26920
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM26930
C	SIGMA 2 DIMENSION STATEMENTS END	EAM26940
C	1000 FORMAT(7H MIRCAL)	EAM26950
	1001 FORMAT(/,13X,2HAM)	EAM26960
C	GO TO (1,2,3,4,5,6),NENTRY	EAM26970
C	INPUT DATA	EAM26980
C	1 PRINT 1000	EAM26990
	CALL IRANDP(1,NMEASF,IA,IA,IA,IA,IA,IA,4)	EAM27000
	CALL RANDPD(1,DACT,DA,DA,DA,DA,DA,DA,4)	EAM27010
	DB=2.0*DACT*NMEASF	EAM27020
	DB=1.0/DB	EAM27030
	RETURN	EAM27040
C	INITIALIZATION	EAM27050
C	2 IGOA=MODV(12)	EAM27060
	RETURN	EAM27070
C	MIRROR CALIBRATION	EAM27080
C	3 SGAIN=GAINV(1)	EAM27090
	IGOA=MODVQ(6)	EAM27100
	GAINV(1)=0.0	EAM27110
		EAM27120
		EAM27130
		EAM27140
		EAM27150
		EAM27160
		EAM27170
		EAM27180
		EAM27190
		EAM27200
		EAM27210
		EAM27220
		EAM27230

	NTIMS=NFSENS	EAM27240
	NTIMSQ=NTIMS	EAM27250
C	SET AM=0.0	EAM27260
	IA=N*NR	EAM27270
	DO 2100 I=1,IA	EAM27280
2100	AM(I)=0.0	EAM27290
	I=0	EAM27300
2101	I=I+1	EAM27310
	J=0	EAM27320
	IF(I-NR)2103,2103,2200	EAM27330
2103	J=J+1	EAM27340
	IF(J-NMEASF)2104,2104,2208	EAM27350
C	RETURN TO SIGMA 2 TO INITIALIZE EAMCS	EAM27360
2104	GO TO (2105,2106),IGOA	EAM27370
C*****	EAM SOFTWARE TEST CODING*****	EAM27380
2105	CALL EAMCS(8)	EAM27390
	GO TO 4	EAM27400
C*****	EAM SOFTWARE TEST CODING*****	EAM27410
2106	CALL MARK(1,22,8,9,4)	EAM27420
	RETURN	EAM27430
C		EAM27440
4	UFV(I)=-DACT	EAM27450
	QUFV(I)=UFV(I)*ASCALV(I)	EAM27460
	GO TO (2107,2108),IGOA	EAM27470
2108	CALL MARK(1,22,3,9,5)	EAM27480
	RETURN	EAM27490
C	RETURN TO SIGMA 2 TO ADJUST ACTUATORS AND MEASURE FIGURE ERROR	EAM27500
C		EAM27510
C*****	EAM SOFTWARE TEST CODING*****	EAM27520
2107	CALL EAMCS(3)	EAM27530
C*****	EAM SOFTWARE TEST CODING*****	EAM27540
C		EAM27550
5	DO 2202 K=1,N	EAM27560
2202	DUMV(K)=XFV(K)	EAM27570
	UFV(I)=DACT	EAM27580
	QUFV(I)=UFV(I)*ASCALV(I)	EAM27590
	GO TO (2204,2205),IGOA	EAM27600
2205	CALL MARK(1,22,3,9,6)	EAM27610
	RETURN	EAM27620
C	RETURN TO SIGMA 2 TO ADJUST ACTUATORS AND MEASURE FIGURE ERROR	EAM27630
C		EAM27640
C*****	EAM SOFTWARE TEST CODING*****	EAM27650
2204	CALL EAMCS(3)	EAM27660
C*****	EAM SOFTWARE TEST CODING*****	EAM27670
C		EAM27680
6	DO 2203 K=1,N	EAM27690
	DUMV(K)=(XFV(K)-DUMV(K))	EAM27700
2203	AM(K+(I-1)*N)=DUMV(K)+AM(K+(I-1)*N)	EAM27710
2201	GO TO 2103	EAM27720
2208	DO 2207 K=1,N	EAM27730
2207	AM(K+(I-1)*N)=AM(K+(I-1)*N)*DB	EAM27740
	UFV(I)=0.0	EAM27750
	QUFV(I)=0.0	EAM27760
	GO TO 2101	EAM27770
C		EAM27780
C	PRINT OUT MIRROR DEFORMATION-ACTUATOR COMMAND ARRAY	EAM27790
2200	PRINT 1001	EAM27800
	CALL MXRNP(AM,N,NR,3)	EAM27810
	GAINV(1)=SGAIN	EAM27820
	RETURN	EAM27830
C		EAM27840
	END	EAM27850

C	SUBROUTINE MIRMDL(NENTRY,IACT)	EAM27860
C	STRUCTURAL MODEL OF THE MIRROR FOR TESTING THE EAM SOFTWARE	EAM27870
C	PACKAGE	EAM27880
C	SIGMA 5 TYPE A DIMENSION STATEMENTS START	EAM27890
C	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM27900
	1 YFSV(20),XFRV(20),DOMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM27910
	2 GAINM(1600),ASV(3)	EAM27920
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DOMV,UFAV,	EAM27930
	1 DUMVA,GAINV,GAINM,ASV	EAM27940
C	DIMENSION LACTV(20)	EAM27950
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM27960
	1 NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM27970
C	DIMENSION AM(400),AIM(400)	EAM27980
	COMMON/BLKMFC/AM,AIM	EAM27990
C	COMMON/BLKT/T,DT,DTH,DTPLOT,DTNOIS,TPhi,TPRNT,TEND	EAM28000
C	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM28010
	1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM28020
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCVP(10),NMICPV(10),	EAM28030
	3 MODV(20)	EAM28040
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM28050
	1 NCXV,NCPV,NICPV,JCXV, JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM28060
	2 NMCXV,NMCVP,NMICPV,MODV	EAM28070
C	DIMENSION XV(50),NAMV(50),DUMV(20),DUMM(400),PARV(50),IPARV(50),	EAM28080
	1 SXV(50),SPARV(50),ISPARV(50),IDUMV(20)	EAM28090
	COMMON/BLKSIM/XV,NAMV,DUMV,DUMM,PARV,IPARV,SXV,SPARV,ISPARV,	EAM28100
	1 IDUMV	EAM28110
C	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM28120
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM28130
C	SIGMA 5 TYPE A DIMENSION STATEMENTS END	EAM28140
C	SIGMA 2 DIMENSION STATEMENTS START	EAM28150
C	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM28160
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM28170
	2 QASV(3)	EAM28180
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM28190
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENQ,NTIMSQ,LSENS,	EAM28200
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM28210
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM28220
C	SIGMA 2 DIMENSION STATEMENTS END	EAM28230
C	DIMENSION UV(20)	EAM28240
C	1001 FORMAT(/,5X,10HAMM*ASCALE)	EAM28250
C	GO TO(1,2,3),NENTRY	EAM28260
C	INPUT DATA	EAM28270
1	PRINT 1000	EAM28280
C	READ DEFORMATION FORCE MATRIX FOR ACTUAL MIRROR	EAM28290
		EAM28300
		EAM28310
		EAM28320
		EAM28330
		EAM28340
		EAM28350
		EAM28360
		EAM28370
		EAM28380
		EAM28390
		EAM28400
		EAM28410
		EAM28420

C	READ IN REDUCED MATRIX IF MODV(10)=2	EAM28430
	IGO=MODV(10)	EAM28440
	GO TO(2006,2007),IGO	EAM28450
2006	NA=N	EAM28460
	GO TO 2008	EAM28470
2007	NA=NR	EAM28480
2008	CALL MXRNP(AMM,N,NA,4)	EAM28490
C	SCALE AMM	EAM28500
	IF(GAINV(7)-1.0)2003,2004,2003	EAM28510
2003	IA=N*NA	EAM28520
	DO 2005 I=1,IA	EAM28530
2005	AMM(I)=AMM(I)*GAINV(7)	EAM28540
	PRINT 1001	EAM28550
	CALL MXRNP(AMM,N,NA,3)	EAM28560
2004	CONTINUE	EAM28570
C	READ INITIAL DISTURBANCE INDUCED FIGURE ERROR	EAM28580
	CALL MXRNP(XFDV,1,N,4)	EAM28590
C		EAM28600
C	INITIALIZATION	EAM28610
C	SET XFAV=XFDV	EAM28620
2	CALL MCPY(XFDV,XFAV,1,N,0)	EAM28630
	DO 2000 I=1,N	EAM28640
	UFAV(I)=0.0	EAM28650
2000	UV(I)=0.0	EAM28660
	RETURN	EAM28670
C		EAM28680
C	STRUCTURE SIMULATION	EAM28690
C	FORM COMPLETE DISTURBANCE VECTOR UV	EAM28700
3	IGO=MODV(10)	EAM28710
	GO TO(2301,2302),IGO	EAM28720
2301	J=0	EAM28730
	DO 2001 I=1,N	EAM28740
	IF(LACTV(I))2002,2001,2002	EAM28750
2002	J=J+1	EAM28760
	UV(I)=UFAV(J)	EAM28770
C		EAM28780
1000	FORMAT(7H MIRMDL)	EAM28790
2001	CONTINUE	EAM28800
C	DUMBV=AMM*UV	EAM28810
	CALL MPRD(AMM,UV,DUMBV,N,N,0,0,1)	EAM28820
	GO TO 2303	EAM28830
C	DUMBV=AMMR*UFAV	EAM28840
2302	CALL MPRD(AMM,UFAV,DUMBV,N,NA,0,0,1)	EAM28850
C	XFAV=XFDV+DUMBV	EAM28860
2303	CALL MMADD(N,1.0,XFDV,1.0,DUMBV,XFAV)	EAM28870
	RETURN	EAM28880
C		EAM28890
	END	EAM28900

C	SUBROUTINE PINDX(NENTRY,PINDEX,YV)	EAM28910
C		EAM28920
C	SUBROUTINE TO CALCULATE PERFORMANCE INDICES FOR THE MIRROR FIGURE	EAM28930
C	CONTROL SYSTEM	EAM28940
C		EAM28950
C		EAM
C	SIGMA 5 TYPE D DIMENSION STATEMENTS START	EAM
	DIMENSION LACTV(20)	EAM
	COMMON/BKIEAM/LACTV,NCVEL,N,N-,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM
1	NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSSENS,NTIMS	EAM
C	SIGMA 5 TYPE C DIMENSION STATEMENTS END	EAM
C	SIGMA 5 TYPE D DIMENSION STATEMENTS END	EAM
	DIMENSION WGTV(20),YV(1)	EAM
C		EAM28960
	1000 FORMAT(6H PINDX)	EAM28970
C		EAM28980
	GO TO(1,2,3),NENTRY	EAM28990
C		EAM29000
C	INPUT DATA	EAM29010
1	PRINT 1000	EAM29020
	CALL MXRNP(WGTV,1,N,4)	EAM29030
	RETURN	EAM29040
C		EAM29050
C	INITIALIZATION	EAM29060
2	PINDEX=0.0	EAM29070
	RETURN	EAM29080
C		EAM29090
C	EVALUATE PERFORMANCE INDEX	EAM29100
3	PINDEX=0.0	EAM29110
	DO 2000 I=1,N	EAM29120
2000	PINDEX=PINDEX+YV(I)*YV(I)*WGTV(I)	EAM29130
	PINDEX=SQRT(PINDEX)	EAM29140
	RETURN	EAM29150
C		EAM29160
	END	EAM29170
		EAM29180

	SUBROUTINE PLRT(NENTRY,XV,T,DT,NRUN)	EAM29190
C		EAM29200
	DIMENSION XV(1),PMAV(40),FACV(5),PLOTV(45)	EAM29210
C		EAM29220
	DIMENSION IPLOTV(40),IMODV(40),SCALV(40),SCALAV(40),SSCALV(40)	EAM29230
C		EAM29240
	DIMENSION X(200),Y(2000)	EAM29250
C		EAM29260
	1000 FORMAT(5H PLRT)	EAM29270
	1001 FORMAT(10X,5HSCALV)	EAM29280
	1003 FORMAT(11X,4HN RUN,9X,6HN PLOTV,11X,4HNPTS,9X,6HDT PLOT,11X,4HTEND)	EAM29290
	1004 FORMAT(9X,6HI PLOTV)	EAM29300
	1006 FORMAT(10X,5HPMAV)	EAM29310
	1008 FORMAT(3I15,2F15.6)	EAM29320
	1009 FORMAT(6X,9H RUN SCALE)	EAM29330
	1012 FORMAT(14H REDIMENSION Y)	EAM29340
	1013 FORMAT(14H REDIMENSION X)	EAM29350
C		EAM29360
	GO TO(1,2,3,4,5,6,7,8,9,10,11,12),NENTRY	EAM29370
C		EAM29380
	READ PLOT DATA	EAM29390
C		EAM29400
1	PRINT 1000	EAM29410
	CALL IRANDP(1,NPLOTV,IA,IA,IA,IA,IA,IA,4)	EAM29420
	CALL RANDPD(1,DTPLOT,DA,DA,DA,DA,DA,DA,4)	EAM29430
	CALL IMXRNP(IPLOTV,1,NPLOTV,4)	EAM29440
	CALL IMXRNP(IMODV,1,NPLOTV,4)	EAM29450
	CALL MXRNP(SCALV,1,NPLOTV,4)	EAM29460
C	READ IN DATA FOR SYSTEM 360 PLOT ROUTINE	EAM29470
	TEND=T	EAM29480
	CALL STORED(1,RANG,WIDTH,SPAC,SCALAV,X,Y,TEND,NPLOTV,NPTS,NRUN)	EAM29490
	DO 2000 I=1,NPLOTV	EAM29500
	SCALAV(I)=SCALV(I)	EAM29510
2000	SSCALV(I)=SCALV(I)	EAM29520
	NDIMX=200	EAM29530
	NDIMY=2000	EAM29540
	FACV(1)=1.0	EAM29550
	FACV(2)=1.25	EAM29560
	FACV(3)=2.5	EAM29570
	FACV(4)=5.0	EAM29580
	FACV(5)=7.5	EAM29590
	RETURN	EAM29600
C		EAM29610
	INITIALIZE PLOT	EAM29620
C		EAM29630
2	TEND=T	EAM29640
	DTH=DT/2.0	EAM29650
	STP=0.0	EAM29660
C		EAM29670
	START OF PLOT RUN	EAM29680
C		EAM29690
	SET Y AND X TO ZERO	EAM29700
C		EAM29710
3	DO 2004 I=1,NDIMX	EAM29720
2004	X(I)=0.0	EAM29730
	DO 2005 I=1,NDIMY	EAM29740
2005	Y(I)=0.0	EAM29750
C	RESET SCALV TO ORIGINAL VALUE	
	DO 2002 I=1,NPLOTV	
	SCALAV(I)=SSCALV(I)	
2002	SCALV(I)=RANG/SSCALV(I)	

