

UNCLASSIFIED

AD NUMBER

ADB000284

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited.

FROM:

Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; OCT 1974. Other requests shall be referred to Federal Aviation Administration, Supersonic Transport Office, 800 Independence Avenue, SW, Washington, DC 20590.

AUTHORITY

faa ltr, 26 apr 1977

THIS PAGE IS UNCLASSIFIED

THIS REPORT HAS BEEN DELIMITED
AND CLEARED FOR PUBLIC RELEASE
UNDER DOD DIRECTIVE 5200.20 AND
NO RESTRICTIONS ARE IMPOSED UPON
ITS USE AND DISCLOSURE.

DISTRIBUTION STATEMENT A

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED.

Report No. FAA-SS-73-2-2

①

FC

REDUNDANT FLIGHT-CRITICAL CONTROL SYSTEM EVALUATION

Analog and Digital Systems Performance Comparison

M. L. Beattie, J. W. Berwick, R. H. Fulton,
K. A. Hill, A. Maeshiro, J. E. Templeman, D. E. Wegemer

Boeing Commercial Airplane Company
P.O. Box 3707
Seattle, Washington 98124

AD B 0 0 0 2 8 4



D6-60286-2

October 1974

FINAL REPORT

Task IV

DDC
RECEIVED
DEC 5 1974
E

go

AD No. FILE COPY

Approved for U.S. Government only. Transmittal of this document outside the U.S. Government must have prior approval of the Supersonic Transport Office

Prepared for
FEDERAL AVIATION ADMINISTRATION
Supersonic Transport Office
800 Independence Avenue, S.W.
Washington, D.C. 20590

FORMER BY	
DATE	Write Section <input type="checkbox"/>
NO.	Write Section <input checked="" type="checkbox"/>
DATE	<input type="checkbox"/>
BY	
REASON FOR LIABILITY ORDER	
DATE	REASON SPECIAL
13	

12/277p.

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. 18 FAA-SS 73-2-2 ✓	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle 6 REDUNDANT FLIGHT-CRITICAL CONTROL SYSTEM EVALUATION, Analog and Digital Systems Performance Comparison.		5. Report Date 11 Oct 1974 ✓	6. Performing Organization Code
7. Author(s) 10 M. L. Beattie, J. W. Berwick, R. H. Fulton, K. A. Hill, A. Maeshiro, J. E. Templeman, D. E. Wegemer		8. Performing Organization Report No. 14 D6-60286-2 ✓	
9. Performing Organization Name and Address Boeing Commercial Airplane Company P.O. Box 3707 Seattle, Washington 98124 ✓		10. Work Unit No.	
12. Sponsoring Agency Name and Address Federal Aviation Administration Supersonic Transport Office 800 Independence Avenue S.W. Washington, D.C. 20590 ✓		11. Contract or Grant No. 15 DOT-FA72WA-2893 ✓	
5. Supplementary Notes DOT/SST FCD task technical monitors: Messrs. Sid Blatt and M. H. Lowe (ARD-500)		13. Type of Report and Period Covered 9 Final Report, on Phase 2, Task 4, 4	
<p>Abstract</p> <p>The U.S. SST prototype commercial airliner under development from 1967 to 1971 employed redundant flight-critical control systems as an essential part of the airplane's airworthiness. The flight control system electronics were analog for the flight-critical stability augmentation functions and digital for the automatic flight control functions. The digital system, through an automated preflight test function, also served to establish the integrity of the flight-critical elements. The SST program was terminated before these systems became operational. This study deals with the mechanization of redundant electronic systems. Specifically, the study evaluates analog and digital electronic designs for implementing a triplex fail-operational flight-critical control system. The primary subjects studied were analog and digital systems' multiple failure fail-operational capabilities and preflight integrity check requirements. This document deals with analytical and laboratory performance evaluations of the systems studied. Specific areas covered include basic filter processing, closed loop operation, fault tolerant performance, sensor signal selection/failure detection, and digital computer timing and memory relationships.</p> <p>21 Report on SST Technology Follow-up Program. and See also AD-913/3654 AD-B 004 2814.</p>			
17. Key Words Redundant flight controls Digital flight control system Fault tolerant computers Digital control system software Flight-critical control system		18. Distribution Statement Approved for U.S. Government only. Transmittal of this document outside the U.S. Government must have prior approval of the Supersonic Transport Office.	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 270	22. Price

-a- 390 145 ✓

mit

PREFACE

This document is one of a series of documents reporting on work conducted under task IV of the SST Technology Follow-On Program, phase II. Task IV studies are an extension of the Flight Controls Technology studies conducted during phase I of the SST Follow-On Program and were given the subject title Flight Controls Development (FCD). The FCD study results are covered within the reports listed below:

- Report No. FAA-SS-73-1, *SST Longitudinal Control System Design and Design Processes (Hardened Stability Augmentation Design)*
- Report No. FAA-SS-73-2-1, *Redundant Flight-Critical Control System Evaluations (Analog and Digital Systems Descriptions)*
- Report No. FAA-SS-73-2-2, *Redundant Flight-Critical Control System Evaluations (Analog and Digital Systems Performance Comparisons)*
- Report No. FAA-SS-73-3, *Redundant Flight-Critical Control System Evaluations (Analog and Digital Systems Failure Analyses and Preflight Test Designs)*

The evaluation of the redundant analog and digital systems described herein was not an attempt to determine which was the optimum system. Rather, the analysis and laboratory tests were to provide an insight into the technical strength or weakness of each approach as related to flight-critical control system applications. Therefore, the readers must derive their own conclusions, along with those presented, in determining which system features most benefit their application needs.

As a special acknowledgment, the Boeing Commercial Airplane Company wishes to recognize the diligent support provided by the on-site engineering personnel from the General Electric Company. Messrs. R. E. Blanford and L. E. Fairbanks are especially commended for their outstanding role in providing hardware/system operational support during the laboratory testing of the ICPS and WWCS equipment.

CONTENTS

	Page
1.0 INTRODUCTION	1
2.0 FCD TASK APPLICATION MODEL DESCRIPTION	3
2.1 SST Pitch Axis Control System	3
2.2 FCD Task Laboratory Test Configuration	4
2.2.1 Analog Computer Simulation	4
2.2.2 Hydromechanical Test Stand	5
2.2.3 Flight Control Subsystem Functions	5
3.0 APPLICATION MODEL IMPLEMENTATION—ANALOG AND DIGITAL	9
3.1 Analog System	9
3.2 Incremental Digital System	10
3.2.1 HSAS Implementation Using ICPS	10
3.2.2 Gain Schedule Function Implementation Using ICPS	15
3.3 Whole-Word Digital System	16
3.3.1 WWCS Control Law Software Structure	16
3.3.2 HSAS/ECSS/CWS Implementation Using WWCS	18
3.3.3 Gain Schedule Function Implemented Using WWCS	23
3.4 ICPS/WWCS Memory and Timing Comparison	24
4.0 APPLICATION MODEL ANALOG/HSAS/WWCS LABORATORY TESTS	29
4.1 Basic ICPS and WWCS Hardware Characteristics	29
4.1.1 ICPS Hardware Characteristics	29
4.1.2 WWCS Hardware Characteristics	31
4.2 HSAS Filter Implementation Laboratory Evaluation	32
4.2.1 Frequency Response Comparison	33
4.2.2 Step Response Comparison	33
4.2.3 Noise Characteristic Comparison	34
4.3 Redundant HSAS Performance	34
4.3.1 HSAS Response—Triple-Channel Configuration	34
4.3.2 Triple-Channel Response—Pitch Rate Sensor Failure	35
4.3.3 HSAS Failure Monitor Evaluation	36
4.4 Conclusions—HSAS Laboratory Tests	38
5.0 SPECIAL STUDIES—REDUNDANT DIGITAL SYSTEM FUNCTIONS	39
5.1 Signal Selection/Failure Detection	39
5.1.1 SSFD Algorithm Design	39
5.1.2 SSFD Software Implementation	44
5.1.3 SSFD Points—Analog, ICPS, and WWCS	47
5.1.4 SSFD Placement—Mission Reliability Analysis	49
5.2 Digital Filter Implementation	53
5.2.1 ICPS Second-Order Filter Evaluation	53
5.2.2 WWCS Digital Filter Implementation Study	60

CONTENTS—Concluded

	Page
5.2.3 WWCS Second-Order Filter Evaluation Using Bilinear Transformation	65
5.2.4 Digital Filter Study Conclusions	70
5.3 Digital System Computational Overflow	71
5.3.1 ICPS Computational Overflow Protection	72
5.3.2 WWCS Computational Overflow Protection	74
5.4 WWCS Servo Transmitter/Receiver Unit Evaluation	80
5.4.1 Servo Loop Closure	81
5.4.2 Redundant Servo Actuator Equalization	83
6.0 SOFTWARE DEVELOPMENT AND CONTROL	85
6.1 FCD Task Software Development	85
6.1.1 ICPS Programming	86
6.1.2 WWCS Programming	87
6.2 Projected Software Development and Control Requirements	91
6.2.1 Software Requirements	92
6.2.2 Software Acceptance Test	97
7.0 ICPS FLIGHT TEST EVALUATION	101
7.1 AGCS Flight Test Configuration	101
7.2 Flight Control Subsystem	103
7.3 Flight Test Results	106
7.3.1 Failure Monitoring	106
7.3.2 Autoland Performance	117
REFERENCES	123
APPENDIX A—Signal Selection/Failure Detection Function Characteristic Analyses	125
A.1 Signal Selection Noise Considerations	125
A.2 Failure Detection Considerations	128
A.3 SSFD Failure Rate Considerations	129
APPENDIX B—Bilinear Transformation Linearity Characteristic	131

FIGURES

No.	Page
2-1 Laboratory Test Configuration	135
2-2 Electric Command Servo (ECS)	136
2-3 HSAS Functional Block Diagram	137
2-4 HSAS Flight Control System Block Diagram	138
2-5 ECSS Functional Block Diagram	139
2-6 ECSS Gain Schedule Functions	140
2-7 Pitch CWS Functional Block Diagram	141
2-8 CWS Gain Schedule Functions	142
3-1 Breadboard HSAS Filters	143
3-2 Laboratory Test Configuration of Analog HSAS	144
3-3 Analog System Oscillatory Failure Detector (OFD)	145
3-4 ICPS HSAS Scaling (Worst Case) Map	146
3-5 Second Order Lead-Lag Filter General Algorithm	147
3-6 HSAS Filter B Algorithm Map	148
3-7 HSAS Algorithm Map	149
3-8 ICPS Gain Schedule Implementation	151
3-9 Gain Schedule $K_V\phi$ Algorithm Map	153
3-10 $K_V\phi$ Accuracy X-Y Plot (Lab Test)	155
3-11 WWCS Flight Software Timing Sequency	156
3-12 HSAS Digital System Diagram	157
3-13 Computational Sequence of Level II Flight Control Software (HEC)	159
3-14 HSAS Supervisor Macro-Flowchart	161
3-15 ECSS Supervisor Macro-Flowchart	163
3-16 CWS Supervisor Macro-Flowchart	165
4-1 Math Model for ICPS and WWCS Basic Hardware	166
4-2 Sampler Output as a Function of Input Sampling Frequency (Aliasing)	167
4-3 Basic ICPS Hardware Frequency Response	168
4-4 ICPS Empirical Slew Rate Gain Characteristic	169
4-5 Linearity Characteristics of ICPS and WWCS Hardware	170
4-6 Basic WWCS Hardware Frequency Response	171
4-7 HSAS Filter	172
4-8 Analog and ICPS HSAS Frequency Response	173
4-9 WWCS HSAS Filter Frequency Response	174
4-10 HSAS Filter Step Response	175
4-11 Comparison of HSAS Filter Residual Noise Characteristics	176
4-12 Redundant Data Path Block Diagram	177
4-13 Comparison of HSAS Performance in Triple Operation	178
4-14 Comparison of HSAS Performance in Triple Operation With Pitch Rate Sensor Offsets	179
4-15 HSAS Performance With One Pitch Rate Sensor Failure	180
4-16 ICPS and WWCS HSAS Response With Passive Pitch Rate Sensor Failure	181
5-1 Median Selection and Average Output for Three Out-of-Phase Input Signals	182
5-2 Median Selection and Average Algorithm Outputs With Random Noise Inputs	183

FIGURES—Continued

No.	Page
5-3 Signal Selection Output to Input Error Characteristics	184
5-4 Median-Select/Average-of-Two and Average-of-Three/Average-of-Two SSFD Algorithm	185
5-5 Threshold Amplitude of Cross Input Failure Detector for $TH_1 = TH_2$ and $T_1 = 10T_2$	186
5-6 Threshold Amplitude of Cross Input Failure Detector for $TH_1 > TH_2$ and $T_1 = 10T_2$	187
5-7 Lag Equalized Median Selection SSFD Algorithm	188
5-8 Compensated Limited Average SSFD Algorithm	189
5-9 Analog System SSFD Configuration	190
5-10 Oscillatory Failure Detection Characteristics of ICPS Variable SSFD	191
5-11 Single Channel Configuration	192
5-12 Brick-Wall Configuration	192
5-13 ICPS Voting/Monitoring Schematic Diagram	193
5-14 WWCS Voting/Monitoring Schematic Diagram	194
5-15 Probability of Mission Failure (HSAS Function)	195
5-16 Probability of Mission Failure (ECSS Function)	196
5-17 Redundant System Reliability as Function of Single Channel and SSFD Reliabilities	197
5-18 ICPS Second Order Filter Frequency Response $\zeta = 0.3$	198
5-19 ICPS Second Order Filter Frequency Response $\zeta = 1.0$	199
5-20 Slew Rate Effect on Second Order Filter Response	200
5-21 Second Order Filter Bilinear Transformation Block Diagram	201
5-22 Second Order Filter Bilinear Transformation With Steady State Error Compensation	202
5-23 Continuous "S" Domain Second Order Filter	203
5-24 Rectangular Integration Second Order Filter	204
5-25 Trapezoidal Integration Second Order Filter	205
5-26 HSAS Filters	206
5-27 Amplitude Error Response of Complete HSAS Filter Set	207
5-28 Phase Error Response of Complete HSAS Filter Set	208
5-29 Amplitude Error Response of HSAS Filter A	209
5-30 Phase Error Response of HSAS Filter A	210
5-31 Amplitude Error Response of HSAS Filter B	211
5-32 Phase Error Response of HSAS Filter B	212
5-33 HSAS Filter A Steady State Response	213
5-34 HSAS Filter B Steady State Response	214
5-35 WWCS Low Frequency Filter Response for $\xi_D = 0.3, \tau = 0.049$ Sec	215
5-36 WWCS Low Frequency Filter Response for $\xi_D = 0.3, \tau = 0.018$ Sec	216
5-37 WWCS Low Frequency Filter Response for $\xi_D = 0.3, \tau = 0.006$ Sec	217
5-38 WWCS Low Frequency Filter Response for $\xi_D = 1, \tau = 0.049$ Sec	218
5-39 WWCS Low Frequency Filter Response for $\xi_D = 1, \tau = 0.018$ Sec	219
5-40 WWCS Low Frequency Filter Response for $\xi_D = 1, \tau = 0.006$ Sec	220
5-41 Theoretical Frequency Shift Between Continuous Time Domain and Bilinear Implementation	221

FIGURES—Continued

No.	Page
5-42 WWCS Filter Frequency Response for $\xi_D = 0.3, \omega_D = 50$ Rad/Sec	222
5-43 WWCS Filter Frequency Response for $\xi_D = 1, \omega_D = 50$ Rad/Sec	223
5-44 WWCS Filter Frequency Response for $\xi_D = 0.3, \omega_D = 100$ Rad/Sec	224
5-45 WWCS Filter Frequency Response for $\xi_D = 1, \omega_D = 100$ Rad/Sec	225
5-46 HSAS Filter a Steady State Response Using Bilinear Transformation With Compensation (WWCS)	226
5-47 HSAS Filter B Steady State Response Using Bilinear Transformation With Compensation (WWCS)	227
5-48 Filter A Steady State Response for $\tau = 0.049$ Sec Using Bilinear Transformation Without Compensation (WWCS)	228
5-49 Filter A Steady State Response for $\tau = 0.018$ Sec Using Bilinear Transformation Without Compensation (WWCS)	229
5-50 ICPS Overflow Characteristics	230
5-51 Analog System Saturating Limiter	231
5-52 ICPS X Register Example	232
5-53 Baseline Response Without Overflow	233
5-54 Overflow at Input to HSAS D Filter Without the Overflow Routine	234
5-55 Overflow at Input to HSAS D Filter With the Overflow Routine Active	235
5-56 Overflow Within Recursive HSAS D Filter With Overflow Routine Active	236
5-57 STRU Laboratory Test Configuration	237
5-58 STRU Study Servo Loop Frequency Response Comparisons	238
5-59 STRU/Computer Unit Data Flow Diagram	239
5-60 Redundant Servo Equalization Test Configuration	240
5-61 ECS Steady State Response With and Without Equalization	241
5-62 Airplane Response With and Without Equalization	242
6-1 WWCS Memory Assignment	243
6-2 Experimental Programs Log	244
6-3 Operational Program Log	245
7-1 AGCS Concept	246
7-2 AGCS Interfaces	247
7-3 Automatic Flight Control System	248
7-4 Flight Control Computer Pallet	249
7-5 ICP-723 Signal Selection Scheme	250
7-6 ICP-723 Signal Voting Configuration	251
7-7 AGCS ICP-723 Failure Annunciation	252
7-8 FCI Pallet	253
7-9 Modification to 737 Power System	254
7-10 AGCS Mode Select Panel	255
7-11 Delta Track Angle Monitoring Scheme	256
7-12 Radio Altimeter Monitor	257
7-13 Radio Altimeter Signal Selection	258
7-14 Servo Monitor Schematic	259
7-15 Aileron/Elevator Servo Monitor Failure Detection Scheme	260
7-16 Servo Monitor Schematic	261

FIGURES—Concluded

No.	Page
7-17 Localizer Ramp Failure	262
7-18 Glide Slope Ramp Failure	263
7-19 Roll Autoland Localizer Fly-Off Maneuver	264
7-20 Pitch Autoland Glide Slope Fly-Off Maneuver	265
7-21 737 AGCS Pitch Axis Autoland	266
7-22 737 AGCS Lateral Axis Autoland	267
7-23 737 AGCS Airspeed Control Law	268
7-24 Roll Autoland Approach	269
7-25 Pitch Autoland Approach	270

TABLES

No.		Page
3-1	ICPS/WWCS Real Time Usage (MSEC) [HSAS, ECSS, and CWS Functions] . . .	27
3-2	ICPS/WWCS Program Memory Usage (Words) [HSAS, ECSS, and CWS Functions]	27
4-1	Failure Mode Response	37
5-1	SSFD Performance Comparison	45
5-2	SSFD Memory and Computational Time Comparison	46
5-3	Algorithm Coefficients for ICPS Second Order Filter Study	59
5-4	Second Order Filter Bilinear Transformation Coefficients	67
5-5	Four-Bit Processor Variable Range (Binary/Decimal)	76
5-6	WWCS (MSP-701) Instructions Which Can Cause Computational Overflow	78
5-7	Synopsis of Overflow Routine Strategy	79
5-8	Execution Times for Overflow Protection Routines	79
6-1	WWCS Program Symbology	95
7-1	Input Sensors	105
7-2	Signal Selection and Failure Detection From Data	108
7-3	System/Radio Altimeter Failure Status	111
7-4	Failure Indications Analysis	114
7-5	Summary of Autolands	121

ABBREVIATIONS

ABS	absolute
A/D	analog to digital
ADEDS	advanced electronic display system
AF	arithmetic fault
AFCS	automatic flight control system
AGCS	advanced guidance and control system
APD	approach progress display
APU	auxiliary power unit
ARINC	Aeronautical Radio, Inc.
BIT	built-in-test
CIU	computer interface unit
CLA	compensated limited average
CMCU	computer monitor and control unit
CPL	complement
CPU	central processing unit
CWS	control wheel steering
D/A	digital to analog
DAS	data acquisition system
DFD	dynamic failure detector
DISCR	discrete
DIV	divide
DOT	Department of Transportation
ECS	electric command servo

ECSS	electric command and stability system
EXEC	executive software program
FAA	Federal Aviation Agency
FCC	flight control computer
FCD	flight controls development
FCDLIB	flight control development library
FCI	flight control interface
G/S	glide slope
GSE	groundspeed error
HDOT	vertical speed
HEC	HSAS, ECSS, and CWS software program
HSAS	hardened stability augmentation system
ICP-723	incremental control processor (built in 1972)
ICPS	incremental control processor (triplex) subsystem
ILS	instrument landing system
I/O	input/output
LDA	LOOK data array
LOC ENG	localizer engaged
LRU	line replaceable unit
MCP-701	modular control processor (central processing unit)
MCP-703	modular control processor (triplex subsystem, built in 1973)
MEA	maneuver equation averages
MLV	majority logic voter
MSL	maximum scaling levels

MSP	mode select panel
MU	machine units
OFD	oscillatory failure detector
PCU	power control unit
q_c	dynamic pressure
REDMAN	redundancy management software program
ROM	read only memory
RSFS	research support flight system
SA	sign adder
SFD	static failure detector
SLI	synchronous logic interface
SSCU	system status and control unit
SSFD	signal selection/failure detection
SST	supersonic transport (U.S.)
STRU	servo transmitter/receiver unit
TKA	track angle
WWCS	whole-word computer (triplex) subsystem

1.0 INTRODUCTION

In order to refine aircraft handling qualities and improve aircraft mission performance, aircraft designers are developing a new generation of airborne vehicles that employ electronic subsystems as a basic part of the aircraft design. Such electronic subsystems, independently or collectively, are being used to replace elements of the heretofore mechanical primary control system, drive control surfaces to augment the basic airframe aerodynamic stability, provide desired handling quality improvements, or control the vehicle flightpath with greater precision than obtainable using the man/machine interface. These applications have varying levels of criticality relative to meeting aircraft safety or mission reliability requirements. To meet set safety or functional reliability requirements, the systems are mechanized with redundant paths of electronic computational hardware.

The flight controls development study (task IV of the SST Technology Follow-On phase II) deals with the mechanization of redundant electronic subsystems. Specifically, the study compares analog and digital electronic designs for implementing a triple-redundant flight-critical system. The application model is based on the stability augmentation systems under development for the U.S. supersonic transport at the time of the SST program cancellation in 1971.

The FCD task statement of work, however, was not directed at developing a flight control system for a given vehicle but was drafted to permit technology investigations into selected areas of triple-channel, fail-operational, analog, and digital system designs. Interface requirements; operational performance, and failure protection mechanizations were fundamental areas to be investigated. Hardware/software relationships with respect to a digital system failure analysis were a special item of interest. Laboratory testing of a baseline analog and candidate digital systems was a part of the FCD statement of work.

The SST hardened stability augmentation system (HSAS) was selected as the baseline function for deriving analytical and laboratory comparison data between analog and digital design approaches. In addition, functions within the SST electric command and stability system (ECSS) and automatic flight control system (AFCS) were used to develop comparison data in the areas of complex gain scheduling and automated preflight system test.

The general background and scope of the FCD task are presented in FAA-SS-73-2-1 (ref. 1), along with detailed descriptions of the one analog and two digital systems studied. The two digital systems evaluated were a variable incremental control processor subsystem (ICPS) (developed during the early stages of the SST program) and a whole-word computer subsystem (WWCS), a general-purpose processor design similar to those currently under development by many flight control system electronics suppliers. Specific areas identified for examination were:

- General control function operational performance
- Sensor signal selection and failure detection processing

- Redundancy management for fail-operational/fail-passive system response
- System preflight test requirements and methods, supported by failure modes and effects analysis

The primary purpose of this document is to present results of laboratory tests and analytical work covering the first three areas listed above. A secondary purpose is to present comparisons of the two digital system's computing time and memory requirements for implementing the complex control functions found in the SST control system design. The preflight test and failure modes and effects study results are presented in FAA-SS-73-3 (ref. 2).

Section 2.0 of this document describes the SST flight control system application model used for the FCD task. The analog, ICPS, and WWCS mechanizations of the application model are presented in section 3.0. Section 3.0 also provides a comparison of the memory and real-time budgets required by the ICPS and WWCS for implementing the application model functions. Sections 4.0 and 5.0 present laboratory test results and analytical performance data on the three systems and system elements evaluated.

Section 6.0 is a review of the software development and control procedures utilized during the FCD task. This section also presents a summary of projected software management requirements for the development of a digital flight control system.

As an add-on effort within the FCD task, the ICPS was flight test evaluated in conjunction with a flight test program conducted as part of the advanced electronic display system (ADEDS) task, task VI, of the SST Technology Follow-On Program. Section 7.0 deals with the flight test configuration, objectives, and test results.

2.0 FCD TASK APPLICATION MODEL DESCRIPTION

2.1 SST PITCH AXIS CONTROL SYSTEM

The pitch axis control system functions for the SST were designated to be the FCD task application model. The following is a brief background discussion on the SST pitch control system elements.

The basic SST airframe was aerodynamically unstable in the longitudinal axis for the subsonic region of operation. For this reason, airplane stability was artificially produced through the airplane flight control system. Since stability augmentation for the airplane became flight critical, the control system reliability had to approach that of the basic airframe structure. The resultant requirements for safety and high electronic system integrity brought on the development of two basic stability augmentation systems:

- 1) The hardened stability augmentation system (HSAS)
- 2) The electric command stability system (ECSS)

The HSAS design was the simplest possible system that would assure minimum-safe control of the SST airplane. The design was based on the premise that a very simple redundant channel system would provide the system reliability needed. No gain scheduling was allowed and physical separation of the channels, both electrically and mechanically, was required. Further, failure of any or all of the nonessential systems was not to have an effect on the HSAS. A complete description of the HSAS and other SST pitch axis control system elements can be found in FAA-SS-73-1 (ref. 3).

Normal SST handling quality control was to be provided by the ECSS. Gain scheduling and more sophisticated (complicated) control law functions were to be used in the ECSS. Electronic cross-channel voting/monitoring was allowed to permit isolating ECSS and sensor failures from the HSAS. The ECSS was the only nonessential system to have an interface with the HSAS. Both the HSAS and ECSS were to be four-channel fail-operational-squared (operational after two like failures) analog hardware system designs. The four channels were required to achieve the system functional reliability specified for the SST 2.7-hr nominal mission.

The automatic flight control system (AFCS) for the SST was to be a redundant triple-channel (fail-operational) digital system. This system coupled to the primary control system through the ECSS was to provide a refinement of handling qualities through a control wheel steering (CWS) mode that used not only gain scheduling but also filter time constant scheduling with flight condition. The AFCS was to administrate an automated preflight integrity check of the HSAS and ECSS elements to provide assurance that no latent failures existed in either the HSAS or ECSS just prior to takeoff.

These three system functions (HSAS, ECSS, and AFCS/CWS) were selected as the application flight control model for the DOT/SST FCD task. They contain a variety of typical control system functions such as filters, limiters, nonlinear gain scheduling, and basic

logic conditions. However, the original SST system configurations were somewhat altered for the FCD task to reduce the cost of the FCD study. The studies conducted during the FCD task dealt with a triplex fail-operational system configuration rather than a four-channel fail-operational-squared system.

The following section presents a detailed description of the control system elements and configuration used for the FCD task.

2.2 FCD TASK LABORATORY TEST CONFIGURATION

The basic FCD task laboratory flight control system test configuration consisted of the following elements:

- An analog computer simulation of the SST airframe aerodynamics and various components of the pitch axis control system—simplex
- A hydromechanical test stand (minirig Iron Bird) that contained representative electrical, hydraulic, and mechanical elements of the SST pitch axis electric command servo (ECS) system—triplex
- An electronic flight control processing subsystem—triplex

A block diagram of these elements, as they were interconnected for the FCD studies, is shown in figure 2-1. The block noted as "triplex flight control electronics" was the only configuration variable within the laboratory. It represents the hardware boundary for analog, ICPS, and WWCS electronics substitution into the laboratory baseline configuration.

2.2.1 Analog Computer Simulation

A Systron-Donner (SD-80) analog computer was used to simulate the SST airframe, power control units, pitch trim system, and pilot input command control. The basic airframe was simulated for a single-point flight condition of 610,000 lb aft cg, at an altitude of 26,600 ft and a speed of Mach 0.9. This was a critical flight condition case with regard to basic airframe instability. A low gain (weak) attitude hold closed-loop function was used to stabilize the airplane since this condition remained speed unstable even with the HSAS control applied.

The power control units were simulated as a single-order system with a break frequency of 10 rad/second having an equivalent horizontal control surface authority of $\pm 15^\circ$. The surface position was commanded as a function of the bused output shaft of the minirig redundant servos. A hysteresis of $\pm 0.025^\circ$ was placed in the simulation to reflect normal power control unit hysteresis characteristics.

Three trim control motors were simulated, each responding to its respective electric command servo output with appropriate trim command thresholds and time delays. The trim system produced an equivalent surface trim rate of $1^\circ/\text{sec}$.

Pilot input to the SST control system was via both mechanical and electrical feed-forward paths. The mechanical path went directly to the control surface power control unit, and the electrical command went to the HSAS control function. The mechanical input, however, was fed to each ECS to negate the mechanical action. The mechanical input command path and negation action were included in the simulation.

2.2.2 Hydromechanical Test Stand

The hydromechanical minirig used for the FCD task was a modified test stand originally built to evaluate the redundant servo system designed for the SST rudder control system. The rudder system requirements were somewhat different from the longitudinal flight control requirements; therefore, the minirig did not represent precisely the SST pitch control electric command servo configuration. The following describes the minor differences.

The electric command servo piston stroke on the minirig has an authority of ± 1.5 in. while the servo authority on the actual pitch axis system was ± 1.0 in. Since the power control unit (PCU) was simulated, this difference in actuator stroke was accounted for by the scale factor on the analog computer.

The force limit of the minirig actuator is 640 lb as compared to the force limit of 150 lb on the actual pitch axis actuator. However, since the ECS synchronization shaft stiffness on the minirig is approximately three times higher than the corresponding pitch synch shaft, the end effects of actuator to control surface compliance are similar.

The minirig as used consisted of three channels of servo valves, actuators, and control electronics. Each servo channel is referred to as an electric command servo (ECS). Figure 2-2 is a block diagram of a single-thread ECS channel. The ECS control electronics included servo valve drive amplifiers, feedback sensors, failure monitors, and loop closing circuitry.

2.2.3 Flight Control Subsystem Functions

The following is a summary of the control functions mechanized using the electronic flight control processing subsystem and elements described above.

- 1) *Hardened stability augmentation system (HSAS)*—Figure 2-3 is a simplified block diagram of the HSAS functions used in the FCD task. The primary inputs to the system are the pilot's commands and a pitch rate sensor. The control law consists of a series of structural mode and performance compensation filters to provide acceptable airplane stability and handling qualities. The processed (filtered) inputs are combined with the trim command to formulate the input to the electric command servo. For specific cg cases, the SST prototype was designed to be flyable via a mechanical path without HSAS. In order to reduce transient upsets when reverting to the mechanical control configuration from HSAS (no stability augmentation), the electric servo position was off-loaded into the mechanical system through the trim inputs. This process was labeled LINK-SYNC, and its relationship to the overall HSAS functions is illustrated in figure 2-4. Anytime the ECS position exceeds a value of $\pm 0.25^\circ$ for an interval longer than 2 sec, the trim

motors are commanded to run in the direction the servos are displaced. As the trim motors run, the mechanical path negative feedback causes the servo to retract toward center until the servo displacement is less than $\pm 0.2^\circ$.

- 2) *Electric command stability system (ECSS)*—A block diagram of the ECSS functions used in the FCD task is shown in figure 2-5. This system was to provide desired handling qualities augmentation and, therefore, required airplane parameter inputs of dynamic pressure, Mach number, and pitch attitude in addition to pilot and pitch rate inputs. The ECSS filters were again designed for structural mode and performance compensation. Gain values for this system were higher than those achievable for the HSAS since gains were scheduled as a function of flight condition. The gain schedule relationships are shown in figure 2-6.

The ECSS provided airplane speed stability through a trim control generated from dynamic pressure (q_c) and Mach number. This was designed to work in harmony with normal pilot manual trim inputs. A trim integrator driven from either the manual or automatic trim commands is summed with a rate-limited speed command function to produce a total trim command. A lead term path was taken from the speed trim function and added to the primary ECSS command path to provide desired phugoid damping.

- 3) *Automatic flight control system: control wheel steering*—Figure 2-7 is a block diagram of the SST pitch attitude control wheel steering (CWS) function used in the FCD task. The CWS control law maintained the existing pitch attitude when the control column was less than a preselected displacement threshold from the column force neutral detent. When the column displacement exceeded this threshold, the pitch attitude control loop went into a synchronization mode, and the pilot maneuvered the airplane through the ECSS control laws.

Following the pilot's maneuver command (control column return below the threshold), a pitch rate loop was initially closed to smoothly reduce the pitch rate induced by the column input. When the pitch rate signal became less than a set value (approximately $0.2^\circ/\text{sec}$), or 2 sec after the column returned below the preset threshold, the attitude control loop locked on the existing pitch attitude.

A control column feed-forward path was used to suppress an undesirable pitch attitude overshoot, and a second-order compensation filter was used to add damping to the lightly damped short period response exhibited by the ECSS.

Coefficients of the second-order filter were scheduled as a function of dynamic pressure and Mach number to obtain better tracking of the ECSS dynamic changes throughout the flight regime. A roll angle versine function and a roll rate feedback were introduced to reduce altitude loss and lateral control coupling during turning maneuvers. A continuous automatic trim function was utilized in conjunction with the pitch attitude control loop. The automatic trim function was disabled during flight maneuvers. However, if the pilot was holding the column out of the detent for more than 10 sec, the airplane would be trimmed automatically at a slow rate. Manual trim was operative at any time during the CWS mode engagement.

A speed feedback path was provided with the pitch attitude control loop to reduce the coupling between engine thrust and pitch axis response. Figure 2-8 shows the gain schedules associated with the CWS control. An easy-off function (see fig. 2-7) was used to gradually phase out pitch attitude commands existing at the moment a maneuver was initiated.

3.0 APPLICATION MODEL IMPLEMENTATION—ANALOG AND DIGITAL

Various functions of the three flight control subsystems (HSAS, ECSS, and AFCS/CWS) described above were implemented using three different types of computational electronics; i.e., analog, incremental digital, and whole-word digital. The HSAS control function was used as the baseline for gathering performance comparison data between the three electronic subsystems. It was the only function mechanized with analog circuitry. The more complex ECSS and CWS functions were established to obtain a broad comparison of the two digital subsystems' memory and computing time requirements.

The following sections present the details of how each function was mechanized within the three types of computational electronics. This is followed by a section that compares the software memory and timing budgets for the functions mechanized within the two digital subsystems.

3.1 ANALOG SYSTEM

A triplex channel representation of the HSAS function, described in section 2.2.3, was built up using dedicated analog electronic elements and an analog computer simulation. The HSAS control law filters shown in figure 2-3 were implemented, using breadboard circuits for two channels and an analog computer simulation for the third channel. The breadboard circuits were mechanized as shown by the schematic diagram of figure 3-1. The breadboard circuits were built to represent the piece part and construction standards that were to be used in the actual SST design in order to have a proper model for laboratory evaluation of failure modes and effects of single component failures. Since the analog system was to provide the baseline performance data, care was taken in the development of the breadboard circuits to ensure that the static gain and frequency response characteristics were within 2% of the desired theoretical values.

The triplex analog HSAS did not have sensor or command signal crossties between channels (i.e., a brick-wall configuration). The required channel tolerance equalization and failure monitoring functions were mechanized as part of the electric command servo electronics, downstream of the control law filter circuits. Figure 3-2 is a block diagram of the analog HSAS implemented in the laboratory.

Inputs to the equalization signal path originated from an actuator bypass valve, which was activated when the force across the actuator reached a preset force level (detent). The bypass valve hydraulically reduced the pressure across the actuator and, through an electrical transducer path, acted to equalize the servo output by reducing the servo command signal. Both proportional and lag-hold (pseudointegral) equalization paths were provided. The proportional path reduced dynamic differences within an authority of $\pm 1^\circ$ of equivalent surface command, and the lag-hold path reduced static differences within an authority of $\pm 4^\circ$ of equivalent surface command. The bypass valve also acted to effectively remove a channel off-line should the servo command continue to drive the servo beyond the equalization limits. Since the outputs of the redundant servo actuators were force summed

an actuator remaining in a fully bypassed state was overpowered by the remaining good channels. This arrangement is an implementation of a two-out-of-three majority logic hydromechanical voter.

In-line failure monitoring was used in the analog system; that is, the channel failure detection logic indicated a failure when an error signal exceeded a predetermined trip level within that channel. The original SST pitch servo failure monitoring utilized cross-channel failure detection and indicated a failure when the difference between channels exceeded a predetermined trip level. Since the threshold, time constant, and trip level of the in-line failure monitors were set equal to the cross-channel failure monitor of the original system, no significant difference existed in the monitor performance.

Three forms of failure monitors were used to detect failures classified as either dynamic, static, or oscillatory (refer to fig. 3-2). The dynamic failure detector (DFD) registered a failure when a 1° surface command error remained at the input to the ECS drive amplifier for 1 sec. The static failure detector (SFD) registered a failure when the output of the lag-hold circuit exceeded 4° of equivalent surface command for 1 sec. The oscillatory failure detector (OFD) was designed to register failure states that caused high-frequency servo system oscillations not detectable by either the DFD or SFD. A block diagram of the OFD is presented in figure 3-3. If an oscillation existed within one channel of the control system, independent of the other two channels, it was reflected by the equalization detent. The OFD converted periodic oscillations into a limited square wave, demodulated the square wave into a dc value that drove a pseudointegrator, and registered a failure when the integrator value reached a preselected level. A washout filter was used to remove static offsets to the OFD and the pseudointegrator slowly decayed following dynamic transients, so that a detectable failure had to be periodic and above approximately 0.22 Hz.

3.2 INCREMENTAL DIGITAL SYSTEM

All control law functions of the flight control subsystems described in section 2.2.3 (HSAS, ECSS, and AFCS/CWS) were implemented with the incremental control processor subsystem (ICPS). Although all functions were programmed, only the HSAS function was tested as a closed-loop system in the laboratory. The ECSS and CWS functions were programmed only to gather software utilization data on the ICPS relative to a large and complex flight control system problem. The following sections present details related to implementing the HSAS function and ECSS gain schedule function using the ICPS.

3.2.1 HSAS Implementation Using ICPS

The HSAS control law functions shown in figure 2-3 were programmed into the ICPS. The following programming steps were used to develop an operational program:

- 1) Draw a scaling map showing required scaling for all input and output variables.
- 2) Draw an algorithm map, following the system defining block diagram, and derive algorithm coefficients consistent with the scaling map requirements.

- 3) Translate the algorithm map representation into source deck control statements and process the source deck through the assembler program to produce an object program tape.

These three steps are discussed below in detail for the HSAS function.

3.2.1.1 ICPS/HSAS Scaling Map

The main consideration in developing scaling for the variables of a flight-critical digital system is to achieve the desired resolution without risking the chance of incurring a computational overflow (a somewhat unique characteristic of fixed-point digital processors). Computational overflow occurs whenever internal arithmetic or logic processing of variables causes the variable value to exceed the full range of the computer word size. Overflow protection can be approached through careful scaling of worst case variable states (i.e., scale to permit all variables to be maximum and additive at each summing point without exceeding the maximum machine unit capability) or through hardware or software algorithms that will not permit a machine to process beyond a maximum value state. The first approach considerably reduces the resolution scaling flexibility, especially in a situation where a number of variables and high gain are required at a summing node. Ideally, the second approach removes the concern of computational overflow and permits selecting variable scaling to meet normal and not worst case performance levels. A detailed discussion of overflow as it pertains to digital flight control systems is presented in section 5.3.

The ICPS did not contain hardware or software protection against a computational overflow; therefore, careful variable scaling was used, taking into account the following:

- Input range
- Input resolution
- Slew rate
- Internal computer range
- Output range
- Output resolution

Input resolution and range for the ICPS are fixed by the 10-Vdc/12-bit analog-to-digital (A/D) converter, which gives a resolution of 0.00488 V and range of ± 2048 machine units (MU). Thus, once range or resolution is selected for the input, the other is given. If an acceptable range/resolution relationship cannot be realized, system gain levels must be redistributed or the A/D word size must be changed.

Maximum signal slew rate for the ICPS processor is 64 MU per iteration. Since the ICPS iteration rate is 162.76 times per second, the processor slew rate limit is 10,417 MU/sec. Once the resolution scaling has been selected, slew rate requirements can be

determined by multiplying the resolution by the maximum rate of the input signal. Exceeding the processor slew rate capability will cause gain and phase loss in the control law processing.

The internal range of the ICPS computer is $\pm 2^{15}$ or $\pm 32,768$ MU. Since overflow would result in a value change of maximum positive to maximum negative, or vice versa, extreme care must be taken when scaling internal points in the program.

Output range and resolution, like the input, are established by the 12-bit/10-Vdc digital-to-analog D/A converter. Unlike the input resolution considerations, output resolution can be a function of the control function loop or forward path gains.

An ICPS scaling map for the HSAS is shown in figure 3-4. The input/output signal range for the SST HSAS control system is enclosed in square brackets. These ranges were determined from SST control system performance requirements. The input, output, and internal scaling, as realized in the incremental computer, is enclosed in parentheses in the scaling diagram. The input scaling was accomplished by selecting the maximum range of the 12-bit A/D (2048 MU) to be equal to the maximum required system range. Then the resulting resolution was computed and compared to the system resolution requirement. For instance, the maximum pitch rate requirement of $20^\circ/\text{sec}$ was set equal to the A/D range (2048 MU). The resolution then became $20/2048$ or $0.00976^\circ/\text{sec}/\text{MU}$, giving 102.4 MU/deg/sec. The system resolution requirement was $0.02^\circ/\text{sec}$, which is greater than one machine unit; therefore, the range/resolution requirements for the pitch rate input were met.

The maximum rate of change (slew rate) of the pitch rate signal will be $20^\circ/\text{sec}^2$. This times 102.4 requires a processor slew rate of 2048 MU/sec, well under the slew rate limit of 10,417 MU/sec.

For internal scaling, each path gain must be considered. For example, prior to the first summing junction in the pitch rate signal path, the internal scaling reflects a forward path gain of 3.1, or a potential maximum value of 3.1 times 2048 (A/D range), a total of 6349 MU. This path summed with the control column input path (with a gain of 0.667) is $0.667(2048) + 6349$ for a total value of 7720 MU. Since the maximum internal machine level is 32,768 MU, this is well within the acceptable range.

In general, the pitch rate and column signal paths sum with opposite signs, since the pilot action is to maneuver the airplane, and the pitch rate feedback is to provide short period damping. However, to establish that no potential overflow situation exists, both signal paths are assumed to be additive with the same sign at their maximum values. This is the worst case (pessimistic) situation for that summing node. Fortunately, the HSAS control laws are not complex (very few input sensor and/or command paths), and the worst case analysis to determine overflow margins does not result in any problems. If a control system utilizes many sensors and has high gain relationships between the variables, resolving potential overflow summing nodes could result in serious range, rate, and resolution scaling problems.

Output scaling was accomplished by setting the maximum D/A output equal to the maximum required surface command value (2048 MU equal to 15°). This gave an output resolution of approximately 0.007°. The SST HSAS output resolution requirement was 0.005°, an unusually tight requirement with respect to contemporary commercial transports. Because of this, no attempt was made to resolve this level of discrepancy for the FCD task.

Output rate saturation was determined by adding all summing node variable rate requirements in a worst case fashion. This produced an output rate processing level of 13,303 MU/sec, greater than the computational slew rate value of 10,417 MU/sec. However, it was found that the saturation rate of the electric command servo was about half of the ICPS slew rate; therefore, this observed slew rate deficiency is inconsequential.

3.2.1.2 ICPS/HSAS Algorithm Map

Programming of the HSAS function was accomplished with the use of the *ICP-723 Incremental Computer User's Manual* catalog of algorithms (ref. 4). This manual was produced by the General Electric Company and includes incremental equation derivations for most common flight control functions. Using the catalog, the HSAS control law block diagram was converted into a representative algorithm map. Proper algorithm coefficients were then calculated using information from both the system block diagram and scaling map. An example follows.

Figure 3-5 shows an algorithm map of a second-order filter, illustrating the relationship between the S domain coefficients and the algorithm coefficients. (A detailed derivation of the filter difference equation is presented in section 5.2.) For the HSAS B filter of figure 2-3,

$$\frac{Z}{X} = \frac{S^2 + 2.31S + 2.72}{S^2 + 5.62S + 3.1}$$

The appropriate algorithm coefficients were calculated using the equations of figure 3-5. The results are presented in figure 3-6.

Following the above procedure, the coefficients for all HSAS algorithms were calculated and the appropriate algorithm control bits were set. The programmer then assigned each algorithm a program number and time (location of program in memory and execution time within the iteration frame). Such an assignment is based on:

- 1) Size of total system program relative to total computer processing capability
- 2) Interdependence of one control function to another
- 3) Input and output timing of variable and discrete data within the iteration frame

The ICPS computer has a memory capacity for 256 algorithm building blocks and a processing time capacity for 128 algorithms per iteration. The basic computer program

execution structure is divided so that algorithms will be processed in odd and even time slot sequential strings. If no branching is utilized, the computer will sequence through algorithms 1 through 64 during even algorithm processing time slots and 65 through 128 during odd processing times. Branching can be used to jump to the higher numbered algorithms in memory (129 through 256), with execution continued sequentially from that algorithm number during whatever processing time slot (odd or even) the branch was made. Generally, any algorithm can be processed during any algorithm processing time slot using branching. However, only 128 algorithms can be processed each iteration, with the program always beginning at the start of an iteration with algorithms 1 and 65.

For small programs that require less than one-half the available processing time slots, very little attention needs to be given to where particular algorithms are located in memory. However, for programs requiring greater than 50% of the processing time slots, care must be taken to assign algorithm times and numbers so that interdependent functions execute in the desired sequence.

Input and output operations for variable data occur during even algorithm time slots. Registers used during this I/O process are also used in making the branch test. Therefore, to effect maximum processing efficiency, programs requiring branching should be located in the odd processing time slots.

Because of the size of the HSAS control law program (29 algorithms), algorithm times and assignments were made considering only the program flow. A complete algorithm map of the HSAS program is shown in figure 3-7. The map is presented to illustrate algorithm usage relative to the types of control law functions found within the HSAS control law. Detailed information on developing such a map and designating the appropriate control bits (dark circles) is given in reference 4.

All the functional elements within the HSAS control law, except for the automatic trim logic, were directly realized using standard algorithms. The trim logic timer circuit was implemented using branching to reset the trim delay function. Twenty-seven algorithm time slots were required for 28 algorithms programmed in memory. This utilized approximately 21% of the processing time and 11% of the available memory.

3.2.1.3 ICPS/HSAS Program Load and Verification

After completion of the algorithm map, a source deck is compiled by translating each algorithm structure onto control cards in the form of assembler language statements. The source deck is then processed through a support software assembler on a host general-purpose computer system (a PDP-8 was used at Boeing during the FCD ICPS studies). An object tape is produced and used to enter (load) the program into the ICPS program memory units.

Verification of the program is achieved through the process of (1) desk checking the algorithm map against the original flight control system block diagram and (2) conducting laboratory performance and resolution tests and comparing results to predicted or known

values. The latter check extensively utilizes the ICPS program monitor unit, which permits selectively observing every input and output, both variable and discrete, while the system is operating.

3.2.2 Gain Schedule Function Implementation Using ICPS

Gain schedule functions are utilized extensively in the makeup of advanced control laws. Since the HSAS control law by a design ground rule could not employ gain scheduling, gain schedule functions of the ECSS were programmed to evaluate the effectiveness of the ICPS in implementing such functions.

The gain schedule function $K_{V\phi}$ shown in figure 2-8 was selected for laboratory evaluation using the ICPS. All gain schedule functions of the ECSS were programmed to establish memory and timing requirements, but not all were tested in the laboratory. The non-real-time integration capability of the ICPS was utilized in implementing gain schedule functions such as that represented by the $K_{V\phi}$ function—a function made up of linear gain relationship segments, having a finite number of discontinuities (breakpoints). Thus, the gain schedule $K_{V\phi}$ was realized by integrating the segment slope values with respect to a variable (in this case, Mach number). The formula of this integration for the ICPS is:

$$K_{V\phi} = \int_0^N f(m) dM + K_{V\phi}(M_0)$$

$$\approx \sum_{i=0}^N f(m_i) \Delta M_i + K_{V\phi}(M_0)$$

where

$f(m)$ = slope of $K_{V\phi}$ linear segments

M_i = an i increment of Mach number

$K_{V\phi}(M_0)$ = initial gain value for a particular segment of the $K_{V\phi}$ function

Figure 3-8 is a functional block diagram of the ICPS process for generating the $K_{V\phi}$ function. The process has three major subfunctions: breakpoint detector, slope generator, and initial value/integrator summer. The breakpoint detector establishes the discontinuity points and sets the slope value for a particular line segment. One algorithm is required for each breakpoint (function discontinuity). The branching feature of the ICPS computer unit is used to efficiently implement the breakpoint function. Therefore, nine algorithm memory locations and five algorithm time slots were used to implement the seven $K_{V\phi}$ breakpoints.

The slope generator provides the multiplying constant that corresponds to a particular line segment slope value. Three algorithm memory locations and one algorithm time slot were used to implement the seven $K_{V\phi}$ line segment slopes.

The initial value and integrator summation functions are generated in a single algorithm. This algorithm performs rectangular integration as well as providing the discontinuities' initial values. The algorithm map for the gain schedule function described above is shown in figure 3-9.

Two types of laboratory tests were conducted on the programmed $K_V\phi$ function. One test dealt with accuracy and the second test checked the generated function stability and drift characteristics. The accuracy test consisted of overplotting the desired function, with the result showing the error remaining with 1% of full scale (fig. 3-10). The stability test consisted of driving the gain schedule function with different ramp rates that ultimately exceeded the maximum predicted input rate and the tracking capability of the programmed function. Tracking up to the computer solution rate gave very accurate results and, after the solution rate was exceeded, the output would always recover to the correct solution. The drift test consisted of driving the gain function through its full range with a 0.1-Hz sawtooth drive signal for 1 hr, then stopping the drive signal and observing the function output for 3 hr. This was repeated several times. These test results established that the programmed function did not exhibit drift.

3.3 WHOLE-WORD DIGITAL SYSTEM

All three flight control subsystems described in section 2.2.3 (HSAS, ECSS, and AFCS/CWS) were programmed for the whole-word computer subsystem (WWCS). Only the HSAS control law and ECSS gain schedule function were evaluated in the laboratory using the WWCS equipment. The additional programming provided information relative to WWCS software timing and memory utilization for a large and complex flight control system problem. In addition to the software required to realize the flight control subsystem functions, a general executive, redundancy management, and peripheral equipment software routines had to be developed.

The following discussion describes the control law software buildup for the WWCS. The general organization of the WWCS executive is presented in the systems descriptions document (ref. 1).

3.3.1 WWCS Control Law Software Structure

A top-down software structure was used to implement the control laws associated with the HSAS, ECSS, and AFCS/CWS subsystems. The combined program was given the name HEC. The first step in the development of HEC was to functionally define the interface between HEC and the WWCS executive (EXEC) program. Next, the block diagrams of the control law functions were used to specify:

- 1) HSAS subsystem supervisory routine (HSAS)
- 2) ECSS subsystem supervisory routine (ECSS)
- 3) CWS subsystem supervisory routine (ECS)

- 4) Interfaces between the supervisory routines and the following:
 - a) Sensor signal selection/failure detection routine (SSFD)
 - b) Discrete selection/failure detection routine (DISCR)
 - c) Redundancy management routine (REDMAN)
 - d) The various modules of the FCD library (FCDLIB)

Once the overall functional specifications and interfaces between all major components of HEC were defined, software engineering of the various components consisted mainly of iterating and nesting a small number of basic or standard control logic structures (macros, functions, subroutines, etc.).

The WWCS in-flight operational programs were subdivided into three levels (groups). The WWCS executive became the primary level (level I) and essentially controls the timely execution of the other program levels. Level II programs are those associated with control law functions (HEC) and the sensor signal selection/redundancy management programs.

Programs used to drive system status and failure warning displays and programs that interact with WWCS control functions, such as ERROR RESET, were classified as background programs and are grouped as level III. Each of these program levels, or portions thereof, are executed each input/output iteration frame (the I/O iteration frame refers to a real-time interrupt that occurs every 6.144 msec).

Figure 3-11 depicts the sequential processing of the program levels with respect to the input/output frame timer interrupt. When an interrupt occurs, the processor is forced into the EXEC routine (level I). The EXEC stores the conditions of the interrupted program and passes control to the level II (HEC) program, updating a frame count reference number (good for 2400 hr of continuous operation). The HEC program is entered as a subroutine, returning control to the EXEC before the next timer interrupt occurs. Since the functions processed in HEC are real-time computations, it is essential to preserving performance fidelity that such functions not incur uncontrolled interrupts. In response to an exit from HEC, the EXEC passes control to the background, level III, programs. The level III program in progress at the time of the last timer interrupt is reentered. Level III programs continue to be processed until the next I/O timer interrupt occurs.


HEC programs that require a processing time greater than the I/O iteration frame time can be subdivided and processed in adjacent I/O iteration frames. Figure 3-11 illustrates this by showing the different level II processing times. Such subdivisions are established by the programming engineer after initial worst case timing analyses have been completed.

3.3.2 HSAS/ECSS/CWS Implementation Using WWCS

A five-step procedure was used in programming the HSAS/ECSS/CWS functions into the HEC program.

- 1) The original control system block diagram was transformed into a digital system diagram showing variable names, observation points, binary scale factors, signal resolution, and maximum signal values.
- 2) The digital system diagram was transformed into a macro flow chart showing significant details and routines used to implement control law functions.
- 3) The macro flow chart was used to develop source deck assembly code for the WWCS computer and processed by the support software assembly program to effect memory paging/assembly validation.
- 4) All HEC programs were combined using the support software linkage editor and processed through the WWCS computer simulator program to effect limited static and dynamic checkout.
- 5) The programs were loaded into the WWCS computer to complete full-scale static/dynamic performance verification.

3.3.2.1 Digital System Diagram

Figure 3-12 presents the digital system diagram for the HSAS functions as transformed from the original system block diagram shown in figure 2-3. It is very important to identify on this diagram all observation points that may be required during laboratory investigations. Once a program has been developed (coded), it is not easy to change or add observation points. Figure 3-12 indicates such points prefaced with a 'P' enclosed in a  shape.

Mnemonics for the variable names (refer to fig. 3-12) were selected for convenience of programming and ease in identification. Since there can be a large number of variables associated with flight control systems, some basic rules were established to guide mnemonic assignments. These rules were:

- 1) Maximum number of characters = 6
Minimum number of characters = 4
- 2) Use of characters:
 - a) First two letters identify the system.
 - b) Following one or two characters identify the function.
 - c) Following one or two characters identify function number.

- 3) If the variables themselves indicate the particular axis or system, the prefixing is not required. For example, the airplane's variables do not require prefixing, i.e., $\theta = \text{THETA}$.

A mnemonic example from the HSAS digital system diagram of figure 3-12 is given below.

$$\text{PS PL O2} = \underbrace{\text{PS}}_{\substack{\text{Pitch} \\ \text{SAS}}} \underbrace{\text{PL}}_{\substack{\text{Position} \\ \text{limit}}} \underbrace{\text{O2}}_{\substack{\text{Second} \\ \text{position} \\ \text{limit}}}$$

Software scaling values are also shown in the diagram of figure 3-12. Scaling was developed to maximize the dynamic signal resolution for each function processed while preventing overflow situations. Overflow occurs when a signal exceeds the maximum range of the computer. Since overflow would result in a value changing from maximum positive to maximum negative, or vice versa, extreme care must be taken to prevent overflow in the processing of flight control functions. Therefore, maximum scaling levels (MSLs) were established for each processing node within the system. The MSL was determined by summing all signals at their maximum value with the same sign. Scale factors are expressed as powers of 2 and denoted in figure 3-12 as (BX), where $\text{BX} = 2^{-X}$. The step-by-step scaling procedure used by the engineer/programmer of the HEC functions was as follows:

- 1) Determine tentative minimum scale factors for all variables and temporary variables along the signal path.
- 2) Determine tentative minimum scale factors for constants, thresholds, limits, etc., compatible with the scaling of the signal leg determined in step (1).
- 3) For each of the summing junctions in the digital system diagram, use arithmetic shifts or adjust scale factors set in steps (1) and (2) to realize a common scale factor for all summing legs.
- 4) Compute the absolute value of a summing junction output using the MSL values of the summing junction inputs.
- 5) If the answer to step (4) is less than 1, proceed to the next summing junction, or to step (7) if all summing junctions have been processed.
- 6) If the answer to step (4) is greater than or equal to 1, reduce the scale factor in one of the summing junction inputs and go to step (3).
- 7) Review the internal resolution of all signals. If the resolution of all signals is \gt the resolution at the I/O interface, stop. If the resolution is \lt the I/O interface resolution, rescale and insert limiters to provide internal resolution \gt the I/O interface resolution.

The WWCS met all input/output resolution scaling requirements of the HSAS function, except for the surface position command and feedback. Like the ICPS, this could only be resolved by extending the A/D and D/A word lengths, a cost penalty unjustifiable for the FCD task.

3.3.2.2 Macro Flow Chart

An important step in programming flight control functions for a whole-word digital system is to prepare a flow chart that organizes the computational sequence of each function in the control law. The following describes the flow charts for the HEC program developed for the WWCS.

The WWCS executive (EXEC) transfers control to the HEC program via an interface routine referred to as LINKS. The LINKS routine administrates the level II (HEC) program, consisting of three subroutines—HSAS, ECSS, and REDMAN. The REDMAN subroutine is the redundancy management program for handling the multichannel output data. The AFCS/CWS function is part of the ECSS subroutine. The relationship of these subroutines to the executive (level I) program is illustrated in figure 3-13.

After receiving control from the EXEC program, LINKS directs the execution of the HSAS subroutine. Following HSAS execution, process control returns to LINKS, which directs the execution of the ECSS subroutine. The control wheel steering (CWS) mode, which uses the ECSS for inner loop damping, is called from the ECSS program if the mode is engaged. Following ECSS execution, control returns to LINKS. The final step for LINKS is to call the REDMAN subroutine to update the control law output command memory buffers. Following this, process control is returned to the level I EXEC.

Figure 3-14 shows the HSAS supervisory macro flow chart. The flow chart defines the initial condition structure for initializing integrators and filters when the computational reset (RESET) function has been activated. When the system is operational following startup initialization, the HSAS supervisory program checks for ECSS engagement. If the ECSS is engaged, the HSAS program returns to LINKS and the ECSS subroutine is called. If the ECSS is not engaged, the HSAS program conducts the housekeeping tasks related to multiple frame time control. As presented in the system description document (ref. 1), the basic input/output iteration (frame) period fixed by hardware in the WWCS is 6.144 msec. Some flight control laws require more than a single 6.144-msec frame to complete all computations. Therefore, the HSAS supervisory program was set up to allow multiple I/O frame passes to complete processing a particular control law (i.e., permit frame time periods of any multiple of the basic 6.144-msec I/O iteration). Thus, the computational frame time could be extended to investigate performance variation as a function of control law iteration rate (sampling period).

Following the HSAS program initial housekeeping, the control law filters of the HSAS begin to be processed (refer to fig. 3-14). The HSAS program calls the sensor signal selection/failure detection (SSFD) subroutine to acquire the pitch rate sensor information. (Details of this process are covered in sec. 5.1.) The bilinear transformation technique (Tustin's substitution) was used for implementing the HSAS filters and integrators,

subroutines respectively referred to as BZTAR and TZTIN. Detailed discussions on these programs are presented in section 5.2. Upon completion of the filter processing, other signal paths of the HSAS are processed and combined with the appropriate filter outputs. Finally, followup housekeeping (i.e., extended frame, resetting SSFD IC flags, etc.) is processed, and the HSAS program returns control to the LINKS program.

Figure 3-15 is the supervisory macro flow chart for the ECSS and CWS control laws. The general flow of this program is very similar to the HSAS program where housekeeping tasks are processed followed by the execution of the control law functions. However, the ECSS routine (including CWS) required a minimum of three I/O frame times for processing, while the HSAS required only one. Thus, the ECSS function was subdivided into three parts, each to be executed during one I/O iteration frame (refer to fig. 3-15). The initial computational pass processed the main ECSS functions, the second pass processed the CWS functions, and the third pass processed the ECSS output functions and returned control to LINKS. If the CWS mode is not engaged, the program bypasses the main CWS control functions and exits the CWS routine. However, even though the CWS mode is not engaged, some functions such as the pitch attitude washout filter must be processed during the second pass. A flow chart of the CWS mode is presented in figure 3-16.

3.3.2.3 Program Code, Assembly, and Validation

Detail programming (coding) of the HEC program followed the operational sequence presented by the macro flow charts. A subroutine modularized (catalog) approach was used in the software buildup. Assembler language coded programs were then prepared and inspected to verify the absence of errors in:

- 1) Control logic (IC, frame, mode, etc.)
- 2) Interface between the various software modules, SSFD, and discrete routines
- 3) Scaling (overflow protection)
- 4) Basic computational procedure of all HEC components required for the HSAS

These programs were then segmented into linked blocks of code (256 words/page) to fit the WWCS memory paging structure and reviewed for errors in the newly created interfaces (external/entry tables).

After the program had been coded, segmented, assembled, linked, and extensively desk checked, the WWCS processor simulator was used to dynamically validate most elements of the HSAS program. At this point, all external routines (i.e., SSFD, discrete I/O logic subroutine—DISCR, etc.) required for the HSAS had been validated and integrated into HEC. Because of extensive laboratory tests to be conducted on the HSAS software resident in the WWCS hardware, together with the high cost of running the simulator program on the IBM 360/370 system, the scope of the simulator validation activity was limited to:

- Verifying that all elemental functions (limiters, threshold detectors, etc.) were operating correctly

- Verifying that all logic modules (frame, mode, IC, trim) were operating correctly
- Verifying the scaling and the nonexistence of overflows under worst case conditions
- Verifying that the filter and integrator routines were operating correctly (fidelity/deadband)
- Verifying that the gain and time response of the HSAS filters were correct (step input)

Because of the voluminous nature of the validation results, which consist of tabulated and printer-plotter data together with simulated MCP-701 memory dumps, only a descriptive summary of the results is included here.

Elemental Functions—Only the symmetrical limit (LIMITS) and threshold (THRES) subroutines are used in the HSAS. LIMITS is called five times and THRES is called four times. Both elemental functions, together with all limit/threshold values, were observed to be present and operating correctly.

Logic Modules—Three control logic modules (frame, IC, and mode) and one system logic module (trim-threshold-hysteresis) are used in the HSAS routine. All modules are executed during each HSAS pass. These modules were checked through the simulator and observed to be present and operating correctly.

Scaling and Overflow—Two conditions were run through the simulator, representing worst case input conditions for the summing junctions of figure 3-12.

<u>Input</u>	<u>Case 1</u>	<u>Case 2</u>
DCOL (t)	+10 V	-10 V
THTD (t)	-10 V	+10 V
TMDN (t)	-10 V	+10 V
TMUP (t)	-10 V	+10 V
MFBK (t)	+10 V	-10 V
THLD (t)	-10 V	+10 V
SFBK (t)	-10 V	+10 V

In case 1, the negative values of the maximum scaling level (MSL) were obtained. For case 2, the positive MSLs were obtained. This check verified the scaling and the nonexistence of overflow for each summing junction shown in figure 3-12.

The final validation of the HEC program was conducted in the laboratory using the WWCS hardware. This validation consisted of obtaining step and frequency responses of each filter and of the overall control law signal path. Single-channel (one of three) system closed-loop responses were compared to previously obtained analog system responses. This

included frequency and time history response checks for both the pitch rate and control column inputs. Full redundant (triplex) performance response checks were then made. The single and redundant configuration tests are discussed in detail in section 4.0.

As mentioned before, no attempt was made to completely check out the ECSS and CWS programs. However, gain schedule functions were fully implemented and evaluated to gather data for comparison with implementation of such functions using the ICPS. The following section discusses the WWCS gain schedule implementation and section 3.4 deals with a comparison of the WWCS and ICPS memory/timing requirements for the HEC functions.

3.3.3 Gain Schedule Function Implemented Using WWCS

The ECSS and CWSS gain schedule functions are shown in figures 2-6 and 2-8, respectively. Two types of schedule functions are presented: polynomial and straight line segment. These two forms of gain schedules were implemented using subroutines referred to as PGSF, a vector dot product function, and LOOK, a table lookup function (refer to app. A).

The CWS polynomial function was generated using PGSF since each polynomial schedule was defined by vector equations of the form $K_c^T M_q$, where

$$K_c = \begin{pmatrix} K_0 \\ K_1 \\ \cdot \\ \cdot \\ \cdot \\ K_n \end{pmatrix} \quad \text{and} \quad M_q = \begin{pmatrix} 1 \\ m \\ m^2 \\ m^3 \\ q \\ q^2 \\ q^3 \end{pmatrix}$$

For the CWS function, the K_c vector elements are the polynomial constants and the M_q vector elements are the Mach number (M) and impact pressure (q) variables raised to the appropriate power. Once the K_c vectors were stored in core, polynomial gain schedule evaluation consisted of calculating the current M_q vector and then computing the vector dot product via the PGSF function.

Function LOOK was used to implement the straight line segment, piece-wise continuous, gain schedule function via linear interpolation. Line segment breakpoint values were scaled and tabulated into separate data tables. The piece-wise continuous data were organized into a LOOK data array (LDA) as follows.

$$LDA^j = [APLX^j, X_1^j, Y_1^j \dots X_n^j, Y_n^j]$$

where

$$j = 1, 2, 3, \dots$$

$$n = 1, 2, 3, \dots$$

$$X_n = (X_1^j, X_2^j, \dots, X_n^j) \text{ is the } j\text{th independent data vector}$$

$$Y_n = (Y_1^j, Y_2^j, \dots, Y_n^j) \text{ is the } j\text{th independent data vector}$$

$$\text{APLX}^j = \text{address of the previous lowest } X_1^j \text{ data element used by LOOK the last time the lookup utilized vector LDA}^j (2n + 1) - \text{initially } \text{APLX}^j = \text{address of } X_1^j$$

Note: The X vector elements are monotonic increasing, i.e., $X_i^j < X_{i+1}^j$

Whenever LOOK is called, it finds the X_i^j and X_{i+1}^j vector elements that bracket the input variable (XIN) through a comparison process. The address of the starting X_i^j for the comparison process is the address contained in APLX^j . Utilization of this address speeds up the comparison process considerably since statistically the starting X_i^j is most likely to be the X_i^j used on the previous call to LOOK. Once XIN has been bracketed, the address of the new X_i^j is stored in APLX^j for the next pass. The output (XOUT) of LOOK is computed as:

$$\text{XOUT}^j = \frac{Y_i^j * (X_{i+1}^j - \text{XIN}) + Y_{i+1}^j * (\text{XIN} - X_i^j)}{(X_{i+1}^j - X_i^j)}$$

The above gain schedule techniques were implemented in the WWCS and tested in the laboratory for accuracy. Although all gain schedule function computations are essentially independent of sampling rate (frame time), the laboratory tests included evaluating the implemented functions at three different sampling periods—6.144 msec, 18.42 msec, and 49.12 msec.

The CWS gain schedule function K_θ was used to evaluate the polynomial routine (PGSF) and the ECSS gain schedule R_E was used to evaluate the table lookup routine (LOOK). The laboratory tests showed that both methods were successful and that neither produced discernible errors.

3.4 ICPS/WWCS MEMORY AND TIMING COMPARISON

For the analog system described in section 2.0, dedicated hardware is used to implement the flight control system control laws. Generally speaking, the complexity of the analog system hardware bears a direct relationship to the complexity or number of control functions that make up the system. This relationship does not hold for digital systems such as the ICPS or WWCS. For these systems, the hardware complexity or number of functions influence only the size of memory (capacity to store given programs) and processing time (iterative solution rate required to meet system performance objectives). This difference in

hardware architecture makes it difficult to compare analog and digital systems, other than to assess the cost, weight, and volume of both types of equipment performing the same given control functions. However, digital systems can be compared to each other in a piecemeal fashion relative to memory storage and computational time required for each function.

In order to determine memory and timing requirements for implementing a function in a programmable digital system, many human/machine factors must be kept in view. For instance, every engineer/programmer has a unique approach and may employ novel techniques for mechanizing a given function. The programmer's approach will be influenced by the computer instruction repertoire, memory available, time constraints imposed by the hardware design and system requirements, and time period (days, weeks, or months) allotted to the programmer for preparing and checking out the software. For a flight-critical system, these factors must be clearly understood and ground rules established to limit (control) individual programmers' unique software preparation. This is especially true when several programmers are to participate in developing different portions of the flight control system software. A discussion on software development and change control is presented in section 6.0.

At the beginning of any program utilizing a digital system, memory and computational time estimates will be made relative to the projected flight control functions to be implemented. Unfortunately, it is very difficult to provide accurate estimates because of the considerations discussed above. The estimation error, of course, will be the greatest when both the hardware and flight control system are new designs.

Analog hardware in a development program is usually specified to have 20% growth relative to computational electronics space. Experience during the FCD task would place computational growth factors for a digital flight control system to be 70% to 100% both in memory and unused computation time. (Computation time here is the iteration frame time specified to meet system performance requirements.)

The basic software development ground rule adopted for the FCD task was that a modularized approach—standard catalog routines—be used in programming the HSAS, ECSS, and AFCS/CWS functions for both the ICPS and WWCS. The modularized software does not provide the best economy in terms of memory and computing time, but it does provide a long-term advantage for validating software elements. Further, once the catalog has been developed, the modularized software improves the accuracy of estimating memory and timing requirements for new control law implementations.

Before software comparisons can be made between the ICPS and WWCS, some basic hardware/software architectural differences must be understood. The ICPS executive is mechanized in fixed hardware, while the WWCS executive is organized in software. A similar situation exists for the sensor signal selection/failure detection algorithms. The WWCS has an approximate 12-msec transport delay imposed by the input/output double buffer design (a feature added to eliminate timing hazards between bit-for-bit and frame timing data processing), but no similar delay exists in the ICPS. Such functions cannot be directly compared in terms of software parameters. Functions that can be directly compared,

however, are those associated with the control law, such as filters, limiters, and gain schedules.

Tables 3-1 and 3-2 summarize the ICPS and WWCS real-time and memory usage to process the functions associated with the HSAS, ECSS, and CWS redundant systems. One observable comparison is the relative time required to process the combined HSAS filters (an equivalent seventh-order filter); the ICPS required 0.48 msec, while the WWCS required 0.69 msec (both required approximately 100 words of memory). This comparison illustrates the efficiency of the ICPS in computing linear difference equations. An observable comparison that reverses this apparent efficiency advantage for the ICPS is the implementation of nonlinear elements such as limiters and thresholds. The ICPS required 0.61 msec to compute the HSAS limiter/threshold functions, while the WWCS required only 0.16 msec (the memory usages were 164 words and 27 words, respectively). This illustrates the great advantage the WWCS has for efficiently processing logical operations. This advantage of the WWCS is further illustrated when the gain schedule numbers are observed. Some general observations follow about both systems that relate to their individual processing capabilities to implement the subject control systems.

The ICPS evaluated in the laboratory could execute the HSAS and ECSS functions within its iteration frame time of 6.144 msec. However, the more complex CWS function required 7.21 msec, placing it beyond the ICPS existing frame time. The ICPS could be modified, however, to decrease its fixed iteration rate to process the CWS control law. The existing design memory size (3072 usable words) would be sufficient to handle the subject functions.

The WWCS that was evaluated could execute all functions using the extended frame software supervisory control discussed in the previous section. The resulting computational frame time, including the 12 msec of I/O transport delay, would be approximately 30 msec. This time, although adequate to meet system performance requirements, could be reduced by approximately 6 msec using a dedicated hardware SSFD algorithm similar to that used in the ICPS.

TABLE 3-1.-ICPS/WWCS REAL TIME USAGE (MSEC) [HSAS, ECSS, AND CWS FUNCTIONS]

Function		EXEC	Filters	Limiters and thresholds	Gain schedules	Sensor (variables) SSFD	Discrete SSFD	Summers	Other	Total (msec)	Remarks
System											
HSAS	ICPS	1	0.48	0.61	-	0.01	Negligible	0.20	0.01	1.31	1 Mechanized in hardware; does not affect processing time.
	WWCS	0.80 2	0.69	0.16	-	2.92	0.69	Negligible	0.05	5.31 2	2 Plus ≈ 12 msec of I/O transport delay
ECSS	ICPS	1	0.50	1.12	0.79	0.01	Negligible	0.43	0.50	3.35	
	WWCS	1.68 2	1.04	0.12	0.36	4.08	0.64	Negligible	0.08	8.00 2	
CWS	ICPS	1	1.08	1.73	2.66	0.01	Negligible	0.72	1.01	7.21	Numbers show total computing time for CWS mode; includes the ECSS inner loop time
	WWCS	2.07 2	2.90	0.28	1.38	6.21	0.83	Negligible	0.14	13.81 2	

TABLE 3-2.-ICPS/WWCS PROGRAM MEMORY USAGE (WORDS) [HSAS, ECSS, AND CWS FUNCTIONS]

Function		EXEC	Filters	Limiters and thresholds	Gain schedules	Sensor (variables) SSFD	Discrete SSFD	Summers	Other	Total (words)	Remarks
System											
HSAS	ICPS	1	121	164	-	1	1	47	4	336	1 Mechanized in hardware. Does not affect program memory size
	WWCS	252	90	27	-	378	117	Negligible	36	900	
ECSS	ICPS	1	118	346	440	1	1	118	168	1190	Numbers show total memory used for CWS mode; includes the ECSS inner loop memory values
	WWCS	262	105	52	115	357	147	Negligible	12	1050	
CWS	ICPS	1	166	166	820	1	1	118	240	1510	
	WWCS	482	234	56	170	451	151	Negligible	246	1790	

4.0 APPLICATION MODEL ANALOG/HSAS/WWCS LABORATORY TESTS

The prototype SST stability augmentation systems were designed to be implemented using analog piece-part components. In order to assess performance differences that may exist between analog and digital SAS electronics, a series of tests were conducted to evaluate the HSAS function implemented using the ICPS and WWCS and comparing test results to an analog implementation. It was expected that some performance discrepancies could exist because the analog system control laws were developed using root locus synthesis/analysis techniques, and the same control laws were to be implemented in the discrete domain of the digital system. Further, potential discrepancies were anticipated in the digital systems performance resulting from the programming technique used and/or from hardware design aspects such as word length, sample rate, analog input signal prefiltering, etc.

To minimize those discrepancies that may appear from the concerns mentioned above, the digital HSAS implementation was systematically developed beginning with an analysis of the digital system's basic hardware performance characteristics. Next, the HSAS filters were implemented and evaluated from both a dynamic and static performance viewpoint. Finally, the redundant HSAS configuration was fully established and evaluated relative to operational performance and failure mode survivability.

4.1 BASIC ICPS AND WWCS HARDWARE CHARACTERISTICS

The basic hardware characteristics of the ICPS and WWCS were investigated using a single channel of the triple-channel systems. Laboratory tests were conducted to establish the hardware input/output bandpass and linearity characteristics.

4.1.1 ICPS Hardware Characteristics

The basic ICPS hardware can be represented as a prefilter, computational delay, and zero-order hold, as shown in figure 4-1. The prefilter attenuated the analog input signal high-frequency components to suppress "aliasing" from occurring during the analog-to-digital (A/D) conversion. Aliasing is the characteristic of an A/D process where a high-frequency analog signal is sampled (any signal above 1/2 the sampling rate) and a false (folded) low-frequency signal is produced in the discrete time domain. Figure 4-2 illustrates the aliasing phenomenon. The prefilter was selected as a function of the system performance and stability requirements, frame time (sample rate) selection, and knowledge of the input signal frequency content.

Effective suppression of the aliasing error may always be achieved by utilizing a high sampling iteration rate and an analog prefilter combination that has a negligible impact on the system stability. However, this approach may not be practical because of the available hardware and the broad bandpass requirements of some applications. For the systems evaluated in this study, the prefilter was selected to have a double break such that folded frequency components into the bandpass of interest (assumed to be ± 1 Hz) were no greater than 1% of the input signal magnitude. This required a double break filter (40 dB per decade attenuation) to be placed at 105 rad. Two single lag filters were mechanized in the actual hardware, having break frequencies of 100 and 125 rad as shown in figure 4-1.

The computational delay function shown in figure 4-1 is the math model associated with inputting, processing, and outputting a signal through the computer. For the ICPS, this delay time (T_1) is made up of:

- 1) A/D and signal selection processing time (150 μ sec)
- 2) Processor time (48 μ sec times the number of algorithms processed to implement the function)
- 3) Output transfer time (D/A processing time, 48 μ sec)

Thus, the ICPS computational delay T_1 is:

$$T_1 = [150 + 48 (\text{number of algorithms}) + 48] \mu\text{sec}$$

Finally, the zero-order hold function shown in figure 4-1 is the mathematical expression for the computer sample-hold characteristic. For the ICPS, which has a fixed iteration rate, the T_2 value is 6.144 msec.

The preceding math model description covers the linear aspect of the ICPS hardware. However, the ICPS has a nonlinear characteristic referred to as slew rate limiting. This characteristic results from the incremental arithmetic operation having a finite limit on the incremental value that can be processed during one computational iteration. In other words, the maximum size of a variable change (increment value) that can be processed during one iteration is 64 MU (2^6). This number times the ICPS iteration rate of 162.76 solutions per sec yields a slew rate limit of 10,416 MU/sec. When the input signal reaches a rate that demands solution values at greater than 10,416 MU/sec, the ICPS output will be distorted both in phase and magnitude. The slew rate limit mathematical boundary for an input signal $A \sin \omega t$ is expressed as:

$$\frac{d(\text{input})}{dt} = \omega A \cos \omega t = 10,416 \text{ MU/sec}$$

where

A = input signal magnitude

ω = input signal frequency

From this, the slew rate limit can be seen to be directly proportional to the product of the input amplitude and frequency.

A frequency response of the basic ICPS hardware was obtained in the laboratory to compare the preceding mathematical model with laboratory results. A straight input and output function was programmed; i.e., an analog input was acquired at algorithm time 18 and that value was output at algorithm time 64. The input signal was varied to produce slew rate limiting. Figure 4-3 presents the Bode plot of the laboratory frequency response and

mathematical model (below the slew rate limit boundary). The plot shows that the math model represents the actual response very closely. It further shows that the predominate basic frequency response characteristic of the hardware comes from the prefilter element.

Figure 4-4 shows a family of frequency response curves in which the input magnitude was parametrically increased to illustrate the effect of slew rate limiting. The curves show that the slew rate limit affects the output response, for a maximum signal input (10 V) and a gain of one through the system, at approximately 6 rad. This is predictable from the preceding discussion and the recognition that the 10-V input produces 2048 MU following the A/D conversion. This slew rate characteristic may have undesirable effects on the system performance. Scaling can be used to avoid slew rate limiting; however, care must be taken to not violate resolution requirements.

Figure 4-5 shows the linear (hysteresis) characteristic of the ICPS and WWCS. These data were obtained by slowly changing the input signal positive and negative about zero. The results showed the ICPS to have very good tracking (repeatability) characteristics.

4.1.2 WWCS Hardware Characteristics

The WWCS basic hardware, like the ICPS, can be represented by a prefilter, computational delay, and zero-order hold (refer to fig. 4-1). The prefilter discussion presented for the ICPS holds for the WWCS, since the ICPS and WWCS utilized identical prefilter and A/D hardware designs.

The computational delay for the WWCS is longer than that found for the ICPS. This results from the data processing double buffering used at the WWCS computer interface to link the bit-for-bit synchronized input/output timing structure with the frame time synchronized timing structure of the central processor. This interface design is described in detail in the systems description document (ref. 1). Essentially, the input and output data processing iteratively occurs over a 6.144-msec frame time, with the central processor programs keyed to the beginning of each I/O operation. In order to avoid some data processing timing hazards, data double buffering was used to permit the central processor to use data from the I/O frame previous to the current frame and place the results in memory for outputting during the I/O frame following the current frame. Thus, the WWCS computational delay is made up of:

- 1) Input timing between the data sampling time and the beginning of the next I/O iteration
- 2) Control processor data handling time, one I/O iteration
- 3) Output timing interval between the end of the I/O iteration just past and the time at which the data were passed to the sample and hold output stage

For the basic hardware tests conducted on the WWCS, an input sampling time was selected at 1.248 msec before the next I/O iteration, and an output strobe time was selected

at 0.336 msec following the end of the previous I/O iteration. Therefore, the WWCS computational delay (T_1) for the case tested became:

$$T_1 = \begin{array}{r} \text{(Input Time)} \\ 1.248 \text{ msec} \end{array} + \begin{array}{r} \text{(Processor Time)} \\ 6.144 \text{ msec} \end{array} + \begin{array}{r} \text{(Output Time)} \\ 0.336 \text{ msec} \end{array} = 7.728 \text{ msec}$$

The delay of the sample and hold is a function of the frame time utilized for processing the particular control law. For the basic hardware tests, three frame times (sample and hold update rates) were tested. These provided sample and hold delay times (T_2) of 6.144 msec, 3 times 6.144 msec, and 8 times 6.144 msec.

Figure 4-6 shows the frequency response of the WWCS basic hardware for the three different frame times tested. The results correlated very well with the calculated response. For the 6.144-msec frame time case, the results were very similar to the ICPS, with the prefilter response having the predominate effect on the amplitude response. There was, however, more phase shift showing up in the WWCS response. This added phase shift is directly attributed to the computational delay brought on by the WWCS computer interface data processing. For the increased frame time tests, the amplitude response became more attenuated and the phase shift increased as anticipated.

The linearity characteristic of the WWCS was tested using a very low-frequency sinusoidal input. The results are shown, along with the ICPS results, in figure 4-5. As can be seen, the WWCS exhibited good linearity. No change could be discerned relative to the linearity response when the frame time was varied.

4.2 HSAS FILTER IMPLEMENTATION LABORATORY EVALUATION

Shaping filters were used very extensively in the SST flight control system to effectively stabilize the basic airframe and to improve the airplane's handling qualities. A block diagram of the filters used as part of the HSAS is shown in figure 4-7 (the general HSAS block diagram is presented in fig. 2-3). The first three filters (A, B, and C) were designed to shape the basic handling qualities of the SST, and the last filter (D) was designed to prevent coupling between the flight control system and the airframe body bending characteristic. These four filters were implemented using analog, ICPS, and WWCS electronics to evaluate performance differences that may arise from the three forms of computational processes.

The analog HSAS filters were built up using breadboard electronics, with circuits designed in the same manner as those actually to be used on the prototype SST. A careful selection of the components was made to ensure that the filter response characteristics would remain within 2% of the mathematical ideal.

For the ICPS, the HSAS filters were implemented using programmable algorithms derived from incremental difference equations. The algorithms were developed by General Electric and were contained in the ICPS User's Manual catalog of functions (ref. 4). The filter algorithm derivation is treated in more detail in section 5.2.

A wide variety of filter implementation methods was available for use with the WWCS. A trade study was therefore conducted to assess various techniques and select one for use in implementing the HSAS filters in the WWCS. This trade study is described in section 5.2. The bilinear transformation technique (or Tustin's method) was selected as it appeared to offer economy in both memory and real-time utilization.

The implemented filters in the three sets of electronics were compared using frequency response, step response, and noise response data. The results are discussed below.

4.2.1 Frequency Response Comparison

A laboratory PDP-8 computer was used to administrate the frequency response tests of the three sets of electronics (analog, ICPS, and WWCS). The PDP-8 generated a sinusoidal input command, acquired the output response, and plotted the results in a Bode plot format. The analog system was used as the baseline, following substantiation that the analog system response was within 2% of the ideal.

Figure 4-8 presents the frequency response comparison between the analog and ICPS HSAS filter responses. The ICPS response was obtained using an input signal of ± 2 V to avoid the ICPS slew rate limit characteristic. Referring to figure 4-8, the analog and ICPS performance match almost identically in the gain response, with the ICPS exhibiting some additional phase shift. This gain disparity is attributed to the prefilter characteristics of the ICPS, contributing approximately 12° of phase shift at a frequency of 10 rad.

Figure 4-9 presents the HSAS filter frequency response of the WWCS. Three frame time values (sampling rates) were tested (6.144 msec, 18.432 msec, and 49.152 msec). Included in figure 4-9 is the analog HSAS filter response to permit comparing the WWCS response to the ideal. The largest error relative to the filter gain response was with the lowest frame time (highest sampling rate). This was attributed to the difficulty in providing accurate coefficient values for the bilinear transformation equation for high sampling rates. This difficulty was related to the computer word length and the attendant coefficient truncation errors. The large phase errors for the slow sampling rate (longest frame time) were caused by the basic hardware transport delays associated with long frame time computations. A more detailed discussion of the errors introduced for filter implementations using the Tustin method is given in section 5.2.

4.2.2 Step Response Comparison

Figure 4-10 shows the step response of the HSAS filter for each of the systems tested. The forcing function was applied at the input to the A filter and recorded at the output of the D filter.

Overlaying the three systems' step responses did not bring out significant differences in their response characteristics. The WWCS was tested for the three sampling rates discussed above, without producing discernible differences in the response profiles.

4.2.3 Noise Characteristic Comparison

The residual noise characteristic of the implemented HSAS filter was checked by applying a sinusoidal signal forcing function of 47 mV, at 0.001 Hz, to the A filter and recording the D filter output on an X-Y plotter. The results are shown in figure 4-11.

The analog system output displayed excellent linear characteristics with very little noise content. The ICPS output displayed a noise content three to four times the magnitude of the analog trace. The A/D converter was found to be the primary source of the noise observed. The WWCS was again tested for the three sampling rates defined above. The results showed larger noise excursions as the frame time was expanded. The basic noise, as in the ICPS, was caused by the A/D converter. The added noise observed for the longer frame times was determined to be the result of the longer sample and hold dwell time, allowing the X-Y plotter to respond to the sample and hold output. The noise levels observed were not disconcerting, since they were well within the system required resolution of 0.05 V.

4.3 REDUNDANT HSAS PERFORMANCE

In the previous section, the implementation of the HSAS filter was evaluated on a single-channel basis. This section deals essentially with the implementation of the entire HSAS function, with tests conducted to evaluate the three types of subsystem electronics performance for a triply redundant configuration operating in a closed loop laboratory setup (refer to sec. 2.2). The redundant operations of the analog, ICPS, and WWCS configurations were not functionally identical. Cross-channel multiple voting nodes for the analog system were not considered because of the cost/failure mode concerns established during the SST system definition phase. The digital systems were organized with cross-channel data paths for evaluation during this program, since multiplexed serial digital data paths did not appear to have the disadvantages associated with the analog configuration. Figure 4-12 shows the three systems' redundant configuration data paths, in a simplified form.

The evaluations of the analog, ICPS, and WWCS HSAS configurations were conducted by observing the simulated airplane's time history for various disturbance conditions. Since the HSAS did not fully speed-stabilize the SST flight condition simulated, a weak attitude control loop was added to represent a pilot loosely maintaining a constant pitch attitude. The test conditions consisted of normal system operation response, system response with rate sensor failures, and an evaluation of the tested systems failure monitoring schemes.

4.3.1 HSAS Response—Triple-Channel Configuration

Basic triple-channel HSAS performance was established by placing a 10-sec step command as the pilot control column signal input. Figure 4-13 shows the response time histories for the three systems. The test was conducted with a minimum of channel tolerance offsets by aligning the minirig to achieve balanced load pressures across the three actuators. The step command drove both the mechanical and electrical signal paths. As a result, a small initial peak showed on the time history response reflecting the mechanical path input before the negator loop canceled the mechanical command.

The responses of the three subsystems were nearly identical (refer to fig. 4-13). The load pressure traces showed no load force fighting. There was, however, some difference in the ECS response trace. The analog system exhibited a standoff compared to the two digital systems. This was caused by the higher servo off-loading threshold, required in the analog system, to cope with the sensor to servo channel tolerance stackup eliminated in the digital systems by their voting nodes.

The traces of figure 4-14 show the HSAS response with opposite 0.5°/sec offsets placed in two of the pitch rate sensor signal paths. The sensor offsets were held within the normal tolerance band of the sensor. The load pressure traces of the analog system illustrated the resultant ECS force fight. No force fight appeared in the digital system traces because of the voting (signal selection) process that removed the offsets prior to developing the ECS command. Although all three systems met the performance requirements for this test case, the voting nodes offered an advantage by reducing the wear exposure to ECS bearings, seals, and mounting structure potentially caused by upstream sensor and processing electronics channel tolerance differences.

4.3.2 Triple-Channel Response—Pitch Rate Sensor Failure

The triple-channel redundant system HSAS design for the FCD task had to be operational following any first failure. The system, with the failure, had to provide all required functions with no performance degradation.

Figure 4-15 shows the response of the three systems with one pitch rate sensor detected as failed. The three systems' configurations at the time of the command disturbance had been altered to reflect their fault isolation moding relative to the failed rate sensor. The moding configuration changes were:

- Analog HSAS: channel associated with the failed rate sensor was disengaged.
- ICPS HSAS: hardware sensor selection algorithm for the rate sensor signals became the average of two from a midvalue selection (isolating the failed rate sensor signal).
- WWCS HSAS: software sensor selection algorithm for the rate sensor signals became the average of two from a limited average of three (isolating the failed rate sensor signal).

A discussion of the signal selection algorithms evaluated during the FCD task is presented in section 5.1.

A test was conducted to observe the response of the three systems when subjected to a passive (zero output) failure of one rate sensor signal while the remaining two were slightly offset in opposite directions. It is possible that such a failure could go undetected until an airplane maneuver is induced to force channel differences to exceed the failure monitor threshold. In the quiescent environment of the laboratory tests, this was the case for the analog system, even though the basic SST airframe was unstable. The averaging effect of the

ECS hydromechanical voting structure resolved the offset difference and provided continued smooth control, failure undetected, until a maneuver was induced.

The digital systems responses to the above test case were different from those observed for the analog system. The difference was directly related to the signal selection algorithm designs used in the digital systems. For the ICPS, the combination of a passive rate sensor (zero failure) and sensor offsets with the midvalue logic of the signal selection algorithm created a deadband characteristic. This is clearly observed in the ICPS traces of figure 4-16. A limit cycle was established following the sensor passive failure. The time period and airplane response amplitude were primarily governed by the basic airframe instability characteristics and the remaining good sensors offset values. The failure was ultimately detected by the ICPS monitoring logic, and the signal selection algorithm for the rate sensor signal was moded from a midvalue logic to an average of the two good sensor signals. The averaging process resolved the offset deadband and the airplane returned to a stable condition.

For the WWCS, the sensor signal selection was the average of all three rate signals, including the failed (zero) sensor signal, until the failure was detected. The WWCS response to the above test case is shown in figure 4-16. With the failed sensor, the WWCS loop gain was reduced to two-thirds of the original value. Following the failure detection, the signal selection algorithm was moded to the average of two, and the full loop gain was restored.

4.3.3 HSAS Failure Monitor Evaluation

A series of tests were conducted to evaluate the failure detection capability of the failure monitor mechanizations in the three redundant fail-operative subsystems— analog, ICPS, and WWCS. The digital system's failure monitors were required to provide, at a minimum, all the failure detection features found with the analog system's static, dynamic, and oscillatory failure detection monitors.

The tests were conducted in the following manner. A failure was inserted into the system. If the failure went undetected for approximately 30 sec, a 1° column pulse of 1-sec duration was applied to see if the disturbance produced a failure detection. If the failure continued to be undetected, the column pulse was again applied for a 10-sec duration. If this did not produce a failure detection, the failure was classified as undetectable. Aircraft parameters were continuously observed to note other than normal performance during the test. This manner of testing was not to serve as a rigorous failure mode study but rather to provide some means for comparing the failure monitoring performance of the three systems.

The monitor threshold values established for the ICPS and WWCS configurations were selected to be comparable, where possible, to the analog system. Table 4-1 provides a summary of the particular failures tested and test results for the three systems. The results show that the failure detection capability of the monitors implemented in the digital systems was superior to that of the analog system. It was observed that the WWCS detection of passive failures was slightly better than found with the ICPS.

TABLE 4-1.-FAILURE MODE RESPONSE

Failure conditions	Failure indicated		Analog Stimuli Required		Failure indicated		ICPS Stimuli Required		Failure indicated		WPCS Stimuli Required		
	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	
First Failures													
(1) Pitch rate Hardover	x		None				None					None	1. HISAS filter will not allow OSC frequency to appear on ECS failure monitors. 2. δ_c step input does not allow sufficient error to appear on ECS failure monitors.
OSC	x		Does not want to detect ¹				None				None		
Slow over	x		None				None				None		
Passive (open circuit)	x		Failure indication with 10 sec δ_c pulse				Failure indication with 1 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
High gain = 10	x		Failure indication with 10 sec δ_c pulse				Failure indication with 1 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
Low gain = 0.1	x		Failure indication with 10 sec δ_c pulse				Failure indication with 1 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
(2) ECS failure													
Servo drive open	x		Does not want to detect ²				Failure indication with 10 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
Open feedback	x		None				None				None		
Hardover	x		None				None				None		
Passive (power ch 2)	x		Does not want to detect ²				Failure indication with 10 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
(3) Negator loop (ch 2)													
Open	x		Does not want to detect ²				Failure indication with 1 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
Hardover	x		None				None				None		
(4) Column input													
Passive	x		Does not want to detect ²				Failure indication with 1 sec δ_c pulse				Failure indication with 10 sec δ_c pulse		
Hardover	x		None				None				None		
(5) Trim command													
Passive	x		Failure indicated with trim command				Failure indication with trim command				Failure indication with trim command		
Hardover	x		None				None				None		
(6) Lyn-syn													
Passive	x		Does not want to detect ²				Failure indications with 10 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
Hardover	x		None				None				None		
(7) Digital													
D/A ch 2	N/A						None				None		
Loss of power	N/A						None				None		
Ch B interface unit	N/A						None				None		
Ch B computer unit	N/A						None				None		
Ch B memory unit	N/A						None				None		
Second unlike failures													
(1) ECS A hyd off							None				None		
Pitch rate B 1° step							None				None		
ECS B power off							None				None		
Pitch rate A passive							Failure indication with 10 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
(2) Pitch rate A passive							Failure indication with 1 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
ECS B hyd off							None				None		
(3) Pitch rate A passive							Failure indication with 1 sec δ_c pulse				Failure indication with 1 sec δ_c pulse		
ECS B hyd off							None				None		
(4) Pitch rate A passive							None				None		
ECS B hyd off							None				None		
(5) Interface unit A failed							None				None		
ECS B hyd off							None				None		
(6) Computer (A) off							None				None		
Computer (A) off							None				None		
Pitch rate B hardover							None				None		

4.4 CONCLUSIONS—HSAS LABORATORY TESTS

- Digital systems basic hardware characteristics can be mathematically modeled for analysis purposes. The analog input signal prefilter values must be included in the hardware model and, depending on the sample rate selected and aliasing protection desired, can have a large influence on the overall system stability and bandpass capability.
- Application bandpass limitations result from the ICPS slew rate characteristic and the WWCS input/output double buffer transport delay. However, both of these systems easily exhibit control law processing capabilities adequate for functions associated with rigid airframe stability augmentation and automatic flightpath control.
- All three systems could achieve the HSAS performance requirements, short of the digital systems D/A resolution limitation. Redundant operations of the digital systems were found to be indistinguishable from the excellent simplex performance-matching capability between the digital and analog systems. No performance irregularities were uncovered relative to the ICPS and WWCS cross-channel voting data paths.
- Failure detection schemes within the digital systems offer substantially more versatility in failure detection threshold selection than found with an analog system.

5.0 SPECIAL STUDIES—REDUNDANT DIGITAL SYSTEM FUNCTIONS

The tests described in section 4.0 essentially illustrate that a system such as the HSAS can be realized using digital technology. The tests were conducted from a total system overview, with the evaluation made after the HSAS functions had been established within the three forms of computational electronics. This section deals with the trade studies and special investigations conducted to evaluate or select the various functional elements that went into the makeup of the redundant digital system configurations. Section 5.1 presents the study made to develop a signal selection/failure detection algorithm for the WWCS, with comparisons drawn relative to the analog and ICPS schemes. Section 5.2 discusses an investigation made to assess the dynamic and static response of digital filter implementations. Section 5.3 treats the study made to define protection schemes for preventing computational overflow in the two digital systems. Section 5.4 discusses special tests conducted with the WWCS servo transmitter/receiver unit (STRU).

5.1 SIGNAL SELECTION/FAILURE DETECTION

All redundant systems require some form of signal selection and failure detection if fault isolation is required. Many approaches can be taken relative to the signal selection/failure detection (SSFD) mechanization to satisfy the application model safety, performance, and mission reliability requirements. The study conducted for the FCD task was restricted to treating a triple-channel fail-operative system, where all three channels were to be active and on line for normal system operation.

Three considerations are identified with the selection of an SSFD function for a digital system:

- 1) Algorithm architecture—the design of the SSFD so that the system performance will be acceptable for normal, failure transient, and failure isolation operation
- 2) Mechanization method—implementation of the algorithm in dedicated hardware or software, or a combination of both
- 3) Placement—physical location of the SSFD function relative to the hardware elements that make up the system signal path, sensors through servos

The following discussion deals generally with SSFD functions that process sensor input information. However, for purposes of comparison, any system voting/monitoring function (e.g., the ICPS majority logic voter) is considered an SSFD device.

5.1.1 SSFD Algorithm Design

In this section, the general characteristics of several "variable" signal selection/failure detection functions are discussed. To begin with, an introduction is presented on the signal output characteristics found when three signals are processed by a median selection or averaging algorithm. A summary discussion is then presented on the failure monitoring approach used with the signal selection algorithms. This introductory material is followed

by detailed discussions on specific SSFD algorithms simulated and evaluated using an EAI 8400 digital computer. Finally, a summary is given of the closed-loop performance comparison tests made on the SSFD algorithms investigated.

5.1.1.1 Median Selection Versus Averaging Algorithm

The median selection and averaging algorithms are the most common and basic functions for deriving one signal that is representative of three similar signals. Median selection simply provides a signal output that is the middle signal of the three incoming signals. If two signals are identical, the median will be that signal. Averaging provides a signal output that is the average of the three incoming signals.

The median selection process output signal can exhibit an abrupt change when one signal fails to a hardover state and that signal was in, or passed through, the middle region of the three signals. Used in a flight control system, this will cause a transient disturbance whose effect depends upon the control law gain and airplane dynamic response. This characteristic must be carefully evaluated in a flight-critical system application using closed-loop simulation analysis. Another failure state case that must be considered if a median selection process is to be used is a signal failure to or near zero if the three reference signal's normal excursions are about zero. This case (where the two remaining active signals have bias offsets opposite one another creates an effective deadband, about zero, equivalent to the two remaining signal's difference. Such a deadband can cause a limit cycle reaction in a closed-loop system, depending on the airplane's dynamic characteristics and specific feedback parameter being processed.

The averaging process does not create a deadband effect when failures occur but will exhibit a transient output with a signal hardover failure potentially causing a larger disturbance than found with the median process. The larger disturbance would result from the fact that, until the failed signal is removed, the failed signal continues to contribute to the output signal value. Another characteristic of the average process with a failed signal is that the path (loop) gain is reduced until the average process is moded to use only those signals that are not failed.

Figure 5-1 shows the median selection and average output signals for three sinusoidal inputs of slightly different frequencies. This example is used to illustrate the signal discontinuities that can occur with median selection. It also illustrates how the average process tends to follow the drifting A input, while the median selection essentially ignores it.

Noise rejection qualities of the two signal selection algorithms were assessed analytically. Figure 5-2 shows the probability of the median selection and average output signal error (noise) exceeding a given value for various standard deviations of the input signal noise (error). It was assumed, in calculating the curves presented in figure 5-2, that the input noise had a Gaussian density distribution with a zero mean value. For purposes of comparison, figure 5-2 also includes the noise signal exceedance probability for a simplex signal. The data show that both the median selection and averaging reduce the noise content probability. The averaging process, however, provides the greatest reduction of noise content in the output signal. This same trend can be observed in the time history traces

shown in figure 5-1. The mathematical derivation of the probability curves presented in figure 5-2 is detailed in appendix A, section A.1.

5.1.1.2 Failure Detection Algorithm

The most common technique of failure monitoring in a redundant system is to take the difference of two like signals and set a failure detection threshold at some value of that difference (with or without a time delay). Failure isolation can then be provided in a triplex system by using the mutual difference of all three signals, where the faulty signal can be paired off and identified. Two methods are available for deriving the failure monitoring difference signal, in conjunction with a system signal selection function. The basic method is to compare (take the difference) of the signals coming into the signal selection algorithm. The second method is to take the difference of each incoming signal relative to the output of the signal selection algorithm. The failure detection thresholds are generally set as small as possible. As the threshold is reduced, however, the probability of a false failure detection (as a function of the signal noise content) increases.

An analysis was made to assess the basic characteristics of the signal difference between the signal selection output and input for both the median selection and averaging algorithms. Figure 5-3 shows the probability of this difference exceeding a given value for the largest of the three input errors. The probability curves indicate that, if the signal differences across the selection algorithm are to be used for failure detection, the threshold of the monitor for the averaging algorithm can be set lower than the threshold of the monitor for the median selection algorithm. This assumes that both monitors would have the same probability for a false failure indication. The mathematical derivation of the data presented in figure 5-3 is given in appendix A, section A.2.

A potential problem to be recognized when difference signals are to be used for failure identification is the situation where one difference signal set indicates a failure and the other two do not. This can result from the tolerance differences that may exist in the failure detection hardware elements or signal conditioning elements from one channel to another. For example, assume three inputs A, B, and C. If A-B is greater than a given threshold (failure detected) and B-C and C-A are less than the threshold, it cannot be ascertained which signal is at fault, A or B. For this situation, a first-failure indication should be registered, but failure identification (isolation) action cannot be taken. Should the faulty signal (A or B) fail further in magnitude, the monitor will identify the failed signal and the first-failure status will be unchanged.

The following describes in detail four SSFD algorithms that were investigated. Others that were studied had less potential or their desirable features were adapted into the described algorithms.

5.1.1.3 Median-Selection/Average-of-Two

The median-selection/average-of-two title refers to the SSFD operation for both the no-failure and first-failure states. Median selection is used on all incoming signals (triplex) until a failure has been detected. The algorithm then modes to the average-of-two for that

signal set having one signal classified as failed. The algorithm is multiplexed to process each sensor type. An input signal cross-comparison method was used for the monitor associated with this signal selection algorithm.

Figure 5-4 depicts the median-select/average-of-two SSFD studied. Moding the SSFD from median selection following a failure detection is directed at avoiding the potential deadzone/limit-cycle state that may result from a signal failing to zero and the remaining signals having opposite bias errors. The failure monitoring scheme shown in figure 5-4 utilizes real-time filters to achieve failure detection frequency separation on the difference of the incoming signals. Since most sensor predominate error occurs over a given frequency band, the frequency separation technique was employed to optimize failure detection for each sensor type normal error and failure response characteristics. The failure counter following the failure detection thresholds, downstream of the signal difference filters, is used to provide further discretion of the permissible errors versus failure states of each sensor.

In the SSFD of figure 5-4 there are six monitor parameters held to be selectable (programmable) for each type of sensor signal processed. These parameters (time constants T_1 and T_2 , thresholds TH_1 and TH_2 , and time delays TD_1 and TD_2) permit optimizing the failure detection logic for each sensor (incoming signal) type. Figure 5-5 graphically illustrates the relationships between the failure detection thresholds and break frequency of the lag and washout filters. For a sinusoidal difference signal input, figure 5-5 shows how failure detection can be more sensitive for midband frequencies. Figure 5-6 is an alteration of the parameter values of figure 5-5 to illustrate how upper frequency bands can be desensitized compared to the low-frequency zone. These two examples show the flexibility of the failure monitor construction for oscillatory failure states. In order to select the desirable monitor parameter values, however, transient as well as oscillatory failure responses of the difference signal must be analyzed. This must generally be done using closed-loop simulation to evaluate the failure detection sensitivity relative to the airplane's response (if any) to the failure condition.

5.1.1.4 Average-of-Three/Average-of-Two

This SSFD algorithm is identical to the one described above with the exception that the median-selection function is replaced with the average-of-three function. The algorithm organization is therefore represented by figure 5-4.

The difference between the median-selection and average-of-three will be in the response of the system (airplane) during a failure transient state before the SSFD is moded to the average-of-two. The average-of-three will always permit the failed signal to affect the selected output equal to one-third the faulty input signals value. For an input signal failure to zero, the effective path gain is reduced by one-third.

5.1.1.5 Lag-Equalized Median-Selection

The basic median selection algorithm output is subject to signal discontinuities and abrupt failure-induced transients when compared to an averaging function. These discontinuities and abrupt responses can be softened by inserting an equalization loop around the

median-selection function. Figure 5-7 presents the lag-equalized median-selection SSFD algorithm with a monitor that is a modified version of the cross-selector method. The selected output is the median signal of the lag-equalized input signals, where the input signals are continuously being equalized toward the selected output relative to a $K_L/T_L S + 1$ function. The degree of input signal compensation (equalization) is controlled by the gain K_L and the rate of compensation is controlled by the time constant T_L , parameters that can be selected differently for each sensor (signal) type.

Failure monitoring is similar to the monitoring functions applied to the previous two SSFD algorithms, except that the input signal differences are not used; rather, the compensated (lag-equalized) signal differences are used. In addition, a monitor function was placed on the equalization path to detect signal bias failures that may exceed the desired bias error to be equalized. A first-failure detection, and resulting failure identification, will mode the SSFD median-selection process to hold constant the equalized value of the faulty signal path and continue to operate using the median selection process. This form of first-failure moding would not be effective on other than signals operating about zero.

5.1.1.6 Compensated Limited Average

The compensated limited average (CLA) SSFD algorithm was developed to utilize the apparent strengths of both the median-selection and averaging functions without being subject to the inherent weaknesses of those functions. The objective was to use primarily the averaging function to gain a smooth output without incurring output signal excursions as a result of input signal failures. The CLA is organized into three functional stages: a bias error compensation stage, a limited average selection stage, and a failure monitoring stage. Figure 5-8 is a block diagram depicting the operation of the CLA SSFD algorithm.

In order to provide an average output that would not be affected by a failed signal, the algorithm output was set up to mode from the average-of-three to the average-of-two for very small differences of the input signals. To keep the moding threshold small, the input signal bias errors were removed using a bias error compensation stage. In the bias error compensation stage (refer to fig. 5-8), each input is compared to the average of the three input signals and equalized toward the average using a limited quasi-integration function. This compensation is processed via a feedforward path rather than with a lag function feedback path, as described in the previous section, to effect 100% equalization in a steady-state condition. The bias error compensated input signals are the input signals to the algorithm final averaging output stage and are used to effect the output stage average-of-three to average-of-two moding.

Limited average refers to the final averaging output stage, where the average-of-three is the selected output only when the compensated input signal values remain within a given (limited) difference of their median value. If one of the compensated signals goes beyond the preestablished limit, a DO NOT USE signal modes the output stage to the average of the other two compensated signals. If the DO NOT USE signal remains for a specific period, a failure would be registered.

The failure monitoring stage encompasses a failure detection monitor for the bias differences (static failure detection), the DO NOT USE logic (dynamic failure detection),

and failure latch logic. Once a failure has been latched, the average-of-three/average-of-two functions remain in the average-of-two mode, and the median-selection function replaces the failed compensated signal with the median output before the failure.

This SSFD algorithm has the versatility of moding between the average-of-three and average-of-two depending on the compensated signal differences and the time duration a difference exceeds a given threshold. Thus, the input signal failure excursions are blocked from significantly influencing the algorithms output, and signal transient differences can be ignored for a given period to reduce the exposure to false failure indications. Selection of failure detection thresholds, DO NOT USE count-out values, and bias compensation rate limit values can be different for each signal (sensor) type.

5.1.1.7 SSFD Performance Comparison

The four SSFD algorithms described above were evaluated using the simulated SST airframe having a simplified ECSS control law. Various types of sensor signal failures were introduced; e.g., hardover, ramp, passive, oscillatory, step, etc. The results are summarized in table 5-1. It should be noted that the basic SST airframe response for the flight condition simulated was extremely unstable; i.e., it had a pure divergence mode with a time to double amplitude of about 2 sec.

Both the primary median-selection SSFD functions resulted in some limit cycle failure mode responses due to the basic airframe instability characteristic. The system with a pure averaging SSFD function exhibited some undesirable large transient responses to rapidly changing signal failure modes. The CLA function provided the best overall failure mode performance and could easily be viewed as superior to the other three SSFD functions.

5.1.2 SSFD Software Implementation

The SSFD design study conducted for the FCD task was primarily to select an input sensor SSFD algorithm to be specified for the whole-word computer subsystem (WWCS). It has been determined that the WWCS SSFD algorithm would be implemented in software since the incremental (ICPS) system's SSFD algorithm had been mechanized in hardware. Even though the implementation method should be carefully considered (software, hardware, or both) relative to cost, performance, system testing, etc., the WWCS software approach would provide effective data to compare to the ICPS hardware implementation.

As part of the design selection process, the memory and computational time for each of the SSFD functions described above were compiled. This compilation is shown in table 5-2. The software parameters are based on the algorithms being programmed as subroutines. Observations that can be drawn from the table are:

- The CLA, even with its complexity, is well within the total memory and timing levels of the other algorithms.
- The signal selection portion of the median-selection algorithm requires twice the memory and time for that same portion of the averaging algorithm.

TABLE 5-1.--SSFD PERFORMANCE COMPARISON

Failure condition	SSFD algorithm	Median-selection/ average-of-two				Lag-equalized median-selection		Compensated limited average (CLA)
		Median-selection/ average-of-two	Average-of-three/ average-of-two	Average-of-three/ average-of-two	Lag-equalized median-selection	Compensated limited average (CLA)		
Step to null	Threshold not exceeded	0.1 g step followed by 0.1 g limit cycle	0.033 g step followed by reduced damping (2/3 gain)	0.1 g step followed by >> 0.1 g limit cycle	0.1 g step followed by >> 0.1 g limit cycle	0.033 g step followed by reduced damping (2/3 gain)	0.033 g maximum transient	
	Failure detected	0.1 g maximum transient	>> 0.1 g transient	>> 0.1 g maximum transient	0.1 g maximum transient	0.033 g maximum transient	0.033 g maximum transient	
	Failure detected	0.1 g step transient	>> 0.1 g transient	0.1 g step transient	0.1 g step transient	0.033 g maximum transient	0.033 g maximum transient	
Ramp to hardover	Failure detected	Output follows ramp input with 0.05 g transient to average-of-two	Output 1/3 ramp input with negligible transient to average-of-two	Output follows ramp input with negligible transient to average-of-two	Output follows ramp input with negligible transient to average-of-two	Output 1/3 ramp input with negligible transient failure detection	Output 1/3 ramp input with negligible transient failure detection	
	Threshold not exceeded	Follows input ramp rate to zero, then 0.1 g limit cycle	1/3 input ramp rate toward zero and reduced damping	1/3 input ramp rate toward zero, >> 0.1 g limit cycle	Follows input ramp rate to zero, >> 0.1 g limit cycle	1/3 input ramp rate toward zero and reduced damping	1/3 input ramp rate toward zero and reduced damping	
Ramp to null	Failure detected	Input ramp rate toward zero with < 0.1 g transient to average-of-two	1/3 input ramp rate toward zero with < 0.1 g transient to average-of-two	Input ramp rate toward zero with < 0.1 g transient to average-of-two	Input ramp rate toward zero with < 0.1 g transient to average-of-two	1/3 input ramp rate toward zero with negligible transient failure detection	1/3 input ramp rate toward zero with negligible transient failure detection	
	Threshold not exceeded	< 0.1 g step transient	< 0.033 g step transient	< 0.1 g step transient	< 0.1 g step transient	< 0.033 g step transient	< 0.033 g step transient	
Bias level shift	Failure detected	0.1 g step transient	>> 0.1 g transient	>> 0.1 g step transient	0.1 g step transient	0.033 g maximum transient	0.033 g maximum transient	
	Failure detected	Follows oscillation, then < 0.1 g transient to average-of-two	1/3 oscillation magnitude, then >> 0.1 g transient	Follows oscillation, then < 0.1 g transient to average-of-two	< oscillation, then < 0.1 g transient	< 1/3 oscillation magnitude, then < 0.033 g transient	< 1/3 oscillation magnitude, then < 0.033 g transient	
Limit cycle oscillation	Threshold not exceeded	Follows limit cycle magnitude	1/3 limit cycle magnitude	Follows limit cycle magnitude	< limit cycle magnitude	< 1/3 limit cycle magnitude	< 1/3 limit cycle magnitude	
	Failure detected	Follows oscillation, then < 0.1 g transient to average-of-two	1/3 oscillation magnitude, then >> 0.1 g transient	Follows oscillation, then < 0.1 g transient to average-of-two	< oscillation, then < 0.1 g transient	< 1/3 oscillation magnitude, then < 0.033 g transient	< 1/3 oscillation magnitude, then < 0.033 g transient	
Excessive random noise	Failure detected	< noise level, then < 0.1 g transient	< noise level, then >> 0.1 g transient	< noise level, then >> 0.1 g transient	< noise level, then < 0.1 g transient	< noise level, then < 0.1 g transient	< noise level, then < 0.1 g transient	
	Failure detected	< noise level, then < 0.1 g transient	< noise level, then >> 0.1 g transient	< noise level, then >> 0.1 g transient	< noise level, then < 0.1 g transient	< noise level, then < 0.1 g transient	< noise level, then < 0.1 g transient	

*Single sensor failure, worst case signal tolerance distribution.

TABLE 5-2.—SSFD MEMORY AND COMPUTATIONAL TIME COMPARISON

SSFD algorithm Software requirements		Median-selection/ average-of-two	Average-of-three/ average-of-two	Lag-equalization median selection	Compensated limited average (CLA)
Memory (words)	Algorithm setup	42	42	42	42
	Signal selection	56	28	65	99
	Compensation	0	0	31	51
	Failure detection	94	94	100	28
	Total	192	164	238	220
Additional memory per each new sensor (signal) type		23	23	26	19
Timing (μ sec)	Algorithm setup	81	81	81	64
	Signal selection	170	80	190	268
	Compensation	0	0	210	169
	Failure detection	420	420	440	90
	Total	671	581	921	591

- The lag-equalized median-selection algorithm is the most inefficient, both in memory and time, of the four algorithms.
- Failure detection is a large portion of the computational time, except for the CLA algorithm.

5.1.3 SSFD Points—Analog, ICPS, and WWCS

The following sections provide a summary description of the signal selection/failure detection points in the three systems studied. The placement of the SSFD points in the digital systems was not selected to meet a given system reliability requirement but was selected by the equipment supplier to maximize the subsystem (triplex system electronics) functional survivability by providing failure isolation for each line replaceable unit. A system reliability analysis relative to the SSFD placement is presented in section 5.1.4.

5.1.3.1 Analog SSFD Configuration

Signal selection (voting) nodes and associated failure detection logic for the analog HSAS appear in only one place. The analog system SSFD node is the force summing, authority limited, hydromechanical coupling point at the output of the electric command servos. Three identical paths (channels) of system elements are upstream of this point, completely isolated from one another. This configuration is often referred to as a brick-wall system and is represented by the block diagram shown in figure 5-9.

The hydromechanical signal selection function is a form of a limited average algorithm. As long as all servo commands agree within a specified force level, the output is essentially the average of the three commands. If one servo command exceeds its force authority, the output is approximately the average of the two other servo commands. In order to control the force differences resulting from sensor and electronic component tolerances, proportional and pseudointegral equalization feedback was incorporated into the servo system configuration. Proportional equalization compensated for dynamic differences (gradient errors) in the signal paths, and lag-hold (pseudointegral) compensation was used to offset steady-state and/or quasi-steady-state bias errors of the signal path.

Failure detection logic for the analog system hydromechanical signal selection node consists of separate dynamic, static, and oscillatory failure monitors. All three of these monitors are essentially an inline cross-selector type. Additional descriptive information about these monitors is presented in section 3.1. The monitors were implemented with dedicated hardware having fixed thresholds for failure detection.

The analog system SSFD, located at the end of the sensor/computational string of electronics, makes it difficult to select optimum failure detection thresholds that assure safety without causing an undue level of exposure to false failure indications. The monitor thresholds are a compromise relative to each sensor's failure modes. Since a significant portion of the tolerance levels is generated from the sensors themselves, the above problem could be substantially reduced if each sensor signal was voted and monitored before it was blended into the control law. However, for complex systems, it would be extremely costly

to implement SSFD functions for each signal with dedicated hardware. Therefore, most analog redundant systems are of the brick-wall design approach.

5.1.3.2 ICPS SSFD Configuration

Digital system electronics can be effectively multiplexed to provide SSFD functions for each input sensor signal. Where desired, these electronics can be programmed to process each signal with different SSFD algorithm parameter values. Therefore, the ICPS was developed with a programmable SSFD hardware algorithm to be used to process all incoming variable signals. In addition, the ICPS utilizes majority logic voting/monitoring nodes to isolate line replaceable unit failures. This latter SSFD function involves very little hardware because of the bit-for-bit multichannel processing synchronization used in the ICPS. Three types of SSFD algorithms exist for the HSAS function implementation using the ICPS. They are:

- 1) Variable signal selection/failure detection—a median-selection/average-of-two algorithm essentially identical to the one presented in section 5.1.1.3
- 2) Majority logic signal selection/failure detection—an algorithm that outputs on a bit-for-bit serial processing basis, a bit that reflects the majority of three incoming bits. Details of this function are presented with the ICPS description in reference 1, section 5.2.5
- 3) Hydromechanical SSFD algorithm, previously described for the analog system

The variable SSFD algorithm mechanized in the ICPS automatically resets the failure counter functions (refer to fig. 5-4) every 3.14 sec, regardless of the status of the failure count. This reset scheme creates some failure detection potential ambiguity. For instance, if a failure occurs 1 sec before the counter resets and the failure detection count threshold is 1.5 sec, the actual failure annunciation takes place with an effective time count of 2.5 sec. The effect of the counter reset on the determination of an oscillatory failure detection threshold is plotted in figure 5-10. The plot shows the sensor input signal amplitude versus frequency that is required before a failure will be registered (case shown for counter threshold of 1.57 sec, threshold comparator values of 1.25% of full scale, and lag and washout time constants of 0.015 sec.

5.1.3.3 WWCS SSFD Configuration

The signal selection/failure detection functions of the WWCS are similar to those presented for the ICPS; that is, the HSAS function implemented using the WWCS contains three types of SSFD algorithms. Two are identical to the ICPS; majority logic signal selection/failure detection and the hydromechanical algorithm. The other, although a variable SSFD function, is mechanized in software rather than in hardware, as was the case for the ICPS. The WWCS variable SSFD algorithm is the compensated limited average function discussed in section 5.1.1.6.

It should be noted here that the ICPS hardware mechanized SSFD is much faster than the WWCS CLA (or equivalent software median-selection/average-of-two). The processing

time for the ICPS algorithm is 96 μ sec; for the WWCS CLA algorithm, the time is 591 μ sec, an approximately six-to-one speed advantage of processing an SSFD function with dedicated hardware versus software. This fact should be carefully reviewed when specifying a system that would be processing a large number of sensors, keeping in mind that the SSFD processing time percentage for the HSAS, ECSS, and CWS implementation with the WWCS ran between 51% and 69% of the total processing time (refer to sec. 3.4).

5.1.4 SSFD Placement—Mission Reliability Analysis

Signal selection/failure detection (voting) nodes in a triplex redundant flight control system are used to isolate failure transients from disturbing the airplane's flightpath and to extend the operational reliability of the flight control system. A large number of SSFD points in a system appears to extend a system's mission reliability by providing a high level of functional survivability for many dissimilar failure states. This concept, however, can be observed to have diminishing returns when the unreliability of an additional SSFD function degrades the system reliability more than the voting node extends the mission reliability. In order to gain some insight into this subject for the systems evaluated in the FCD task, an analysis was made to determine the mission reliability sensitivity of each SSFD point in the analog, ICPS, and WWCS configurations. The analysis results should be viewed relative to the mission reliability trends shown for each configuration and not as a direct comparison of the analog, ICPS, and WWCS equipment, since the three systems were not developed from identical application and hardware design specifications.

The analysis was directed toward finding the probability of system failure (loss of function) for a 3-hr mission. Five configurations of the ICPS and WWCS were investigated, starting from a simple single-channel system and proceeding in steps of more redundancy and SSFD arrangements until the current configurations were reconstructed. For each configuration modification, the mission reliability was calculated. The study utilized the SST HSAS and ECSS control system functions as the application models. No attempt was made to size the computational electronics of the ICPS and WWCS to be directly compatible with the above control system's functional needs.

5.1.4.1 Configuration Modification Definitions

Configuration 1 is a *single-channel* configuration where all hardware was eliminated that pertained to redundant fail-operational/fail-passive operation. Figure 5-11 illustrates the single-channel configurations evaluated. No redundancy provisions remained in the analog system, and the ICPS and WWCS contained a simplex clock and one power supply per line replaceable unit.

Configuration 2 is a fail-operational *brick-wall* configuration made up of three of the above single channels whose outputs are tied together through a hydromechanical SSFD voting node. All three of the brick-wall systems required equalization; i.e., the digital systems remained asynchronous. This configuration was evaluated for both single and dual power supplies per line replaceable unit. A dual power supply permits primary LRU reference power to be derived from two (or one of the two if a power failure occurs) asynchronous 400-Hz airplane power buses. A configuration alteration was added because of the significance of the power supply modules and aircraft power reference system's

unreliability. The brick-wall configuration was the extent of the analog system configuration studies. Figure 5-12 depicts the brick-wall configuration.

Configuration 3 added input/output data synchronization and an input SSFD to the brick-wall digital systems. The input SSFD was mechanized in a dedicated hardware module for the ICPS and in software for the WWCS. The added hardware included the primary/secondary fail-operational clock logic and iteration reset majority logic voter (MLV) in the computer interface units (CIUs) of the ICPS and WWCS. This modification synchronized the data processing and, as a result, the servo command outputs of the three channels became identical. Figures 5-13 and 5-14 show the entire SSFD structure for the ICPS and WWCS, respectively. Those elements labeled "configuration 3" are the items added to the basic brick-wall configuration.

Configuration 4 placed operational fault isolation logic in the computer unit LRU so that the computer unit became independent of its channel's CIU input data. This modification added voted timing logic (clock and iteration reset), which allowed the computer unit to continue processing input data if failures occurred in its CIU data processing or timing circuits. The added circuits are shown by the "configuration 4" notation in figures 5-13 and 5-14.

Configuration 5 is the completed reconstruction of the ICPS and WWCS configurations. The output timing and data voting nodes are added to configuration 4, providing complete independence of operation between the computer interface unit(s) and computer unit. With this configuration, computer unit failures (system first failures) will not affect the servo output commands. The configuration 5 notations in figures 5-13 and 5-14 identify the added circuits to complete the ICPS and WWCS configuration buildup.

5.1.4.2 Analysis Ground Rules and Assumptions

The ground rules and assumptions listed below were followed in developing the reliability analysis results:

- 1) A 3-hr use of the HSAS or ECSS function was defined as a mission.
- 2) Mission failures were defined as:
 - a) Any failure in a single-channel configuration resulting in the loss of the flight control function
 - b) Any failure combination within the three channels of a brick-wall configuration resulting in the inability of two of the three channels to perform the flight control function
 - c) Any failure combination within a triple channel multiple SSFD point system resulting in a loss of function in two different channels between the same SSFD points

- 3) Servo monitoring for ICPS and WWCS brick-wall configurations would use the analog system monitoring circuits.
- 4) Servo monitoring of the ICPS and WWCS configuration 3 would utilize the input SSFD algorithm.
- 5) Three airplane electrical and hydraulic primary systems would exist for the triple-channel configurations.
- 6) Control surface power control units, linkage, and related hydromechanical detent failures were to be assumed negligible.
- 7) Preflight test and built-in-test circuits used to check the SSFD points prior to dispatch were not to be included in the mission reliability calculations.
- 8) Reliability computations for the ICPS were to be based on the use of a solid-state memory in the computer unit (no separate memory LRU).

Reliability data used in the analysis came from Boeing 747 autopilot reliability estimations for the analog system circuits, General Electric reliability estimations for the digital system circuits, and Boeing reliability records for sensor, electrical, and hydraulic system elements. No reliability number was associated with the digital systems software, since the software was assumed to be completely validated through systematic desk and laboratory analysis.

5.1.4.3 Reliability Analysis Results

The probability of mission failure for the configurations presented above was calculated for both the HSAS and ECSS flight control system functions. These two system functions required different levels of digital equipment complexity for their implementation. For instance, some computational circuits or special registers of the ICPS and WWCS used to implement the ECSS function were not used in implementing the HSAS function, a fact that must be considered for conducting a rigorous reliability analysis. For this study, a fixed failure rate of the computer unit was used regardless of the control function mechanized. Therefore, the difference between the HSAS and ECSS computed reliability results from the increased number of sensors utilized by the ECSS function. The data presented should not be viewed as an absolute indication of the evaluated system's reliability but should be compared as the configuration changes, to gain an insight into the mission success diminishing return as the systems grow more complex with a higher level of dissimilar failure survivability.

A summary of the mission success probabilities of the HSAS and ECSS functions for the various configuration changes made is presented in figures 5-15 and 5-16, respectively. Naturally, the single-channel configurations have the highest probability of mission failure. The single-channel calculations do provide some indication of the relative complexity of the three hardware types, keeping in mind that the analog system was made up of only the essential elements for the HSAS and ECSS mechanizations. The single-channel differences between the ICPS and WWCS stem from the basic complexity difference of the related

computer units. However, the numbers reflect the computer units evaluated in the laboratory, where the computer unit of the WWCS had substantially more computational capacity (resident program capability) than did the ICPS computer unit. The higher probability of mission failure for the ECSS single-channel systems over the HSAS single-channel systems points out the influence of the added, more complex (unreliable), sensor systems.

When three single-channel systems were combined through the hydromechanical SSFD output voting node (brick-wall configuration), a significant improvement in mission reliability was observed. This improvement, however, heavily depends on the failure rate of the hydromechanical SSFD elements. For example, if the failure rate of the SSFD is higher than one-third of the single channel system failure rate, using three channels and an SSFD node would not improve the mission reliability over a single-channel system. The general effect of the SSFD failure rate on the brick-wall configuration reliability relative to the reliability of a single-channel system is shown in figure 5-17. The mathematical derivation of the data presented in figure 5-17 is given in appendix A, section A.3.

Dual power source reference capability for the brick-wall systems (major LRUs) improves the system reliability by an additional large increment (refer to fig. 5-15 and 5-16). This somewhat demonstrates the influence of the aircraft's primary power system failure rate on the total system's basic reliability.

Adding the input SSFD (configuration 3) did not effectively improve the system reliability for the HSAS function, since the HSAS utilized very few sensors and their failure rates were very low. The input SSFD for the ECSS function, however, does produce a desirable reliability advantage, illustrating the benefit of an input SSFD where a large number of sensors and/or high sensor failure rates exist. The input SSFD for the WWCS was mechanized in software. This does not add to the unreliability of the hardware elements as compared to the ICPS hardware SSFD. Therefore, the WWCS ECSS reliability improvement with an SSFD is better than that achieved for the ICPS ECSS, even though some memory unreliability was added to account for the WWCS SSFD function (refer to fig. 5-16). This illustrates an advantage of software SSFD functions over hardware SSFD functions.

Very little overall system reliability improvements resulted from the configuration 4 and 5 modifications. For comparison purposes, the dual power modification reliability improvement of configuration 5 is shown in both figures 5-15 and 5-16. Again, the dual power arrangement provided a significant level of system reliability improvement. While the configuration 4 and 5 changes did not directly improve the system reliability, a high level of LRU fault isolation was achieved.

It should be noted here that the evaluation of the SSFD placement, as presented in this section, was conducted from a mission reliability point of view. Some form of SSFD voting and monitoring may be highly useful for identifying and isolating failures as a maintenance aid. Thus, the number and location of SSFD functions will be determined from both operational and maintenance requirements.

5.2 DIGITAL FILTER IMPLEMENTATION

The purpose of a stability augmentation flight control system on an aircraft is to enhance the pilot's ability to control the vehicle and hence improve safety. Flight control system electronics normally process filters that selectively amplify or attenuate the basic frequency response of the airframe. The synthesis of these filters has generally been made in the continuous S domain. With the application of digital computers to airborne flight control systems, such filters will be implemented in a sampled-data environment, and filter synthesis could be made in the discrete sampled Z or W domain. However, much design is still accomplished in the continuous domain, and the functions are translated into a sampled-data representation. For example, on the SST the control wheel steering autopilot function was designed using standard S domain techniques, although the plan was to mechanize the function on a digital computer.

When filters are designed in the continuous domain and mechanized in the discrete (digital) domain, some disparity in response is expected because of the finite word length and computational rate associated with the airborne digital computer. The disparity, however, would be acceptable if kept within reasonable limits for the particular flight control system. As part of the FCD task, a study was made to determine the static and dynamic performance difference between sampled-data and continuous implementations of filters. The approach taken in the study was to program a second-order filter for the digital computers of the ICPS and WWCS and compare their response with the continuous domain theoretical response.

For the ICPS, a standard algorithm developed by General Electric was used to implement the second-order filter. The processing rate of the ICPS computer is fixed by hardware, providing a solution iteration of 163 times per second.

There are many ways to implement a second-order filter using the WWCS computer. The general-purpose nature of the central processor unit permits the variation of the solution iteration rate along with the altering of the filter mechanization method. For the FCD task, three techniques for filter implementation were evaluated. These were the bilinear transformation (Tustin's substitution) and two numerical integration methods, rectangular and trapezoidal. Three solution iteration rates were also investigated, giving computational frame times of approximately 6 msec, 20 msec, and 50 msec.

5.2.1 ICPS Second-Order Filter Evaluation

Filter studies with the ICPS consisted of programming a second-order filter and evaluating the filter static and dynamic performance for five different natural frequencies and two different damping ratios. Implementation of the second-order filter followed a standard algorithm from a software catalog of functions provided in the G.E. ICP-723 User's Manual (ref. 4). A detailed derivation of the filter algorithm is presented in the section below, followed by a section dealing with the laboratory test results.

5.2.1.1 ICPS Filter Implementation

The following presents a detailed derivation of an incremental arithmetic equation representing a second-order filter. The basic method involved in this derivation is to rewrite the basic function in a form compatible with the ICPS algorithm equations. The resulting expression requires two algorithm memory locations (and time slots) when mechanized by the ICPS computer.

The second-order filter transfer function is given by the expression:

$$\frac{Z(s)}{V(s)} = \frac{K_1 (T_1^2 S^2 + 2\xi_1 T_1 S + 1)}{K_2 (T_2^2 S^2 + 2\xi_2 T_2 S + 1)} \quad (5-1)$$

where $1/T_2$ is the natural frequency and ξ_2 is the damping factor. Cross multiplying equation (5-1) yields

$$\begin{aligned} K_2 T_2^2 S^2 Z(s) + 2K_2 \xi_2 T_2 S Z(s) + K_2 Z(s) \\ = K_1 T_1^2 S^2 V(s) + 2K_1 \xi_1 T_1 S V(s) + K_1 V(s) \end{aligned} \quad (5-2)$$

Multiplying each term of equation (5-2) by $N/T_1^2 T_2^2 S^2$ gives

$$\begin{aligned} NK_2 \omega_1^2 Z(s) + 2NK_2 \xi_2 \omega_1^2 \omega_2 \frac{Z(s)}{S} + NK_2 \omega_1^2 \omega_2^2 \frac{Z(s)}{S^2} \\ = NK_1 \omega_2^2 V(s) + 2NK_1 \xi_1 \omega_1 \omega_2^2 \frac{V(s)}{S} + NK_1 \omega_1^2 \omega_2^2 \frac{V(s)}{S^2} \end{aligned} \quad (5-3)$$

where $\omega_1 = 1/T_1$ and $\omega_2 = 1/T_2$. Assuming zero initial conditions, equation (5-3) can be written in the time domain as

$$\begin{aligned} NK_2 \omega_1^2 Z(t) + 2NK_2 \xi_2 \omega_1^2 \omega_2 \int_0^t Z(t) dt + NK_2 \omega_1^2 \omega_2^2 \int_0^t \int_0^t g(t) dt \\ = NK_1 \omega_2^2 V(t) + 2NK_1 \xi_1 \omega_1 \omega_2^2 \int_0^t V(t) dt + NK_1 \omega_1^2 \omega_2^2 \int_0^t \int_0^t h(t) dt \end{aligned} \quad (5-4)$$

where

$$g(t) = \int_0^t Z(t)dt \text{ and } h(t) = \int_0^t V(t)dt.$$

Approximating all the integrals with rectangular approximations, equation (5-4) yields at the i th time

$$\begin{aligned} R_i + NK_2\omega_1^2 Z_i + 2NK_2\xi_2\omega_1^2\omega_2^2 \sum_{n=1}^i Z_n \Delta t + NK_2\omega_1^2\omega_2^2 \sum_{n=1}^i g_n \Delta t \\ = NK_1\omega_2^2 V_i + 2NK_1\xi_1\omega_1\omega_2^2 \sum_{n=1}^i V_n \Delta t + NK_1\omega_1^2\omega_2^2 \sum_{n=1}^i h_n \Delta t \end{aligned} \quad (5-5)$$

where R_1 accounts for any truncation error and

$$g_n = \sum_{m=1}^n Z_m \Delta t, \quad h_n = \sum_{m=1}^n V_m \Delta t$$

An equation similar to equation (5-5) can be written to be valid at the $i-1$ st iteration. Subtracting the $i-1$ st equation from the i th equation yields

$$\begin{aligned} R_i + NK_2\omega_1^2 \Delta Z_i + 2K_2\xi_2\omega_1^2\omega_2^2 Z_i + K_2\omega_1^2\omega_2^2 G_i \\ = R_{i-1} + NK_1\omega_2^2 \Delta V_i + 2K_1\xi_1\omega_1\omega_2^2 V_i + K_1\omega_1^2\omega_2^2 H_i \end{aligned} \quad (5-6)$$

as a single equation representing the incremental computation required to generate the i th output increment ΔZ_i . The remainder of this derivation is associated with representing this computation in a form that can be computed in the general algorithm of the ICP-723. See equation (5-13).

Noting that

$$G_i = \sum_{m=1}^i Z_m \Delta t = \Delta Z_i \Delta t + Z_{i-1} \Delta t + \sum_{m=1}^{i-1} Z_m \Delta t$$

equation (5-6) can be written as

$$\begin{aligned}
 & R_i + \Delta Z_i (NK_2\omega_1^2 + K_2\omega_1^2\omega_2^2/N + 2K_2\xi_2\omega_1^2\omega_2) \\
 & + Z_{i-1} (K_2\omega_1^2\omega_2^2/N + 2K_2\xi_2\omega_1^2\omega_2) + K_2\omega_1^2\omega_2^2 \sum_{m=1}^{i-1} Z_m \Delta t \\
 & = R_{i-1} + NK_1\omega_2^2 \Delta V_i + 2K_1\xi_1\omega_1\omega_2^2 V_i + K_1\omega_1^2\omega_2^2 \sum_{m=1}^i V_m \Delta t
 \end{aligned} \tag{5-7}$$

This equation does not involve Z_i in any term. Note that Z_i is not available until after ΔZ_i has been computed.

Defining ρ_i as

$$\rho_i \triangleq \frac{R_i + \Delta Z_i (NK_2\omega_1^2 + K_2\omega_1^2\omega_2^2/N + 2K_2\xi_2\omega_1^2\omega_2)}{(K_2\omega_1^2\omega_2^2/N + 2K_2\xi_2\omega_1^2\omega_2)}$$

equation (5-7) can be expressed as

$$\rho_i = \rho_{i-1} - A\Delta Z_{i-1} - Z_{i-1} - B \sum_{m=1}^{i-1} Z_m \Delta t + C\Delta V_i + DV_i \frac{K_1 B}{K_2} \sum V_m \Delta t \tag{5-8}$$

where

$$\begin{aligned}
 A &= \frac{N + \omega_2^2/N + 2\xi_2\omega_2}{\omega_2^2/N + 2\xi_2\omega_2} & B &= \frac{\omega_2}{\omega_2/N + 2\xi_2} \\
 C &= \frac{NK_1\omega_2}{K_2\omega_1^2\omega_2/N + 2K_2\xi_2\omega_1^2} & D &= \frac{2K_1\xi_1\omega_2}{K_2\omega_1\omega_2/N + 2K_2\xi_2\omega_1}
 \end{aligned}$$

The form of equation (5-8) is not the same as the general algorithm equation shown in equation (5-13) because of the two summations. The presence of these two summations indicates a need to realize this function using two algorithms, equations (5-13) and (5-14).

Now, a new variable Y_i is defined as

$$R_i + \frac{N}{B} Y_i = \frac{ND}{B} V_i + N \left(\frac{K_1}{K_2} \sum_{m=1}^i V_m \Delta t - \sum_{m=1}^{i-1} Z_m \Delta t \right) \tag{5-9}$$

A similar expression may be defined to be valid at the i -1st iteration. Subtracting the expressions on both sides of the i -1st equation from the corresponding expressions in equation (5-9) yields

$$R_i + \frac{N}{B} \Delta Y_i = R_{i-1} + \frac{ND}{B} \Delta V_i + \frac{K_1}{K_2} V_i - Z_{i-1} \quad (5-10)$$

Defining

$$\rho_i = R_i + \frac{N}{B} \Delta Y_i$$

equation (5-10) becomes

$$\rho_i = \rho_{i-1} - \frac{N}{B} \Delta Y_{i-1} + \frac{ND}{B} \Delta V_i + \frac{K_1}{K_2} V_i - Z_{i-1} \quad (5-11)$$

This equation defines a computation that can be performed in one algorithm time to produce an output increment ΔY_i , where Y_i is defined by equation (5-9). The definition of Y_i then may be used to reduce equation (5-8) to a form that can be implemented in one algorithm time. Substituting this value for Y_i into equation (5-8) yields

$$\rho_i = \rho_{i-1} - A \Delta Z_{i-1} - Z_{i-1} + C \Delta V_i + Y_i \quad (5-12)$$

as the defining equation for one of the algorithms, the other being equation (5-11).

The equations computed by the two algorithms in terms of algorithm parameters are

Algorithm 1

$$\rho_i = \rho_{i-1} + S_{p1} \Delta V_i + \Delta T_1 V_i - \Delta W_1 Z_{i-1} - S_{q1} \Delta Y_{i-1} \quad (5-13)$$

Algorithm 2

$$\rho_i = \rho_{i-1} + S_{p2} \Delta V_i + \Delta T_2 Y_i - \Delta W_2 Z_{i-1} - S_{q2} \Delta Z_{i-1} \quad (5-14)$$

Comparing equations (5-13) and (5-14) with equations (5-11) and (5-12) requires gain factors of

$$\begin{aligned} S_{p1} &= \frac{ND}{B}, \Delta T_1 = \frac{K_1}{K_2}, \Delta W_1 = 1, S_{q1} = \frac{N}{B} \\ S_{p2} &= C, \Delta T_2 = 1, \Delta W_2 = 1, S_{q2} = A \end{aligned} \quad (5-15)$$

or

$$S_{p1} = 2NK_1\xi_1C_z/K_2\omega_1C_v = 2NK_1\xi_1T_1C_z/K_2C_v$$

$$S_{q1} = 1 + 2N\xi_2/\omega_2 = 1 + 2N\xi_2T_2$$

$$S_{p2} = \frac{N^2K_1\omega_2C_z}{K_2\omega_1^2C_v(\omega_2 + 2N\xi_2)} = \frac{N^2K_1T_1^2C_z}{K_2C_v(1 + 2N\xi_2T_2)}$$

$$S_{q2} = \frac{N^2 + 2N\xi_2\omega_2 + \omega_2^2}{\omega_2^2 + 2N\xi_2\omega_2} = \frac{N^2T_2^2 + 2N\xi_2T_2 + 1}{1 + 2N\xi_2T_2}$$

$$\Delta W_1 = \Delta T_2 = \Delta W_2 = 1 \quad \Delta T_1 = \frac{K_1C_z}{K_2C_v}$$

where C_v and C_z are the input and output scale factors in machine units per variable.

The algorithm for the second-order lead-lag is shown in figure 3-5. Alternate representations are possible through a different choice of Y_i (equation (5-9)), which would yield a different equation (5-12). With the algorithm connection shown in figure 3-5, the overall implementation is given by

$$\begin{aligned} & (S_{q2} - \Delta W_2) Z_i S^2 + \left(N\Delta W_2 - \frac{N\Delta T_2\Delta W_1}{S_{q1}} \right) Z_i S + \left(\frac{N^2\Delta T_2\Delta W_1}{S_{q1}} \right) Z_i \\ & = (S_{p2}) V_i S^2 + \left(\frac{N\Delta T_2 S_{p1}}{S_{q1}} \right) V_i S + \left(\frac{N^2\Delta T_1\Delta T_2}{S_{q1}} \right) V_i \end{aligned}$$

Equating equations (5-15) and (5-2) requires that

$$S_{q2} - \Delta W_2 = \frac{K_2T_2^2}{C_z}, \quad N\Delta W_2 - \frac{N\Delta T_2\Delta W_1}{S_{q1}} = \frac{2K_2\xi_2T_2}{C_z}$$

$$\frac{N^2\Delta T_2\Delta W_1}{S_{q1}} = \frac{K_2}{C_z}, \quad S_{p2} = \frac{K_1T_1^2}{C_v}$$

$$\frac{N\Delta T_2 S_{p1}}{S_{q1}} = \frac{2K_1\xi_1T_1}{C_v}, \quad \frac{N^2\Delta T_1\Delta T_2}{S_{q1}} = \frac{K_1}{C_v}$$

Thus, any gain factor choice that satisfies the above is acceptable; this selection is just one possible set.

5.2.1.2 ICPS Filter Laboratory Test Results

Ten second-order filters were mechanized using the ICPS, having five natural frequencies ($\omega_n = 1, 5, 10, 50, \text{ and } 100$) with two different damping ratios ($\xi = 0.3$ and 1.0). Both frequency response and static (linearity/resolution) tests were conducted on each filter to assess the ICPS filter processing performance relative to continuous systems theoretical predictions. The ICPS has a fixed solution rate of 163 times per second; therefore, the only parameter changes made were the filter parameters given above. The frequency response tests were made with an input driving signal held to 4% of the maximum input allowable to avoid the slew rate limit characteristic of the ICPS computer (discussed in sec. 4.1.1). Table 5-3 presents the ICPS second-order filter algorithm coefficients used in the tests.

TABLE 5-3.—ALGORITHM COEFFICIENTS FOR ICPS SECOND ORDER FILTER STUDY

Damping ratio $\xi = 0.3$

Algorithm coefficients \ Natural frequency (ω)	1	5	10	50	100
Sq_1	653	529	531	481	272
Sq_2	164	136	142	154	104
ΔT_1 and ΔT_2	2	8	16	64	64
ΔW_1 and ΔW_2	2	8	16	64	64

$$Sp_1 = Sp_2 = 0$$

Damping ratio $\xi = 1.0$

Algorithm coefficients \ Natural frequency (ω)	1	5	10	50	100
Sq_1	197	164	172	189	126
Sq_2	539	421	410	294	150
ΔT_1 and ΔT_2	2	8	16	64	64
ΔW_1 and ΔW_2	2	8	16	64	64

$$Sp_1 = Sp_2 = 0$$

Figures 5-18 and 5-19 show the frequency response results of the laboratory tests. These plots indicate that the ICPS matches well the continuous domain predicted dynamic characteristics for filters having natural frequencies of 10 rad/sec or below. The errors at the higher frequencies are attributed to the ICPS prefilter and slew rate limit characteristics.

The slew rate effect is normally encountered at the higher frequencies but can be manifested when the amplitude response of a filter increases as a result of the filter's low damping ratio. For this case, the frequency response would exhibit a clipping of the output amplitude signal. Figure 5-20 illustrates slew rate limit effects for a second-order filter with a natural frequency of 10 rad and damping ratio of 1. The effect is shown as the input amplitude signal is increased from the 4% level (0.4 V) to an 80% level.

The prefilter effect is associated with the double breakpoint filter placed on the analog input signal conditioning amplifier to suppress aliasing of the input signal due to discrete sampling (discussed in sec. 4.1). These filters for the ICPS have break frequencies at approximately 100 rad. This effect is related to the basic frequency response of the ICPS; therefore, the basic ICPS hardware response has been superimposed on the frequency response plots of figures 5-18 and 5-19.

Static tests did not produce discernible errors in repeatability or linearity. Filter performance limitations for the ICPS are related only to the prefilter and slew rate limit characteristics. The prefilter can be modified to reduce its effect in the frequency band of interest where the slew rate limit is a fixed characteristic of the hardware. Thus, the slew rate limit is considered the primary constraint in high-frequency filter implementations.

5.2.2 WWCS Digital Filter Implementation Study

Various techniques are available for programming filters to be processed by the WWCS computer. A literature survey was conducted to determine if there existed a universal best method. Unanimity could not be found with those in industry involved with airborne digital computers; therefore, three methods were somewhat arbitrarily chosen for investigation within the FCD task. These were the bilinear transformation (Tustin's substitution), rectangular numerical integration, and trapezoidal numerical integration. These methods were evaluated in terms of dynamic/static performance, memory requirements, and real-time processing requirements.

The following describes the mathematical background to the programming methods selected, followed by a discussion of laboratory tests conducted to compare the performance of the three methods. From the three techniques evaluated, the bilinear transformation method was selected to be used for the WWCS general filter study and section 5.2.3 presents the study results.

5.2.2.1 Bilinear Transformation

The general second-order filter in the continuous (S) domain can be represented as:

$$\frac{Y(s)}{X(s)} = \frac{K_n \left(\frac{1}{\omega_n^2} S^2 + 2\zeta_n \frac{1}{\omega_n} S + 1 \right)}{K_D \left(\frac{1}{\omega_D^2} S^2 + 2\zeta_n \frac{1}{\omega_n} S + 1 \right)} \quad (5-16)$$

where

- ω = natural frequency
- ζ = damping ratio
- K = steady-state gain
- X = input function (signal)
- Y = output signal

The following is a simplification of equation (5-16) to minimize terms and provide a somewhat more general expression.

$$\frac{Y(s)}{X(s)} = \frac{a_0 + a_1 S + a_2 S^2}{b_0 + b_1 S + b_2 S^2} \quad (5-17)$$

where

$$\begin{aligned} a_0 &= K_N & b_0 &= K_D \\ a_1 &= 2K_N \zeta_n \frac{1}{\omega_n} & b_1 &= 2K_D \zeta_D \frac{1}{\omega_D} \\ a_2 &= K_N \frac{1}{\omega_n^2} & b_2 &= K_D \frac{1}{\omega_D^2} \end{aligned}$$

This equation is then used to make the bilinear transformation by substituting

$$\frac{2}{T} \left(\frac{1 - Z^{-1}}{1 + Z^{-1}} \right)$$

for the operator S, where $Z^{-1} = e^{-ST}$ and T = sampling period. This substitution (Tustin's method) is the representation of an integral expression using trapezoidal integration.

The substitution yields:

$$\begin{aligned} \frac{Y(Z^{-1})}{X(Z^{-1})} &= \frac{a_0 + a_1 \left[\frac{2}{T} \left(\frac{1-Z^{-1}}{1+Z^{-1}} \right) \right] + a_2 \left[\frac{2}{T} \left(\frac{1-Z^{-1}}{1+Z^{-1}} \right) \right]^2}{b_0 + b_1 \left[\frac{2}{T} \left(\frac{1-Z^{-1}}{1+Z^{-1}} \right) \right] + b_2 \left[\frac{2}{T} \left(\frac{1-Z^{-1}}{1+Z^{-1}} \right) \right]^2} \\ &= \frac{A_0 + A_1 Z^{-1} + A_2 Z^{-2}}{1 + B_1 Z^{-1} + B_2 Z^{-2}} \end{aligned} \quad (5-18)$$

where

$$A_0 = \frac{a_0 T^2 + 2a_1 T + 4a_2}{b_0 T^2 + 2b_1 T + 4b_2}$$

$$A_1 = \frac{2a_0 T^2 - 8a_2}{b_0 T^2 + 2b_1 T + 4b_2}$$

$$A_2 = \frac{a_0 T^2 - 2a_1 T - 4a_2}{b_0 T^2 + 2b_1 T + 4b_2}$$

$$B_1 = \frac{2b_0 T^2 - 8b_2}{b_0 T^2 + 2b_1 T + 4b_2}$$

$$B_2 = \frac{b_0 T^2 - 2b_1 T - 4b_2}{b_0 T^2 + 2b_1 T + 4b_2}$$

Cross-multiplying equation (5-18), solving for the filter output Y_N , and bearing in mind that Z^{-1} simply means a one-frame time delay, we have:

$$Y_N = A_0 X_N + A_1 X_{N-1} + A_2 X_{N-2} - B_1 Y_{N-1} - B_2 Y_{N-2}$$

where N = current value (sample).

Figure 5-21 is a block diagram representation of the Y_N equation. Because of the restrictions of fixed-point arithmetic, scale factors 2^{-K_1} and K_2 are included in the diagram to provide the structure for implementing the function in the WWCS.

Unfortunately, the basic bilinear transformation produced large steady-state errors, attributable to the processed solution truncation error caused by the computer's finite word length. This error was overcome by adding truncation compensation to the basic bilinear transformation block diagram. Figure 5-22 shows the basic bilinear method of figure 5-21 with the compensation required to achieve acceptable steady-state performance (test data are presented for the uncompensated case in a following laboratory test discussion). The compensation scheme used first accumulates the input signal/coefficient products at double precision; then the $N-1$ arithmetic residual value is added to that sum. This result is then partitioned into two parts, the most significant bits (MSBS) and the least significant bits (LSBS). The MSBS form the single-precision result that is used as the filter output. The LSBS are saved within the filter computational process and used as the arithmetic residual value for the $N+1$ computations.

5.2.2.2 Numerical Integration Methods

Numerical integration involves directly replacing the continuous domain integrator ($1/S$) with a numerical approximation function. Figure 5-23 presents a block diagram of the continuous domain second-order filter. For the rectangular integration method, the $1/S$ function is replaced by $T/(1 - Z^{-1})$, where $Z^{-1} = e^{-ST}$ and T is the sampling period. The block diagram for a rectangular integration second-order filter is shown in figure 5-24.

For the trapezoidal numerical integration method, the $1/S$ term is replaced by

$$\frac{T}{2} \left(\frac{1 + Z^{-1}}{1 - Z^{-1}} \right)$$

The trapezoidal integration second-order filter block diagram is presented in figure 5-25. It should be noted that, unlike Tustin's method, no attempt is made to solve the Z domain equation to eliminate the transport delay in the feedback paths of the B_0 and B_1 coefficients.

5.2.2.3 Digital Filter Implementation Laboratory Study

A laboratory comparison was made (prior to receipt of the WWCS) of the three filter mechanization techniques described above using a small Boeing-built general-purpose digital computer (18-bit fixed point). The approach taken was to implement the HSAS filter set using the rectangular, trapezoidal, and bilinear methods and determine the variations in: (1) timing and core (memory) utilization, (2) time response to step input, (3) frequency response, and (4) steady-state resolution. The study included tests for frame times of 20 and 50 msec and 18- as well as 16-bit word lengths (WWCS was to have basic 16-bit word length). The HSAS filters used in the study are shown in figure 5-26.

Timing and memory utilizations are tabulated below for the HSAS filters implemented on the Boeing computer using the three methods discussed.

<u>Method</u>	<u>Timing</u> <u>(% of base)</u>	<u>Memory</u> <u>(% of base)</u>
Bilinear	Base	Base
Rectangular	164	130
Trapezoidal	196	143

These numbers represent the basic bilinear implementation, which does not include the compensation function. From this comparison, the bilinear method appears the most attractive.

Filter time history responses for the three methods showed only subtle differences, and no conclusions could be drawn from these tests.

Frequency response comparisons were made by plotting the gain and phase difference between the three filter implementation techniques and the theoretical predicted continuous response. Figures 5-27 and 5-28 show this gain and phase difference, respectively, for the complete HSAS filter string (filter A through filter D, fig. 5-26). Figures 5-29 and 5-30 show the same relationships for just the HSAS filter A. Figures 5-31 and 5-32 repeat the comparison for filter B of the HSAS filter set. These responses clearly show that the bilinear transformation method has the least amount of error in gain and phase to the theoretical values. The responses also show that the gain and phase for all cases decrease as the sampling rate is increased (50-msec frame time to 20-msec frame time). It is important to note that these data were taken for a 16-bit word length without the use of double-precision accumulation. No difference was discernible for an 18-bit word length.

Steady-state resolution data were generated by updating the input value in a software controlled manner. After the input was changed, the filter output was tabulated following 1000 solution iterations. The input values were modified as if there had been a one-bit change in the least significant bit of a 10-bit analog-to-digital converter (the A/D word length on the Boeing computer). The results were tabulated with respect to a 12-bit digital-to-analog converter. This method permitted data to be taken without the influence of offsets found in the analog interface and eliminated a one-bit noise factor associated with the A/D converter.

Steady-state response plots for the HSAS filter A using the three programming methods are shown in figure 5-33. These plots are for a 20-msec frame time with a 16-bit word length. The results are similar for the three methods with the bilinear method giving slightly better resolution and the trapezoidal method giving the worst resolution. The filter A steady-state response was tested for an increase in both word length (18 bits) and frame time (50 msec). In general, the steady-state offsets were reduced when either the word length or frame time increased. It was of interest to note that the 18-bit, 20-msec case showed the rectangular method to have the best performance and the bilinear method to have the worst.

Resolution plots for filter B of the HSAS filter set are shown in figure 5-34, again for a 20-msec frame time and a 16-bit word length. The plots show the rectangular method has the best performance and the bilinear method the worst, to a degree that was considered unacceptable. The filter B resolution improved for all cases when the frame time or word length was increased. However, the bilinear method results remained the worst and were considered outside acceptable limits.

This laboratory study using the Boeing computer revealed that the bilinear method was by far the best of the three in computing time, memory utilization, and dynamic (frequency response) performance. However, for the tests conducted, it exhibited unacceptable steady-state characteristics. It was these results that led to the development of a compensation scheme to improve the bilinear method steady-state performance for low-frequency filter implementations (the natural frequency of filter B was approximately 1.8 rad where the natural frequency of filter A was 6 rad). A study conducted to identify the cause of the bilinear method poor steady-state performance is summarized in appendix B.

5.2.3 WWCS Second-Order Filter Evaluation Using Bilinear Transformation

The compensated bilinear transformation method (refer to fig. 5-22) was used to mechanize second-order filters in the WWCS computer to conduct a general study of the input-to-output WWCS filter implementation performance. The study covered 10 second-order filters having five different natural frequencies and two different damping ratios. In addition, the filter set was evaluated for three computational frame times.

It was found that the calculation of the discrete (Z) domain filter coefficients was a significant source of error. Therefore, section 5.2.3.1 discusses these calculations and their relative impact on the implemented filter performance.

The filter frequency response test results are discussed in section 5.2.3.2. Performance errors to the continuous domain theoretical predicted values could be associated with:

- 1) The coefficient calculation errors
- 2) A frequency-warping phenomenon related to digital filter implementations
- 3) A/D and D/A signal conditioning and time delays

Accounting for all of these errors, reasonable correlation between the theoretical response and the actual response was obtained. The WWCS was found to be capable of representing second-order filters with good fidelity for natural frequencies up to 10 rad with frame times from 6 to 50 msec. Good filter response fidelity could be achieved for a natural frequency of 100 rad if the sampling (frame) time was ≤ 6 msec.

Steady-state resolution was examined for second-order filters using the HSAS filters A and B. The compensated bilinear method provided excellent results as described in section 5.2.3.3.

5.2.3.1 Filter Coefficient Calculation

Section 5.2.2.1 presented a derivation of the second-order filter bilinear transformation coefficients. From equation (5-18) of that section, it can be seen that the coefficients are not simple. Furthermore, there does not exist an easy correlation between the parameters of a second-order filter (gain, natural frequency, and damping ratio) and the bilinear equation coefficients. Since these parameters were to be altered for the study, a routine was developed within the WWCS computer to translate the second-order filter parameter changes into appropriate bilinear coefficient values.

The Z domain coefficients (refer to equation (5-18) of sec. 5.2.2.1) have the following bounds for second-order parameters of ω from 1 to 100 and ζ from 0.3 to 1.0, with frame times of 6.144 msec and 49.152 msec.

9×10^{-6}	A_2	0.42
1.8×10^{-5}	A_1	0.84
9×10^{-6}	A_0	0.42
-0.994	B_2	-0.001
-0.73	B_1	1.994

Coefficients are enterable into the WWCS computer with the following restriction: $0.0000305175 \leq |\text{Coefficient}| \leq 0.9999694824$. This restriction conflicts with the bounds given by the above A and B coefficients. In order to satisfy the upper bound, the coefficients were scaled within the WWCS coefficient calculating routine by 0.5. This scaling aggravated the lower bound because the smallest nonzero coefficient value became equal to $0.000061035 (2^{-14})$.

The preceding discussion brings out the difficulty in providing coefficient value accuracy for a bilinear transformation second-order filter algorithm when the filter frequency response values cover such a wide range. There are ways in which the significant digits of the small coefficients can be improved. Some of the most obvious are to enlarge the frame time, use a longer word length computer, provide software or hardware floating-point processing, or internally scale within the filter computations. This latter approach involves scaling the small coefficients up by 2^n and then scaling their collected sum down by the same value. This would compromise the desire to adapt a universal filter algorithm where the continuous domain filter parameters can be changed without having to rescale or modify the frame time period.

Table 5-4 shows the coefficient values used in the WWCS filter implementation study. Coefficient scaling was used, and the table includes the scaled values. For example, A_1 was scaled up by 2^{14} (a decimal factor of 16384) for $\omega_D = 1.0$ rad/sec, $T = 6.144$ msec, and $\zeta_D = 1.0$. This illustrates how large the scale factor can become for a relative straightforward filter case.

It should be noted that the B coefficient signs were changed from those obtained from the coefficient equation, so that the filter routine sums all product values rather than subtracting the B terms.

TABLE 5-4.—SECOND ORDER FILTER BILINEAR TRANSFORMATION COEFFICIENTS
(REFERENCE SECTION 5.2.2.1, EQUATION 5.2.2-3)

Filter case	Natural frequency (ω_D , rad/sec)	Frame time (T, sec)	Damping ratio (ξ_D)	Bilinear coefficients (hexadecimal notation)				A_1 scale factor (reference fig. 5-22)
				A2 = A0	A1	B2	B1	
1	1	0.006144	0.3	013C	027E	C03C	7FC3	2-13
2	1	0.018432	0.3	2640	4C80	C0B5	7F46	2-12
3	1	0.049152	0.3	0009	0013	C1DC	7DFD	1.0
4	1	0.006144	1.0	13AC	2758	C0C8	7F37	2-14
5	1	0.018432	1.0	0577	0AEE	C252	7DA9	2-10
6	1	0.049152	1.0	0009	0012	C5FE	79DC	1.0
7	5	0.006144	0.3	0003	0006	C12A	7EC9	1.0
8	5	0.018432	0.3	0021	0042	C36C	7C0F	1.0
9	5	0.049152	0.3	00DF	01BF	CA86	73DC	1.0
10	5	0.006144	1.0	0003	0006	C3CD	7C28	1.0
11	5	0.018432	1.0	001F	003E	CAC3	74BF	1.0
12	5	0.049152	1.0	00C1	01A2	D8EB	640F	1.0
13	10	0.006144	0.3	000A	0014	C1EF	7DE7	1.0
14	10	0.018432	0.3	006D	00DA	C599	78R4	1.0
15	10	0.049152	0.3	02B7	056F	CD69	67BF	1.0
16	10	0.006144	1.0	0009	0013	C63B	79A0	1.0
17	10	0.018432	1.0	0063	00C6	D0F7	6D7D	1.0
18	10	0.049152	1.0	022E	045D	E3FA	534C	1.0
19	50	0.006144	0.3	011F	023E	C9E8	7285	1.0
20	50	0.018432	0.3	0924	1249	D7A8	4427	1.0
21	50	0.049152	0.3	1DB6	386D	DD2B	EC30	1.0
22	50	0.006144	1.0	00ED	01DA	DDC2	5E9B	1.0
23	50	0.018432	1.0	0657	06AF	F77F	2F47	1.0
24	50	0.049152	1.0	1359	26B2	FFAB	F319	1.0
25	100	0.006144	0.3	0972	12E4	CB34	6665	1.0
26	100	0.018432	0.3	1924	3248	D44A	0924	1.0
27	100	0.049152	0.3	2FFF	5FFF	D001	8000	1.0
28	100	0.006144	1.0	02C8	0590	F3CA	42C7	1.0
29	100	0.018432	1.0	0D89	1B13	05F3	04EC	1.0
30	100	0.049152	1.0	1DE4	3BC8	FA98	CE2E	1.0

5.2.3.2 WWCS Second-Order Filter Frequency Response Data

Frequency response data were taken for the filter cases defined in table 5-4 using a frequency analyzer across the WWCS A/D and D/A interface. As previously stated, the WWCS-implemented filter frequency response is influenced by the basic A/D and D/A hardware processes, Z domain coefficient accuracy, and a frequency-warping phenomenon. Since the coefficient inaccuracies would affect the lower frequency filter response results and the hardware characteristics discussed in section 4.1 would affect the higher frequency response results, the frequency response discussion was separated to treat the low-frequency cases ($\omega_D = 1, 5, \text{ and } 10$) first, followed by a treatment of the high-frequency filter cases ($\omega_D = 50 \text{ and } 100$).

Figure 5-35 shows the frequency response of the second-order filters having an $\omega = 1$, 5, and 10 and a damping ratio of $\zeta = 0.3$, for a sampling period (frame time) of 49 msec. With the WWCS routine calculated Z domain coefficients, the frequency response of the $\omega_D = 1$ case deviated substantially from the desired value. This case was rerun using hand-calculated scaled-up coefficients and the response closely resembled the ideal, as shown in figure 5-35. The frequency responses for the $\omega_D = 5$ and 10 cases, with the WWCS-calculated coefficients, closely resembled the theoretical response. Phase errors of these latter cases are attributable to the basic hardware characteristics.

Figures 5-36 and 5-37 show the above filter cases with sampling periods of 18 and 6 msec, respectively. As before, the $\omega_D = 1$ response for the 18-msec frame time required scaled-up coefficients to approach the theoretical ideal. However, the $\omega_D = 1$ case for the 6-msec frame time could not be obtained, even when maximum scaled-up coefficients were used. This implies that a 16-bit word length is not sufficient to achieve the scaled values required to realize the filter. The ω_D cases of 5 and 10 follow the results of the 49-msec sampling period cases.

Figures 5-38 through 5-40 show the above filter responses with a damping ratio of one ($\zeta = 1$). Here again hand-calculated coefficients were used to acquire the desired results for filters with the lowest breakpoint frequencies. As before, the $\omega_D = 1$ case for a sampling period of 6 msec could not be obtained.

As the bandpass (break frequencies) of the second-order filter are moved above 10 rad/sec, the frequency response not only suffers more from the basic hardware characteristics, but begins to be affected by a frequency-warping phenomenon. This frequency warping is a shift of the apparent continuous domain (S domain) natural frequency when the discrete domain (Z domain) transformation is made. The warping effect can be determined by setting $S = j\omega$ in the Z domain transformation. For the bilinear transformation

$$S = \frac{2}{T} \left[\frac{1 - Z^{-1}}{1 + Z^{-1}} \right]$$

where

$$Z^{-1} = e^{-ST}$$

$$T = \text{sampling period}$$

the substitution yields

$$\begin{aligned} j\omega_S &= \frac{2}{T} \left[\frac{1 - e^{-j\omega_Z T}}{1 + e^{-j\omega_Z T}} \right] \\ &= \frac{2}{T} \left[\frac{j \sin \omega_Z T}{1 + \cos \omega_Z T} \right] \\ &= j \frac{2}{T} \tan \frac{\omega_Z T}{2} \end{aligned}$$

or,

$$\omega_S \frac{T}{2} = \tan \frac{\omega_Z T}{2}$$

Solving for ω_Z yields:

$$\omega_Z = \frac{2}{T} \tan^{-1} \frac{\omega_S T}{2}$$

The relationship between ω_Z and ω_S is plotted in figure 5-41 for the three sampling periods used in the filter implementation study.

In order to more accurately compare the theoretical response with the actual response of the high-frequency filter cases, the data were prepared showing:

- 1) Theoretical response of the second-order filter based on the ω and ξ calculated from the Z domain coefficients used
- 2) Frequency-warping response, theoretical response modified by the warping effect discussed above
- 3) WWCS filter response as recorded by the frequency analyzer
- 4) The WWCS filter response after removing the effect of the basic hardware frequency response

Figure 5-42 shows the responses of the $\omega_D = 50$, $\zeta = 0.3$ filter cases for the three different sampling periods. The frequency warping can be observed to drop the filter break frequency substantially as the sampling period is increased (sampling rate decreased). In all cases, the filter alone phase lag is slightly greater than the warped frequency response predictions. The same data were produced for a damping ratio $\zeta = 1.0$ and are shown in figure 5-43. Again the warping effect and hardware response significantly alter the implemental filter performance at the long sampling periods, with the warping effect becoming insignificant when the sampling period becomes 6 msec.

For the filter cases having natural frequency of $\omega_D = 100$, the $\zeta = 0.03$ results are shown in figure 5-44 and the $\zeta = 1.0$ results are shown in figure 5-45. As expected, the warping effect and hardware characteristics significantly alter the implemented second-order filter response, especially for the long sampling period. The filter gain peaking shift is substantial for the low damping ratio low sampling rate cases. The high damping ratio responses exhibited the same trends found for the filter natural frequency of 50 rad/sec.

These results provide an insight into the continuous domain filter characteristic changes that can occur when the filter is simply transformed into the Z domain, programmed, and processed in a digital computer. Many digital system parameters influence the programmed filter's response; therefore, great care should be taken to understand these effects relative to the flight-critical system's overall dynamic performance requirements.

5.2.3.3 WWCS Filter Implementation Steady-State Response

The steady-state WWCS input/output characteristics for the bilinear transformation filter implementation were evaluated using the HSAS filter A and filter B functions (refer to sec. 5.2.2.3). With the arithmetic residue compensation, the output of filter A precisely followed the input (fig. 5-46). The filter B output followed the input within one machine unit, where the output tended to oscillate one machine unit for a given input with an average period of 1.5 min (fig. 5-47). Thus, the compensation method used resolved the unacceptable performance observed with the basic bilinear transformation during the Boeing computer tests of section 5.2.2.

Data were taken on the HSAS filter A with the residue compensation path removed. As seen in figures 5-48 and 5-49, the output developed a negative bias but followed a more consistent pattern than the one obtained during the Boeing computer study. The performance differences between the two tests are attributable to the use of double-precision product accumulation in the WWCS.

5.2.4 Digital Filter Study Conclusions

The preceding filter implementation studies indicate that both the ICPS and WWCS can be used to process, in a straightforward manner, filter functions with bandpass (breakpoint) frequencies from 1 to approximately 30 rad/sec. Filter functions outside this region will require special treatment to properly account for the basic hardware and/or software (discrete domain programming) characteristics. Analog prefilter values can be selected to change what is here referred to as an element of the basic hardware characteristic, and software implementation techniques (along with careful scaling) can be employed to refine

the digital system's filter processing performance. However, certain hardware limitations, such as the ICPS slew rate limit or the WWCS 16-bit word length, present absolute boundaries for the realization of some filter functions.

While the bilinear transformation method (Tustin's substitution) was used to implement the filter functions in the WWCS, it is difficult to recommend this or either of the other methods evaluated. The advantages found with the bilinear approach (speed, memory utilization, dynamic fidelity, etc.) may lose their weight for some applications. For instance, in conventional flight control systems, filter functions are a small part of the total flight control function and the speed/memory advantage of the bilinear method may be inconsequential to the overall time and memory requirements. Rate-limiting functions are usual items in automatic flight control systems; and the numerical integration methods are more suited for implementing such a function within the filter computations. Another basic consideration is the ease by which an engineer can associate numerical integration coefficients with the filter parameters (gain, time constant, and damping ratio), as compared to the bilinear coefficient complexity. This consideration may be very important in a flight test program when such parameters are being changed to refine the vehicle's dynamic performance from a continuous domain analysis point of view. All methods will require careful analysis and laboratory testing to gain assurance that the desired function has indeed been realized after the hardware system has been essentially fully integrated.

5.3 DIGITAL SYSTEM COMPUTATIONAL OVERFLOW

Computational overflow occurs in a digital computer whenever "the generation of a number within the computer extends outside the representation range capability of the computer." All digital computing processors have clearly defined ranges of variable representation, with the range established when the computer word length/computation scheme is selected. With the digital computer application to flight controls, computational overflow presents a potential operational hazard in that a full-scale positive variable can become a full-scale negative value, or vice versa, as an increment is added to full-scale value. This could result in a large surface command transient or full authority in one direction to full authority in the other direction, or create a sign reversal in the control path, which would have a destabilizing effect.

Computational overflow has been normally avoided (eliminated) through proper scaling as the software is checked out. In a redundant flight-critical system, however, cause for concern exists when the question is asked, "Can scaling alone assure that the system/software will not incur an overflow for any and all situations that may arise in flight?" The initial reaction is to consider that while an overflow may occur in one computer (channel), the other redundant channels will continue to safely control the airplane. The fallacy in this line of reasoning lies in the fact that all computers should be performing like, or the same, control computations and if one processor incurs a computational overflow, the others in all probability will also incur a computational overflow. Overflow situations resulting from hardware failures would be "voted out" in a redundant system, but overflow cases allowable within the software would generally result in a nearly simultaneous overflow within all redundant processors. Scaling of the variables to prevent overflow must be accomplished considering worst case in-flight conditions. One approach is to define the worst case to be where all inputs are at their maximum value and

the signs of the signal paths at each summing node are additive. This would cover the situation envisioned where the airplane becomes upset by the environment and the pilot's commands are in the same direction as the stabilizing sensors. This requires a detailed analysis from one summing node to another to define the steady-state and/or dynamic state worst case conditions that would establish the variable maximum scaling at each summing node. The analysis would have to be repeated each time the control functions were modified during the system development phase. This approach could result in conflicts between scale factor values and system resolution requirements.

The above worst case scaling approach was used for the ICPS and WWCS HSAS implementation, discussed in sections 3.2 and 3.3, respectively. For the HSAS control function, steady-state worst case analysis and scaling appeared to be adequate. However, the HSAS function is a fairly simple control law in that it utilizes very few sensor variables and has fixed signal path gains. It would be a very difficult task to provide this form of analysis on a multivariable control system having a wide range of flight condition dependent gain scheduling.

Analog systems have the inherent characteristic that when a variable exceeds its scaled maximum value, the analog device whose output represents the variable (or a partial sum of several variables) will saturate, holding the maximum value until the upstream signals no longer force the variable to, or beyond, its maximum value. This variable saturation characteristic (signal limit) considerably reduces the scaling task, and scaling/resolution conflicts can generally be easily resolved.

When such limits are encountered during an airplane maneuver, the airplane's response may change, but all command and/or feedback signals will retain their relative signs, preserving the proper stability conventions. The airplane's safety is not generally threatened, and system parameter values can be changed by relative large percentages without requiring a complete rescaling analysis of the entire control law processing.

With the above overview of analog and digital systems' signal saturation behavior and scaling requirements and the concern arising with the digital systems overflow characteristics, a study was conducted to determine the feasibility of providing overflow protection in both the ICPS and WWCS. The study was directed toward defining an overflow protection scheme which, as a basic functional element of the ICPS or WWCS, would produce a saturated signal effect similar to that found with the analog system. The following sections present the study results: an overflow protection scheme proposed for the ICPS and derived from a paper analysis, and an overflow protection scheme for the WWCS, implemented and tested in the laboratory, utilizing both hardware and software.

5.3.1 ICPS Computational Overflow Protection

Computational overflow in the ICPS incremental computer can only occur with respect to five whole-word machine variables (U, V, X, Y, and ρ). These machine variables are provided by recirculating shift registers that make a unique system variable available each algorithm time slot (ref. 1, sec. 5.3.1). The U, V, X, and Y register word size is 16 bits and the ρ register word size (data portion) is 35 bits. A two's complement integer binary

representation is used for each of these variables. Thus, the numeric range boundaries for these variables are:

$$-2^{15} \leq U, V, X, \text{ or } Y \leq 2^{15} - 1$$

$$-2^{34} \leq \rho \leq 2^{34} - 1$$

where $2^{15} = 32,768$ and $2^{34} = 17,179,869,184$ machine units. Overflow occurs with any one of these variables whenever the next computer value of the variable exceeds the above machine unit range. Figure 5-50 illustrates the overflow characteristic by comparing the actual stored result with the ideal computed result.

The obvious overflow protection solution would be to incorporate a saturating limiter around each whole-word register in the ICPS computer. Unfortunately, this concept, which is quite simply utilized in a continuous analog system, cannot be utilized in the incremental computer with the same results as those found in the analog case. In order to understand the reasons for this, it is necessary to carefully consider the nature of a saturating limiter in the continuous sense and compare this to the nature of the whole-word variable representation in the incremental computer.

First, a saturating limiter in a continuous system is generally based upon a gain-changing process. This is illustrated in figure 5-51 for a simple input/output device that has a fixed gain (K_1) until the input reaches a preselected value ($\pm L$) where the gain is changed to a new value (K_2). The gain K_2 may in general be much less than K_1 ; however, the nearly infinite resolution of the continuous system maintains some proportionality. (No practical continuous system ever achieves a value of $K_2 = \text{zero}$.) The important characteristic of this type of limiter is that no hysteresis exists. The input/output relationship always remains uniquely proportional. Thus, the process of going into and out of the limit is completely and repetitively reversible.

Now, let us consider the nature of the whole-word variables in the ICPS for which we would like to provide an overflow limit function. For the purpose of this discussion, U, V, X, and Y may be considered as one case: the 16-bit case. The variable X will be used in all subsequent discussions to indicate the 16-bit case.

The present value of the variable X on any given iteration, i , is given by the following:

$$X_i = \sum_{j=1}^i \Delta X_j = X_{i-1} + \Delta X_i$$

If an arbitrary gain constant K is added to this expression so that

$$\bar{X}_i = KX_i = \sum_{j=1}^i K\Delta X_j$$

the ICPS X register drawing in figure 5-52 can be used as reference in discussing the limiting problem.

In the present incremental computer circuitry, the value of the gain constant K is unity. If a new circuit was added to detect the last value \bar{X}_{i-1} and the gain constant K was changed whenever \bar{X}_{i-1} became "close" to the full-scale limit, then it would appear that the X register might operate as if it had a limiter around it. By making K small enough, it would certainly appear possible to slow down the rate of approach to overflow and potentially eliminate overflow. The K value required would be less than 1, so that for integer value variables a division process is implied.

This approach does not work because there is not adequate resolution available on the variable value $\Delta X_i/K'$, where $K' = 1/K$ (refer to fig. 5-52). The lack of resolution will cause the limiter to "hang up" in the limit, either permanently, or, depending upon the value of K' and the signal significant lost, until a large decreasing direction increment occurs. With the increment size for ΔX_i restricted to powers of two (up to $\pm 2^6$), and the fact that the mechanization for division would require K' to be a power of two, it is obvious that the resolution available is inadequate. Furthermore, even if adequate resolution were available, a finite value of K' would mean that an eventual overflow would occur if ΔX_i were a constant, regardless of the word length ultimately used for the register.

A scheme for the ICPS that would give a practical saturation effect on the variable registers would be to detect the occurrence of overflow and substitute a full-scale value for that register's output while still maintaining the entire history of the register. The substituted maximum value would be used in the processor computations and the register would be permitted to move through the discontinuity shown in figure 5-50. This approach would not break down until the register value exceeded $3 \times 32,768$, or 98,304, machine units, a value sufficiently greater than the maximum value which would protect against overflow situations that would be brought on by scale factor discrepancies resulting from minor control parameter changes. It would also permit some flexibility to resolve scaling/resolution incompatibilities.

A hard limit scheme for the ICPS, free from any potential overflow, could be developed utilizing principles of the above approach. However, from a hardware viewpoint, the hard limit would require extensive changes to the current ICPS computer design. The above approach could be instituted with minor modifications and is felt to represent a satisfactory strategy for the ICPS, along with scaling, for dealing with the overflow concerns.

5.3.2 WWCS Computational Overflow Protection

The WWCS central processor (MCP-701) is a 16-bit, fractional, two's complement fixed-point computer. Computational overflow, as in the ICPS, occurs in the WWCS whenever the processor produces a variable value outside the maximum value representation range for that variable, a range directly related to the binary word length designed into the computer for variable magnitude storage. While the ICPS used recirculating registers for variable information storage, the WWCS can store variable information in practically any location selected in memory. Figure 5-50 presents an illustration of overflow by comparing

the actual variable stored value with the ideal computed variable value. The following is an alternate illustration of overflow dealing with the numerical representation of the variable in a binary arithmetic format.

To simplify the illustration, a four-bit fractional, two's complement, fixed-point variable representation processor is discussed. Positive variables are represented by the four bits with the leading bit zero and the remaining three bits ones or zeros as needed to represent the variable magnitude. Negative variable values are the same, with the exception that the leading bit is a one. The range of values representable by this four-bit fractional number is shown in table 5-5. Within this numbering system it is easy to understand the process of addition and subtraction. For example, consider the following addition:

<u>Decimal</u>	<u>Binary</u>
0.250	0010
+ 0.250	+ 0010
0.500	0100

where, $1 + 1 = 0$ with a 1 carry to the next higher bit position. This would be a normal operation in the four-bit processor since the binary sum of 0100 is equivalent to the decimal 0.500 value. Now consider the following addition:

<u>Decimal</u>	<u>Binary</u>
0.500	0100
+ 0.500	+ 0100
1.000	1000

Note that the fractional two's complement number set in table 5-5 does not have a binary representation for the variable decimal number of +1. Note further that the binary 1000 number actually represents a full-scale negative number. This example illustrates how the addition of two positive binary numbers will result in the computer interpreting the answer as a negative value.

Computers are capable of performing a wide variety of arithmetic operations, most of which are simply variations on the above example, and all can generally produce a number outside the representation range capability of the computer if misapplied. It should be noted that the occurrence of overflow would be reduced if the entire computation were performed using a floating-point variable representation scheme. In a floating-point scheme, numbers are represented by a mantissa and an exponent, thus greatly expanding the representation range of the computer and proportionately decreasing the potential for generating a number outside the valid range of representation of the computer. The potential still exists, however, since we should never underestimate the power of a programmer. The use of a floating-point representation involves a substantially larger number of operations, thus requiring a larger and more complex processor and slower execution time for the functions to be computed.

Detection of the overflow condition can be performed in either hardware or software. It can be very burdensome on both time and memory to perform this detection in software.

TABLE 5-5.—FOUR-BIT PROCESSOR VARIABLE RANGE (BINARY/DECIMAL)

Binary bit representation	Variable decimal value
0111	+0.875
0110	+0.750
0101	+0.625
0100	+0.500
0011	+0.375
0010	+0.250
0001	+0.125
0000	0
1111	-0.125
1110	-0.250
1101	-0.375
1100	-0.500
1011	-0.625
1010	-0.750
1001	-0.875
1000	-1.000

$\pm 2^0$	2^{-1}	2^{-2}	2^{-3}
X	X	X	X

Four-bit fractional twos complement representation.

In the WWCS (MCP-701) processor, the overflow detection is performed in hardware for all instructions that are capable of overflow generation except one, the arithmetic left shift. Upon detection of the overflow condition, an arithmetic fault (AF) interrupt is generated in the MCP-701, which stores the address where the computer is executing and forces the processor to begin executing at a fixed location. The programmer can then determine in software what should be the response to the fault condition.

In a laboratory system development environment, a simple overflow indication to a programmer can be very useful during the software debugging phase. Overflow in any digital computer system is not commonplace and generally indicates a software error or a lack of proper scaling. For an in-flight phase of system operation, the simple indication of overflow would clearly be an inadequate response.

5.3.2.1 WWCS Overflow Protection Design

Development ground rules for the WWCS overflow protection routine were:

- 1) Limit the affected register and/or variable memory location to full-scale positive or negative value, as appropriate, when an overflow is deducted.
- 2) Provide a record of the overflow occurrence.

- 3) Return the processor to the routine (program) in progress when the overflow occurred.

This would provide a saturated signal response similar to that associated with an analog system.

The first step toward developing a WWCS overflow protection routine was to identify those instructions that have the potential for causing a computational overflow. There are five types of overflow situations in the WWCS central processor (MCP-701). The first, and simplest to understand, is an overflow resulting from an ADD or SUBTRACT operation. The overflow is detected by the arithmetic fault hardware logic, and the sign adder (SA) indicator can be used to identify which sign should be used for the forced full-scale value. The SA will be opposite to the arithmetically correct value following the overflow.

The second overflow situation is related to an illegal multiply (MPY). If the multiplier in memory equals a -1, the multiplication will take place but the sign of the product will be wrong. Special hardware logic detects this situation and an arithmetic fault will be indicated for any multiplication with a memory multiplier equal to a -1. For all but a -1 in the upper register, the wrong sign can be corrected by taking the two's complement of the product.

A third overflow situation exists for the divide (DIV) instruction when the $|\text{dividend}| > |\text{divisor}|$. The magnitude of the results is unpredictable, other than it will be greater than one, but the sign is completely predictable using the SA (sign adder) indicator. Therefore, a full-scale value can be forced as the solution for this divide situation.

A fourth type of overflow detection (arithmetic fault) occurs for the two's complement (CPL) and absolute value (ABS) operations. The two's complement numbering system is such that a full-scale negative value does not have an equivalent positive number, i.e., the two's complement number system range is: $(-2^h) \leq X \leq (2^h - 1)$. The result of taking the two's complement or absolute value of a full-scale negative number is a full-scale negative number [ABS(-1) = (-1)]. The -1 case is detected by the arithmetic fault logic and a full-scale positive value can be forced as the answer.

The fifth overflow situation for the MCP-701 processor is not hardware monitored. This overflow situation can occur for a left shift operation, where the instruction SLZ shifts a register's contents to the left a specified number of bit locations and places zeros in the least significant bit locations. If this operation is utilized for arithmetic scaling, the possibility exists for shifting data into the sign bit location, thus altering the sign significant. Without hardware overflow detection logic for the left shift overflow case, a software scheme would have to be devised before the SLZ instruction could be used in coding flight control functions.

Table 5-6 presents a summary of the instructions that can result in a computational overflow and the conditions under which the overflow occurs. Table 5-7 summarizes the strategies to be taken by the software overflow routine once an overflow is detected and the specific instruction causing the overflow has been identified. Upon the detection of the overflow condition, an arithmetic fault (AF) interrupt is generated, which stores the address where the computer was executing and forces the processor to begin executing the overflow

TABLE 5-6.—WWCS (MSP-701) INSTRUCTIONS WHICH CAN CAUSE COMPUTATIONAL OVERFLOW

Instruction*	Overflow detected by hardware	Treated by overflow protection subroutine	Condition causing computational overflow
1. ADU	Yes	Yes	$UR + [M] > 1 - 2^{-15}$ or $UR + [M] < -1$
2. ADM	Yes	Yes	$UR + [M] > 1 - 2^{-15}$ or $UR + [M] < -1$
3. ADD	Yes	Yes	$UL + ([M+1][M]) > 1 - 2^{-31}$ or $UL + ([M+1][M]) < -1$
4. SBU	Yes	Yes	$UR - [M] > 1 - 2^{-15}$ or $UR - [M] < -1$
5. SBD	Yes	Yes	$UL - ([M+1][M]) > 1 - 2^{-31}$ or $UL - ([M+1][M]) < -1$
6. MPY	Yes	Yes	$[M] = -1$
7. DIV	Yes	Yes	$ [M] \leq UL $
8. MDS	Yes	Yes	$[M] = -1$
9. RASN	Yes	No	$[R] > 1 - 2^{-15}$ or $[R] < -1$
10. RASP	Yes	Yes	$[R] > 1 - 2^{-15}$ or $[R] < -1$
11. RSSN	Yes	Yes	$[R] > 1 - 2^{-15}$ or $[R] < -1$
12. RSSP	Yes	No	$[R] > 1 - 2^{-15}$ or $[R] < -1$
13. CPL	Yes	Yes	$[R] = -1$
14. ABS	Yes	Yes	$[R] = -1$
15. SLZ	No	No	Sign significance shifted out of register

UR → Upper register
M → Memory
UL → Upper and lower registers
R → register

*Reference report FAA-SS-73-2-1 section 6.3.2.3 and MCP-701 Programmers Reference Manual

protection routine. The address of the instruction being executed is interrogated, the instruction identified, and action taken as specified by table 5-7. The execution times for the various strategies of the overflow routine are given in table 5-8.