C	DEFINE AND PROTECT THE DATA SET	EAM29760
	DA=TEND+0.01*DTPLOT	EAM29770
	IA=DA/DTPLOT	EAM29780
	NPTS=IA+1	EAM29790
	NSTORE=0	EAM29800
	IF(NPTS-NDIMX)2120,2120,2130	EAM29810
2130	PRINT 1013	EAM29820
	NPTS=NDIMX	EAM29830
	DA=NPTS-1	EAM29840
	DTPLOT=TEND/DA	EAM29850
2120	IA=NPTS*NPLTV	EAM29860
	IF(IA-NDIMY)2100,2100,2110	EAM29870
2110	PRINT 1012	EAM29880
	NPTS=NDIMY/NPLTV	EAM29890
	DA=NPTS-1	EAM29900
	DTPLOT=TEND/DA	EAM29910
C	INITIALIZE 360 PLOTTING ROUTINE	EAM29920
2100	CALL STORED(2,RANG,WIDTH,SPAC,SCALAV,X,Y,TEND,NPLTV,NPTS,NRUN)	EAM29930
	RETURN	EAM29940
C		EAM29950
C	STORE PLOT DATA IN ARRAYS X AND Y EVERY DTPLOT SECONDS	EAM29960
4	IF(T+DTH-STP)2810,2820,2820	EAM29970
2820	STP=STP+DTPLOT	EAM29980
	NSTORE=NSTORE+1	EAM29990
	X(NSTORE)=T	EAM30000
	K=NSTORE	EAM30010
	DO 2800 I=1,NPLTV	EAM30020
	J=IPLTV(I)	EAM30030
	Y(K)=XV(J)	EAM30040
2800	K=K+NPTS	EAM30050
2810	RETURN	EAM30060
C		EAM30070
5	RETURN	EAM30080
6	RETURN	EAM30090
7	RETURN	EAM30100
8	RETURN	EAM30110
C		EAM30120
C	PLOT DATA FOR ONE RUN	EAM30130
C	GENERATE SELECTED SCALE FACTORS	EAM30140
9	DO 2600 I=1,NPLTV	EAM30150
2600	PMAV(I)=0.0	EAM30160
C	FIND MAXIMUM MAGNITUDES OF STORED VARIABLES	EAM30170
	L=1	EAM30180
	M=NPTS	EAM30190
	DO 2610 I=1,NPLTV	EAM30200
	DO 2612 K=L,M	EAM30210
	DA=ABS(Y(K))	EAM30220
	IF(DA-PMAV(I))2612,2612,2613	EAM30230
2613	PMAV(I)=DA	EAM30240
2612	CONTINUE	EAM30250
	L=L+NPTS	EAM30260
2610	M=M+NPTS	EAM30270
C	GENERATE SCALE FACTORS AUTOMATICALLY	EAM30280
	DO 2621 K=1,NPLTV	EAM30290
	IGO=IMODV(K)	EAM30300
	GO TO(2621,2615),IGO	EAM30310
2615	DO 2618 I=1,20	EAM30320

J=I-10	EAM30330
DB=10.0**J	EAM30340
DO 2617 L=1,5	EAM30350
DA=DB*FACV(L)	EAM30360
IF(PMAXV(K)-DA)2619,2617,2617	EAM30370
2617 CONTINUE	EAM30380
2618 CONTINUE	EAM30390
2619 SCALAV(K)=DA	EAM30400
SCALV(K)=RANG/DA	EAM30410
2621 CONTINUE	EAM30420
C PRINT PLOTTED DATA CHARACTERISTICS	EAM30430
PRINT 1003	EAM30440
PRINT 1008,NRUN,NPLOTV,NPTS,DTPLT,TEND	EAM30450
PRINT 1004	EAM30460
CALL IMXRNP(IPLTV,1,NPLOTV,3)	EAM30470
PRINT 1001	EAM30480
CALL MXRNP(SCALAV,1,NPLOTV,3)	EAM30490
PRINT 1006	EAM30500
CALL MXRNP(PMAXV,1,NPLOTV,3)	EAM30510
C SCALE AND LIMIT DATA FOR PLOTTING	EAM30520
L=1	EAM30530
M=NPTS	EAM30540
DO 2410 I=1,NPLOTV	EAM30550
DB=SCALV(I)	EAM30560
DO 2412 K=L,M	EAM30570
Y(K)=Y(K)*DB	EAM30580
2412 CALL SATLIM(Y(K),RANG,IA)	EAM30590
L=L+NPTS	EAM30600
2410 M=M+NPTS	EAM30610
C PLOT DATA FOR ONE RUN USING THE 360 PLOTTING ROUTINE	EAM30620
CALL STORED(3,RANG,WIDTH,SPAC,SCALAV,X,Y,TEND,NPLOTV,NPTS,NRUN)	EAM30630
RETURN	EAM30640
C	EAM30650
10 RETURN	EAM30660
11 RETURN	EAM30670
C	EAM30680
TERMINATE 360 PLOTTING ROUTINE	EAM30690
C	EAM30700
12 CALL STORED(4,RANG,WIDTH,SPAC,SCALAV,X,Y,TEND,NPLOTV,NPTS,NRUN)	EAM30710
RETURN	EAM30720
C	EAM30730
END	

	SUBROUTINE RESPON(NENTRY)	EAM30740
C		EAM30750
C	SUBROUTINE TO GENERATE THE TIME DOMAIN RESPONSE OF THE	EAM30760
C	EXPERIMENTAL ACTIVE MIRROR	EAM30770
C		EAM30780
C	SIGMA 5 TYPE A DIMENSION STATEMENTS START	EAM30790
	DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM30800
	1 YFSV(20),XFRV(20),DOMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM30810
	2 GAINM(1600),ASV(3)	EAM30820
	COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DOMV,UFAV,	EAM30830
	1 DUMVA,GAINV,GAINM,ASV	EAM30840
C		EAM30850
	DIMENSION LACTV(20)	EAM30860
	COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM30870
	1 NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM30880
C		EAM30890
	DIMENSION AM(400),AIM(400)	EAM30900
	COMMON/BLKMFC/AM,AIM	EAM30910
C		EAM30920
	COMMON/BLKT/T,DT,DTH,DTPLT,DTNOIS,TPHI,TPRNT,TEND	EAM30930
C		EAM30940
	DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM30950
	1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM30960
	2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM30970
	3 MODV(20)	EAM30980
	COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM30990
	1 NCXV,NCPV,NICPV,JCXV,JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM31000
	2 NMCXV,NMCPV,NMICPV,MODV	EAM31010
C		EAM31020
	DIMENSION XV(50),NAMV(50),DUMV(20),DUMM(400),PARV(50),IPARV(50),	EAM31030
	1 SXV(50),SPARV(50),ISPARV(50),IDUMV(20)	EAM31040
	COMMON/BLKSIM/XV,NAMV,DUMV,DUMM,PARV,IPARV,SXV,SPARV,ISPARV,	EAM31050
	1 IDUMV	EAM31060
C		EAM31070
	DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM31080
	COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV	EAM31090
C		EAM31100
C	SIGMA 5 TYPE A DIMENSION STATEMENTS END	EAM31110
C		EAM31120
	SIGMA 2 DIMENSION STATEMENTS START	EAM31130
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM31140
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM31150
	2 QASV(3)	EAM31160
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM31170
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSENS,	EAM31180
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM31190
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM31200
C		EAM31210
C	SIGMA 2 DIMENSION STATEMENTS END	EAM31220
	1000 FORMAT(/,13X,2HT=,F15.6,8X,7HPINDEX=,F15.6)	EAM31230
	1001 FORMAT(7H RESPON)	EAM31240
	1002 FORMAT(12X,3HXFV)	EAM31250
	1003 FORMAT(12X,3HUFV)	EAM31260
	1004 FORMAT(11X,4HUFV)	EAM31270
	1005 FORMAT(7F15.6)	EAM31280
	1006 FORMAT(11X,4HFMN,10X,5HXFSIG,10X,5HAMBIG,9X,6HMFMEAS,	EAM31290
	1 11X,4HFSW,9X,6HFLAST,10X,5HLSNS)	EAM31300
	1007 FORMAT(10X,5HJMEAS,10X,5HJWAIT,10X,5HJSSENS,10X,5HISENS,10X,	

1 5HXFACT,9X,6HPINDEX,10X,5HFSOUT)	EAM31310
1008 FORMAT(10X,5HFSERR,7X,8HFSPINDEX,5X,10HFSNOIS SIG,9X,6HFSNOIS,	EAM31320
1 8X,7HRPINDEX)	EAM31330
1009 FORMAT(11X,4HXFAV)	EAM31340
C	EAM31350
GO TO (1,2,3,4,5,6),NENTRY	EAM31360
C	EAM31370
C INPUT DATA	EAM31380
1 PRINT 1001	EAM31390
CALL RANDPD(4,DT,TPRNT,TEND,DTNOIS,DA,DA,DA,4)	EAM31400
CALL IRANDP(1,NSSRUN,IA,IA,IA,IA,IA,IA,4)	EAM31410
RETURN	EAM31420
C	EAM31430
C INITIALIZE SYSTEMS	EAM31440
2 T=0.0	EAM31450
ST=0.0	EAM31460
STN=0.0	EAM31470
STP=0.0	EAM31480
NTIMSQ=1	EAM31490
C INITIALIZE THE MIRROR FIGURE CONTROL SYSTEM	EAM31500
CALL FSMDL(2,I)	EAM31510
CALL MIRMDL(2,I)	EAM31520
C*****STATEMENT TO RESET VALUE OF GAINV(1)=PARV(1)*****	EAM31530
GAINV(1)=PARV(1)	EAM31540
C*****STATEMENT TO RESET VALUE OF GAINV(1)=PARV(1)*****	EAM31550
CALL MAINA(2)	EAM31560
CALL MAINA(4)	EAM31570
C RETURN TO SIGMA 2 TO INITIALIZE EAMCS	EAM31580
C	EAM31590
IGO=MODV(11)	EAM31600
C TEST REMOTE TERMINAL CONTROL VIA TYPCON IF MODV(11)=2	EAM31610
IA=8	EAM31620
GO TO(2201,2202),IGO	EAM31630
2202 IA=2	EAM31640
2201 CONTINUE	EAM31650
C	EAM31660
IGO=MODV(12)	EAM31670
GO TO (2002,2000),IGO	EAM31680
2000 CALL MARK(1,22,IA,7,4)	EAM31690
RETURN	EAM31700
C	EAM31710
C*****EAM SOFTWARE TEST CODING*****	EAM31720
2002 CALL EAMCS(IA)	EAM31730
C*****EAM SOFTWARE TEST CODING*****	EAM31740
4 CALL PLRT(2,XV,TEND,DT,NRUN)	EAM31750
RETURN	EAM31760
C	EAM31770
C PERFORM SIMULATION	EAM31780
3 CONTINUE	EAM31790
C CHECK TO SEE IF SENSE SWITCH NSSRUN IS RESET IF MODV(1)=1	EAM31800
IGO=MODV(1)	EAM31810
GO TO(2011,2007),IGO	EAM31820
2011 CALL TYPOUT(NSSRUN,1)	EAM31830
2010 CONTINUE	EAM31840
IF(SNSWT(NSSRUN))2010,2020,2020	EAM31850
2020 CONTINUE	EAM31860
C	EAM31870

C	START RUN BY SETTING SENSE SWITCH NSSRUN IF MODV(1)=1	EAM31880
2012	CALL TYPOUT(NSSRUN,1)	EAM31890
2004	CONTINUE	EAM31900
	IF(SNSWT(NSSRUN))2007,2004,2004	EAM31910
C		FAM31920
C	STOCHASTIC STRUCTURAL DISTURBANCE GENERATOR	EAM31930
2005	IF(T+DTH-STN)2052,2053,2053	EAM31940
2053	STN=STN+DTNOIS	EAM31950
	CALL NOIS(3)	EAM31960
C		EAM31970
C	OUTPUT PRINTED DATA	EAM31980
2052	IF(T+DTH-ST)2008,2006,2006	EAM31990
2006	CONTINUE	EAM32000
C	OUTPUT PRINT	EAM32010
	PRINT 1000,T,DOMV(13)	EAM32020
	IGO=MODV(8)	EAM32030
	GO TO(2100,2101),IGO	EAM32040
2100	PRINT 1009	FAM32050
	CALL MXRNP(XFAV,1,N,3)	EAM32060
2101	PRINT 1002	EAM32070
	CALL MXRNP(XFV,1,N,3)	EAM32080
	PRINT 1003	EAM32090
	CALL MXRNP(UFV,1,NR,3)	FAM32100
	PRINT 1004	EAM32110
	CALL MXRNP(UFAV,1,NR,3)	EAM32120
	PRINT 1006	FAM32130
	CALL MXRNP(DOMV,1,7,3)	EAM32140
	PRINT 1007	EAM32150
	PRINT 1005,(DOMV(I),I=8,14)	EAM32160
	PRINT 1008	EAM32170
	PRINT 1005,(DOMV(I),I=15,19)	EAM32180
C	OUTPUT PRINT	EAM32190
	ST=ST+TPRNT	FAM32200
2008	CONTINUE	EAM32210
C		FAM32220
C	AUXILLIARY PLOTTED DATA	EAM32230
C	AUXILLIARY PLOTTED DATA	EAM32240
C		EAM32250
C	OUTPUT PLOTTED DATA	EAM32260
C	STORE DATA IF MODV(2)=2	EAM32270
	IGO=MODV(2)	EAM32280
	GO TO(2103,2102),IGO	EAM32290
2102	CALL PLRT(4,XV,T,DT,NRUN)	EAM32300
	GO TO 2104	EAM32310
C	PLOT DATA ONLINE IF MODV(2)=1	FAM32320
2103	CALL PLRT(8,XV,T,DT,NRUN)	EAM32330
C		EAM32340
C	INCREMENT COMPUTER TIME	EAM32350
2104	T=T+DT	EAM32360
C		EAM32370
C	T=TEND	EAM32380
	IF(T+DTH-TEND)2007,2001,2001	EAM32390
C		EAM32400
C	EXPERIMENTAL ACTIVE MIRROR SIMULATION COMPUTATIONS	EAM32410
2007	IGO=MODV(12)	EAM32420
	GO TO (2106,2105),IGO	EAM32430
C*****	EAM SOFTWARE TEST CODING*****EAM32440	EAM32440