It was determined during the course of this study that computational overflows caused by two instructions, RASN—register add and skip on negative and RSSP—register subtract and skip on positive, cannot be effectively handled within the software overflow routine consistent with the other instruction's arithmetic fault strategies. For these instructions, the program counter is pointed two locations after the instruction causing the fault instead of the next instruction. Considering that these instructions are primarily used for address modifications and not arithmetic operations, an overflow routine was not developed to handle their special case. A ground rule was issued not to use these instructions in mechanizing the flight control system control laws.

5.3.2.2 Laboratory Evaluation of WWCS Overflow Protection Routine

Following the development and checkout of the WWCS overflow protection software routine, the routine was integrated with the HSAS software for evaluation in the laboratory.

TABLE 5-7.—SYNOPSIS OF OVERFLOW ROUTINE STRATEGY

Instruction	Overflow protection software routine strategies *
1. ADU	Limit UR to '7FFF' or '8000' as appropriate.
2. ADM	Limit [M] to '7FFF' or '8000' as appropriate.
3. ADD	Limit UL to '7FFF FFFF' or '8000 0000' as appropriate.
4. S8U	Limit UR to '7FFF' or '8000' as appropriate.
5. S8D	Limit UL to '7FFF FFFF' or '8000 0000' as appropriate.
6. MPY	Take twos complement of UL.
7. DIV	Set UR = '7FFF' or '8000' and LR = '0000'.
8. MDS	Set [M] to '8000'.
9. RASN	None
10. RASP	Set the appropriate register to '7FFF' or '8000'.
11. RSSN	Set the appropriate register to '7FFF' or '8000'.
12. RSSP	None
13. CPL	Set the appropriate register to '7FFF' or '8000'.
14. ABS	Set the appropriate register to '7FFF' or '8000'.
15. SLZ	None

*Numbers are noted in hexadecimal format.

TABLE 5-8.—EXECUTION TIMES FOR OVERFLOW PROTECTION ROUTINES

Instruction	Execution time (μ sec)
1. ADU	79.2
2. ADM	127.0
3. ADD	86.4
4. SBU	82.8
5. S8D	86.4
6. MPY	87.4
7. DIV	82.8
8. MDS	138.4
9. RASN	—
10. RASP	96.0
11. RSSN	96.0
12. RSSP	—
13. CPL	93.8
14. ABS	93.8
15. SLZ	—

A series of intentionally created overflow conditions was set up to evaluate the overflow protection routine in a closed-loop operation. The scaling of the HSAS control law was modified without changing the loop gain so that overflow would occur more readily. The flight condition and laboratory configuration were the same as used in previously discussed closed-loop tests and are described in section 2.2.

A 10° to $12^\circ/\text{sec}$, 10-sec pulse, pitch rate signal disturbance was used to upset the airplane to cause an overflow situation. Figure 5-53 is the time history response to the disturbance without an overflow being incurred and represents the system nominal response for the baseline case. Trace 7 is the output of filter D, the last filter in the HSAS control law filter string. Figure 5-54 is the time history response when the disturbance is slightly increased and an overflow occurs, with no overflow corrective action taken. The processors were simply directed to continue computation following the overflow. Notice the violent airplane maneuver following the sign reversal caused by the overflow. Figure 5-55 shows the same case with the overflow protection routine active.

During the laboratory testing of the overflow routine effectiveness, it was discovered that a filter instability occurred when the overflow strategy was applied to an overflow internal to the recursive computations of the D filter. Figure 5-56 shows the system/vehicle response when the nonlinear overflow routine limited computation values within the D filter recursive computations. The filter broke into a very high frequency oscillation. To avoid this problem, a limiter was placed at the input to the D filter, since the magnitude of the input that can cause overflow in the recursive expression can be predicted very accurately. These data are presented to show that the strategy included in the overflow routine, while representing the best approach for mechanizing overflow protection in the WWCS, is not a panacea for all overflow problems.

These results point out the need for the inclusion of a limiter at the input of each filter to ensure that overflow does not occur internal to the recursive computations, or that nonrecursive schemes should be used for filter realization in conjunction with the overflow protection routine.

5.4 WWCS SERVO TRANSMITTER/RECEIVER UNIT EVALUATION

The servo transmitter/receiver unit (STRU) was conceived as one method for reducing the number of interconnecting cables between the digital flight control computation electronics and the surface control actuators. A digital interface between the triplex MCP-703 computer units and envisioned remotely located control surface servo drive electronics reduces the number of long signal path runs required for the control of each actuator. Each channel of the STRU communicates with each MCP-703 computer unit via serial digital data links that permit independent commands to be formulated for each servo. A majority logic voter combines the computer units command data to each servo, followed by demultiplexing and digital-to-analog signal conditioning stages. The STRU processes both discrete and analog signals. A more detailed description of the STRU is given in reference 1, section 6.2.3.

Three operational/performance aspects of the STRU were investigated. The first was to determine if problems arose while transmitting serial digital data at least 50 ft between the computer units and a remotely located STRU. The second dealt with processing the electric command servo (ECS) loop closure signals through the computer unit and comparing the servo response to the analog loop closure configuration. The third was to evaluate the effectiveness of processing redundant servo actuators offset equalization through the computer unit.

No special tests were conducted for the first item of interest other than to observe that no operational anomalies resulted from the STRU/computer unit 50-ft digital data transmission link. The cable was installed at the beginning of the WWCS laboratory test program and, other than to acquire some comparison data using a 5-ft link, remained during all WWCS laboratory tests. Each STRU cable (three total) consisted of 15 shielded, twisted pairs of wires. The WWCS STRU interface can accommodate five independent redundant servo subsystems over the existing cable; however, only one set of servo subsystem signal conditioning electronics exists within the STRU experimental unit.

5.4.1 Servo Loop Closure

The laboratory configuration for testing the electric command servo subsystem with the basic position feedback closed through the WWCS is shown in figure 5-57. Tests were conducted to gather frequency response data of the baseline analog system and of the digital system for various hardware/software configurations. The digital system variations consisted of:

- 1) Processing the analog servo input command through the WWCS
- 2) Replacing the analog position feedback path by a feedback path through the WWCS processor
- 3) Same as (2) with the STRU analog signal conditioning prefilter bypassed
- 4) Same as (3) plus removal of the servo feedback signal WWCS input/output double buffering to minimize the processing transport delay

Figure 5-58 shows the frequency response data for the baseline analog servo system compared to the digital system configurations outlined above. The baseline analog servo frequency response (trace 1) is similar to a single-order filter with a characteristic frequency of approximately 36 rad/sec. Passing the analog servo commands through the WWCS (trace 2) adds to the baseline analog response the gain attenuation and phase shift characteristics of the digital hardware, dominated by the analog input signal prefilter. Processing the servo position feedback through the STRU/computer unit (trace 3) significantly changes the servo response. The servo loop becomes lightly damped with a peaking frequency of approximately 28 rad/sec. The peaking results from the gain loss and phase shift in the servo feedback path caused by the digital system prefilter, input/output processing transport delay, and zero order hold digital-to-analog converter. While the effect of the prefilter is independent of the software program structure, the other two processes, i.e., input/output transport delay and zero order hold, are affected by the input/output time slots chosen by the programmer. The following is a further explanation of the I/O process.

The input/output process of the WWCS is a somewhat hardware-fixed operation that provides data double buffer storage to avoid a system timing hazard between the synchronous bit-for-bit I/O operation and the asynchronous operation of the computer unit central processors. Figure 5-59 illustrates the STRU/computer unit data flow process. Three steps are involved: input, computation, and output. For the input operation, data are transmitted from the STRU to the computer unit and read into one of two memory buffer locations, where the two memory locations receive the data alternately every I/O iteration. The programmer can assign one of seven input time slots during the I/O iteration cycle for the variable data to be read into memory. During the computation step, the central processor extracts data from, generally, the memory data buffer not to be updated during the I/O cycle the processor calls for the data, performs computations with the data, and returns the computation results to one of two output data memory buffer locations, alternately selecting the output data memory location not to be output that I/O iteration cycle. The output operation is just the reverse of the input step. Data are extracted from one of two data memory buffer locations, alternately each I/O iteration, and transmitted to the STRU digital-to-analog signal conditioning stage. The programmer can select one of seven output time slots during the I/O iteration cycle for the particular output variable to be transmitted to the STRU.

Without modifying the above data flow process, a programmer can alter the transport delay time of the STRU/computer unit processed signal from a maximum of just under 18 msec to a minimum of approximately 8.4 msec (cases 1 and 2 in fig. 5-59, respectively). Trace 3 of figure 5-58 corresponds to the case 2 situation just described.

The transport delay can be further reduced by altering the software to not utilize the double buffering described, and outputting the servo feedback signal as soon after the input as possible. Case 3 of figure 5-59 results in a transport delay of approximately 2.26 msec when the central processor memory buffer location selection is synchronized with the I/O memory buffer selection. In order to reduce the transport delay time further, the executive routine had to be shortened to allow case 4 of figure 5-59, providing a time of approximately 1.54 msec.

Trace 4 of figure 5-58 is the frequency response of the servo system with the feedback closed, using the case 4 scheme just described. The servo response improved relative to trace 3 and the analog baseline, but the low damping peaking characteristic remained evident.

Trace 5 of figure 5-58 shows the servo frequency response when the servo feedback signal STRU prefilter is bypassed and the digital data processing case 4 scheme is used. The servo response gain characteristic fell essentially coincident with the analog baseline, with the phase characteristic almost unchanged from that recorded for trace 4. This response phase error is attributed to the prefilter and transport delay characteristics of processing the servo command signal through the WWCS.

This portion of the STRU study illustrates that servo feedback signals can be processed through the digital computer subsystem, provided great care is taken to overcome the deleterious effects of the specific systems analog-to-digital and digital-to-analog processes. In practice, as long as the servo system interface requires A/D and D/A processing, the servo

loop will generally be closed with an analog feedback path. However, the servo system response characteristic changes that can be realized using a digital feedback path should be kept in mind as the system configuration is synthesized.

5.4.2 Redundant Servo Actuator Equalization

The laboratory configuration for investigating redundant servo actuator equalization using the WWCS/STRU interface is shown in figure 5-60. Analog signal feedback paths were used for the basic servo position loop closures. The servo load pressure signal (an indication of the redundant servos output mismatch) was input to all three computer units via the STRU. The equalization command signal was derived by integrating the load pressure signal above a preset threshold and summing this signal with the servo command in a negative feedback fashion. Each servo equalization command was calculated in each computer unit to allow different servo command signals to be developed within all three computer units. The servo commands, with equalization, were then transmitted to the appropriate STRU channel.

Two types of tests were conducted. The first involved acquiring X-Y plots of the ECS output versus the servo command input, with and without equalization, with simulated offsets applied to the redundant servo channels. The second test series was to acquire time history traces of the airplane closed-loop response for the offset cases described below, with and without equalization. The servo offsets consisted of applying equal and opposite bias errors to the A and C channel servo loops by introducing a voltage on the servo drive amplifiers. Two test conditions were then evaluated with and without equalization. The first test condition was to have all three channels (A, B, and C) operational with the above offsets. The second test condition was to shut down the B servo (hydraulically deenergizing the B servo), as if the B servo suffered a failure and was disengaged, with the A and C channel offsets still in effect.

The static (X-Y plot) response of the ECS is shown in figure 5-61. Trace I shows the ECS response when channels A and C are offset, channel B remains active, and there is no equalization. A small, but noticeable, hysteresis is evident. Trace II shows the input/output relationship for the same conditions with the B channel disengaged. A large dead zone around null develops as a result of the ECS mechanical midvalue voter characteristic, a typical result predictable for this ECS design whenever offsets exist and one channel is placed at null (spring detented to neutral) in response to a failure state. Trace III shows the same case as trace II with equalization applied. The dead zone is completely removed and the system shows excellent linear characteristics. The equalization not only compensated for the dead zone failure state effect but also eliminated any system hysteresis. Essentially no difference could be observed when equalization was applied to the conditions of trace I or trace II.

Figure 5-62 shows the time histories of an airplane disturbance for the test conditions where channels A and C are offset and channel B is disengaged with and without equalization. The control system function is the HSAS, and the flight condition is that described in section 2.2.1. The airplane disturbance was induced by a 1°, 10-sec duration, column pulse. Trace A of figure 5-62 (no equalization) shows much larger transients to the disturbance than trace B (with equalization) and exhibits very loose control relative to

damping the maneuver following the upset. The ineffectiveness of the servo command of trace A can be observed by noting that the channel A and C load pressures are saturated for long periods.

Equalization through the WWCS/STRU interface to the electric command subsystem can be very effective. In general, with identical commands being produced by the WWCS to the ECS, the need for equalization is substantially reduced. Only compensation for the tolerance differences between the ECS servo channels has to be considered. Such rigging and hardware tolerances may not require equalization compensation to effect acceptable system performance; however, equalization within such a servo system mechanization can greatly reduce system servo valve and actuator wear.

6.0 SOFTWARE DEVELOPMENT AND CONTROL

Risk uncertainty relative to an analog redundant flight-critical control system is kept within acceptable bounds as a result of the current detailed knowledge of analog system circuitry behavior and the procedures developed for analyzing and verifying the operation and failure effects of such circuitry. The entire analog system configuration/operation is defined through documentation that describes the linking together of specified hardware elements, all having documented performance specifications as to their operational norm and tolerances about that norm. A change to the system, i.e., adding, deleting, or substituting one element for another, must be accompanied by an appropriate analysis that evaluates the system change effect on the surrounding elements' performance and then to the overall system operation. The objective is to ensure that the risk uncertainty of the redundant system's predicted operation and failure effects remains acceptable. The change is recorded on a one-for-one correspondence between the element changed and the system documentation covering the specification of that element and how it is used in the system.

The above essentially introduces the fact that analog systems have but one documentation format for maintaining system configuration definition and control. While the definition and control of a digital redundant flight-critical system must only meet standards comparable to those currently used for analog system designs, two forms of documentation must be preserved to complete the digital system's operational description; hardware, as with the analog system, and software, a description of the resident computer memory program that controls the operational state of the hardware in the discrete time domain. Standards must therefore be developed pertaining to the digital system's software development, validation, documentation, and change control to provide the configuration definition and control needed to ensure a low level of risk uncertainty for flight-critical system applications.

This section of the report presents a brief description of the software development procedures utilized during the FCD task and projects software documentation and control requirements envisioned as a result of the experience gained.

6.1 FCD TASK SOFTWARE DEVELOPMENT

Application program software preparation can be the responsibility of either the digital computer system hardware manufacturer or the overall system integration prime contractor. For the FCD task, Boeing retained full responsibility for preparing the resident programs of the two digital systems, short of the built-in-test and control panel interface service routines for the WWCS. This provided the necessary experience desired to understand the software development cycle in order to identify the procedures and documentation deemed necessary in the development of a flight-critical digital system.

Software coding was directed and prepared by flight control system engineers with the aid of engineering programmers versed in assembly language programming. General Electric, the digital computer system hardware supplier, provided software development support through consultation, machine language indoctrination courses, and software preparational aids, i.e., assembler, linkage editor, and simulator software support packages.

6.1.1 ICPS Programming

Software preparation for the incremental control processor subsystem (ICPS) involves programming only the control law application functions for the ICP-723 computer and the signal selection/failure detection parameter values for the SSFD function in the computer interface unit. All other ICPS operations, including the redundancy management functions, are fixed by the hardware architecture, with only a limited level of variations easily made through hardware changes; e.g., input/output variable time slot selection within the processing iteration frame.

An alterable core memory was used with the ICP-723 computer unit, permitting program loading to be made via a punched tape input or manually through the program loader unit. Program insertion during the FCD task utilized punched tape with corrections or minor alterations made manually. The programming procedure for the control law functions followed the steps described in section 3.2.1, *HSAS Implementation Using ICPS*, with software control maintained through the establishment of master source deck/algorithm map documentation. Since the ICPS computer unit programming consists of only those functions dealing with the application control laws, memory usage during the FCD task was covered by the memory summary information presented in table 3-2. Out of an available 3072 usable memory locations, 1630 were utilized for the functions programmed. While this constituted only a little over 50% of the memory available, the programs developed collectively required more algorithm times than the available 128 per iteration.

Programming the ICP-723, as with most airborne computers, required becoming very familiar with the computer's particular programming language. To efficiently and effectively program the ICP-723, however, requires that the programmer have a greater knowledge of the hardware operation than is generally the case for programmers dealing with a general-purpose computer architecture, such as that associated with the WWCS. This greater knowledge requirement is substantially offset from a control system engineer's viewpoint in that the ICP algorithm map structure permits an easy correlation overview between the functions programmed and the functions defined by a system functional block diagram.

Programming the failure detection parameters for the CIU SSFD function required programming reprogrammable read only memory (PROM) devices. Six parameters (time constants, thresholds, and time delays) had to be set for each sensor signal type. This programming was primarily a manual task where the specific memory device (plug-in) was removed from the CIU and was programmed utilizing special laboratory support equipment for programming such devices. The support equipment did facilitate punched tape programming of the device; therefore, a software support package was developed to generate a tape and provide a hard copy (listing) of the parameter values. Reprogramming the parameter memory devices was a time consuming task for the limited number of input signals investigated and the changes made, eight sensor signal types out of a possible ICPS capacity of 64. A more efficient method of implementing the parameter values or changes should be developed if the ICPS is applied toward a production development program.

Specifying the ICPS software would follow the same guidelines adopted for any digital airborne system (discussed in sec. 6.2). Documentation would include the algorithm map which, in some ways, replaces a general purpose system requirement to have a macro flow chart of the system software.

6.1.2 WWCS Programming

Software (resident programs) of the whole-word computer subsystem (WWCS) provides a great deal of the system's operational and redundancy management configuration definition. Software programs cover not only the application control law functions but also all the functions related to the basic executive structure of the processor program sequencing, signal selection/failure detection processes, failure mode determination, initial condition management, built-in-test administration, and the essential interactive system structure to interface the computer unit with the computer monitor and system status control units. As a result, the resident program for the MCP-703 computer unit memory began as many independent program development efforts, assigned to different engineer/programmers, ultimately to be linked together and integrated with the hardware to become an operational fail-operative computer subsystem.

Software specifications and development and control procedures, although discussed, were not established by appropriate documentation at the outset of the WWCS software development activity. Boeing maintained software development responsibilities for the WWCS executive, control law routines, redundancy management, and the system application preflight test program. General Electric developed the ground-support-equipment/control-panel service routines and the built-in-test program. Prior to the WWCS equipment delivery to Boeing, the executive was integrated with the GE-furnished software and became an operational element in the WWCS acceptance test.

As part of the Boeing preparation for receiving and laboratory-evaluating the whole-word system, a software control procedure was established to go into effect two weeks after the arrival of the WWCS hardware. Such a procedure was instituted to exercise the questions pertaining to software control and to serve as an experience factor in determining the usefulness of such procedures. The following section presents the control procedures set forth, followed by a section that discusses the direct experience and observations derived.

The total program capacity of the WWCS memory is 8192 words. Of this, the final WWCS resident program consisted of approximately 1800 words programmed by GE and 4300 words programmed by Boeing. A dedicated section of memory (1024 words) was utilized for input/output variable and discrete data, with the remainder considered spare. Figure 6-1 summarizes the final WWCS memory usage relative to the application programs developed under the FCD task. Although the entire input/output section is hardware dedicated through programmed ROM devices, less than 10% was utilized in support of the FCD application programs.

Support software for programming the WWCS MCP-701 processor consisted of an assembler, linkage editor, and simulator (ref. 1, sec. 6.4). The assembler and linkage editor are vital elements in the development of system software, especially when several engineer/programmers are coding separate but interrelated programs. The simulator, while generally effective in the debugging phase of software routines, has only limited use with respect to a triplex system's software development. The simulator is representative of only a single processor and cannot be used to debug code involving the exchange of information between triplex processors. For the FCD task, the simulator was used in the development of

the control law programs. Use of a simulator can be very expensive relative to the other support software items and alternate aids should be sought, the most effective being an actual computer very similar to, or a prototype of, the processor to be used in the triplex system. Another alternative is to phase (schedule) the hardware/software development tasks so that hardware is available to support software development prior to the total system integration effort.

6.1.2.1 WWCS Laboratory Phase Software Control Procedure

The software control procedure presented below was established to become effective shortly after the WWCS hardware was delivered. The reasons identified for adopting some form of control were:

- 1) To minimize the probability of incurring the following types of confusion while conducting the FCD laboratory evaluation of the WWCS:
 - a) Source decks lost, destroyed, or inadvertently changed
 - b) Program tape confusion
 - c) Disc file chaos
 - d) Mismatch of decks, listings, disc files, and tapes
 - e) No central awareness of current software status
- 2) To provide a working background from which recommendations can be derived for future programs

Since a considerable amount of debugging and experimentation was anticipated during WWCS laboratory tests, considerable freedom was to be given the individual programmer/experimenter while at the same time ensuring that untried or unwelcome changes did not pass into the controlled "Operational" deck. The baseline "Operational" deck would be proclaimed following WWCS startup in Seattle and integration of the FCD flight control functions into the software delivered with the hardware.

WWCS Software Control Procedure Directive

- 1) Software control board: This board will be made up of all members within the FCD group. One member, designated as chairman of the board, will have the following responsibilities:
 - Chair software control board meetings.
 - Designate "experimental" deck and "operational" deck numbers. Issue such numbers and new decks with concurrence of the program manager.
 - Oversee the WWCS software library.

- Provide the central awareness of WWCS current software status.

An FCD group meeting will be held weekly to review WWCS laboratory testing activity and to convene the WWCS software control board. The software control board library will retain up to two copies of the current operational source decks and listings, along with pertinent historical copies of outdated operational and experimental decks.

- 2) Software organization: To minimize card handling during studies requiring several new assemblies, the software has been partitioned into five major blocks:

- Bootstrap load
- Executive block
- Redundancy management block
- BIT block
- Flight control block

These blocks will be located in respective disc file locations on the IBM-360/370 systems having the WWCS support software. Each experimenter will generally be dealing with only one block and can, therefore, update a program handling only one block's worth of source cards.

- 3) Library numbering system: The initial baseline operational system (deck) will be labeled P01. Subsequent control board designated updates will be P02, etc. The initial experimental block(s) source deck will be designated X01 with subsequent experimental blocks labeled X02, etc.

The P or X designator *must be included in the TTL card of every processed program*. The TTL card will appear as follows:

TTL DOT/SST/FCD X18

Use of the TTL cards in this manner places labels on the deck, listing, and disc file. Punched tapes must be labeled by hand, for example:

X18 3-15-74
New Overflow Protection

- 4) Record keeping: Approximately 2 weeks after the arrival of the WWCS equipment in Seattle, a baseline operational deck, listing, and appropriate disc files will be created. All will carry the P01 designation. *The decks (cards) will be sequenced in columns 73-80.*

At the weekly FCD group meeting, the software control board will approve and issue X numbered test decks, recording the fact in a test deck log (see fig. 6-2) along with the requestor's name and purpose. These decks will essentially be copies of the current P designated block or blocks requiring the TTL cards to be changed to reflect the X designation (a most important step).

For the duration of the particular X test, the programmer/experimenter is completely free to modify the X designated block(s) issued for that test. During the link edit process, the test block(s) should be placed in a scratch disc file since the block(s) permanent file will contain the latest P version. At the conclusion of the X study, the experimenter will report to the control board the study results and recommendations as follows:

- a) *Close X*: Study failed to produce desired result relative to error correction or improved configuration/performance benefits. The X test log entry will be closed out and the X source deck will be destroyed.
- b) *Save X*: Study produced qualified results; i.e., objectives of test were not completely met or the board withheld authorizing the proposed improvements from being entered into the operational P program. The X log will be closed out noting the study status or board action and the X source deck and listing will be retained in the WWCS software library.
- c) *Update P*: Study produced results that are desirable for error correction and/or configuration improvements. After board approval, a full review of the proposed change(s) will be made with respect to all other aspects of the total operational program and outstanding X studies. After this has been completed, the X source cards will be upgraded to P on the TTL card and the appropriate permanent block disc file updated. An entry will be placed in the P log (fig. 6-3) reflecting the block(s) changed and the reason for the change. A copy of the preceding operational P source deck will be retained in the WWCS software library.

6.1.2.2 WWCS Software Control Results

In actual usage, some modifications to the above procedure were made. Because the WWCS was an experimental system and the laboratory testing at Boeing was to be the initial integration of the hardware/software, many last minute patch changes were made to the executive and built-in-test programs just prior to and during the equipment acceptance test. It was felt that these programs should be corrected (cleaned up) prior to issuing the first operational (P01) master program. A setback also occurred, following equipment delivery, in the scaling of the flight control law programs (simulator debugged routines) when it was recognized that the assumed input variable data (12-bit information) positioning in the memory buffer (16-bit word) was incorrect. In fact, instead of approximately 2 weeks after the arrival of the WWCS at the Boeing laboratory, the P01 freeze did not occur until 6-1/2 weeks into the laboratory phase, and then the built-in-test program was omitted to allow further uncontrolled corrections to be made.

The problem that immediately became apparent was that P01 became nothing more than a reference point and not the working program for the experimenter/programmer. All users wanted to use the latest test modules (X01, X02, etc.) rather than P01. Thus, it was decided to place the latest X programs on the working disc files. This meant that the updaters of X01, X02, etc., had to be very meticulous about their program checkout. Users who did not wish to use the X files did have access to the P01 program. This situation of using the X files worked well, but it must be recognized that during the FCD task only one engineer/programmer was assigned to each major block of software. If several programmers had been working on different sections of the same major module, the procedural change would not have been acceptable.

A second problem arose with respect to the P programs relative to the calendar time involved for duplicating, sequencing, interpreting, assembling, and checking out an updated P designation. It became apparent that the creation of multiple P decks would adversely affect the FCD laboratory schedule so the impact was minimized by only producing two P versions, the initial P01 and a P02 that was the integrated master program at the close of the WWCS laboratory experimentation.

There were many benefits derived from the software control procedure exercise. No confusion arose over what versions of which program were where. The change documentation allowed historical records of the changes made with respect to the P01 program. No disc files were corrupted, and the TTL labeling scheme along with the date and time provided in the linkage editor output proved highly successful. Because of clear labeling, no confusion arose relative to the Mylar/paper tapes used in the laboratory.

Probably the most important thing visualized from the software control procedure activity was recognizing that there are considerable costs involved in software control. Programmer productivity drops in order to actively retain appropriate records, and significant program (project) costs can be incurred in producing master program and experimental program duplicate decks, listings, etc. Such software controls should never be scheduled, therefore, until the program maturity and overall project schedule reflect the need for the benefits derived. In addition, it should be noted that the most rigorous software procedure can be defeated without cooperation and time allocated to exercise the control process.

6.2 PROJECTED SOFTWARE DEVELOPMENT AND CONTROL REQUIREMENTS

The discussion that follows deals with the documentation requirements projected to be associated with the procurement of a digital flight control system. Digital system documentation differs from contemporary analog system documentation in that hardware descriptive information alone does not define the operational subsystem. Software, documentation that describes and records the functional program residing in the computer memory, must become a part of the system design description akin to schematics, circuit diagrams, and parts lists that describe elements of an analog system, or for that matter, that describe the digital system hardware. Therefore, in order to procure a flightworthy digital system that satisfies the user's specific performance requirements, the system software must be specified, developed, documented, accepted, and controlled in a manner similar to that now used for flightworthy hardware.

As with current analog flight control system procurements, a specification must be issued to define the standards to be followed for the hardware design, performance, fabrication, and acceptance testing. Where this often covered the system operational requirements as a result of the specified hardware functional organization, a supplemental specification will have to be issued for a digital system that covers resident software functional organization, programming standards to be followed, documentation, and acceptance testing. Such specifications will generally be the responsibility of the prime contractor. If the system is procured as an operational subsystem (versus procuring computers and developing a system in-house), the supplier would become responsible for delivering, along with the hardware, documentation which:

- Describes the hardware
- Describes the resident program
- Provides the acceptance tests for both hardware and software

Errors in the design of an analog system must be removed through careful analysis and laboratory testing of the system in parts as well as the whole. Digital system software, as such, will not pose a formidable risk in the development of a flight-critical system if standards and practices are used which are patterned after those currently in use to assure that an analog system does not have built-in design shortcomings. The following sections discuss possible approaches to the software specification (requirements document) and digital system acceptance tests.

6.2.1 Software Requirements

A software requirements document is the formal channel by which information should be passed to the digital system software supplier's programmer. Many requirements may be identical to specifications found in the hardware specification documentation and cross-references should be used rather than attempting to duplicate the requirement. Software requirements should address three areas:

- Performance (operational) requirements
- Design requirements
- Test requirements

The following subsections cover the first two areas with section 6.2.2 covering the third area, along with a discussion on software acceptance.

6.2.1.1 Performance Requirements

The performance requirements section of the software specification should present system definitions in terms of the function to be performed, provide block diagrams, and state hardware design details that may affect the software. Normally, flight control design specifications provide a block diagram of the system function expressed in Laplace

transform (S domain) notation transfer functions, which usually read from left to right. This same information must be presented to a digital system programmer in such a way as to provide the following features:

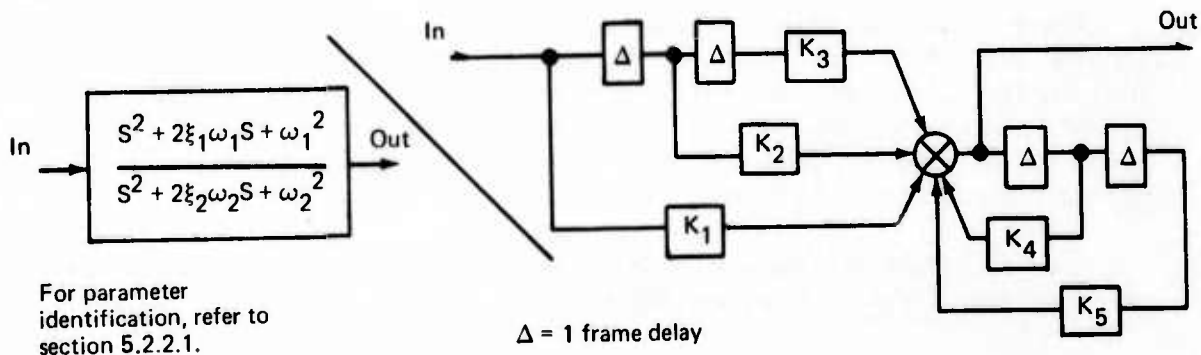
- Total unambiguity
- Ease of communication between programmer and system engineer (who may be at widely separated locations)
- Straightforward translation from the information provided to the actual program

To accomplish this, it is suggested that special purpose block diagrams should be produced to contain the following information:

- Elemental forms of all transfer functions
- Complete logic equations
- Complete mode control specifications for every integrator
- Variable names where an observation point is needed, or for gains or time constants
- Binary scaling

If diagrams presenting this information are produced, then program flow charts as such may not be necessary, except for the system executive program. From this point, the special purpose diagram will be referred to as a "system diagram" to distinguish it from the more usual "block diagram." The five aspects of the system diagram are explained in greater detail below.

Reduction to Elemental Forms—The reason for reduction to elemental forms is to force the engineer to visualize the system as it is implemented by software. This will be of great help to the engineer when the program is in the debugging stage, and it makes the programming task considerably easier. The particular elemental forms presented will depend on the algorithm chosen to represent transfer functions. For example, if the method of bilinear transform is selected, a generalized second-order transfer function would appear as shown.



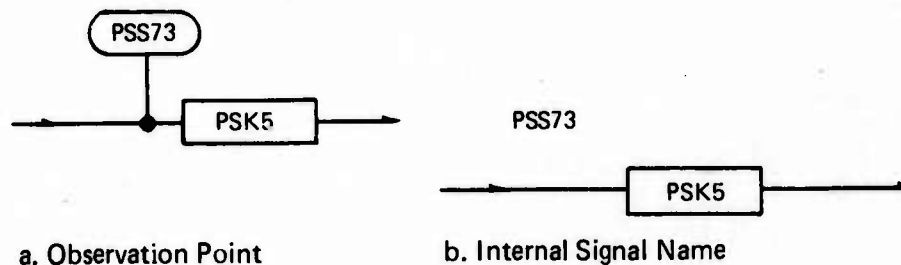
Complete Logic Equations—The area of logic equations is usually one where specification to programmers is weakest. Two typical problems are:

- 1) Are the switches of the latching type, and if so, what condition breaks the latch?
- 2) Incomplete specification—e.g., “open for condition A, close for condition B.” Can we ever have neither A nor B? If so, what then? Can we ever have A and B at the same time? If so, what then?

If a set of Boolean logic equations using the operators AND, OR, and NOT are provided, then the problems are resolved. The information in a logic equation is readily programmed on a general-purpose digital computer.

Mode Control for Every Integrator—It is very tempting for a programmer either to omit mode control where not specified or to invent his own. This is frequently a source of trouble and may be resolved by engineering specification of the conditions under which each integrator is allowed to run, reset, or hold.

Assignment of Variable Names—Observation variables within the system diagram must be indicated by a special code, e.g.,



Observation points must be chosen with a view to checkout and debugging of the system. It should be remembered that it is not easy to add observation points once the program is written and that observation points may add to the core size of the program.

These and other variable names should be assigned on a basis similar to that shown by table 6-1. The programmer will, of course, assign additional names as the program is being written. It is neither necessary nor desirable that these names appear on the system diagram.

Binary Scaling—In a fixed-point machine, the number range that can be held by the computer is $-1.0 \leq n < 1.0$. Any attempt to produce a number outside this range will result in an overflow. Overflow is, in most circumstances, a failure condition that must not be allowed to occur. Overflow for some specific functions can be protected against through careful scaling. However, great care must be taken to specify these cases, or overflow protection schemes similar to those discussed in section 5.3 must be imposed.

Input Requirements—This section should provide precise details of all inputs to the software system. The points to be covered are:

TABLE 6-1.—WWCS PROGRAM SYMBOLOGY

Symbology character guidelines	6-character maximum/4-character minimum 2-character system mnemonic 1- or 2-character function mnemonic 1- or 2-character digit numerals		
<u>Example: system mnemonics</u>		<u>Example: function mnemonics</u>	
PS	Pitch SAS	FL	Flag
PE	Pitch ECSS	SW	Switch
PC	Pitch control wheel steering	S	Signal
RS	Roll SAS	SP	Signal from past frame
RE	Roll ECSS	RP	Rate limit value (positive)
RC	Roll CWS	RN	Rate limit value (negative)
YS	Yaw SAS	LP	Position limit value (positive)
AT	Autothrottle	LN	Position limit value (negative)
SE	System executive	L	Position limit for symmetrical limiter
SO	System output	DZ	Symmetrical deadzone
SI	System input	DP	Deadzone + value
SD	System discrete	DN	Deadzone - value
		K	Simple gain
		TC	Time constant
		CN	Fixed constant

- Maximum signal range
- Conversion factors from engineering units to volts to a binary number
- Signal update rate
- Maximum rate of change of signal
- Number of significant bits input
- Synchronous or asynchronous arrival of data
- Maximum arrival rate (discrete inputs)

Output Requirements—This section should specify all constraints to be imposed on the output signal. The points to be covered are:

- Maximum signal range
- Conversion factors
- Minimum tolerable update rate

- Number of significant bits to be provided
- Maximum transport delay between arrival of input and production of corresponding output

6.2.1.2 Design Requirements

This section should specify requirements that affect the design of the system. Examples of design requirements are language usage restrictions and subroutine structure. The following general remarks may serve as an example of some design requirements.

The program listing itself should be considered to be a major part of the documentation of the flight control system. The programs should thus be modular, logical, and at least partially comprehensible to a nonprogramming engineer. The interfaces between the various subroutines and the executive should be as simple as possible, well described and documented.

To do this, the program should make use of macros and subroutines wherever possible, which means that the chosen computer should have an assembler with macro capability available. Use of macros leads to very fast program checkout and facilitates program changes. Macros should be produced for such things as integrators, dead zones, rate and position limits, and the various first- and second-order transfer functions. In certain cases, use of macros for, say, integration means that different integration algorithms may be tested without changing the main body of the code.

The programmer should make heavy use of comments especially to give scale factors. The purpose and call sequence of each subroutine should be commented at the head. Clever programming such as instruction modification should be avoided, but in those cases where it is expedient to use it, comments should be used for clarification. It should be remembered that the program will probably have to be modified at a later date by someone who is totally unfamiliar with the program.

The flight control system should be separated into functional entities that should be coded as separate subroutines. Every subroutine will be called once per frame by the "main" vectoring program. The reason for this is that we may wish to control the rate at which one block operates with respect to another. This will be referred to as a multirate system. Each subroutine should have a frame control word associated with it containing two items of data:

- 1) If we wish the subroutine to be executed every nth frame, this number (a) should be equal to "n."
- 2) The starting frame number (b) should be less than or equal to "n."

For example,

- $a = 1, b = 1$, means execute the subroutine every frame.

- $a = 5, b = 3$, means that the subroutine is executed every fifth frame starting on frame 3; i.e., on frames 3, 8, 13, etc.

This system provides a high degree of flexibility and relieves the executive program of having to make decisions about execution of the subroutines. Each subroutine should create its own local version of the frame time interval as a function of the frame control word to account for multirate iteration intervals. The global frame time interval should never be modified.

If core space is available during a systems software development, and some kind of difference equation method is used, each subroutine should compute the Z domain equation coefficients directly from the time domain parameter values. These calculations need to be done only once and certainly not in the run mode. The advantage of this is that during checkout and system development, some changes to time constants may be made directly, and multirate experiments may be carried out with very little change to the system.

When the program has been declared operational, these calculations may be removed and the constants built into the program.

6.2.2 Software Acceptance Test

When a contemporary analog flight control system has been designed and fabricated, the product undergoes an acceptance test procedure, the single goal of which is to ensure that the system operates within specified tolerances. Tolerances must always be given because the system has been manufactured using discrete components which only approximate a nominal value or will drift from their nominal value with time.

With a flight control system mechanized using general-purpose (whole-word) digital computers, the equivalent task should be split into two distinct categories:

- 1) Acceptance test of the computer and its related hardware
- 2) Software acceptance test

The computer system hardware acceptance test is not discussed here other than to point out that hardware tests will usually involve some kind of executable software program. The flight control law program itself will generally be unsuitable for such testing because it will be inherently unable to create the worst case conditions to exercise the hardware. Hardware testing will have to be repeated for every system manufactured. In contrast, full software testing for a production system needs to be done only once, with the production memory verified against the master program.

If the software is ever changed after the initial production version, further software test and acceptance would have to be conducted, an aspect that should be covered by the change control procedure.

Because of finite word length and computation speed, digital computers can only approximate desired performance in flight control applications. The required performance and tolerances will have been specified in the software requirements document, and part of the acceptance test procedure will be to verify that the algorithms and programming methods used have created a system that meets the specifications and tolerances.

A common aphorism is "there is no such thing as a checked-out program." Fortunately, this is not always true and in general it is most applicable to vast software systems. The traditional method of program checkout is to provide known inputs to the program under test and then to compare the output with an independently calculated (or computed) source. This testing method is designed to reveal errors in the program, but even when the results are correct, it has not been established that the program is correct. This is a fine distinction, but an important one.

What is needed is a new dimension for looking at the program. The computer sees the program as a series of executable instructions, but by actually reading the listings we can sweep through the program from top to bottom by tracing variable usages, subroutine by subroutine, macro by macro, etc. To follow a program listing should not be difficult, especially if the program has been written in a modular manner and is clearly commented. A symbol cross-reference table can be an invaluable aid in debugging since it traces out every usage of every symbol in the program. For example, if a flag is created by a logic equation and used twice, we should expect to see three usages in the symbol table. More or less than three usages would be a reason to investigate but would not necessarily indicate an error.

In short, the program listing should be checked against the system block diagram (which will contain many of the variable names) before any attempt is made to check it by the method of actually running it.

The software test procedure should have the following objectives.

- Ensure that all the elemental functions (switches, rate limits, etc.) are present and operating correctly.
- Ensure that all the flight control system logic is present and executing correctly.
- Ensure that the end-to-end performance is adequate in terms of resolution.
- Ensure that the end-to-end performance is adequate in terms of frequency response.
- Ensure that there are no scaling errors of analysis or implementation.

In theory, all of the test objectives could be met by the use of a simulator. However, simulation would not reveal potential problem areas such as interrupt timing and CPU-I/O interaction, and the whole procedure would have to be prefaced by a complete test of the simulator. For these reasons this approach is not recommended, and it is assumed that software testing will be done using the flight control computer itself. The following approach is envisioned for laboratory testing the system software.

6.2.2.1 Elemental Function (Software Algorithm) Check

If the flight control system is simple, then a complete check can be done by monitoring the normal outputs and applying known conditions to each input. Normally, however, the small effects being investigated will be masked by other elements in such an end-to-end test procedure. Thus, the capability to isolate each functional element within the rest of the flight control system and to force a known condition onto its input is needed in the software. Each elemental function should be checked using appropriate stimuli from a function generator (potentially a software function generator could be used) having the capability to produce:

- Pulses of variable amplitude and duration
- Ramps of variable slope
- Sinusoidal signals of variable amplitude and frequency

The functional generator output should, in general, be capable of being patched anywhere in the program. Similarly, the test output should be arranged in a general enough way that any cell in the program can be output and monitored. This is analogous to the provision of test points within conventional flight control equipment.

To test the individual elements, every effort must be made to cover all possible conditions. It would be tedious to tabulate element types and suggest tests to be performed on each but, in general, signals of both positive and negative polarities should be used and amplitudes adjusted so that performance both within and outside limit values can be measured.

6.2.2.2 Switching Logic Test

In general, flight control system switching logic is done in three distinct stages:

- 1) Create individual condition flags.
- 2) Combine condition flags in a logic equation.
- 3) Use the logic equation result to effect the operational state of the system.

The first step is to check the conditions necessary to set a flag. Generally, this will be accomplished similarly to the way logic is tested in an analog system. Where possible, outside conditions will be set to bring about the condition(s) desired, and the state of the system will be observed to note the action expected from the logic equation. However, some logic functions will be internal to the software, and these functions must be checked in a modular fashion as an elemental function. This should preclude testing the logic equation for all permutations of the logic statements.

6.2.2.3 Resolution Testing

For this test, all input signals except one should be clamped, and the magnitude of input to produce a discernible (single bit) output change should be controlled. This procedure should be repeated for all relevant input signals using both polarities. The function output resolution should be less than or equal to those specified in a software requirements document. The purpose of this test is to observe the cumulative effect of cascading transfer functions, many of which may have linearity discontinuities (causing deadzone effects), depending on the implementation method and scaling used.

6.2.2.4 Frequency Response Test

For this test all input signals except one should be clamped and a small amplitude signal fed into the remaining input. The output will be measured, and gain and phase shift versus frequency of input will be plotted. The purpose of this test is to determine if the transfer function implementation method has produced any deviation from the ideal frequency response which is greater than that allowed by the specification in the software requirements document.

6.2.2.5 Binary Scaling Test

The previous tests will have gone a long way toward testing for bad scaling, but a final series of tests should be run, designed to create maximum signals at critical points in the system. To do this at least two input signals would have to be used simultaneously. The input type (i.e., sine wave or step) that creates the worst condition will have to be established through analysis of the particular flight control system signal path(s).

7.0 ICPS FLIGHT TEST EVALUATION

Supplemental to the ICPS laboratory evaluation covered by the FCD task, the ICPS was flight test evaluated in conjunction with the flight test phase of the advanced electronic display system (AEDS) task, task VI, of the SST Technology Follow-On Program. This flight test program became part of a NASA terminal control vehicle (TCV) project, which included a contract with Boeing for a research support flight system (RSFS) installation on a NASA-purchased 737 airplane (contract NAS1-12122). This section presents a summary of the flight test program conducted with the ICPS. Further details are documented in the supplemental flight test report issued as part of the aforementioned NASA contract.

7.1 AGCS FLIGHT TEST CONFIGURATION

The supplemental flight test configuration represents a significant portion of the advanced guidance and control system (AGCS) implementation. The Boeing-developed AGCS concept treats the composite task of navigation, guidance, display, and automatic flight control on a system basis. The incentives for the AGCS approach, in lieu of traditional approaches, are the potentials for cost, performance, and safety improvements.

AGCS hardware, figure 7-1, provides guidance and control functions for a commercial aircraft in the future air traffic control environment. The elements of an AGCS are sensors, computers, servos, and cockpit displays and controls. The AGCS interfaces with the airplane primary flight controls, engine controls, electrical and hydraulic power, and with the flightcrew.

The experimental AGCS implemented in the NASA B737-100 airplane (N515NA) for the supplemental flight test experiments was configured from Government-furnished equipment used in the AEDS and FCD programs, complemented with elements furnished by Boeing. The redundancy configuration was limited, therefore, in comparison with a fully redundant AGCS, to that which could be mechanized within the constraints of the available equipment and the existing airplane (fig. 7-1).

The first distinction made in the AGCS is that of associating each function with its potential effect upon flight safety. A function (e.g., autoland) that, in the event of a failure in that function, could adversely affect short-term flight safety while it is in operation, is designated "flight-critical." Other functions that may suffer a failure without adversely affecting flight safety are designated non-flight-critical or noncritical.

Noncritical functions of the experimental AGCS, performed by the Litton C-4000 navigation computer and the GE display system, include:

- Navigation
- Map data generation
- Display generation

- 2D, 3D, 4D guidance
 - Command generation for display
 - Path error generation for steering
- Altitude, flightpath angle, track angle select/hold
- Airspeed select/hold
- EPR limit control

Flight-critical functions of the experimental AGCS were performed by the GE ICP-723 triple redundant flight control processing system. They include:

- Attitude control wheel steering
- Velocity vector control wheel steering
- Final approach track capture
- Autoland—straight in segment
- Failure monitoring and isolation

Communication between the flight control computers and the navigation computer is via two digital serial lines and discrete lines. Three serial digital lines to and one from the display system carry the display data processed in the navigation computer, plus required mode control information. Figure 7-2 shows the interface between the units.

The AGCS mode select panel (MSP) interfaces with the flight control computers are triple-redundant discrete lines both ways. MSP communication with the navigation computer is on serial digital lines.

Existing sensors in the airplane were interfaced with the navigation and flight control computers. Sensors were added to the experimental equipment to provide the required redundancy level for flight-critical functions, including a third ILS receiver and a complete set of triple pitch and roll rate gyros. A third air data computer was added to the airplane installation, and the CWS force sensors were made triply redundant.

The onboard data acquisition system (DAS) provided for gathering of data of four different formats. Forty-seven analog signals could be recorded at any one time on FM tape recorders. All redundant sensor signals entering the flight control computers were reformatted and recorded on tape in digital form. For the ADEDS program, 32 words of data in the ARINC 561 format were recorded, and video recordings were made of the EADI and MFD displays.

7.2 FLIGHT CONTROL SUBSYSTEM

The heart of the automatic flight control system, figure 7-3, is the GE ICP-723 incremental control processing system (ICPS). The ICPS is mounted in the flight control computer (FCC) pallet, figure 7-4.

The hardware elements of the ICP-723 computer system consist of the following 12 units:

- 1) Flightworthy Equipment
 - a) ICP-723 computer (three)
 - b) System interface unit (three)
 - c) Program memory unit (three)
- 2) System Support Equipment
 - a) Program monitor unit
 - b) Program loader unit
 - c) System status and control unit

The ICP-23 computer unit is a time-shared variable increment control processor with 128 algorithms of on-line computational capacity. This measure of capacity reflects the incremental nature of the computation as well as the fixed iteration rate of the machine. "Algorithm" is a name for the computational operations that are performed by the time-shared incremental arithmetic section during a specified time interval during each computer iteration. A detailed description of the ICP-723 computer is presented in the system description document, reference 1, section 5.3.

The computer contains extensive on-line internal monitoring. Parity checks and timing monitors are used to provide a comprehensive failure detection capability. In addition, computational overflow monitors are provided.

The computer interface unit performs all input and output signal conditioning necessary for the computer to communicate with other elements of the system. In addition, it performs failure correction and monitoring on all inputs and outputs and provides a highly reliable timing signal to the computer.

Sensor input data are first converted to digital form and then cross-fed to the other two channels on one-way serial data lines. Each individual input is failure detected with a cross-channel monitor that compares them in whole-word digital form, figure 7-5. The monitor is organized to process a maximum of sixty-four 16-bit two's complement serial binary words that represent 64 individual sensor inputs. All of the monitor hardware is time shared so that it is used 64 times within each computer iteration (163/sec.).

The failure monitor contains two filters, two thresholds, and two time delays, and all six of these parameters can be tailored differently for each of 64 individual sensor monitor functions. Program memory for the monitor is contained in PROM devices.

Following the monitoring functions, input data are failure corrected with a median value selector, figure 7-6. This accepts a digital whole-word input from each of the three channels and gives one output that is identical with the median value of the three inputs before the monitor has detected a failure. Following the detection of one failure by a sensor monitor, the output of the sensor selection function will be the average of the two remaining good channel signals for the affected signal only. This has no effect on all other signals, which will continue to be median selected in the normal fashion until a first failure in one of them is detected. Following the second failure of a given sensor, second-failure information is generated to initiate appropriate shutdown switching.

With this particular configuration, the computer and interface units operate redundantly independently so that a failure in a computer in any one channel does not cause the loss of the interface unit in the same channel, and vice versa.

The program memory unit provides an electrically alterable program storage, which permits convenient and rapid reprogramming of the computer during the laboratory and flight test development phase of a program.

The unit contains a 4K x 18-bit core memory that provides 256 algorithms of nonvolatile program storage. Timing, control, and input/output buffering are provided for the transfer of data to the computer, and also for read/write operations with the program loader unit.

Sensor failures detected by the ICPS are annunciated by type and channel on a sensor failure display, figure 7-7, on the flight control interface (FCI) pallet, figure 7-8. Local and first-failure states are also annunciated on the ICP system status and control unit (SSCU), and first-failure state is annunciated to the pilots on the approach progress displays (APD).

All interfaces between the FCC pallet and the rest of the AGCS, the airplane sensors, and the airplane servos are provided through the FCI pallet as indicated by figure 7-3. The types and number of sensor signals to the FCC are listed in table 7-1. Strict electrical separation of redundant signals into channels A, B, and C is maintained for flight control functions in the FCI. Separate connectors are used for each channel and three isolated power sources are available.

Signal conditioning electronics for the analog sensor signals and servo drive electronics for the servo commands to elevator, aileron, and stabilizer trim servos are located in the FCI pallet. Triplex monitoring of analog signals is provided through pushbutton selection. Test inputs can be inserted from a system test panel in the pallet, and test functions in radio altimeters, ILS receivers, and rate gyros can be remotely exercised and monitored from the pallet. Loss of valid signals from sensors used by the flight control system is flagged on a display panel.

TABLE 7-1.—INPUT SENSORS

Type	Sensor	Number of sensors	Flight Critical
Analog:	Radio altimeter No. 1	1	X
	Radio altimeter No. 2	1	X
	Vertical acceleration	3	X
	Altitude rate	3	X
	Glide slope beam	3	X
	Localizer beam	3	X
	Pitch attitude	3	X
	Roll attitude	3	X
	Pitch rate	3	X
	Roll rate	3	X
	Column force	3	X
	Wheel force	3	X
	Pitch q	3	X
	Elevator position	2 (plus model)	X
	Aileron position	2 (plus model)	X
	Auto stabilizer trim potentiometer	1	X
	Roll q	3	X
	Pitch servo amplifier	1	
	Roll servo amplifier	1	
	Delta track	3	X
Digital	Vertical path command	1	
	Horizontal path command	1	
	Delta track "A"	1	X
	Delta track "B"	1	X
	Delta track "C"	1	X
	Groundspeed	3	X
	Delta track	3	X

The FCI pallet houses a third ILS receiver with its control head and a complete set of triplex rate gyros for pitch and roll. A power distribution panel providing the triplex ac and dc power is located at the bottom.

The third independent power source requires the use of the APU generator, figure 7-9. With the APU shut down, the third power reference source is powered from the No. 1 airplane bus.

A or B system hydraulics are selected with a switch on the AFCS engage panel. The servo configuration is always nonredundant (see fig. 7-3) and pitch and roll servo models in the FCI electronics, driven by the C channel computer interface unit, are used for servo monitoring.

The primary mode of operation with the ICPS is attitude control wheel steering (ATT CWS). Once the ATT CWS mode is engaged, the pilot can select other automatic flight control modes through the AGCS mode select panel (MSP, fig. 7-10). The MSP controls the ICPS and the flight-critical control modes, through the four lower left-hand backlighted pushbuttons on the panel face. Behind each of these four buttons are triply redundant, electrically isolated switches for communication with each computer interface unit (CIU). Each of the three CIUs in turn lights one bulb in the appropriate mode button, making the mode annunciation triply redundant for flight-critical functions, which also include autoland (AUTO + LAND) and velocity vector control wheel steering (VEL CWS).

Selecting the AUTO mode on the MSP, except in connection with LAND, accesses noncritical flight control modes performed by a navigation-guidance computer.

7.3 FLIGHT TEST RESULTS

7.3.1 Failure Monitoring

The intent of the supplemental flight test program with respect to the sensor signal selection/failure detection (SSFD) functions was to observe and evaluate the system performance under various conditions. The sensor SSFD system was monitored throughout the entire flight program so that an evaluation could be made of the system throughout the flight regime in all autopilot modes, in all types of weather conditions, and in as many operational conditions as could be typified as normal airline operation.

The sensor SSFD system is, generally speaking, a full-time monitoring system (with the exception of the ILS beam signals); therefore, performance of the system, in all flight conditions and autopilot modes is of interest since the airplane dynamics, and hence sensor signal characteristics, are somewhat dependent upon these conditions.

While weather conditions are not a controlled parameter, specific interest is directed to sensor SSFD performance in turbulence since the short-term characteristics of certain sensors such as rate gyros, accelerometers, altitude rate sensor, and elevator and aileron feedback sensors are manifested during turbulence. The sensor SSFD system performance was monitored during the ADEDS flight testing and hence was subjected to what could be termed a cross sectional view of operational procedures. The system was monitored for a total of 76 flight hours and the nature of all failures was uncovered.

Testing the failure detection system for known failures (programmed failures) was limited. The most critical type of failures (beam ramp failures) was tested, and the results show that failure detection is sufficient to prohibit undesirable maneuvers.

Insufficient data were collected to determine the optimal programmed maneuver (fly-off) required to assure detection of a latent passive failure in an ILS receiver. The fly-off program tested was marginal and the traces show, at most, the typical minimum maneuver required to test for passive ILS receivers.

7.3.1.1 Description of Sensor SSFD for Flight Test

The basic sensor signal selection/failure detection function of the ICPS is described in section 5.1.1.3. This SSFD configuration contains six constants that can be selectively programmed to fit the characteristics of each particular type of sensor. Table 7-2 provides the detailed information of the final configuration of the programmed constants on all multiple sensor inputs used during the supplemental flight test program. Table 7-1 provides the status (flight-critical or noncritical) as well as the list and number of each type of sensor used in the system.

Delta tracks A, B, and C are listed to show the mode and ROM select program for these inputs. The track angle signal selection/failure detection scheme is discussed in section 7.3.1.1.1.

Although radio altitude No. 1 and No. 2 constitute single sensor inputs, failure detection is accomplished by cross-signal monitoring and the constants are programmed in dual on these inputs. Section 7.3.1.1.2 provides additional data on the radio altimeter monitoring scheme.

Special treatment of the control surface servo actuator monitor for the flight test program is discussed in section 7.3.1.1.3.

7.3.1.1.1 Track Angle Monitoring and Signal Selection—The track angle error (ΔTKA) monitoring system is shown schematically in figure 7-11. Each ΔTKA reference signal (A, B, or C) is brought into the computer and stored in two different addresses. One address (algorithm time) is common to all three inputs that go through the normal process of median selection, identified as $\Delta\psi_{GM}$. The monitor thresholds for this input are set relatively large to allow for maximum differences anticipated from asynchronous Schuler oscillations. $\Delta\psi_{GM}$ is common to all computers when not LOC ENG (i.e., in cruise modes); therefore, channel differences permitted by the large thresholds are not considered catastrophic. Prior to LOC ENG, $\Delta\psi_{GA}'$, $\Delta\psi_{GB}'$, and $\Delta\psi_{GC}'$ are all identical. All three of these signals are routed external to the computer to an input identified by $\Delta\psi_{GM}'$, which provides additional selection and monitoring. In the cruise mode, this additional process is redundant and serves no useful purpose except when a computer or interface unit should fail.

When localizer is engaged (LOC ENG), the computer input reference for each channel is then incrementally switched to its corresponding ΔTKA input such that

$$\Delta\psi_{GA}' = \Delta\psi_{GM} \text{ at LOC ENG} + \text{incremental changes in } \Delta\psi_{GA} \text{ after LOC ENG.}$$

The anticipated difference between any two prime track angle errors (e.g., $\Delta\psi_{GA}' - \Delta\psi_{GB}'$) would then be the maximum Schuler rate times $2\Delta T$, where ΔT is the length of the approach time. The threshold level required to avoid nuisance trips for this latter case is reduced by a factor of 4 to 5, compared to the cruise situation where time is less bounded, permitting tighter monitor thresholds to protect against large track angle differences that, during approach, could prove to be catastrophic. In summary, the criterion for setting the

TABLE 7-2.--SIGNAL SELECTION AND FAILURE DETECTION FROM DATA

Parameter	Scale Mu/unit	Threshold 1		Threshold 2		Time const 1		Time const 2		TD 1 Count	TD 2 Count	Input address	Prom address	Stored data		
		Eng units	% FS	Eng units	% FS	τ sec	ω rad	τ sec	ω rad					LS8	MSB	678
Roll ϕ	40.96	1.035°	1.407	2.54°	5.08	0.784	1.27	1.57	0.635	16	16	\$ 30	60	00	100	011
Roll rate $\dot{\phi}$	102.4°/sec	1.95°/s	9.77	1.87°/s	9.38	3.14	0.318	3.14	0.318	16	16	\$ 34	68	00	101	101
Loc beam η	409.6	0.293°	5.86	0.195°	3.91	1.57	0.635	1.57	0.635	16	16	\$ 26	52	00	100	100
TKA $\Delta\psi/G$	45.51	1.408°	3.13	4.57°	10.16	1.57	0.635	0.098	10.2	64	64	\$ 54	108	00	000	100
Aileron δ_a	204.8	0.391°	3.91	0.312°	3.12	6.29	0.158	0.784	1.27	16	16	\$ 44	88	00	011	110
Vertical acceleration \ddot{h}	102.4/fps	0.47 fps ²	2.35	0.234	1.17	3.146	0.32	12.58	0.08	4	64	\$ 20	40	00	111	101
Altitude rate \dot{h}	25.6/fps	5.62 fps	7.03	1.25	1.56	12.58	0.08	6.29	0.16	4	64	\$ 22	44	00	110	111
GSE beam error β_E	2048/deg	0.101°	10.16	0.023°	2.34	12.58	0.08	6.29	0.16	4	64	\$ 24	48	00	110	111
Pitch attitude θ	102.4/deg	2.032°	10.16	0.468°	2.34	12.58	0.08	6.29	0.16	4	64	\$ 28	56	00	110	111
													57	00	101	100
													58	00	110	000
													59	01	100	010

MSB
678
LS8
12 345 678

TABLE 7-2.—Continued

Parameter	Scale Mu/unit	Threshold 1		Threshold 2		Time const 1		Time const 2		TD 1 Count	TD 2 Count	Input address	Prom address	Stored data			
		Eng units	%FS	Eng units	%FS	τ sec	ω rad	τ sec	ω rad					LS8	12	345	078
Pitch rate $\dot{\theta}$	204.8/deg	0.5°/s	5.07	0.47°/s	1.17	12.58	0.08	0.16	6.29	0.16	4	64	S 32	64	00	110	111
Elevator δ_e	204.8/deg	0.70°	7.03	0.312°	3.12	3.14	0.32	5.1	0.196	16	32		S 42	65	01	011	000
Groundspeed V_{GS}	5./kt	25.6 kt	0.39	25.6 kt	0.39	12.58	0.08	0.16	6.29	4	64		S 86	172	00	110	111
														173	01	000	000
Roll q Rq	2048 full scale	1.01 v	10.16	4.96 v	49.61	1.57	0.635	2.54	0.393	4	64		S 48	174	x1	000	000
														175	01	100	010
Pitch Q Pq	2048 full scale	1.16 v	10.16	4.96 v	49.61	1.57	0.635	2.54	0.393	4	64		S 40	96	00	010	100
														97	00	101	100
Wheel force F_w	61.44/lb	3.38 lb	10.16	16.54 lb	49.61	1.57	0.635	2.54	0.393	4	64		S 38	98	x1	111	111
														99	01	100	010
Column force F_c	61.44/lb	3.38 lb	10.16	16.51 lb	49.61	1.57	0.635	2.54	0.393	4	64		S 36	80	00	010	100
														81	00	101	100
TKA $\Delta\psi_G$	18.2/deg	4.92°	2.73	4.92°	2.73	0.098	10.2	S 88	76	00	010	100	77	00	101	100	
									78	x1	111	111					
TKA $\Delta\psi_{GA}$	182/deg	—	49.61	—	49.61	12.58	0.079	1024	12.58	1024	1024	S 80	79	01	100	010	
													72	00	010	100	
													73	00	101	100	
														74	x1	111	111
														75	01	100	000
														76	00	000	100
														177	01	110	000
														178	x1	110	000
														179	00	100	010
														1	10	111	111
														161	11	111	111
														162	x1	111	111
														163	10	101	010

TABLE 7-2. — Concluded

Parameter	Scale		Threshold 1		Threshold 2		Time const 1		Time const 2		TD 1 Count	TD 2 Count	Input address	Prom address	Stored data				
	Mu/unit		Eng units	%FS	Eng units	%FS	τ sec	ω rad	τ sec	ω rad					LSB	12	345	078	MSB
TKA $\Delta\psi$ GB	182/deg	—	—	49.61	—	49.61	12.58	0.079	12.58	0.079	1024	1024	\$ 82	164	1	11	111	111	A
															11	111	111	111	A
															x1	111	111	111	A
TKA $\Delta\psi$ GC	182/deg	—	—	49.61	—	49.61	12.58	0.079	12.58	0.079	1024	1024	\$ 84	168	1	01	111	111	A
															11	111	111	111	A
															x1	111	111	111	A
R/Ait	2048/1500 ft	99.6'	99.6'	6.64	—	6.64	1.57	0.635	0.392	2.54	4	64	\$ 16	32	00	010	100	100	A
															01	000	100	100	A
															x1	000	100	100	A
R/Ait	2048/1500 ft	99.6'	99.6'	6.64	—	6.64	1.57	0.635	0.392	2.54	4	64	\$ 18	36	00	010	100	100	A
															01	000	100	100	A
															x1	000	100	100	A

Chan	Alg time		
	80	82	84
	A	10	11
B	01	10	11
C	11	01	10

¹BIT 1 and 2 as shown for channel A only.

²BIT 1 and 2 program according to channel.

monitor thresholds for $\Delta\psi_{GM}$ is determined by the airplane performance in a failure mode rather than the avoidance of nuisance trips due to high Schuler rate differences.

7.3.1.1.2 *Radio Altimeter Monitoring and Signal Selection*—The additional failure monitor provided by the ICP-723 for the radio altimeters (R/A) is shown in figure 7-12. The basic R/A monitor is provided in the R/A itself, which provides a flag warning (R/A VALID) that is also used in the ICP-723 (fig. 7-13). The failure monitor provides additional protection in case an R/A input is lost at the ICP interface and yet the R/A indicates a valid input. This additional monitoring is conveniently provided since the algorithm time is already dedicated to the altimeter input. This scheme utilizes cross-channel monitoring between R/A No. 1 and No. 2. The additional monitoring does reduce R/A No. 1 and No. 2 reliability since A CIU and B CIU are within the signal input loop. Table 7-3 shows the failure modes where second fail could occur because of A or B CIU failure and a nuisance flag on the alternate R/A.

TABLE 7-3.—SYSTEM/RADIO ALTIMETER FAILURE STATUS

Loss of No. 1 R/A valid	Loss of No. 2 R/A valid	A CIU fail	B CIU fail	C CIU fail	Signal selected at No. 1 input	Signal selected at No. 2 input
0	0	1	0	0	Average	No. 2
0	0	0	1	0	No. 1	Average
0	0	0	0	1	No. 1	No. 2
1	0	0	0	0	No. 2	No. 2
0	1	0	0	0	No. 1	No. 1
1	0	1	0	0	No. 2	No. 2
1	0	0	1	0	Second F	Second F
1	0	0	0	1	No. 2	No. 2
0	1	1	0	0	Second F	Second F
0	1	0	1	0	No. 1	No. 1
0	1	0	0	1	No. 1	No. 1

Figure 7-13 shows the signal selection path in any given CIU for the R/A input. R/A No. 1 is wired to No. 1 input (algorithm time 16) on A and B CIUs and to No. 2 input (algorithm time 18) on C CIU. R/A No. 2 is wired to No. 2 input on A and B CIUs and to No. 1 on C CIU. Consequently, in any given CIU on either No. 1 or No. 2 input, the C input will originate from the alternate R/A.

7.3.1.1.3 *Servo Monitoring*—The original servo monitor configuration used at the beginning of the flight test program is shown in figure 7-14. The servo position signal was compared to the servo model feedback as shown in figure 7-15. Thus, the failure monitor indications were either no failure or a C channel failure. A C channel failure simply indicated that there was disagreement between the position feedback and the servo model.

Because of nuisance failures in the autopilot disengage mode and the inability to reset the failure monitor in the disengage mode when the position feedback was a nonzero value greater than the trip threshold, the configuration was changed to that shown in figure 7-16. This latter configuration bypassed the ICP computer in the disengage mode (allowing for computer reset at any time) and also increased the model servo loop gain by a factor of 10 to reduce the lag between the model and the position feedback to avoid nuisance failures in the disengage mode.

A second tracking relay K2 (operated on AEE₂ logic, fig. 7-16) is incorporated to provide monitoring for relay K1 to ensure switchover to the ICP at AEE to provide active servo monitoring in the engaged mode. Otherwise, a latent failure in relay K1 could occur in the AEE position, and the monitor would never trip.

7.3.1.2 Evaluation Criteria

Many of the failure monitor parameter and signal selection configuration studies of the more critical sensors were conducted with computer simulations prior to the flight test program. The general guidelines adopted during these studies to arrive at a flight test configuration were that the signal selection and failure monitor system should be such that:

- 1) A sensor failure shall be detected (a failed sensor is defined as one that will not pass a bench test when tested according to the vendor specifications).
- 2) A sensor failure during an approach shall not put the airplane in a critical attitude, cause an excessive path deviation, or cause excessive g maneuvers. (G maneuvers due to failures were kept below 0.1 g.)
- 3) Nuisance failure indications shall be kept to a minimum.

Nuisance failure indication evaluation during flight test began with observing failure indications and determining the cause or contributing factor leading to them. Changes could then be made in system configuration or failure monitor thresholds for further evaluation. If failure indication of a nuisance nature could not be kept to an acceptable level while adhering to safety guidelines, then the system would be deemed inadequate.

Failure detection capability was to be evaluated from two aspects:

- 1) Would the failure monitor detect a known failure?
- 2) Was the airplane maneuver safe in all respects during a critical failure injection and subsequent automatic autopilot disconnect?

Adequacy or need for programmed maneuvers to test for passive failures was to be on the basis of failure detection of known sensor failures during such a maneuver and reviewing the consequences if there was no detection of failure.

7.3.1.3 Test Procedure

No special procedure was adopted to test for nuisance failures. The system was monitored throughout the flight test program in all modes of operation, flight conditions, and environmental conditions encountered. Flight notes were maintained on all failure indications observed during the flight test.

Fly-off maneuvers for both the pitch and lateral axes were programmed into the respective control laws during the ILS approach to ensure detection of passive ILS receiver failures.

The localizer and glide slope receiver monitors were inhibited when not in autoland arm state to avoid nuisance failure indications due to intermittent losses of receiver valids when out of range of the ILS transmitters. If the autoland mode was armed prior to a predetermined minimum beam error threshold, the fly-off maneuver was inhibited for the respective axis. If the autoland mode was armed below the minimum threshold, the fly-off would be initiated for the corresponding axis. The procedure for testing the adequacy of the fly-off maneuver was to fly the airplane manually below the minimum thresholds (as near track center as possible in both axes), fail one ILS receiver by pulling the circuit breaker, and then select the autoland mode. The fly-off maneuver would then be initiated, and if the maneuver was adequate, a first failure indication would be caused by the intentionally failed receiver.

One failure mode that could conceivably be critical occurs when one receiver has failed, the signal selector averages the remaining two beam error signals, and a second receiver ramps off at a rate undetected by the washout filter threshold detector. The airplane will deviate from the beam center at a rate equal to the ramp failure rate and will continue to do so until the autopilot is disconnected and the pilot takes over.

The procedure adopted to test this case was to capture the beam and allow the airplane to acquire beam center, insert a hardover failure in one channel in the axis to be faulted, and then insert a low ramp rate such that the second failure detection would occur at approximately flare altitude. The second failure indication would automatically disconnect the autopilot, thereby reverting to a manual takeover. Camera data were taken during the last 400 ft of altitude to touchdown to record path deviation. The failure monitor was then evaluated on the basis of maximum path deviation encountered during the test case.

7.3.1.4 Evaluation Results

Significant data acquired during the supplemental flight test program for evaluating the SSFD system are provided in table 7-4 and figures 7-17 through 7-20.

Attempts to evaluate the system were made from three aspects:

- 1) Nature and frequency of nuisance problems due to low thresholds

TABLE 7-4.—FAILURE INDICATIONS ANALYSIS

Failure indication	Frequency per flight	Cause/contributing factors	Corrective action
δ_e δ_a	Numerous in early part of flight test program	1. Due to mechanization failure indications could not be reset whenever steady state error (surface trim position) was greater than monitor threshold. 2. Low lag threshold.	1. Remechanized the servo monitors to bypass the ICP computer when autopilot disengaged to allow fast synchronization and avoid problem associated with reset. 2. Doubled the lag threshold.
δ_e δ_a	1.5/flight Once during latter part of flight test program	3. Monitor servo loop gain too low when autopilot in disengage mode. Elevator command limit greater than autopilot authority limit. Cam-out occurred in turbulence at cruise. Cam-out occurs in flare. Turbulence causing high roll rates up to 9.57°/sec. Roll servo model rate limit improperly adjusted.	Reprogram the elevator command limit in the software to stay within the autopilot authority limit in all modes. None unless nuisance rate becomes higher.
First fail	0.9/approach	Gradient error between two localizer receivers exceeded threshold. Marginal threshold.	TBD. Receivers have been returned to vendor for adjustment and calibration.
β	0.95/approach	G/S receiver A, on a separate antenna, receivers B and C, on the same antenna. G/S error differences magnified near the G/S transmitter.	G/S failure indications to be inhibited after flare and during go-around.
VGS	2.0/flight	Legitimate failures. Intermittent loss of accelerometer output affecting groundspeed output from INS. Hardware malfunction.	Hardware corrected
Δ TKA	3.0/flight	Legitimate failures. INS operating out of specification.	Vendor Action
HDOT	2.0/flight	"A" CADC on one pitot/static source. "B" and "C" CADC on the same pitot/static source. A pneumatic load different from B and C causing sensor output on A to have greater lag. Problem manifested with high transient altitude rates on initiating climb or descent maneuvers.	TBD
R/A	3.0/flight in early part of flight test program	Legitimate failures. Sensor failures were coincident with radio altimeter flag indications. Intermittent failures of short duration.	Problem no longer exists.
F_c	Once during flight test program	Legitimate failure. P.C. card in the FCI pallet improperly seated.	No further action required.
θ	1.0 for each climb or descent maneuver	INS gradient error, one INS operating out of specification.	INS returned to Vendor

- 2) Capability of the failure detection system to detect critical failures (a critical failure being one that could lead to a catastrophic event if not detected soon enough)
- 3) The need for and adequacy of programmed fly-off maneuvers during an autoland

Table 7-4 provides a list of all the failure indications encountered during the flight test program. All failure indications are listed because their appearance during flight test took on the nature of a nuisance failure whenever their immediate cause was not readily apparent. Even when the cause of a failure indication was known, the frequency of occurrence would give rise to a degree of annoyance.

Continued monitoring of failure modes led to the correlation of occurrences and contributing factors of some sensor failures shown in the table, while others were not correlated until recorded flight test data were reviewed subsequent to the flight test program.

The most critical failure conceivable in an autoland mode is the ILS receiver ramp failure. The data resulting from tests of such failures are shown in figures 7-17 and 7-18. Each figure is marked to show:

- 1) Beam ramp failure rate
- 2) The altitude the failure was injected
- 3) The point at which second failure (autopilot automatic disconnect) occurred

Evaluation of the fly-off maneuvers programmed into the autoland control laws to check for ILS receiver passive failures was based on two factors:

- 1) Did the fly-off maneuvers detect a passive receiver in every instance?
- 2) Was the airplane maneuver excessive or did it cause any adverse residual effects?

Figures 7-19 and 7-20 provide pertinent traces showing the typical maneuver during and after a fly-off. The point in time where the failure was detected for each axis is also indicated.

7.3.1.5 Failure Monitoring Flight Test Conclusions

The high degree of correlation between cause and failure indications shown in table 7-4 provides a good basis for determining the corrective measures required. As can be seen from the table, there are only three suspect sensors that may provide nuisance failures in the future.

- 1) The localizer receiver gradient error failure indication should be eliminated. Provisions exist in the threshold levels to permit a 12% gradient error between two receivers at 2.5° beam error. This allows for a worst case error of 5% for each

receiver and 1% for each interface card. If the receivers and their corresponding interface equipment are kept within performance specification, no problem should exist. If nuisances manifest themselves in this area, a more specific cause for gradient error needs to be determined.

- 2) The aileron monitor threshold is apparently marginal and subject to nuisance failures in turbulence when experiencing high roll accelerations where the phase difference between the model and the aileron gives rise to dynamic errors that are detected by the washout filter threshold. Increasing the washout filter threshold can be the solution. The upper limit was previously dictated by the loss of feedback on the interface card. Loss of feedback involves the loss of one specific resistor (feedback resistor) on the servo loop printed circuit card. An alternate means for monitoring the integrity of this resistor path could be developed to allow raising the washout threshold.
- 3) Altitude rate (HDOT)—the threshold levels for this sensor were chosen primarily on the basis of nuisance failures; however, the true nature of the differences was not properly understood prior to the flight test program. The consequences of raising the dynamic threshold to a level above nuisance threshold are not readily apparent. The most critical failure of HDOT would be during or immediately prior to flare. The nature of the failure could be similar to that of a beam ramp failure (i.e., HDOT ramp failure). HDOT divergence would be critical during flare when the glide slope beam integrator is removed. Therefore, raising the threshold levels is not recommended without further simulation studies. It may be necessary to change the algorithm for the HDOT sensor input if the high altitude nuisance avoidance requirements are incompatible with the flare mode failure protection. The problem may be resolved from the sensor standpoint either by balancing the loads or otherwise compensating for the pitot static source differences.

Figures 7-17 and 7-18 show that the most critical types of system ramp failures (low ramp rate) are all detectable without causing undesirable system errors or maneuvers.

An insufficient number of fly-off maneuvers were tested with passive receiver failures to determine the effectiveness of the maneuver. It is obvious from the traces (fig. 7-19 and 7-20) that the lateral maneuver is marginal at most and the vertical fly-off was insufficient in the first case.

From the aileron, roll, and track angle traces, it can be seen that there is a residual 10-sec low amplitude oscillation that is not otherwise present. The glide slope trace appears to manifest the same tendency to oscillate; however, these data were taken where the true pattern of the ILS path was not generally without clutter. The data are insufficient to determine effectiveness or objectionableness of the fly-off maneuvers.

7.3.2 Autoland Performance

The autoland performance experiments during the supplemental flight test period had the following purposes:

- 1) Initial adjustment of pitch, roll, and thrust control laws for the automatic ILS capture, track, and touchdown phases, developed on the simulator, to achieve optimum performance with the actual airplane in flight
- 2) Collection of autoland performance data for a wide range of environmental and procedural conditions to form a data basis on which to judge system performance during normal operation

Eighteen performance evaluation approaches were flown against a ground-based camera data system with a limited number of environmental conditions encountered. A satisfactory configuration of the flare control law was not established before the end of the supplemental flight test period. Therefore, data on longitudinal touchdown dispersion were inconclusive.

Satisfactory ILS tracking performance was evidenced in both pitch and roll with lateral touchdown dispersion acceptable despite the presence of a lightly damped lateral path mode.

7.3.2.1 Configuration Description

The configurations of the pitch, roll, and airspeed autoland control laws arrived at after the initial checkout and adjustment flights are shown in block diagram form on figures 7-21 through 7-23.

7.3.2.1.1 Pitch Autoland Control—If properly armed for capture, the autoland function, figure 7-21, assumes control in pitch when the vertical beam sensor detects less than $\pm 0.108^\circ$ glide slope error signal level. The balance between a composite altitude rate signal and the glide slope signal, the magnitude of which decreases as the aircraft approaches the center of the beam, forces a pitch maneuver into the glide slope beam, nose down or nose up depending upon the capture being from below or from above beam center. The beam tracking mode is engaged 10 sec after the initiation of the capture. A composite beam error signal is formed through a 15-sec complimentary filter on the raw beam error plus a computed beam error rate, formed by the composite altitude rate minus a nominal sink rate compensated for actual groundspeed. The raw beam error signal is gain scheduled with radio altitude to compensate for beam convergence as the aircraft approaches the transmitter. A parallel integral path operates on the beam error signal.

The composite altitude rate signal is formed by complimentary filtering barometric altitude rate and vertical acceleration from the INS with a time constant of 16 sec.

The path error signal from the above computation is summed with vertical acceleration and washed out pitch rate to form the elevator command for the ILS capture and track mode.

The automatic flare mode is designed to decrease and maintain the sink rate to 2 ft/sec from 5 ft of altitude to touchdown. The flare is initiated at an altitude that depends on the actual sink rate; the 10-ft/sec flare starts at 40 ft.

7.3.2.1.2 Roll Autoland Control—The lateral ILS capture mode, figure 7-22, is initiated when the lateral beam sensor detects less than $\pm 2^\circ$ of localizer deviation. Beam error and track angle error relative to the runway centerline then control the aircraft to the center of the beam flying a track parallel to it.

The capture mode is terminated and the on-course mode is entered when beam error, computed beam rate, and bank angle are all within specific limits. In the on-course mode, a composite beam error signal is formed through complimentary filtering of beam error and track angle error, which is proportional to beam rate. The complimentary filter time constant is 20 sec.

An integral path for the beam error signal is also active in the on-course mode. Radio altitude is used for gain scheduling the beam error signal to compensate for beam convergence as the aircraft approaches the transmitter. A variable limit on the beam signal, to limit the effect of a ground station hardover failure, is also scheduled as a function of radio altitude.

The composite beam error, the straight beam error integral, and track angle error form the bank angle command, which is rate- and magnitude-limited to $4^\circ/\text{sec}$ and 10° , respectively, in the on-course mode. During flare, the bank angle command is further limited to 5° .

7.3.2.1.3 Airspeed Autoland Control—The airspeed control function, figure 7-23, captures and holds the calibrated airspeed selected by the pilot on the mode select panel. At 30 ft of altitude an automatic flare mode is initiated to retard the throttles with a variable rate, dependent upon the amount of overspeed as measured by the computed angle of attack at 30 ft.

The longitudinal component of wind shear is detected by comparison between true airspeed and integrated longitudinal acceleration. A bias signal is summed to the acceleration feedback to offset the unwanted effect of the inertial feedback in the presence of wind shear.

7.3.2.1.4 Autoland Evaluation Procedure—The autoland performance was evaluated for beam tracking accuracy, touchdown accuracy, and other dynamic characteristics related to ride quality and pilot acceptance.

- 1) Beam tracking accuracy: both the FAA requirements of AC 120-29 and Boeing-developed maneuver criteria were applied to the beam tracking performance.
- 2) Touchdown accuracy: the Boeing-developed footprint criteria were applied to airplane position and rate behavior in pitch and roll from 100 ft altitude to flare and touchdown, respectively. FAA touchdown criteria are defined by AC 20-57A.

- 3) Other characteristics: dynamic characteristics, such as control activity, maneuver and tracking smoothness, touchdown behavior, and mode switching transients, were subjected to pilot evaluation.

To evaluate system performance against the above criteria, standard coupled ILS approaches to touchdown, with autothrottle control of airspeed, were conducted. Metric cameras were used to record airplane position every other second from approximately 400 ft of altitude to touchdown. Through postflight data processing, position data from the metric cameras were mixed with inertial data recorded onboard to yield continuous traces of airplane position relative to the runway through touchdown.

Table 7-5 summarizes 18 performance approaches to the camera instrumented runway. All of these approaches were conducted with 40° flaps, autothrottle, and yaw damper on with the fully redundant sensor configuration.

Localizer capture conditions covered intercept angles from 90° to less than 15° and capture ranges from 4 to 9 miles from the runway threshold.

Reported wind conditions at the airport during the listed tests included 8-kt headwind components, 10-kt tailwind components, and 8-kt crosswind components. Strong windshear conditions on approach were not encountered during these test conditions.

7.3.2.1.5 Flight Test Evaluation Results—Table 7-5 summarizes test conditions, touchdown points, and maneuver equation averages (MEA) between 350 and 50 feet of altitude for the 18 performance approaches. The longitudinal touchdown reference is the location of the glide slope antenna, and lateral reference is the runway centerline.

During the first eight approaches, the flare control law was still being adjusted. The longitudinal touchdown dispersion is therefore not representative of one single configuration. For the last 10 approaches, the 1σ value for the longitudinal touchdown was 331 ft about a mean of +237 ft.

The lateral 1σ value for all 18 approaches was 8.0 ft about a mean of +2.7 ft.

Figures 7-24 and 7-25 show the last 200 sec of raw sensor data from four representative approaches, including one very close-in capture 4.5 miles from the runway with 60° localizer intercept, approach No. 3. These plots were compiled by a computer using an automatic scaling feature; the scales therefore vary between similar plots. The FAA tracking requirements on localizer and glide slope beams are indicated on these plots.

For the conditions evaluated, the beam tracking capabilities in both roll and pitch are well within the FAA-required 35/25 μ A and 35 μ A/12 ft, respectively. The MEAs from 350- to 50-ft altitude shown in table 7-5 also indicate substantial margins to the Boeing-proposed maneuver equation criteria, which require less than 60-ft lateral deviation below 100 ft of altitude and less than 16-ft vertical deviation below 180 ft.

Despite the presence of lightly damped modes, as evidenced by the reduced data, the performance traces of the 18 approaches are well collected in the center of the footprint plots.

Although only a limited number of conditions have been tested, data indicate that the lateral touchdown dispersion is acceptable but can be further improved through optimization of the lateral path damping.

TABLE 7-5.—SUMMARY OF AUTOLANDS

	Localizer capture		Reported wind component		Touchdown point		MEA	
	Angle (deg)	Distance (mi) from runway	Cross L (+) R (-)	Head (+) Tail (-)	Lon (ft)	Lat (ft)	Lon (ft)	Lat (ft)
1	30	8	(n)	-8	253	-1.4	2.2	4.1
2	45	4	(n)	-10	505	9.9	3.1	12.1
3	60	4.5	(n)	-9	676	-2.2	1.7	8.3
4	15	4	-3	(n)	454	6.4	2.8	9.7
5	30	6	-3	(n)	405	3.9	2.9	7.4
6	45	7	-3	(n)	1040	-5.1	3.3	5.1
7	60	8	-3	(n)	1344	9.0	4.3	6.3
8	90	7	-3	(n)	532	-5.6	5.3	8.5
9	30	8	+2-5	(n)	-128	7.8	2.8	5.8
10	30	9	+2-5	(n)	449	-2.6	3.7	6.3
11	15	4	+5	(n)	- 28	5.4	4.9	13.7
12	30	9	+1	-1	-145	16.2	2.6	10.1
13	30	9	+7	-10	- 23	-5.1	4.5	9.9
14	30	9	+13	-13	- 47	-12.5	3.7	5.4
15	30	8	+8	+8	756	8.0	4.2	4.9
16	45	8	+8	+8	450	-5.4	3.6	7.7
17	45	8	+8	+8	388	2.7	4.4	2.8
18	30	8	-	+4	701	18.4	4.5	8.5

REFERENCES

1. *Redundant Flight-Critical Control System Evaluations (Analog and Digital Systems Descriptions)*, D6-60286-1, The Boeing Company, report FAA-SS-73-2-1.
2. *Redundant Flight-Critical Control System Evaluations (Analog and Digital Systems Failure Analyses and Preflight Test Designs)*, D6-60287, The Boeing Company, report FAA-SS-73-3.
3. *SST Longitudinal Control System Design and Design Processes (Hardened Stability Augmentation Design)*, D6-60285, The Boeing Company, report FAA-SS-73-1, June 1973.
4. *ICP-723 Incremental Computer User's Manual*, ACS 10,435, General Electric Aircraft Equipment Division.

APPENDIX A

SIGNAL SELECTION/FAILURE DETECTION FUNCTION CHARACTERISTIC ANALYSES

A.1 SIGNAL SELECTION NOISE CONSIDERATIONS

From the statistical theory for ordered random variables we find that:

$$E[X_{(i)}] = \frac{n!}{(i-1)!(n-i)!} \int_{-\infty}^{\infty} x f(x) [F(x)]^{i-1} [1-F(x)]^{n-1} dx$$

$$E[X_{(i)}^2] = \frac{n!}{(i-1)!(n-1)!} \int_{-\infty}^{\infty} x^2 f(x) [F(x)]^{i-1} [1-F(x)]^{n-1} dx$$

$$E[X_{(i)}X_{(j)}] = \frac{n!}{(i-1)!(j-i-1)!(n-j)!} \int_{-\infty}^{\infty} \int_{-\infty}^y xy f(x) f(y) [F(x)]^{i-1} [1-F(y)]^{n-j} [F(y)-F(x)]^{j-i-1} dx dy$$

where

$$f(x) = \frac{1}{\sqrt{2\pi} \sigma_s} e^{-\frac{x^2}{2\sigma_s^2}}$$

$$F(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi} \sigma_s} e^{-\frac{t^2}{2\sigma_s^2}} dt$$

and $X_{(1)}, X_{(2)}, \dots, X_{(n)}$ are observations drawn from a normal distribution with a standard deviation of σ_s and arranged in ascending order of magnitude.

For $n = 3$ (triplex system) the variance $(E[X_{(i)}^2])$ of

$$X_{(1)} = 0.559467 \sigma_s^2$$

$$X_{(2)} = 0.448671 \sigma_s^2$$

$$X_{(3)} = 0.559467 \sigma_s^2$$

the covariances ($E [X_{(i)}X_{(j)}]$) of

$$X_{(1)}X_{(2)} = 0.275665 \sigma_s^2$$

$$X_{(1)}X_{(3)} = 0.164868 \sigma_s^2$$

$$X_{(2)}X_{(3)} = 0.275665 \sigma_s^2$$

and the means ($E [X_{(i)}]$) of

$$X_{(1)} = -0.84628 \sigma_s$$

$$X_{(2)} = 0.0 \sigma_s$$

$$X_{(3)} = 0.84628 \sigma_s$$

Since the variance is the square of the standard deviation

$$\sigma_{X_{(1)}} = 0.747975 \sigma_s$$

$$\sigma_{X_{(2)}} = 0.669829 \sigma_s$$

$$\sigma_{X_{(3)}} = 0.747975 \sigma_s$$

The density functions of the standard (X_s) and the normal approximations to the lowest extreme ($X_{(1)}$), middle or median ($X_{(2)}$), and the largest extreme ($X_{(3)}$) are qualitatively illustrated in figure A-1.

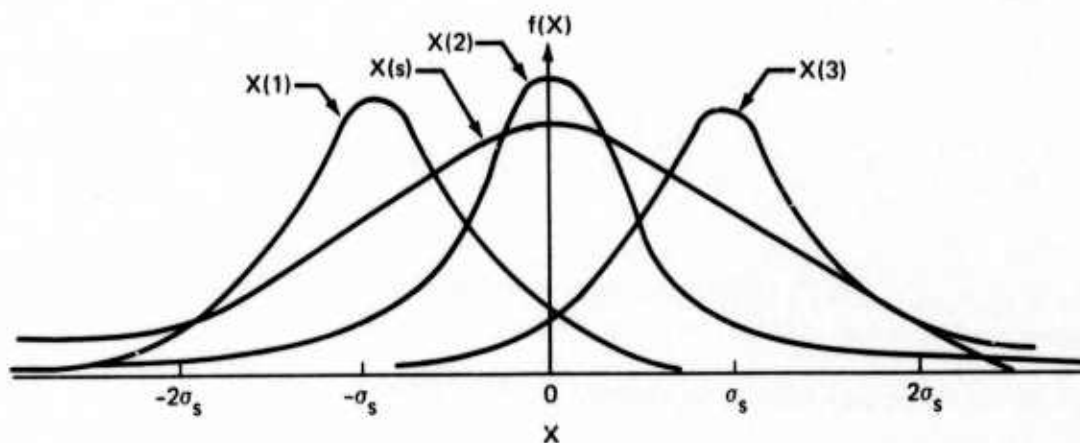


FIGURE A-1.—DENSITY FUNCTIONS OF STANDARD, EXTREMES, AND MEDIAN

The variance of the average of the three (σ_{A3}^2) can be found by considering

$$\begin{aligned} E \left[\left(\frac{X_{(1)} + X_{(2)} + X_{(3)}}{3} \right)^2 \right] &= \frac{1}{9} \left[E(X_{(1)}^2) + 2E(X_{(1)}X_{(2)}) \right. \\ &\quad + E(X_{(2)}^2) + 2E(X_{(2)}X_{(3)}) \\ &\quad \left. + E(X_{(3)}^2) + 2E(X_{(3)}X_{(1)}) \right] \\ &= 0.333333 \sigma_s^2 \end{aligned}$$

and the mean (μ_{A3}) by considering

$$E \left[\frac{X_{(1)} + X_{(2)} + X_{(3)}}{3} \right] = \frac{1}{3} \left[E(X_{(1)}) + E(X_{(2)}) + E(X_{(3)}) \right] = 0.0 \sigma_s.$$

The standard deviation of the average of three (σ_{A3}) is thus $0.577350 \sigma_s$.

The density function [$f(X_{A3})$] of the average of three is qualitatively illustrated in figure A-2.

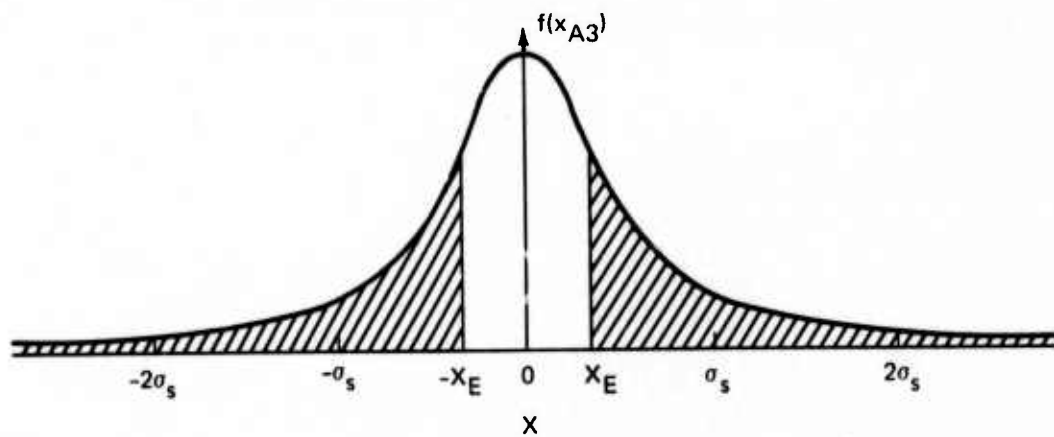


FIGURE A-2.—AVERAGE-OF-THREE DENSITY FUNCTION

The distribution function [$F(X_{A3})$] is the probability that the random variable, in this case the average of three (X_{A3}), will be less than or equal to any arbitrary value (from $-\infty$ to ∞) and is the integral of the density function [$f(X_{A3})$] from $-\infty$ to that arbitrary value.

The probability that the absolute value of X_{A3} will exceed some arbitrary value X_E can be obtained by summing the integrals of $f(X_{A3})$ from $-\infty$ to $-X_E$ and from $+X_E$ to $+\infty$.

The probabilities that the absolute value of the standard (X_S), the median (X_M), and the average of three (X_{A3}) will exceed any arbitrary value from 0 to $3.5\sigma_s$ are roughly plotted in figure 5-2.

A.2 FAILURE DETECTION CONSIDERATIONS

The means and standard deviations of the density functions for (one extreme minus the median) and (one extreme minus the average of three) can be determined similarly to the average-of-three determination by considering

$$E \left[(X_{(1)} - X_{(2)})^2 \right] = 0.45682\sigma_s^2 \Rightarrow \sigma_{E-M} = 0.6759\sigma_s$$

$$E (X_{(1)} - X_{(2)}) = 0.846\sigma_s = \mu_{E-M}$$

and

$$E \left[(X_{(1)} - X_{A3})^2 \right] = 0.22614\sigma_s^2 \Rightarrow \sigma_{E-A3} = 0.4755\sigma_s$$

$$E (X_{(1)} - X_{A3}) = 0.846\sigma_s = \mu_{E-A3}$$

The resulting density functions are qualitatively illustrated in figure A-3.

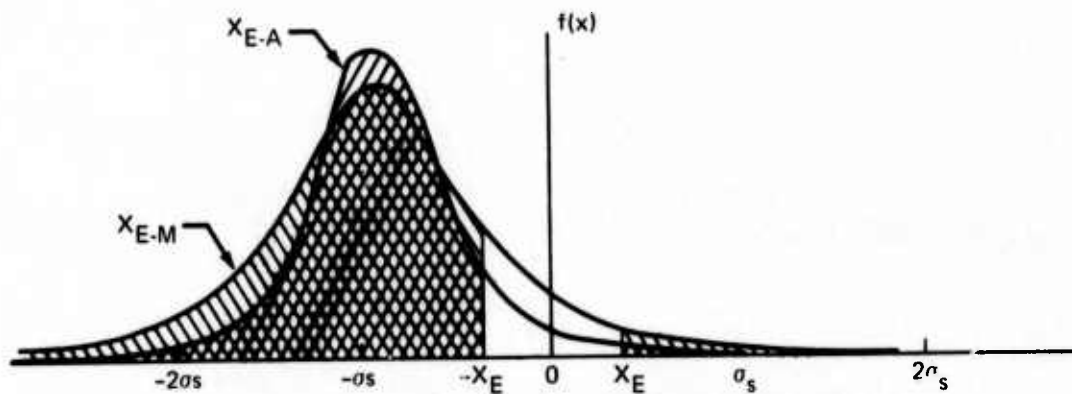
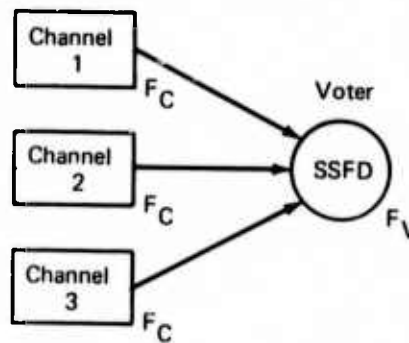


FIGURE A-3.—DENSITY FUNCTIONS—(EXTREME-MEDIUM) AND (EXTREME-AVERAGE OF THREE)

The probability that the absolute value of X_{E-A3} will exceed an arbitrary value X_E can be obtained, as before, by summing the integrals of $f(X_{E-A3})$ from $-\infty$ to $-X_E$ and from $+X_E$ to $+\infty$. The probabilities that the absolute value of the differences [(extreme minus the median) and (extreme minus the average of three)] will exceed arbitrary values from 0 to $3.5\sigma_s$ are plotted in figure 5-3.

A.3 SSFD FAILURE RATE CONSIDERATIONS

The reliability of a voted triple-channel system relative to the failure probability of the voter and the channel elements can be determined assuming the following configuration:



where

F_C = probability of channel failure

F_V = probability of voter failure

Let the total probability of failure be F_T and let F_S be the probability of a system failure; i.e., the probability of failure of at least two channels of a three-channel system. If F_C is small, F_S can be expressed as:

$$F_S = (F_C)^3 + 3(1 - F_C)F_C^2 \approx 3F_C^2$$

then

$$\begin{aligned} F_T &= F_C(1 - F_V) + (1 - F_C)F_V + F_C F_V \\ &\approx F_C + F_V = 3F_C^2 + F_V \end{aligned}$$

Now letting $F_V = \beta F_C$, where β is a percentage value:

$$F_T = 3F_C^2 + \beta F_C = (3F_C + \beta)F_C$$

The ratio F_T/F_c becomes

$$\frac{F_T}{F_c} = 3 F_c + \beta$$

The plot of this function is presented in figure 5-17.

APPENDIX B

BILINEAR TRANSFORMATION LINEARITY CHARACTERISTIC

Section 5.2.3 states that the basic bilinear transformation method (Tustin's substitution) presents unacceptable steady-state response characteristics. An analysis was conducted to determine the cause of the linearity irregularities. A summary of the results of this investigation is presented below.

A block diagram of the bilinear implementation of a second-order filter is shown in figure B-1, where the X_n , Y_n , A_i , and B_i values are expressed in digital (binary) 15-bit significant and the summation is accumulated at double precision (31-bit word length, plus sign). Neglecting any error in the coefficients A_i or B_i and assuming that the product summations are error free, the only error expected within the bilinear transformation process would be in the quantization of the final summation. This quantization error is denoted as E_o in figure B-1. Now assuming that the correct output Y_c is different from the actual output Y_n , their difference can be expressed as:

$$|Y_n - Y_c| = |Y_E| = \frac{E_o}{1 + B_1 + B_2}$$

Substituting the original continuous domain coefficients for B_1 and B_2 (refer to section 5.2.2.1, equation (5-18)) yields:

$$\begin{aligned} |Y_E| &= E_o \left[\frac{1}{4} + \frac{1}{(\omega_d T)^2} + \frac{\xi_d}{\omega_d T} \right] \\ &\approx E_o \frac{1}{(\omega_d T)^2} \end{aligned}$$

Figure B-2 illustrates the quantization error on the second-order filter output Y_n as a function of the filter natural frequency ω_d and sample period T . This plot shows that the linearity error increases as the filter natural frequency and/or sampling period is decreased. This relationship can be observed for the bilinear implementation of the HSAS A and B filters steady-state responses shown in figures 5-33 and 5-34, respectively. The B filter has the lowest natural frequency and thus exhibits the greatest linearity errors. The same trend was substantiated in the laboratory for changes in the sampling period.

Since the quantization error is a function of the coefficient significant and final summation significant, longer word length computations would reduce the linearity error of the basic bilinear transformation filter implementation. The quantization error E_o would decrease as a power of 2 for each bit increase in word length.

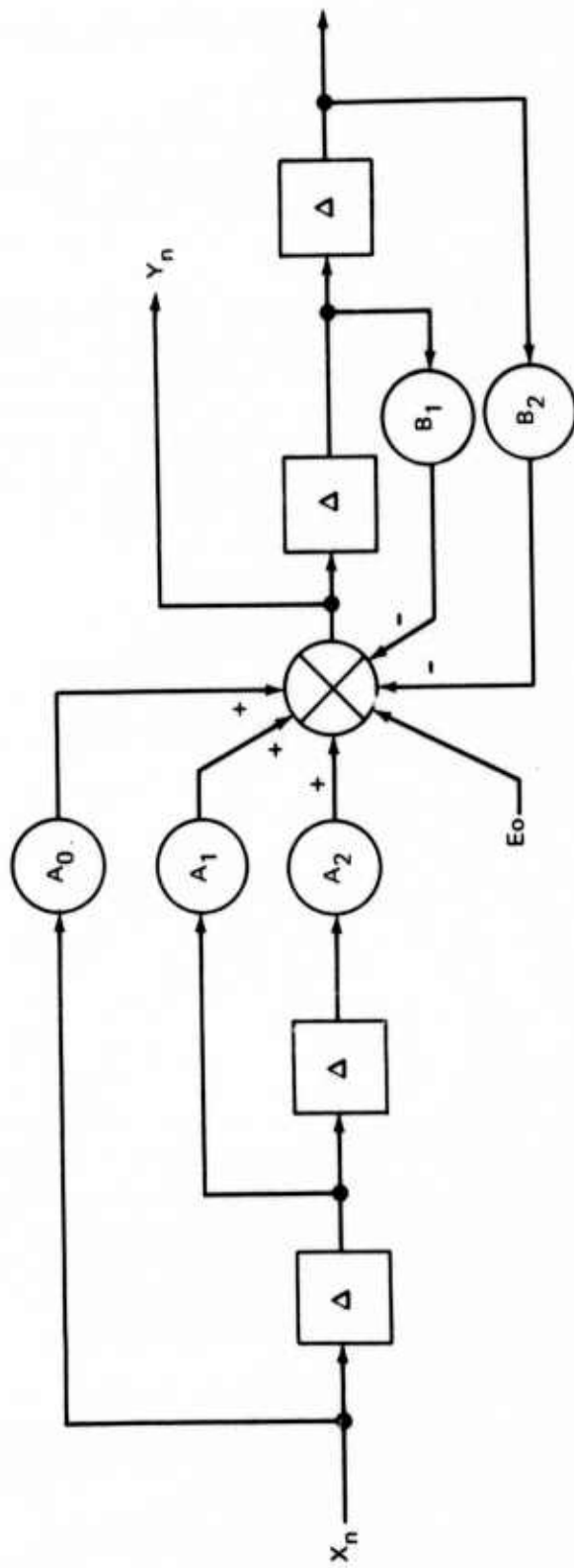


FIGURE B-1.—BILINEAR TRANSFORMATION IMPLEMENTATION—SECOND ORDER FILTER WITH QUANTIZATION ERROR

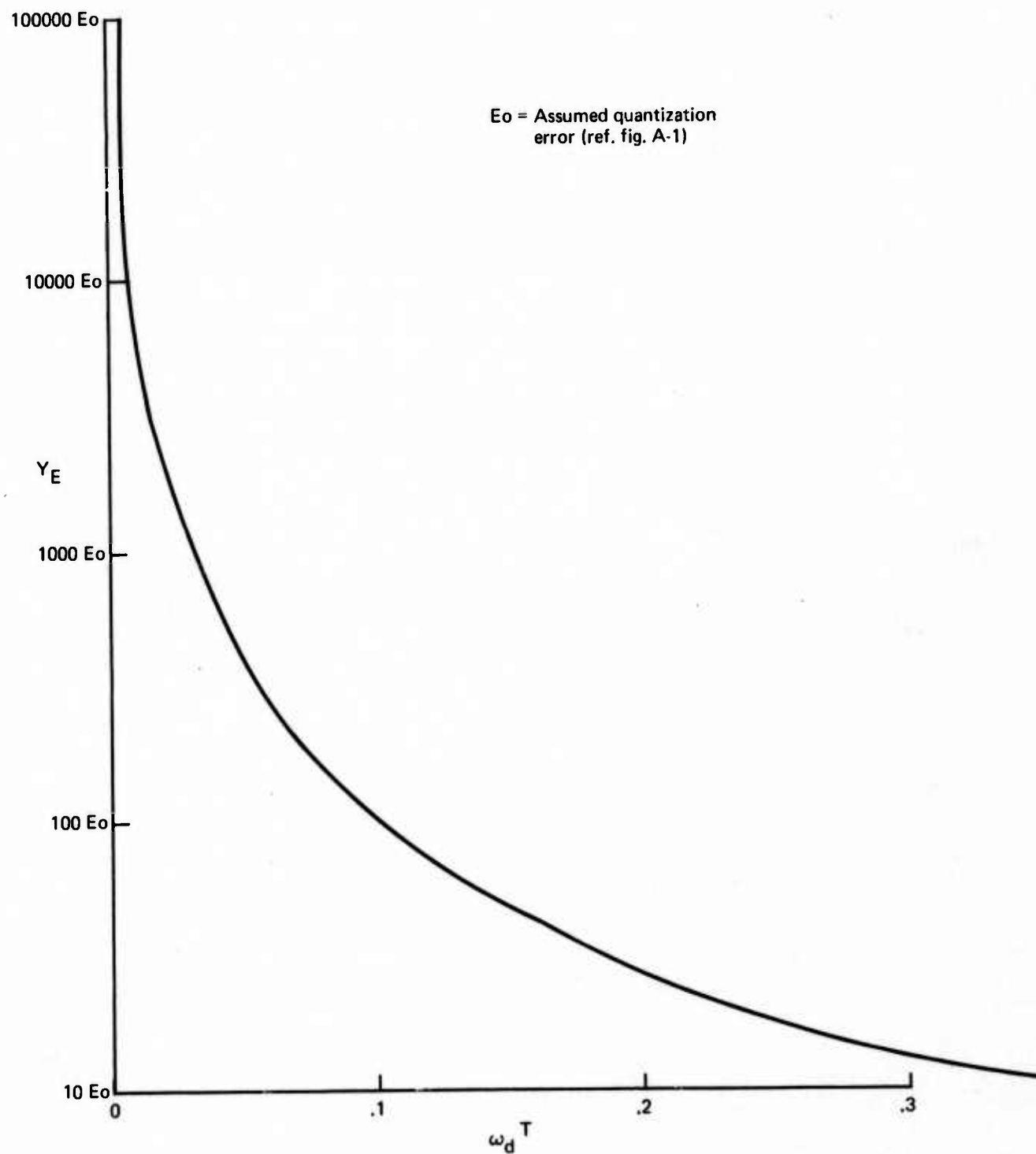
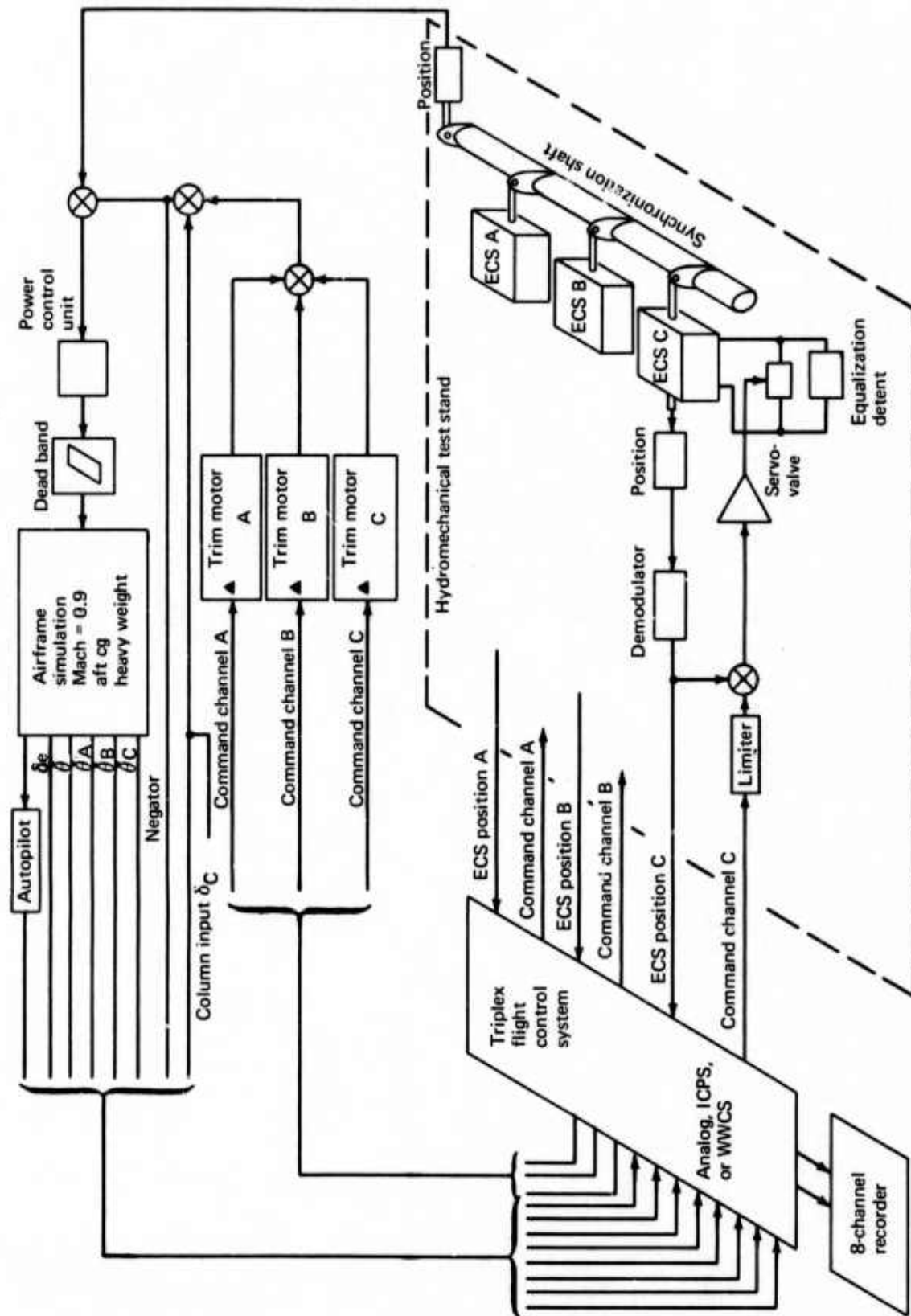


FIGURE B-2.—LINEARITY ERROR ESTIMATE



▲ Simulated on SD-80 analog computer

FIGURE 2-1.—LABORATORY TEST CONFIGURATION

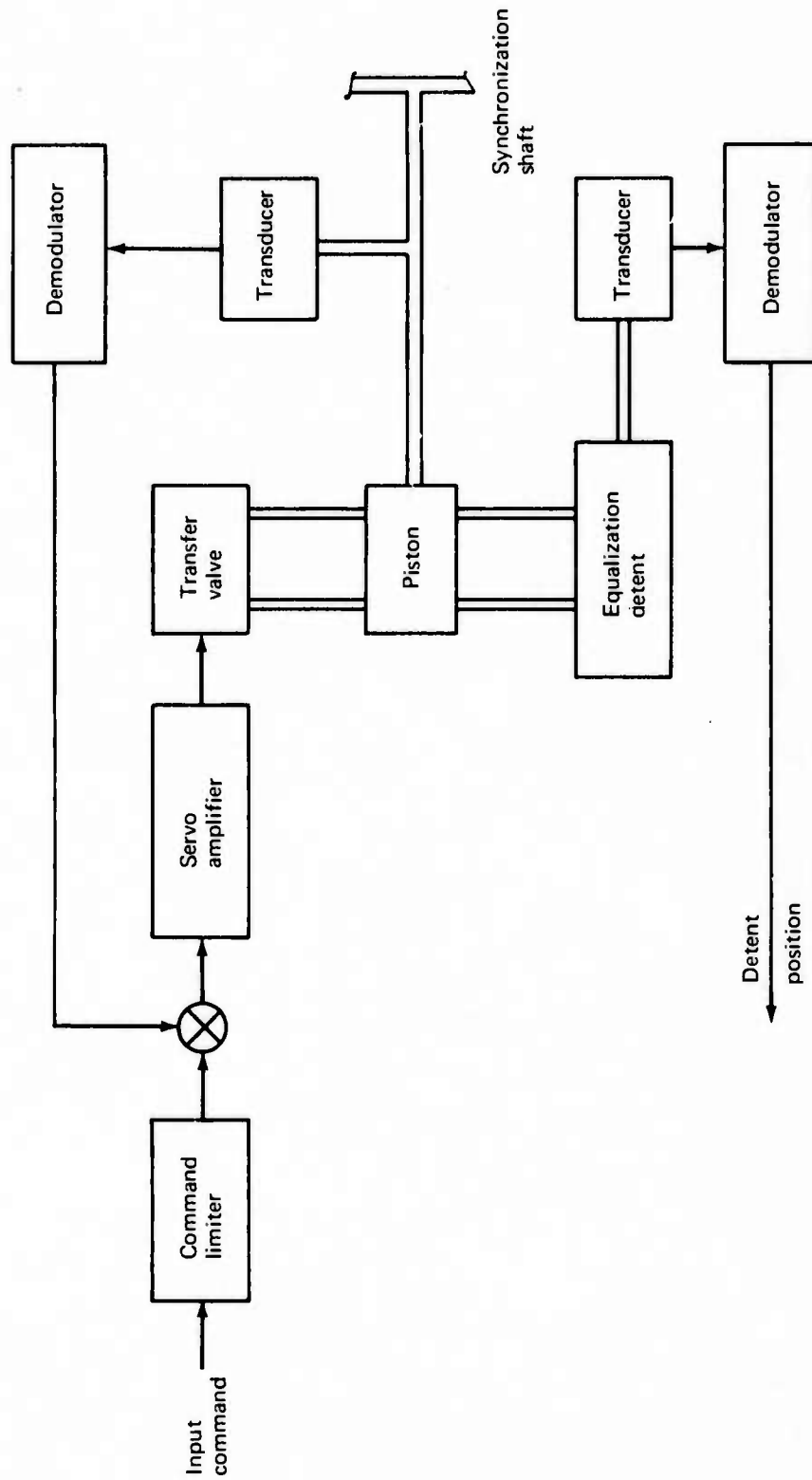


FIGURE 2-2.—ELECTRIC COMMAND SERVO (ECS)

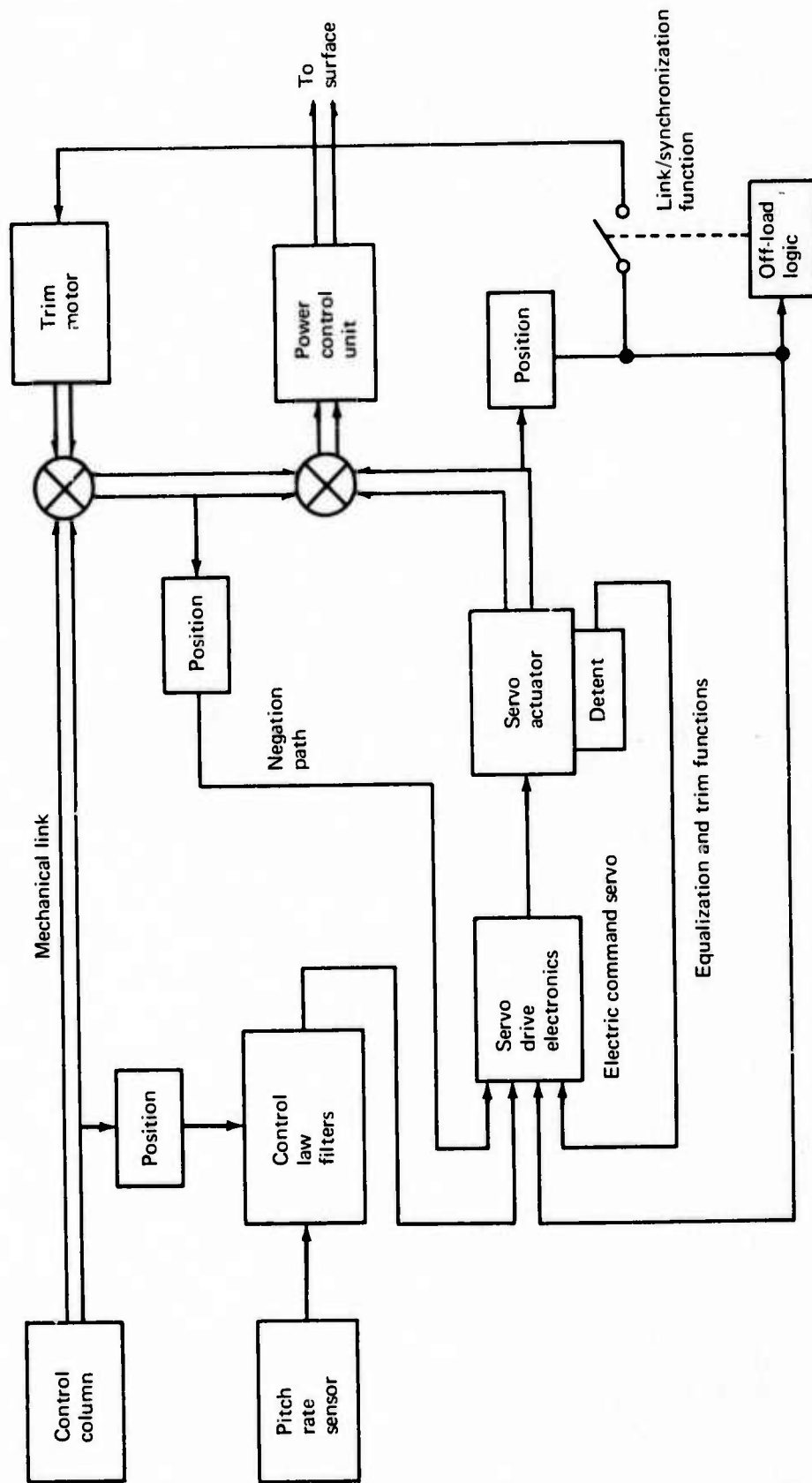


FIGURE 2-4.—HISAS FLIGHT CONTROL SYSTEM BLOCK DIAGRAM

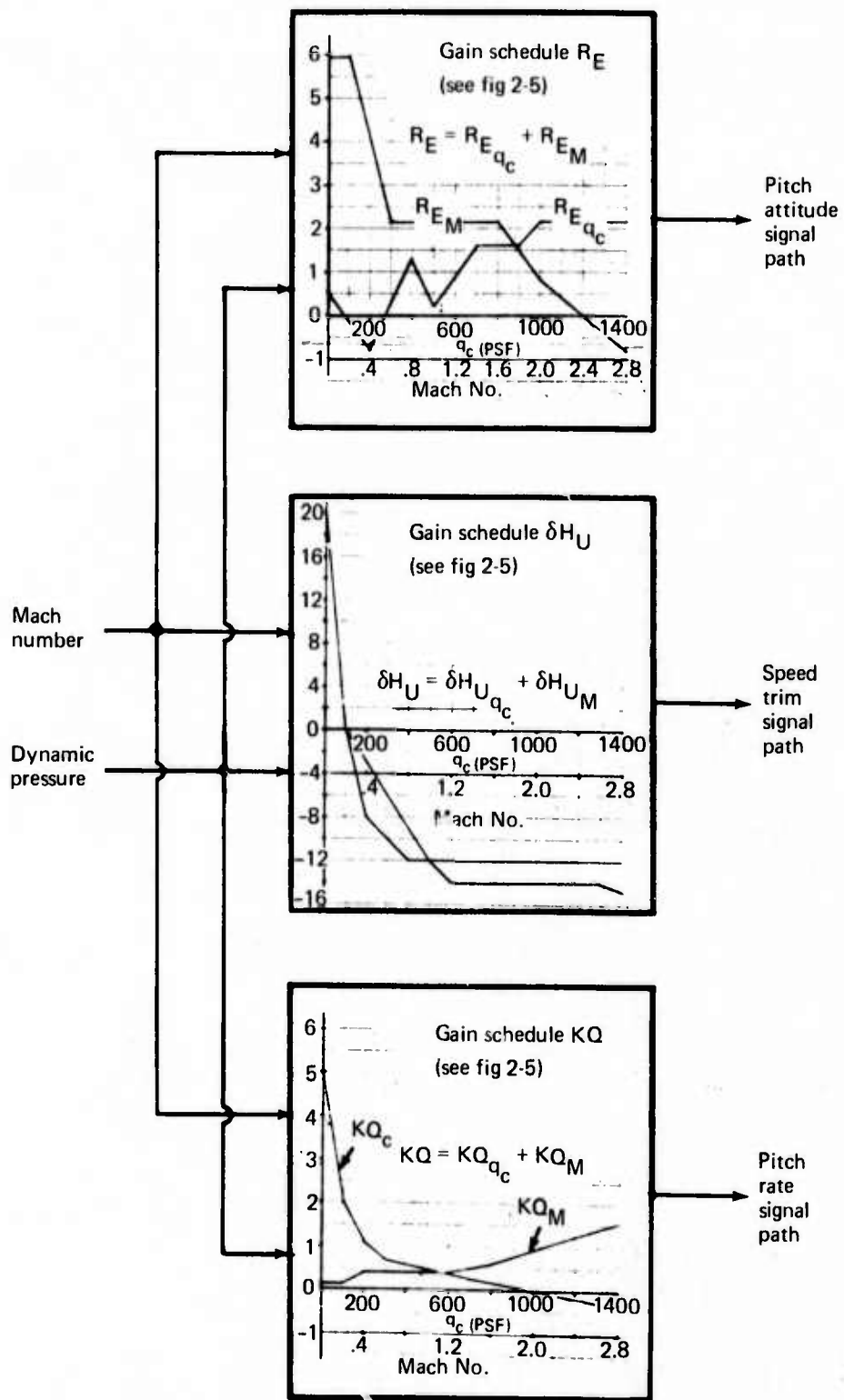
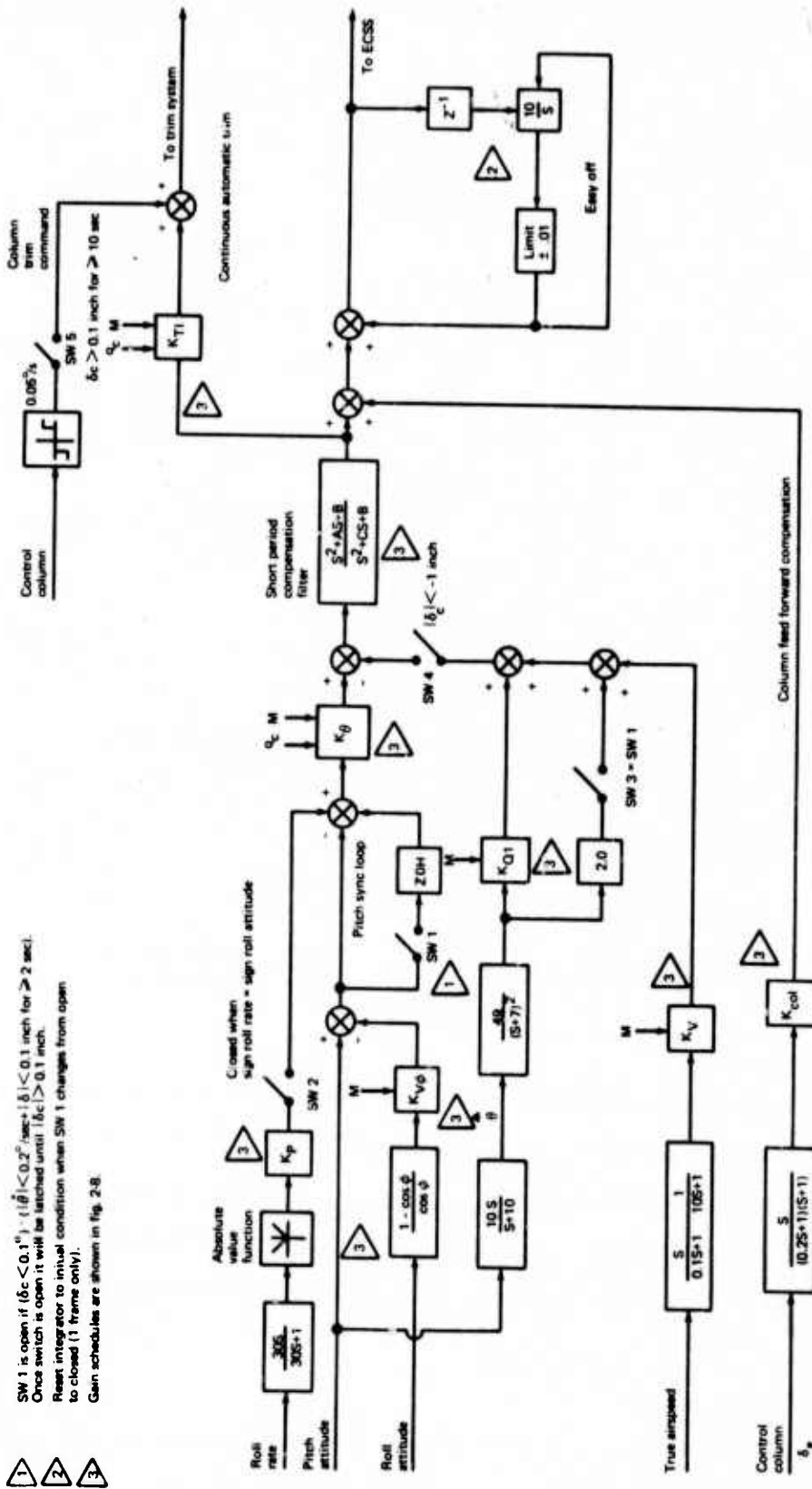


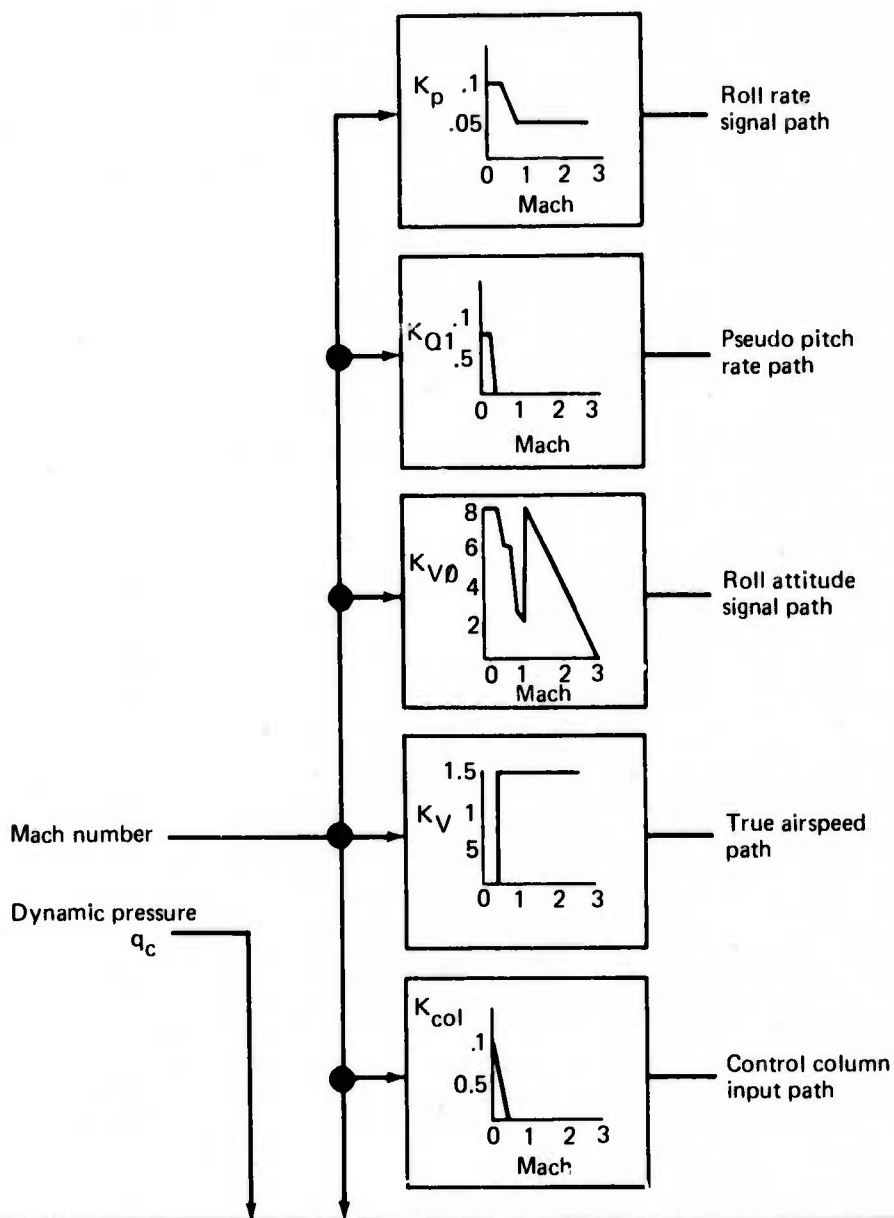
FIGURE 2-6.—ECSS GAIN SCHEDULE FUNCTIONS



SW 1 is open if $(\delta c < 0.1)$ or $(|\delta_c| < 0.2)$ / sec; $|\delta_c| < 0.1$ inch for ≥ 2 sec. Once switch is open it will be latched until $|\delta c| > 0.1$ inch. Reset integrator to initial condition when SW 1 changes from open to closed (1 frame only). Gain schedules are shown in fig. 2-8.

- 1
- 2
- 3

FIGURE 2-7.—PITCH CWS FUNCTIONAL BLOCK DIAGRAM



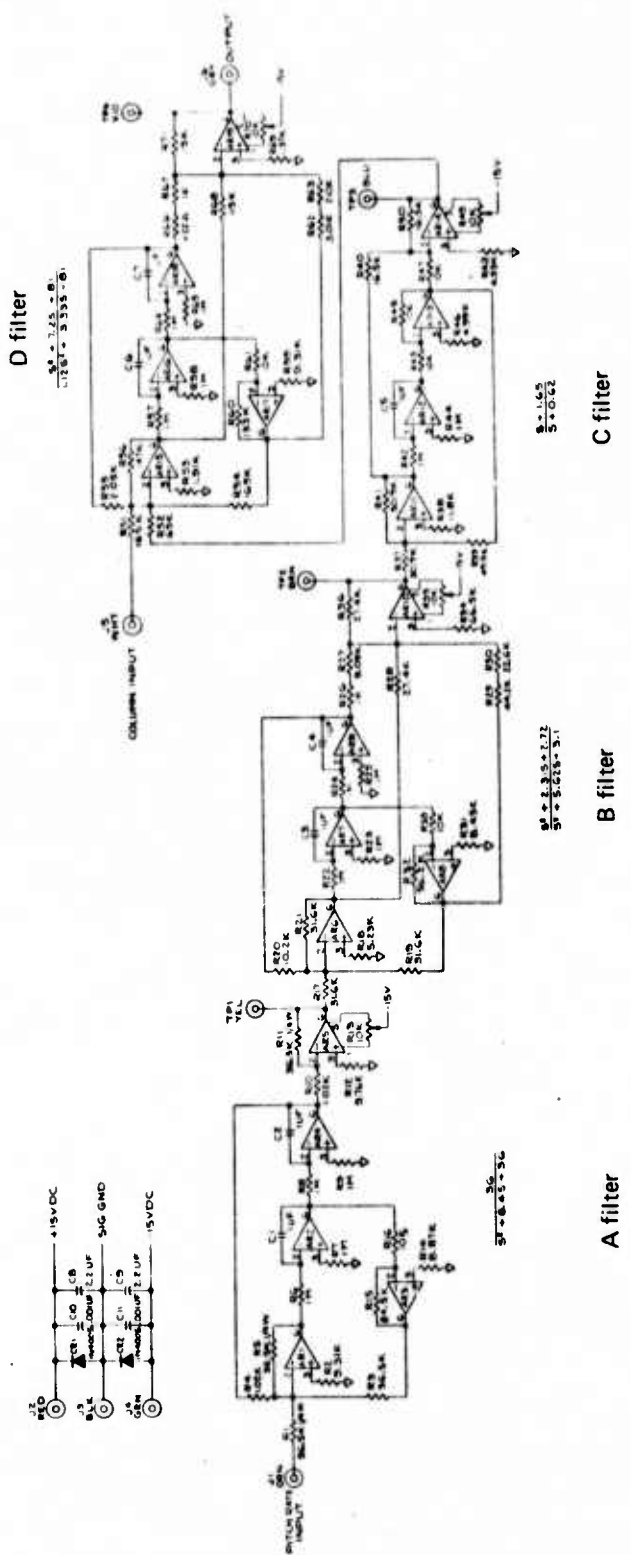
$K\theta$	0.9499	-0.41512	0.63136	-0.089288	0.19960	-0.057210	-0.0028096	1
K_{T1}	0.08547	0.028303	0.016984	-0.007469	-0.026171	0.0040572	-0.0001775	Mach
$\frac{A}{2}$	2.2042	0.47158	-1.44484	0.36188	0.28734	-0.013812	-0.00041206	Mach ²
$\frac{C}{2}$	0.93017	1.7514	-1.5718	0.37729	-0.12409	-0.015290	-0.0006782	Mach ³

$B = \left(\frac{A}{2}\right)^2 + \left(\frac{C}{2}\right)^2$

$K\theta, K_{T1}, A, B \text{ \& } C$

$\frac{q_c}{100}$	$\left(\frac{q_c}{100}\right)^2$	$\left(\frac{q_c}{100}\right)^3$
-------------------	----------------------------------	----------------------------------

FIGURE 2-8.—CWS GAIN SCHEDULE FUNCTIONS



Note:
 Except as noted, all resistors 1/8 W, 1%
 on all amplifiers.

FIGURE 3-1.—BREADBOARD HSAS FILTERS

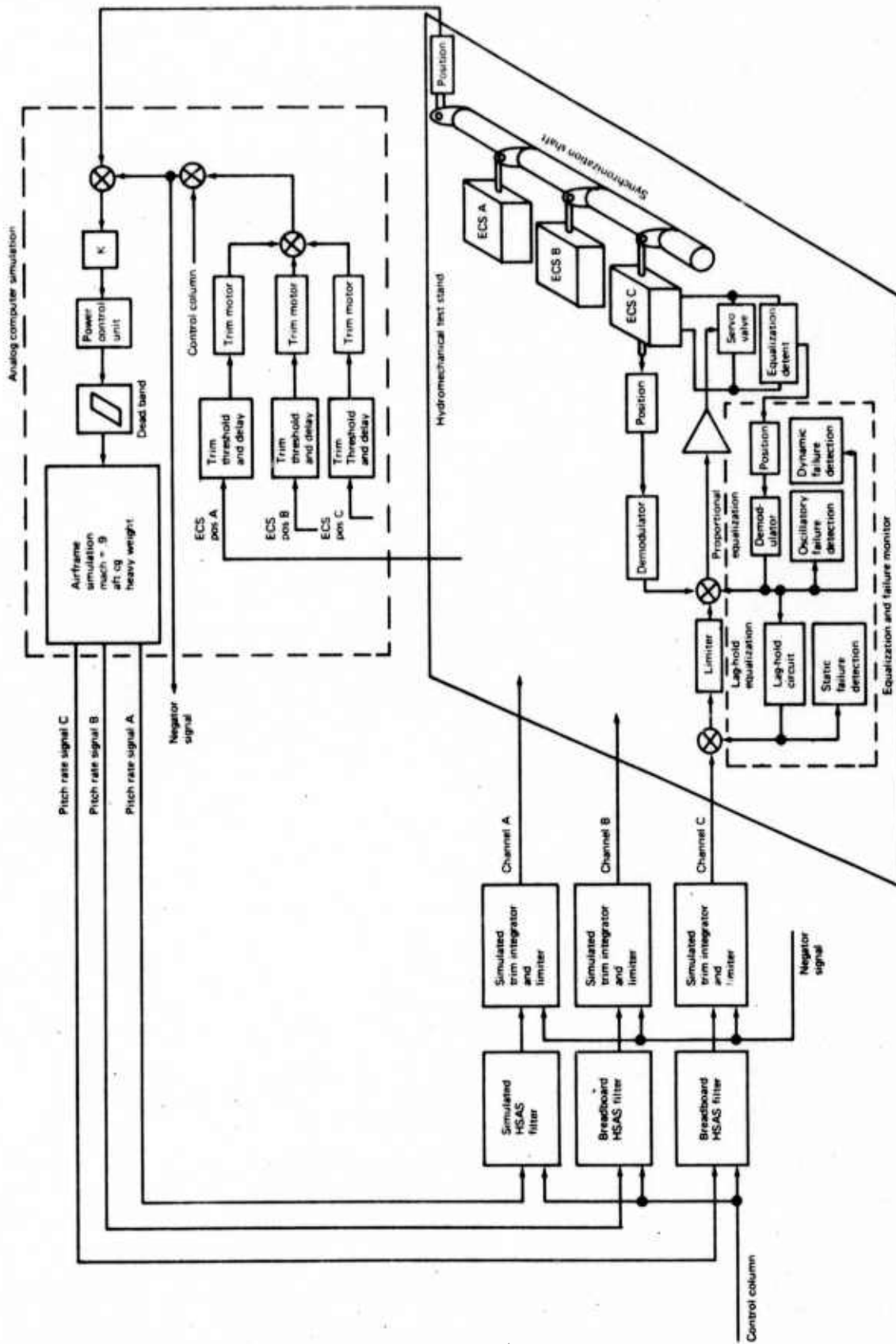


FIGURE 3.2.—LABORATORY TEST CONFIGURATION OF ANALOG HSAS

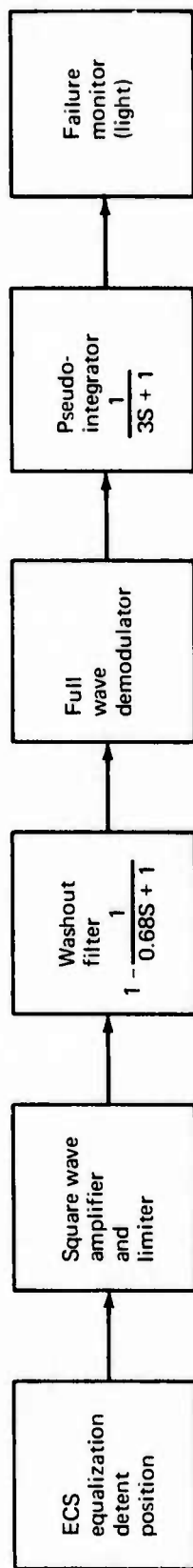


FIGURE 3.3.—ANALOG SYSTEM OSCILLATORY FAILURE DETECTOR (OFD)

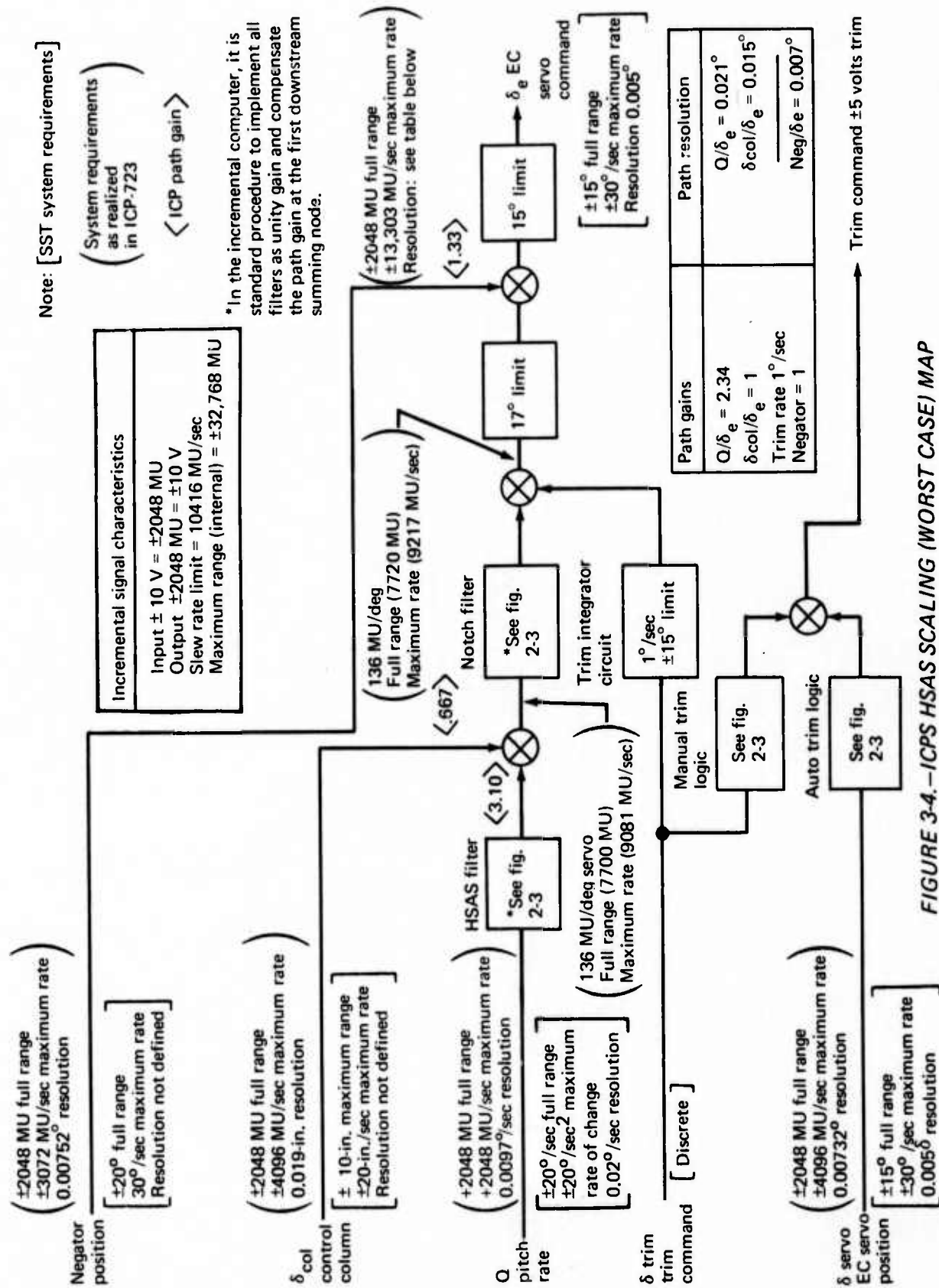


FIGURE 3-4.-ICPS HSAS SCALING (WORST CASE) MAP

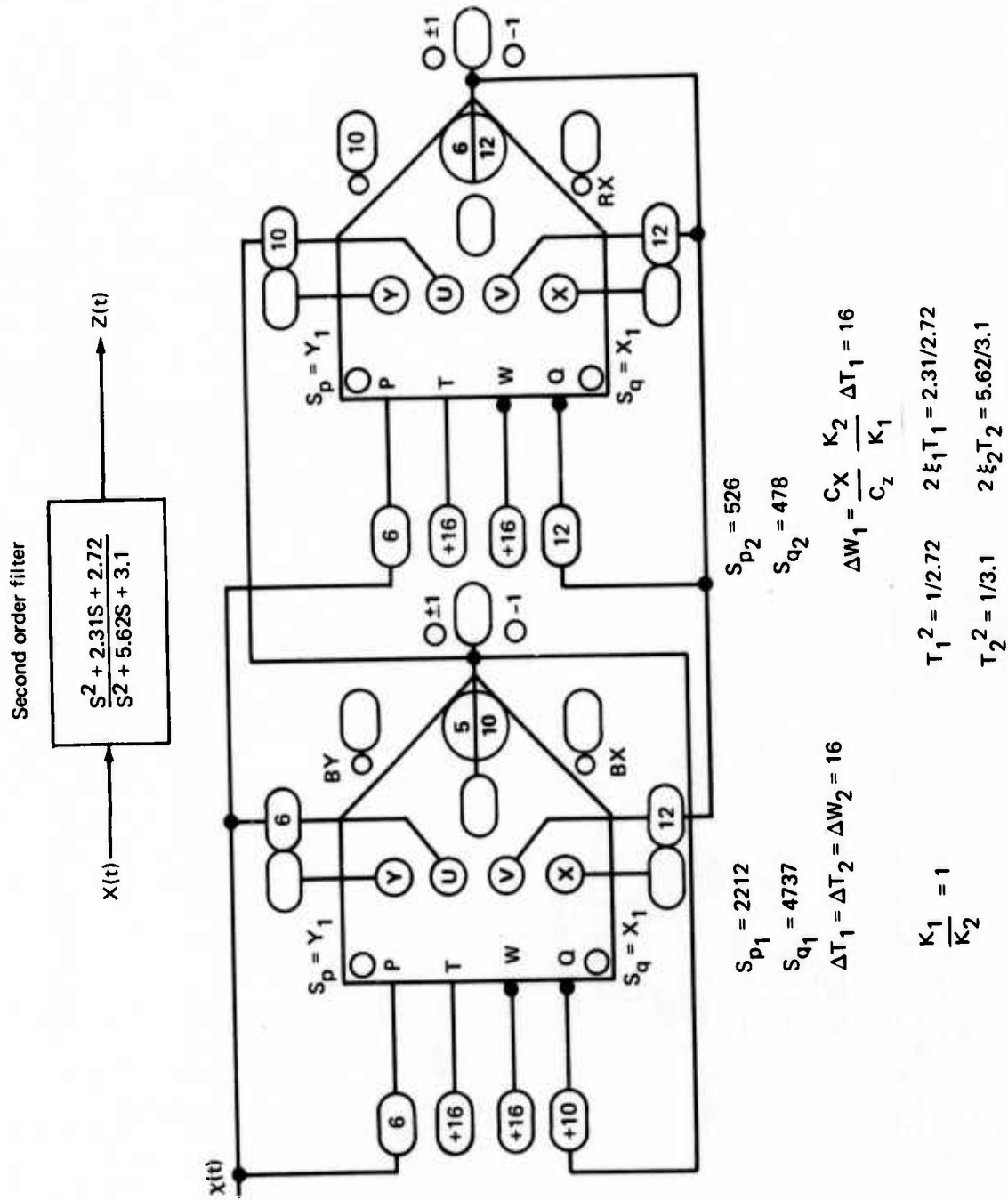
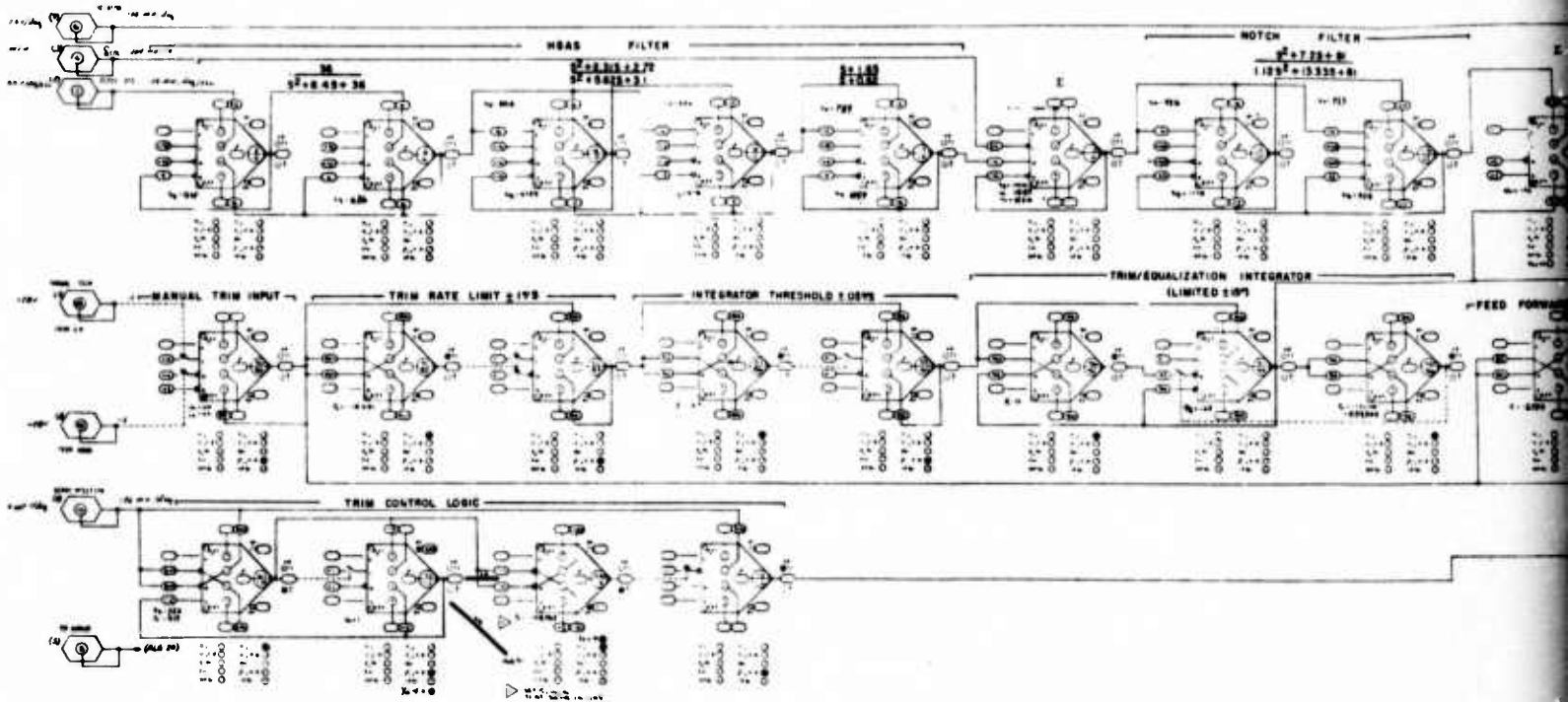
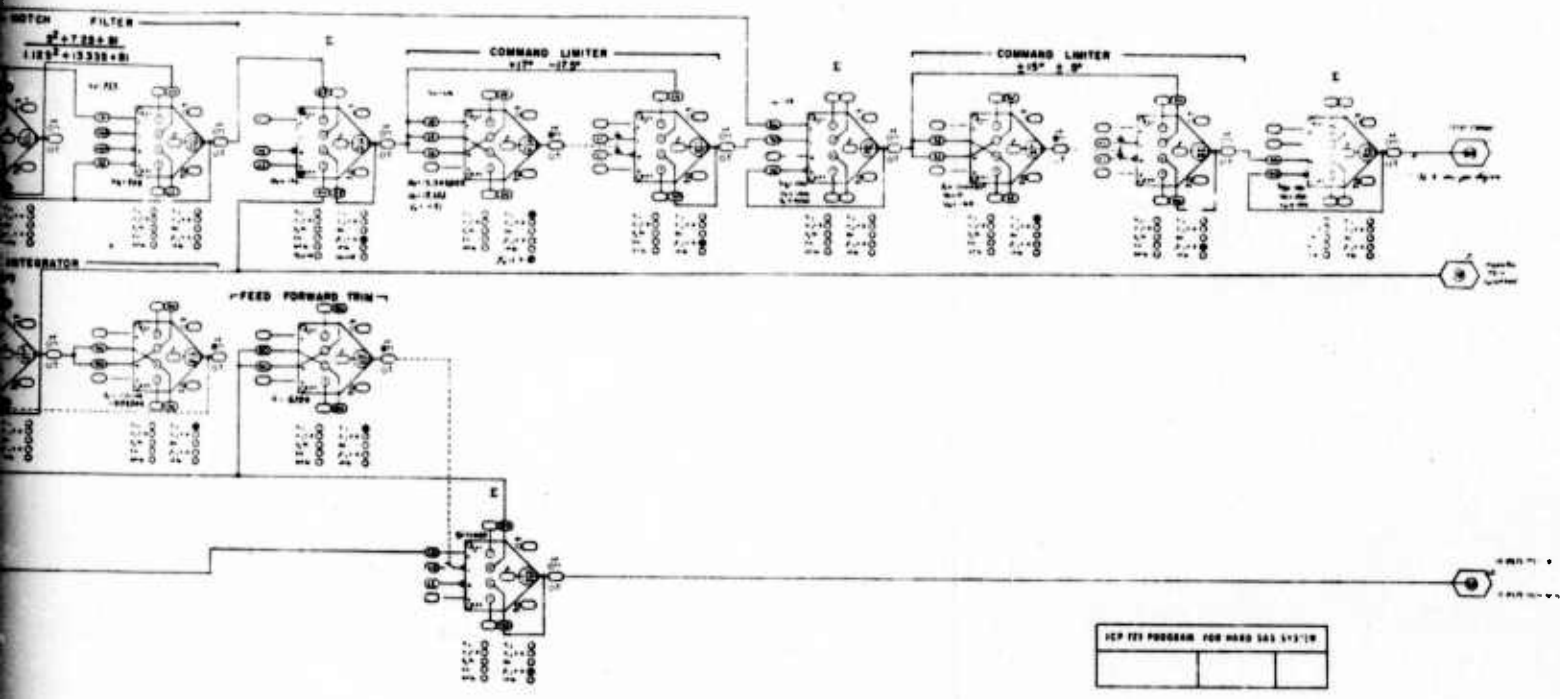


FIGURE 3-6.—HSAS FILTER B ALGORITHM MAP





Note:
 Dark circles shown in the algorithm map indicate control bits actuated for each function.

FIGURE 3-7.--HSAS ALGORITHM MAP

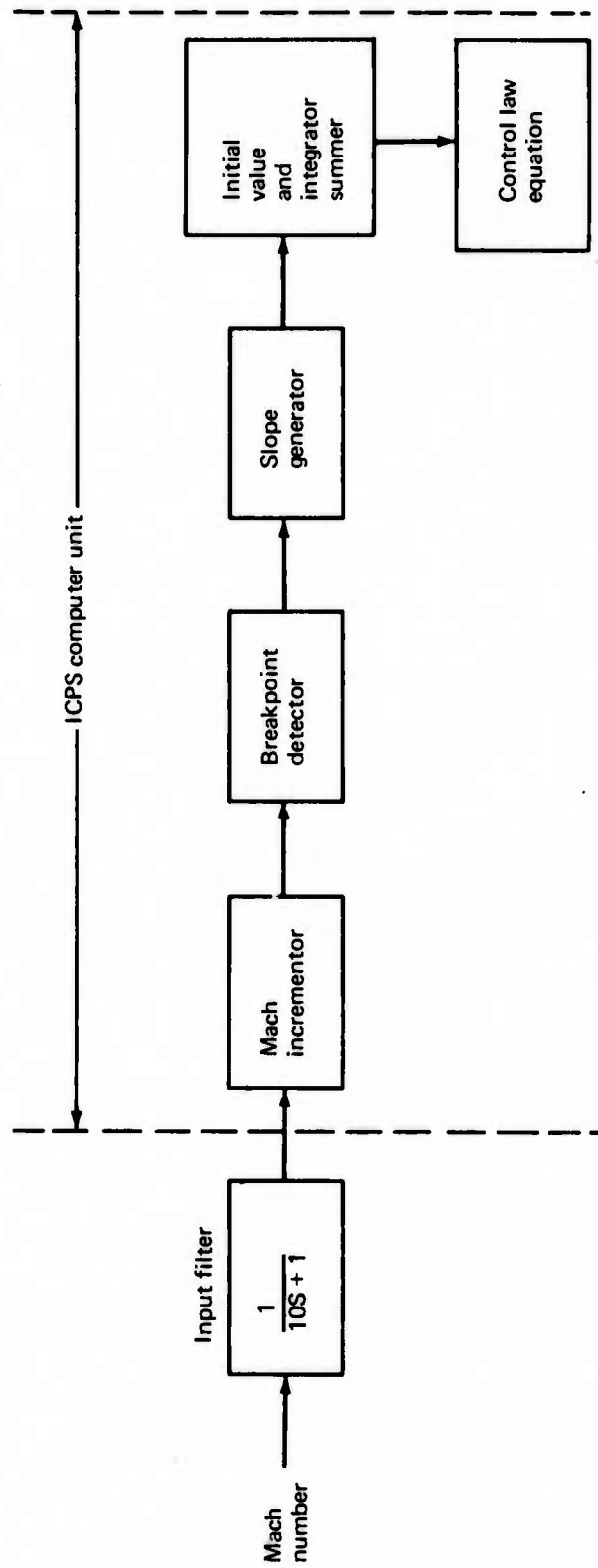
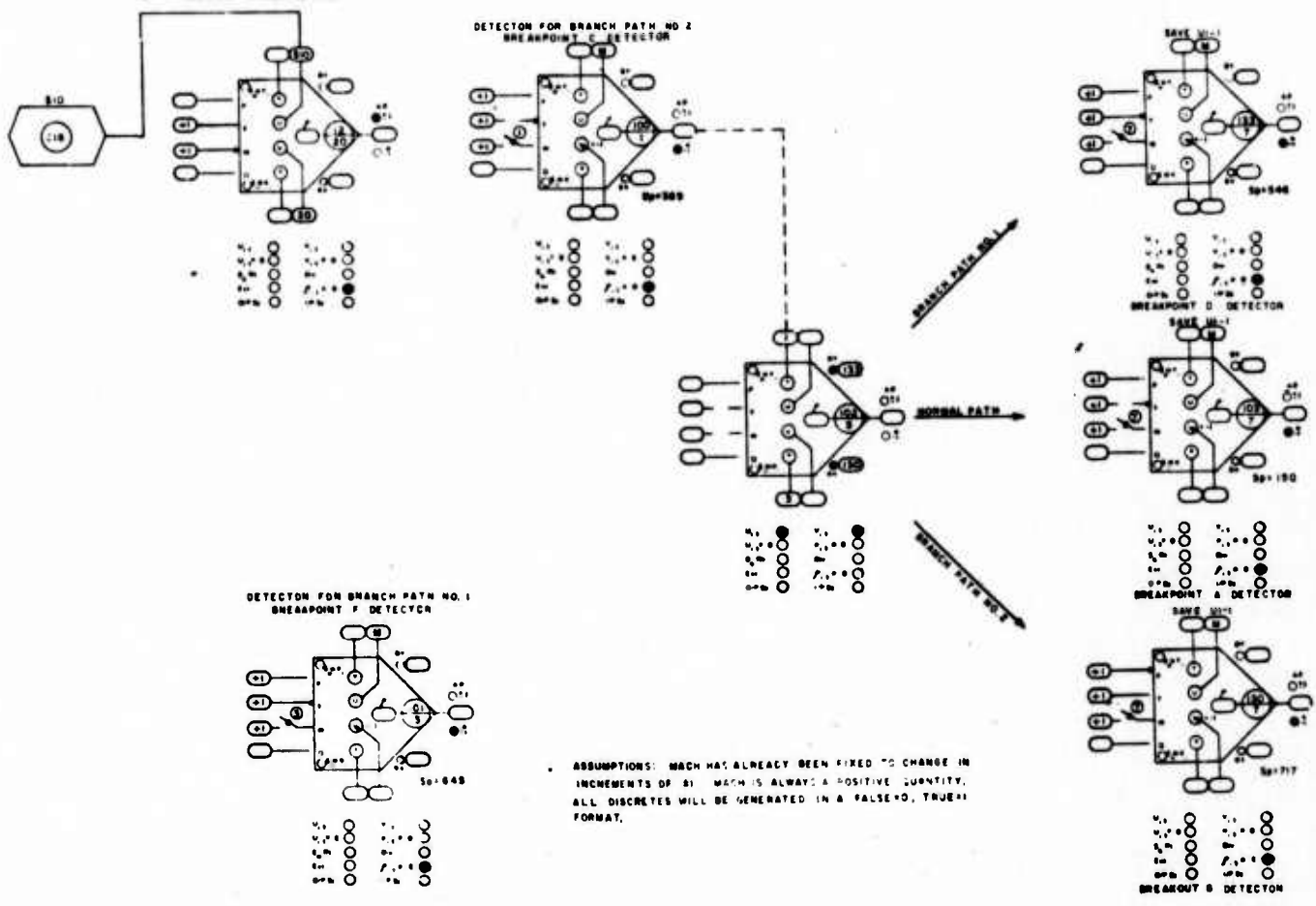


FIGURE 3-8.—ICPS GAIN SCHEDULE IMPLEMENTATION

← MACH INCREMENTOR →

← BREAKPOINT DETECTORS →



FACTORS ← SLOPE GENERATORS → SUMMER/INITIAL VALUE →

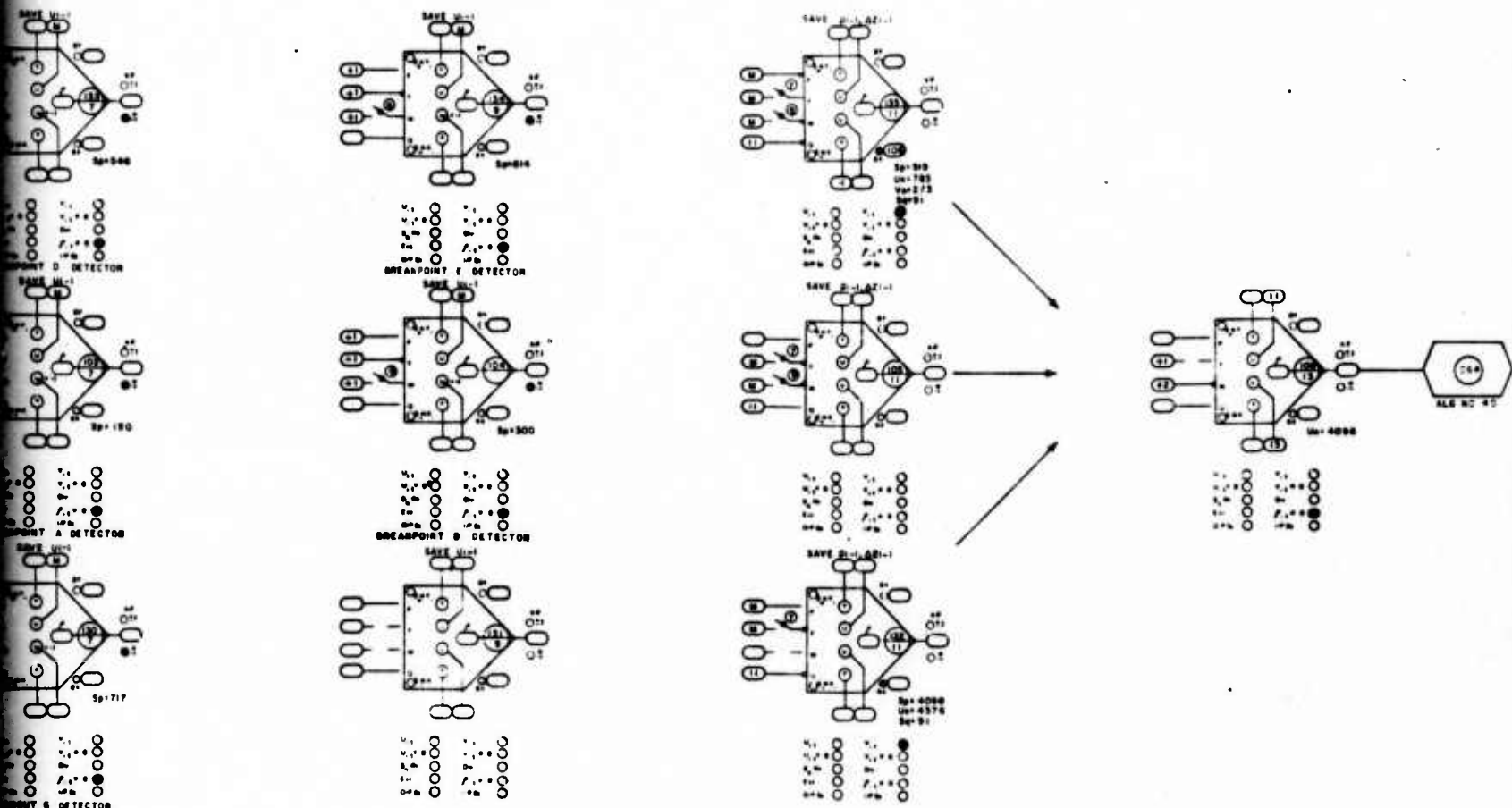


FIGURE 3-9.—GAIN SCHEDULE $K_{V\phi}$ ALGORITHM MAP

2

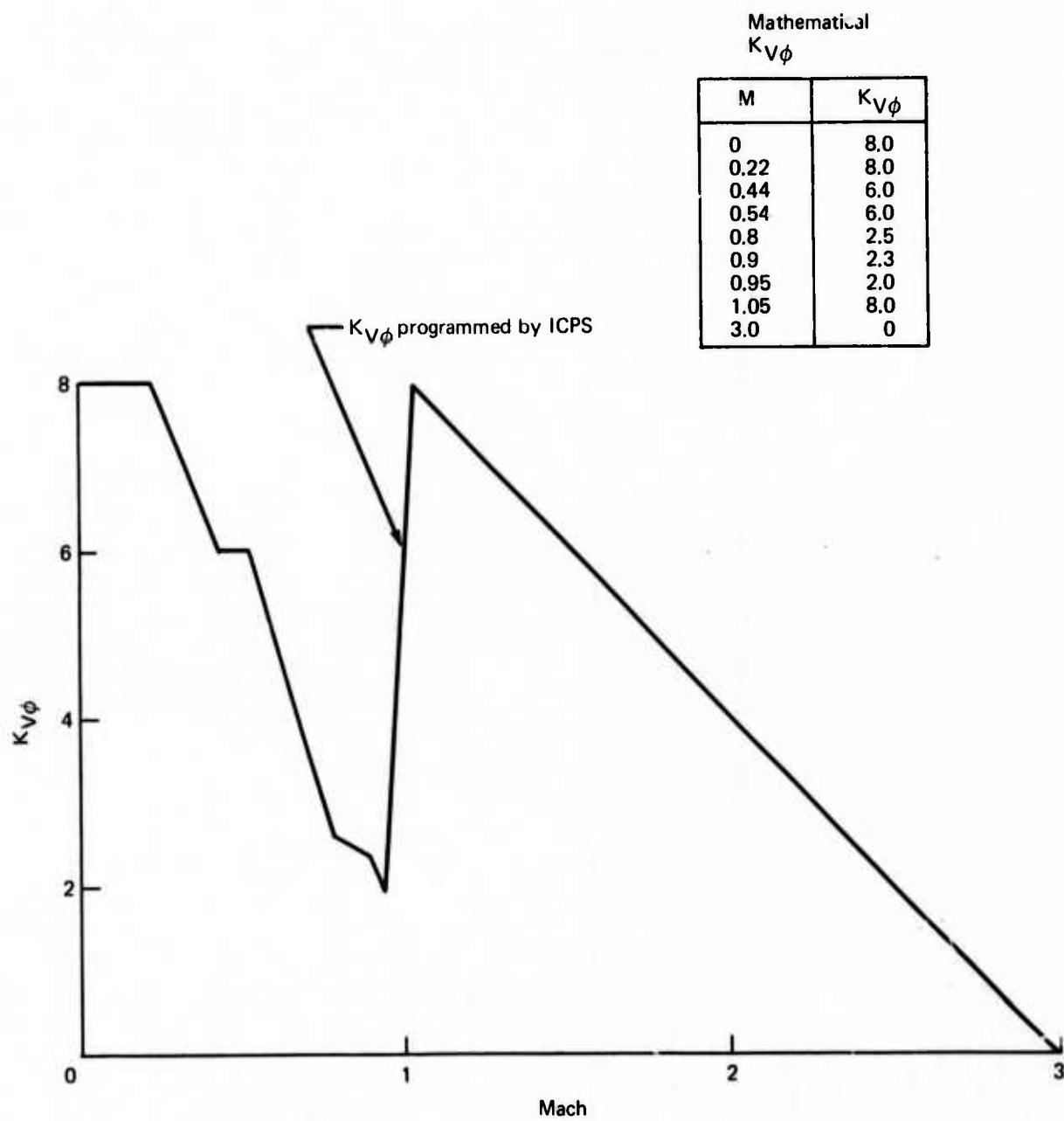


FIGURE 3-10.— $K_{V\phi}$ ACCURACY X-Y PLOT (LAB TEST)

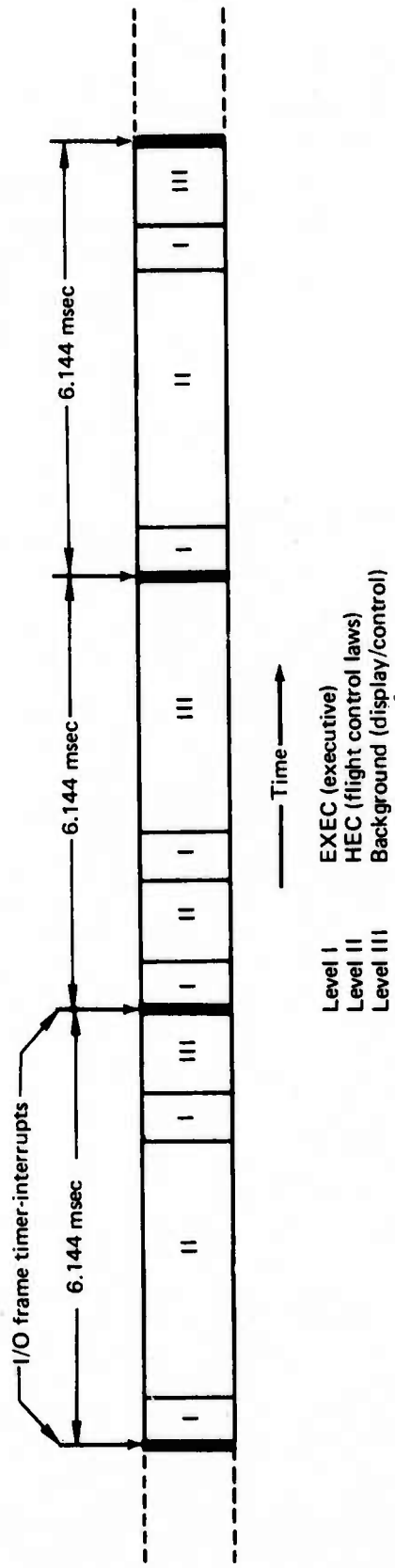
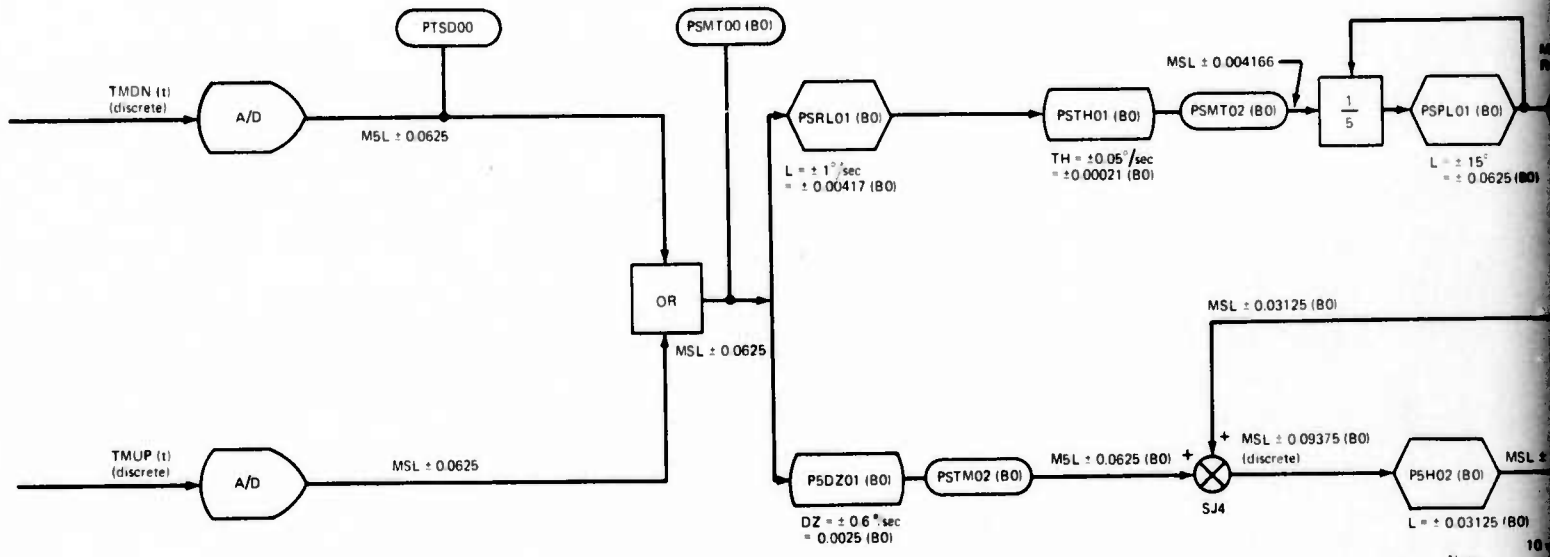
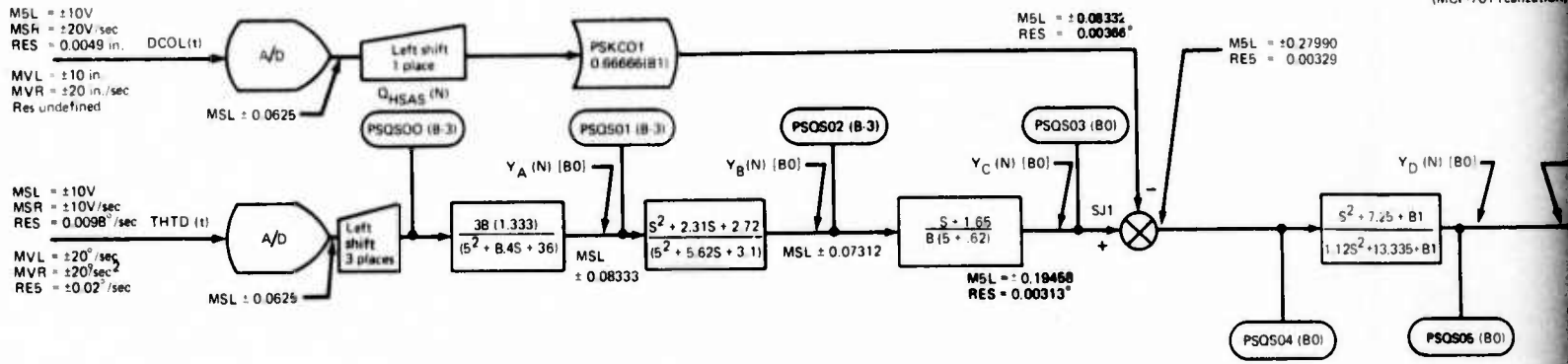


FIGURE 3-11.—WWCS FLIGHT SOFTWARE TIMING SEQUENCE

[SST system requirements
(MCP-701 realization)]



Note:

2¹⁶ = 2048 MU = X'07FF'

0.27990
0.00329

[SST system requirements]
[MCP 701 realization]

MSL = maximum signal level
MSR = maximum signal rate
MVL = maximum variable level
MVR = maximum variable rate
IMP = $1000/(N \cdot 6.144 \cdot N = 1, 2, \dots)$

Scaling factor PSES01 (BX)
denotes that the signal
is scaled by a factor of 2^X

Overflow protection: MSLXXX
denotes the maximum signal
level under worst case
conditions (in decimal volts)

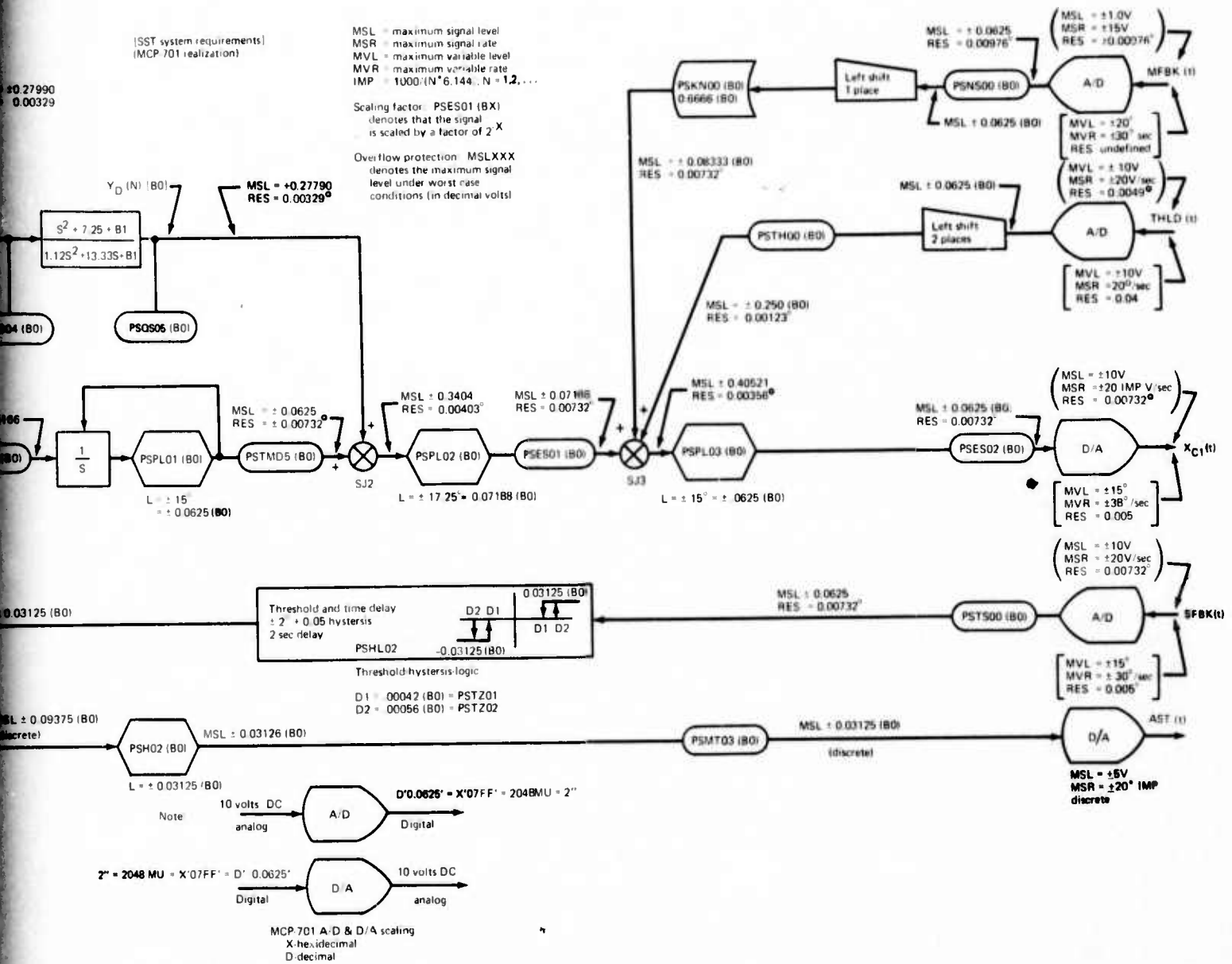


FIGURE 3-12.—HSAS DIGITAL SYSTEM DIAGRAM

2

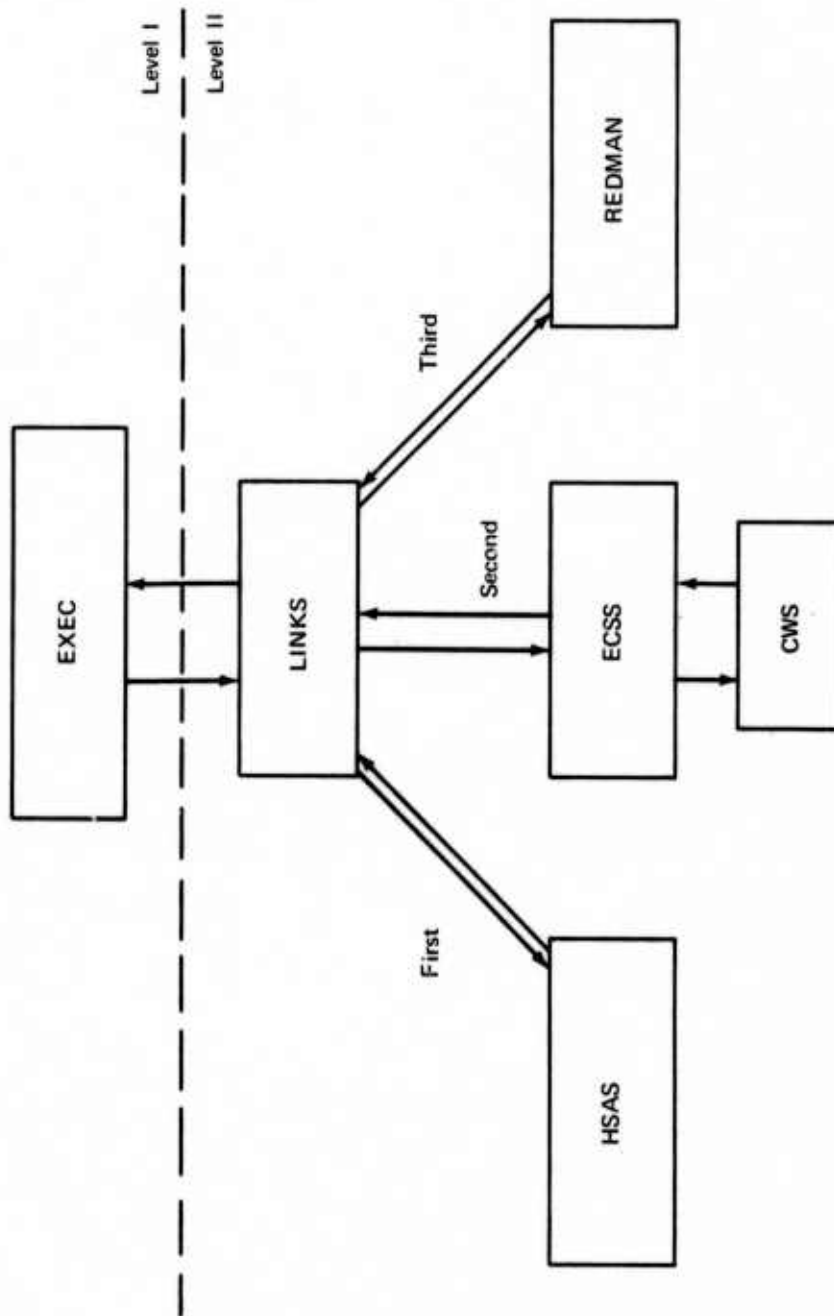
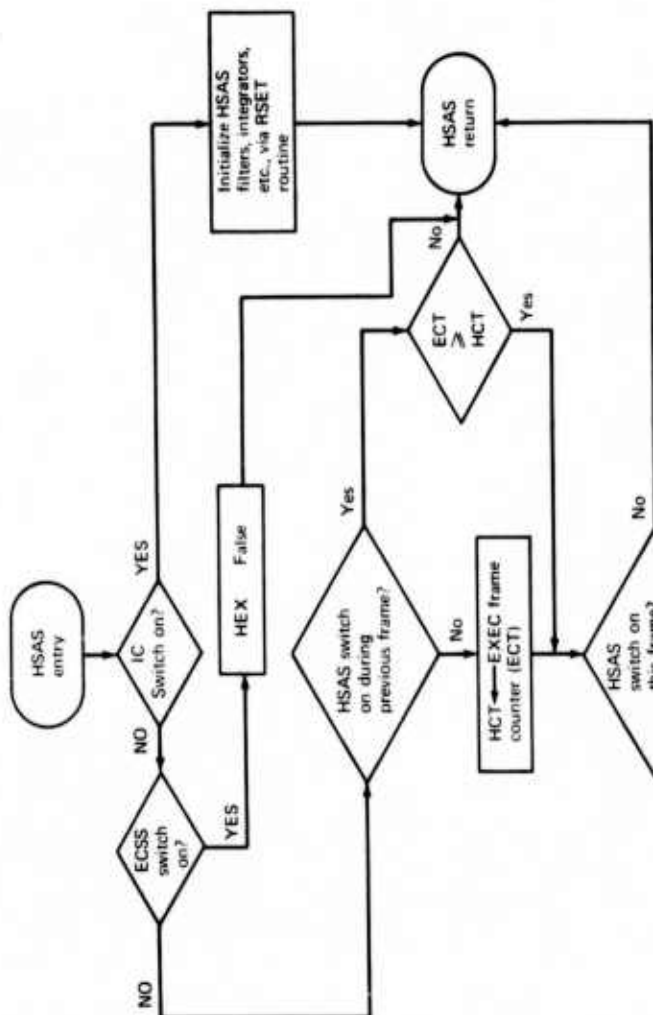
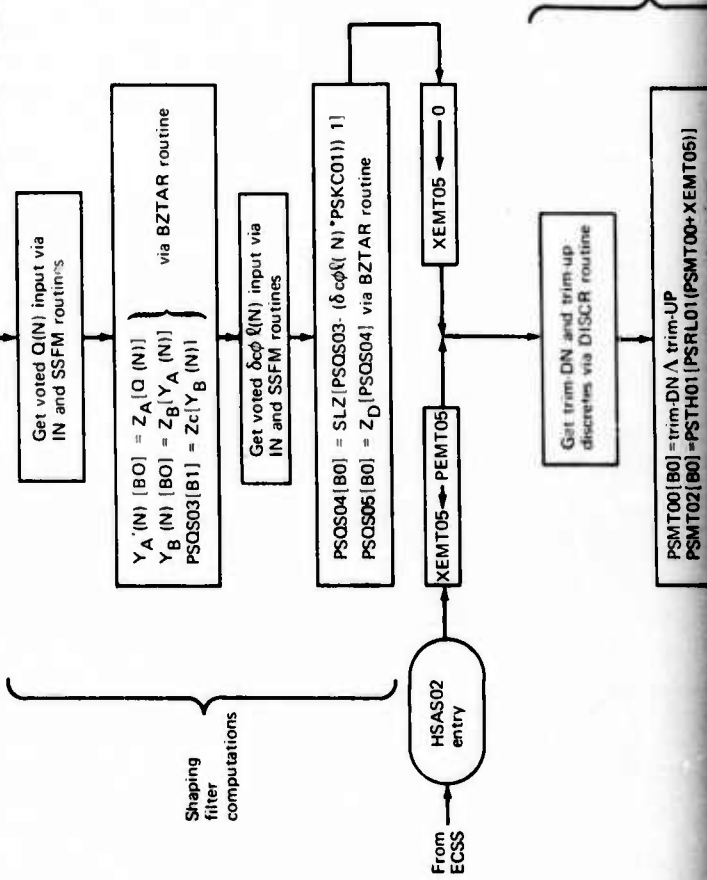


FIGURE 3-13.—COMPUTATIONAL SEQUENCE OF LEVEL II FLIGHT CONTROL SOFTWARE (HEC)



Z_X = transfer function of stage X
 A = logical OR
 SLZ = shift left (enter zeros)
 SRS = shift right (repeat sign)



Shaping filter computations

First leg of manual trim computations

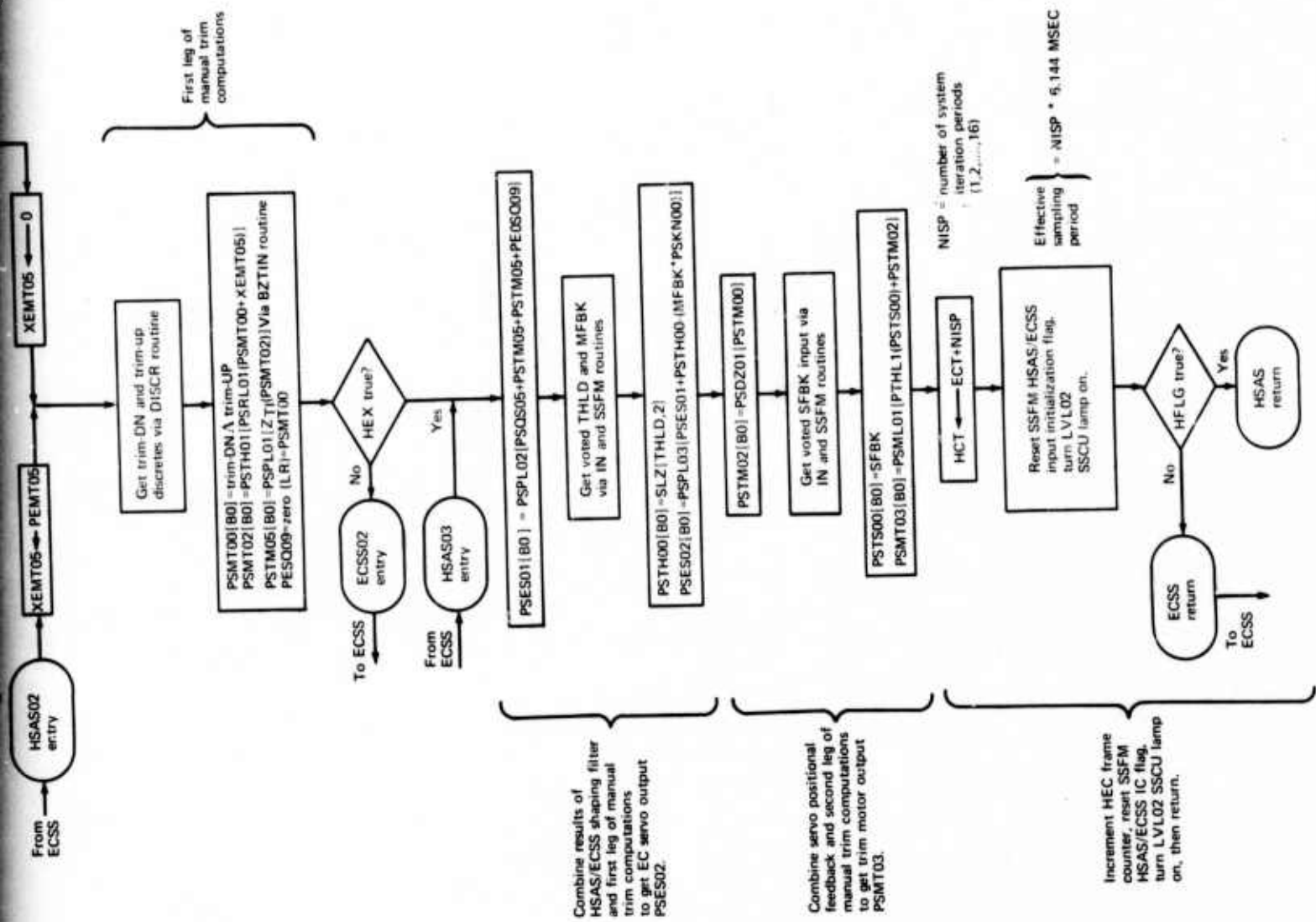
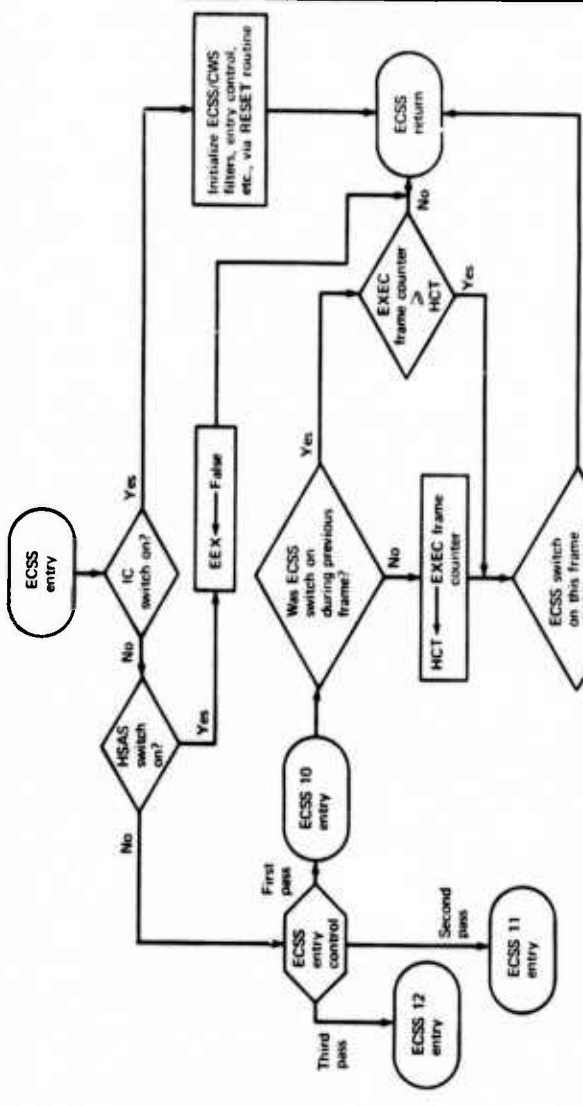


FIGURE 3-14.—HSAS SUPERVISOR MACRO-FLOWCHART



ZX = transfer function of stage X
 SLZ = shift left (enter zeros)
 SRS = shift right (repeat sign)
 (YR) = contents of Y register

EEX ← True
 Get voted IPRS and MNUM inputs via IN and SSFM routines.