2106 CALL EAMCS(4)	EAM32450
GO TO 5	EAM32460
C*****EAM SOFTWARE TEST CODING*****	EAM32470
2105 CALL MARK(1,22,4,7,5)	EAM32480
RETURN	EAM32490
C	EAM32500
5 CONTINUE	EAM32510
C	EAM32520
C SIMULATION CYCLE TERMINATION	EAM32530
C TERMINATE RUN IF SENSE SWITCH NSSRUN IS RESET	EAM32540
C AND MODV(1)=1	EAM32550
2121 IGO=MODV(1)	EAM32560
GO TO(2122,2005),IGO	EAM32570
2122 IF(SNSWT(NSSRUN))2005,2001,2001	EAM32580
2001 RETURN	EAM32590
C	EAM32600
C OPERATE THE FIGURE CONTROL SYSTEM	EAM32610
6 RETURN	EAM32620
C	EAM32630
END	EAM32640

SUBROUTINE SIMSYS(NENTRY)	EAM32650
C	EAM32660
C MAIN CONTROL PROGRAM FOR SIMULATION	EAM32670
C	EAM32680
C SIGMA 5 TYPE A DIMENSION STATEMENTS START	EAM32690
DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),	EAM32700
1 YFSV(20),XFRV(20),DOMV(20),UFAV(20),DUMVA(20),GAINV(10),	EAM32710
2 GAINM(1600),ASV(3)	EAM32720
COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DOMV,UFAV,	EAM32730
1 DUMVA,GAINV,GAINM,ASV	EAM32740
C	EAM32750
C	EAM32760
DIMENSION LACTV(20)	EAM32770
COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,	EAM32780
1 NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS	EAM32790
C	EAM32800
DIMENSION AM(400),AIM(400)	EAM32810
COMMON/BLKMFC/AM,AIM	EAM32820
C	EAM32830
COMMON/BLKT/T,DT,DTH,DTPLT,DTNOIS,TPhi,TPRNT,TEND	EAM32840
C	EAM32850
DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),	EAM32860
1 JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),	EAM32870
2 CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),	EAM32880
3 MODV(20)	EAM32890
COMMON/BLKIV/IAV,NIIV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,	EAM32900
1 NCXV,NCPV,NICPV,JCXV, JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,	EAM32910
2 NMCXV,NMCPV,NMICPV,MODV	EAM32920
C	EAM32930
DIMENSION XV(50),NAMV(50),DUMV(20),DUMM(400),PARV(50),IPARV(50),	EAM32940
1 SXV(50),SPARV(50),ISPARV(50),IDUMV(20)	EAM32950
COMMON/BLKSIM/XV,NAMV,DUMV,DUMM,PARV,IPARV,SXV,SPARV,ISPARV,	EAM32960
1 IDUMV	EAM32970
C	EAM32980
DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)	EAM32990

C	COMMON/BLKMDL/AMM,WV,DUMRV,XFAV,XFDV	EAM33000
C	SIGMA 5 TYPE A DIMENSION STATEMENTS END	EAM33010
C		FAM33020
		EAM33030
		EAM33040
C	SIGMA 2 DIMENSION STATEMENTS START	EAM33050
	DIMENSION QXFVS(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	FAM33060
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM33070
	2 QASV(3)	EAM33080
	COMMON/SIGTWO/QXFVS,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM33090
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMFASQ,NFSENQ,NTIMSQ,LSENS,	FAM33100
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM33110
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM33120
C	SIGMA 2 DIMENSION STATEMENTS END	EAM33130
C		FAM33140
	1000 FORMAT(7H SIMSYS)	EAM33150
	1001 FORMAT(3X,1H*,9A4,4X,9A4)	EAM33160
	1002 FORMAT(3X,1H ,9A4,3X,1H*,9A4)	FAM33170
	1003 FORMAT(/,23H SIMULATION TIMING DATA)	EAM33180
	1004 FORMAT(/,42H RUN SET IDENTIFICATION AND NUMBER OF RUNS,/)	FAM33190
	1005 FORMAT(/,14H PLOTTING DATA)	FAM33200
	1006 FORMAT(/,28H DATA MODIFICATIONS FOR RUNS,/)	EAM33210
	1010 FORMAT(/,1X,7HNEW RUN,7X,5HNRUN=,110,4X,6HNRUNC=,15,/)	EAM33220
	1011 FORMAT(1H1)	FAM33230
	1020 FORMAT(18A4)	FAM33240
	1021 FORMAT(/,1X,16HOPERATING MODES*,/)	EAM33250
	1022 FORMAT(/)	FAM33260
	1023 FORMAT(62H THE MARSHALL SPACE FLIGHT CENTER MIRROR FIGURE CONTROL	FAM33270
	1SYSTEM,/,53H DEVELOPED BY THE MIT CHARLES STARK DRAPER LABORATORY,	EAM33280
	2 /)	EAM33290
C		FAM33300
	GO TO (1,2,3,4,5,6),NENTRY	EAM33310
C		FAM33320
C	READ AND PRINT PROGRAM HEADING CONSISTING OF N CARDS IN A	EAM33330
C	FORMAT	FAM33340
	1 CALL IRANDP(1,N,1A,1A,1A,1A,1A,1A,4)	EAM33350
	DO 2008 I=1,N	EAM33360
	READ 1020,(NAMV(J),J=1,18)	EAM33370
	PRINT 1020,(NAMV(J),J=1,18)	EAM33380
	2008 CONTINUE	FAM33390
	PRINT 1011	EAM33400
	PRINT 1023	FAM33410
	PRINT 1000	EAM33420
C		EAM33430
C	READ IN BASIC SIMULATION DATA	EAM33440
C	READ MODV AND IDENTIFY OPERATING MODES	EAM33450
	NMODV=12	EAM33460
	CALL IMXRNP(MODV,1,NMODV,4)	EAM33470
	PRINT 1021	EAM33480
	DO 2005 I=1,NMODV	EAM33490
	READ 1020,(NAMV(J),J=1,18)	EAM33500
	IGO=MODV(I)	EAM33510
	GO TO(2006,2007),IGO	EAM33520
	2006 PRINT 1001,(NAMV(J),J=1,9),(NAMV(J),J=10,18)	EAM33530
	GO TO 2005	EAM33540
	2007 PRINT 1002,(NAMV(J),J=1,9),(NAMV(J),J=10,18)	EAM33550
	2005 CONTINUE	EAM33560
	PRINT 1022	EAM33570
C		EAM33580
C	TRANSFER MODV DATA TO THE SIGMA 2	EAM33590
	DO 2010 I=1,NMODV	EAM33600
	2010 MODVQ(I)=MODV(I)	EAM33610

C	PRINT 1003	EAM33620
C	READ IN RESPON DATA	EAM33630
	CALL RESPON(1)	EAM33640
	DTH=DT/2.0	EAM33650
C		EAM33660
C	READ IN STARTING RUN NUMBER AND TOTAL NUMBER OF RUNS TO BE MADE	EAM33670
	PRINT 1011	EAM33680
	PRINT 1004	EAM33690
	CALL IRANDP(2,NRUN,NRUNM,IA,IA,IA,IA,IA,4)	EAM33700
C		EAM33710
	NRUNC=0	EAM33720
	NXV=50	EAM33730
	NPV=50	EAM33740
	NIPV=50	EAM33750
C	SET SIMULATION VALUES TO ZERO INITIALLY	EAM33760
	DO 2100 I=1,NXV	EAM33770
2100	XV(I)=0.0	EAM33780
	DO 2101 I=1,NPV	EAM33790
2101	PARV(I)=0.0	EAM33800
	DO 2102 I=1,NIPV	EAM33810
2102	IPARV(I)=0	EAM33820
C		EAM33830
C	READ IN NAMES AND NUMBERS OF EDITED ELEMENTS	EAM33840
	PRINT 1000	EAM33850
	PRINT 1006	EAM33860
	CALL IRANDP(3,NCXV,NCPV,NICPV,IA,IA,IA,IA,4)	EAM33870
	CALL NPDRNP(DUMV,JCXV,NMCXV,1,NCXV,4)	EAM33880
	CALL NPDRNP(DUMV,JCPV,NMCPV,1,NCPV,4)	EAM33890
	CALL NPDRNP(DUMV,JICPV,NMICPV,1,NICPV,4)	EAM33900
C		EAM33910
	IGO=MODV(1)	EAM33920
	GO TO(2003,2002),IGO	EAM33930
C		EAM33940
C	AUTOMATIC MODE	EAM33950
C	READ STORED VALUES OF PARV AND IPARV	EAM33960
2002	CALL MXRNP(CXM,NRUNM,NCXV,4)	EAM33970
	CALL MXRNP(CPM,NRUNM,NCPV,4)	EAM33980
	CALL IMXRNP(ICPM,NRUNM,NICPV,4)	EAM33990
C		EAM34000
C	READ DATA IN SUBROUTINES	EAM34010
	PRINT 1011	EAM34020
2003	CALL TYPOUT(IA,4)	EAM34030
	PRINT 1005	EAM34040
	CALL PLRT(1,XV,T,DT,NRUN)	EAM34050
C		EAM34060
C	INITIALIZE DATA STORAGE FILE	EAM34070
2004	CALL PLRT(2,XV,TEND,DT,NRUN)	EAM34080
	NRUNC=0	EAM34090
	IF(NRUN)2110,2110,2120	EAM34100
2110	NRUN=1	EAM34110
C	ESTABLISH NEW FILE	EAM34120
2111	CALL PLRT(3,XV,T,DT,NRUN)	EAM34130
	GO TO 2121	EAM34140
C	FIND START OF RUN IN OLD FILE	EAM34150
2120	CALL PLRT(5,XV,T,DT,NRUN)	EAM34160
	GO TO 2111	EAM34170
2121	CONTINUE	EAM34180
	PRINT 1011	EAM34190
C		EAM34200
C	READ IN DATA FOR THE EXPERIMENTAL ACTIVE MIRROR	EAM34210
	CALL MFCS(1)	EAM34220
	CALL MAINA(1)	EAM34230
		EAM34240

C									EAM34250
C	TRANSFER EAM DATA TO XV,PARV AND IPARV								EAM34260
C	DEFINE UTILIZED DIMENSIONS OF XV,PARV AND IPARV								EAM34270
	NXMAX=0								EAM34280
	NPARV=10								EAM34290
	NIPARV=10								EAM34300
C	XV IS NOT USED FOR DATA STORAGE AT THE MOMENT								EAM34310
C	CONTENTS OF PARV=GAINV								EAM34320
C	GAIN DT TEND FSTFLT PACTTC FSNSIG								EAM34330
C	ASCALE								EAM34340
	GAINV(2)=DT								EAM34350
	GAINV(3)=TEND								EAM34360
	DO 2131 I=1,NPARV								EAM34370
2131	PARV(I)=GAINV(I)								EAM34380
C	CONTENTS OF IPARV								EAM34390
C	NSENS NWAIT NPOS NMINT NMEAS NTIMS								EAM34400
C	MODOP								EAM34410
	IPARV(1)=NSENS								EAM34420
	IPARV(2)=NWAIT								EAM34430
	IPARV(3)=NPOS								EAM34440
	IPARV(4)=NMINT								EAM34450
	IPARV(5)=NMEAS								EAM34460
	IPARV(6)=NTIMS								EAM34470
	IPARV(7)=MODOP								EAM34480
C									EAM34490
C	STORE XV,PARV, AND IPARV IN SXV,SPARV, AND ISPARV								EAM34500
	CALL MCPY(XV,SXV,1,NXMAX,0)								EAM34510
	CALL MCPY(PARV,SPARV,1,NPARV,0)								EAM34520
	CALL IMCPY(IPARV,ISPARV,1,NIPARV,0)								EAM34530
C									EAM34540
C									EAM34550
C									EAM34560
C	REINITIALIZE SIMULATION								EAM34570
2	PRINT 1010,NRUN,NRUNC								EAM34580
	T=0.0								EAM34590
C									EAM34600
C	RESET XV,PARV AND IPARV TO ORIGINAL VALUES								EAM34610
	CALL MCPY(SXV,XV,1,NXMAX,0)								EAM34620
	CALL MCPY(SPAPV,PARV,1,NPARV,0)								EAM34630
	CALL IMCPY(ISPARV,IPARV,1,NIPARV,0)								EAM34640
	IGO=MODV(1)								EAM34650
	GO TO(2401,2402),IGO								EAM34660
C									EAM34670
C	MANUAL MODE								EAM34680
C	READ IN NEW VALUES AND EDIT XV,PARV AND IPARV								EAM34690
2401	CALL EDITA(XV,CXV,JCXV,NCXV,2)								EAM34700
	CALL EDITA(PARV,CPV,JCPV,NCPV,2)								EAM34710
	CALL IEDITA(IPARV,ICPV,JICPV,NICPV,2)								EAM34720
2402	IGO=MODV(1)								EAM34730
	GO TO(2200,2400),IGO								EAM34740
C									EAM34750
C	AUTOMATIC MODE								EAM34760
C	EXTRACT NEW VALUES FROM MEMORY								EAM34770
2400	IF(NRUNC-NRUNM)2103,2108,2108								EAM34780
2108	RETURN								EAM34790
2103	IA=NRUNC+1								EAM34800
	DO 2105 I=1,NCXV								EAM34810
2105	CXV(I)=ELM(CXM,IA,I,NRUNM)								EAM34820
	DO 2106 I=1,NCPV								EAM34830
2106	CPV(I)=ELM(CPM,IA,I,NRUNM)								EAM34840
	DO 2107 I=1,NICPV								EAM34850
2107	ICPV(I)=ELM(ICPM,IA,I,NRUNM)								EAM34860

C		EAM34870
C	EDIT XV,PARV AND IPARV	EAM34880
	CALL EDITA(XV,CXV,JCXV,NCXV,3)	EAM34890
	CALL EDITA(PARV,CPV,JCPV,NCPV,3)	EAM34900
	CALL IEDITA(IPARV,ICPV,JICPV,NICPV,3)	EAM34910
C		EAM34920
C	PRINT MODIFIED VALUES OF XV,PARV AND IPARV FOR CHECK PURPOSES	EAM34930
	CALL NPDRNP(CXV,JCXV,NMCXV,1,NCXV,7)	EAM34940
	CALL NPDRNP(CPV,JCPV,NMCPV,1,NCPV,7)	EAM34950
	CALL NPDRNP(DUMV,ICPV,NMICPV,1,NICPV,8)	EAM34960
C		EAM34970
	2200 CONTINUE	EAM34980
C		EAM34990
C	TRANSFER XV,PARV AND IPARV DATA TO THE EXPERIMENTAL ACTIVE MIRROR	EAM35000
	DT=PARV(2)	EAM35010
	TEND=PARV(3)	EAM35020
	NSENS=IPARV(1)	EAM35030
	NWAIT=IPARV(2)	EAM35040
	NPOS=IPARV(3)	EAM35050
	NMINT=IPARV(4)	EAM35060
	NMEAS=IPARV(5)	EAM35070
	NTIMS=IPARV(6)	EAM35080
	MODOP=IPARV(7)	EAM35090
	CALL MCPY(PARV,GAINV,1,NPARV,0)	EAM35100
C		EAM35110
C	INITIALIZE THE EXPERIMENTAL ACTIVE MIRROR	EAM35120
	IGO=MODV(12)	EAM35130
	GO TO (2133,2134),IGO	EAM35140
	2134 CALL MARK(1,7,2,1,4)	EAM35150
	RETURN	EAM35160
C		EAM35170
	2133 CALL RESPON(2)	EAM35180
C		EAM35190
C	SIMULATE EXPERIMENTAL ACTIVE MIRROR	EAM35200
C		EAM35210
	4 IGO=MODV(12)	EAM35220
	GO TO (3,2136),IGO	EAM35230
	2136 CALL MARK(1,7,3,1,5)	EAM35240
	RETURN	EAM35250
C		EAM35260
	3 CALL RESPON(3)	EAM35270
C		EAM35280
C	TERMINATE SIMULATION RUN	EAM35290
C	IDENTIFY END OF RUN	EAM35300
	5 CALL PLRT(9,XV,T,DT,NRUN)	EAM35310
C		EAM35320
C	INCREMENT DATA FILE PARAMETERS	EAM35330
	NRUN=NRUN+1	EAM35340
	CALL PLRT(3,XV,T,DT,NRUN)	EAM35350
	NRUNC=NRUNC+1	EAM35360
C	TERMINATE 360 PLOTTING ROUTINE	EAM35370
	IF(NRUNC-NRUNM)2,2500,2500	EAM35380
	2500 CALL PLRT(12,XV,T,DT,NRUN)	EAM35390
	GO TO 1	EAM35400
C		EAM35410
C	EDIT DATA TO CORRESPOND TO NRUNC=NFLGC	EAM35420
	6 NRUNC=NFLGC-1	EAM35430
	GO TO 2	EAM35440
C		EAM35450
C		EAM35460
	END	EAM35470

	SUBROUTINE STORED(NENTRY,RANG,WIDTH,SPAC,SCALV,X,Y,XSPRED,NPLOTV,	EAM35480
	1 NPTS,NRUN)	EAM35490
C		EAM35500
	DIMENSION SCALV(1),X(1),Y(1),TITLE(2),HEADNG(10)	EAM35510
C		EAM35520
	DOUBLE PRECISION PROB,PROG,PAPER,TYPINK	EAM35530
C		EAM35540
	DATA PROB,PROG,PAPER,TYPINK /8HM9040 ,8H6362 ,8HWHITE ,	EAM35550
	1 8HBLACK /	EAM35560
	DATA TITLE/'NRUN', ' = ' /	EAM35570
	DATA HEADNG/'EXPE','RIME','NTAL',' ACT','IVE ','MIRR','OR S',	EAM35580
	1 'IMUL','ATIO','N ' /	EAM35590
C		EAM35600
	1000 FORMAT(7H STORED)	EAM35610
C		EAM35620
	GO TO(1111,2222,3333,4444),NENTRY	EAM35630
C		EAM35640
	DATA INPUT	EAM35650
C		EAM35660
	1111 PRINT 1000	EAM35670
	CALL RANDPD(3,RANG,WIDTH,SPAC,DA,DA,DA,DA,4)	EAM35680
	CALL NEWPLT(PROB,PROG,PAPER,TYPINK)	EAM35690
	XBEGIN=0.0	EAM35700
	XEND=0.0	EAM35710
	RETURN	EAM35720
C		EAM35730
	INITIALIZATION	EAM35740
C		EAM35750
	2222 DIST=SPAC+2.0*RANG	EAM35760
C		EAM35770
	DY = NO. OF INCHES PER ABSCISSA UNIT	EAM35780
	DY=WIDTH/XSPRED	EAM35790
	SPRED=XSPRED	EAM35800
	RETURN	EAM35810
C		EAM35820
	3333 CONTINUE	EAM35830
	DO 500 I=1,NPTS	EAM35840
C		EAM35850
	CONVERT X VECTOR TO INCHES	EAM35860
C		EAM35870
	500 X(I)=X(I)*DY	EAM35880
		EAM35890
C		EAM35900
	CREATE NEW REFERENCE POINT	EAM35910
C		EAM35920
	XNEW=XEND+2.0*(SPAC+RANG)	EAM35930
	CALL PLOT1(XNEW,0.,-3)	EAM35940
C		EAM35950
	LABEL EACH SET OF PLOTS	EAM35960
C		EAM35970
	DA=-(2.0*SPAC)	EAM35980
	DB=0.5*SPAC	EAM35990
	DC=8.0*DB	EAM36000
	CALL SYMBL5(DA,0.0,DB,HEADNG,90.0,40)	EAM36010
	DA=-SPAC	EAM36020
	CALL SYMBL5(DA,0.0,DB,TITLE,90.0,8)	EAM36030
	CALL NUMBR1(DA,DC,DB,NRUN,90.0,-1)	EAM36040
C		
	PLOT THE NPLOTV GRAPHS	
C		
	III=0	
	DB=0.20*SPAC	
	DC=WIDTH+SPAC	
	DO 540 J=1,NPLOTV	
C		
	DRAW ABSCISSA	

CALL PLOT1(XBEGIN,0.0,3)	EAM36050
XEND=XBEGIN+2.0*RANG	EAM36060
CALL PLOT1(XEND,0.0,2)	EAM36070
C	EAM36080
C DRAW ORDINATE	EAM36090
XMIDDL=XBEGIN+RANG	EAM36100
CALL PLOT1(XMIDDL,0.0,3)	EAM36110
CALL PLOT1(XMIDDL,WIDTH,2)	EAM36120
C	EAM36130
C LABEL WITH SCALE VALUE	EAM36140
DA=XBEGIN-0.10*SPAC	EAM36150
CALL NUMBRI(DA,0.0,DB,SCALV(J),90.0,6)	EAM36160
CALL NUMBRI(DA,DC,DB,J,90.0,-1)	EAM36170
XBEGIN=XBEGIN+DIST	EAM36180
YYY=RANG+(J-1)*DIST	EAM36190
C	EAM36200
C PLOT DATA ON COORDINATES	EAM36210
DO 530 I=1,NPTS	EAM36220
DO 530 I=1,NPTS	EAM36220
III=III+1	EAM36230
C NEGATE Y TO ACCOUNT FOR ROTATION OF 90 DEGREES	EAM36240
C X=-Y FORMER	EAM36250
C Y=X FORMER	EAM36260
Y(III)=-Y(III)+YYY	EAM36270
530 CONTINUE	EAM36280
C JJJ DENOTES THE BEGINNING OF THE JTH COLUMN VECTOR IN ARRAY Y.	EAM36290
JJJ=III-NPTS+1	EAM36300
CALL GRAPH(Y(JJJ),X,NPTS,0.,0.)	EAM36310
540 CONTINUE	EAM36320
C	EAM36330
C PRINT THE ABSISSA SCALE	EAM36340
DA=XBEGIN-DIST+2.0*RANG+0.10*SPAC+DB	EAM36350
CALL NUMBRI(DA,0.0,DB,0.0,90.0,6)	EAM36360
CALL NUMBRI(DA,WIDTH,DB,SPRED,90.0,6)	EAM36370
RETURN	EAM36380
C	EAM36390
4444 XEND=XEND+10.0	EAM36400
CALL PLOT1(XEND,0.,-3)	EAM36410
CALL ENDPLT	EAM36420
RETURN	EAM36430
C	EAM36440
END	EAM36450

	SUBROUTINE SUPE2	EAM36460
C		EAM36470
C	SUPERVISORY PROGRAM FOR THE SIGMA 2 SOFTWARE	EAM36480
C		EAM36490
C	SIGMA 2 DIMENSION STATEMENTS START	EAM36500
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM36510
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM36520
	2 QASV(3)	EAM36530
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM36540
	1 MSEQVQ,NSSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSSENS,	EAM36550
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM36560
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM36570
C	SIGMA 2 DIMENSION STATEMENTS END	EAM36580
C		EAM36590
	GO TO(1,2,3,4),NFLGA	EAM36600
C		EAM36610
C	CALLS TO ACTCMD	EAM36620
1	CALL ACTCMD(NFLGB)	EAM36630
	GO TO 2000	EAM36640
C		EAM36650
C	CALLS TO EAMCS	EAM36660
2	CALL EAMCS(NFLGB)	EAM36670
	GO TO 2000	EAM36680
C	CALLS TO FIGSEN	EAM36690
C		EAM36700
3	CALL FIGSEN(NFLGB)	EAM36710
	GO TO 2000	EAM36720
C		EAM36730
C	CALLS TO TYPCON(NFLGB)	EAM36740
4	CALL TYPCON(NFLGB)	EAM36750
C		EAM36760
C	PUT CODING TO TRANSFER CONTROL TO SIGMA 5 HERE	EAM36770
2000	RETURN	EAM36780
C		EAM36790
	END	EAM36800
C	SUPE5 MAIN SUPERVISORY PROGRAM FOR THE SIGMA 5	EAM36810
C		EAM36820
C	SUPERVISORY PROGRAM FOR THE SIGMA 5 SOFTWARE	EAM36830
C		EAM36840
C	SIGMA 2 DIMENSION STATEMENTS START	EAM36850
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM36860
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM36870
	2 QASV(3)	EAM36880
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM36890
	1 MSEQVQ,NSSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENSQ,NTIMSQ,LSSENS,	EAM36900
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM36910
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELQ	EAM36920
C	SIGMA 2 DIMENSION STATEMENTS END	EAM36930
C		EAM36940
	COMMON/BLKSUP/ITRANS	EAM36950
C		EAM36960
C	INITIALIZATION	EAM36970
2010	IF(ISTART-9999)2000,2007,2000	EAM36980
C	SET ISTART=9999 THE FIRST TIME SUPE5 IS EXECUTED	EAM36990
2000	ISTART=9999	EAM37000

CALL MARK(4,IA,IA,IA,IA)	EAM37010
CALL MARK(1,1,1,19,1)	EAM37020
ISTORE=0	EAM37030
C	EAM37040
C OPERATION	EAM37050
2005 CONTINUE	EAM37060
2004 IF(ITRANS-ISTORE)2002,2002,2001	EAM37070
2001 CALL MARK(2,NFLGA,NFLGB,IA,IA)	EAM37080
GO TO 2003	EAM37090
2002 CALL MARK(3,IA,IA,NFLGA,NFLGB)	EAM37100
2003 ISTORE=ITRANS	EAM37110
C	EAM37120
GO TO(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,	EAM37130
1 20,21,21,21,21),NFLGA	EAM37140
C	EAM37150
1 CALL SIMSYS(NFLGB)	EAM37160
GO TO 2004	EAM37170
2 CALL MFCS(NFLGB)	EAM37180
GO TO 2004	EAM37190
3 CALL MAINA(NFLGB)	EAM37200
GO TO 2004	EAM37210
4 CALL MAINB(NFLGB)	EAM37220
GO TO 2004	EAM37230
5 CALL FSMDL(NFLGB,INDEX)	EAM37240
GO TO 2004	EAM37250
6 CALL MIRMDL(NFLGB,INDEX)	EAM37260
GO TO 2004	EAM37270
7 CALL RESPON(NFLGB)	EAM37280
GO TO 2004	EAM37290
8 CALL ACTCAL(NFLGB)	EAM37300
GO TO 2004	EAM37310
9 CALL MIRCAL(NFLGB)	EAM37320
GO TO 2004	EAM37330
10 CALL ACTMDL(NFLGB,INDEX)	EAM37340
GO TO 2004	EAM37350
11 GO TO 2004	EAM37360
12 GO TO 2004	EAM37370
13 GO TO 2004	EAM37380
14 GO TO 2004	EAM37390
15 GO TO 2004	EAM37400
16 GO TO 2004	EAM37410
17 GO TO 2004	EAM37420
18 GO TO 2004	EAM37430
19 GO TO 2100	EAM37440
20 GO TO 2100	EAM37450
C	EAM37460
C INSERT CODING HERE TO TRANSFER TO THE SIGMA 2 COMPUTER	EAM37470
21 NFLGA=NFLGA-20	EAM37480
C SIGMA 5 CONFIGURATION	EAM37490
2008 CALL SUPE2	EAM37500
C	EAM37510
C ESTABLISH TRANSFERS ON RETURN TO SIGMA 5	EAM37520
2007 CONTINUE	EAM37530
GO TO(2201,2202,2203,2204,2205,2206,2207,2208,2209,2210,	EAM37540
1 2211,2212,2213,2214,2215,2216,2217,2218,2219,2220,2221),NFLGA	EAM37550
C	EAM37560
2201 CALL MARK(1,24,2,22,10)	EAM37570
GO TO 2020	EAM37580
2202 CALL MARK(1,24,2,22,11)	EAM37590
GO TO 2020	EAM37600
2203 CALL MARK(1,21,5,22,12)	EAM37610
GO TO 2020	EAM37620

2204 CALL MARK(1,6,3,22,7)	EAM37630
GO TO 2020	EAM37640
2205 CALL MARK(1,23,2,22,13)	EAM37650
GO TO 2020	EAM37660
2206 GO TO 2020	EAM37670
2207 CALL MARK(1,10,3,21,4)	EAM37680
GO TO 2020	EAM37690
2208 CALL MARK(1,3,6,22,5)	EAM37700
GO TO 2020	EAM37710
2209 CALL MARK(1,21,3,22,16)	EAM37720
GO TO 2020	EAM37730
2210 CALL MARK(1,5,5,22,17)	EAM37740
GO TO 2020	EAM37750
2211 CALL MARK(1,5,3,23,4)	EAM37760
GO TO 2020	EAM37770
2212 CALL MARK(1,4,5,24,3)	EAM37780
GO TO 2020	EAM37790
2213 CALL MARK(1,4,3,24,8)	EAM37800
GO TO 2020	EAM37810
2214 CALL MARK(1,4,4,24,7)	EAM37820
GO TO 2020	EAM37830
2215 GO TO 2020	EAM37840
2216 CALL MARK(1,23,3,22,14)	EAM37850
GO TO 2020	EAM37860
2217 CALL MARK(1,3,7,22,18)	EAM37870
GO TO 2020	EAM37880
2218 CALL MARK(1,3,5,22,6)	EAM37890
GO TO 2020	EAM37900
2219 CALL MARK(1,2,3,24,2)	EAM37910
GO TO 2020	EAM37920
2220 CALL MARK(1,2,4,24,2)	EAM37930
GO TO 2020	EAM37940
2221 CALL MARK(1,24,2,22,19)	EAM37950
GO TO 2020	EAM37960
2222 CALL MARK(1,1,6,24,2)	EAM37970
C	EAM37980
2020 GO TO 2004	EAM37990
C	EAM38000
C RESTART PROGRAM	EAM38010
2100 ISTART=0	EAM38020
GO TO 2010	EAM38030
C	EAM38040
C INSERT CODING TO TRANSFER TO THE SIGMA 2 HERE	EAM38050
2009 CONTINUE	EAM38060
C	EAM38070
END	EAM38080