$q_c = PEIP01[B-4] = Z_{qc} \{ [IPRS(N)] \}$ via BZTAR routine
 $M = PEMN01[B-4] = Z_M \{ [MNUM(N)] \}$
 $TEMP = HU/q_c \cdot M / [N-1]$

$PEST03 = PEST05[B-1] = PESL02[PEKT01 * (TEMP-PEST04)]$

Get voted O(N) input via IN and SSFM routines

$PEQ0501[B-3] = ZAE \{ O(N) \}$
 $PEQ0502[B-2] = ZBE \{ PEQ0501 \}$ via BZTAR routine
 $PEQ0503[B-0] = ZCE \{ PEQ0502 \}$
 $PEST06[B-1] = Z_{t1} \{ PEST05 \}$ via BZTIN routine
 $PEST04[B-1] = Z_{Tj} \{ PEST03 \}$

Get voted $\delta COL(N)$ input via IN and SSFM routines

$PEK001[B-3] = PEKC01 * SLZ \{ COL(N), 3 \}$

ECSS 11 exit

ECSS 11 entry

Compute PCWS02 [B0] and PCWS01 [B-3] via CWS routine

Air data and first part of speed trim computations via subroutine AIR

Second part of speed trim subsystem, O(N) shaping filter, integration/lag, and column input computations

Control wheel steering

First pass computations

Second pass

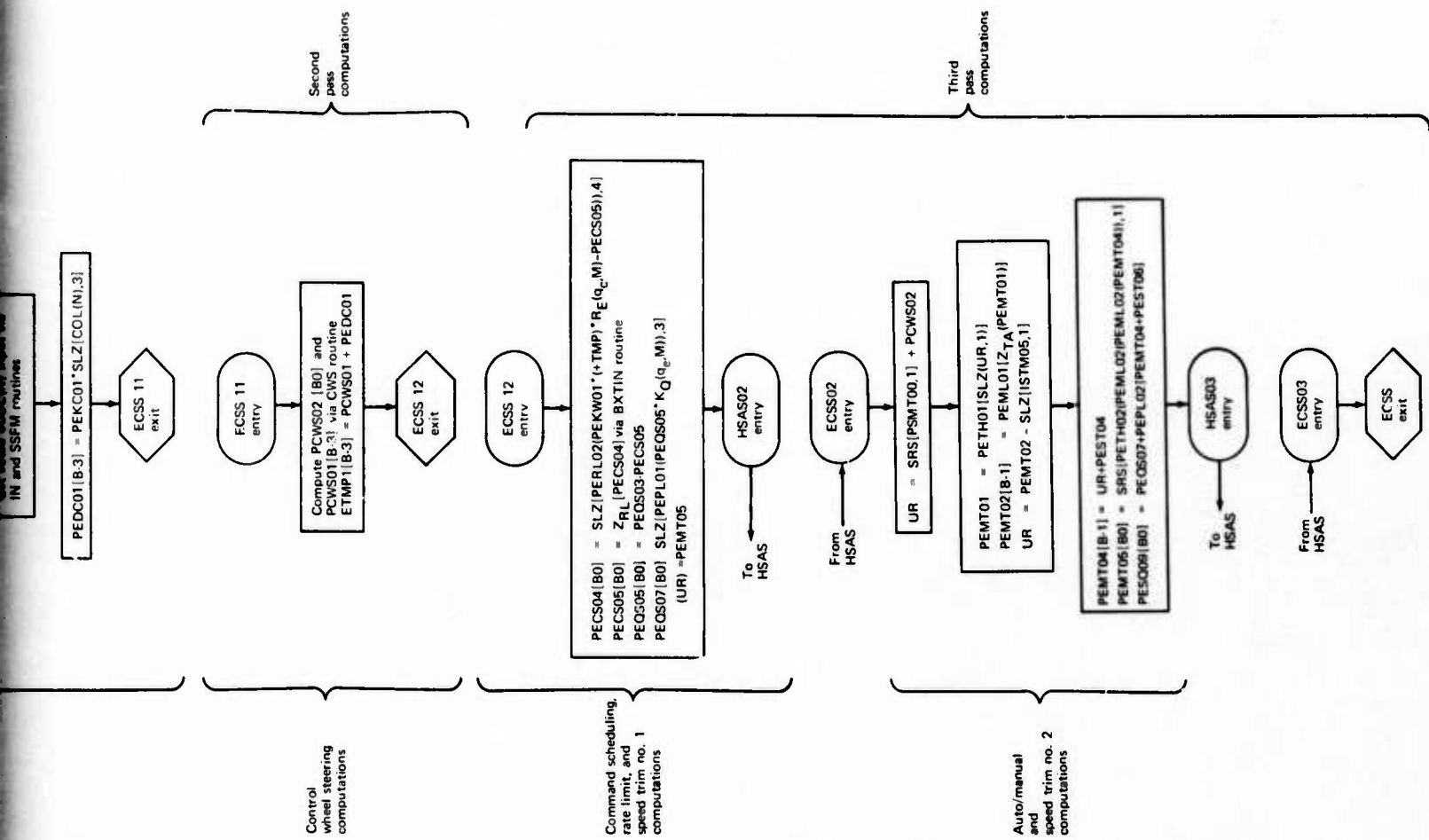


FIGURE 3-15.-ECSS SUPERVISOR MACRO-FLOWCHART

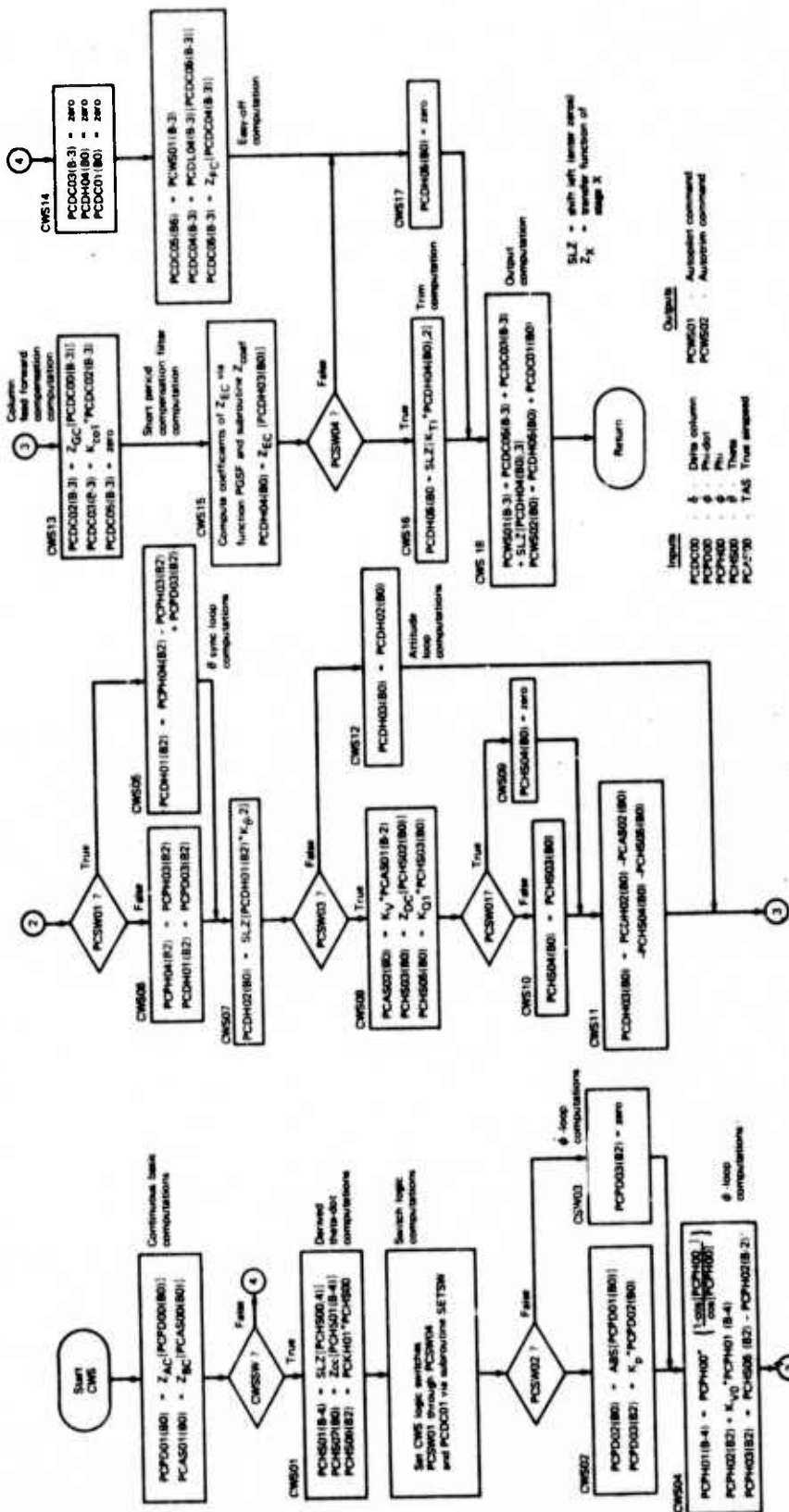


FIGURE 3-16.-CWS SUPERVISOR MACRO-FLOWCHART

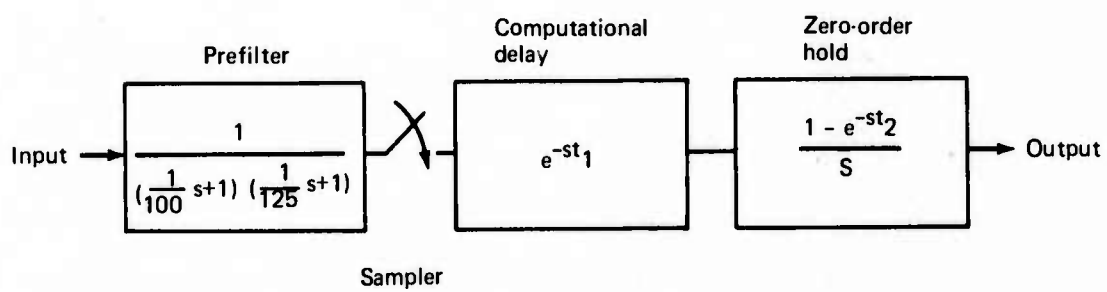


FIGURE 4-1.—MATH MODEL FOR ICPS AND WWCS BASIC HARDWARE

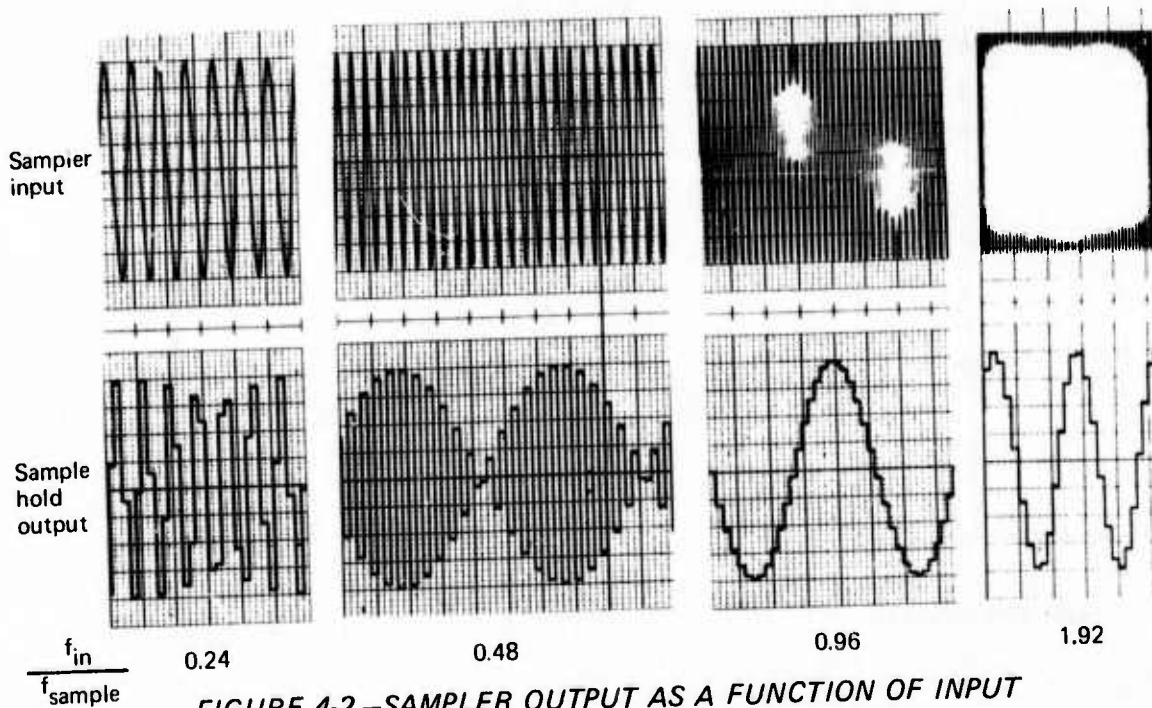
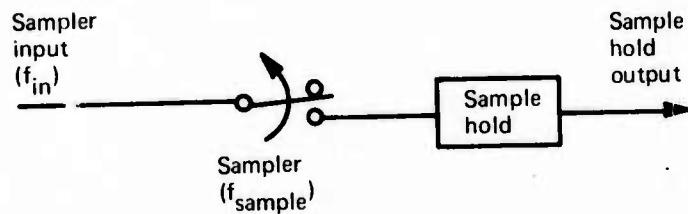
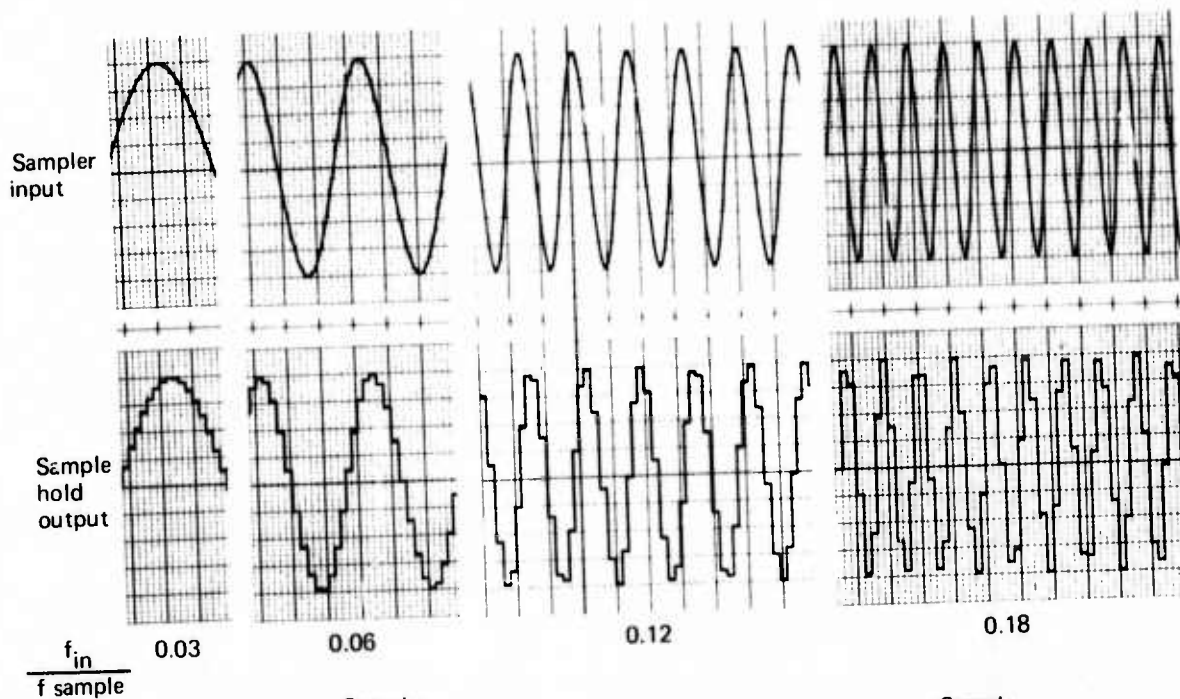


FIGURE 4-2.—SAMPLER OUTPUT AS A FUNCTION OF INPUT SAMPLING FREQUENCY (ALIASING)

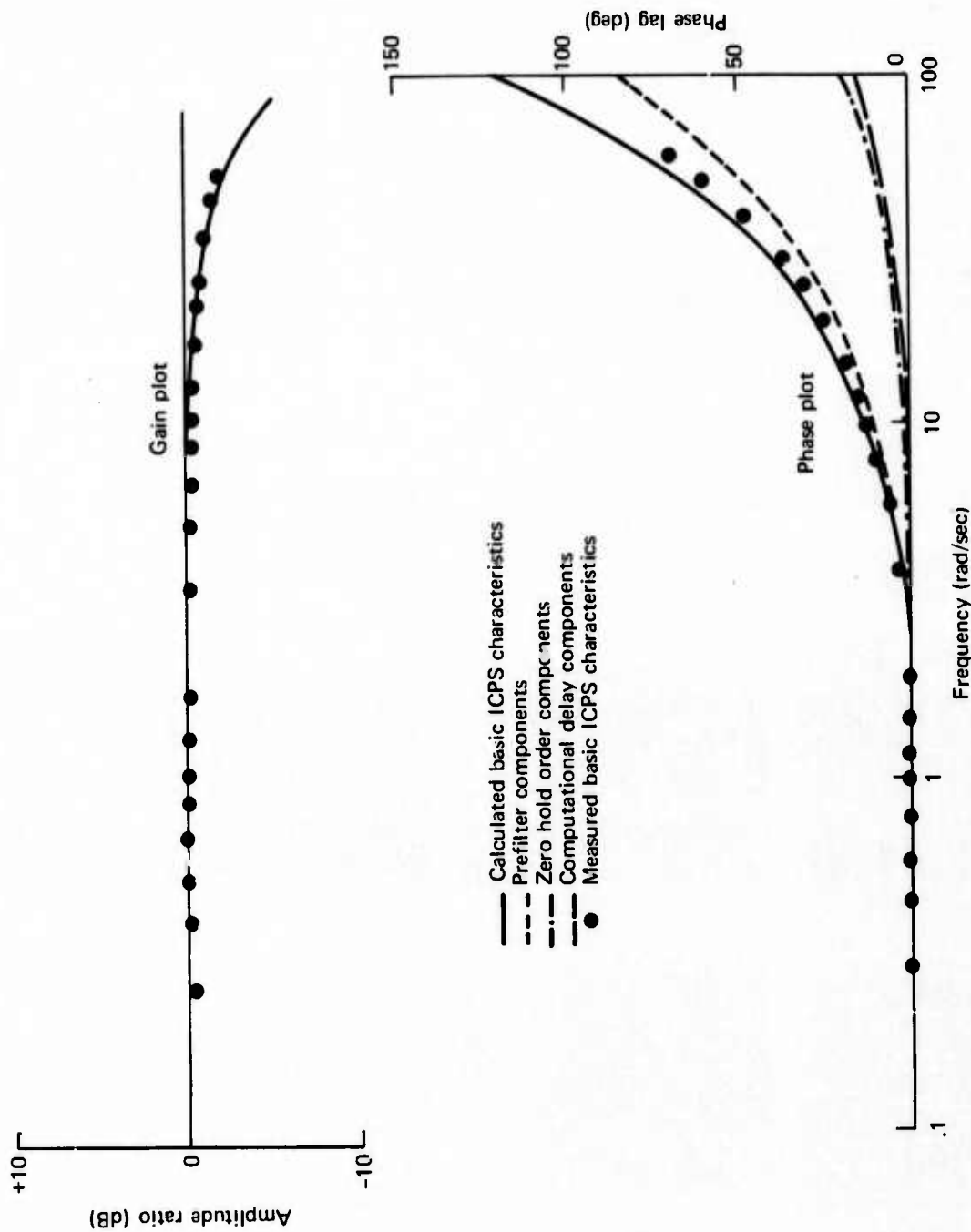


FIGURE 4-3.—BASIC ICPS HARDWARE FREQUENCY RESPONSE

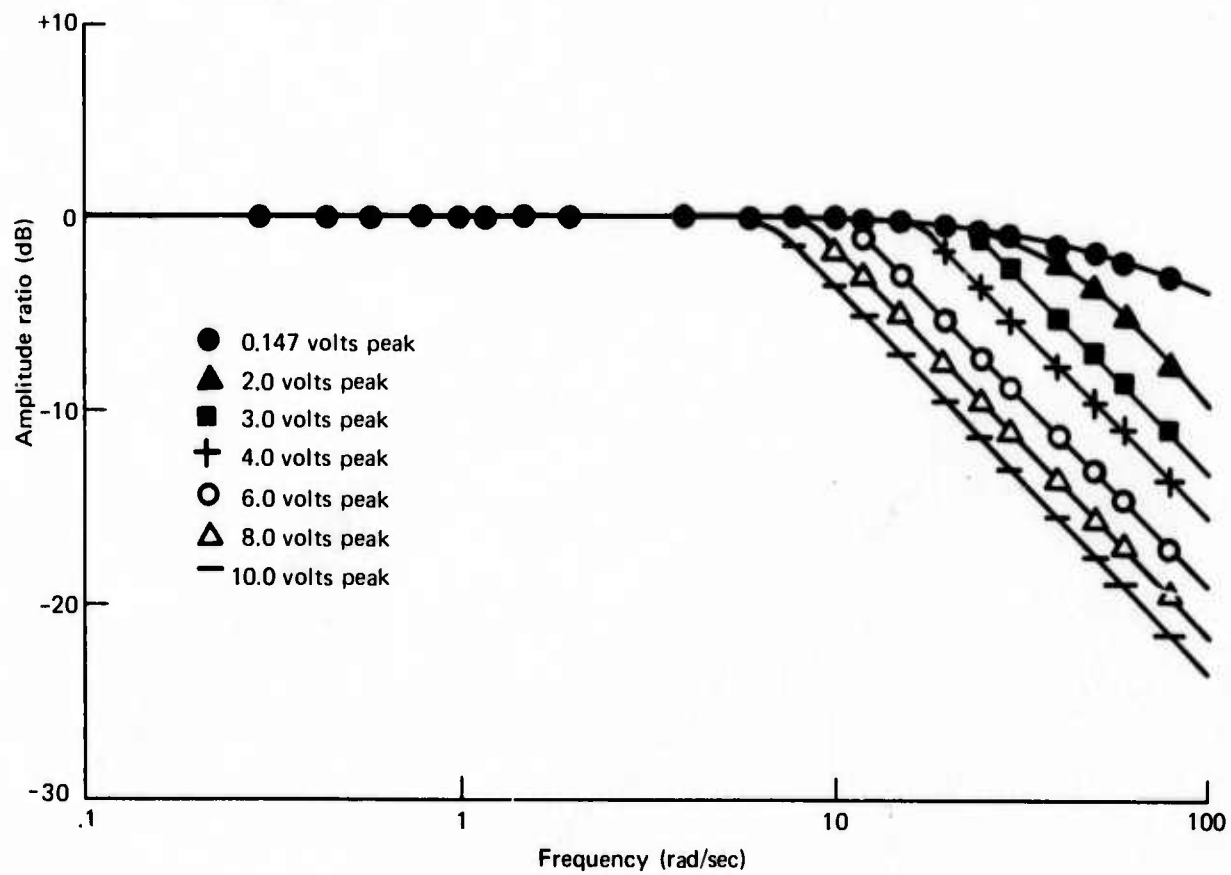


FIGURE 4-4.—ICPS EMPIRICAL SLEW RATE GAIN CHARACTERISTIC

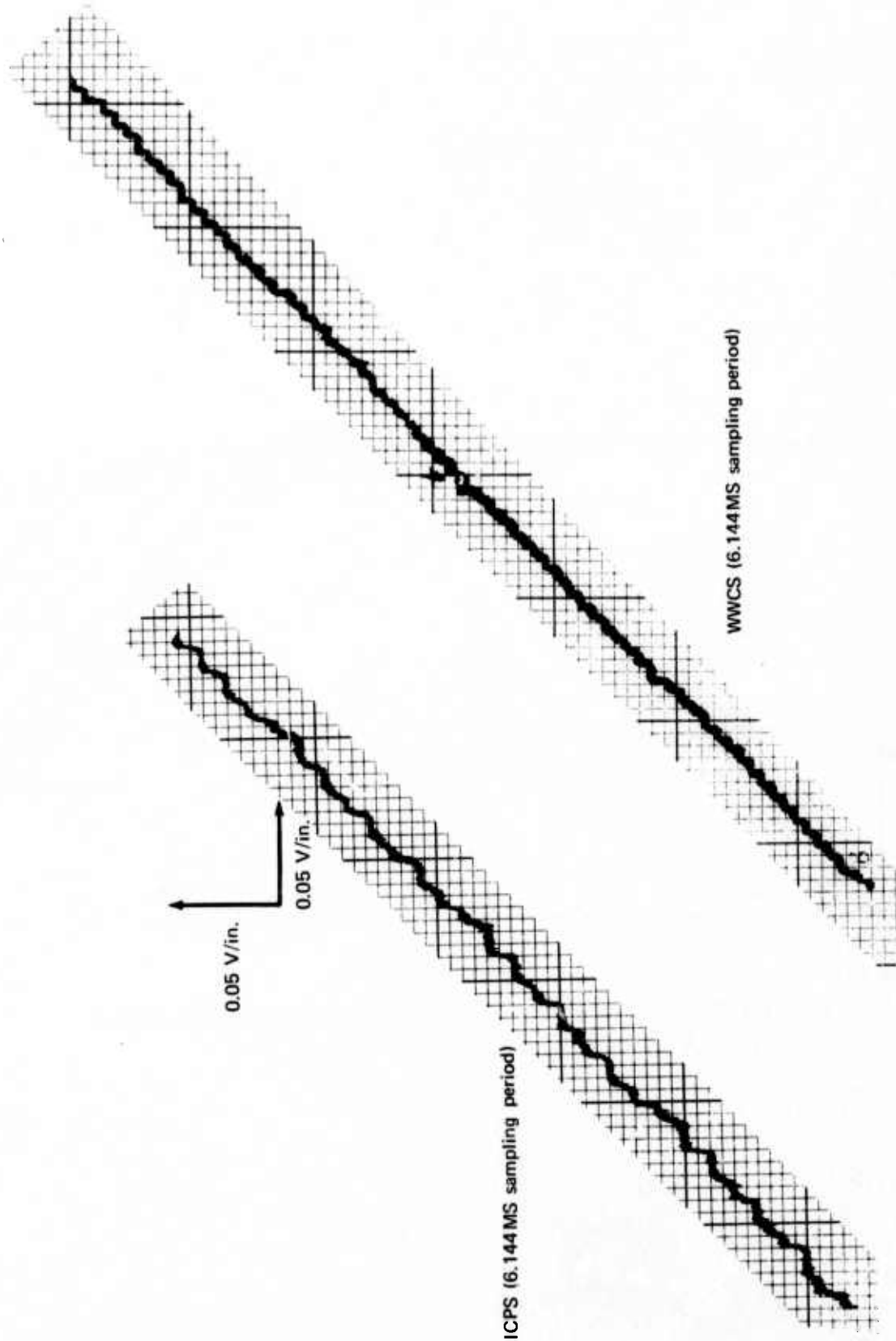


FIGURE 4-5.—LINEARITY CHARACTERISTICS OF ICPS AND WWCS HARDWARE

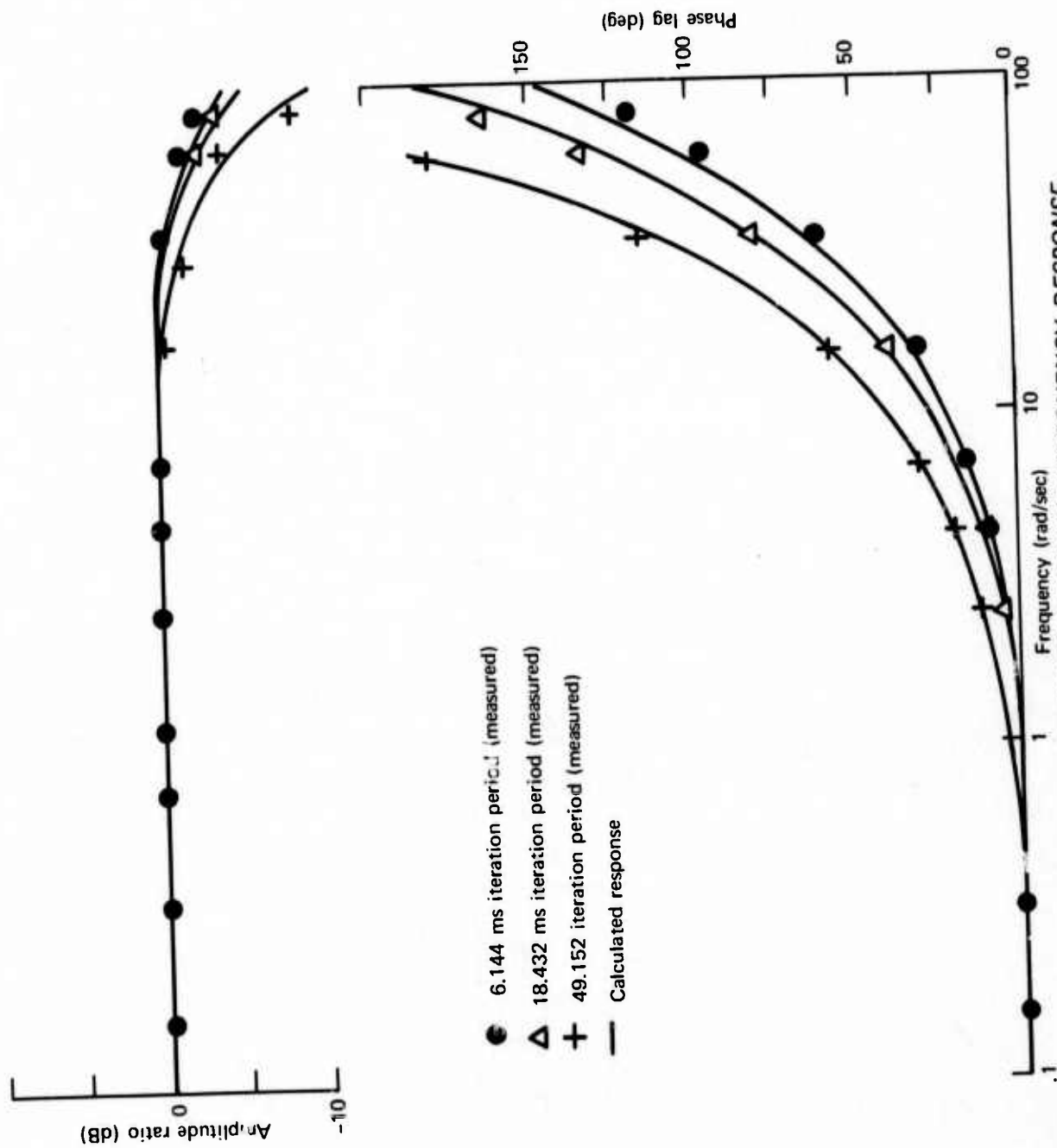


FIGURE 4-6.—BASIC WWCS HARDWARE FREQUENCY RESPONSE

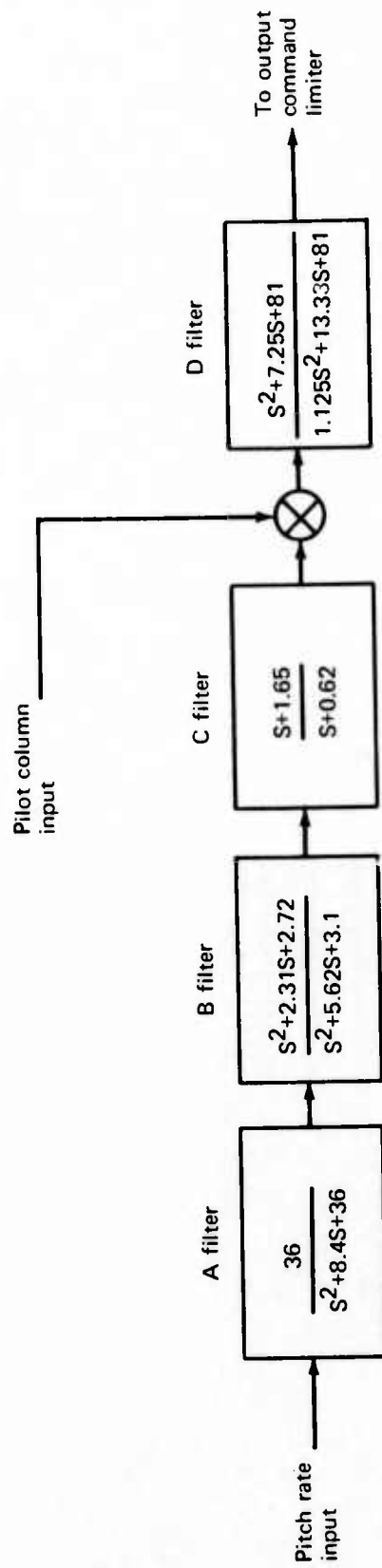


FIGURE 4-7.—HSAS FILTER

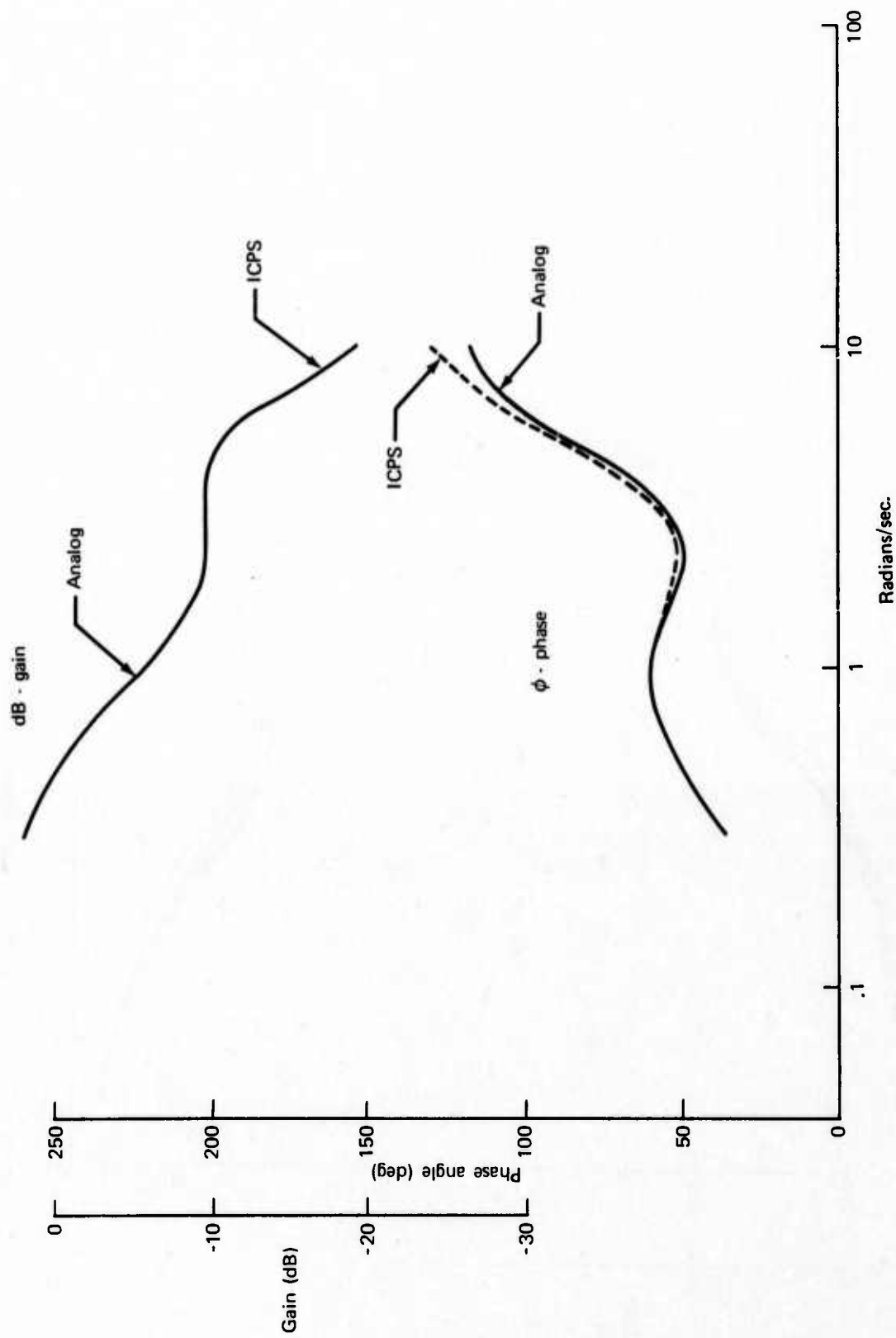


FIGURE 4-8.—ANALOG AND ICPS HSAS FREQUENCY RESPONSE

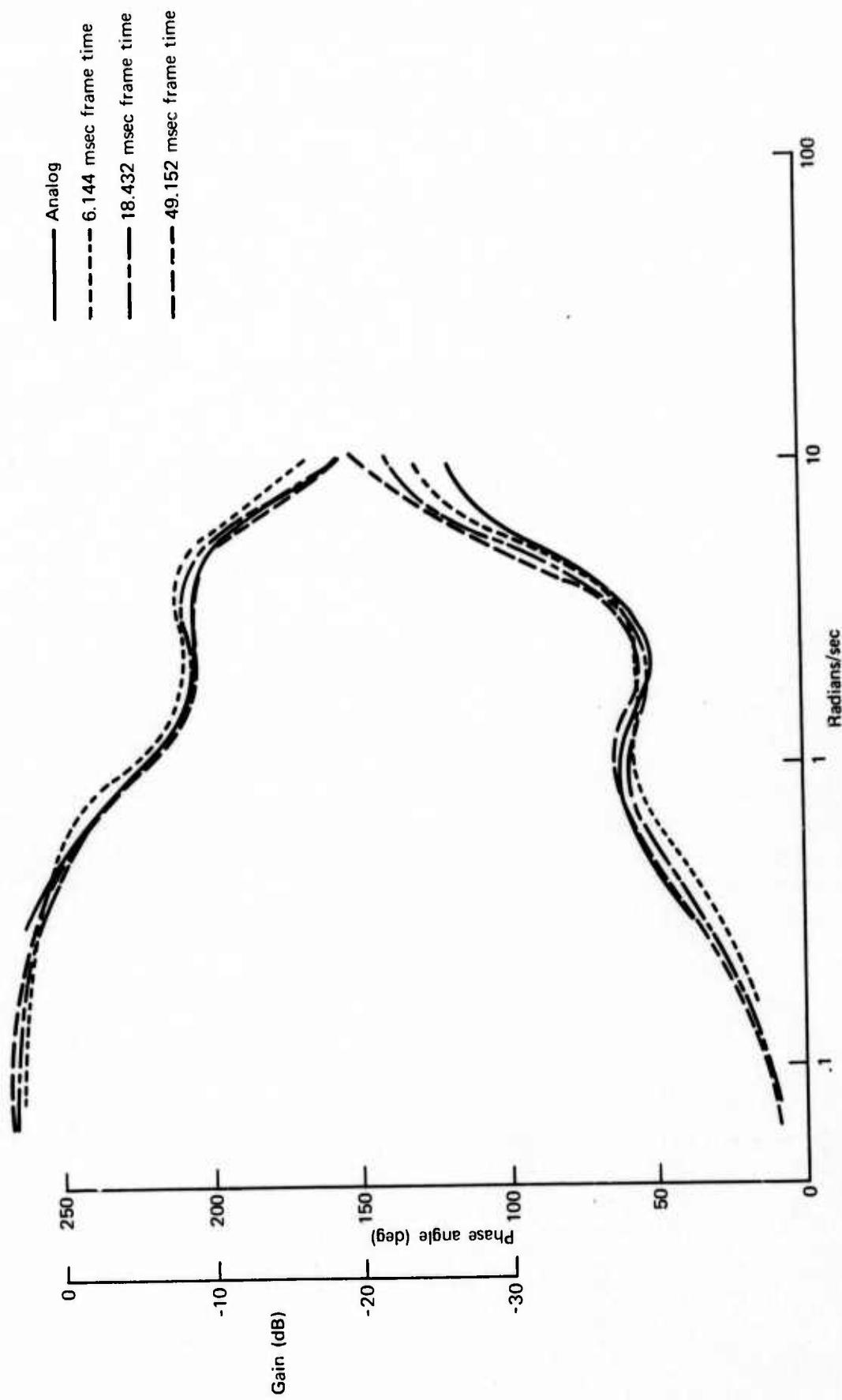


FIGURE 4-9.—WWCS HSAS FILTER FREQUENCY RESPONSE

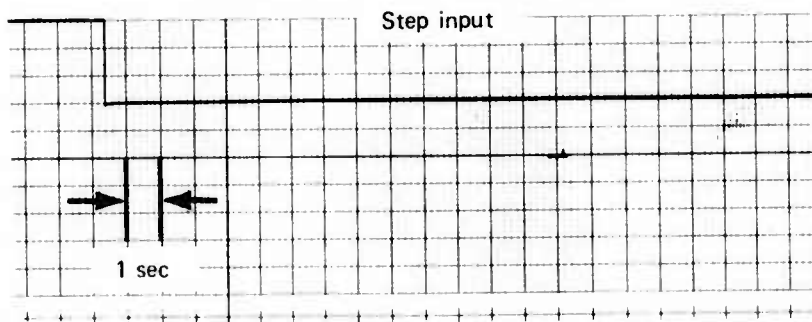
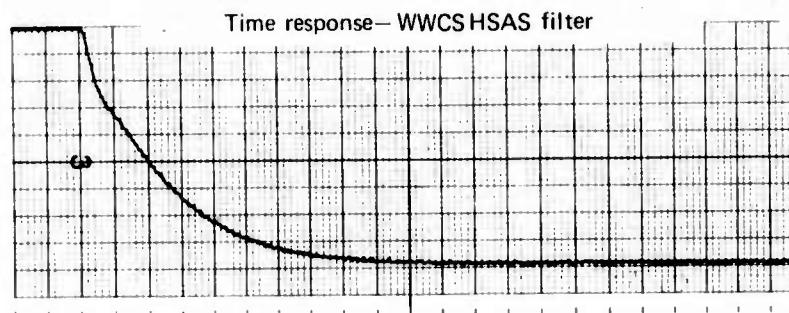
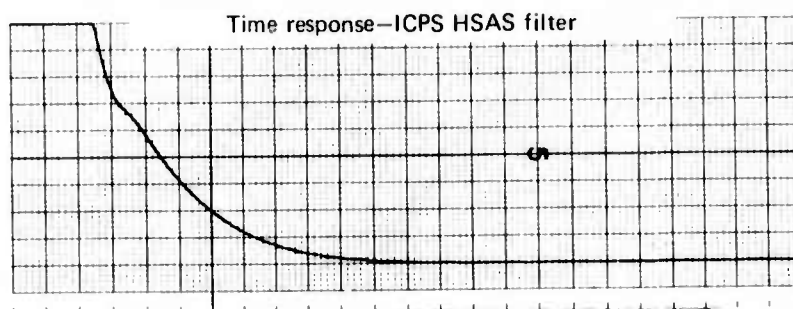
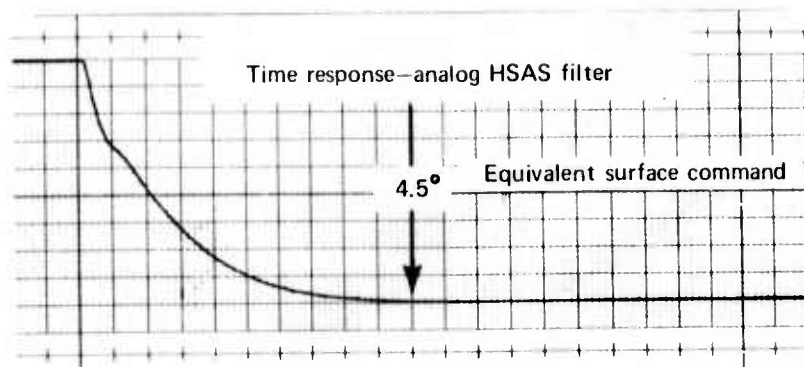


FIGURE 4-10.—HSAS FILTER STEP RESPONSE

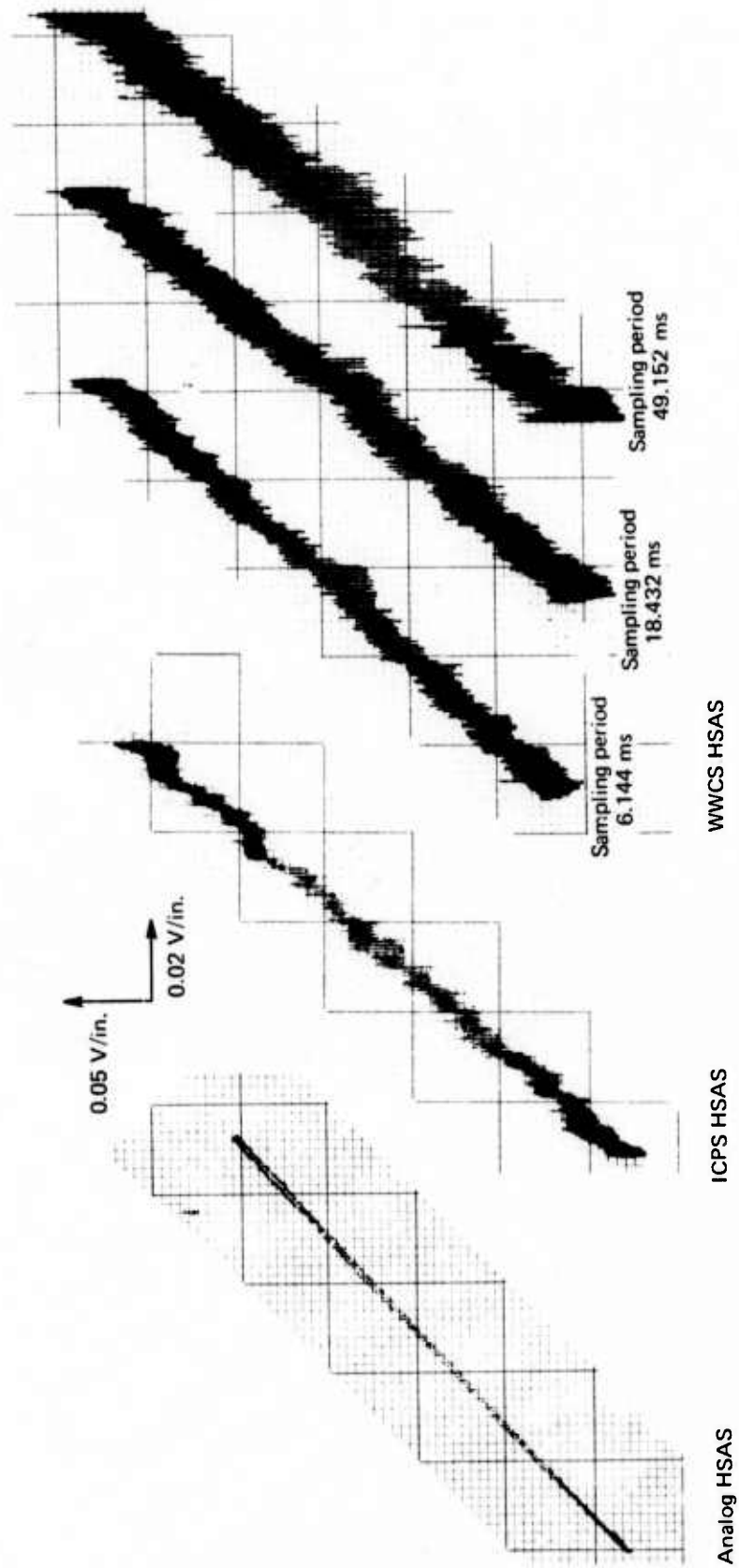
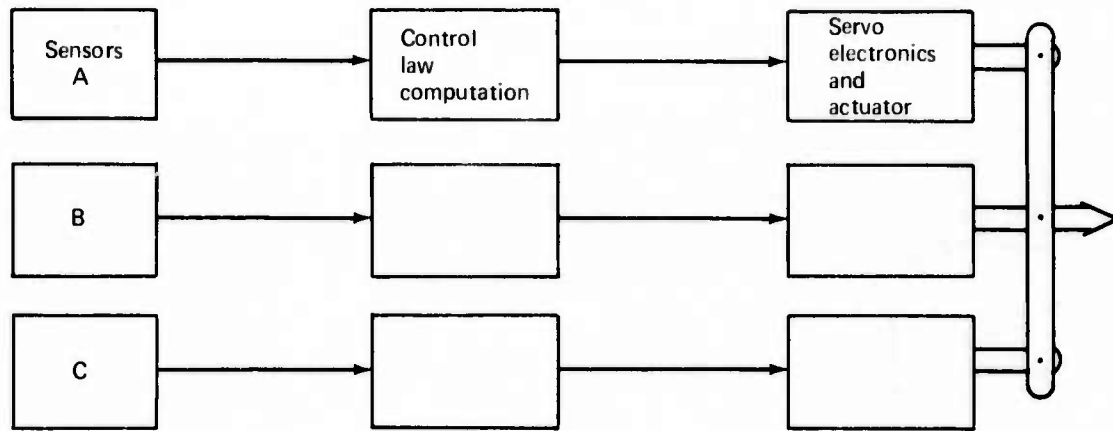
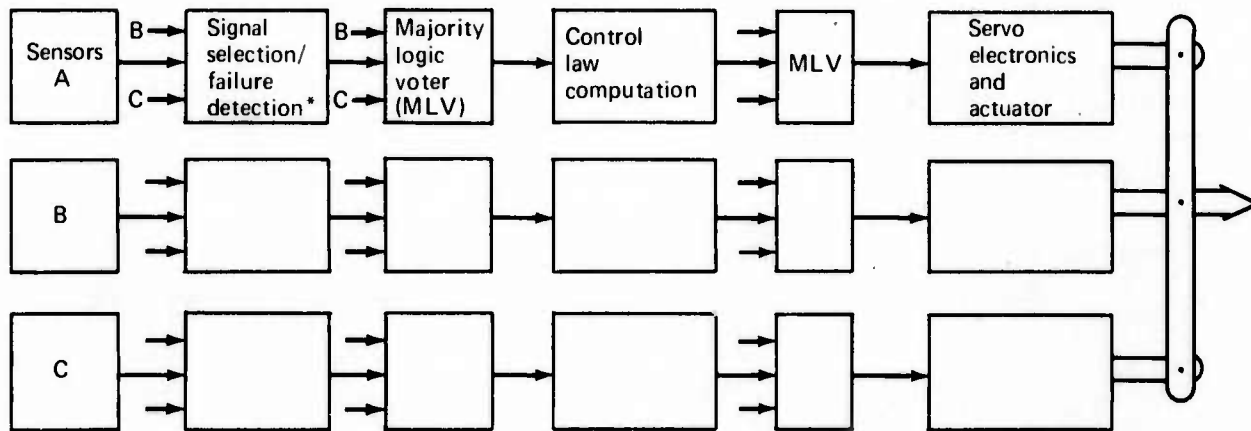


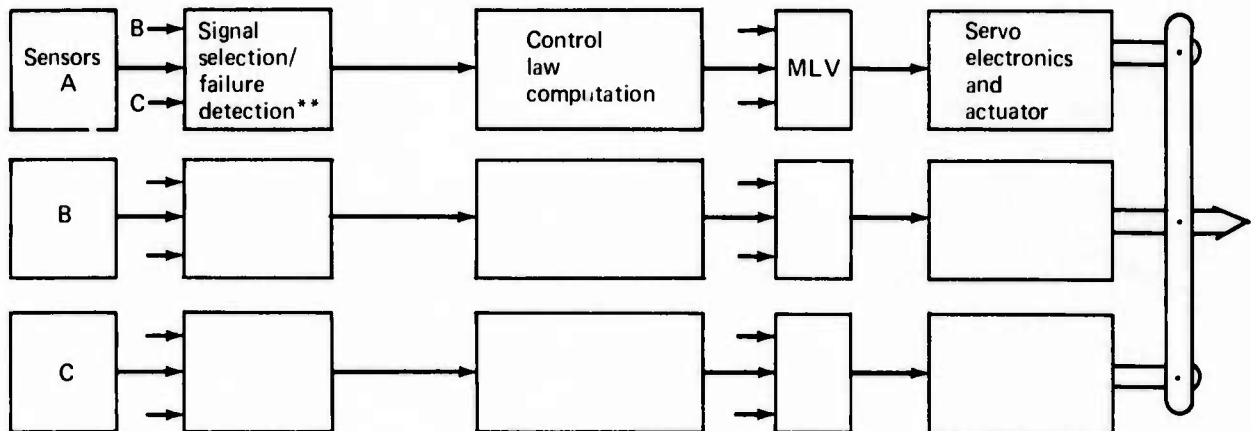
FIGURE 4-11.—COMPARISON OF HSAS FILTER RESIDUAL NOISE CHARACTERISTICS
(0.001 Hz SINE WAVE INPUT)



Analog System



Incremental Control Processor System



Whole Word Computer System

* Hardware implemented
 ** Software implemented

FIGURE 4-12.—REDUNDANT DATA PATH BLOCK DIAGRAM

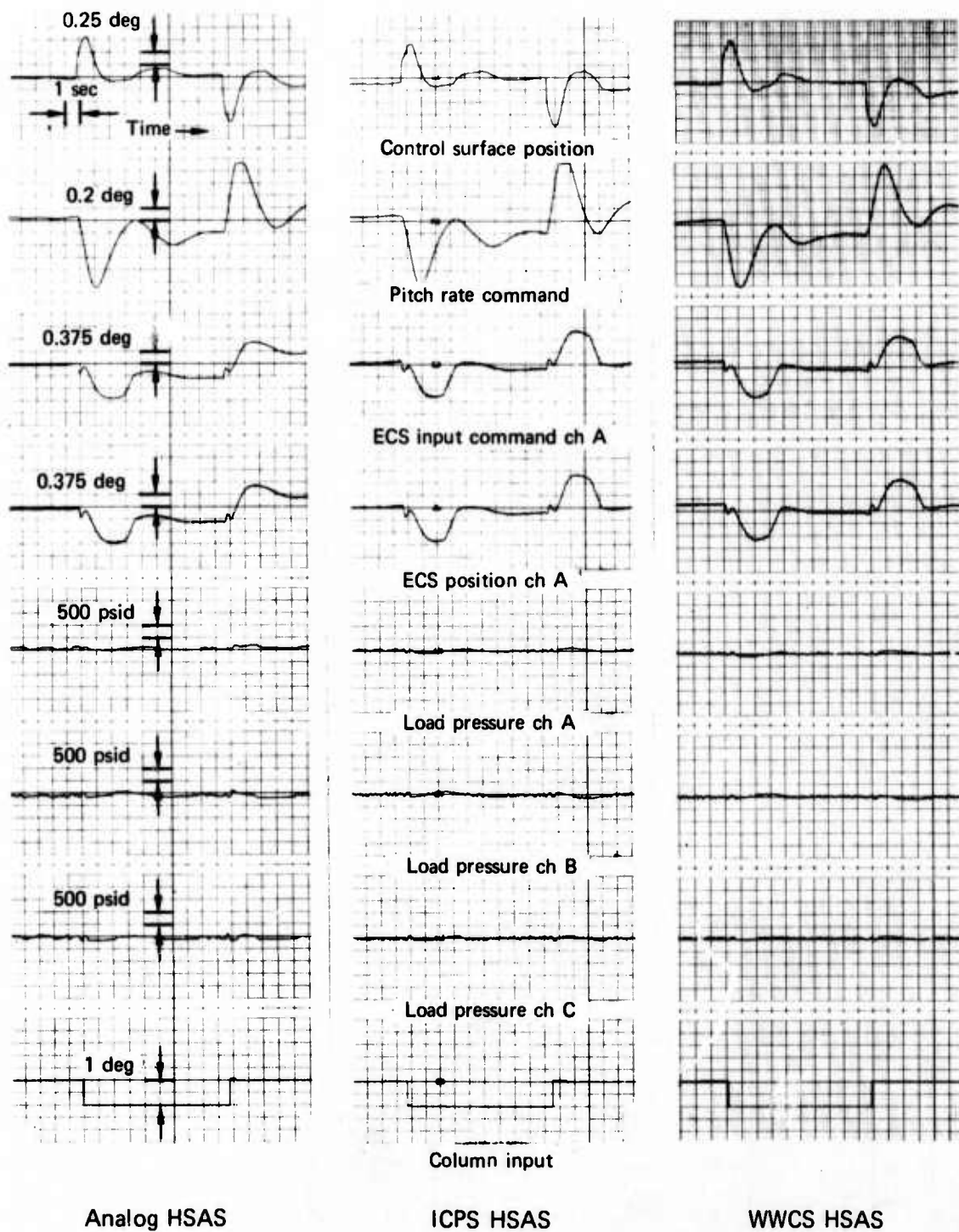


FIGURE 4-13.—COMPARISON OF HSAS PERFORMANCE IN TRIPLE OPERATION (WITHOUT SENSOR OFFSETS)

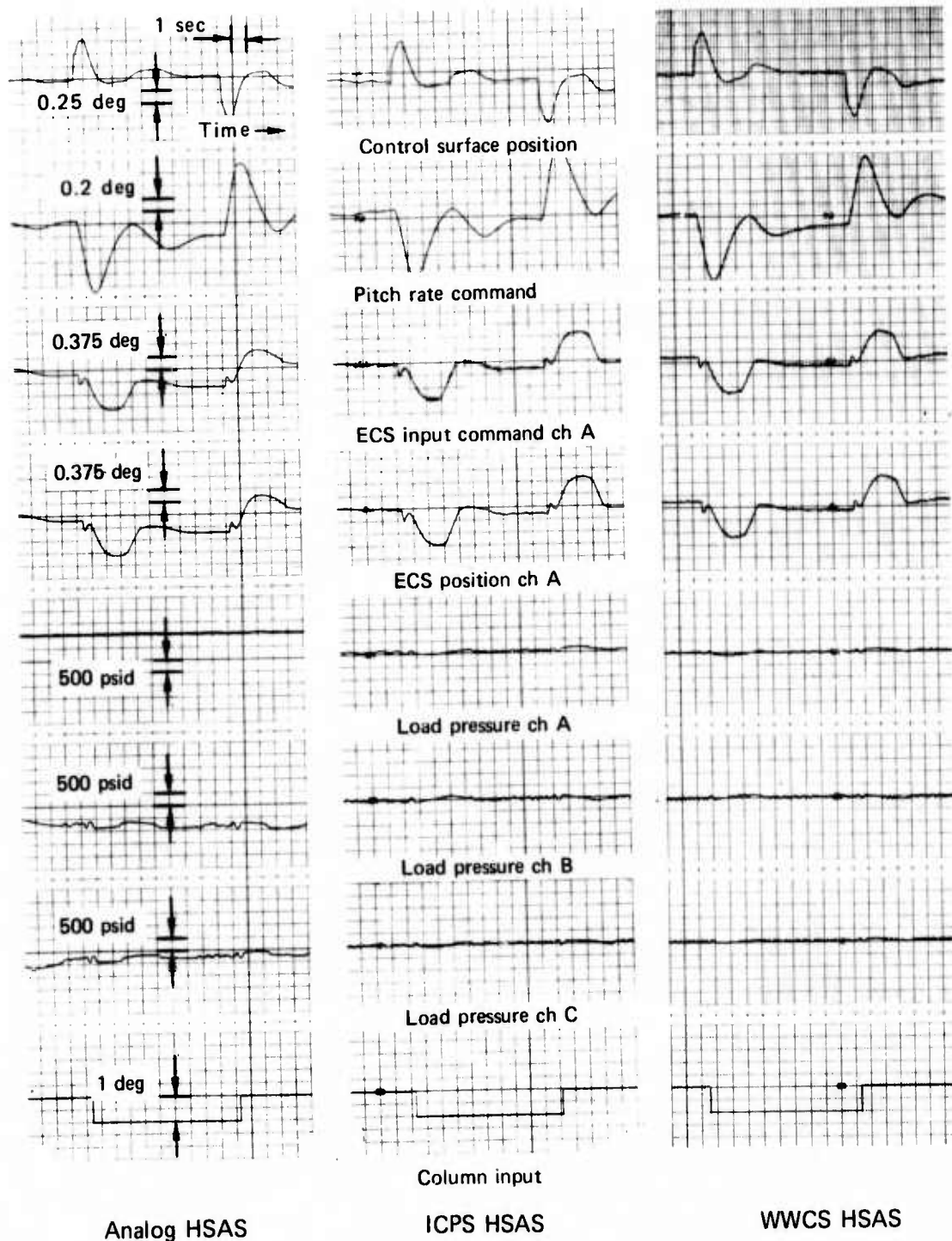


FIGURE 4-14.—COMPARISON OF HSAS PERFORMANCE IN TRIPLE OPERATION WITH PITCH RATE SENSOR OFFSETS ($\pm 0.5^\circ$ /SEC.)

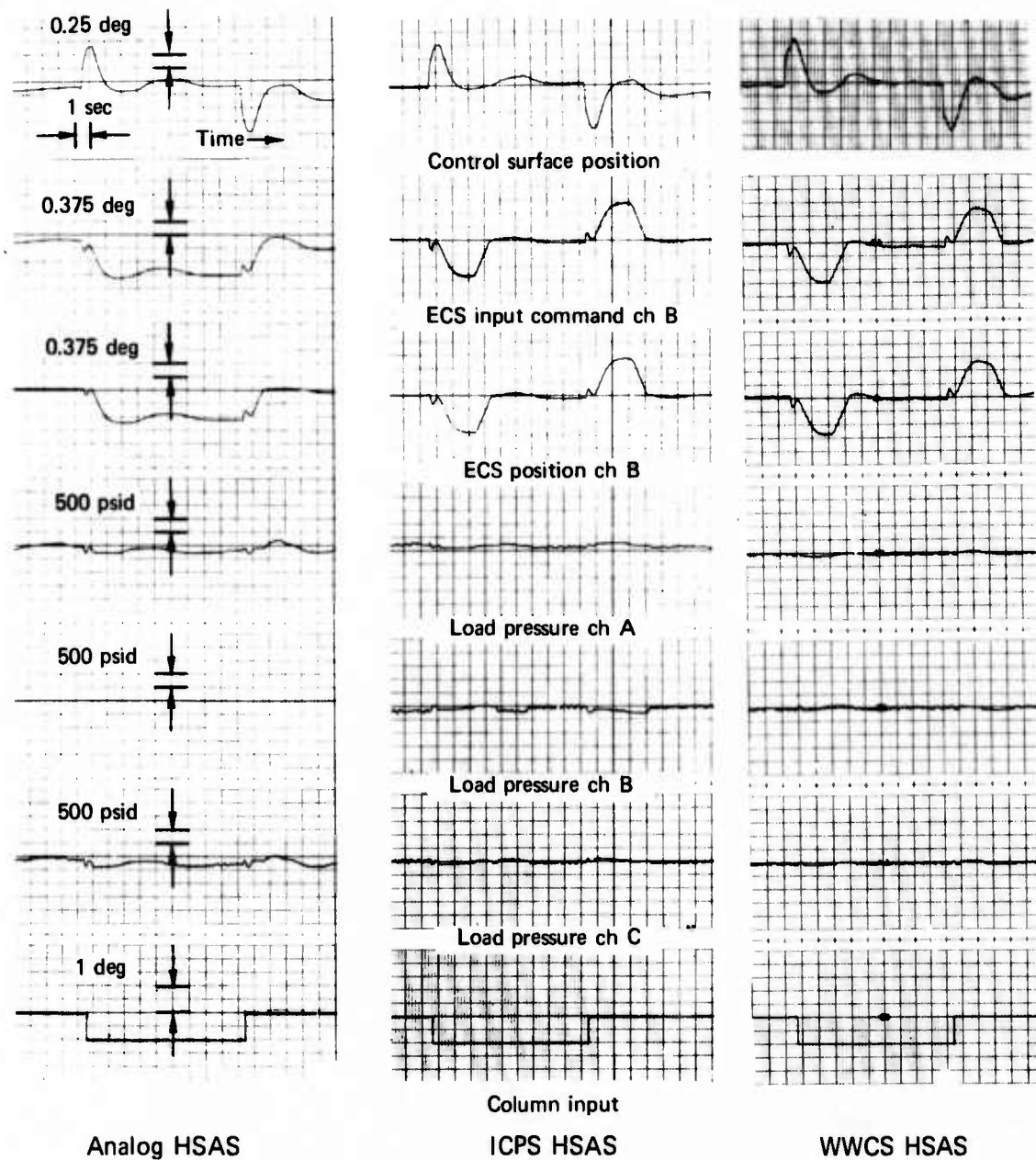


FIGURE 4-15.—HSAS PERFORMANCE WITH ONE PITCH RATE SENSOR FAILURE

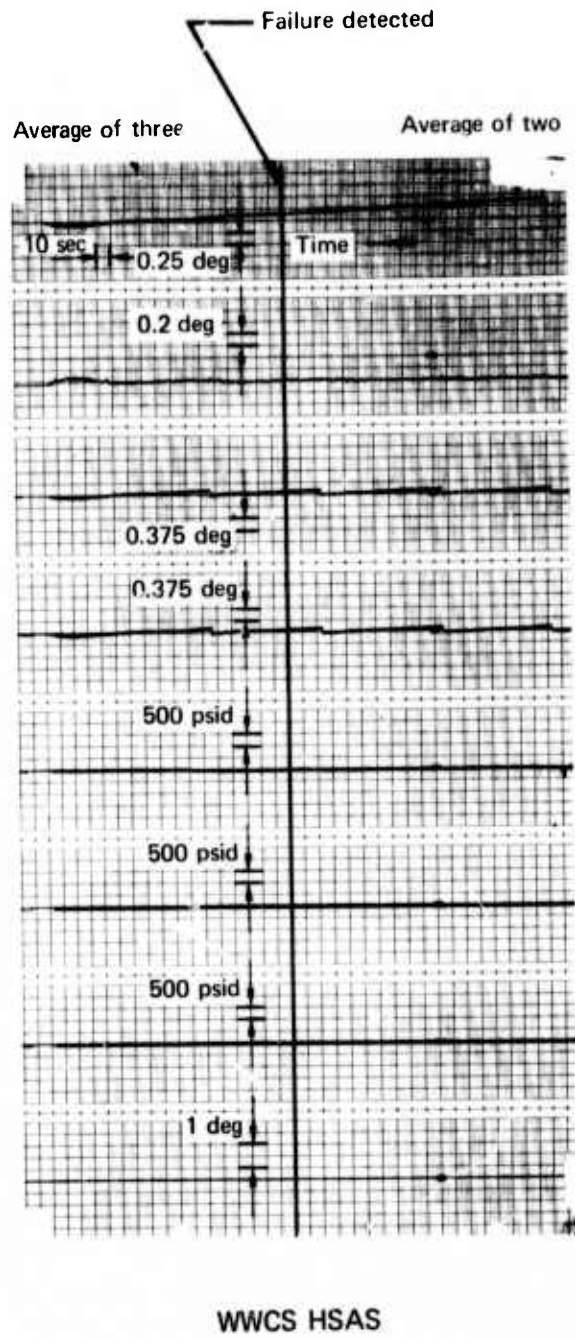
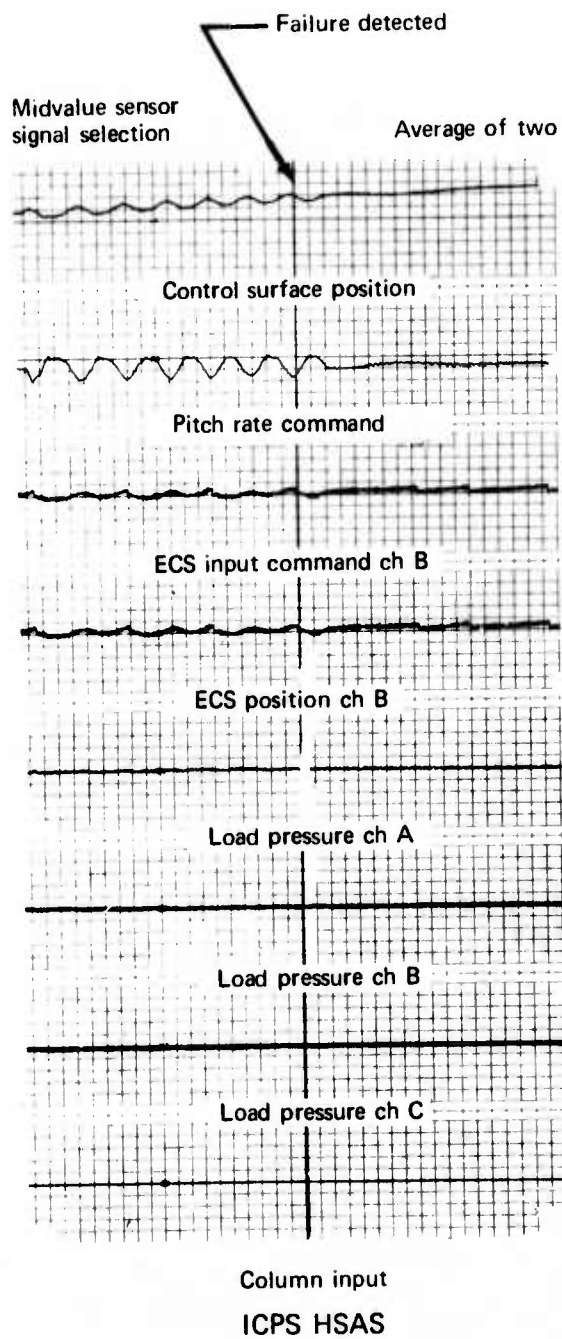


FIGURE 4-16.—ICPS AND WWCS HSAS RESPONSE WITH PASSIVE PITCH RATE SENSOR FAILURE

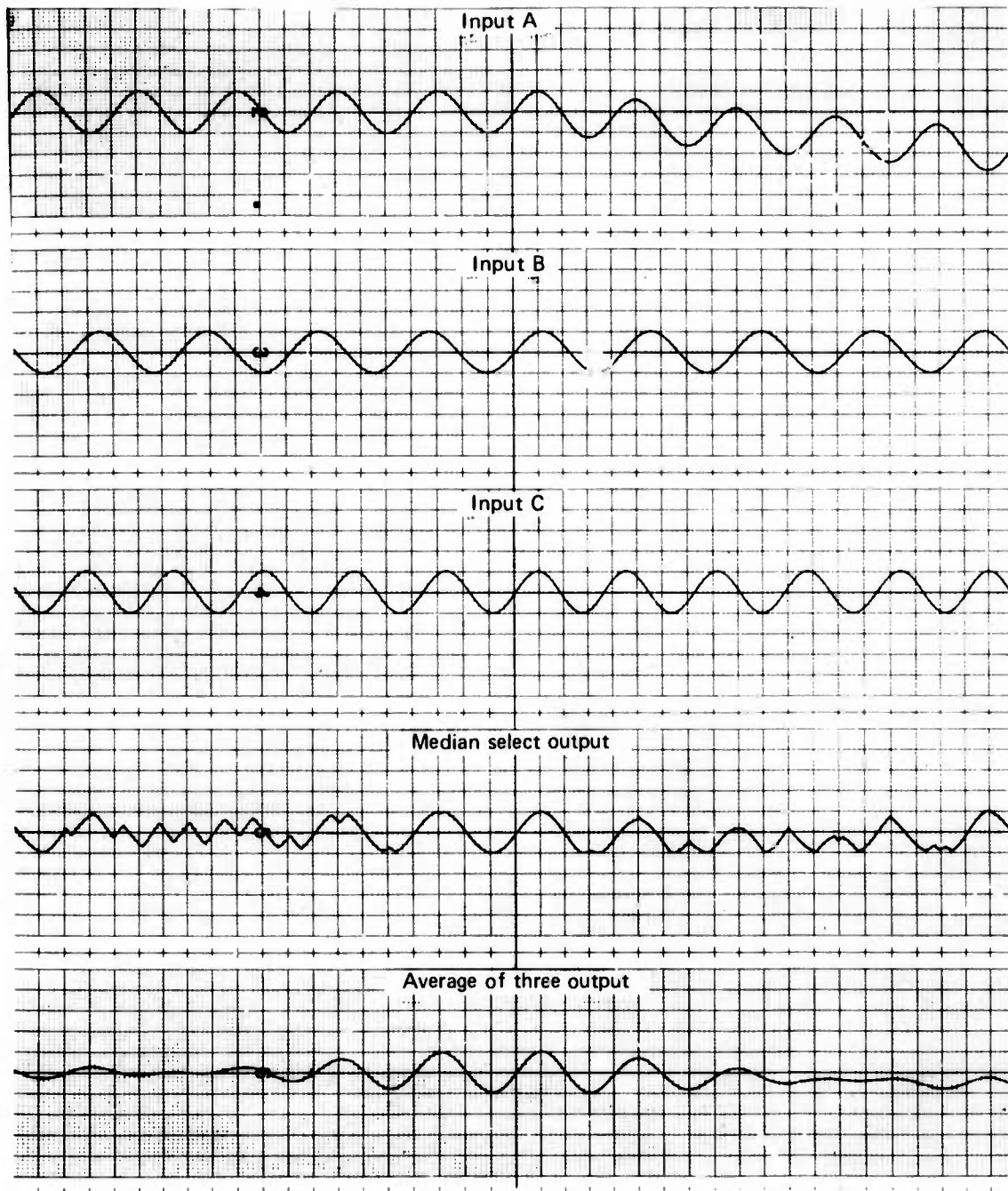


FIGURE 5-1.—MEDIAN SELECTION AND AVERAGE OUTPUT FOR THREE OUT-OF-PHASE INPUT SIGNALS

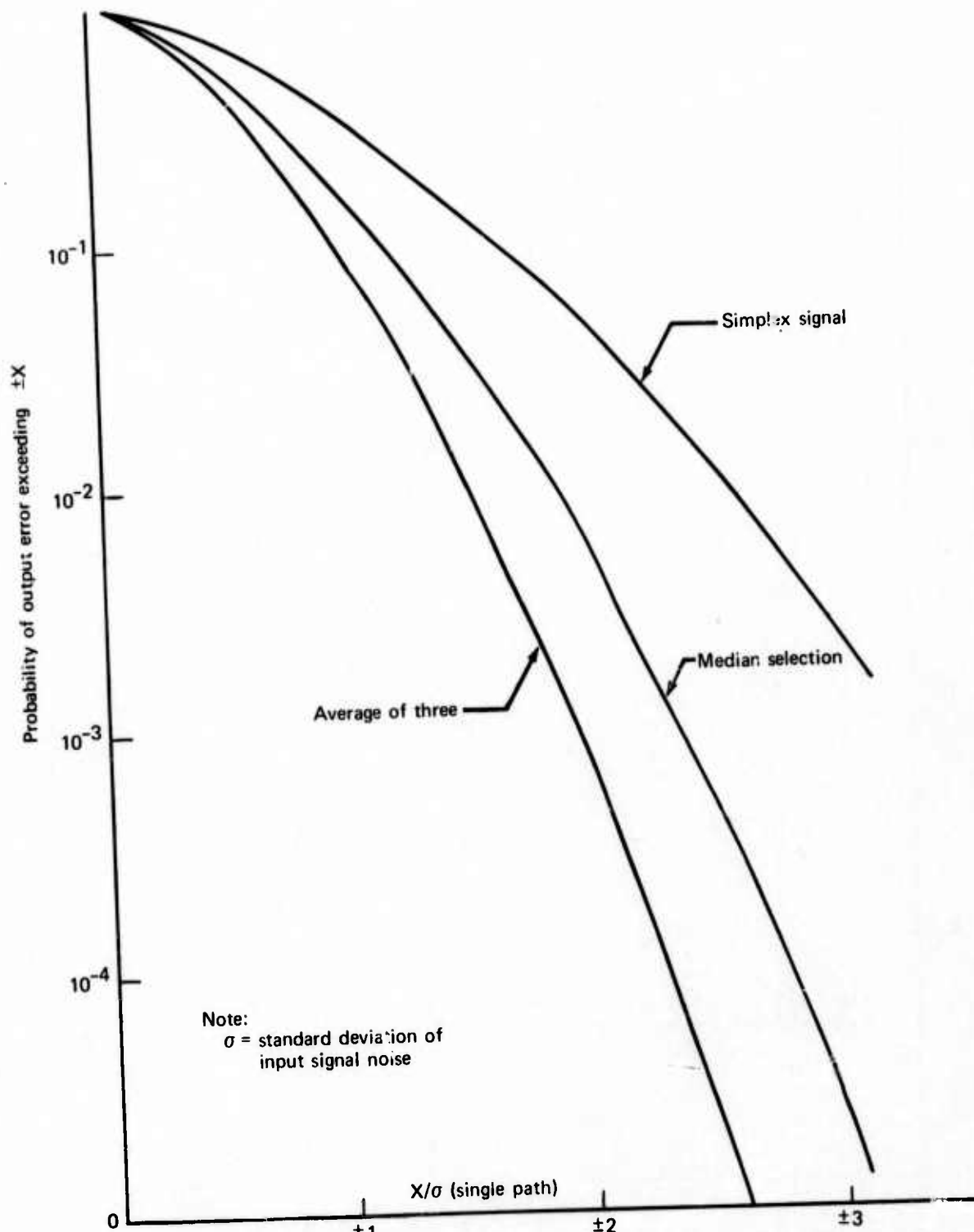


FIGURE 5-2.—MEDIAN SELECTION AND AVERAGE ALGORITHM OUTPUTS WITH RANDOM NOISE INPUTS

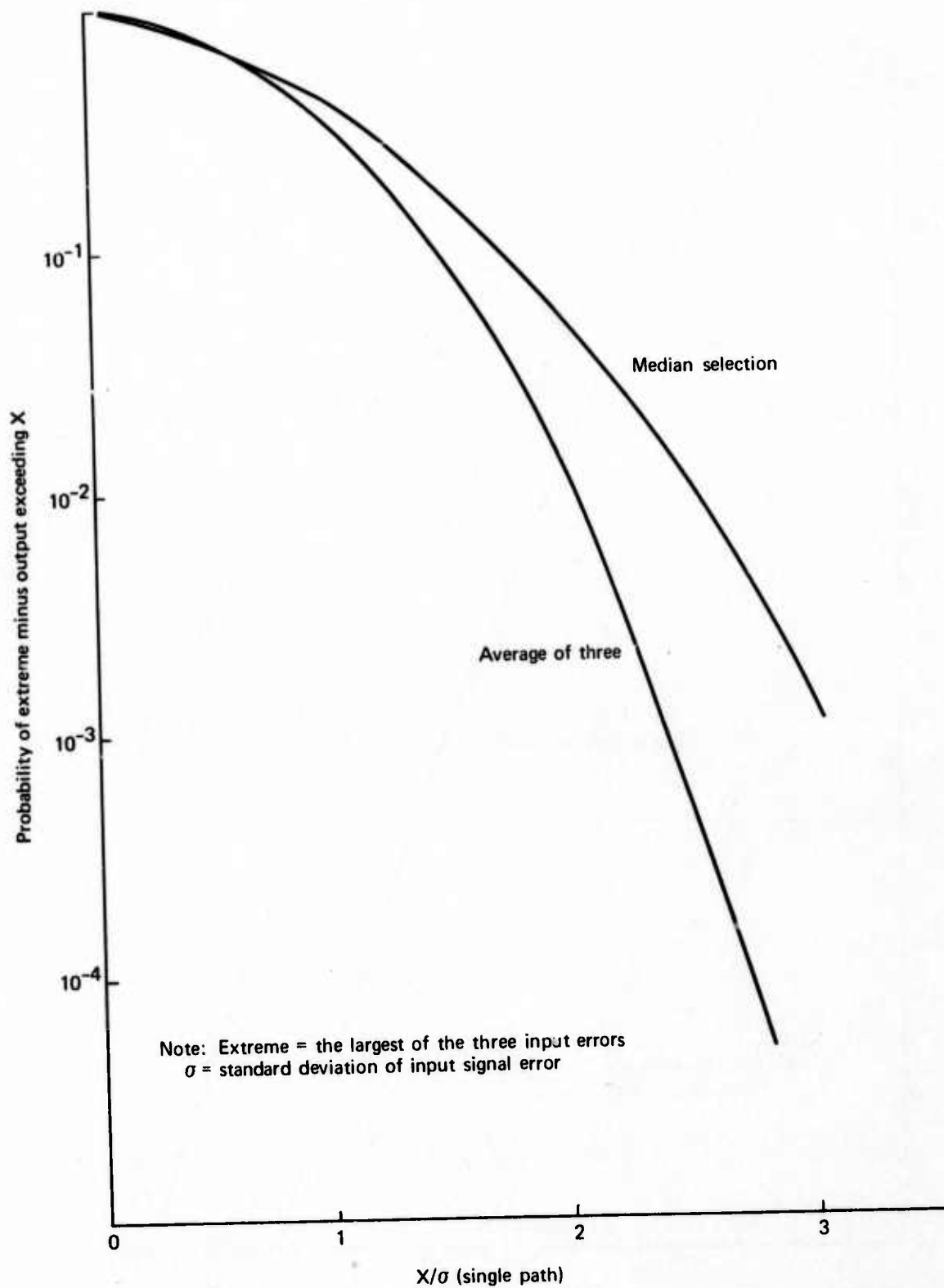


FIGURE 5-3.—SIGNAL SELECTION OUTPUT TO INPUT ERROR CHARACTERISTICS

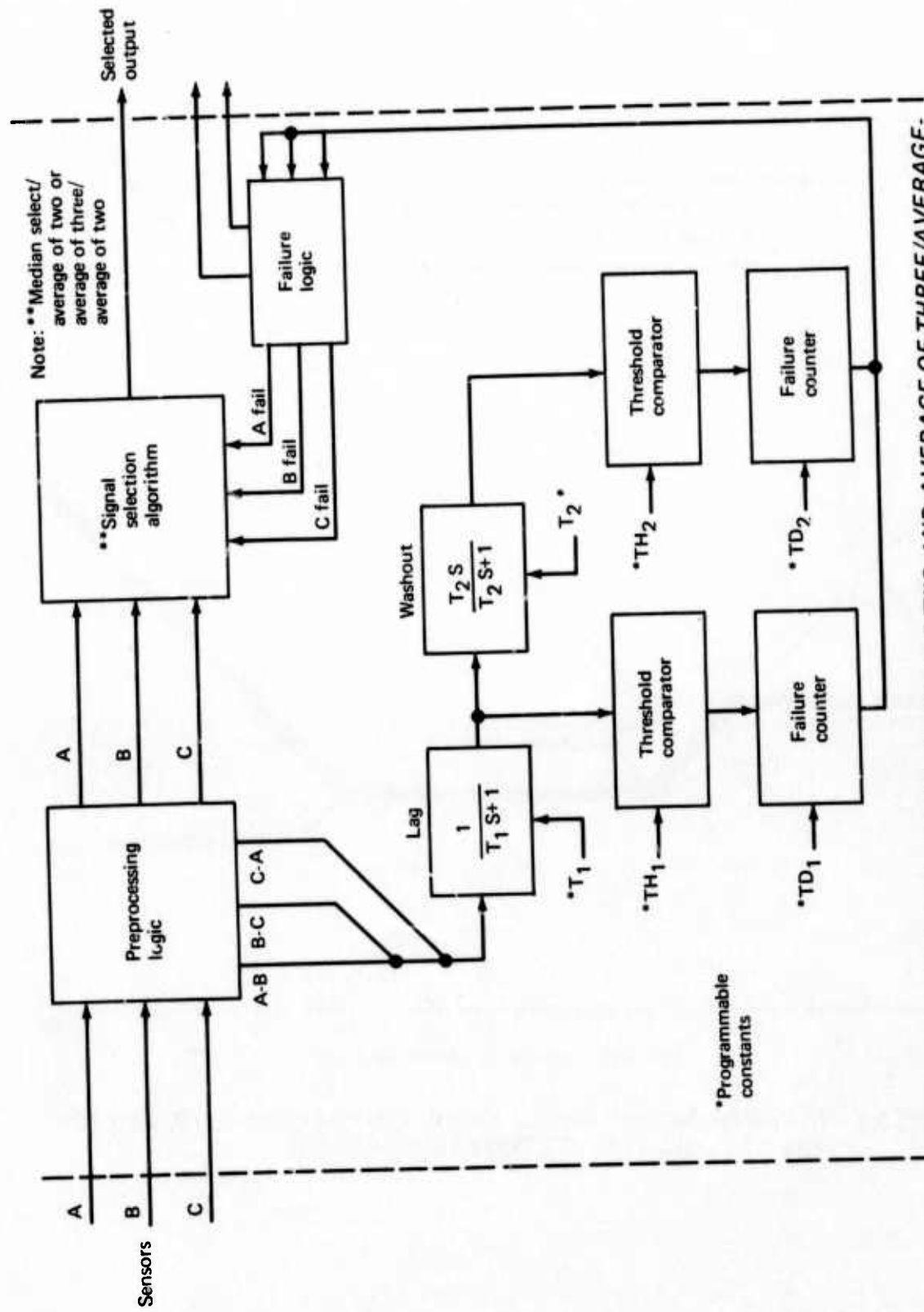


FIGURE 5-4. -MEDIAN-SELECT/AVERAGE-OF-TWO AND AVERAGE-OF-THREE/AVERAGE-OF-TWO SSFD ALGORITHM

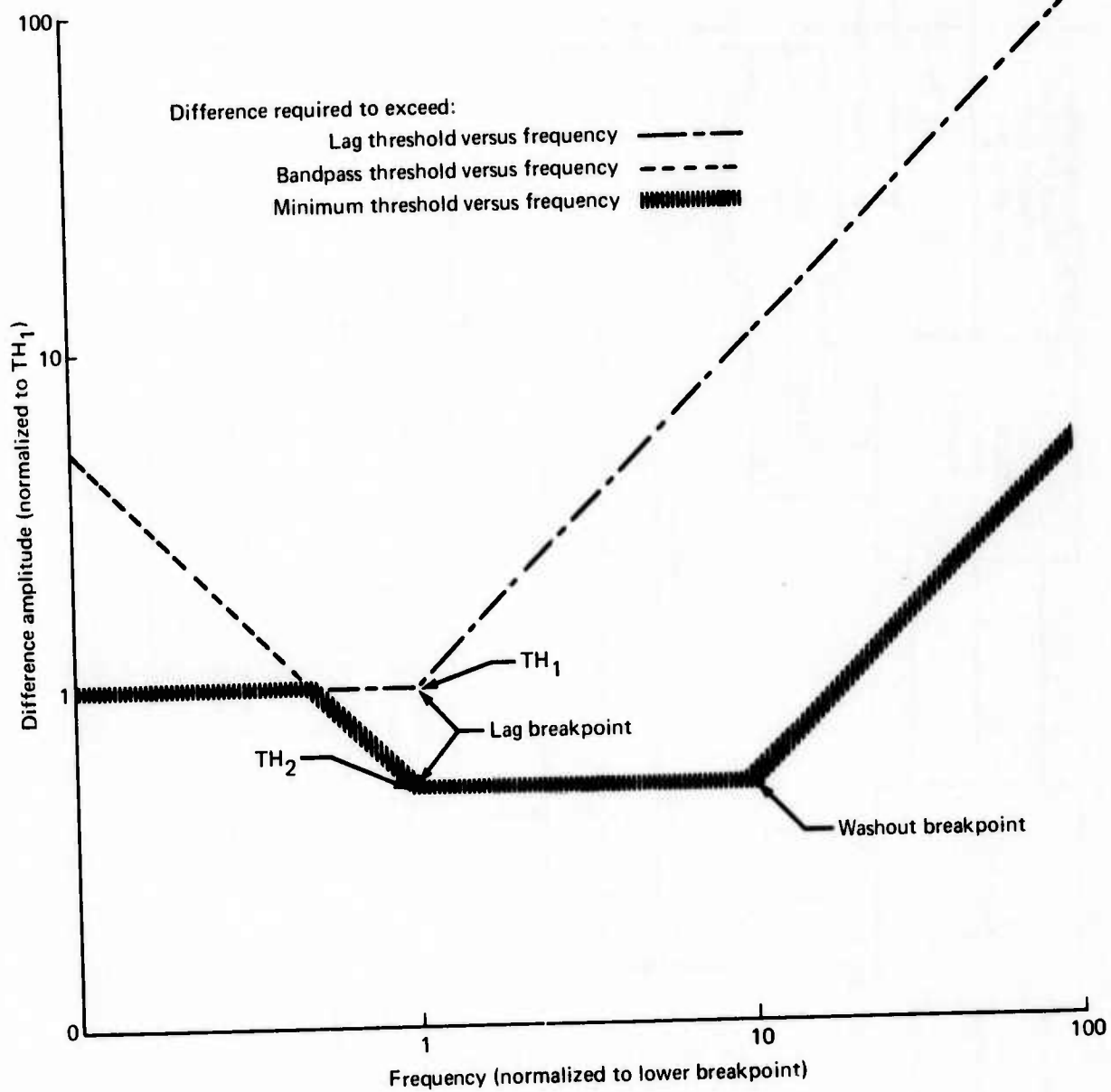


FIGURE 5-5.—THRESHOLD AMPLITUDE OF CROSS INPUT FAILURE DETECTOR FOR $TH_1 = TH_2$ AND $T_1 = 10T_2$ (REFERENCE FIG. 5-4)