	SUBROUTINE TYPCON(NENTRY)	EAM38090
C		EAM38100
C	SIGMA 2 ROUTINE FOR TYPEWRITER CONTROL OF THE EXPERIMENTAL	EAM38110
C	ACTIVE MIRROR	EAM38120
C		EAM38130
C	SIGMA 2 DIMENSION STATEMENTS START	EAM38140
	DIMENSION QXFSV(20),QYFSV(20),QDUMVA(20),QDUMVB(20),QDUMVC(20),	EAM38150
	1 QUFV(20),QUFAV(20),MSEQVQ(20),MODVQ(20),IDUMVQ(10),QUFERV(20),	EAM38160
	2 QASV(3)	EAM38170
	COMMON/SIGTWO/QXFSV,QYFSV,QDUMVA,QDUMVB,QDUMVC,QUFV,QUFAV,QXF,	EAM38180
	1 MSEQVQ,NSENSQ,NWAITQ,NPOSQ,NMINTQ,NMEASQ,NFSENQ,NTIMSQ,LSENS,	EAM38190
	2 NFLGA,NFLGB,NFLGC,NFLGD,NFLGE,NTYPOQ,NTYPIQ,MODVQ,NQ,NRQ,	EAM38200
	3 QDT,QDTE,QUFMAX,MODEQ,QGA,QGB,QUFERV,IDUMVQ,QASV,NCVELO	EAM38210
C	SIGMA 2 DIMENSION STATEMENTS END	EAM38220
C		EAM38230
	1000 FORMAT(1X,A4)	EAM38240
	1001 FORMAT(11H WRONG NAME)	EAM38250
	1002 FORMAT(10I3)	EAM38260
	1003 FORMAT(7H NAME=?)	EAM38270
	1004 FORMAT(4H I=?)	EAM38280
	1005 FORMAT(10F12.6)	EAM38290
	1006 FORMAT(7H MODE=?)	EAM38300
	1007 FORMAT(5H INIT)	EAM38310
	1008 FORMAT(5H STRT)	EAM38320
	1009 FORMAT(7H TYPCON)	EAM38330
	1010 FORMAT(14H ACTUATOR TEST)	EAM38340
	1011 FORMAT(12H MIRROR TEST)	EAM38350
	1012 FORMAT(7H MODFIN)	EAM38360
	1016 FORMAT(5H STOP)	EAM38370
	1017 FORMAT(3H JM)	EAM38380
	1018 FORMAT(18H ACCEPT NEW VALUES)	EAM38390
	1019 FORMAT(12H NEW VALUE=?)	EAM38400
	1020 FORMAT(13H MODE TOO BIG)	EAM38410
	1021 FORMAT(16H DIAGNOSTIC MODE)	EAM38420
	1022 FORMAT(13H EDITING MODE)	EAM38430
	1023 FORMAT(18H PERFORMANCE INDEX)	EAM38440
	1024 FORMAT(6H I,J=?)	EAM38450
	1025 FORMAT(12H RESTART JOB)	EAM38460
	1026 FORMAT(/,2HT=,F12.6,3HJM=,F12.6)	EAM38470
C		EAM38480
	GO TO (1,2,3,4,5,6,7,8,9),NENTRY	EAM38490
C		EAM38500
C	INITIALIZATION	EAM38510
1	RETURN	EAM38520
C		EAM38530
C	OPERATION	EAM38540
2	CONTINUE	EAM38550
C		EAM38560
C	SELECT EXPERIMENTAL MODE	EAM38570
2280	WRITE(NTYPO,1006)	EAM38580
	READ(NTYPI,1002) MODEQ	EAM38590
	WRITE(NTYPO,1002) MODEQ	EAM38600
	ICHNG=2	EAM38610
C		EAM38620
	GO TO (2211,2212,2213,2214,2215,2216,2217,2218,2219,2220,2221,	EAM38630
1	2222),MODEQ	EAM38640
C		EAM38650

C	INITIALIZE MFCS	EAM38660
2211	WRITE(NTYPO,1007)	EAM38670
	GO TO 2207	EAM38680
C		EAM38690
C	START MFCS	EAM38700
2212	WRITE(NTYPO,1008)	EAM38710
	GO TO 2207	EAM38720
C		EAM38730
C	STOP MIRROR FIGURE CONTROL SYSTEM	EAM38740
2213	WRITE(NTYPO,1016)	EAM38750
	GO TO 2207	EAM38760
C		EAM38770
C	TEST ACTUATORS	EAM38780
2214	WRITE(NTYPO,1010)	EAM38790
	NFLGA=19	EAM38800
	RETURN	EAM38810
C		EAM38820
C	TEST MIRROR	EAM38830
2215	WRITE(NTYPO,1011)	EAM38840
	NFLGA=20	EAM38850
	RETURN	EAM38860
C		EAM38870
C	DIAGNOSTIC MODE	EAM38880
2216	WRITE(NTYPO,1021)	EAM38890
	GO TO 2990	EAM38900
C		EAM38910
C	MODIFY DATA TO NRUNC=NFLGC	EAM38920
2217	READ(NTYPI,1002)NFLGC	EAM38930
	WRITE(NTYPO,1002)NFLGC	EAM38940
	NFLGA=22	EAM38950
	RETURN	EAM38960
C	RETURN TO SIMSYS(6) AND REENTER TYPCON(2)	EAM38970
C		EAM38980
C	UNUSED OPERATING MODE MODEQ=8	EAM38990
2218	CONTINUE	EAM39000
	GO TO 2280	EAM39010
C		EAM39020
C	EVALUATE AND TYPE FIGURE PERFORMANCE INDEX	EAM39030
2219	WRITE(NTYPO,1023)	EAM39040
	WRITE(NTYPO,1017)	EAM39050
C	RETURN TO SIGMA 5 TO CALCULATE THE PERFORMANCE INDEX	EAM39060
	NFLGA=12	EAM39070
	RETURN	EAM39080
C	RETURN TO MAINB(5) AND REENTER TYPCON(3)	EAM39090
C		EAM39100
C	TYPE VALUE OF THE PERFORMANCE INDEX	EAM39110
3	WRITE(NTYPO,1005)QDUMVA(1)	EAM39120
	GO TO 2207	EAM39130
C		EAM39140
C	MODIFY DATA BUSS VALUE	EAM39150
2220	WRITE(NTYPO,1022)	EAM39160
C	SET ICHNG=1	EAM39170
	ICHNG=1	EAM39180
	WRITE(NTYPO,1018)	EAM39190
	GO TO 2990	EAM39200
C		EAM39210
C	UNUSED OPERATING MODE MODEQ=11	EAM39220

2221 GO TO 2207	EAM39230
C	EAM39240
C REQUEST MODE AGAIN IF MODE VALUE IS TOO LARGE	EAM39250
2222 WRITE(NTYPO,1020)	EAM39260
GO TO 2280	EAM39270
C	EAM39280
C IDENTIFY VARIABLE NAME	EAM39290
2990 WRITE(NTYPO,1003)	EAM39300
READ(NTYPI,1000) NFLGC	EAM39310
WRITE(NTYPO,1000) NFLGC	EAM39320
C RETURN TO SIGMA 5 TO CATALOG AND CHECK VARIABLE NAME	EAM39330
NFLGA=13	EAM39340
IGO=MODVQ(12)	EAM39350
GO TO (2995,2994),IGO	EAM39360
C*****EAM SOFTWARE TEST CODING*****	EAM39370
2995 CALL MAINB(3)	EAM39380
8 GO TO(5,6,4,2280),NFLGC	EAM39390
C*****EAM SOFTWARE TEST CODING*****	EAM39400
2994 RETURN	EAM39410
C	EAM39420
C OUTPUT ERROR MESSAGE IF NAME IS INCORRECT	EAM39430
4 WRITE(NTYPO,1001)	EAM39440
GO TO 2990	EAM39450
C	EAM39460
C IDENTIFY VARIABLE INDEX	EAM39470
5 WRITE(NTYPO,1004)	EAM39480
READ(NTYPI,1002) NFLGD	EAM39490
WRITE(NTYPO,1002) NFLGD	EAM39500
GO TO 2330	EAM39510
6 WRITE(NTYPO,1024)	EAM39520
READ(NTYPI,1002) NFLGD,NFLGE	EAM39530
WRITE(NTYPI,1002) NFLGD,NFLGE	EAM39540
C	EAM39550
C ACCEPT NEW VALUE IF ICHNG=1	EAM39560
2330 GO TO(2331,2340),ICHNG	EAM39570
2331 WRITE(NTYPO,1019)	EAM39580
READ(NTYPI,1005) QDUMVA(1)	EAM39590
C RETURN TO SIGMA 5 T6 MODIFY AND/OR EXTRACT VALUE OF INTERROGATED	EAM39600
C VARIABLE	EAM39610
2340 NFLGA=14	EAM39620
NFLGC=ICHNG	EAM39630
RETURN	EAM39640
C RETURN TO MIINB(4) AND REENTER TYPCON(7)	EAM39650
C	EAM39660
C DISPLAY VALUE OF INTERROGATED VARIABLE	EAM39670
7 WRITE(NTYPO,1005) QDUMVA(1)	EAM39680
GO TO 2990	EAM39690
C	EAM39700
C NORMAL RETURN TO EAMCS	EAM39710
2207 NFLGA=6	EAM39720
RETURN	EAM39730
C	EAM39740
C OUTPUT EXPERIMENT DATA ON REMOTE I/O DEVICE	EAM39750
C	EAM39760
9 WRITE(NTYPO,1026) QDUMVA(2),QDUMVA(1)	EAM39770
RETURN	EAM39780
C RETURN TO EAMCS	EAM39790
C	EAM39800
END	EAM39810

APPENDIX B

EXPERIMENTAL ACTIVE MIRROR LIBRARY ROUTINE LISTINGS

B.1 EAM Library Routines

Considerable memory space can be saved by using subroutines to perform operations which are repeated a large number of times. Many small subroutines have been developed at MIT/DL to provide such common operations as matrix multiplication, data input and transfer, etc. In the EAM software package these subroutines appear as members of three libraries described in the following sections.

B.2 Miscellaneous Functions Package

This section presents listings of the subroutines which are used to perform a variety of program operations associated with the EAM but are not considered important enough for inclusion in Appendix A. This section describes the following programs.

```
SUBROUTINE EDITA(A,B,IB,NB,NENTRY)
SUBROUTINE IEDITA(IA,IB,IIB,NIB,NENTRY)
SUBROUTINE MARK(NENTRY,NSBA,NTYA,NSBB,NTYB)
SUBROUTINE NOIS(NENTRY)
SUBROUTINE REALT(TREAL)
SUBROUTINE REDUAM(NENTRY)
SUBROUTINE SATLIM(X,R,I)
FUNCTION SGN(X)
FUNCTION SNSWT(NENTRY)
SUBROUTINE TYPOUT(I,NENTRY)
```

```

SUBROUTINE EDITA(A,B,IB,NB,NENTRY)
C
C SUBROUTINE TO EDIT DATA
C
C DIMENSION A(1),B(1),IB(1)
C
C GO TO(1,2,3,4),NENTRY
C
C READ NB AND IB
1 CALL IRANDP(1,NB,IA,IA,IA,IA,IA,IA,4)
  CALL IMXRNP(IB,1,NB,4)
  RETURN
C
C READ NEW VALUES AND CHANGE A
2 CALL MXRNP(B,1,NB,4)
C
C CHANGE A ONLY
3 DO 2000 I=1,NB
  J=IB(I)
2000 A(J)=B(I)
  RETURN
C
C STORE OLD VALUES OF A IN B
4 DO 2001 I=1,NB
  J=IB(I)
2001 B(I)=A(J)
  RETURN
C
END

```

```

      SUBROUTINE IEDITA(IA,IB,IIB,NIB,NENTRY)
C
C      SUBROUTINE TO EDIT DATA
C
      DIMENSION IA(1),IB(1),IIB(1)
C
      GO TO(1,2,3,4),NENTRY
C
C      READ NIB AND IIB
C
1     CALL IRANDP(1,NIB,IK,IK,IK,IK,IK,IK,4)
      CALL IMXRNP(IIB,1,NB,4)
      RETURN
C
C      READ NEW VALUES AND CHANGE IA
C
2     CALL IMXRNP(IB,1,NIB,4)
C
C      CHANGE IA ONLY
C
3     DO 2000 I=1,NIB
        J=IIB(I)
      2000 IA(J)=IB(I)
      RETURN
C
C      STORE OLD VALUES OF IA IN IB
C
4     DO 2001 I=1,NIB
        J=IIB(I)
      2001 IB(I)=IA(J)
      RETURN
C
      END

```

```

SUBROUTINE MARK(NENTRY,NSBA,NTYA,NSBB,NTYB)
C
  DIMENSION NSAV(20),NTAV(20),NSBV(20),NTBV(20)
  COMMON/BLKSUP/ITRANS
C
  1000 FORMAT(17H TRANSITION ERROR,SI10)
C
  GO TO(1,2,2,4),NENTRY
C
  1   ITRANS=ITRANS+1
    IF(ITRANS-20)2003,2003,2004
  2004 PRINT 1000,ITRANS,NSBA,NTYA,NSBB,NTYB
    RETURN
C
  2003 NSAV(ITRANS)=NSBA
    NTAV(ITRANS)=NTYA
    NSBV(ITRANS)=NSBB
    NTBV(ITRANS)=NTYB
    RETURN
C
  CALL TO EXTRACT DESTINATION
C
  2   IF(ITRANS)2000,2000,2001
  2000 PRINT 1000,ITRANS,NSBA,NTYA,NSBB,NTYB
    RETURN
C
  2001 GO TO(2002,2002,3,4),NENTRY
C
  2002 NSBA=NSAV(ITRANS)
    NTYA=NTAV(ITRANS)
    NSBB=NSBV(ITRANS)
    NTYB=NTBV(ITRANS)
    RETURN
C
  CALL TO EXTRACT RETURN ADDRESS AND
  DECREMENT TRANSITION COUNTER
C
  3   NSBB=NSBV(ITRANS)
    NTYB=NTBV(ITRANS)
    NSBA=NSAV(ITRANS)
    NTYA=NTAV(ITRANS)
    ITRANS=ITRANS-1
    RETURN
C
  SET TRANSITION COUNTER TO ZERO
C
  4   ITRANS=0
    RETURN
C
  END

```

```
      SUBROUTINE NOIS(NENTRY)
C
C      SUBROUTINE TO GENERATE DISTURBANCES ON SYSTEM
C      DUMMY VERSION
C
      RETURN
C
      END
```

```

SUBROUTINE REALT(TREAL)
C
C SUBROUTINE TO INTERROGATE REAL TIME CLOCK
C
C TREAL=REAL TIME
C
C
C INSERT REAL TIME CLOCK INTERROGATION SOFTWARE HERE
C
C*****EAM SOFTWARE TEST CODING*****
COMMON /BLKT/ T,DT,DTH,DTPLOT,DTNOIS,TPHI,TPRNT,TEND
TREAL=T
C*****EAM SOFTWARE TEST CODING*****
RETURN
C
END

```

```

SUBROUTINE REDUAM(NENTRY)
C
C SUBROUTINE TO GENERATE AR AND ARR FROM A
C
C SIGMA 5 TYPE B DIMENSION STATEMENTS START
C   DIMENSION XFV(20),UFV(20),ASCALV(20),FSCALV(20),XFSV(20),
1   YFSV(20),XFRV(20),DUMV(20),UFAV(20),DUMVA(20),GAINV(10),
2   GAINM(1600),ASV(3)
C   COMMON/BLKEAM/XFV,UFV,ASCALV,FSCALV,XFSV,YFSV,XFRV,DUMV,UFAV,
1   DUMVA,GAINV,GAINM,ASV
C
C   DIMENSION LACTV(20).
C   COMMON/BKIEAM/LACTV,NCVEL,N,NR,NRA,MODE,MODOP,NSNSWT,NTYPI,
1   NTYPO,NPUNCH,NMAG,NSENS,NWAIT,NPOS,NMINT,NMEAS,NFSENS,NTIMS
C
C   DIMENSION AM(400),AIM(400)
C   COMMON/BLKMFC/AM,AIM
C
C   DIMENSION IAV(30),IBV(30),ICV(30),IDV(30),IEV(30),
1   JCXV(10),JCPV(10),JICPV(10),CXV(10),CPV(10),ICPV(10),
2   CXM(100),CPM(100),ICPM(100),NMCXV(10),NMCPV(10),NMICPV(10),
3   MODV(20)
C   COMMON/BLKIV/IAV,IAV,IBV,NIBV,ICV,NICV,IDV,NIDV,IEV,NIEV,NX,NU,
1   NCXV,NCPV,NICPV,JCXV, JCPV,JICPV,CXV,CPV,ICPV,CXM,CPM,ICPM,
2   NMCXV,NMCPV,NMICPV,MODV
C
C   DIMENSION AMM(400),WV(20),DUMBV(20),XFAV(20),XFDV(20)
C   COMMON/BLKMDL/AMM,WV,DUMBV,XFAV,XFDV
C   SIGMA 5 TYPE B DIMENSION STATEMENTS END
C
1000 FORMAT(/,13X,2HAR)
1010 FORMAT(/,12X,3HARR)
1020 FORMAT(7H REDUAM)
C
GO TO(1,2),NENTRY
C
C GENERATE AR BY REMOVING COLUMNS FROM A
1 PRINT 1020
K=0
DO 2000 J=1,N
IF(LACTV(J))2010,2000,2010
2010 K=K+1
DO 2020 I=1,N
CALL LOC(I,J,IA,N,N,0)
CALL LOC(I,K,IB,N,N,0)
2020 AIM(IB)=AM(IA)
2000 CONTINUE
C
C COPY RESULT INTO AM
C CALL MCPY(AIM,AM,N,NR,0)
C
C PRINT AR
C PRINT 1000
C CALL MXRNP(AM,N,NR,3)
C RETURN
C
C GENERATE ARR FROM AR BY REMOVING ROWS FROM AR
2 K=0
DO 2100 I=1,N
IF(LACTV(I))2110,2100,2110

```

```

2110 K=K+1
      DO 2120 J=1,NR
      CALL LOC(I,J,IA,N,NR,0)
      CALL LOC(K,J,IB,NR,NR,0)
2120 AIM(IB)=AM(IA)
2100 CONTINUE
C
C   COPY RESULT INTO AM
C   CALL MCPY(AIM,AM,NR,NR,0)
C
C   PRINT ARR
C   PRINT 1010
C   CALL MXRNP(AM,NR,NR,3)
C   RETURN
C
C   END

```

```

      SUBROUTINE SATLIM(X,R,I)
C
C      SUBROUTINE TO LIMIT THE RANGE OF A VARIABLE X TO  $-R < X < R$ 
C
      IF (ABS(X)-R) 2000,2000,2001
2001 X=R*SGN(X)
2000 RETURN
C
      END

```

```

      FUNCTION   SGN(X)
C
C   THE VALUE OF THE FUNCTION IS THE SIGN OF X
C
      IF(X)2000,2001,2001
2000 SGN=-1.0
      GO TO 2002
2001 SGN=+1.0
2002 RETURN
C
      END

```

```
FUNCTION  SNSWT(NENTRY)
C
C  DUMMY ROUTINE TO INTERROGATE SENSE SWITCHES
C
  SNSWT=0.0
  RETURN
C
  END
```

```

      SUBROUTINE TYP0UT(I,NENTRY)
C
C*****SUBROUTINE TO TYPE OUT MESSAGES
C
      GO TO(1,2,3,4,5),NENTRY
C
1000 FORMAT(7H TYP0UT)
1201 FORMAT(2HSS,1I)
1202 FORMAT(12HXX TOO LARGE)
1203 FORMAT(3HSS1)
C
1  WRITE(NTYPE,1201) I
   RETURN
C
2  WRITE(NTYPE,1202)
   RETURN
C
3  WRITE(NTYPE,1203)
   RETURN
C
4  PRINT 1000
   CALL IRANDP(1,NTYPE,1A,1A,1A,1A,1A,1A,4)
   RETURN
C
5  CONTINUE
   RETURN
C
      END

```

B.3 Mathematical Operations Package

This section presents listings of the subroutines which are used to perform mathematical operations on arrays.

```
FUNCTION   ELM(A,L,M,N)
SUBROUTINE ELMA(NENTRY,A,I,J,V,N)
SUBROUTINE GMADD(A,B,R,N,M)
SUBROUTINE GMPRD(A,B,R,N,M,L)
SUBROUTINE GMSUB(A,B,R,N,M)
SUBROUTINE GMTRA(A,R,N,M)
SUBROUTINE GTOSYM(X,XS,NX)
FUNCTION   IELM(IA,L,M,N)
SUBROUTINE IMCPY(IA,IR,N,M,MS)
SUBROUTINE LUC(I,J,IR,N,M,MS)
SUBROUTINE MCPY(A,R,N,M,MS)
SUBROUTINE MMADD(N,ALPHA,A,BETA,B,C)
SUBROUTINE MPRD(A,B,R,N,M,MSA,MSB,L)
SUBROUTINE MTRA(A,R,N,M,MS)
SUBROUTINE SINV(N,AI,B,A,D)
SUBROUTINE SYMTUG(XS,X,NX)
```

```

C      FUNCTION ELM(A,L,M,N)
C
C      FUNCTION RETURNS THE VALUE OF THE L,M TH ELEMENT OF THE MATRIX
C      A WHICH HAS N ROWS AND AN ARBITRARY NUMBER OF COLUMNS
C      A WHICH HAS N ROWS AND AN ARBITRARY NUMBER OF COLUMNS
C      A IS STORED IN GENERAL FORM COLUMN BY COLUMN
C
C      DIMENSION A(1)
C
C      ELM=A(M*N-N+L)
C      RETURN
C
C      END

```

```

SUBROUTINE ELMA(NENTRY,A,I,J,V,N)
C
C SUBROUTINE TO WRITE INTO AND READ FROM MEMORY THE I,JTH ELEMENT
C OF THE MATRIX A
C A HAS N ROWS AND AN ARBITRARY NUMBER OF COLUMNS
C A IS STORED IN GENERAL FORM COLUMN BY COLUMN
C
C DIMENSION A(1)
C
C GO TO(1,2),NENTRY
C
C A(I,J)=V
C 1 A(I+(J-1)*N)=V
C RETURN
C
C V=A(I,J)
C 2 V=A(I+(J-1)*N)
C
C RETURN
C
C END

```

```

SUBROUTINE GMADD(A,B,R,N,M)
C
C SUBROUTINE PERFORMS MATRIX ADDITION, R=A+B, WHERE A,B AND R ARE
C N BY M MATRICES ARE GENERAL MATRICES STORED IN GENERAL FORM
C COLUMN BY COLUMN
C
C DIMENSION A(1),B(1),R(1)
C
C NM=N*M
C DO 110 I=1,NM
110 R(I)=A(I)+B(I)
C RETURN
C
C END

```

```

SUBROUTINE GMPRD(A,B,R,N,M,L)
C
C FORM THE PRODUCT R=A*B WHERE A IS A N*M MATRIX AND B IS A M*L
C MATRIX
C A,B AND R ARE STORED IN GENERAL MATRIX FORM COLUMN BY COLUMN
C
C DIMENSION A(1),B(1),R(1)
C
C IR=0
C IK=-M
C DO 10 K=1,L
C IK=IK+M
C DO 10 J=1,N
C IR=IR+1
C JI=J-N
C IB=IK
C R(IR)=0.0
C DO 10 I=1,M
C JI=JI+N
C IB=IB+1
10 R(IR)=R(IR)+A(JI)*B(IB)
C RETURN
C
C END

```

```

SUBROUTINE GMSUB(A,B,R,N,M)
C
C SUBROUTINE PERFORMS MATRIX SUBTRACTION, R=A-B, WHERE A,B AND R ARE
C N BY M MATRICES.
C A,B AND R ARE STORED IN GENERAL MATRIX FORM COLUMN BY COLUMN
C
C DIMENSION A(1),B(1),R(1)
C
C NM=N*M
C DO 110 I=1,NM
110 R(I)=A(I)-B(I)
C RETURN
C
C END

```

```

SUBROUTINE GMTRA(A,R,N,M)
C
C   TRANSPOSE A GENERAL MATRIX
C
C   A - NAME OF MATRIX TO BE TRANSPOSED
C   R - NAME OF RESULTANT MATRIX
C   N - NUMBER OF ROWS OF A AND COLUMNS OF R
C   M - NUMBER OF COLUMNS OF A AND ROWS OF R
C
C   DIMENSION A(1),R(1)
C
C   IR=0
C   DO 10 I=1,N
C   IJ=I-N
C   DO 10 J=1,M
C   IJ=IJ+N
C   IR=IR+1
10  R(IR)=A(IJ)
C   RETURN
C
C   END

```

```

SUBROUTINE GTOSYM(X,XS,NX)
C
C PROGRAM CONVERTS A SQUARE NX BY NX MATRIX INTO A VECTOR OF LENGTH
C NF*(NF+1)/2 AND WHOSE ELEMENTS CONSIST OF THE UPPER TRIANGLE OF
C THE NX BY NX MATRIX, STORED IN COLUMNAR FORM.
C THE MATRIX X MUST BE STORED IN GENERAL FORM COLUMN BY COLUMN
C
C DIMENSION X(1),XS(1)
C
C LL=0
C DO 10 J=1,NX
C DO 10 I=1,J
C LL=LL+1
C K=(J-1)*NX+I
10 XS(LL)=X(K)
C RETURN
C
C END

```

```

      FUNCTION IELM(IA,L,M,N)
C
C  FUNCTION RETURNS THE VALUE OF THE L,M TH ELEMENT OF THE MATRIX
C  IA WHICH HAS N ROWS AND AN ARBITRARY NUMBER OF COLUMNS
C
      DIMENSION IA(1)
      IELM=IA(M*N-N+L)
      RETURN
C
      END

```

```

SUBROUTINE IMCPY(IA,IR,N,M,MS)
C
C   MCPY COPIES ENTIRE  N BY M MATRIX IA INTO N BY M MATRIX IR
C   MS   - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX IA (AND IR
C           0 - GENERAL
C           1 - SYMMETRIC
C           2 - DIAGONAL
C
C   DIMENSION IA(1),IR(1)
C
C   COMPUTE VECTOR LENGTH, IT
C   CALL LOC(N,M,IT,N,M,MS)
C   COPY MATRIX
C   DO 1 I=1,IT
C   1  IR(I)=IA(I)
C   RETURN
C
C   END

```

```

SUBROUTINE LOC(I,J,IR,N,M,MS)
C
C SUBROUTINE TO GENERATE VECTOR SUBSCRIPT FOR AN ELEMENT IN A MATRIX
C OF SPECIFIED STORAGE MODE.
C MS =0 SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N*M ELEMENTS
C IN STORAGE (GENERAL MATRIX)
C MS=1 SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N*(N+1)/2 IN
C STORAGE (UPPER TRIANGLE OF SUMMETRIC MATRIX). IF
C ELEMENT IS IN LOWER TRIANGULAR PORTION, SUBSCRIPT IS
C CORRESPONDING ELEMENT IN UPPER TRIANGLE.
C MS=2 SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N ELEMENTS
C IN STORAGE (DIAGONAL ELEMENTS OF DIAGONAL MATRIX).
C IF ELEMENT IS NOT ON DIAGONAL (AND THEREFORE NOT IN
C STORAGE), IR IS SET TO ZERO.
C
      IX=I
      JX=J
      IA=I-J
      IF (MS-1) 10,20,30
10      IRX=N*(JX-1)+IX
      GO TO 36
20      IF (IA) 22,24,24
22      IRX=IX+(JX*JX-JX)/2
      GO TO 36
24      IRX=JX+(IX*IX-IX)/2
      GO TO 36
30      IRX=0
      IF (IX-JX) 36,32,36
32      IRX=IX
36      IR=IRX
      RETURN
C
      END