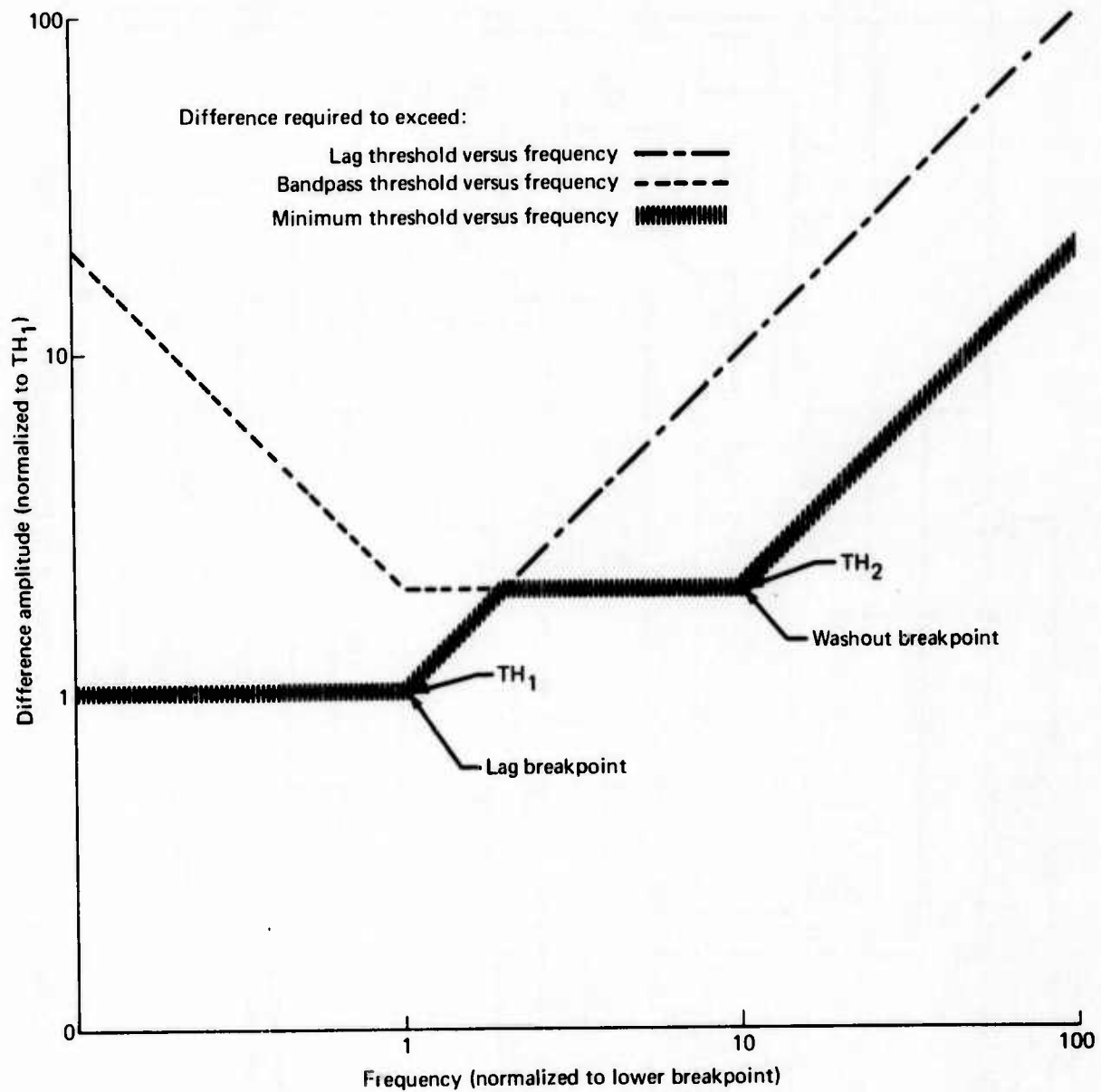


FIGURE 5-6.—THRESHOLD AMPLITUDE OF CROSS INPUT FAILURE DETECTOR FOR $TH_1 > TH_2$ AND $T_1 = 10T_2$ (REFERENCE FIG. 5-4)

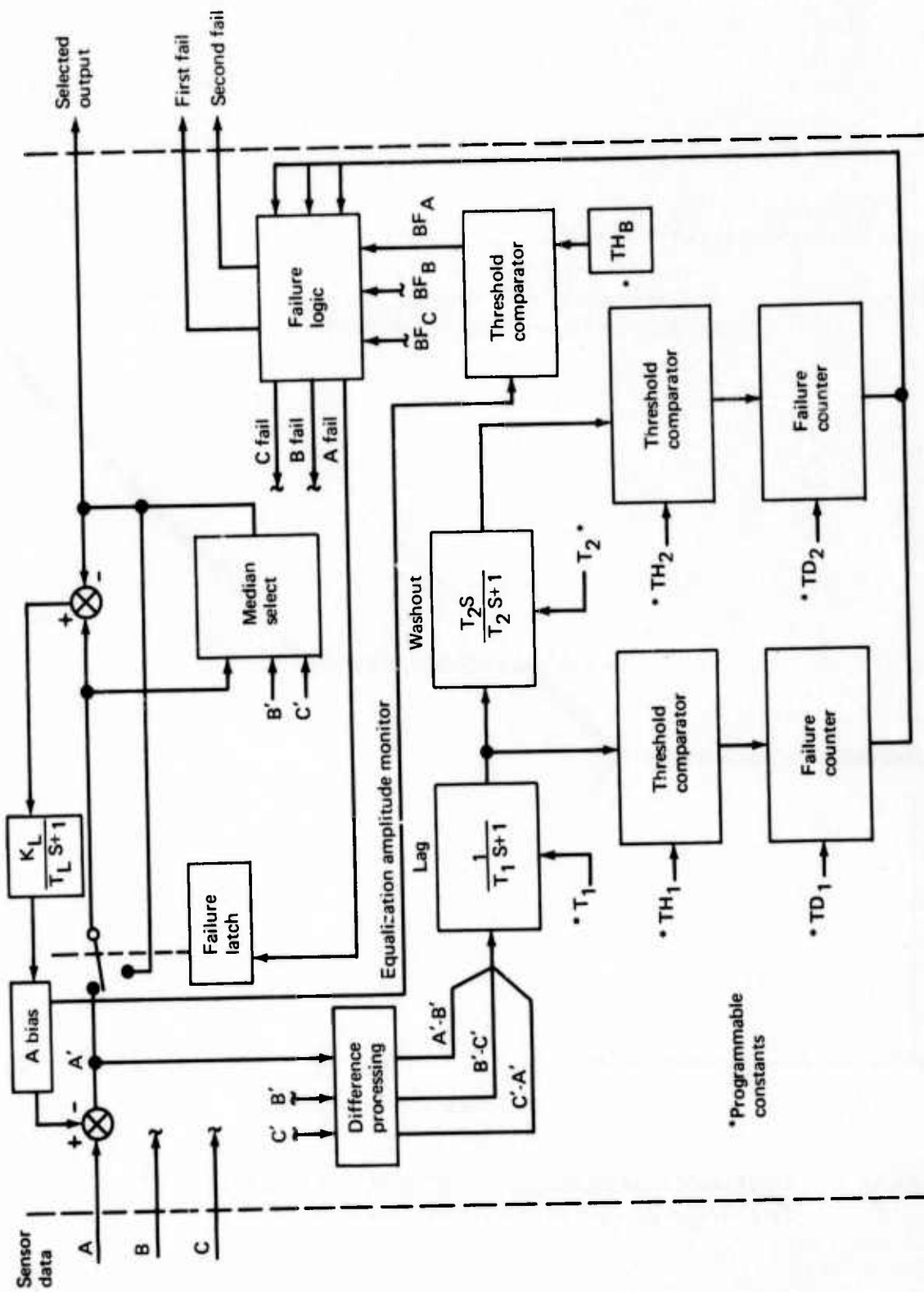


FIGURE 5-7.—LAG EQUALIZED MEDIAN SELECTION SSFD ALGORITHM

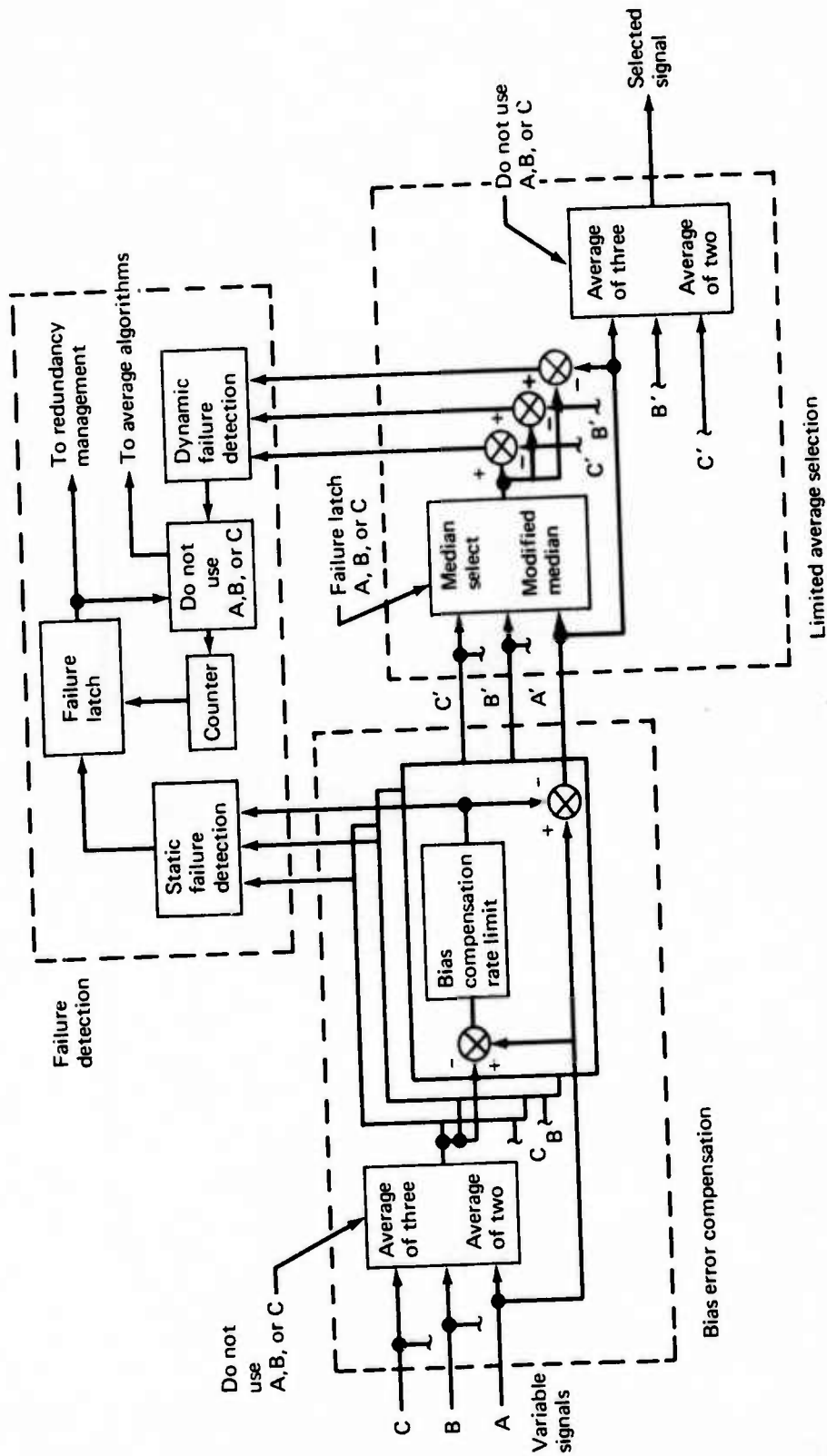


FIGURE 5-8.—COMPENSATED LIMITED AVERAGE SSFD ALGORITHM

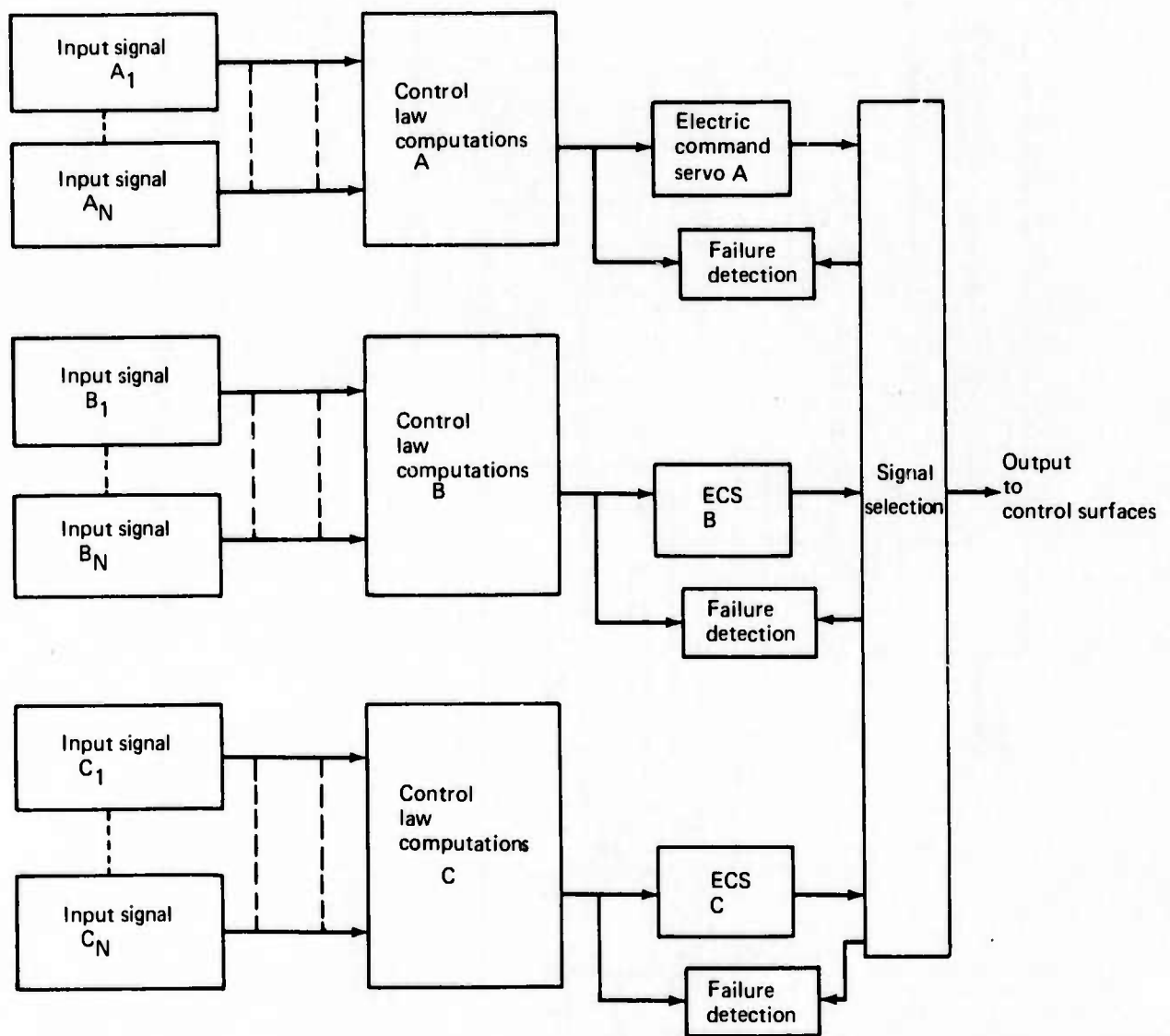


FIGURE 5-9.—ANALOG SYSTEM SSFD CONFIGURATION

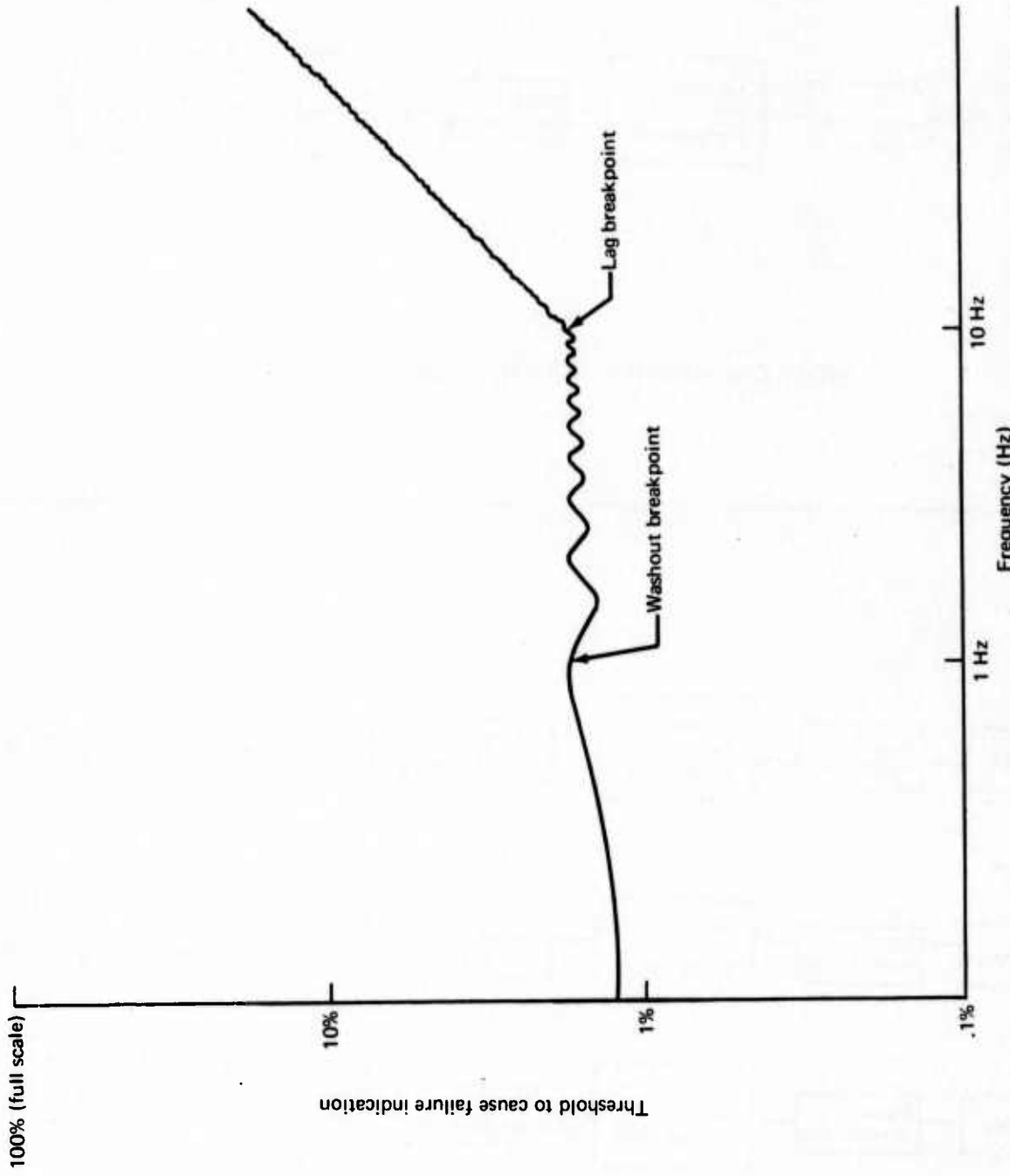


FIGURE 5-10.—OSCILLATORY FAILURE DETECTION CHARACTERISTICS OF ICPS VARIABLE SSFD

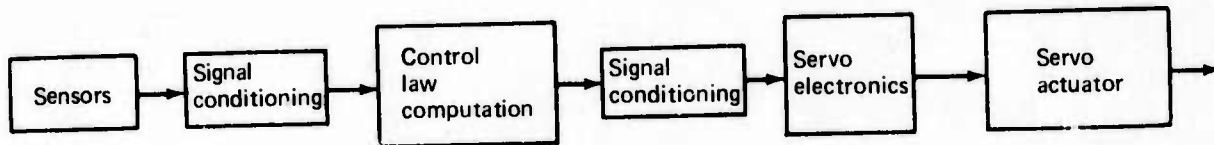


FIGURE 5-11.—SINGLE CHANNEL CONFIGURATION

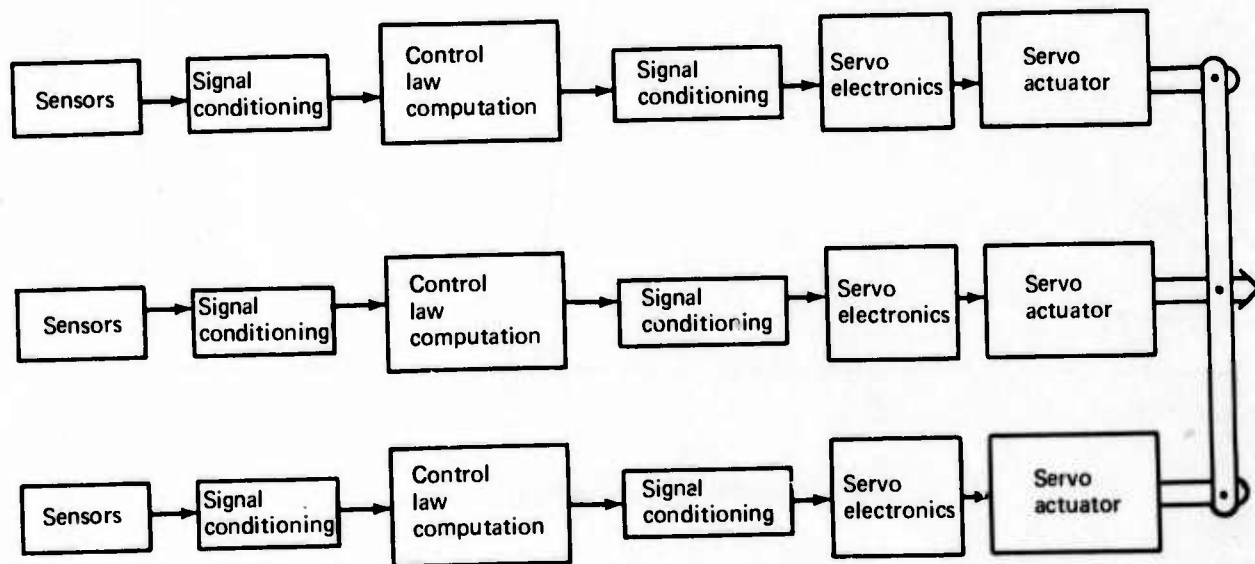


FIGURE 5-12.—BRICK-WALL CONFIGURATION

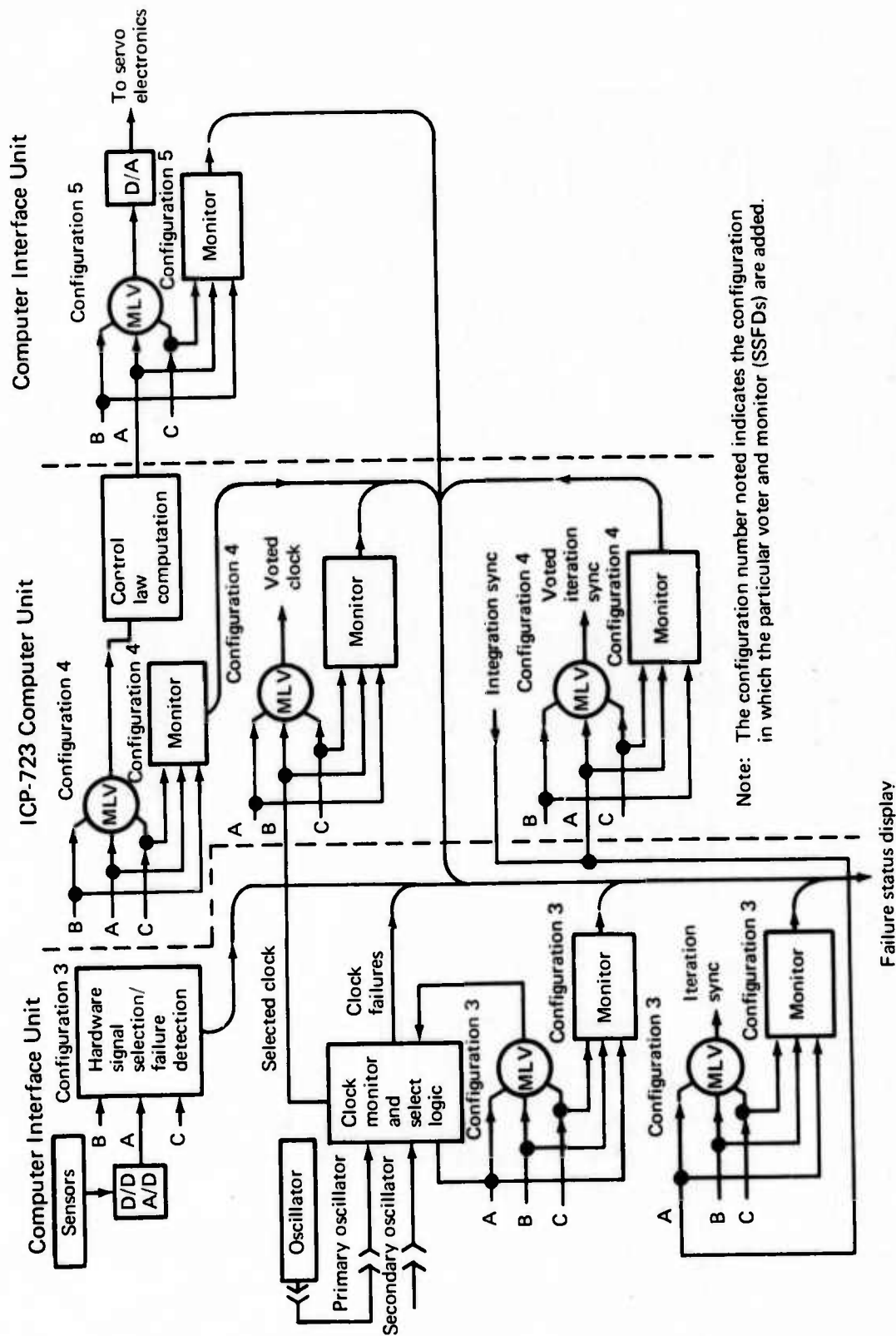


FIGURE 5-13.—ICPS VOTING/MONITORING SCHEMATIC DIAGRAM

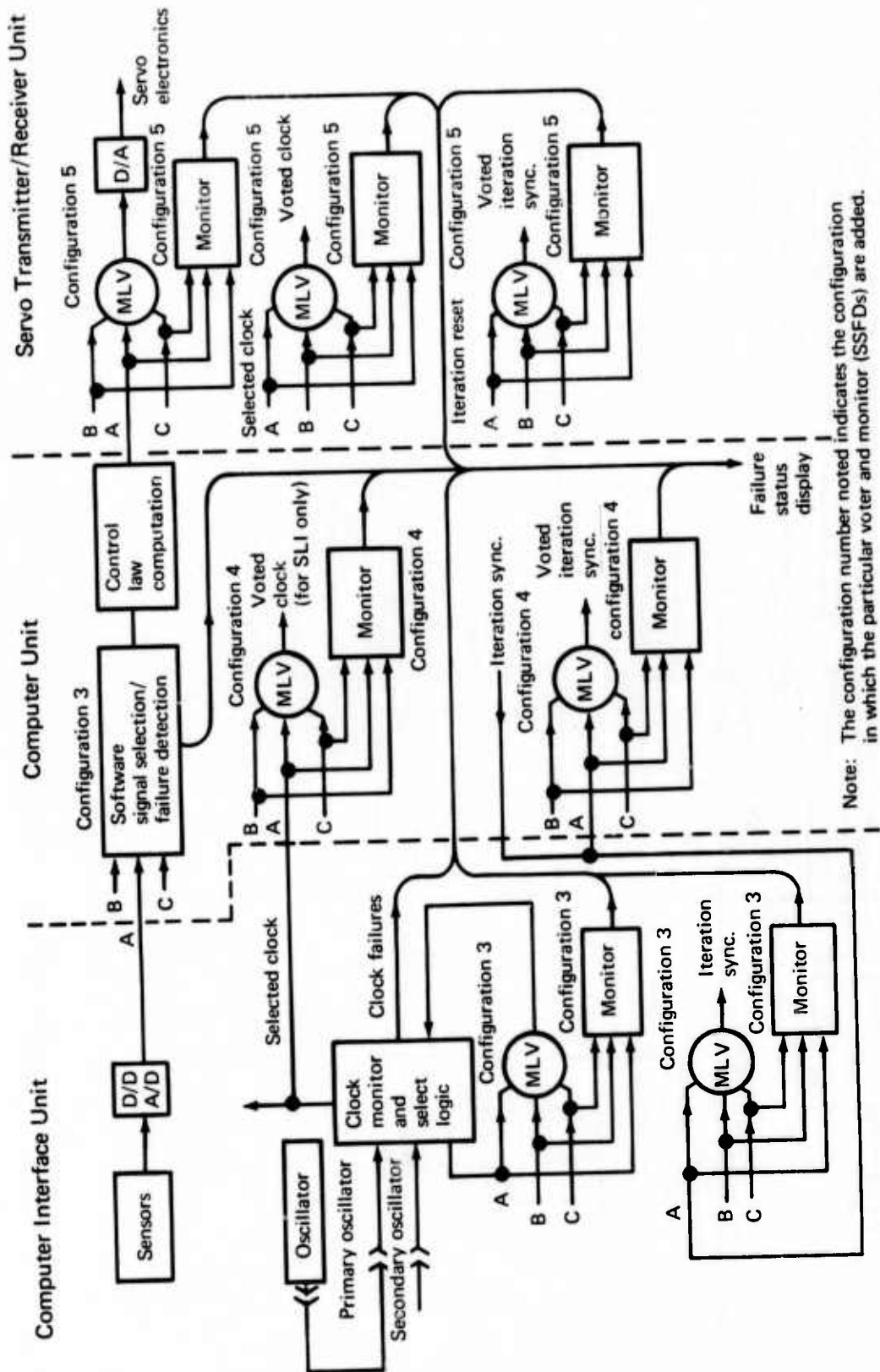


FIGURE 5-14. - WWCS VOTING/MONITORING SCHEMATIC DIAGRAM

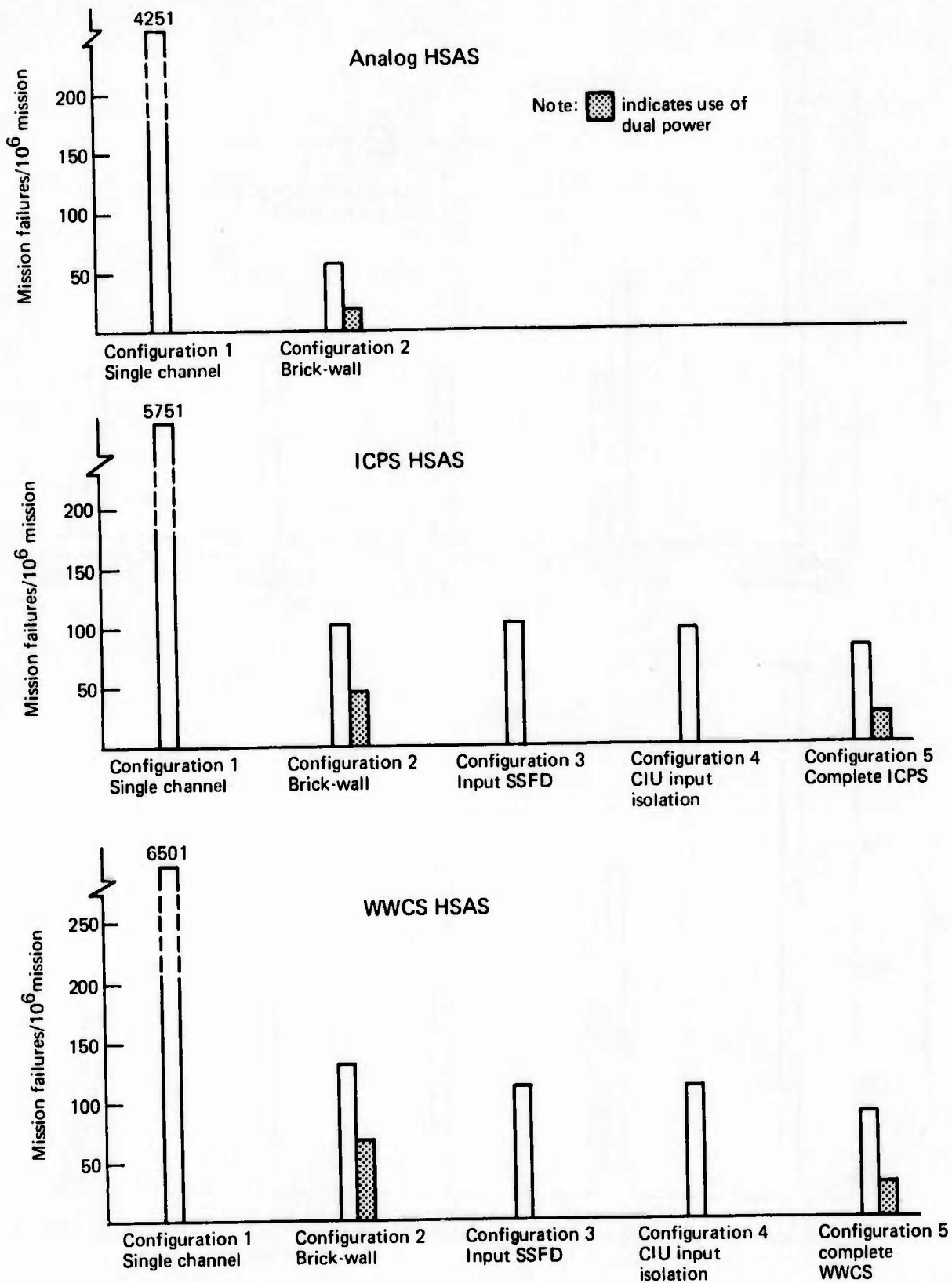


FIGURE 5-15.—PROBABILITY OF MISSION FAILURE (HSAS FUNCTION)

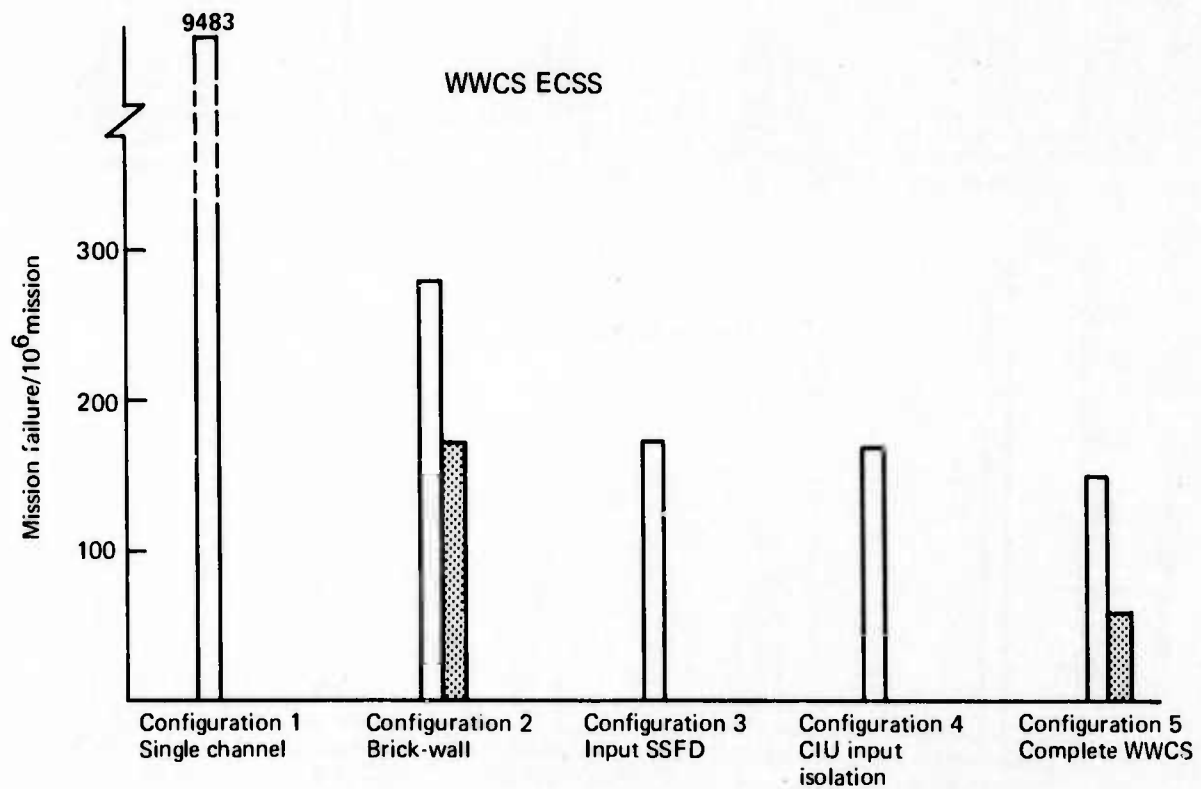
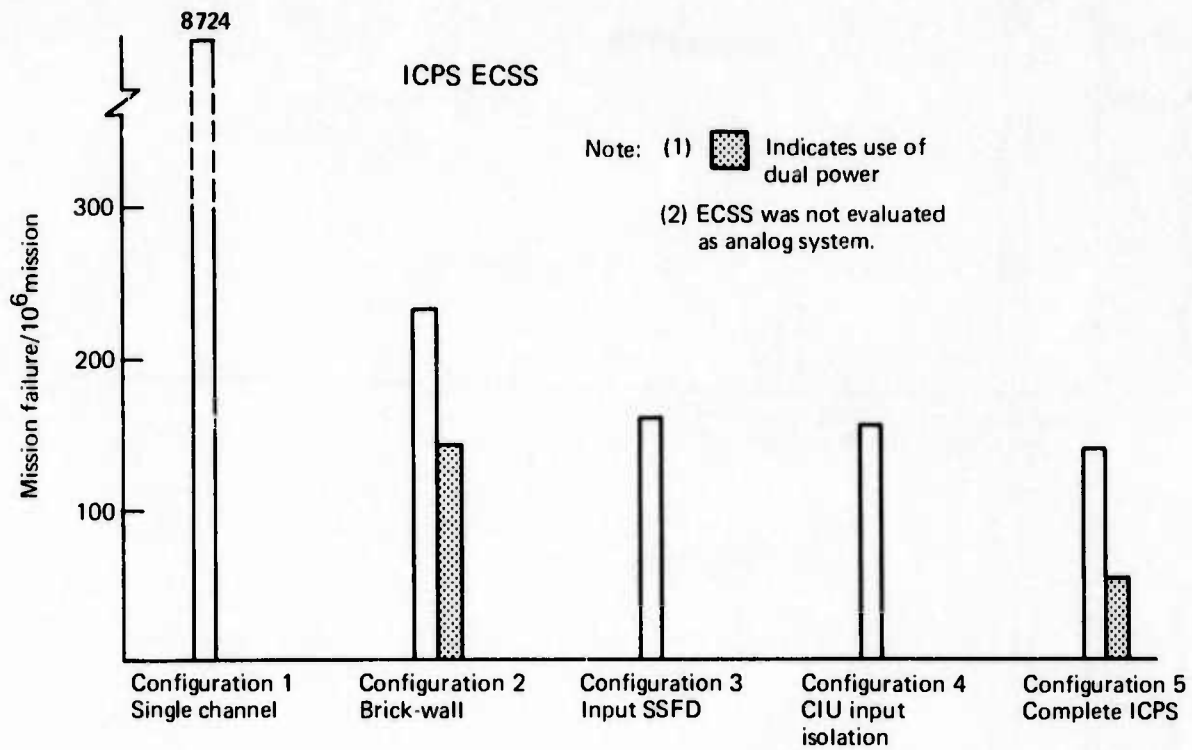


FIGURE 5-16.—PROBABILITY OF MISSION FAILURE (ECSS FUNCTION)

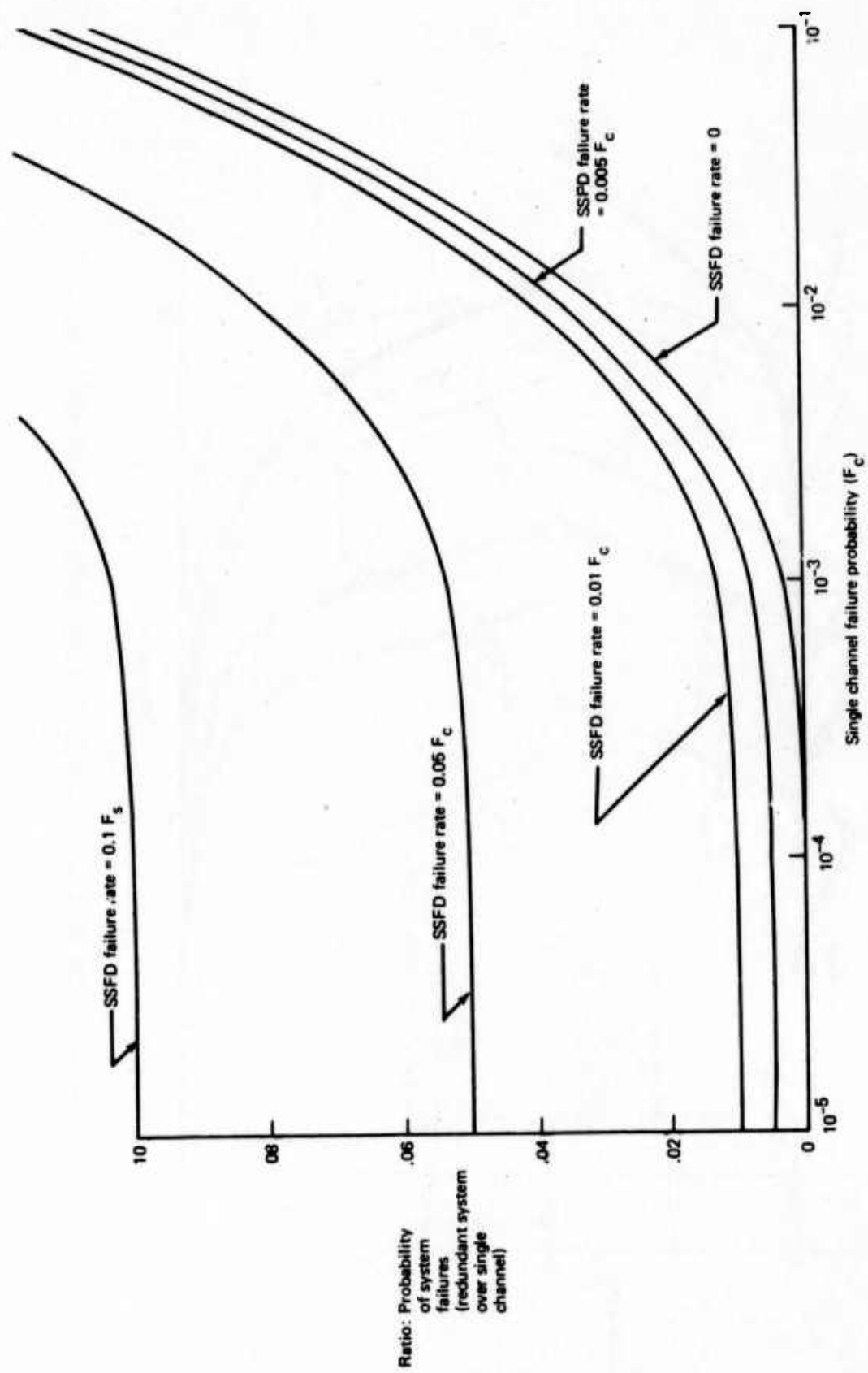


FIGURE 5-17.—REDUNDANT SYSTEM RELIABILITY AS FUNCTION OF SINGLE CHANNEL AND SSFD RELIABILITIES

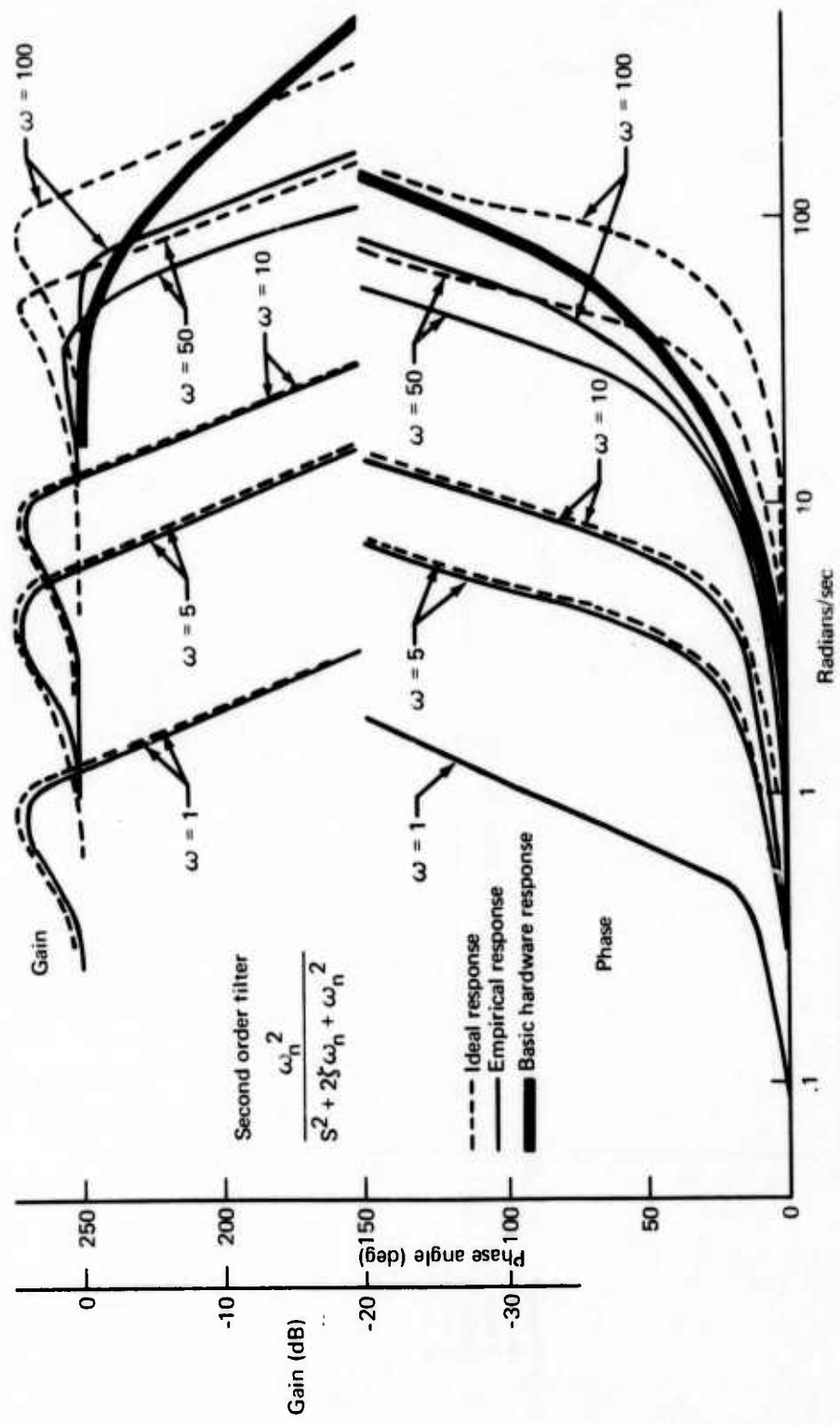


FIGURE 5-18.—100PS SECOND ORDER FILTER FREQUENCY RESPONSE $\zeta = 0.3$

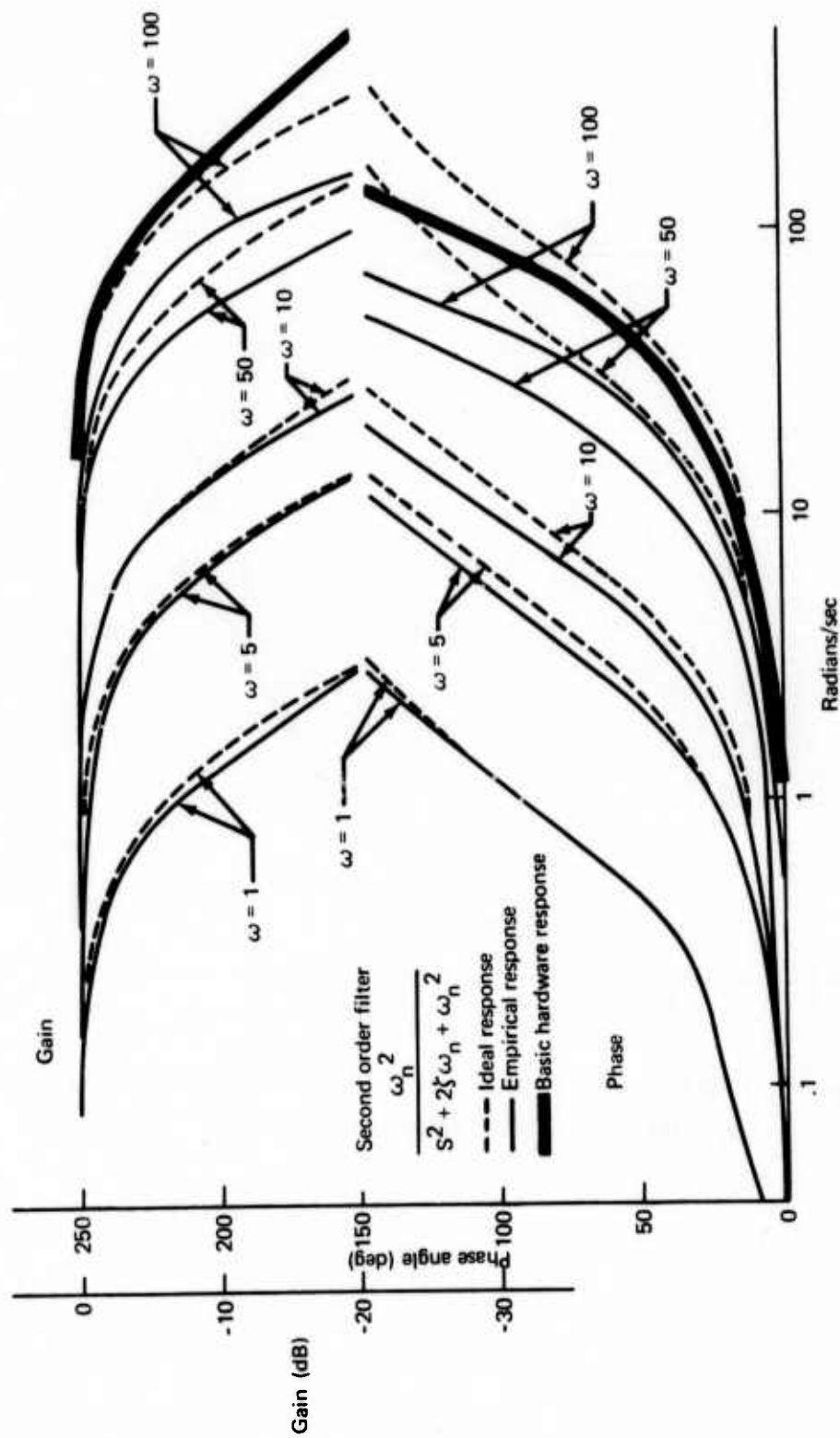


FIGURE 5-19.—ICPS SECOND ORDER FILTER FREQUENCY RESPONSE $\zeta = 1.0$

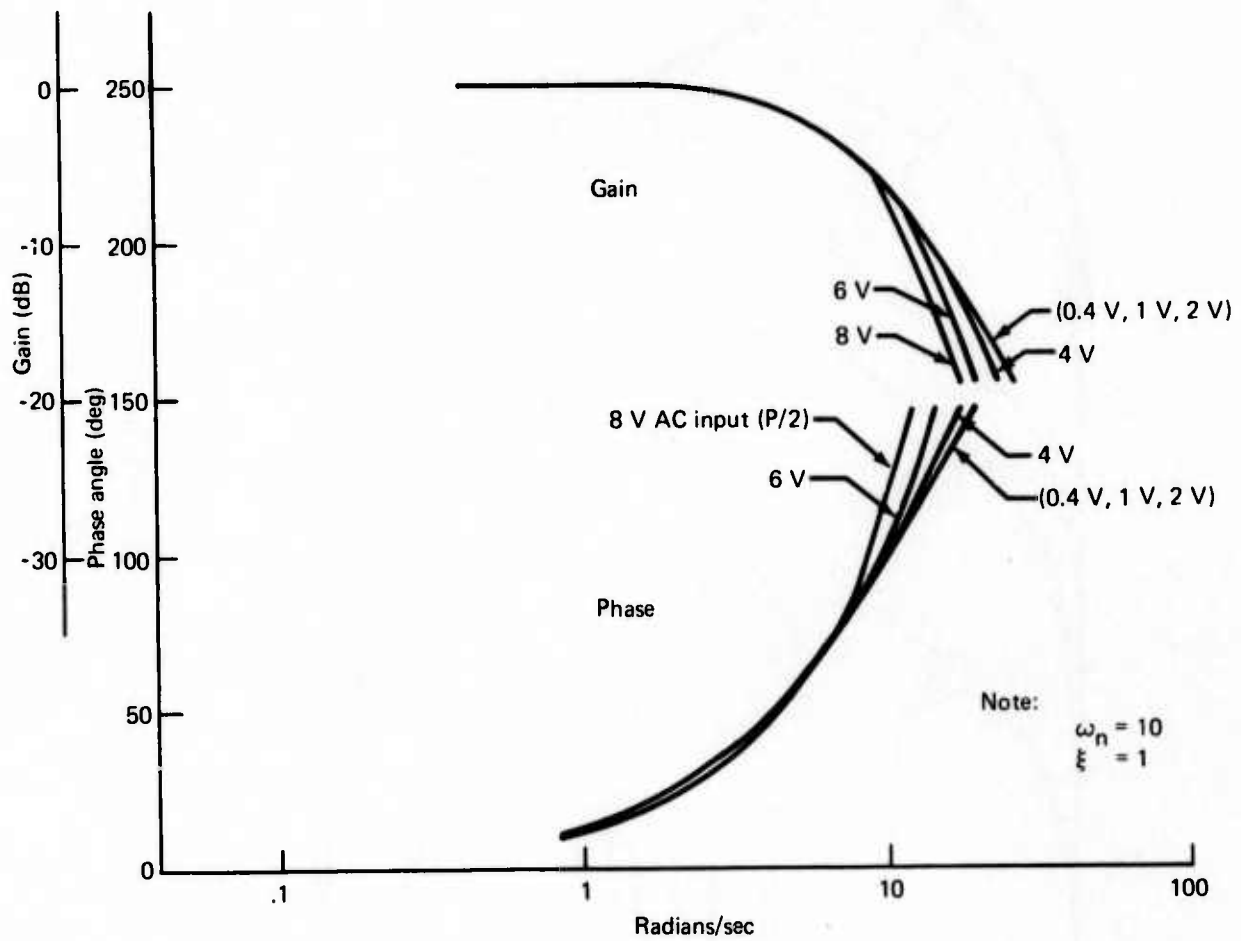
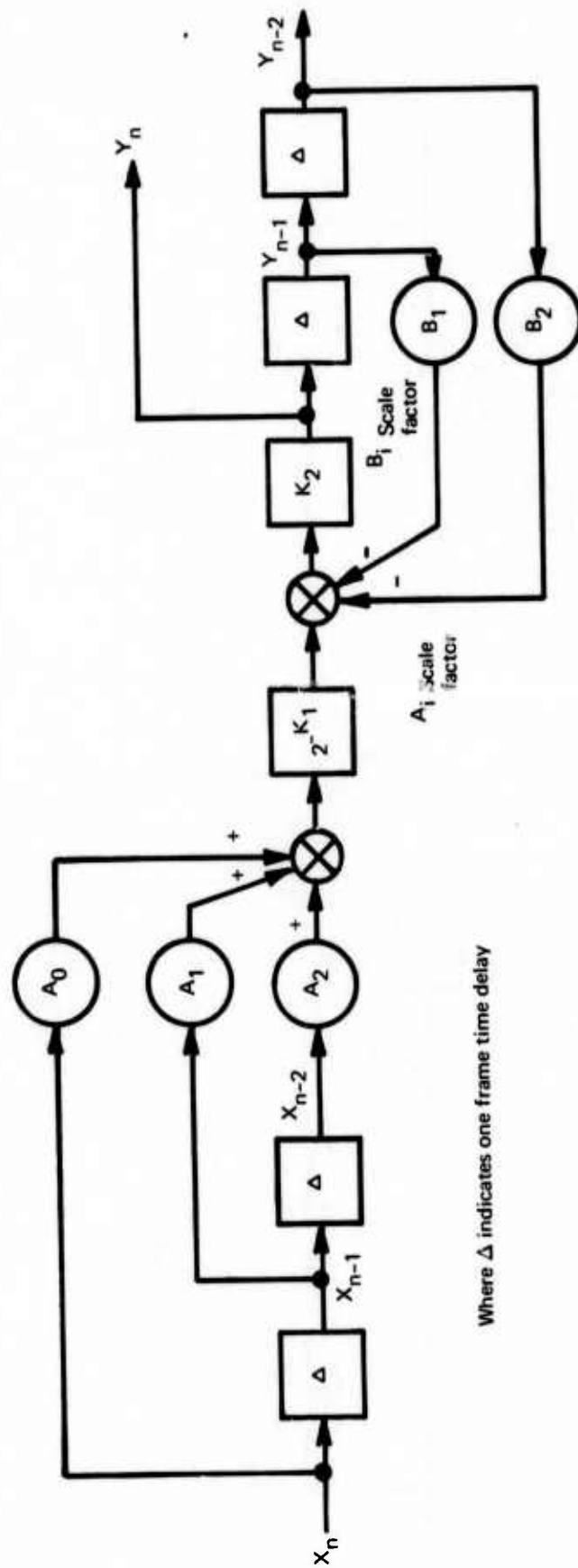


FIGURE 5-20.—SLEW RATE EFFECT ON SECOND ORDER FILTER RESPONSE
(INCREASING INPUT AMPLITUDE)



Where Δ indicates one frame time delay

FIGURE 5-21.—SECOND ORDER FILTER BILINEAR TRANSFORMATION BLOCK DIAGRAM

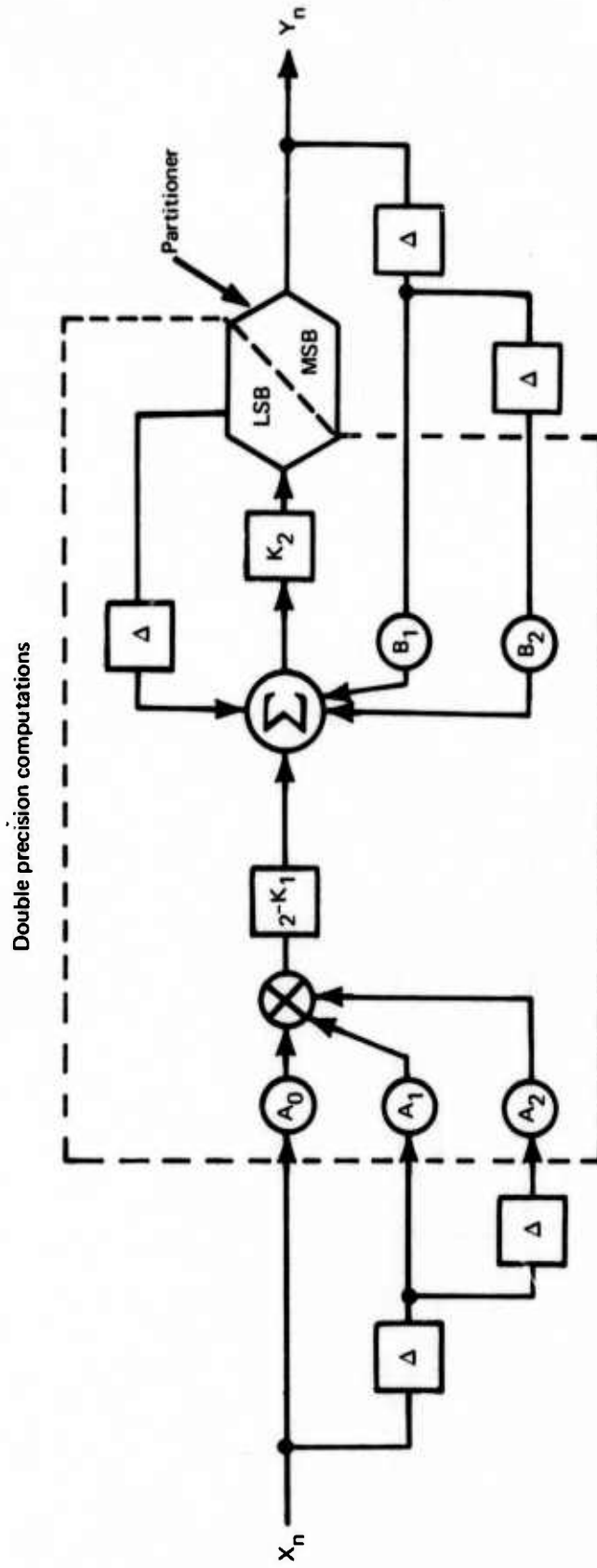
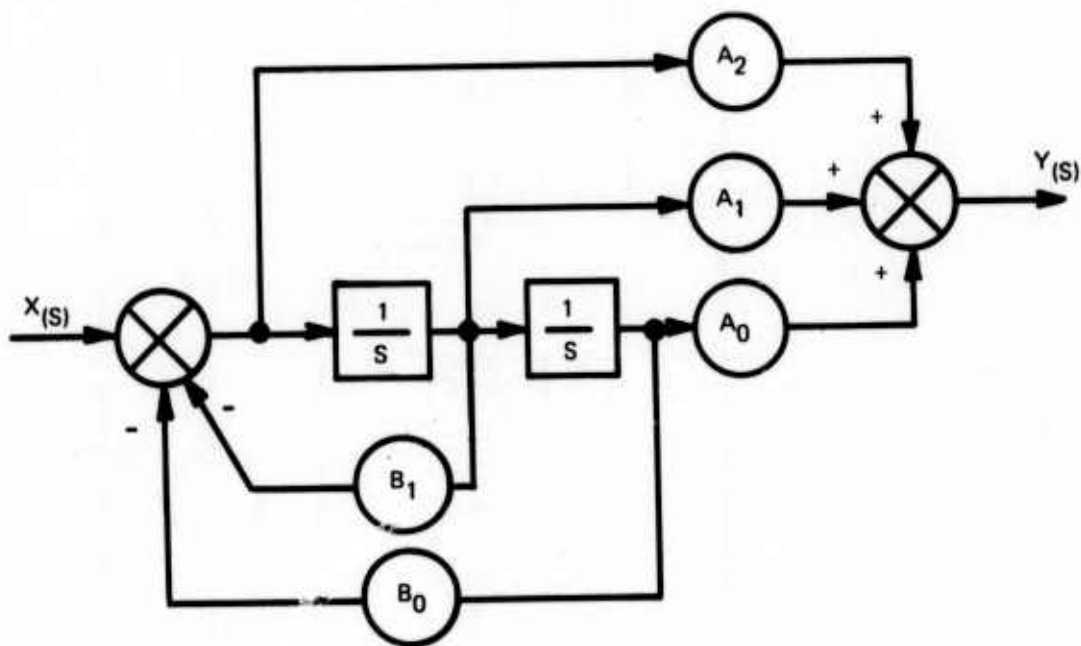