```

```

C      SUBROUTINE MCPY(A,R,N,M,MS)
C      MCPY COPIES ENTIRE N BY M MATRIX A INTO N BY M MATRIX R
C      MS - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R)
C           0 - GENERAL
C           1 - SYMMETRIC
C           2 - DIAGONAL
C
C      DIMENSION A(1),R(1)
C
C      COMPUTE VECTOR LENGTH, IT
C      CALL LOC(N,M,IT,N,M,MS)
C      COPY MATRIX
C      DO 1 I=1,IT
C      1  R(I)=A(I)
C      RETURN
C
C      END

```

```

SUBROUTINE MMADD(N,ALPHA,A,BETA,B,C)
C
C SUBROUTINE TO FORM THE WEIGHTED SUM OF TWO ARRAYS
C OF DIMENSION N
C C=ALPHA*A+BETA*B
C
C DIMENSION A(1),B(1),C(1)
C
C DO 1 I=1,N
1 C(I)=ALPHA*A(I)+BETA*B(I)
C RETURN
C
END

```

```

SUBROUTINE MPRD(A,B,R,N,M,MSA,MSB,L)
C
C MPRD MULTIPLIES N BY M MATRIX A BY M BY L MATRIX B AND STORES THE
C PRODUCT INTO N BY L MATRIX R
C MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C 0 - GENERAL
C 1 - SYMMETRIC
C 2 - DIAGONAL
C MSB - SAME AS MSA EXCEPT FOR MATRIX B
C
C DIMENSION A(1),B(1),R(1)
C
C SPECIAL CASE FOR DIAGONAL BY DIAGONAL
MS=MSA*10.+MSB
IF(MS-22) 30,10,30
10 DO 20 I=1,N
20 R(I)=A(I)*B(I)
RETURN
C
C ALL OTHER CASES
30 IR=1
DO 90 K=1,L
DO 90 J=1,N
R(IR)=0
DO 80 I=1,M
IF(MS)40,60,40
40 CALL LOC(J,I,IA,N,M,MSA)
CALL LOC(I,K,IB,M,L,MSB)
IF(IA)50,80,50
50 IF(IB)70,80,70
60 IA=N*(I-1)+J
IB=M*(K-1)+I
70 R(IR)=R(IR)+A(IA)*B(IB)
80 CONTINUE
90 IR=IR+1
RETURN
C
END

```

```

SUBROUTINE MTRA(A,R,N,M,MS)
C
C MTRA TRANSPOSES N BY M MATRIX A TO FORM M BY N MATRIX R
C MS - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R)
C 0 - GENERAL
C 1 - SYMMETRIC
C 2 - DIAGONAL
C
C DIMENSION A(1),R(1)
C
C IF MS IS 1 OR 2, COPY A
C IF(MS) 10,20,10
10 CALL MCPY(A,R,N,N,MS)
C RETURN
C
C TRANSPOSE GENERAL MATRIX
20 IR=0
C DO 30 I=1,N
C IJ=I-N
C DO 30 J=1,M
C IJ=IJ+N
C IR=IR+1
30 R(IR)=A(IJ)
C RETURN
C
C END

```

SUBROUTINE SINV(N,AI,B,A,D)

```

C
C*****SUBROUTINE TO GENERATE THE INVERSE OF THE MATRIX AI
C   THE MATRICES AI AND B ARE STORED IN GENERAL FORM
C   INPUT MATRIX IS AI
C   OUTPUT INVERSE MATRIX IS B
C   N IS THE ORDER OF AI
C   D IS THE DETERMINANT OF AI
C
C       L - WORK VECTOR OF LENGTH N
C       M - WORK VECTOR OF LENGTH N
C
C       THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT
C       IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT
C       THE MATRIX IS SINGULAR.
C*****WITH MODIFICATIONS TO INPUT MATRIX IN VECTOR FORMAT
C
C
C   DIMENSION AI(20,20),B(20,20),A(400),L(20),M(20)
C   DIMENSION AI(1),B(1),A(1),L(20),M(20)
C
C   .....
C
C   IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C   C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C   STATEMENT WHICH FOLLOWS.
C
C   DOUBLE PRECISION A,D,BIGA,HOLD
C
C   THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C   APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C   ROUTINE.
C
C   THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C   CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS. ABS IN STATEMENT
C   10 MUST BE CHANGED TO DABS.
C   STORAGE OF AI ELEMENT IN A
C
C   KK=N*N
C   DO 5 J=1, KK
5   A(J)=AI(J)
C
C   .....
C
C   SEARCH FOR LARGEST ELEMENT
C
C   D=1.0
C   NK=-N
C   DO 80 K=1,N
C   NK=NK+N
C   L(K)=K
C   M(K)=K
C   KK=NK*K
C   BIGA=A(KK)
C   DO 20 J=K,N
C   IZ=N*(J-1)
C   DO 20 I=K,N
C   IJ=IZ+I
10  IF (ABS(BIGA)-ABS(A(IJ))) 15,20,20
15  BIGA=A(IJ)

```

```

      L(K)=I
      M(K)=J
20    CONTINUE
C
C      INTERCHANGE ROWS
C
      J=L(K)
      IF(J-K)35,35,25
25    KI=K-N
      DO 30 I=1,N
      KI=KI+N
      HOLD=-A(KI)
      JI=KI-K+J
      A(KI)=A(JI)
30    A(JI)=HOLD
C
C      INTERCHANGE COLUMNS
C
35    I=M(K)
      IF(I-K)45,45,38
38    JP=N*(I-1)
      DO 40 J=1,N
      JK=NK+J
      JI=JP+J
      HOLD=-A(JK)
      A(JK)=A(JI)
40    A(JI)=HOLD
C
C      DIVIDE COLUMN BY MINUS PIVOT (VALUE OF PIVOT ELEMENT IS
C      CONTAINED IN BIGA)
C
45    IF(BIGA) 48,46,48
46    D=0.0
      GO TO 150
48    DO 55 I=1,N
      IF(I-K)50,55,50
50    IK=NK+I
      A(IK)=A(IK)/(-BIGA)
55    CONTINUE
C
C      REDUCE MATRIX
C
      DO 65 I=1,N
      IK=NK+I
      HOLD=A(IK)
      IJ=I-N
      DO 65 J=1,N
      IJ=IJ+N
      IF(I-K)60,65,60
60    IF(J-K)62,65,62
62    KJ=IJ-I+K
      A(IJ)=HOLD*A(KJ)+A(IJ)
65    CONTINUE
C
C      DIVIDE ROW BY PIVOT
C
      KJ=K-N
      DO 75 J=1,N
      KJ=KJ+N
      IF(J-K)70,75,70

```

```

70  A(KJ)=A(KJ)/BIGA
75  CONTINUE
C
C      PRODUCT OF PIVOTS
C
D=D*BIGA
C
C      REPLACE PIVOT BY RECIPROCAL
C
A(KK)=1.0/BIGA
80  CONTINUE
C
C      FINAL ROW AND COLUMN INTERCHANGE
C
K=N
100 K=(K-1)
    IF(K) 150,150,105
105  I=L(K)
    IF(I-K) 120,120,103
108  JQ=N*(K-1)
    JR=N*(I-1)
    DO 110 J=1,N
        JK=JQ+J
        HOLD=A(JK)
        JI=JR+J
        A(JK)=-A(JI)
110  A(JI)=HOLD
120  J=M(K)
    IF(J-K) 100,100,125
125  KI=K-N
    DO 130 I=1,N
        KI=KI+N
        HOLD=A(KI)
        JI=KI-K+J
        A(KI)=-A(JI)
130  A(JI)=HOLD
    GO TO 100
150  LL=0
C      DO 151 J=1,N
        KK=N*N
        DO 151 J=1, KK
151  B(J)=A(J)
    RETURN
C
END

```

```

SUBROUTINE SYMTUG(XS,X,NX)
C
C PROGRAM CONVERTS A SYM. MATRIX VECTOR (IN SUPPRESSED SYM. STORAGE)
C WHOSE LENGTH IS  $NX*(NX+1)/2$ , INTO A GENERAL MATRIX VECTOR WHOSE
C LENGTH IS  $NX*NX$ .
C
C DIMENSION X(1),XS(1)
C
LL=0
DO 10 J=1,NX
DO 10 I=1,J
LL=LL+1
K=(J-1)*NX+I
M=(I-1)*NX+J
X(M)=XS(LL)
10 X(K)=XS(LL)
RETURN
C
END

```

B. 4 Input-Output Operations Package

This section presents listings and usage descriptions of subroutines which are used to perform input-output data operations.

```
SUBROUTINE IMXRNP(M,NA,NB,NENTRY)
SUBROUTINE IRANDP(ND,IA,IB,IC,ID,IE,IF,IG,NENTRY)
SUBROUTINE MXRNP(VA,NA,NB,NENTRY)
SUBROUTINE NAMRNP(M,NA,NB,NENTRY)
SUBROUTINE RANDP(NENTRY)
SUBROUTINE RANDPD(ND,DA,DB,DC,DD,DE,DF,DG,NENTRY)
```

```

SUBROUTINE IMXRNP(M,NA,NB,NENTRY)
C
C SUBROUTINE READS,PRINTS AND STORES INTEGER NA*NB MATRIX
C MATRIX IS STORED IN GENERAL FORM COLUMN BY COLUMN
C
C DIMENSION M(1)
C
1000 FORMAT(7I10)
1002 FORMAT(7I15)
C
C READ IN NA BY NB MATRIX ROW-WISE AND STORE INTO 1 DIMENSION
C VECTOR COLUMN-WISE.
C GO TO(1,1,2,4,2),NENTRY
C
1 J=NA*NB-NA+1
DO 15 I=1,NA
READ 1000, (M(K),K=I,J,NA)
15 J=J+1
C
GO TO(2,3,3,2),NENTRY
C
C PRINT NA BY NB MATRIX ROW-WISE
C CONTINUE
2 JJ=NA*NB-NA+1
DO 11 II=1,NA
IF(NENTRY-5)12,10,12
10 PUNCH 1000, (M(L),L=II,JJ,NA)
GO TO 11
12 PRINT 1002, (M(L),L=II,JJ,NA)
11 JJ=JJ+1
C
3 RETURN
C
C READ AND PRINT HEADING CARD BEFORE READING AND PRINTING MATRIX
C CALL RANDP(4)
4 GO TO 1
C
END

```

```

SUBROUTINE IRANDP(ND,IA,IB,IC,ID,IE,IF,IG,NENTRY)
C
C SUBROUTINE TO READ AND PRINT INTEGER DATA
C
C   DIMENSION IV(7)
C
C   1000 FORMAT(7I10)
C   1010 FORMAT(7I15)
C
C   GO TO(1,1,2,4),NENTRY
1  READ 1000,IA,IB,IC,ID,IE,IF,IG
C   GO TO(2,3,3,2),NENTRY
2  IV(1)=IA
   IV(2)=IB
   IV(3)=IC
   IV(4)=ID
   IV(5)=IE
   IV(6)=IF
   IV(7)=IG
   PRINT 1010,(IV(I),I=1,ND)
3  RETURN
C
4  CALL RANDP(4)
   GO TO 1
C
END

```

```

SUBROUTINE MXRNP(VA,NA,NB,NENTRY)
C
C SUBROUTINE READS AND/OR PRINTS THE NA*NB MATRIX VA WHICH IS STORED
C GENERAL FORM COLUMN BY COLUMN
C
C DIMENSION VA(1)
C
1000 FORMAT(7E10.0)
1002 FORMAT(7F15.6)
1003 FORMAT(7F10.4)
C
GO TO(1,1,2,4,2),NENTRY
C
C READ IN NA BY NB MATRIX ROW-WISE AND STORE INTO 1 DIMENSION
C VECTOR COLUMN-WISE.
1 J=NA*NB-NA+1
DO 15 I=1,NA
READ 1000, (VA(K),K=I,J,NA)
15 J=J+1
GO TO(2,3,3,2),NENTRY
C
2 CONTINUE
C PRINT NA BY NB MATRIX ROW-WISE
JJ=NA*NB-NA+1
DO 11 II=1,NA
IF(NENTRY-5)12,10,12
10 PUNCH 1003, (VA(L),L=II,JJ,NA)
GO TO 11
12 PRINT 1002, (VA(L),L=II,JJ,NA)
11 JJ=JJ+1
RETURN
C
3 RETURN
C
C READ AND PRINT HEADING CARD BEFORE READING AND PRINTING MATRIX
4 CALL RANDP(4)
GO TO 1
C
END

```

```

SUBROUTINE NAMRNP(M,NA,NB,NENTRY)
C
C SUBROUTINE READS,PRINTS AND STORES INTEGER NA*NB MATRIX
C OF FOUR CHARACTER NAMES
C MATRIX IS STORED IN GENERAL FORM COLUMN BY COLUMN
C
C DIMENSION M(1)
C
1000 FORMAT(1X,A4,1X,A4,1X,A4,1X,A4,1X,A4,1X,A4,1X,A4,1X,A4,
1 1X,A4,1X,A4,1X,A4,1X,A4,1X,A4)
1002 FORMAT(11X,A4,11X,A4,11X,A4,11X,A4,11X,A4,11X,A4,11X,A4)
C
GO TO(1,1,2,4),NENTRY
C
C READ IN NA BY NB MATRIX ROW-WISE AND STORE INTO 1 DIMENSION
C VECTOR COLUMN-WISE.
1 J=NA*NB-NA+1
DO 15 I=1,NA
READ 1000, (M(K),K=1,J,NA)
15 J=J+1
GO TO(2,3,3,2),NENTRY
C
C PRINT NA BY NB MATRIX ROW-WISE
2 CONTINUE
JJ=NA*NB-NA+1
DO 11 II=1,NA
PRINT 1002, (M(L),L=11,JJ,NA)
11 JJ=JJ+1
C
3 RETURN
C
C READ IN HEADING CARD BEFORE READING AND PRINTING M
4 CALL RANDP(4)
GO TO 1
C
END

```

```

      SUBROUTINE RANDP(NENTRY)
C
C      SUBROUTINE TO READ AND PRINT HEADING CARDS
C
      DIMENSION FNAME(8)
      DOUBLE PRECISION FNAME
C
1000  FORMAT(8A8)
1001  FORMAT(1H1)
1010  FORMAT(2X,A8,2X,A8,2X,A8,2X,A8,2X,A8,2X,A8,2X,A8)
1020  FORMAT(/X,A8,7X,A8,7X,A8,7X,A8,7X,A8,7X,A8,7X,A8)
C
      GO TO(1,2,3,4),NENTRY
C
1      PRINT 1001
2      READ 1000,(FNAME(I),I=1,8)
      PRINT 1000,(FNAME(I),I=1,8)
      RETURN
C
3      PRINT 1001
4      READ 1010,(FNAME(I),I=1,7)
      PRINT 1020,(FNAME(I),I=1,7)
      RETURN
C
      END

```

```

      SUBROUTINE RANDPD(ND,DA,DB,DC,DD,DE,DF,DG,NENTRY)
C
C      SUBROUTINE TO READ AND PRINT FLOATING POINT DATA
C
      DIMENSION DV(7)
C
      1000 FORMAT(7E10.0)
      1010 FORMAT(7F15.6)
C
      GO TO(1,1,2,4),NENTRY
C
      1  READ 1000,DA,DB,DC,DD,DE,DF,DG
        GO TO(2,3,3,2),NENTRY
      2  DV(1)=DA
        DV(2)=DB
        DV(3)=DC
        DV(4)=DD
        DV(5)=DE
        DV(6)=DF
        DV(7)=DG
        PRINT 1010,(DV(I),I=1,ND)
      3  RETURN
C
      4  CALL RANDP(4)
        GO TO 1
C
      END

```

TABLE OF REFERENCES

1. MacKinnon, D., P. Madden and P. Farrell, Optical Mirror Figure Control, MIT Draper Laboratory Report R-665, May 1970.
2. MacKinnon, D., Preliminary Software Development for Optical Mirror Figure Control, MIT Draper Laboratory Report R-685, December 1970.
3. MacKinnon, D., "Active Control of Primary Mirror Figure," NASA SP-233 (Proceedings, NASA Workshop on Optical Telescope Technology, MSFC, Huntsville, Alabama, April 29 - May 1, 1969).
4. Active Optical System for Spaceborne Telescopes, NASA CR-66297, Perkin-Elmer Corporation, Norwalk, Connecticut, October 14, 1966.
5. Active Optical System for Spaceborne Telescopes, Vol II, NASA CR-66489, Perkin-Elmer Corporation, Norwalk, Connecticut, December 7, 1967.
6. Development of an Active Optics Concept Using a Thin Deformable Mirror, Final Report, Perkin-Elmer Corporation, Norwalk, Connecticut, 1968.
7. Robertson, Hugh J., Development of an Active Optics Concept Using a Thin Deformable Mirror, NASA CR-1593, Perkin-Elmer Corporation, Norwalk, Connecticut, August 1970.
8. Crane, R., Advanced Figure Sensor, Final Report, NASA Electronics Research Center, September 1969.

9. Watt, G., "Actuators for Active Optics," NASA SP-233, (Proceedings, NASA Workshop on Optical Telescope Technology, MSFC, Huntsville, Alabama, April 29-May 1, 1969).
10. Robertson, H., Real Time Optical Figure Sensor, Perkin-Elmer Report 10463, June 1971.
11. Robertson, H., Real Time Figure Sensor Operations and Maintenance Manual, Perkin-Elmer Report 10464, June 1971.
12. Robertson, H., 20-Inch Deformable Active Mirror System, Perkin-Elmer Report 10482, June 1971.
13. Active Optical System for Spaceborne Telescopes, Perkin-Elmer Report 8525, October 1966.
14. Creedon, J., Discrete Control of Linear Distributed Systems with Application to the Deformable Primary Mirror, Ph.D. Thesis, The University of Rhode Island, 1970.
15. Nasa Memorandum S & A-ASTR-MA-71-104.
16. Soosaar, K., Design of Optical Mirror Structures, MIT Draper Lab Report R-673, January, 1971.
17. ICES-STRUDL II, Engineering Users Manual, Volume 2, 2nd Edition, MIT Civil Engineering Department, June, 1971.

TABLE OF SYMBOLS

Variable	Definition	Software *** Notation
A	deformation-force or position-position matrix	AM
A_m	actual deformation-force or position-position matrix of the mirror	AMM
A_{mr}	reduced version of A_m	*
A_r	reduced A matrix	*
A_{rr}	doubly reduced A matrix	*
a_s	ambiguity sensor outputs	ASV
g_{tilt}	tilt control system gain	GTILT
J_a	actual rms figure error	*
J_{fs}	figure sensor performance index	*
J_m	measured rms figure error	PINDEX
K	feedback gain matrix	GAINM
K_a	actuator error gain	GA
K_g	generalized linear control system gain matrix	GAINM
K_ℓ	simplified linear control system gain matrix	GAINM
$k_{\ell d}$	lead screw gain	
K_o	linear optimal control system gain matrix	GAINM
k_{sa}	integral of actuator output gain	GB
m_c	desired figure actuator outputs	UFV

* Stored temporarily.

** The diagonal elements of the diagonal matrix B are stored in the array WGTV.

*** Corresponding sigma 2 variable names may be generated by prefixing Q to floating point variable names or suffixing Q to fixed point variable names.

Variable	Definition	Software Notation
$\hat{m}_c$	scaled value of m_c	QUFV
m_g	scalar gains relating actuator commands	ASCALV
m_o	measured figure actuator outputs	UFAV
m_{max}	maximum actuator command magnitude	UFMAX
m_m	measured actuator output	UFAV
n	number of measurement positions	N
N	motor gear ratio	
n_{ma}	number of actuator slope measurements to be performed and averaged	NMEAS*
n_{mf}	number of A_r matrices to be measured experimentally and averaged	NMEAS*
n_{meas}	number of measurement samples at each measurement location	NMEAS*
n_{mint}	number of cycles between measurements	NMINT
n_{pos}	number of cycles allotted for figure sensor transients to decay	NPOS
n_r	number of actuators	NR
n_{tilt}	number of measurement position increments during tilt alignment	NTILT
n_{tims}	total number of cycles executed when EAMCS is called	NTIMS
n_{wait}	number of cycles during which the actuators are operative	NWAIT
n_ω	number of control cycles velocity pulse is applied $t_\omega = n_\omega \Delta t$	NCVEL
t	time	T
t_a	actuator time constants	TACTV

* Ambiguity does not occur in the software because NMEAS is not in common.

Variable	Definition	Software Notation
t_f	figure sensor filter time constant	FSTFLT
t_s	control computation cycle time	TSENS
t_w	velocity command pulse width	TCVEL
W	performance index weighting factors	WGTV
x	x figure error position coordinate	XFSV
x_d	initial figure disturbance with zero actuator input	XFDV
x_f	actual figure error	XF
x_{fr}	reduced measured figure error vector	XFRV
x_{fp}	processed figure error data	XFV
x_{fm}	measured figure error	XF
x_{sf}	vector of the sums of the figure measurements at the measurement locations	QDUMVB
x_{ssf}	vector of the sums of the squares of the figure measurements at the measurement locations	QDUMVC
y	y figure error position coordinate	YFSV

GREEK SYMBOLS

α_m	slew control system command	UFV
β_{ab}	ambiguity correction value	AMBIG
β_{as}	mirror model matrix scale factor	ASCALE
β_f	figure sensor phase detector filter output	FSFLT
β_{ft}	threshold level on the rms figure error	SIGLIM
β_g	scalar gain factor	GAIN(1)

Variable	Definition	Software Notation
β_k	control law gain factor	GAIN(1)
β_{lf}	stored value of β_{mf}	
β_{mf}	mean values of the figure measurements	XFMN
β_{mrf}	rms values of the figure measurements	XFSIG
β_{ms}	matrix inversion scale factor	AIMSCL
β_{nf}	noise input to the figure sensor phase detector	FSNOIS
β_p	figure sensor phase detector output	FSPDO
β_{sd}	second-order ambiguity sensor output coefficient	BSDP
β_{sm}	maximum ambiguity sensor output	BSMP
β_{sw}	computer switching boundary	XFSW
β_t	threshold value for rms measurement error	SIGLIM
β_{ts}	threshold value and ambiguity factor computation	BTS
β_{xa}	actual figure sensor error	UFV
β_{xf}	figure error input	XF
β_z	interpolation factor	SLPMN
β_w	amplitude of the position control system velocity drive pulse	CVEL
γ_a	actuator input transition matrices	AGAMV
γ_f	figure sensor filter input transition matrix	FSGAM
Δt	real time control system cycle time	TSENS
δ_{aa}	actuator perturbation for actuator test	DACT *
δ_{af}	actuator perturbation for mirror model generation	DACT *

* Not in common.

Variable	Definition	Software Notation
λ	figure sensor laser operating wavelength	
σ_f	rms figure error noise level	FSNSIG
ν_f	figure sensor noise	FSNOIS
ϕ_a	actuator state transition matrices	APHIV
ϕ_f	figure sensor filter transition matrix	FSPHI

Software ** Notation	Definition	Variable
AGAMV	actuator input transition matrices	γ_a
AIMSCL	matrix inversion scale factor	β_{ms}
AM	deformation-force or position- position matrix	A
AMBIG	ambiguity correction value	β_{ab}
AMM	actual deformation-force or position- position matrix of the mirror	A_m
APHIV	actuator state transition matrices	ϕ_a
ASCALE	mirror model matrix scale factor	β_{as}
ASCALV	scalar gains relating actuator commands	m_g
ASV	vector of ambiguity sensor outputs	a_s
BSDP	second-order ambiguity sensor output coefficient	β_{sd}
BSMP	maximum ambiguity sensor output	β_{sm}
BTS	threshold value and ambiguity factor computation	β_{ts}
CPM	stored PARV modifications	
UFV	amplitude of the position velocity drive pulse	β_w
CXM	stored XV modifications	
DACT*	actuator perturbation for actuator test	δ_{aa}
DACT*	actuator perturbation for mirror model generator	δ_{af}

* Ambiguity does not occur in the software because DACT is not in common.

** Variable names containing Q may be referenced by deleting the Q and referencing the resulting name (i. e., $NQ \rightarrow N$)

Software Notation	Definition	Variable
DT	real time control system cycle time	Δt
DTE	time round off correction factor	
DTNOIS	stochastic noise generation interval	
DTPLOT	plot data storage interval	
FSCALE	figure sensor measurement scale factor	
FSFLTO	figure sensor phase detector filter output	β_f
FSGAM	figure sensor filter input transition matrix	γ_f
FSPHI	figure sensor filter state transition matrix	ϕ_f
FSPDO	figure sensor phase detector output	β_p
FSNOIS	noise input to the figure sensor phase detector	β_{nf}
FSNSIG	rms figure error noise level	σ_f
FSTFLT	figure sensor filter time constant	t_f
GA	actuator error gain	K_a
GAINM	feedback gain matrix	K
GAINM	simplified linear control system gain matrix	K_l
GAINM	linear optimal control system gain matrix	K_o
GAINM	generalized linear control system gain matrix	K_g
GAIN(1)	scalar gain factor	β_g
GB	integral of actuator error gain	k_{sa}
GTILT	tilt control system gain	g_{tilt}
ICPM	stored integer parameter modifications	

Software Notation	Definition	Variable
IMODV	plot scaling control vector	
IPLOTV	plotted elements of the data transfer vector XV	
IRAND	initial starting value for random number generator	
IPARV	input data storage	
JCPV	modified elements of PARV	
JCXV	modified elements of XV	
JICPV	modified elements of IPARV	
LACTV	actuator position assignment vector	
LREFAV	segmented mirror actuator assignments	
MODOP	control system type identifier	
MODV	vector of operating modes	
MSEQV	figure sensor scan sequence	
N	number of figure measurement points	n
NCPV	number of modified parameter values	
NCTILT	number tilt control system control system operating cycles	
NCVEL	number of control cycles velocity pulse is applied $t_{\omega} = n_{\omega} \Delta t$	n_{ω}
NCXV	number of modified initial conditions	
NFLGA- NFLGD	transfer identification variables	
NHC	number of step size halvings	
NHM	maximum number of step size halvings	
NIC	number of successful iterations	

Software Notation	Definition	Variable
NICPV	number of integer parameters to be changed	
NIM	maximum number of iterations	
NMAG	magnetic storage device assignment	
NMCPV	names of modified parameters	
NMCXV	names of modified initial conditions	
NMEAS*	number of measurement samples at each measurement location	n_{meas}
NMEAS*	number of actuator slope measurements to be performed and averaged	n_{ma}
NMEAS*	number of A_r matrices to be measured experimentally and averaged	n_{mf}
NMICPV	names of modified integer parameters	
NMINT	number of cycles between measurements	n_{mint}
NPLOTV	number of plotted variables	
NPOS	number of cycles allotted for figure sensor transients to decay	n_{pos}
NPUNCH	device assignment for punched output	
NR	number of figure actuators	n_r
NRUN	run identification number	
NRUNC	number of completed runs	
NRUNM	maximum number of runs	
NSNSWT	sense switch assignment	

* Ambiguity does not occur in the software because NMEAS is not in common.

Software Notation	Definition	Variable
NTILT	number of measurement position increments during tilt alignment	n_{tilt}
NTIMS	total number of cycles executed when EAMCS is called	n_{tims}
NTYO	number of control computations between monitor data output	
NTYPE	operator display assignment for manual simulation control	
NTYPI	remote control input device assignment	
NTYPO	remote control output device assignment	
NWAIT	number of cycles during which the actuators are operative	n_{wait}
PINDEX	measured rms figure error	J_m
PLOTV	plotted data vector	
PSCALE	measurement position scale factor	
QDUMVB	vector of the sums of the figure measurements at the measurement locations	x_{sf}
QDUMVC	vector of the sums of the squares of the figure measurements at the measurement locations	x_{ssf}
SCALV	plotted data scales	
SIGLIM	threshold level on the rms figure error	β_{ft}
SLPMN	interpolation factor	β_z
SMXV	segmented mirror actuator x coordinate values	
SMYV	segmented mirror actuator y coordinate values	

Software Notation	Definition	Variable
SPARV	input data storage	
SXV	input data storage	
T	time	t
TACTV	actuator time constants	t_a
TEND	simulation run termination time	
TPRNT	interval between simulation output print	
TSENS	figure control computation cycle time	t_s
TVEL	velocity command pulse width	t_w
UFAV	measured figure actuator outputs	m_m
UFMAX	maximum control command magnitude	m_{max}
UFV	desired figure actuator outputs	m_c
WGTV	performance index weighting factors	w
XF	actual figure error	x_f
XFDV	initial figure error ($m_m = 0$)	x_d
XFV	measured figure error	x_{fm}
XFRV	reduced measured figure error vector	x_{fr}
XFMN	mean values of the figure measurements	β_{mf}
XFSIG	rms values of the figure measurements	β_{mrf}
XFSV	x figure error position coordinate	x
XFSW	nearest phase switching boundary	β_{sw}
XFV	actual figure sensor error	β_{xa}
XFV	processed figure error data	x_{fp}
XV	data transfer vector	
YFSV	y figure error position coordinate	y