FIGURE 5-22.—SECOND ORDER FILTER BILINEAR TRANSFORMATION WITH STEADY STATE ERROR COMPENSATION



$$\frac{Y(s)}{X(s)} = \frac{A_0 + A_1s + A_2s^2}{B_0 + B_1s + s^2}$$

FIGURE 5-23.—CONTINUOUS "S" DOMAIN SECOND ORDER FILTER

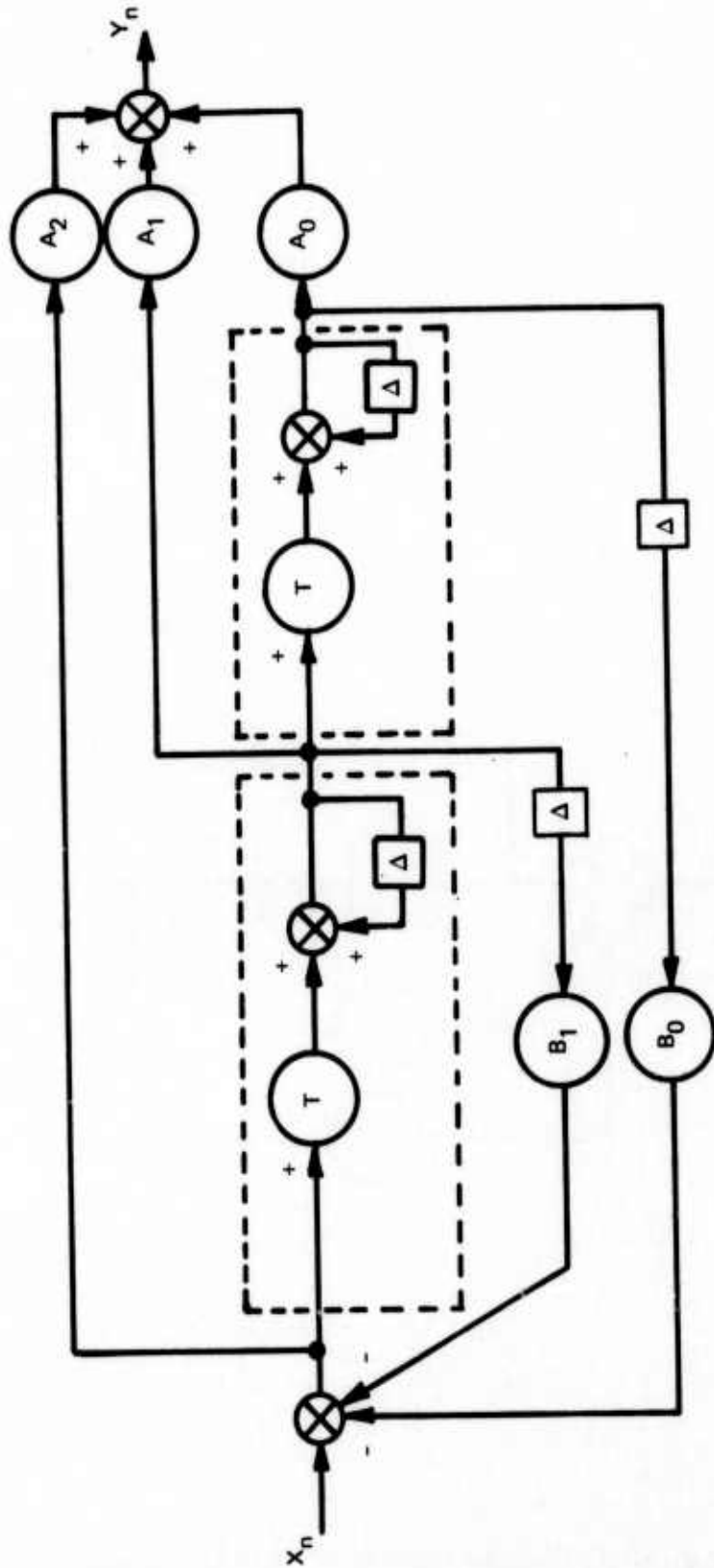


FIGURE 5-24. — RECTANGULAR INTEGRATION SECOND ORDER FILTER

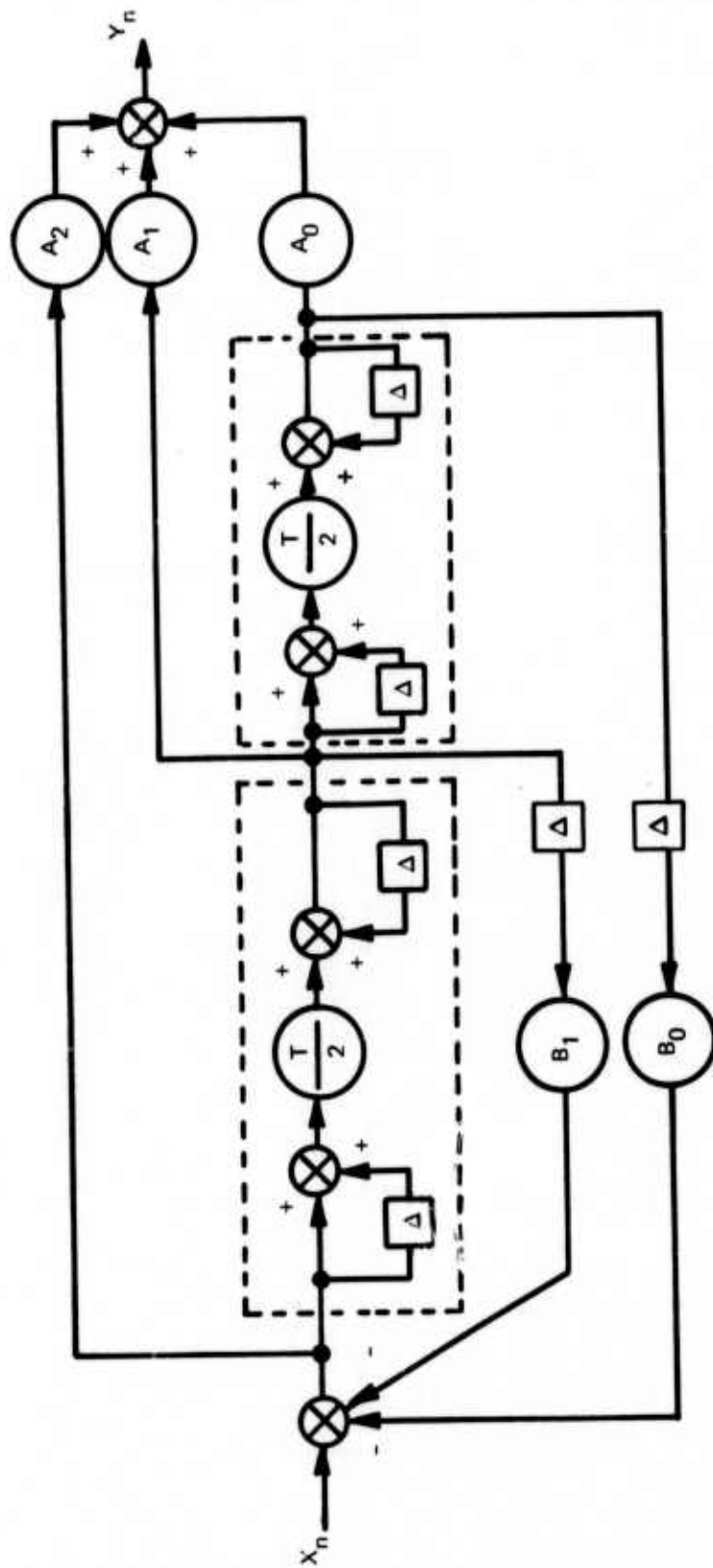


FIGURE 5-25.—TRAPEZOIDAL INTEGRATION SECOND ORDER FILTER

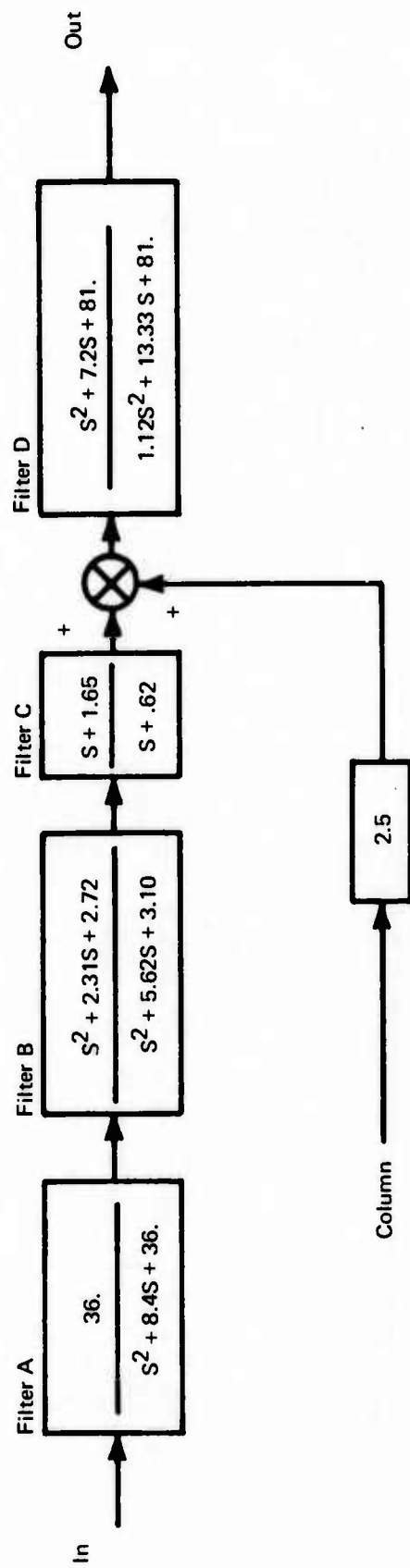


FIGURE 5-26.—HSAS FILTERS

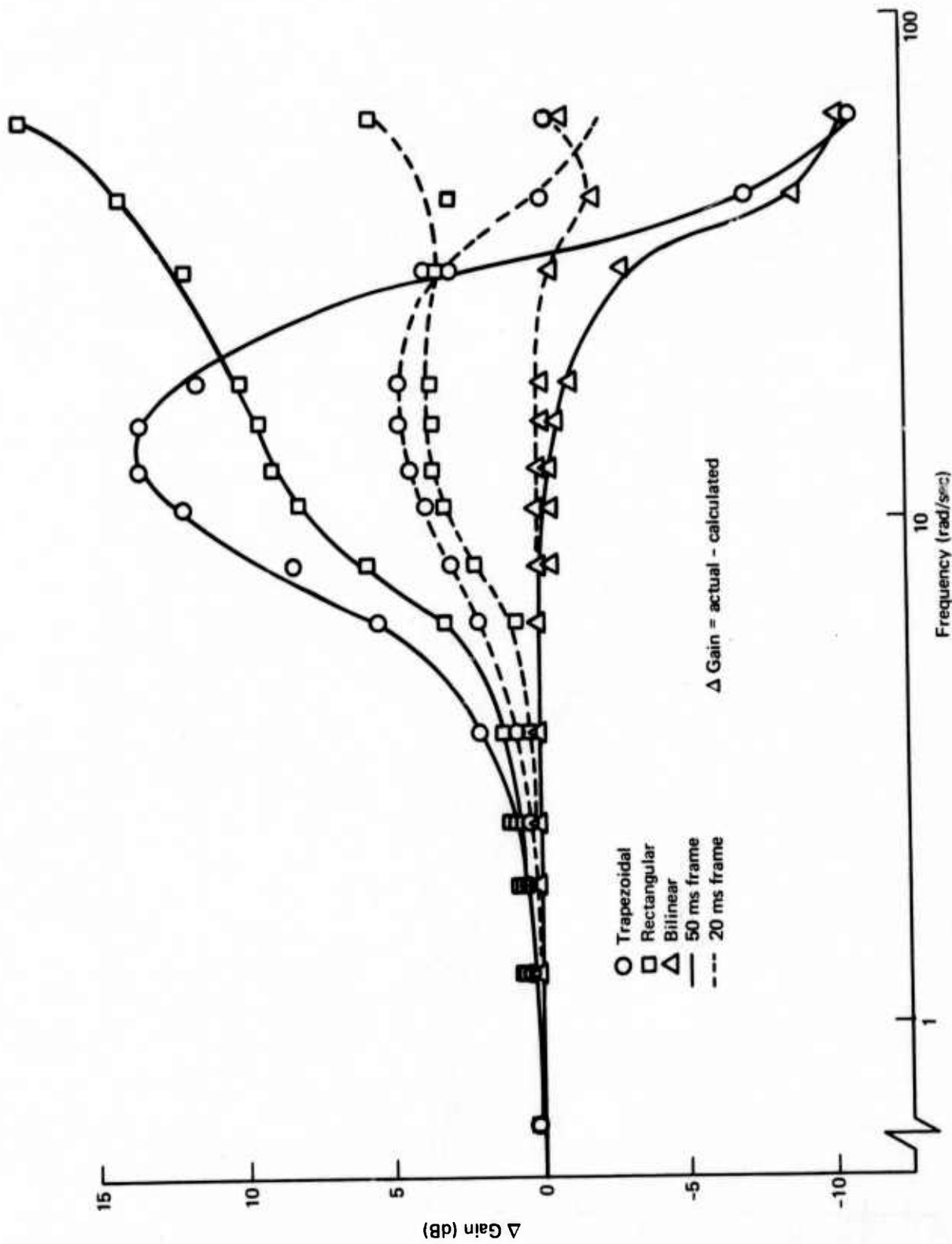


FIGURE 5-27.—AMPLITUDE ERROR RESPONSE OF COMPLETE HSAS FILTER SET

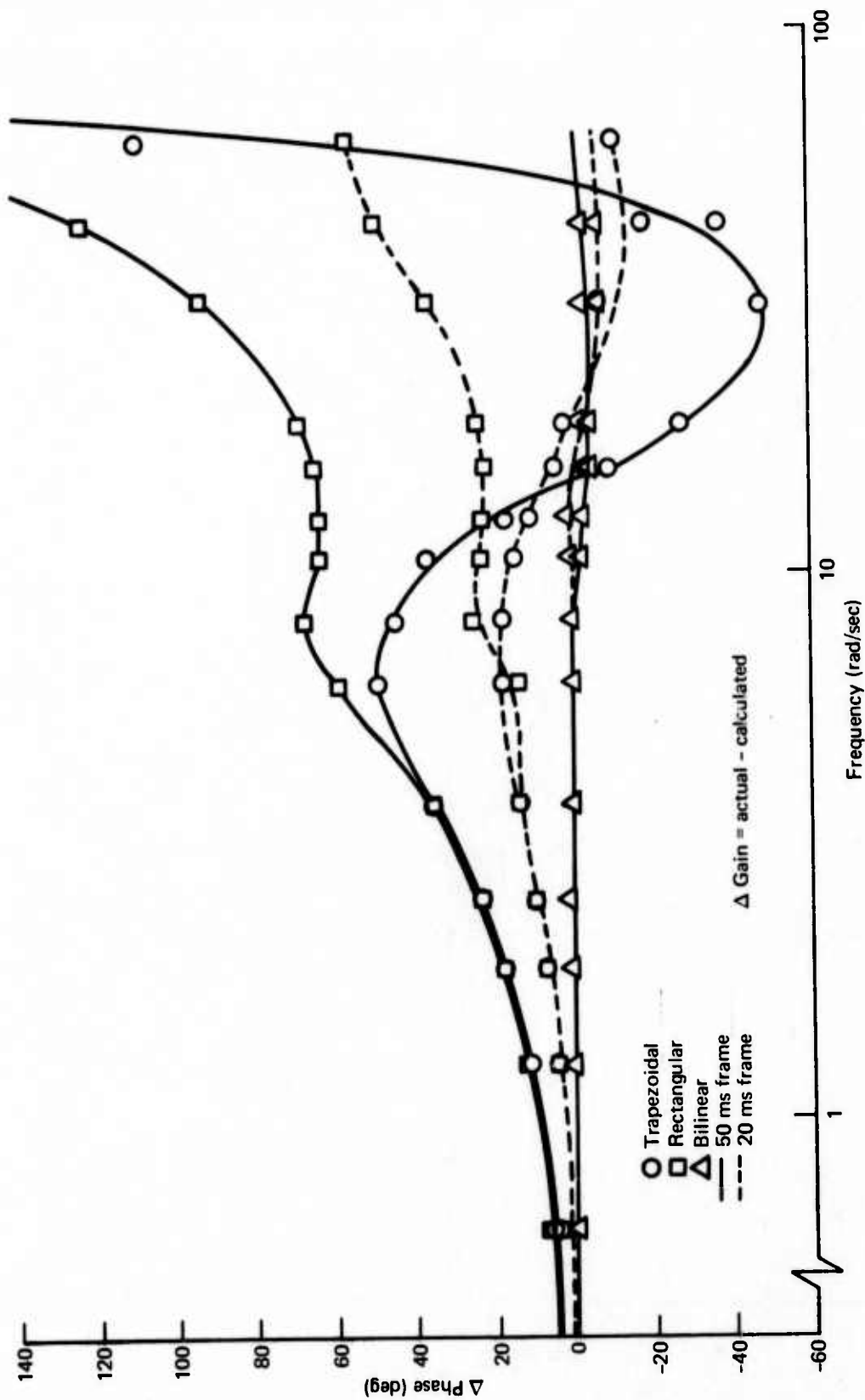


FIGURE 5-28.—PHASE ERROR RESPONSE OF COMPLETE HSAS FILTER SET

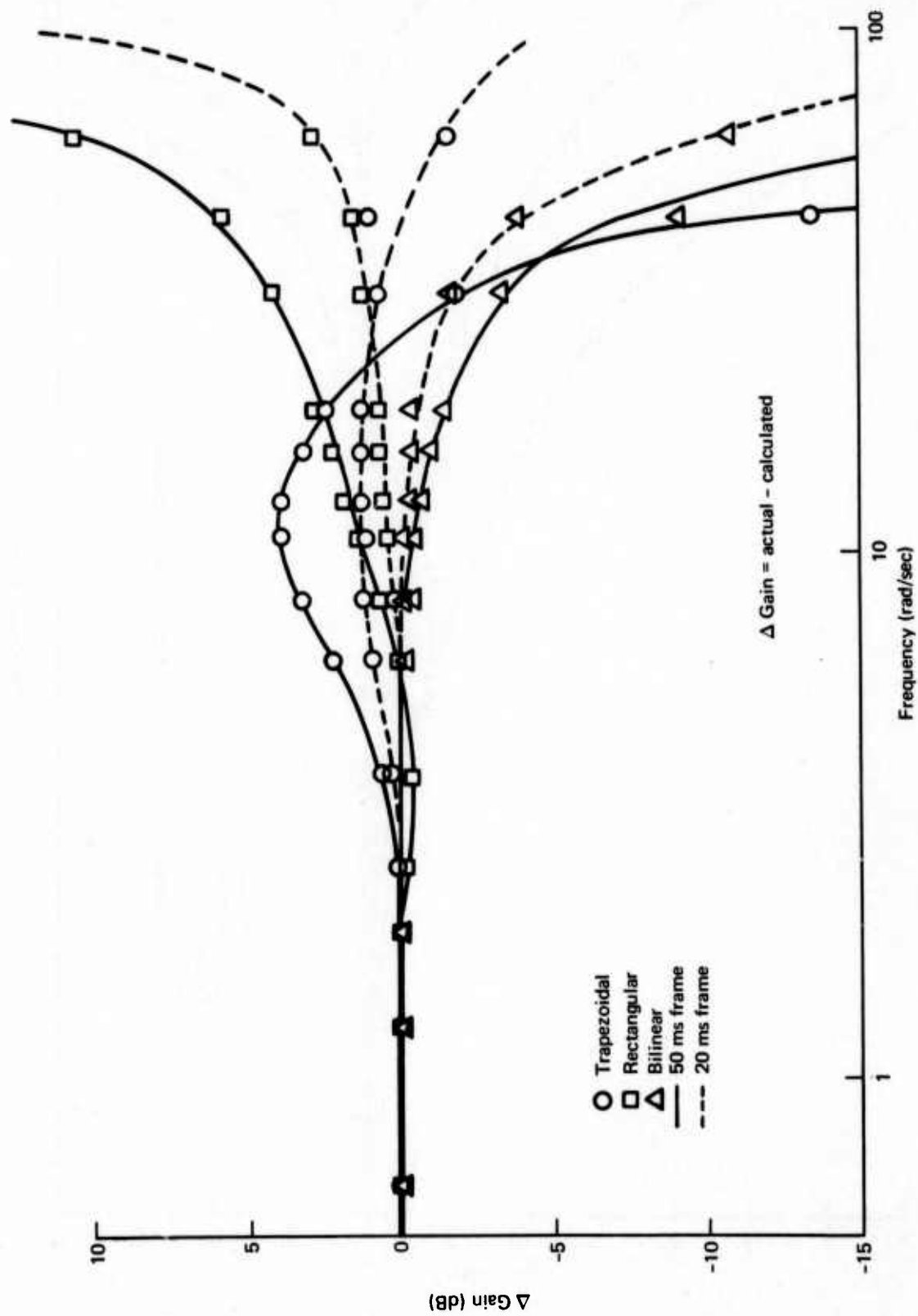


FIGURE 5-29.—AMPLITUDE ERROR RESPONSE OF HSAS FILTER A

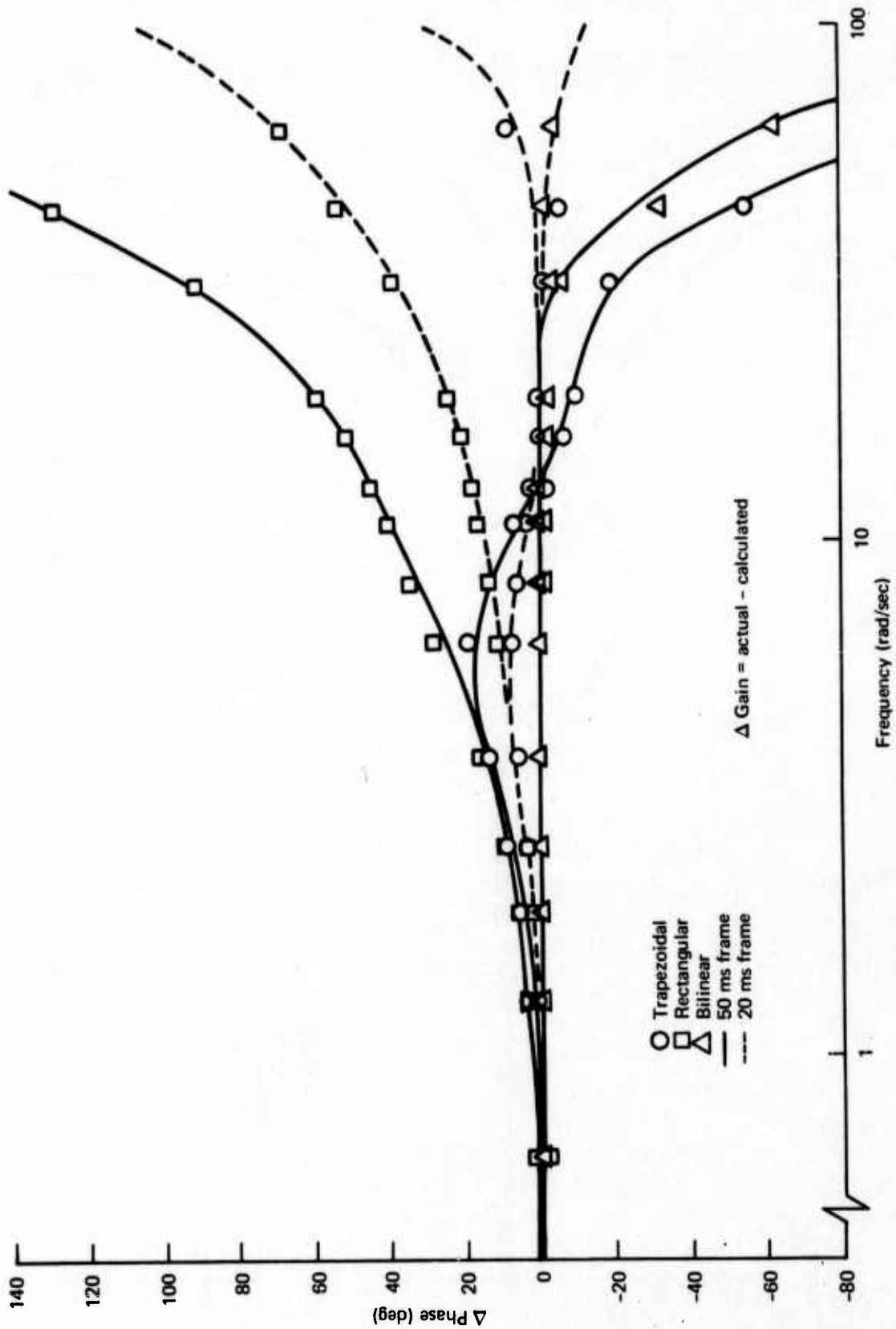


FIGURE 5-30.—PHASE ERROR RESPONSE OF HSAS FILTER A

Note: Bilinear response the same for both solution rates

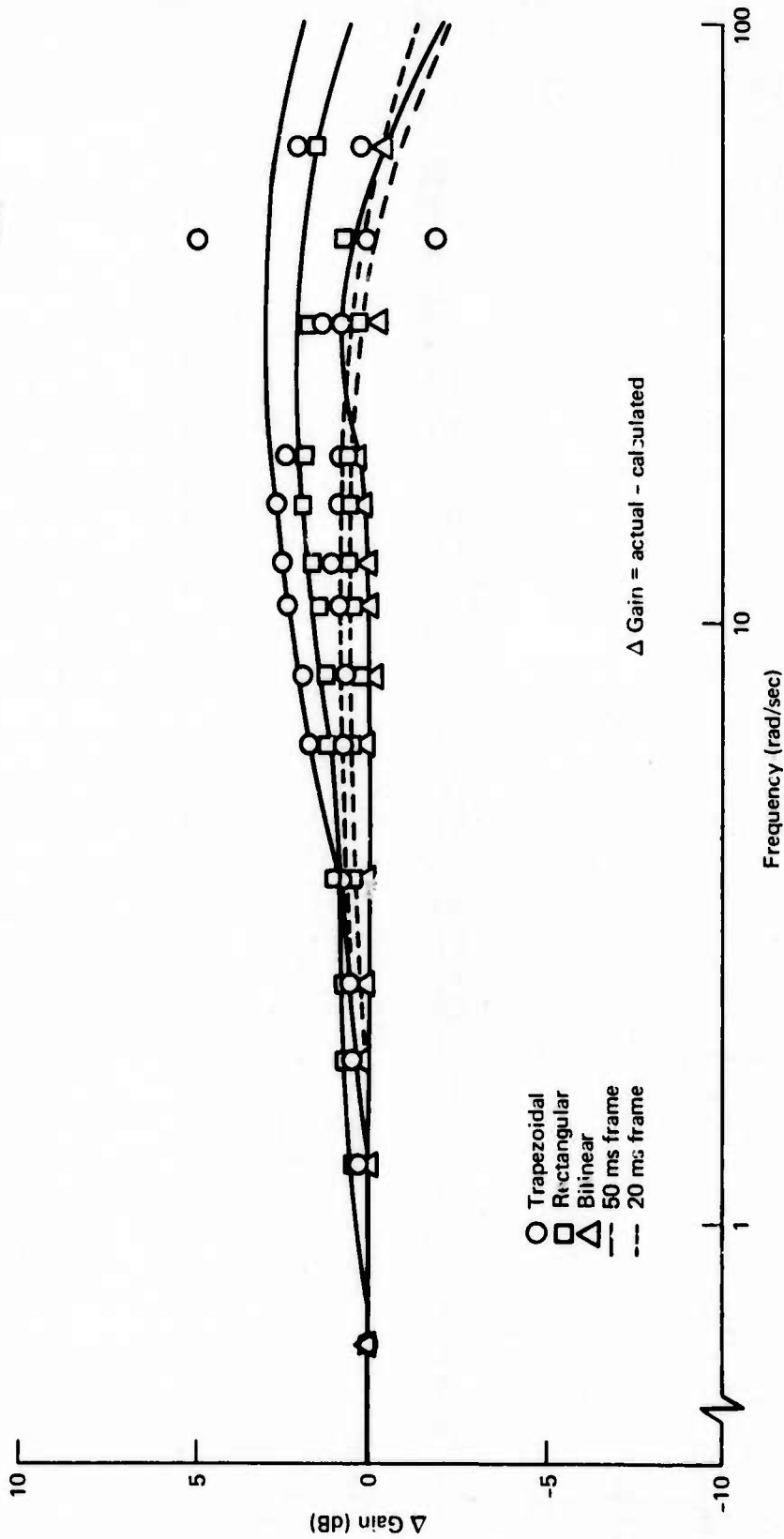


FIGURE 5-31.—AMPLITUDE ERROR RESPONSE OF HSAS FILTER B

Note: 20 ms frame solution
rate bounded within
50 ms frame solution
rate

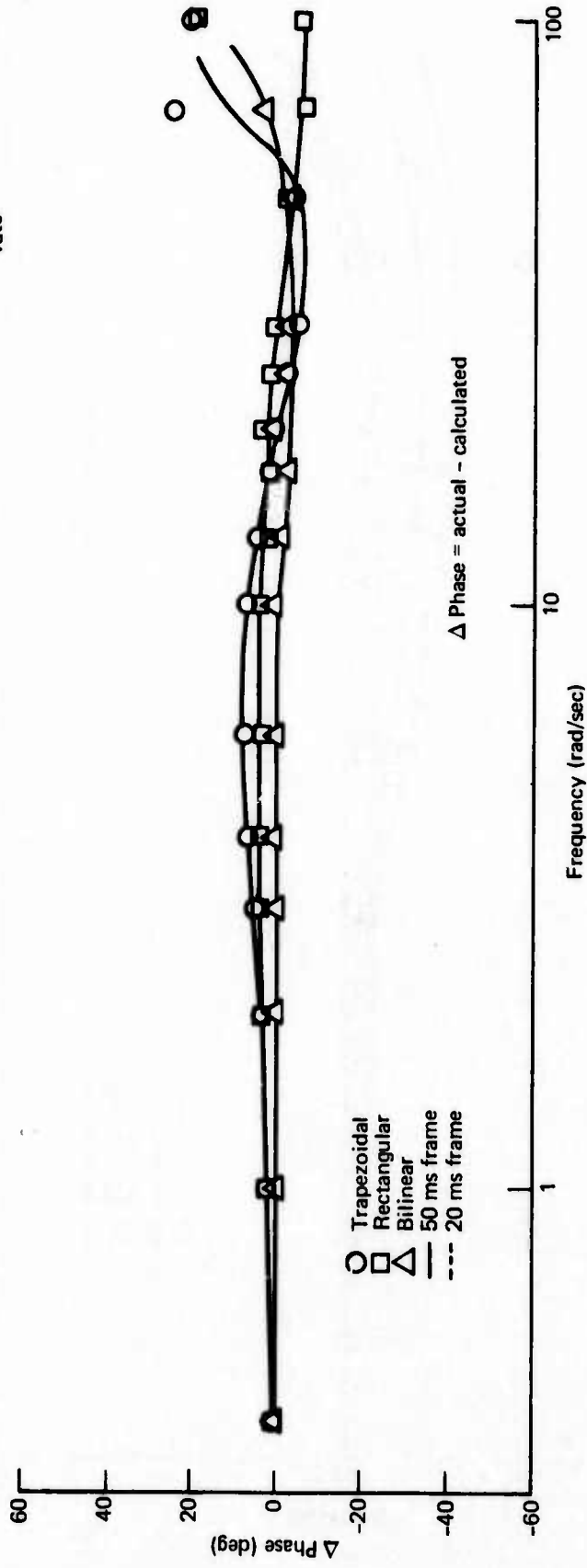


FIGURE 5-32.—PHASE ERROR RESPONSE OF HSAS FILTER B

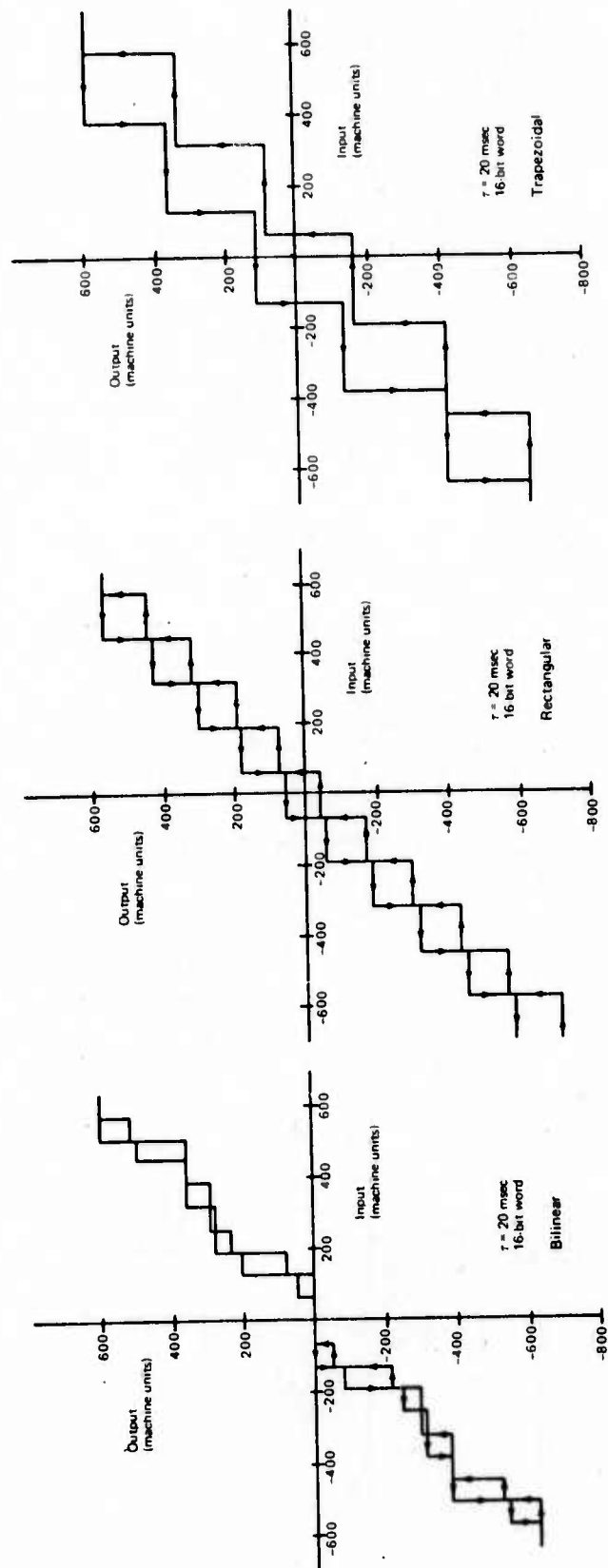


FIGURE 5-33.—HSAS FILTER A STEADY STATE RESPONSE

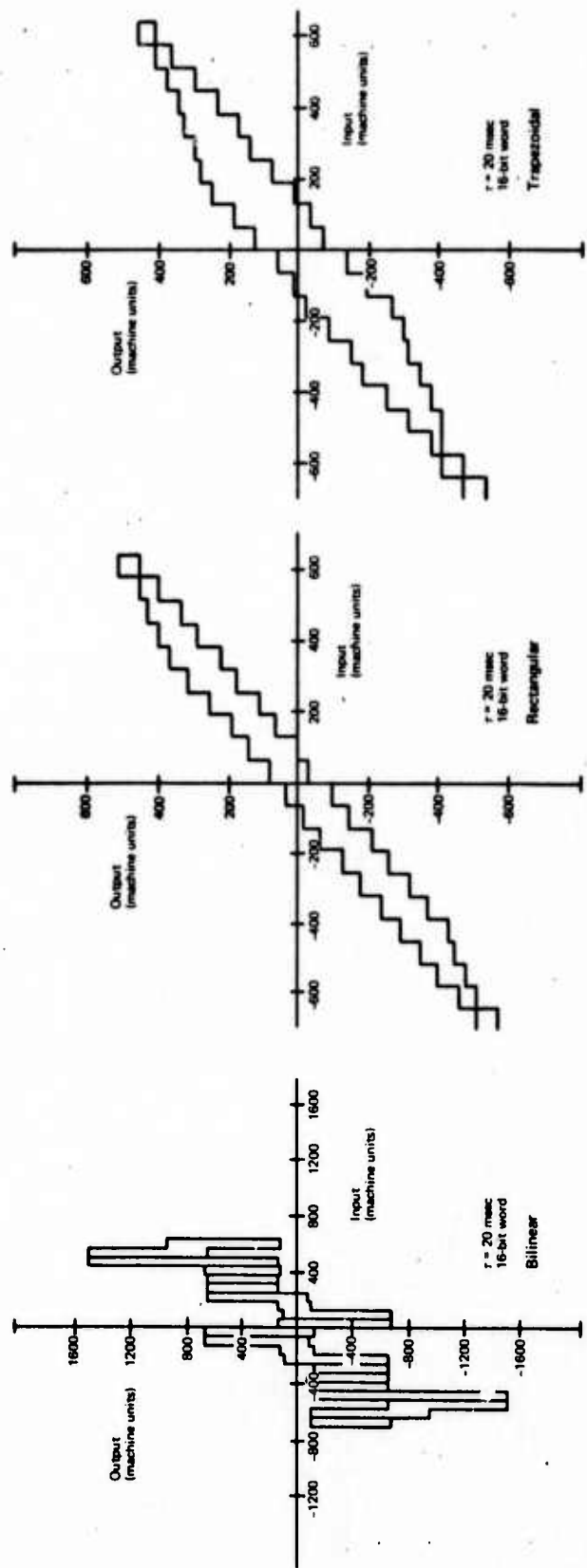


FIGURE 5-34.—HSAS FILTER B STEADY STATE RESPONSE

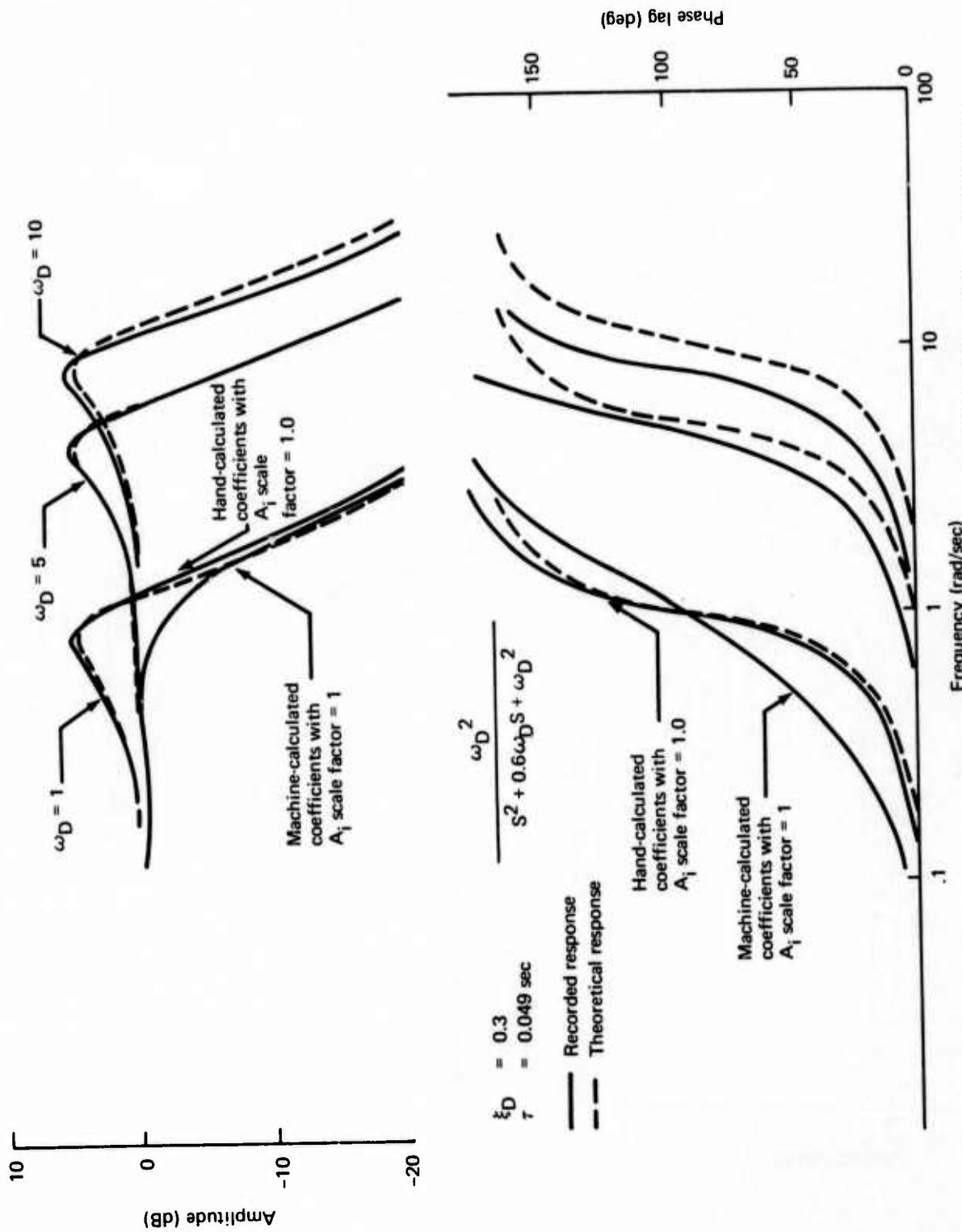


FIGURE 5-35.—WWCS LOW FREQUENCY FILTER RESPONSE FOR $\xi_D = 0.3$, $\tau = 0.049$ SEC

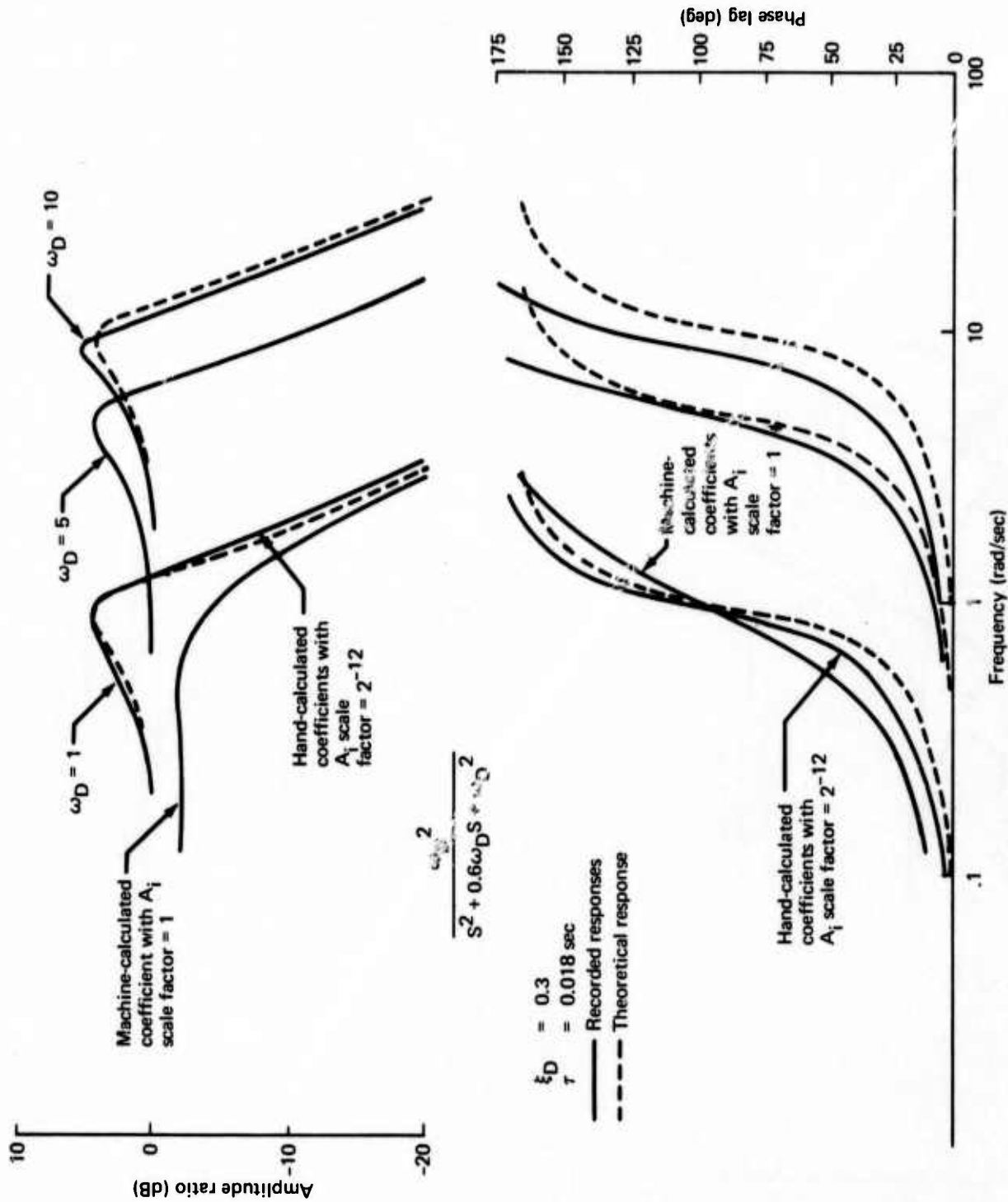


FIGURE 5-36.—WWCS LOW FREQUENCY FILTER RESPONSE FOR $\xi_D = 0.3, \tau = 0.018 \text{ SEC}$

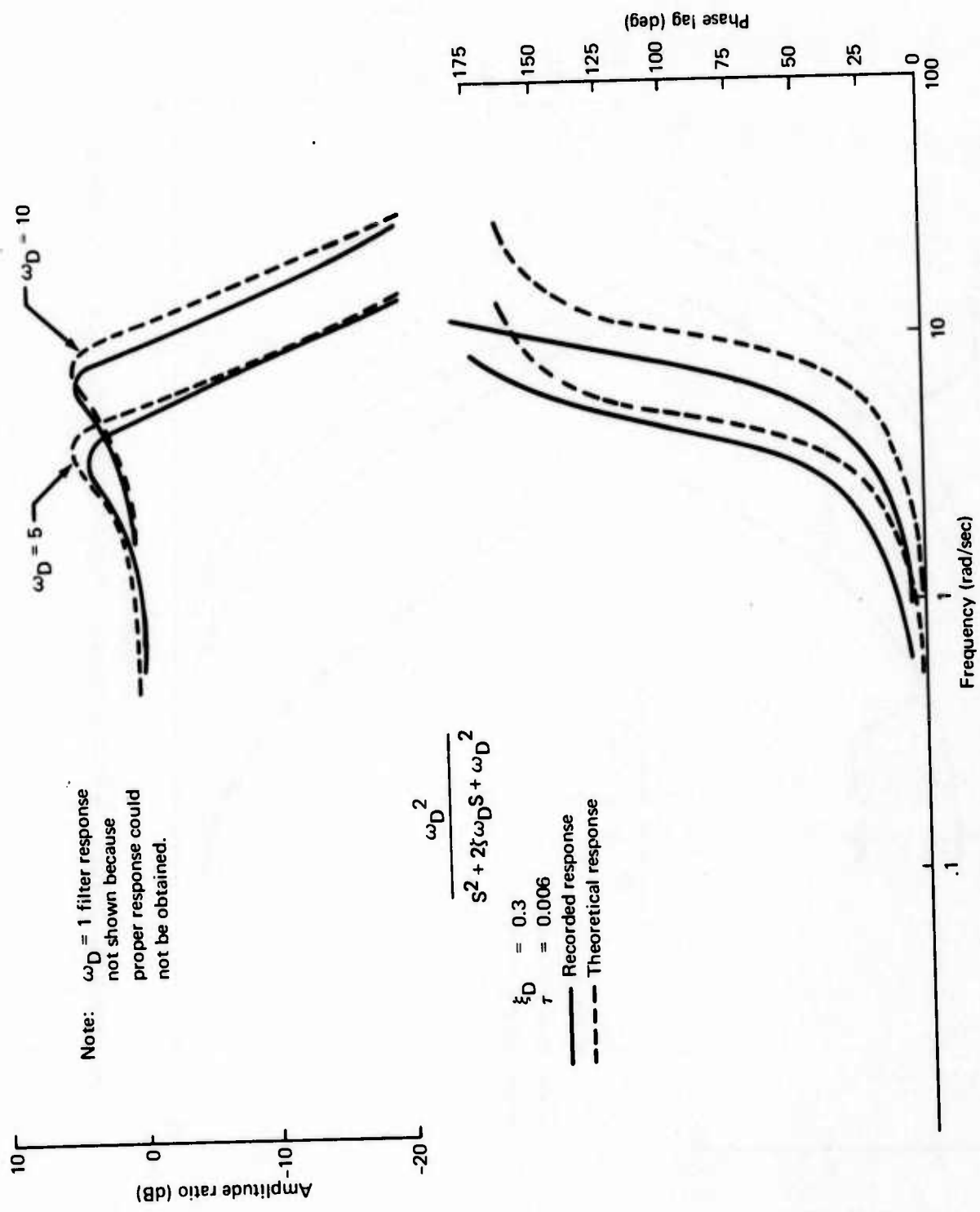


FIGURE 5-37.—WWCS LOW FREQUENCY FILTER RESPONSE FOR $\zeta_D = 0.3, \tau = 0.006$ SEC

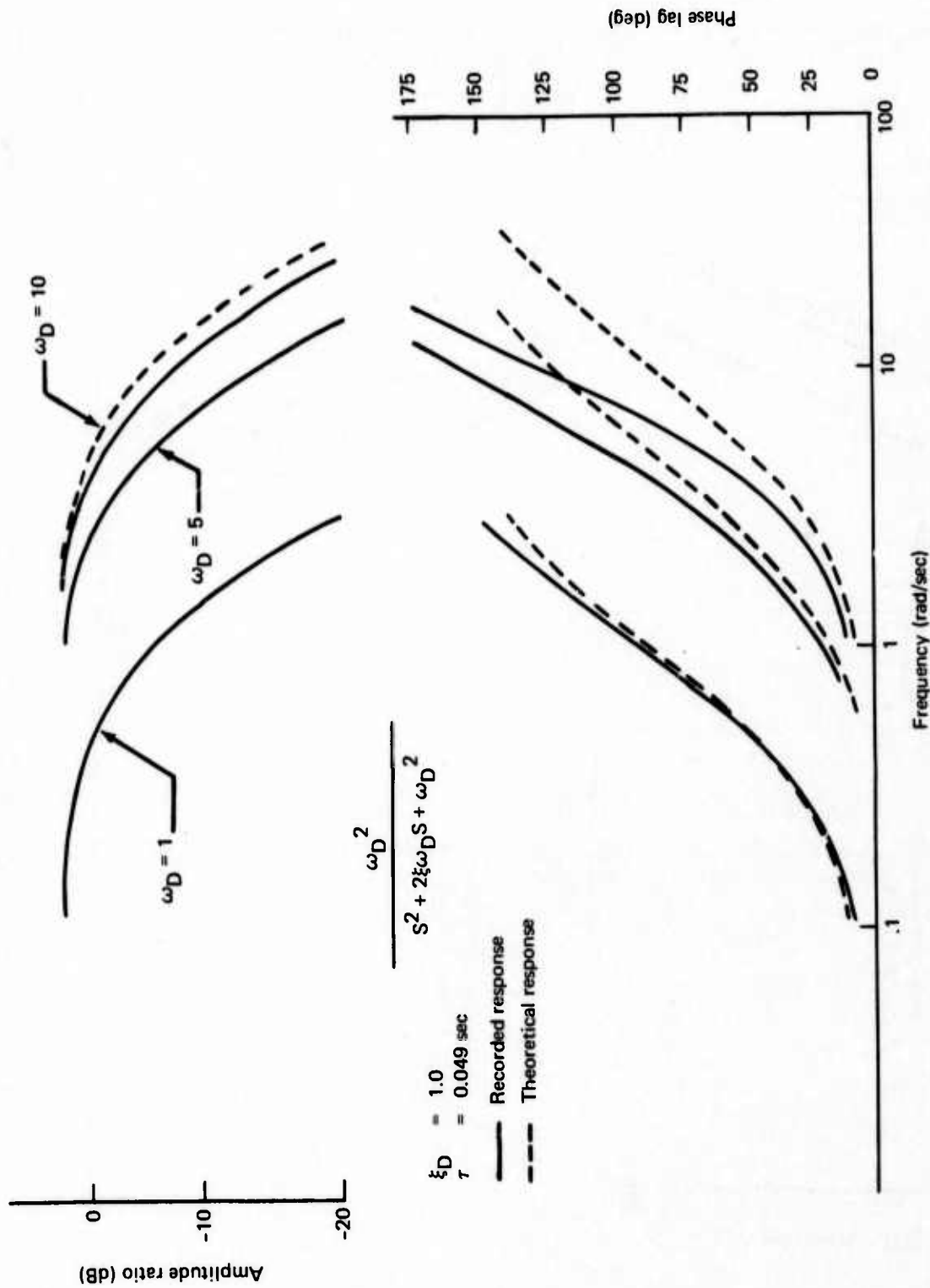


FIGURE 5-38.—WWCS LOW-FREQUENCY FILTER RESPONSE FOR $\xi_D = 1, \tau = 0.049 \text{ SEC}$

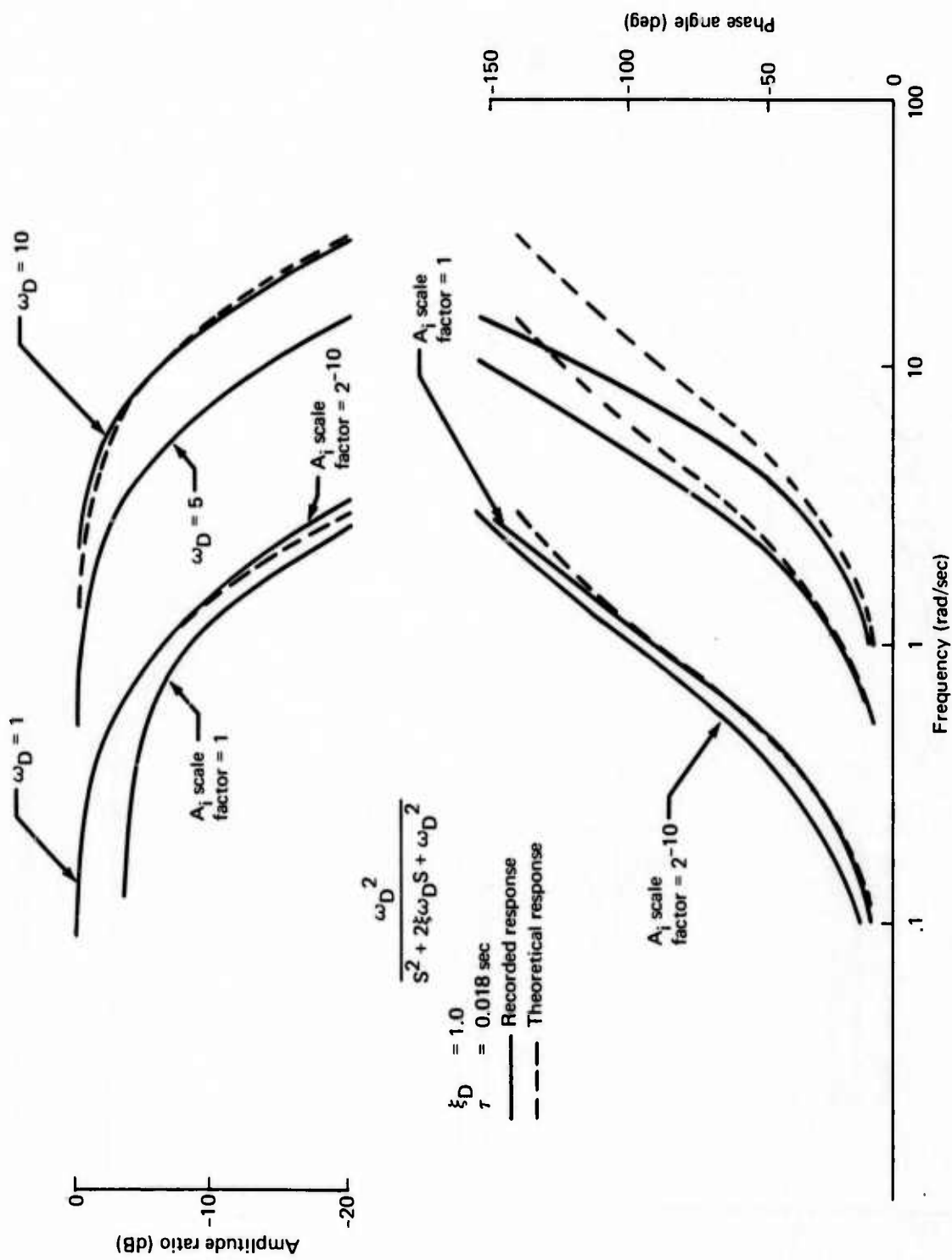


FIGURE 5-39.—WWCS LOW FREQUENCY FILTER RESPONSE FOR $\xi_D = 1, \tau = 0.018$ SEC

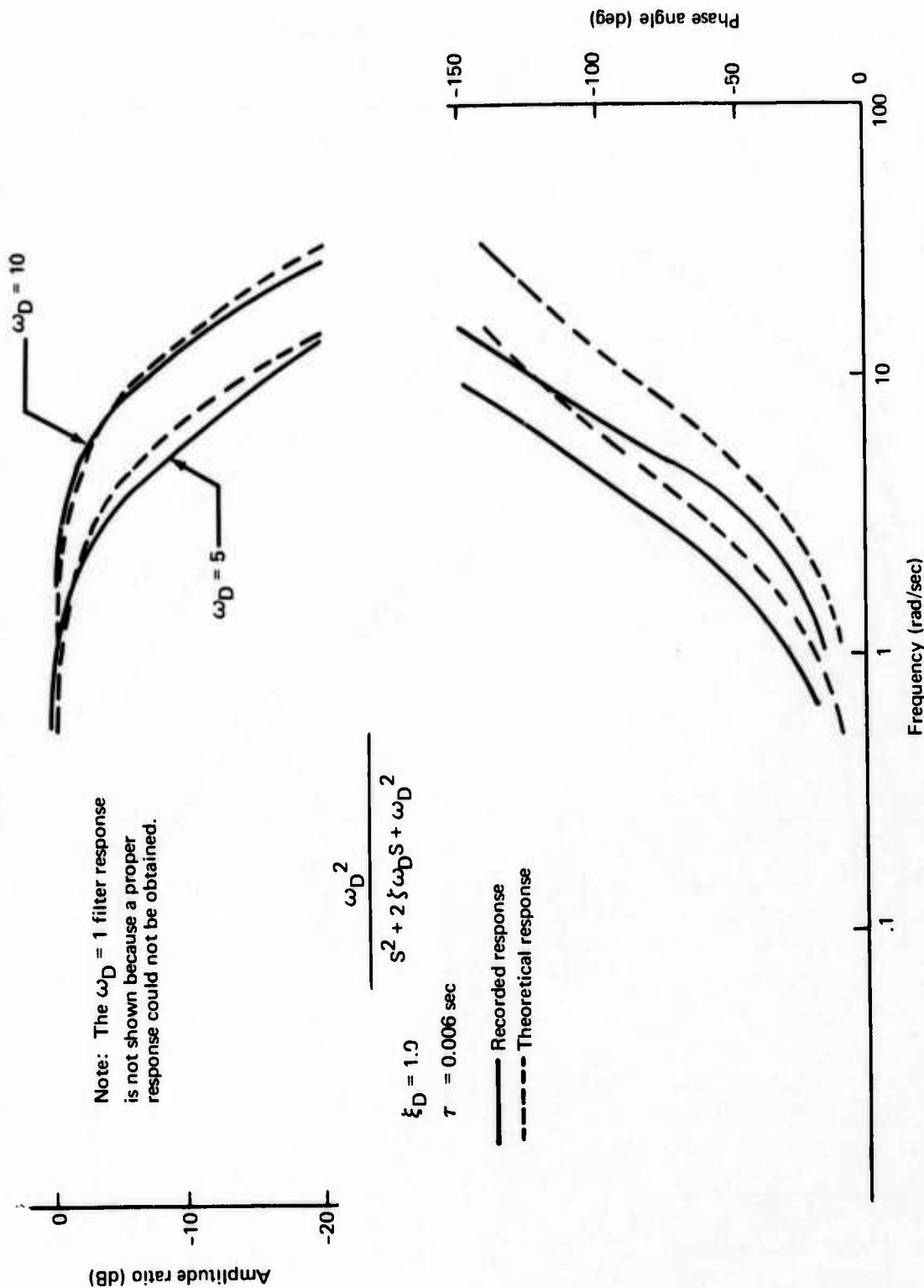


FIGURE 5-40.—WWCS LOW FREQUENCY FILTER RESPONSE FOR $\xi_D = 1, \tau = 0.006 \text{ SEC}$

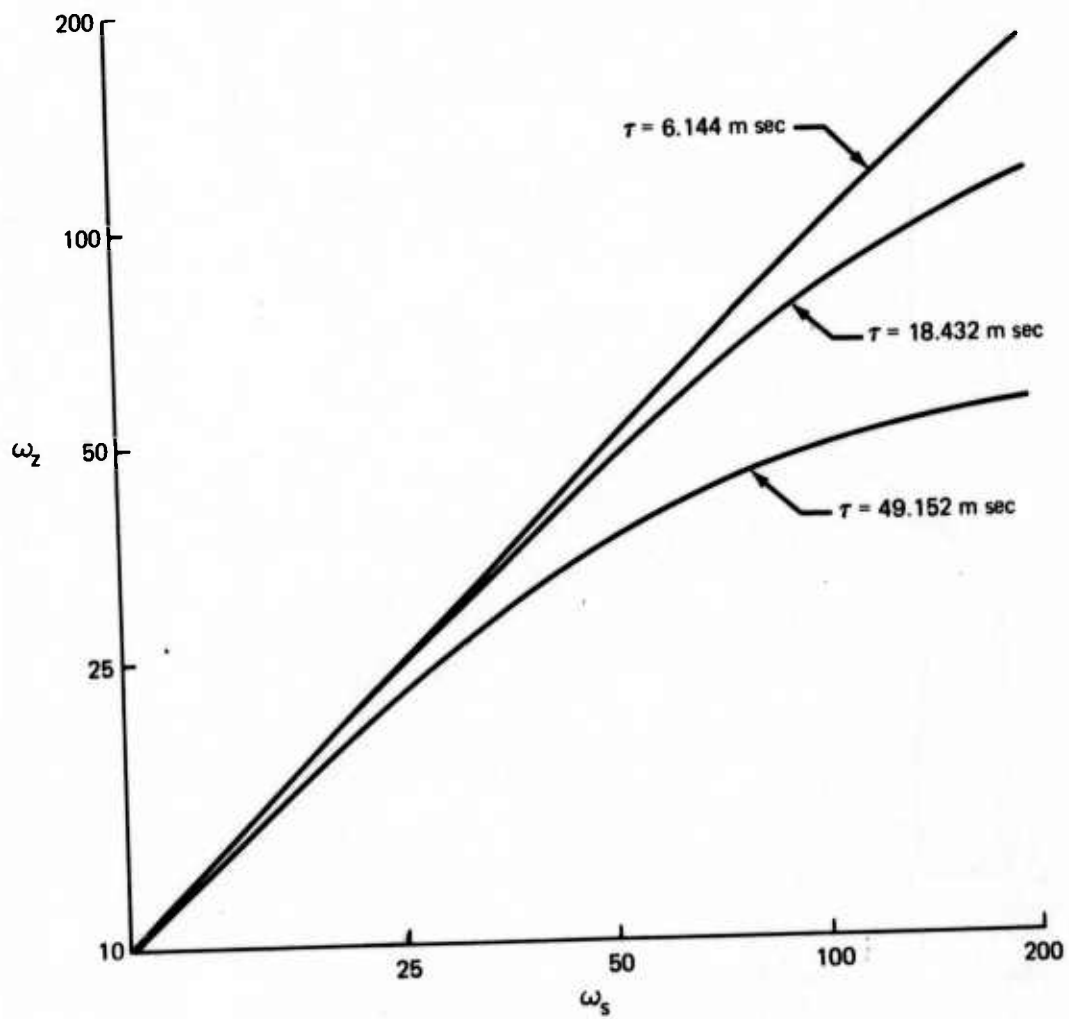


FIGURE 5-41.—THEORETICAL FREQUENCY SHIFT BETWEEN CONTINUOUS TIME DOMAIN AND BILINEAR IMPLEMENTATION

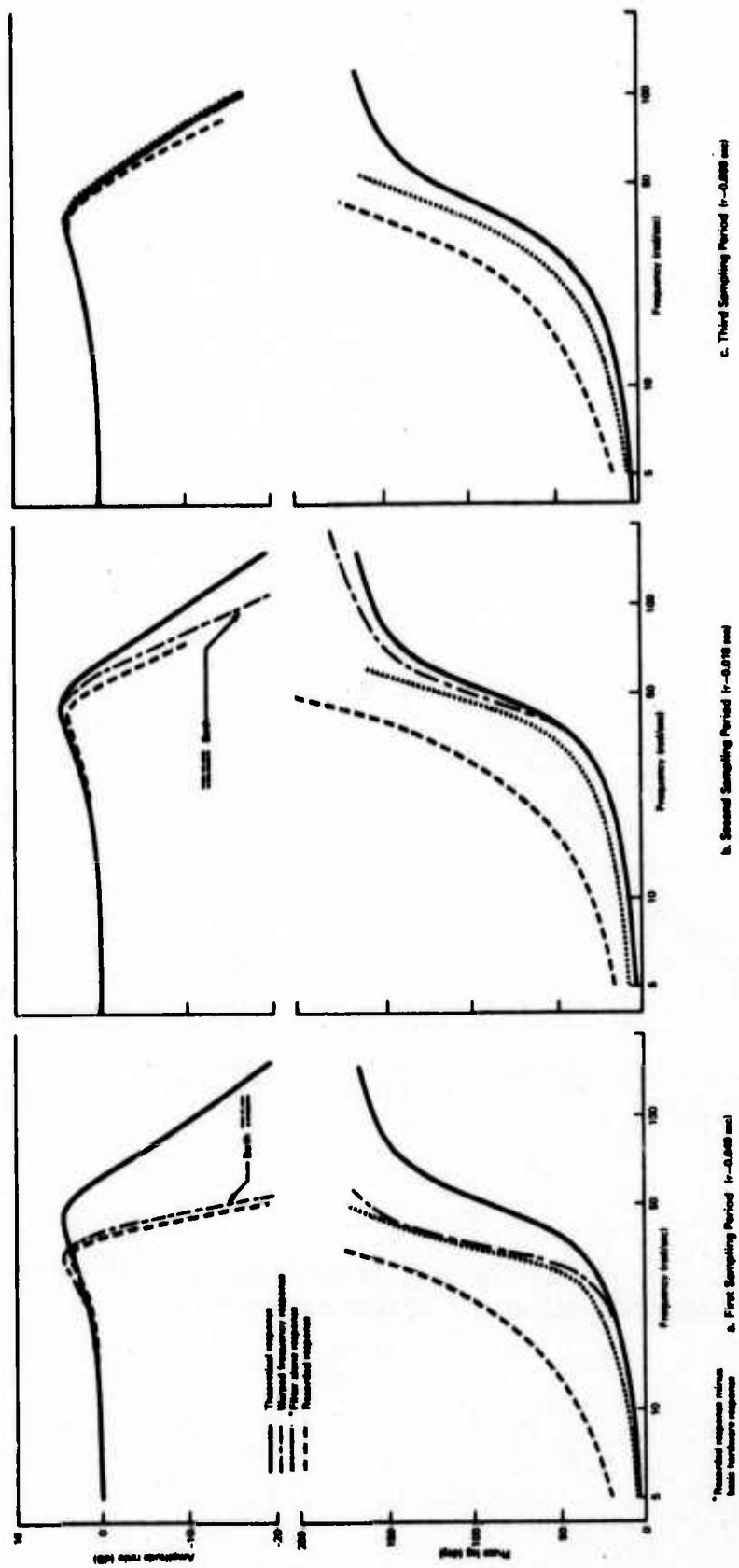
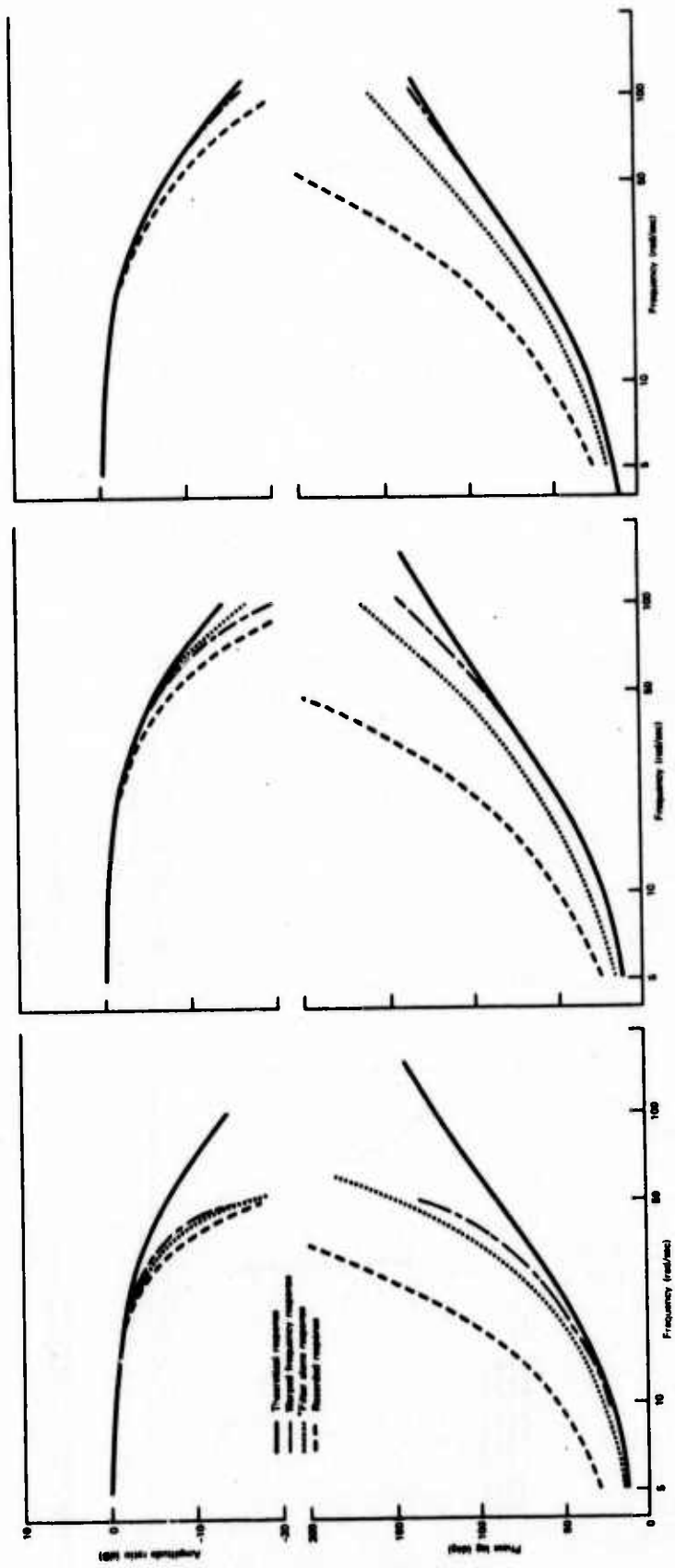


FIGURE 5-42.—WWCS FILTER FREQUENCY RESPONSE FOR $\xi_D = 0.3$, $\omega_D = 50$ RAD/SEC



Recorded response minus basic hardware response
 a. First Sampling Period (1-0.009 sec)
 b. Second Sampling Period (1-0.019 sec)
 c. Third Sampling Period (1-0.029 sec)

FIGURE 5-43.—WWCS FILTER FREQUENCY RESPONSE FOR $\xi_D = 1$, $\omega_D = 50$ RAD/SEC

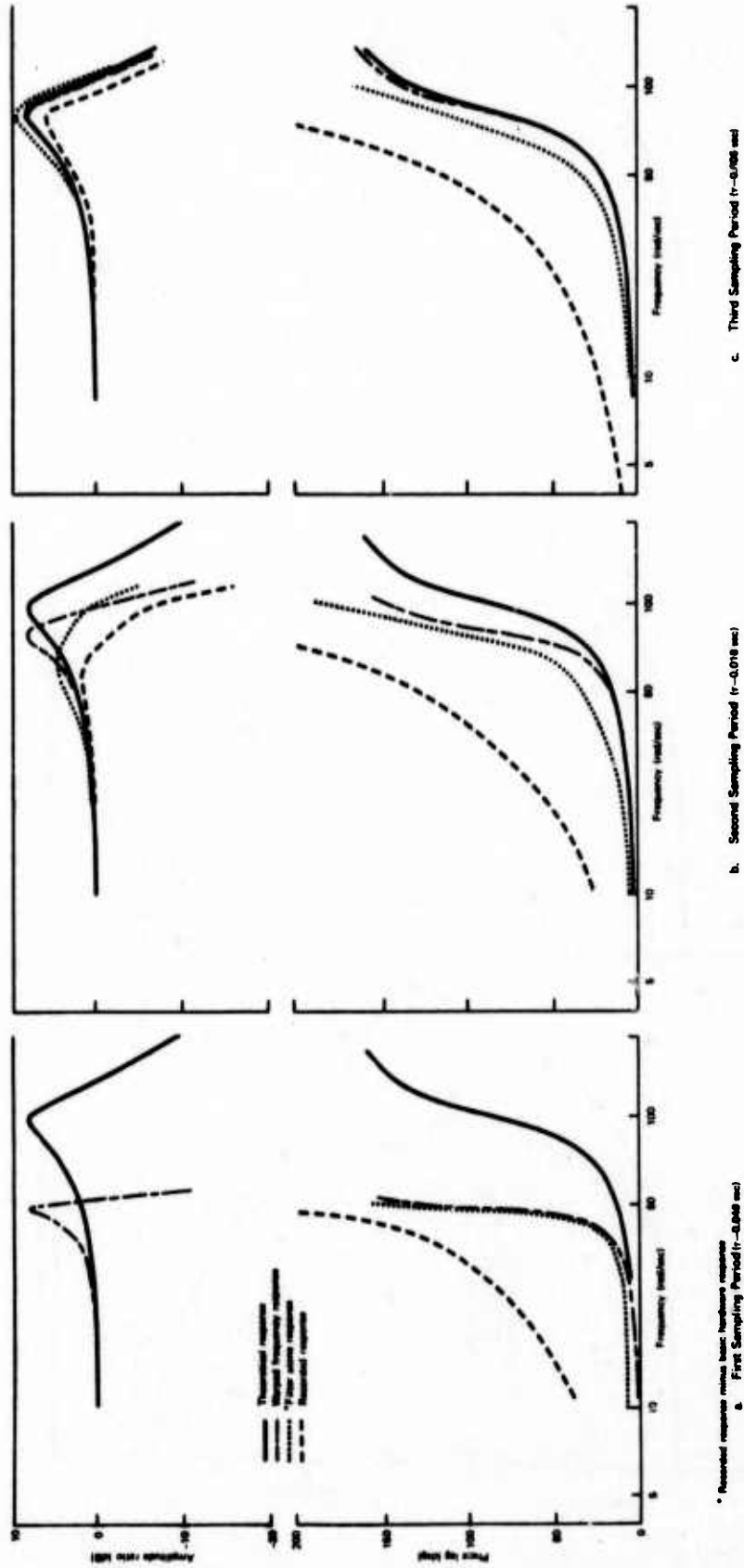
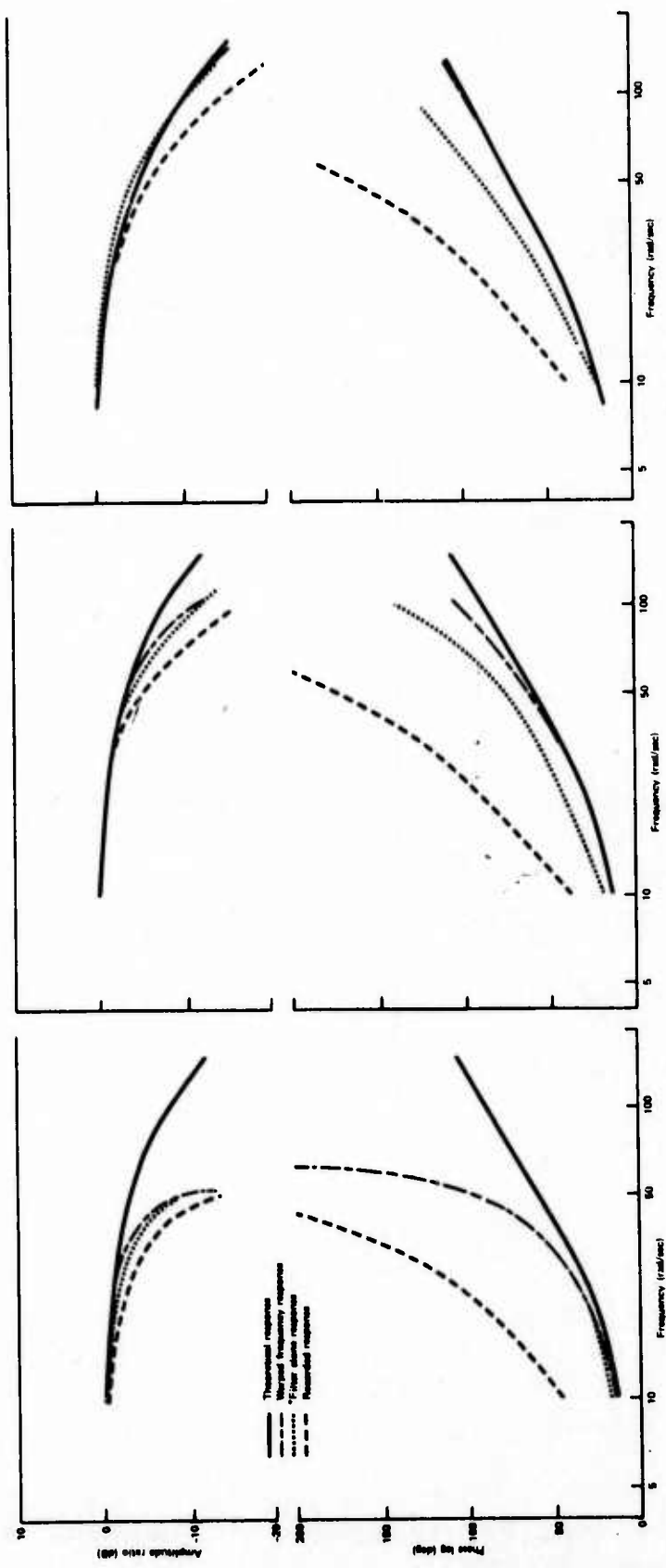


FIGURE 5-44.—WWCS FILTER FREQUENCY RESPONSE FOR $\xi_D = 0.3, \omega_D = 100 \text{ RAD/SEC}$



* Recorded response versus basic hardware response
 a. First Sampling Period $T_s = 0.008 \text{ sec}$
 b. Second Sampling Period $T_s = 0.018 \text{ sec}$
 c. Third Sampling Period $T_s = 0.026 \text{ sec}$

FIGURE 5-45.—WWCS FILTER FREQUENCY RESPONSE FOR $\xi_D = 1$, $\omega_D = 100 \text{ RAD/SEC}$

$\tau = 0.018 \text{ sec}$
& $\tau = 0.049 \text{ sec}$

$$\frac{36}{s^2 + 8.4s + 36}$$

- 16-bit word
- Double precision accumulation
- Truncated outputs

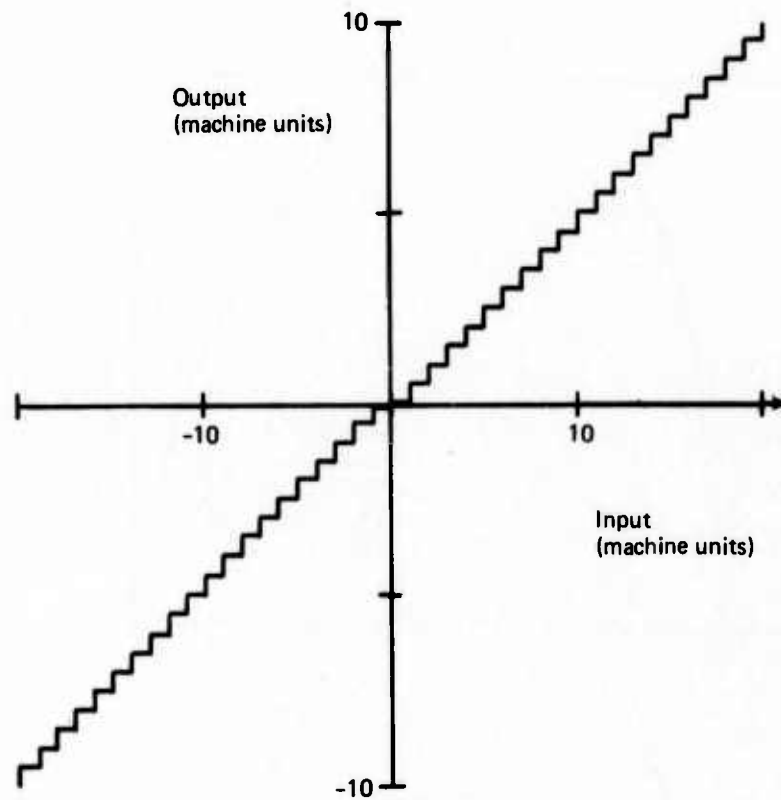
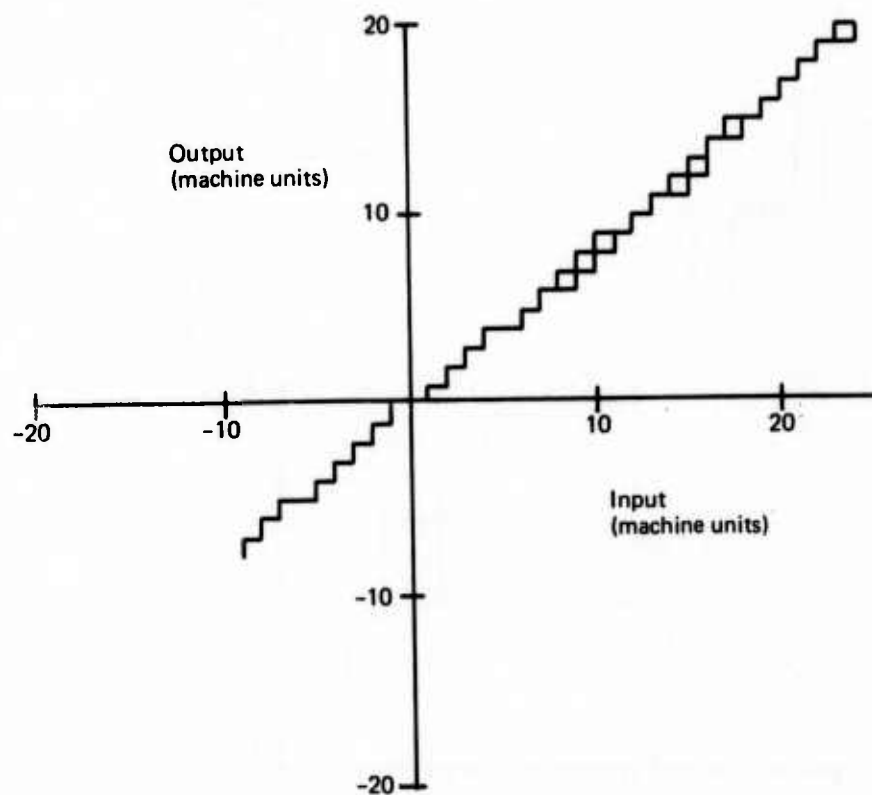


FIGURE 5-46.—HSAS FILTER A STEADY STATE RESPONSE USING BILINEAR TRANSFORMATION WITH COMPENSATION (WWCS)

$$\frac{s^2 + 2.31s + 2.72}{s^2 + 5.62s + 3.1}$$

- 16-bit word
- Double precision accumulation
- Truncated outputs



- Notes: (1) At a given input, the output tends to oscillate between the two values shown with an average period of 1.5 min.
- (2) Little difference was noted between $\tau = 0.018$ and $\tau = 0.049$ sec.

FIGURE 5-47.—HSAS FILTER B STEADY STATE RESPONSE USING BILINEAR TRANSFORMATION WITH COMPENSATION (WWCS)

$$\tau = 0.049 \text{ sec}$$

$$\frac{36}{S^2 + 8.4 S + 36}$$

- 16-bit word
- Double precision accumulation
- Truncated outputs

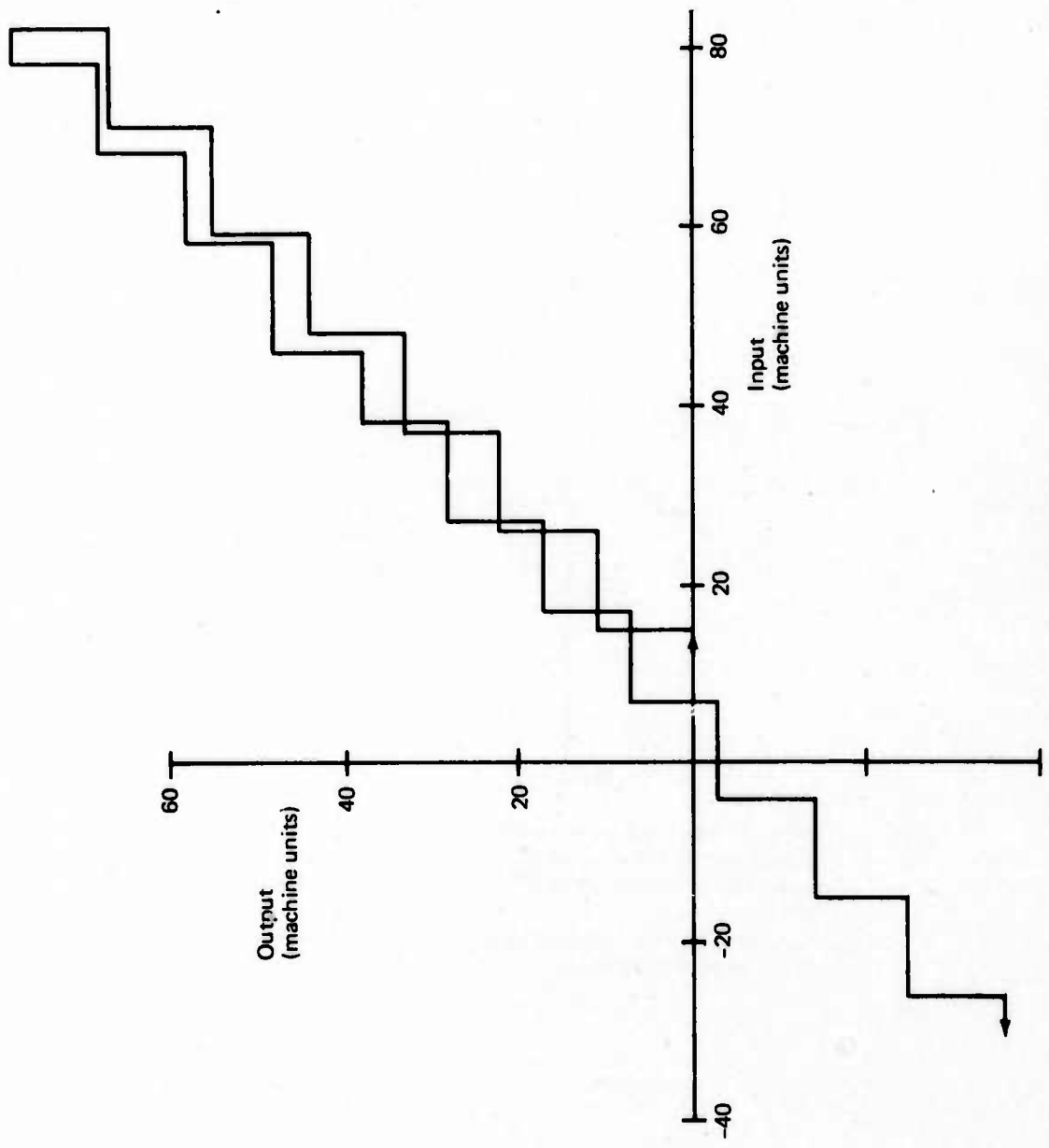


FIGURE 5-48.—FILTER A STEADY STATE RESPONSE FOR $\tau = 0.049$ SEC USING BILINEAR TRANSFORMATION WITHOUT COMPENSATION (WWCS)

Bilinear Z transform filter
 $\tau = 0.018$ sec

- 16-bit word
- Double precision accumulation
- Truncated outputs

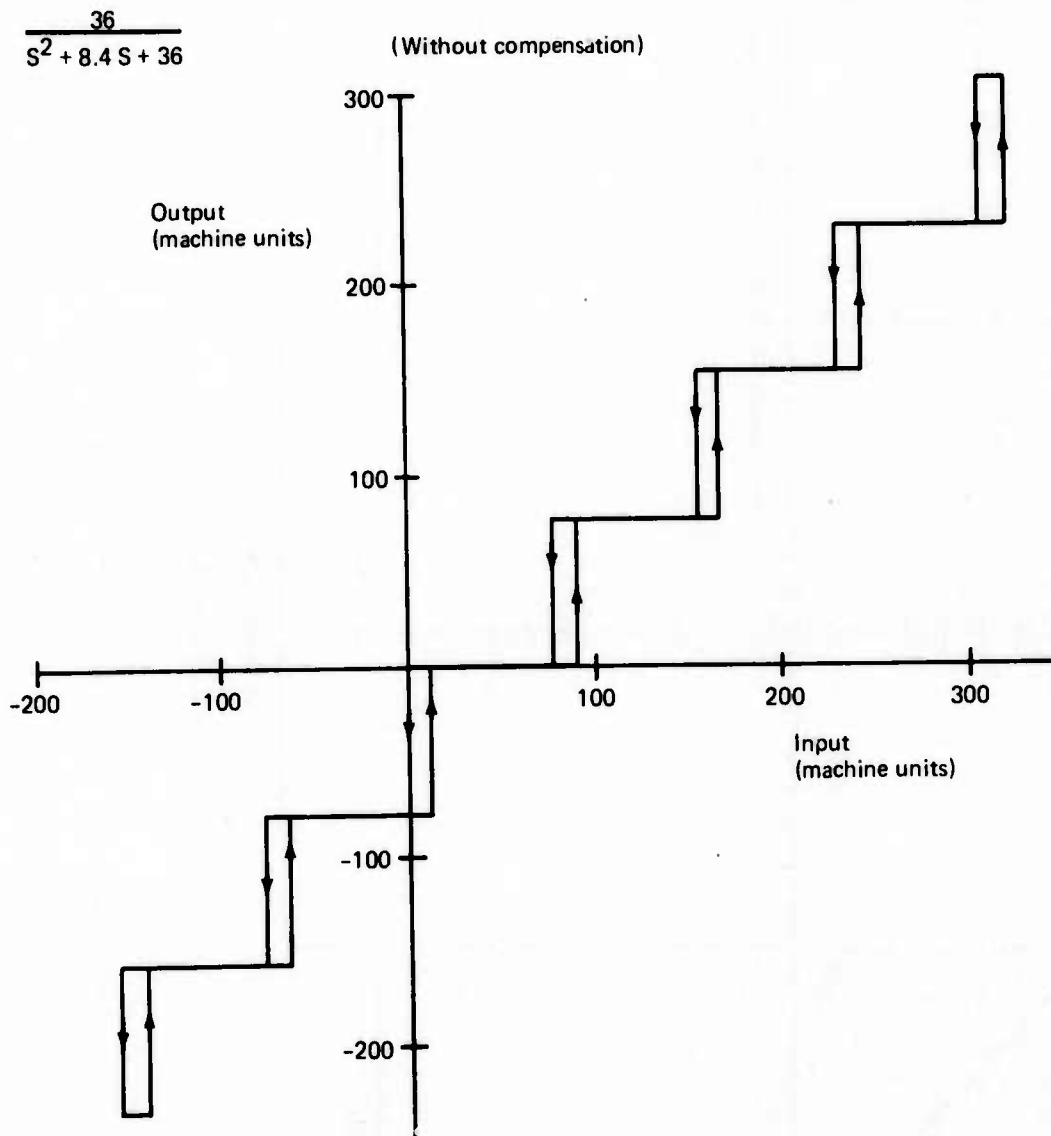


FIGURE 5-49.—FILTER A STEADY STATE RESPONSE FOR $\tau = 0.018$ SEC USING BILINEAR TRANSFORMATION WITHOUT COMPENSATION (WWCS)

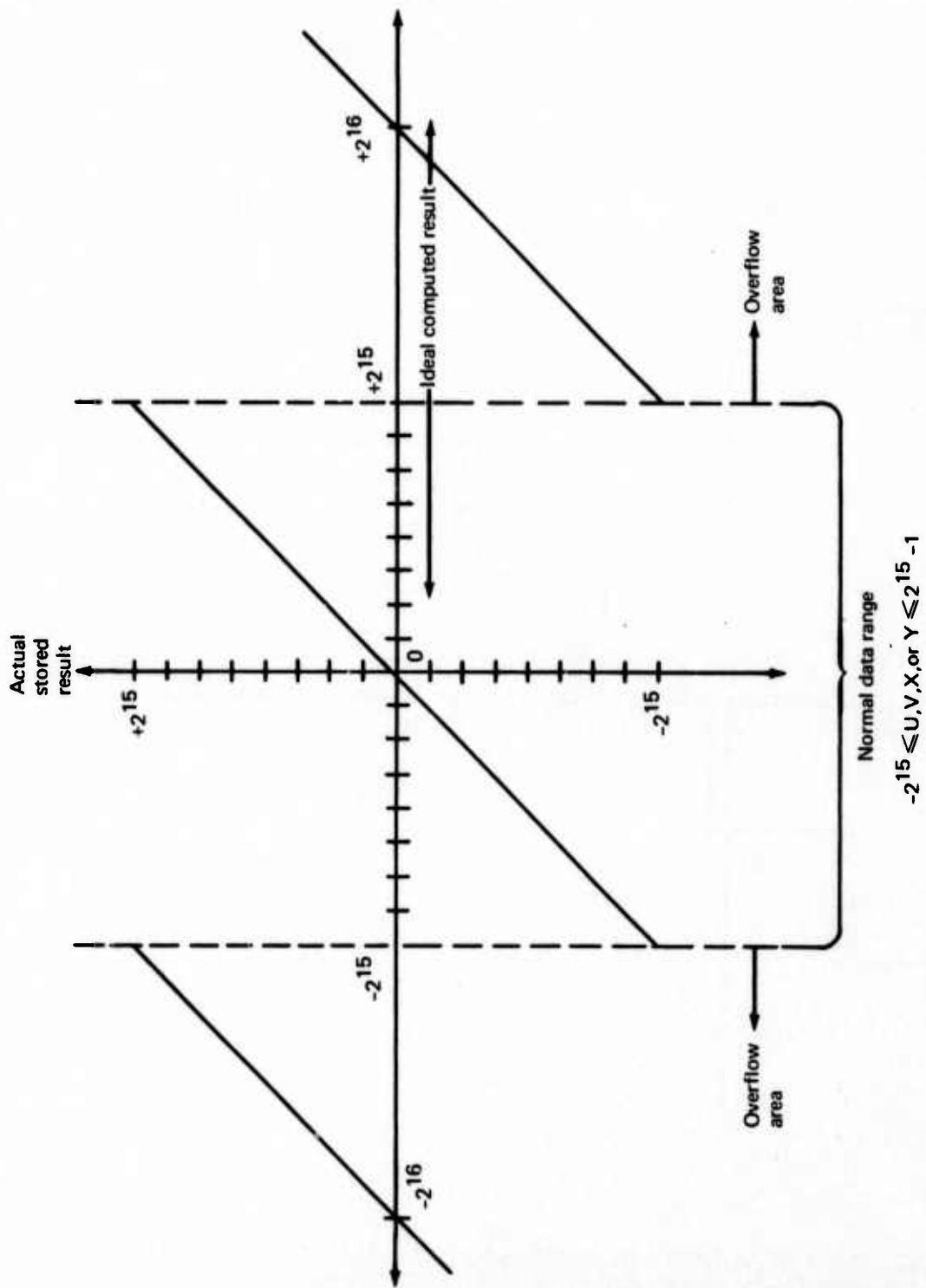


FIGURE 5-50. — ICPS OVERFLOW CHARACTERISTICS

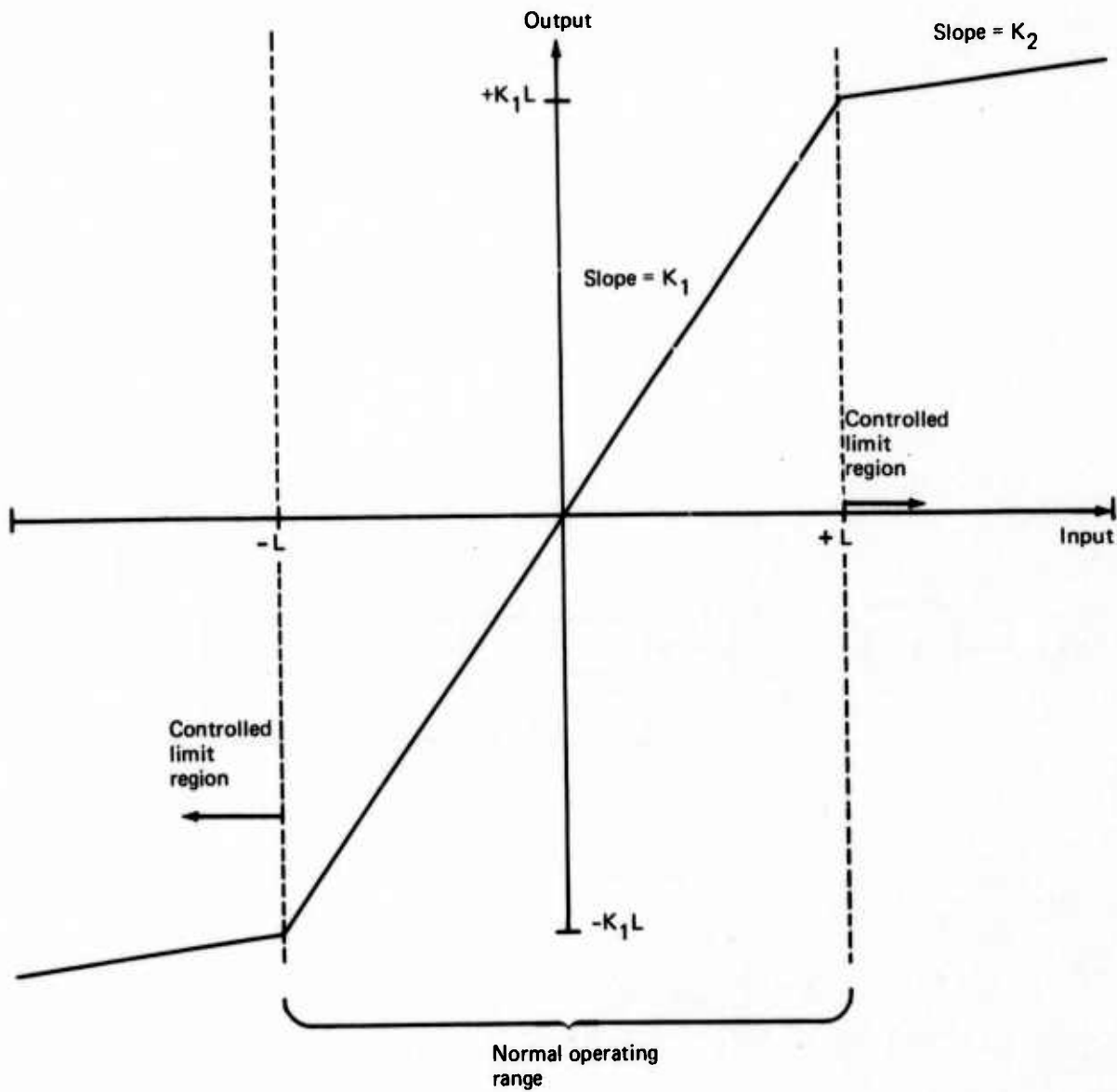
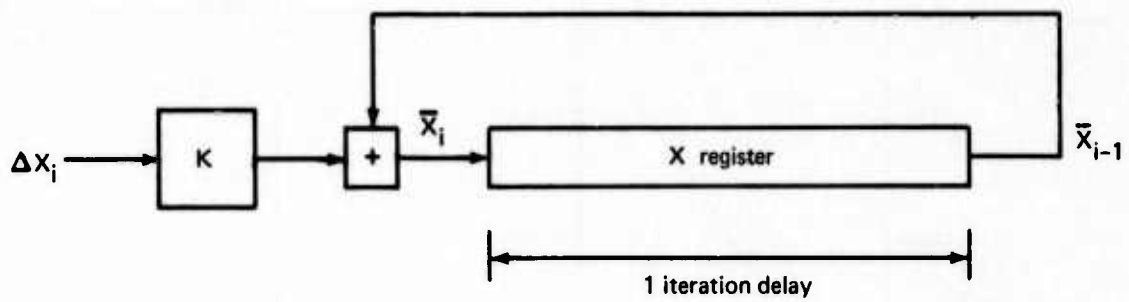


FIGURE 5-51.—ANALOG SYSTEM SATURATING LIMITER



$$\bar{X}_i = K X_i = \sum_{j=1}^i K \Delta X_j$$

$$X_i = \sum_{j=1}^i \Delta X_j = X_{i-1} + \Delta X_i$$

FIGURE 5-52.—ICPS X REGISTER EXAMPLE

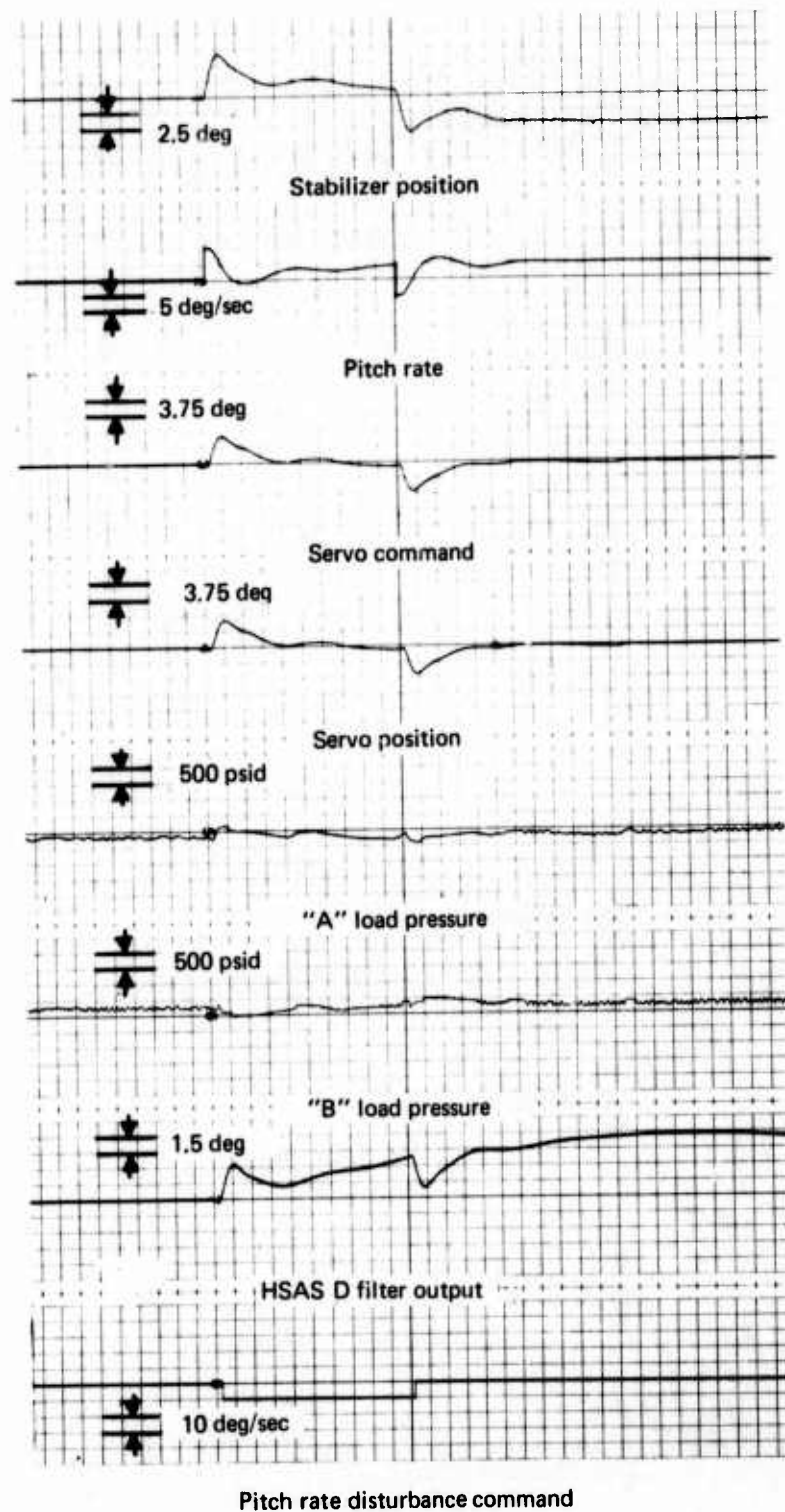


FIGURE 5-53.- BASELINE RESPONSE WITHOUT OVERFLOW

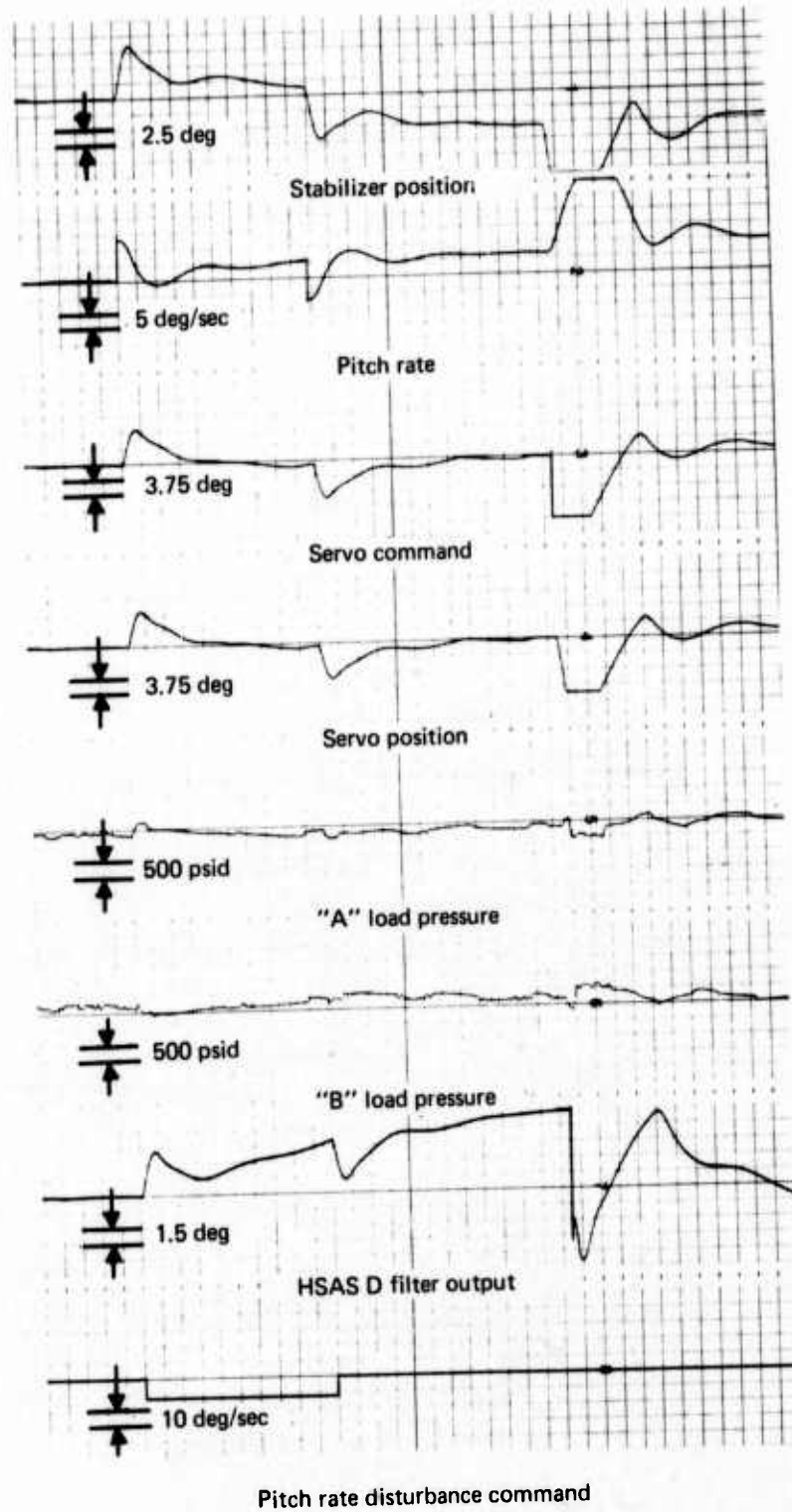


FIGURE 5-54.—OVERFLOW AT INPUT TO HSAS D FILTER WITHOUT THE OVERFLOW ROUTINE

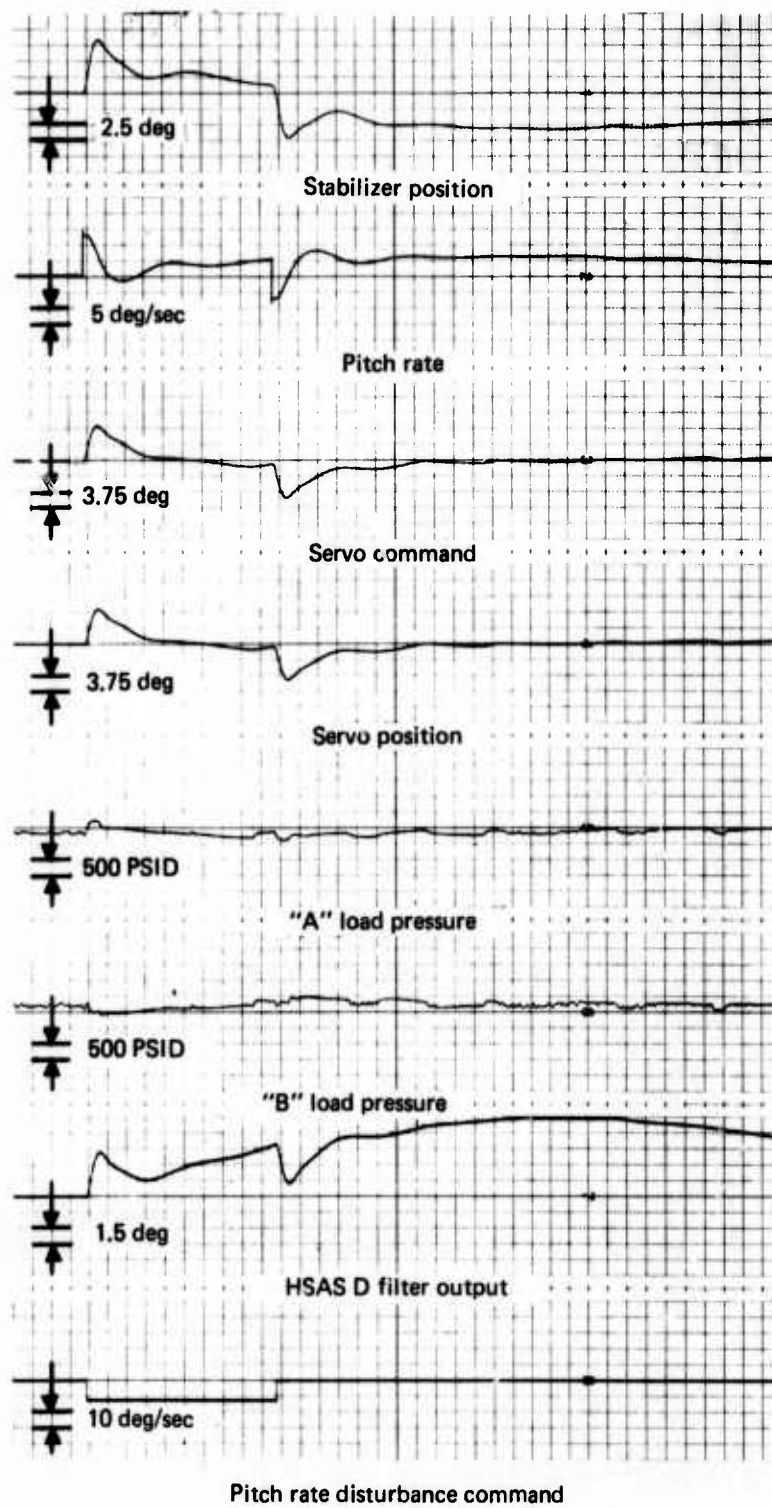


FIGURE 5-55.—OVERFLOW AT INPUT TO HSAS D FILTER WITH THE OVERFLOW ROUTINE ACTIVE

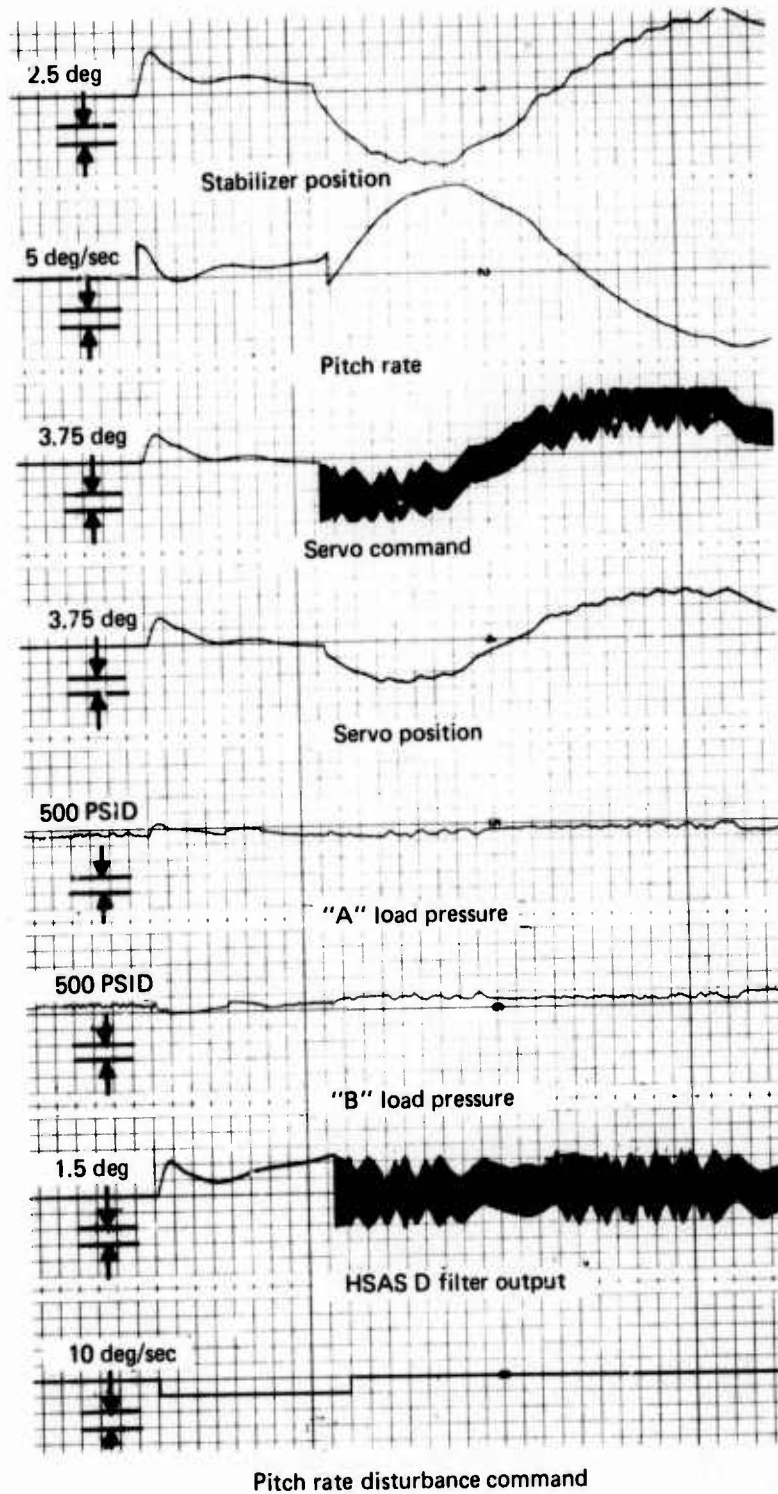


FIGURE 5-56.—OVERFLOW WITHIN RECURSIVE HSAS D FILTER WITH OVERFLOW ROUTINE ACTIVE

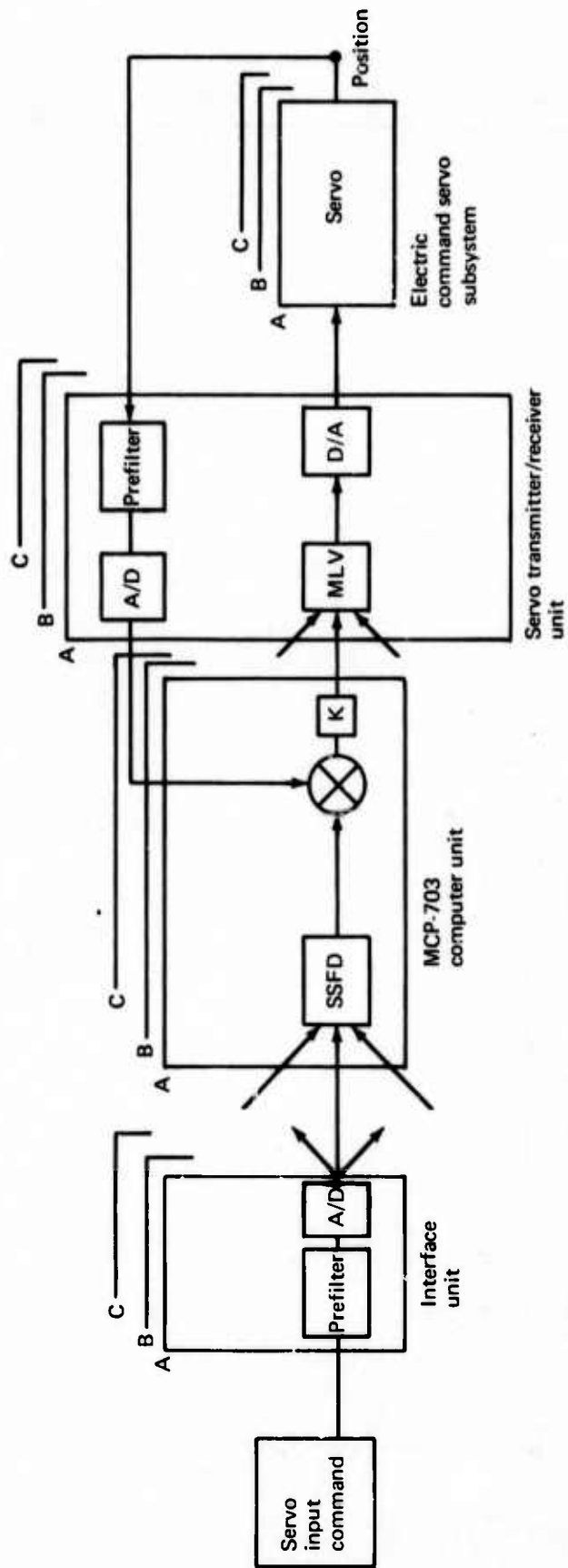


FIGURE 5-57.—STRU LABORATORY TEST CONFIGURATION

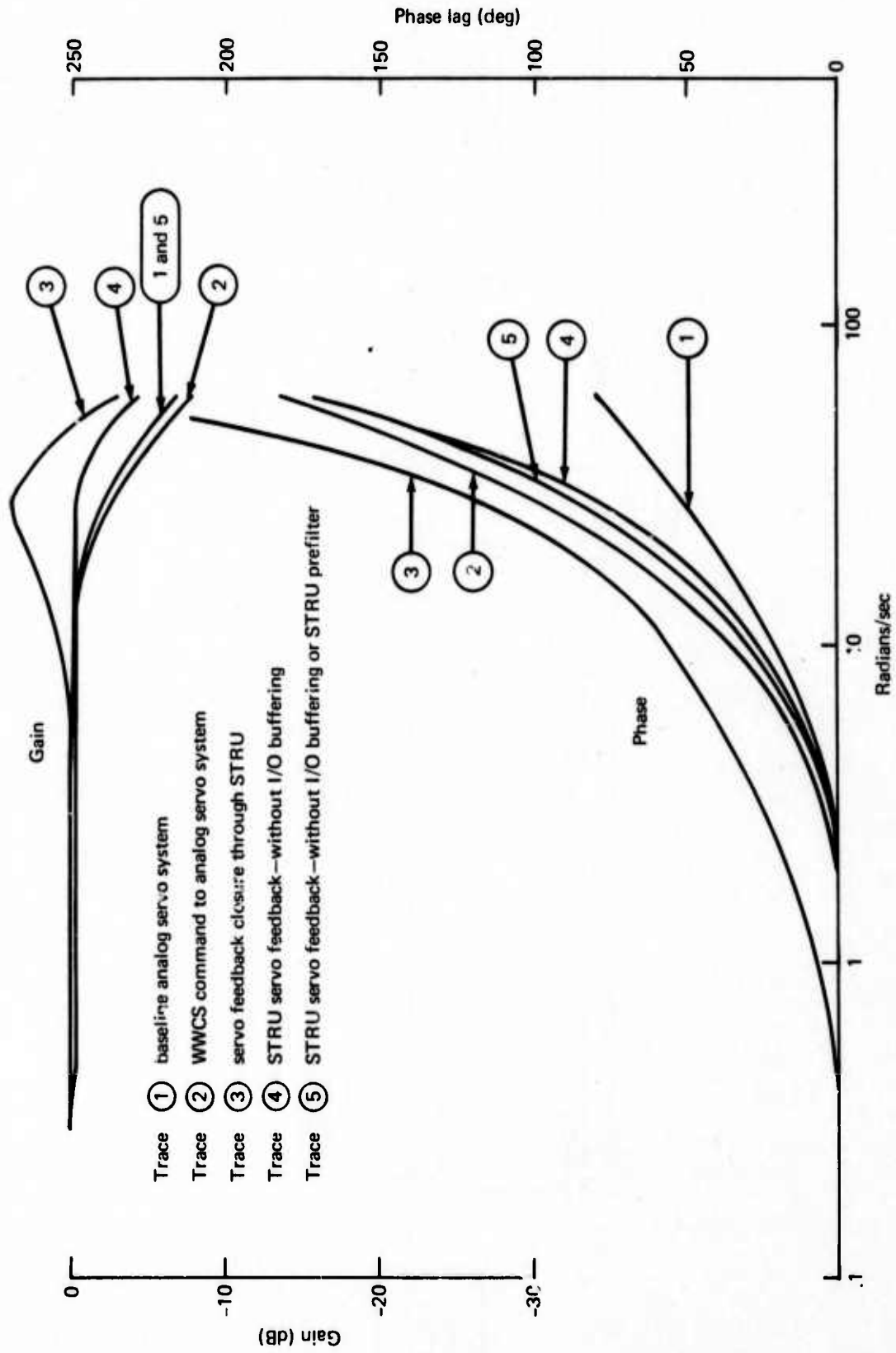


FIGURE 5-58.—STRU STUDY SERVO LOOP FREQUENCY RESPONSE COMPARISONS

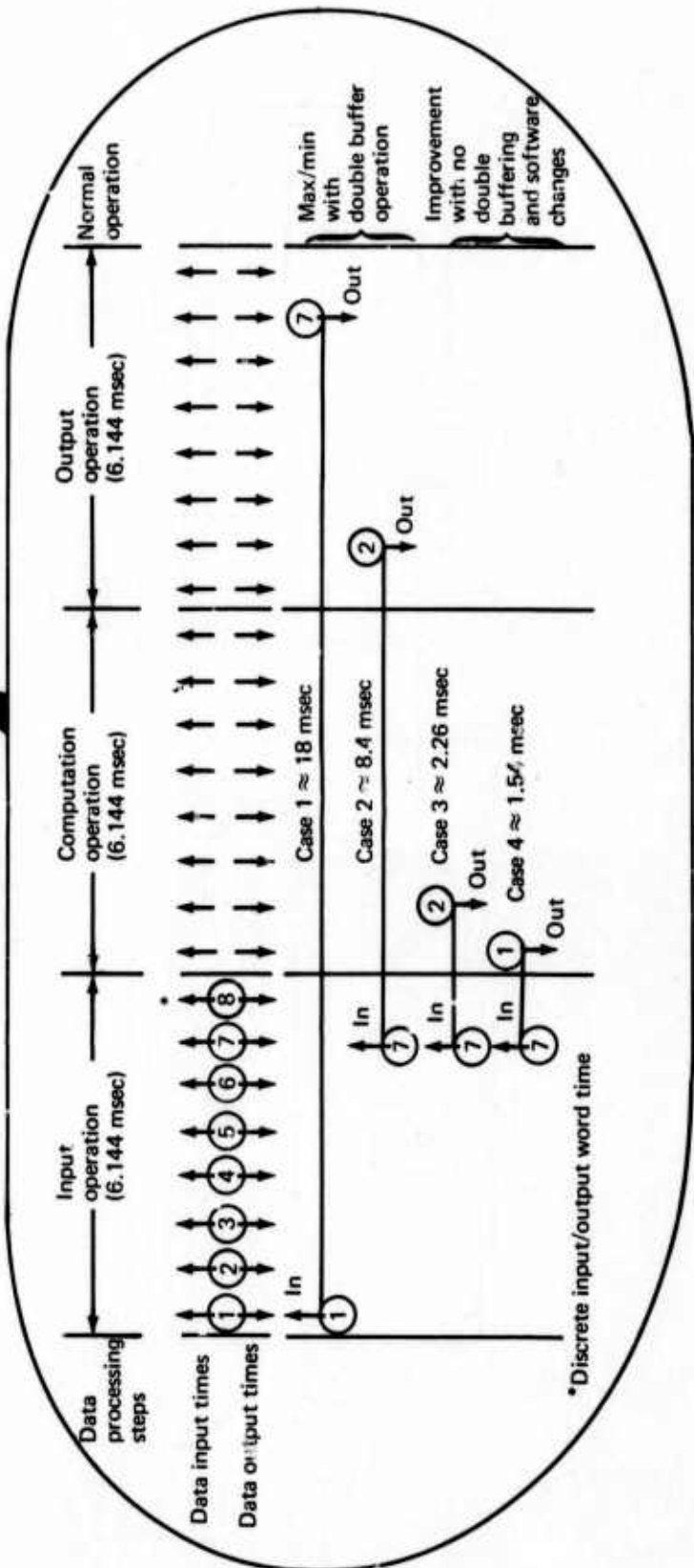
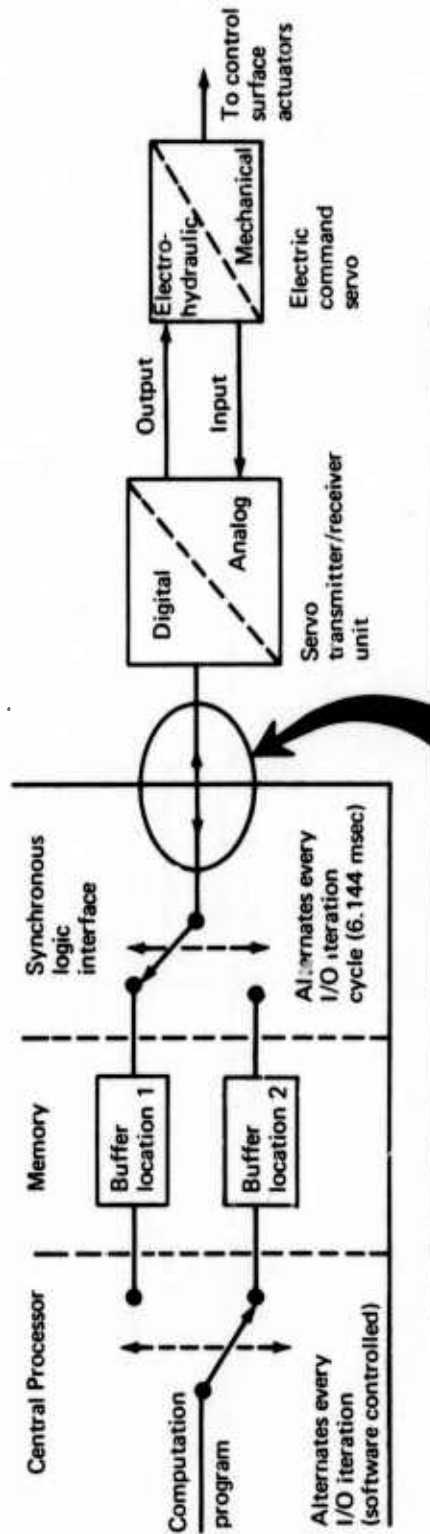


FIGURE 5-59.—STRU/COMPUTER UNIT DATA FLOW DIAGRAM

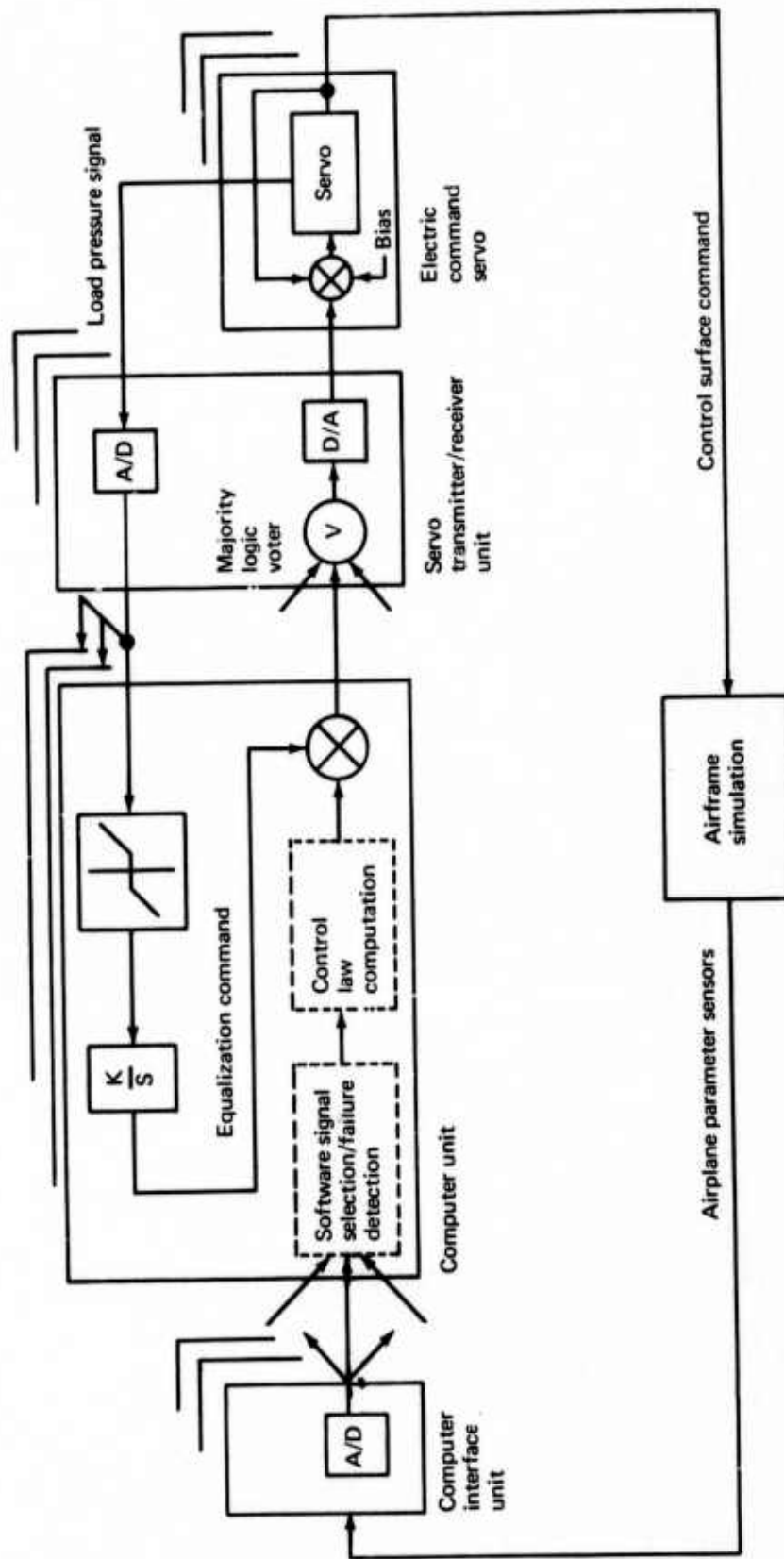


FIGURE 5-60.—REDUNDANT SERVO EQUALIZATION TEST CONFIGURATION

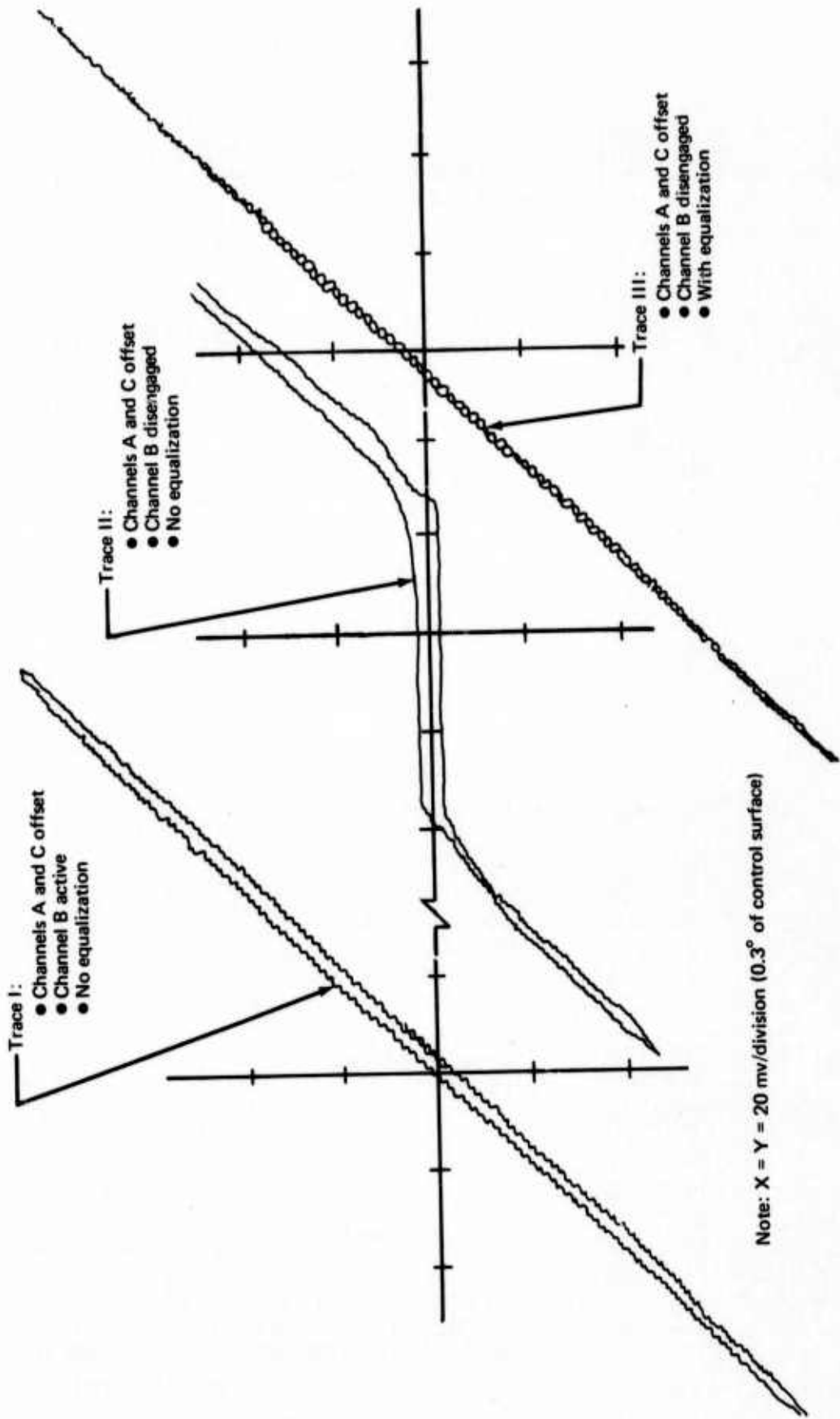


FIGURE 5-61.—ECS STEADY STATE RESPONSE WITH AND WITHOUT EQUALIZATION

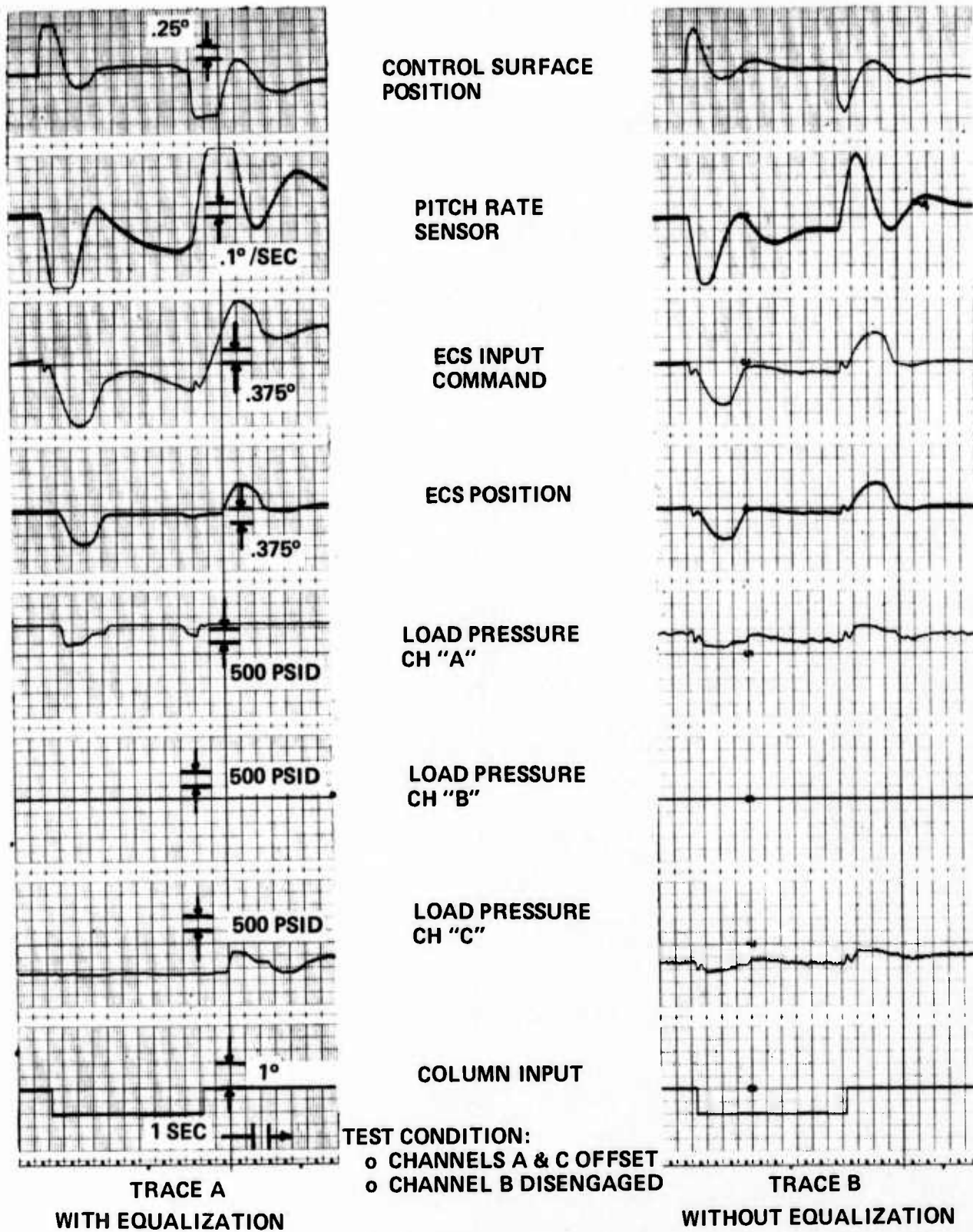


FIGURE 5-62.—AIRPLANE RESPONSE WITH AND WITHOUT EQUALIZATION

Memory page*		Memory page*	
Constants/variables	1		17
Master executive and routines for: ● Software loader ● CMCU drive ● SSCU drive	2	Flight control laws (cont)	18
	3		19
	4	Preflight test (peripheral hardware tests)	20
	5		21
	6		22
7	23		
8	24		
Built-in-test (BIT)	9	Spare	25
	10		26
	11		27
Sensor signal selection/ failure detection and system failure status	12		28
	13		29
Flight control laws	14	Input/output variables and discrettes data	30
	15		31
	16		32

*One memory page = 256 words
Total memory = 8,192 words (≈ 8K)

FIGURE 6-1.—WWCS MEMORY ASSIGNMENT

Block	'X' number	Name	Start date	End date	Purpose	'X' saved	'P' changed
EXEC	01	Hill	12/15/73	1/5/74	Correct error timer interrupt	No	Yes
Flight control	02	Fulton	12/15/73	1/20/74	Correct error HSAS filter	No	Yes
Redundancy	03	Berwick	1/25/74	2/15/74	New SSFD	Yes	No
.FC/EXEC	04	Maeshiro	1/29/74		Overflow protect		

SAMPLE

FIGURE 6-2.—EXPERIMENTAL PROGRAMS LOG

'P' Number	Block(s)	Reason	Date
02	EXEC	Time interrupt error—corrected.	1/5/74
03	Flight controls	HSAS filter—improvement.	1/20/74

SAMPLE

FIGURE 6.3.—OPERATIONAL PROGRAM LOG

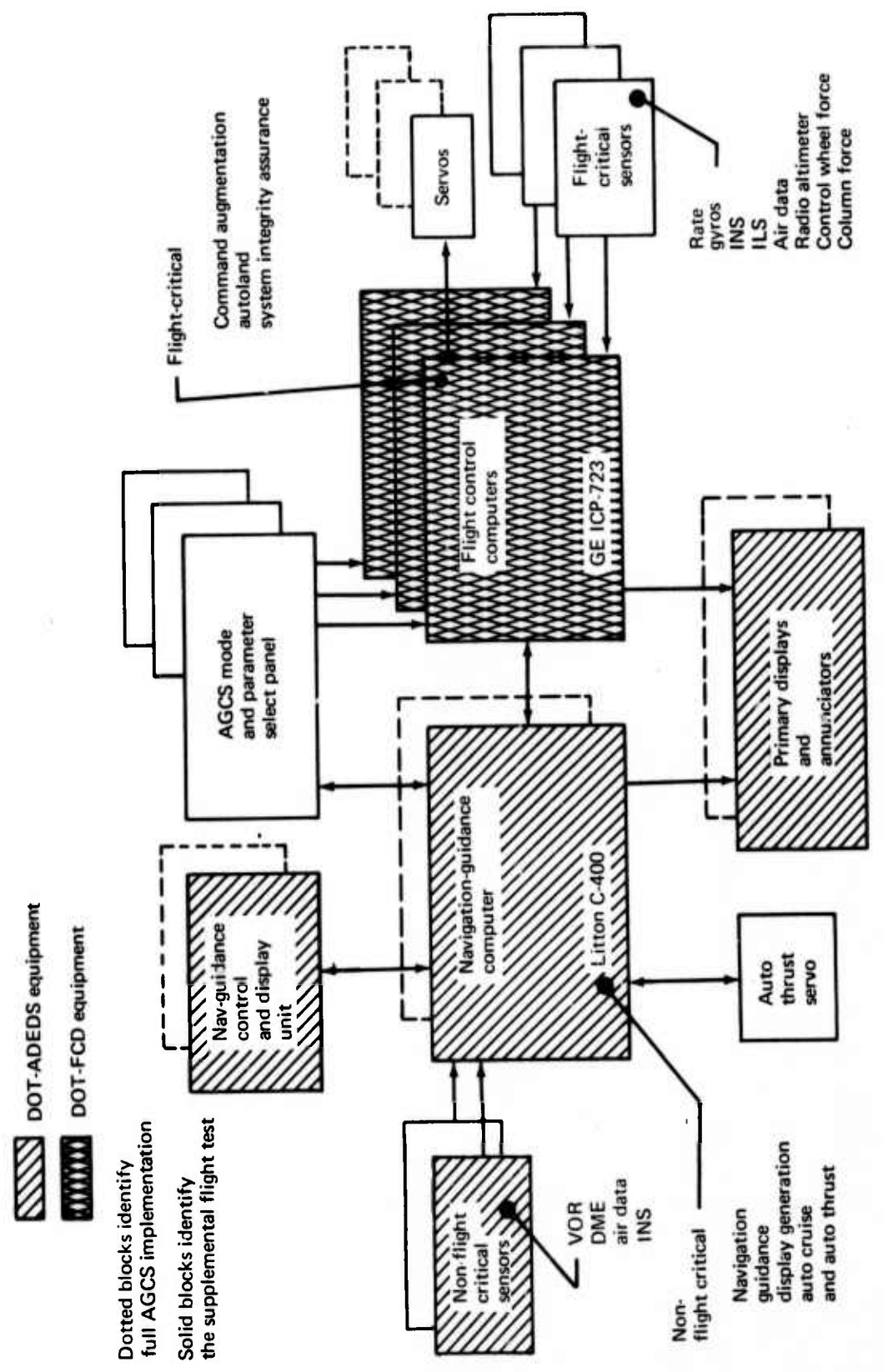


FIGURE 7-1.—AGCS CONCEPT

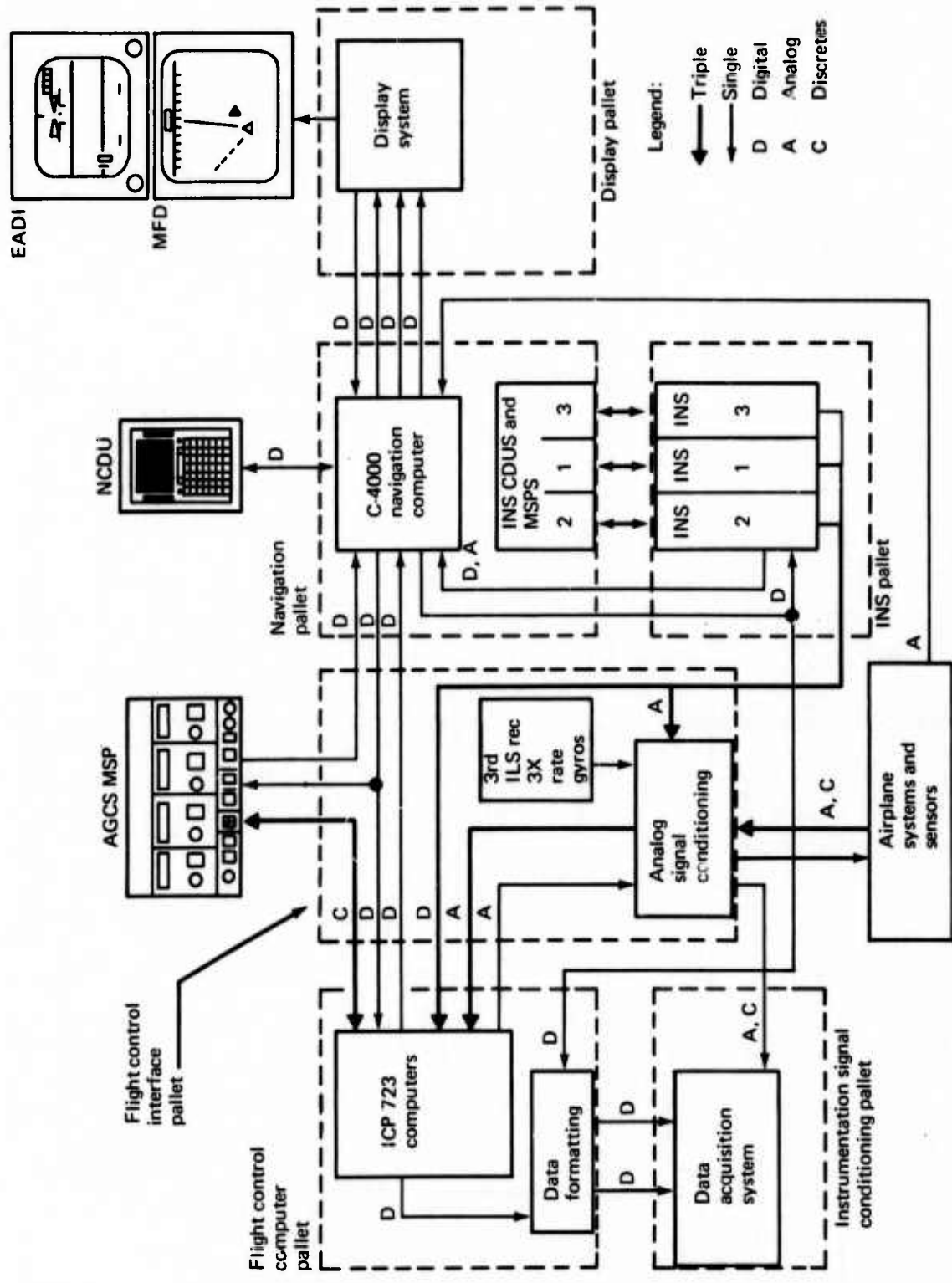


FIGURE 7-2.—AGCS INTERFACES

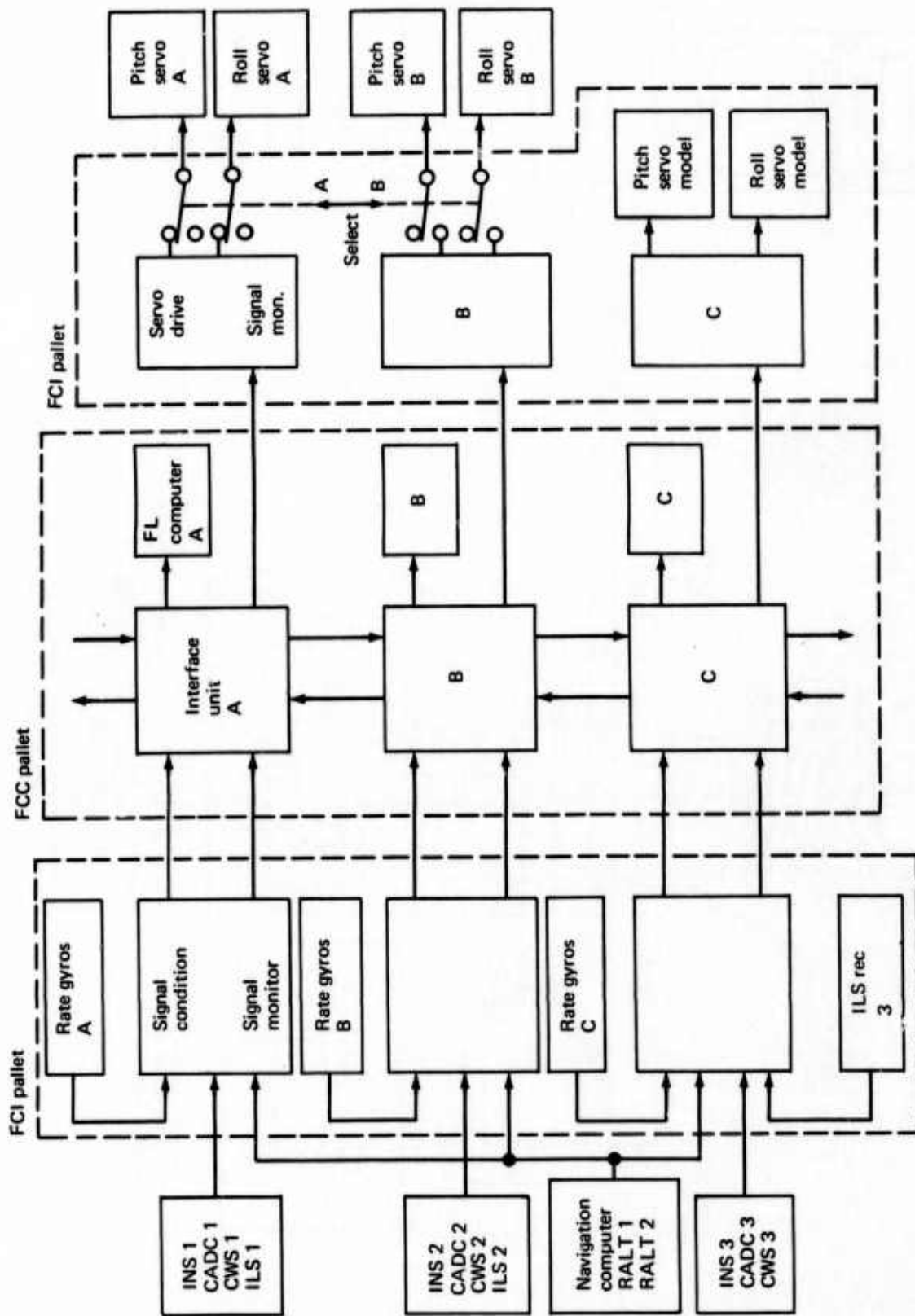


FIGURE 7-3.--AUTOMATIC FLIGHT CONTROL SYSTEM

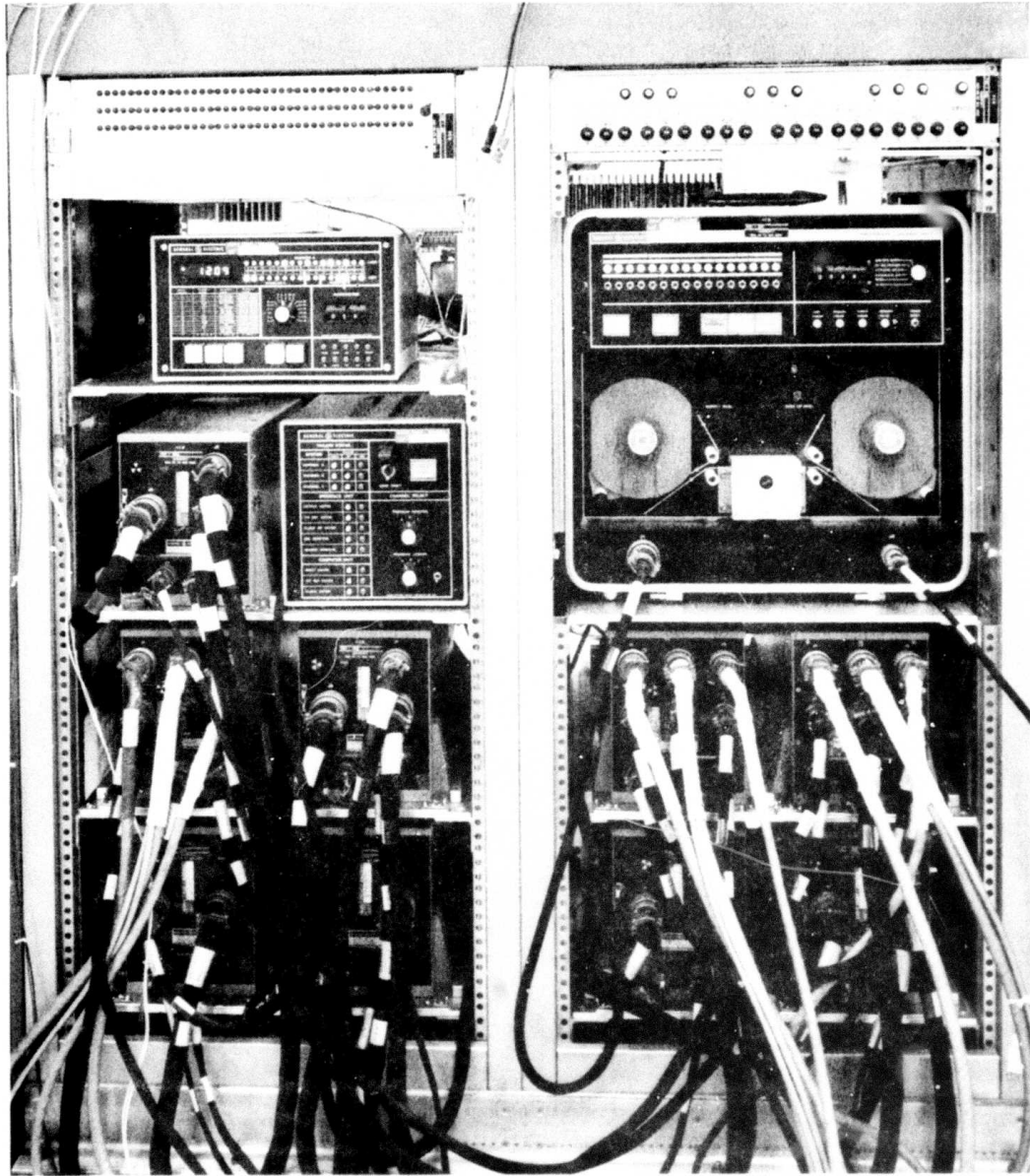


FIGURE 7-4.—FLIGHT CONTROL COMPUTER PALLET

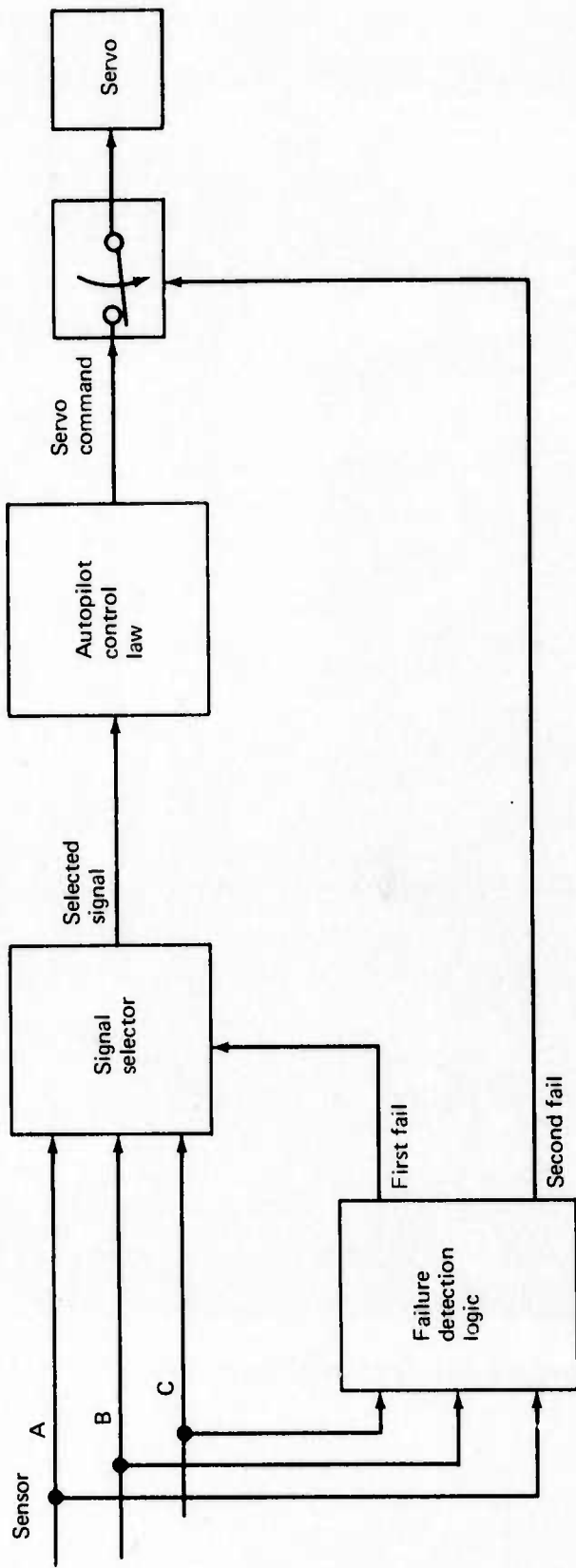


FIGURE 7-5.—ICP-723 SIGNAL SELECTION SCHEME

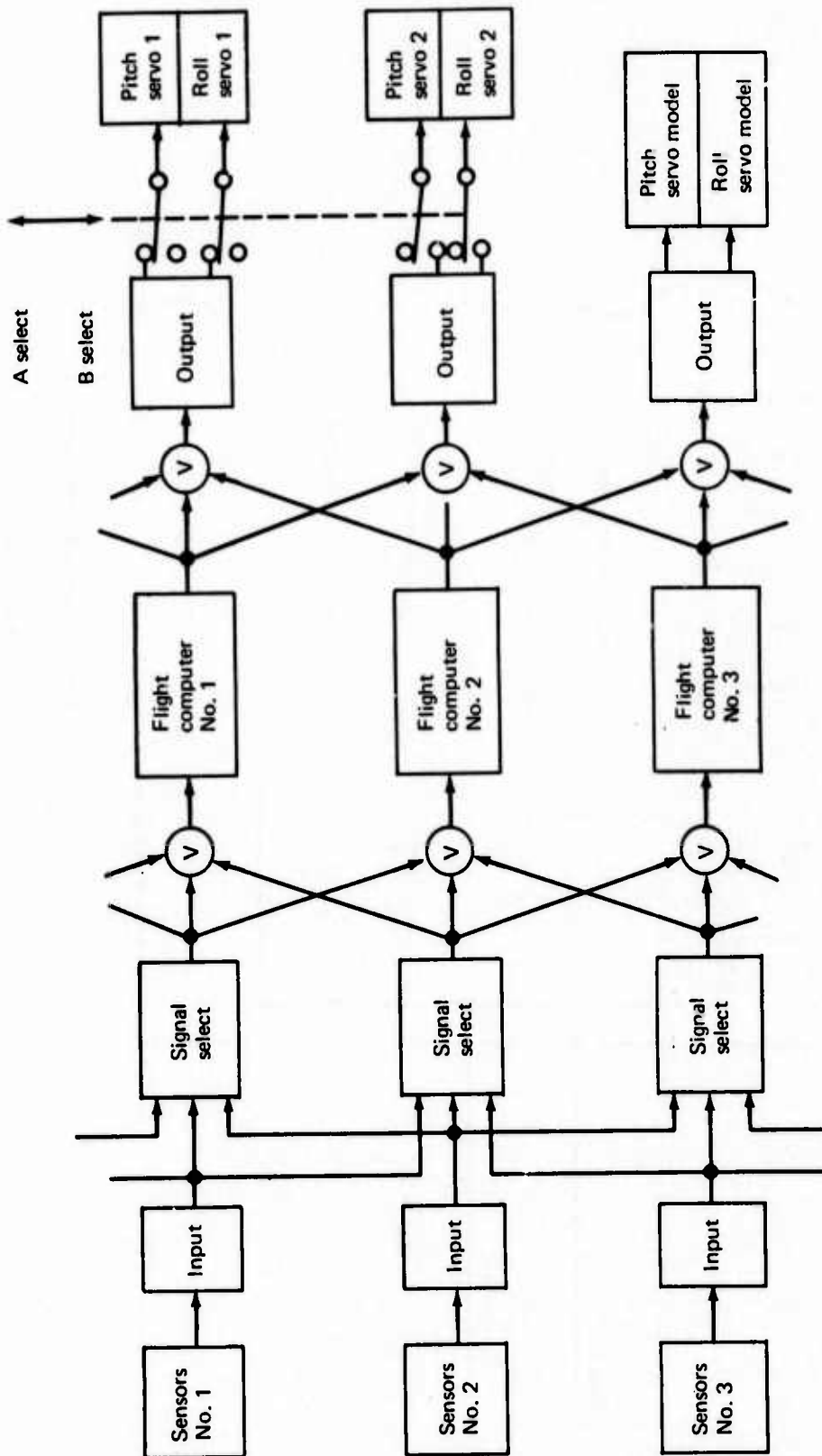


FIGURE 7-6.—ICP-723 SIGNAL VOTING CONFIGURATION

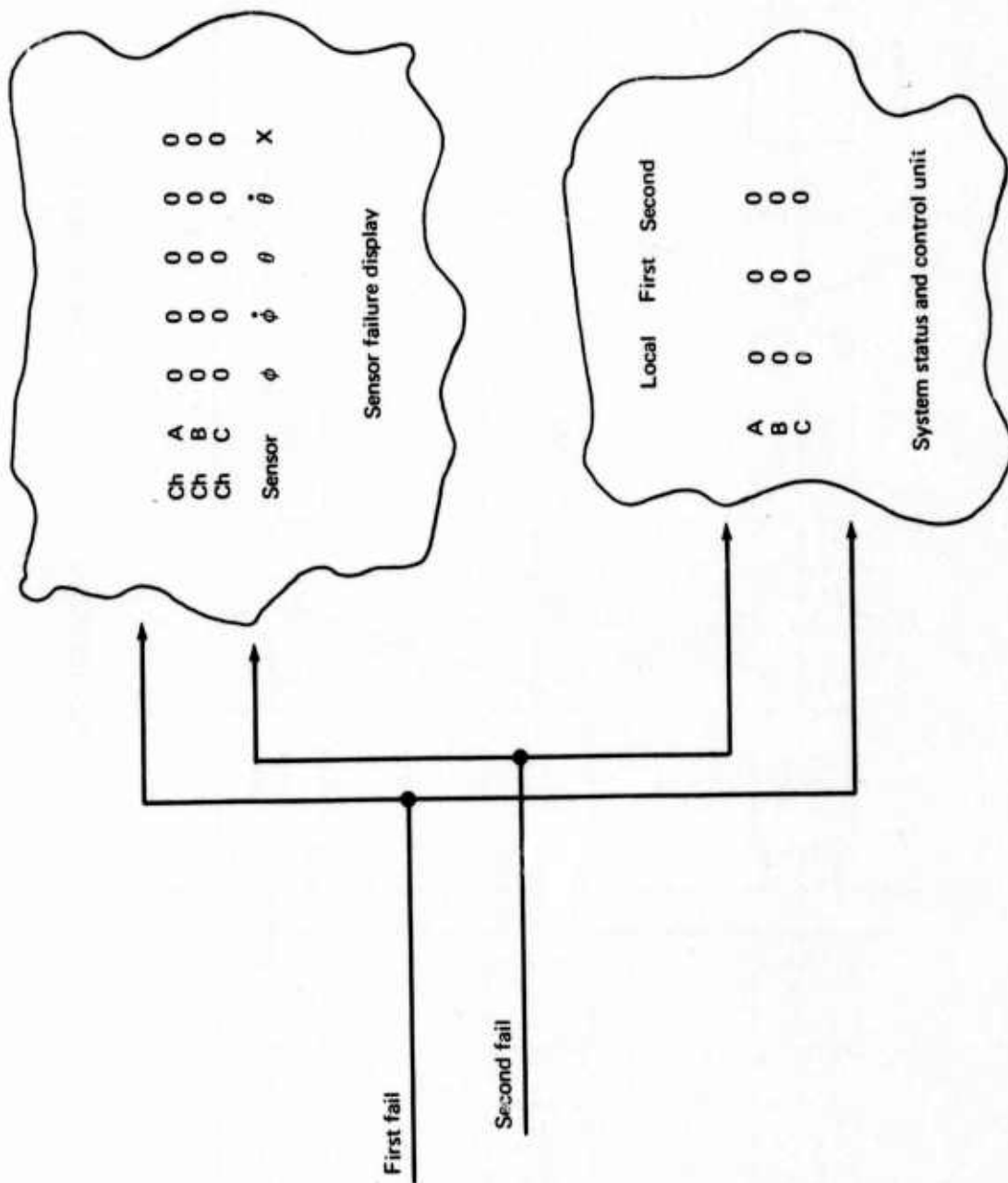


FIGURE 7-7.—AGCS ICP-723 FAILURE ANNUNCIATION

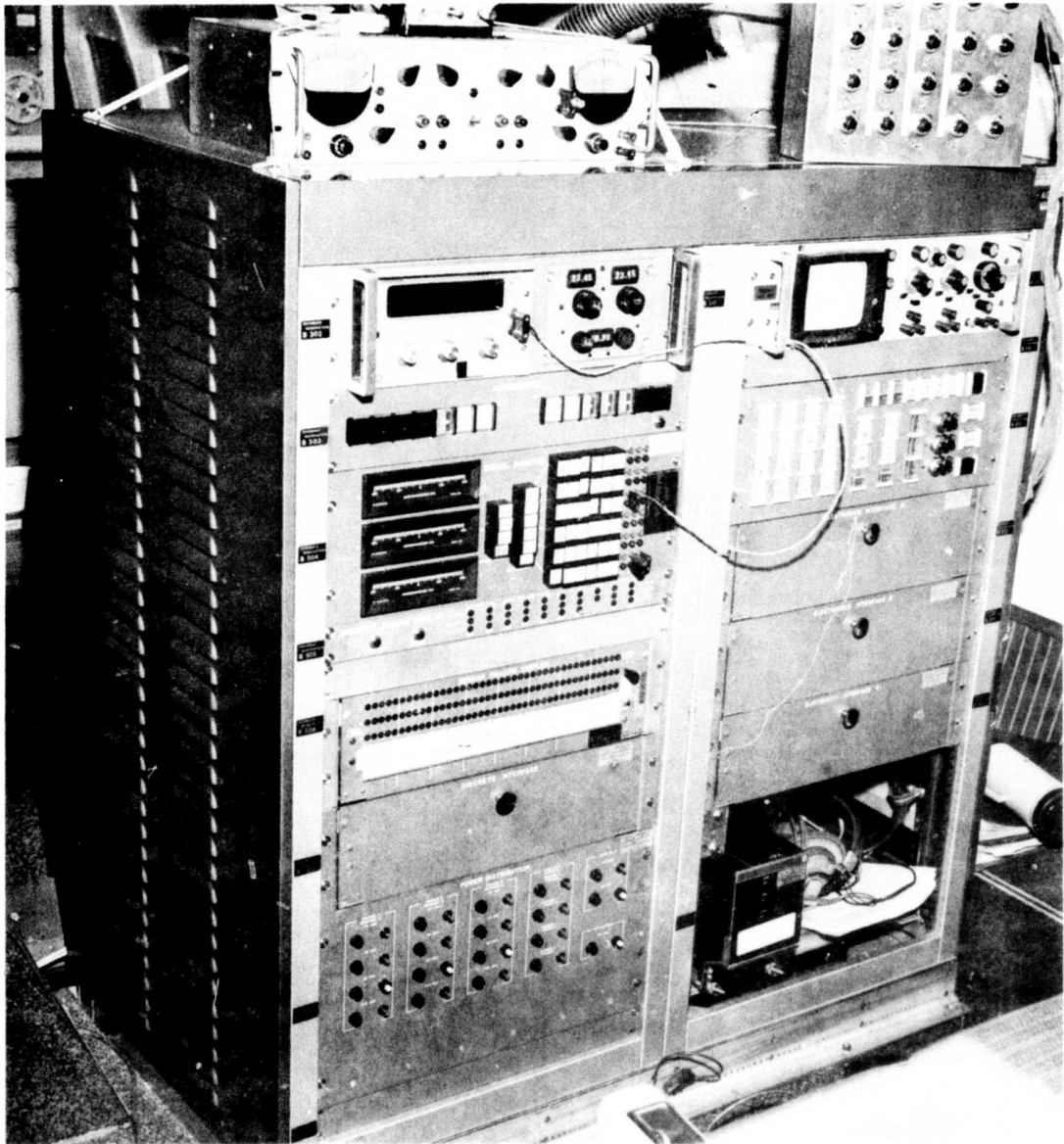


FIGURE 7.8.—FCI PALLET

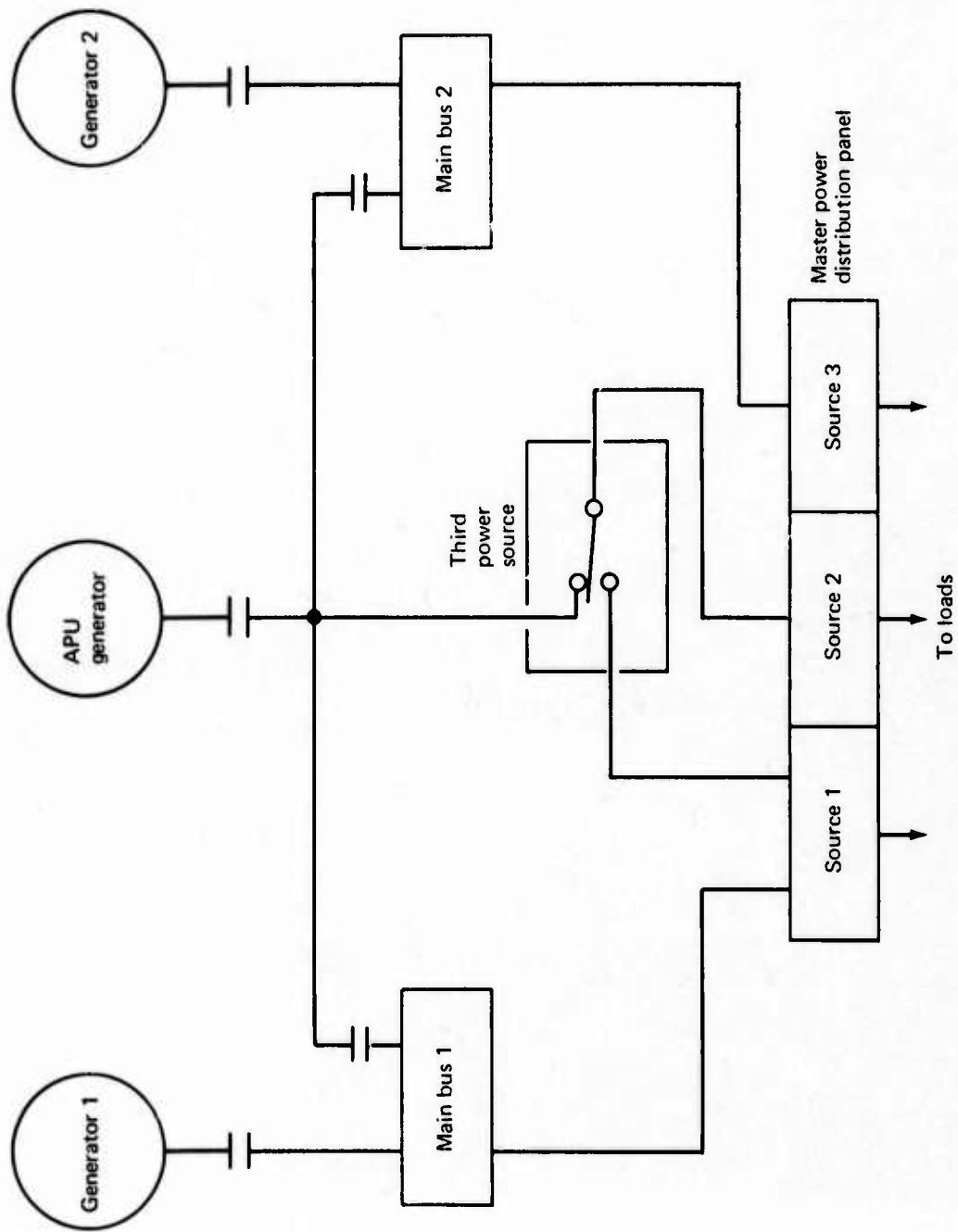


FIGURE 7-9.—MODIFICATION TO 737 POWER SYSTEM
(AGCS SUPPLEMENTAL FLIGHT TEST CONFIGURATION)

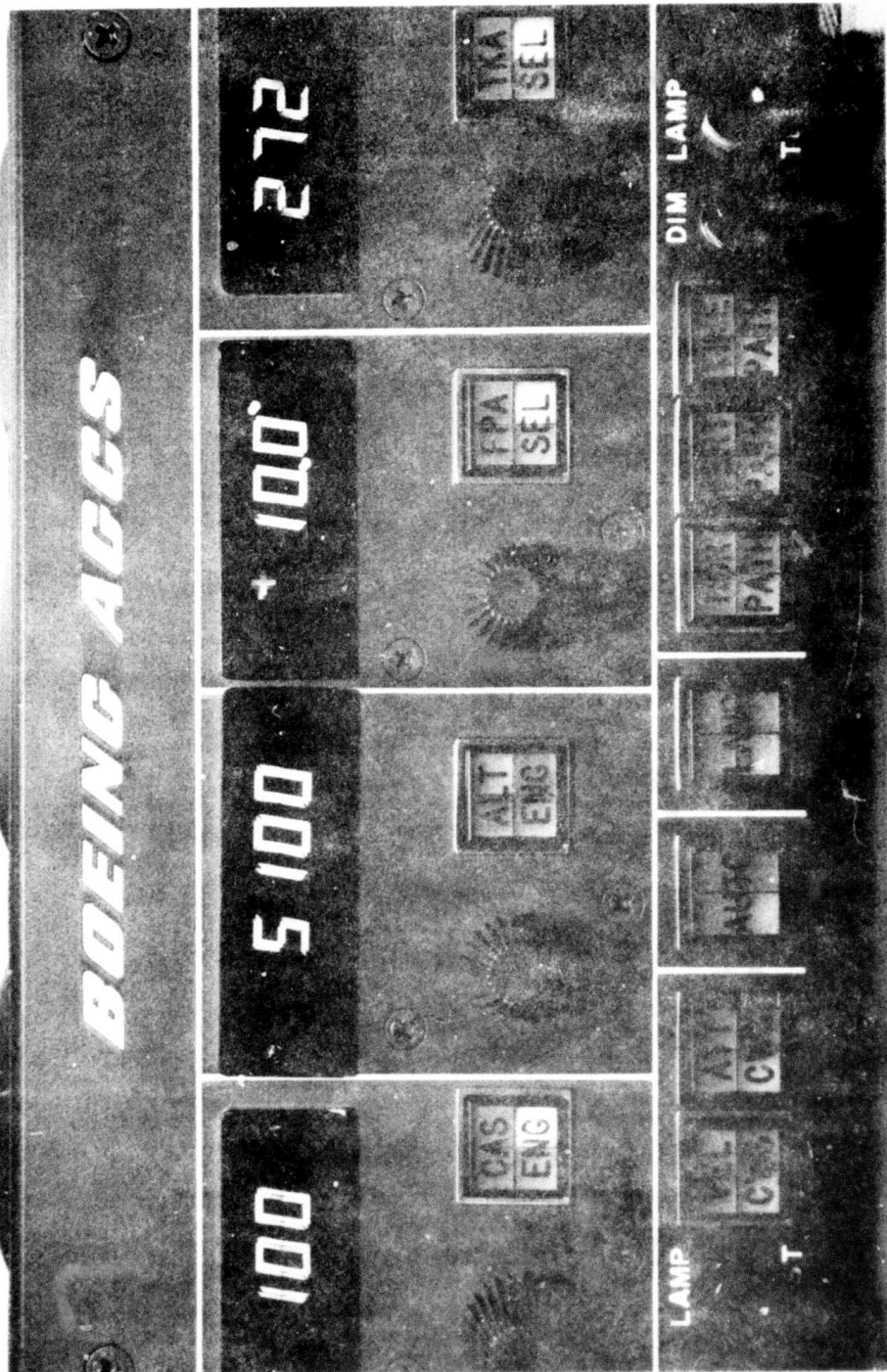


FIGURE 7-10.—AGCS MODE SELECT PANEL

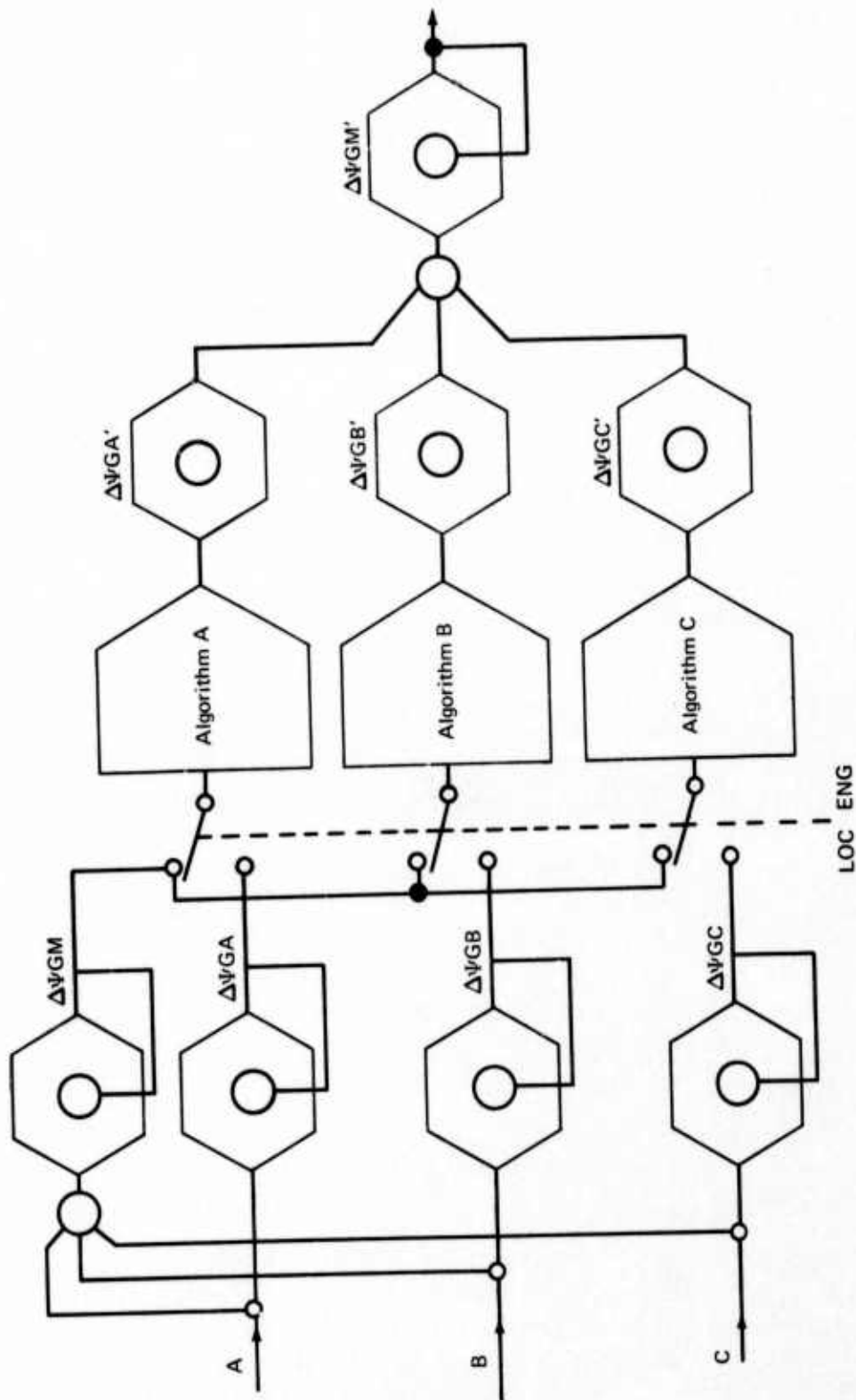


FIGURE 7-11.—DELTA TRACK ANGLE MONITORING SCHEME

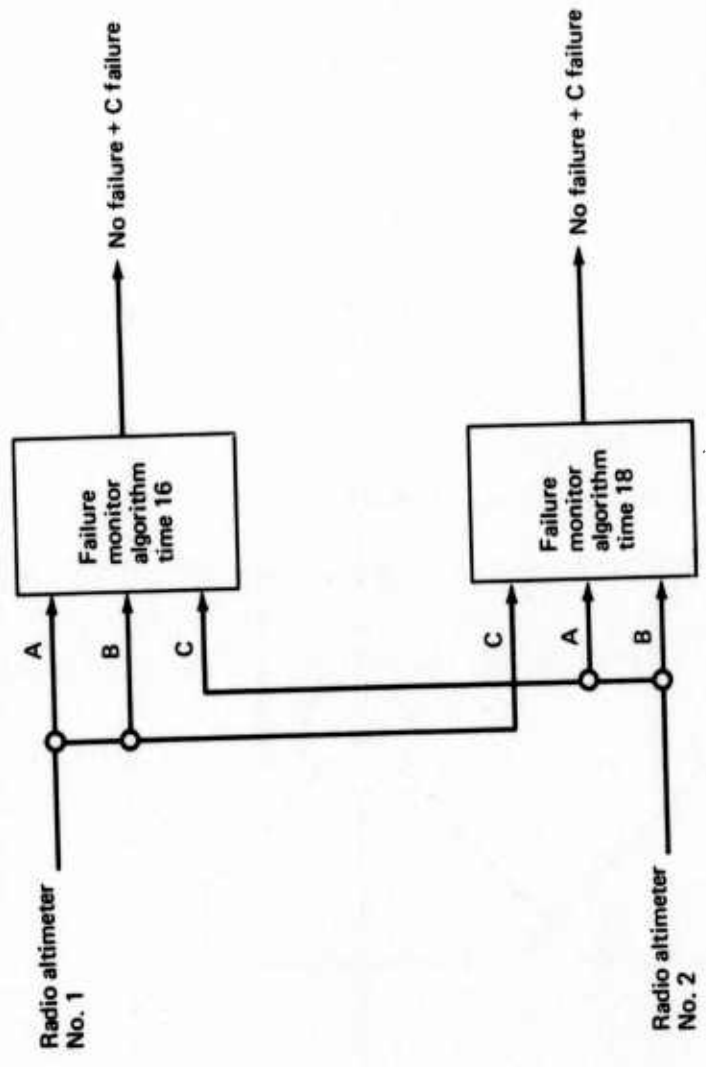


FIGURE 7-12.—RADIO ALTIMETER MONITOR

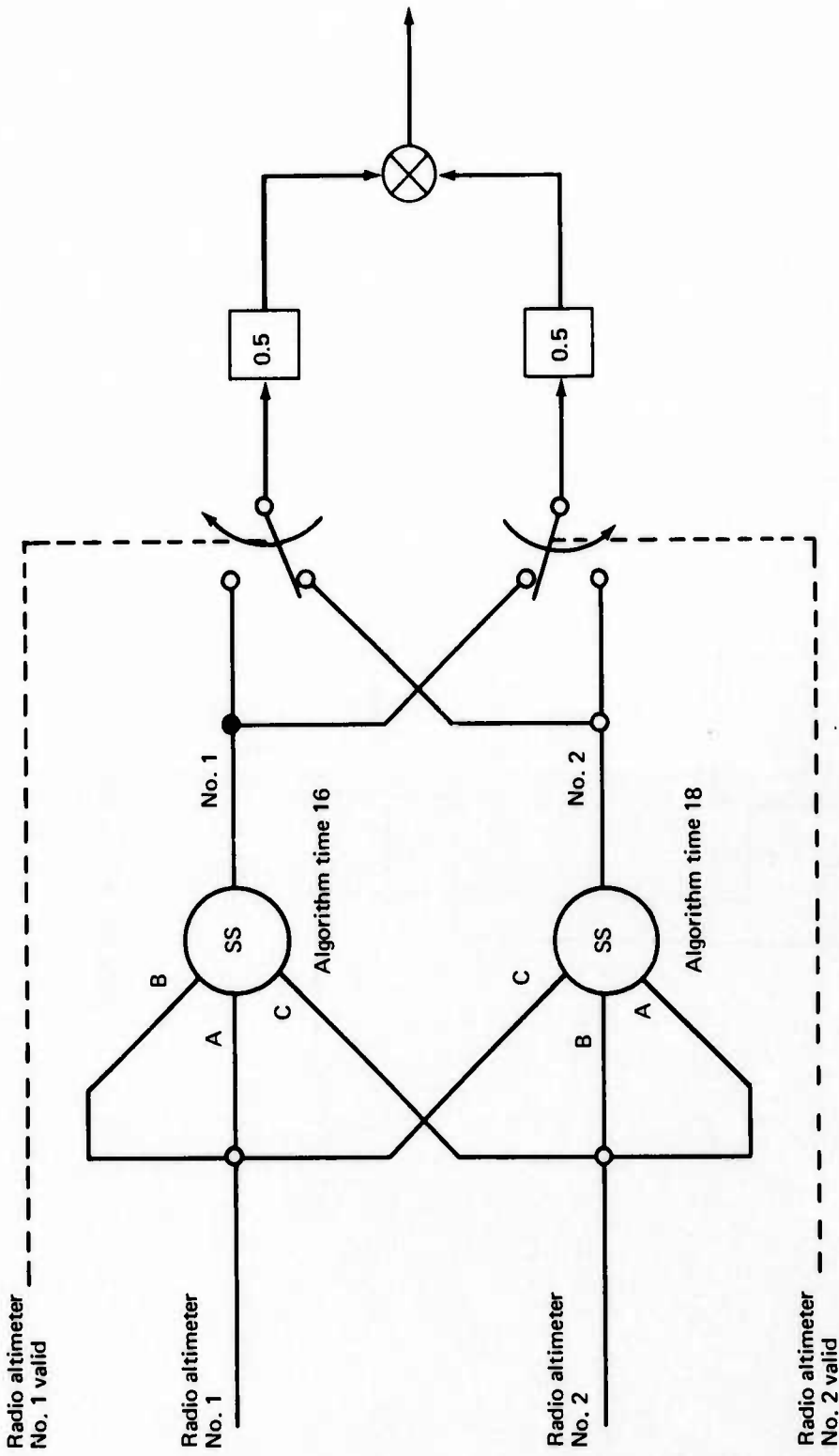


FIGURE 7-13.—RADIO ALTIMETER SIGNAL SELECTION

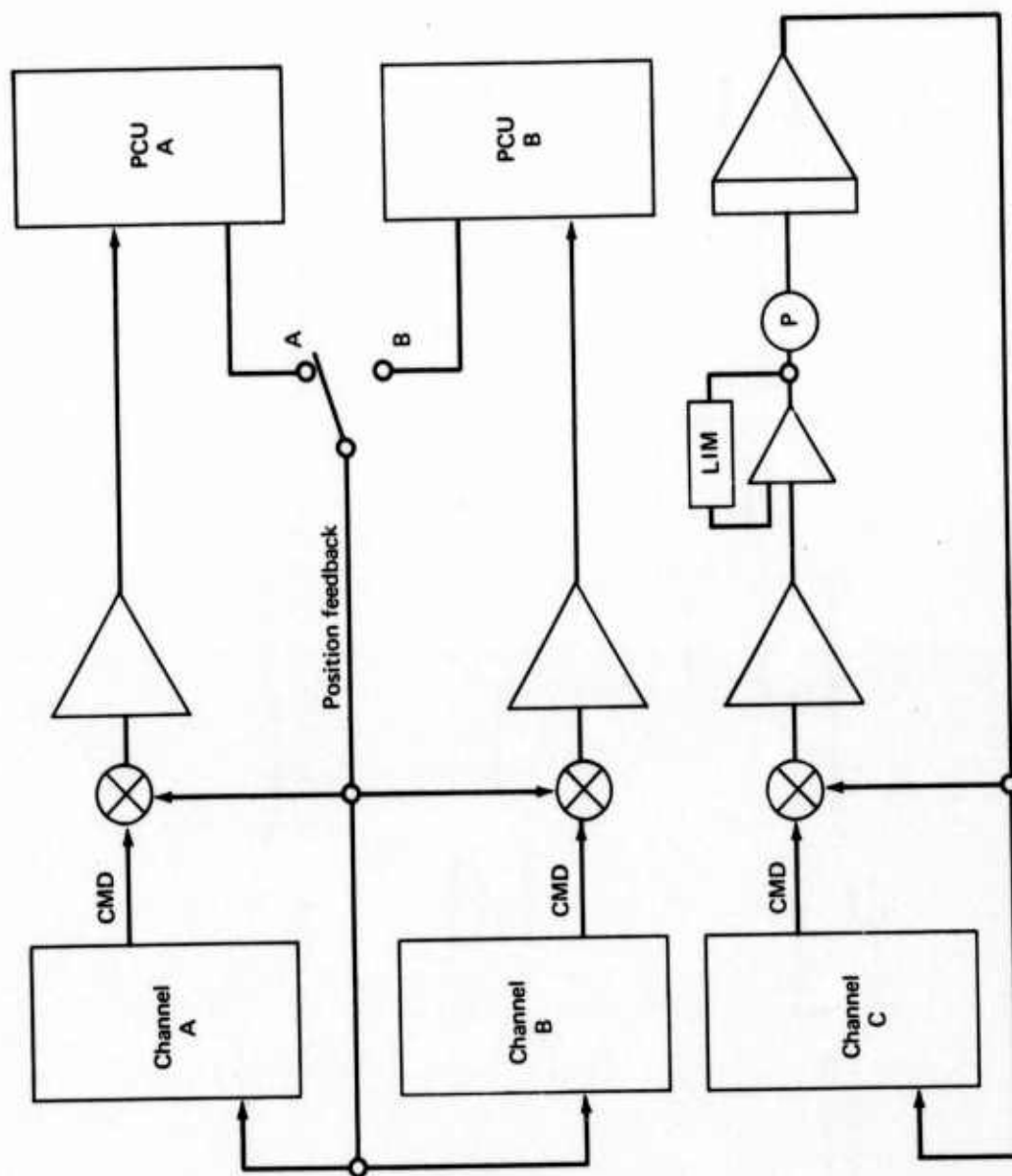


FIGURE 7-14.—SERVO MONITOR SCHEMATIC

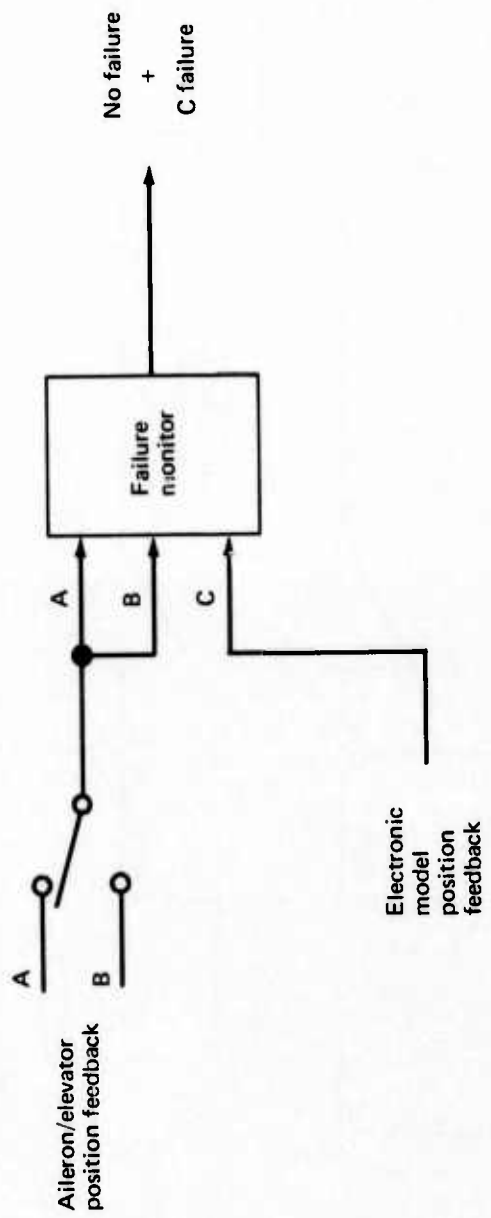


FIGURE 7-15.—AILERON/ELEVATOR SERVO MONITOR FAILURE DETECTION SCHEME

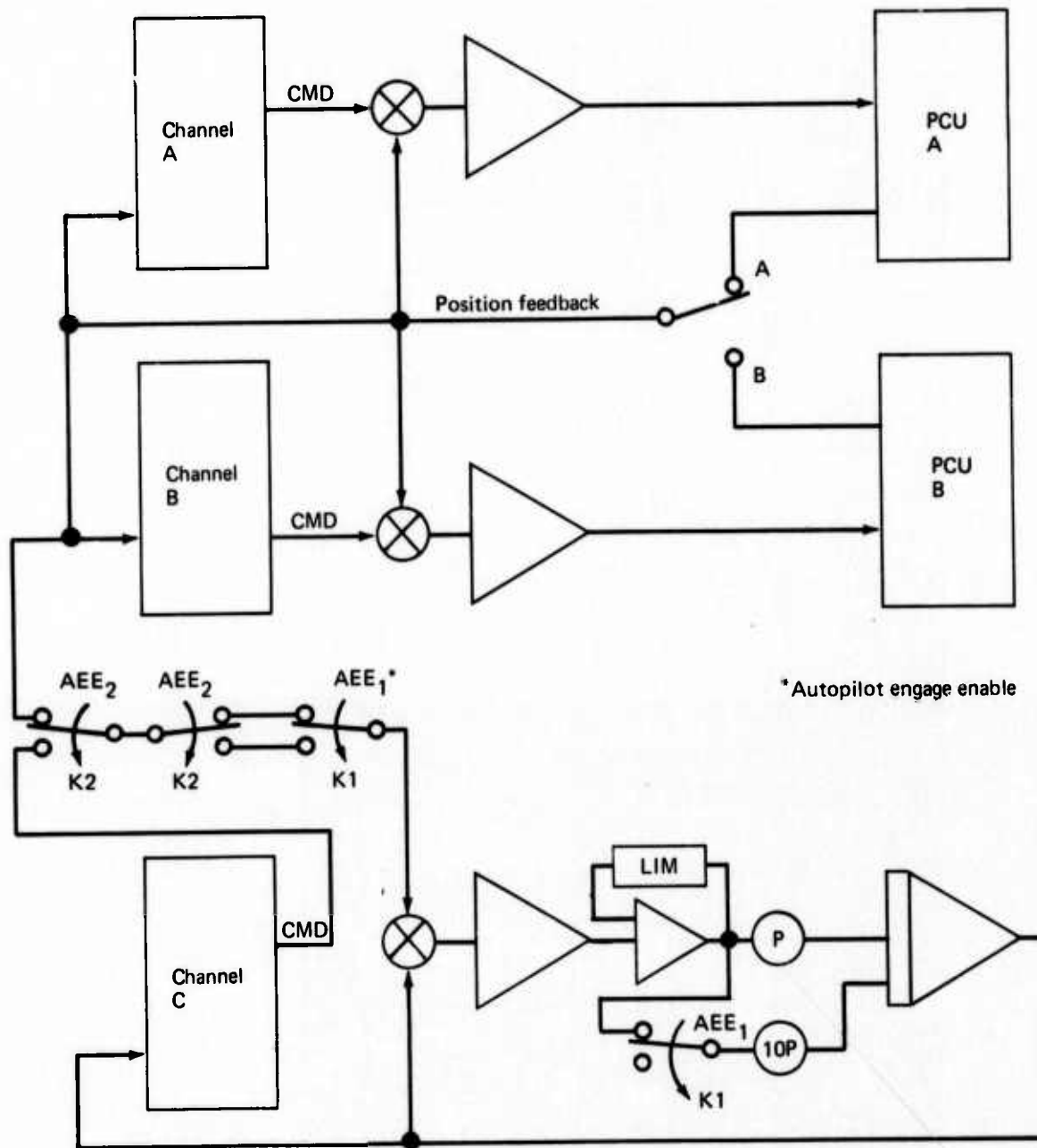


FIGURE 7-16.—SERVO MONITOR SCHEMATIC

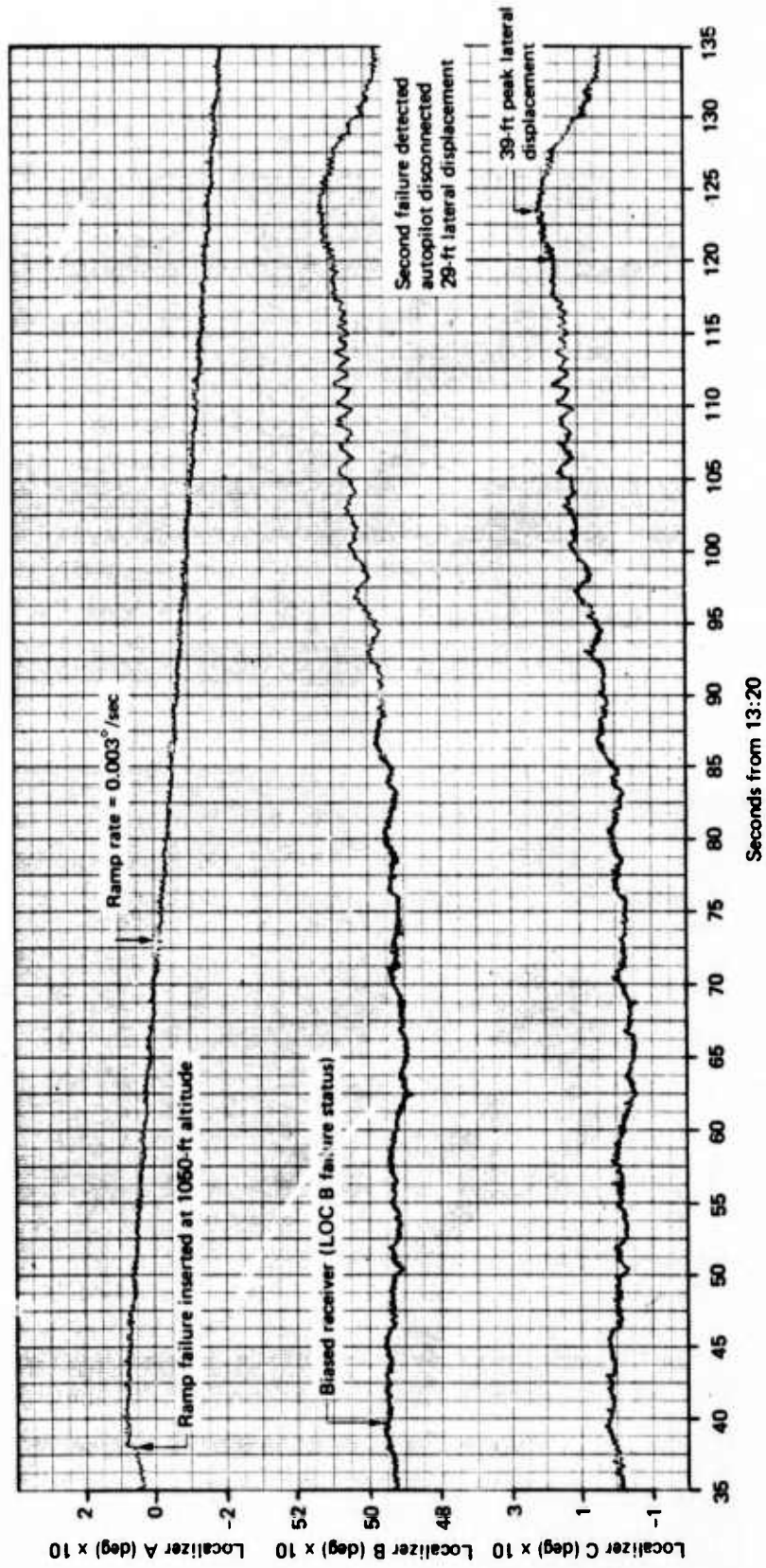


FIGURE 7-17.—LOCALIZER RAMP FAILURE

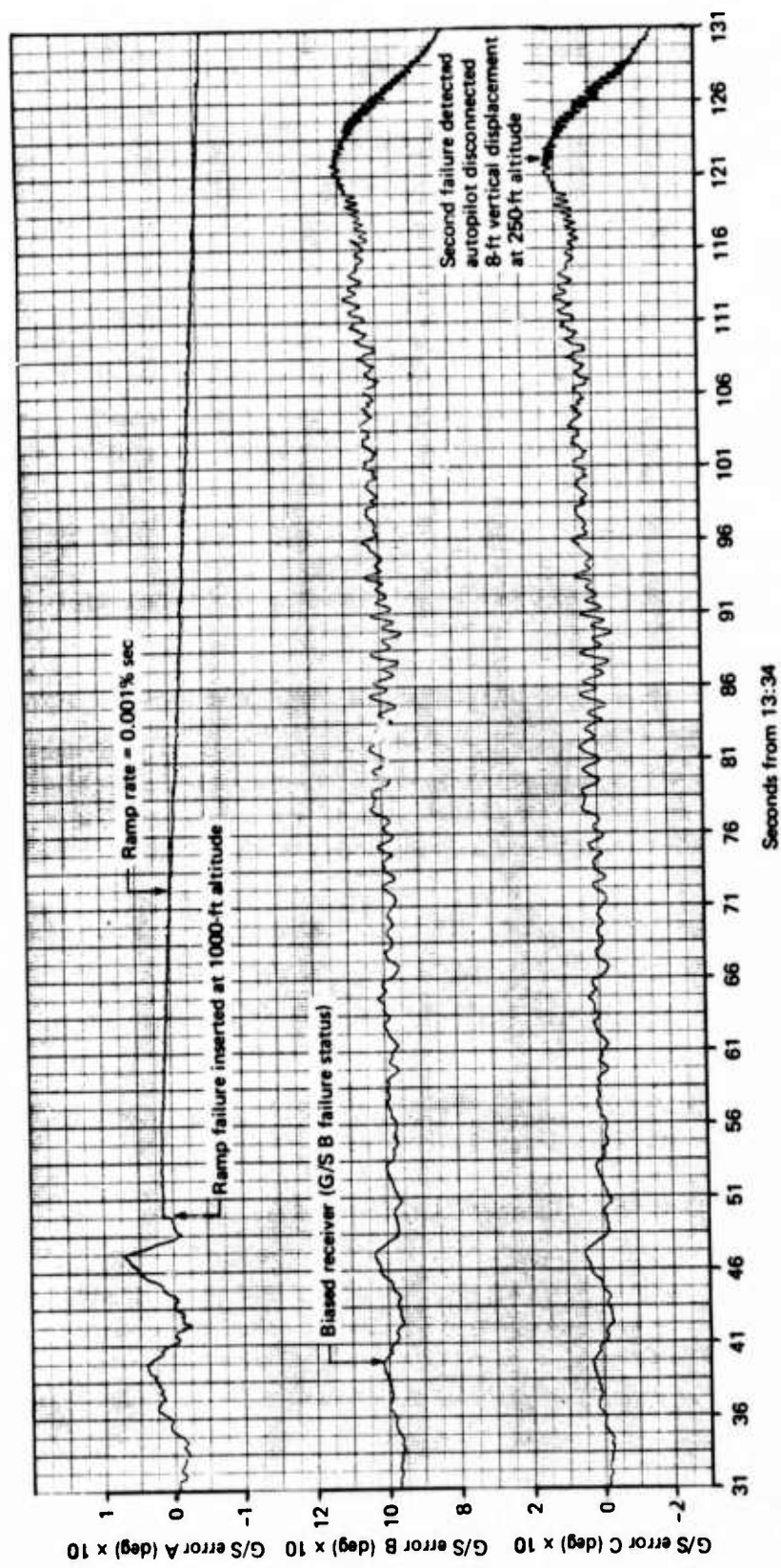
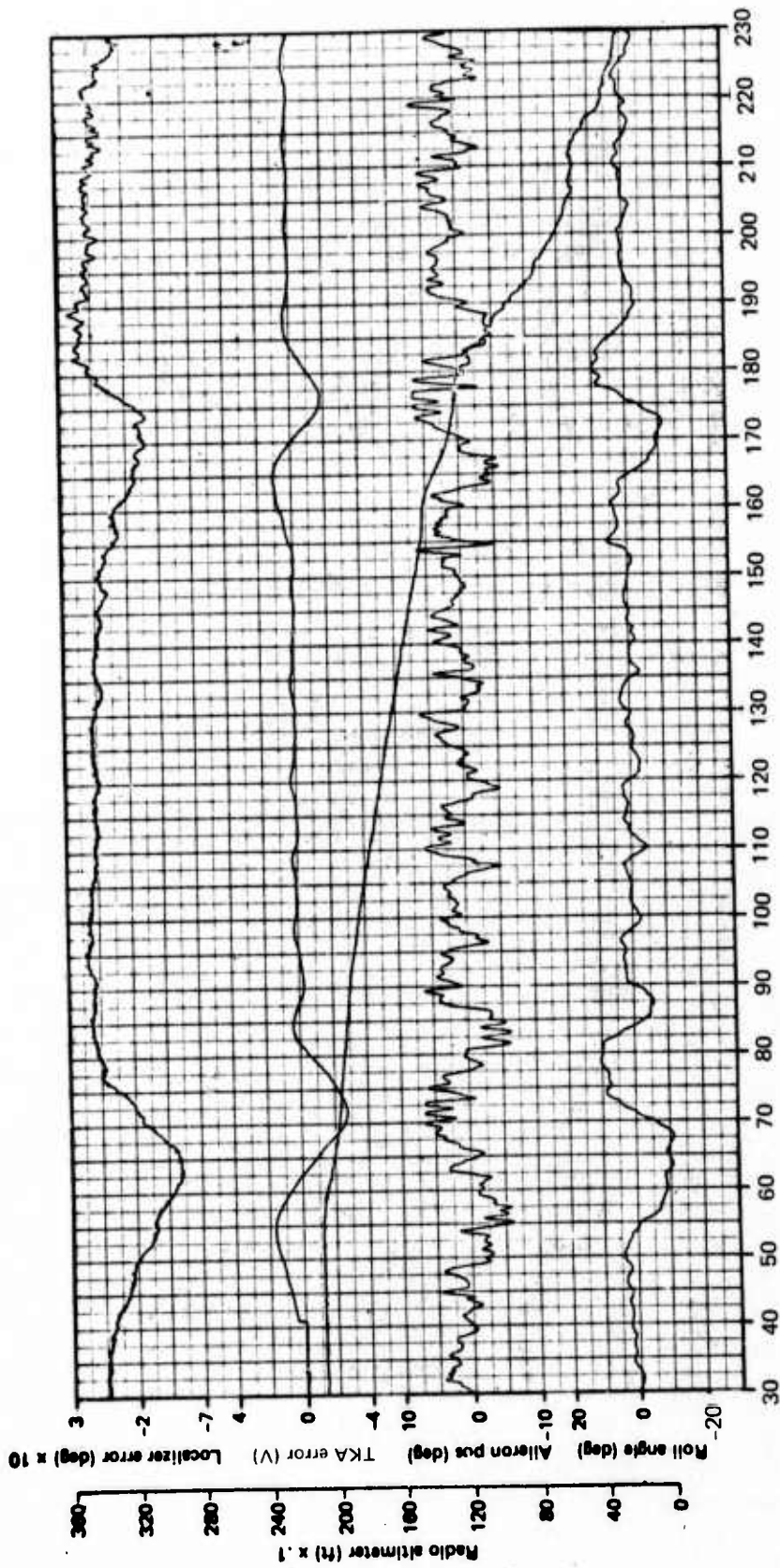


FIGURE 7-18.—GLIDE SLOPE RAMP FAILURE



Seconds from 12:52

FIGURE 7-19.—ROLL AUTOLAND LOCALIZER FLY-OFF MANEUVER

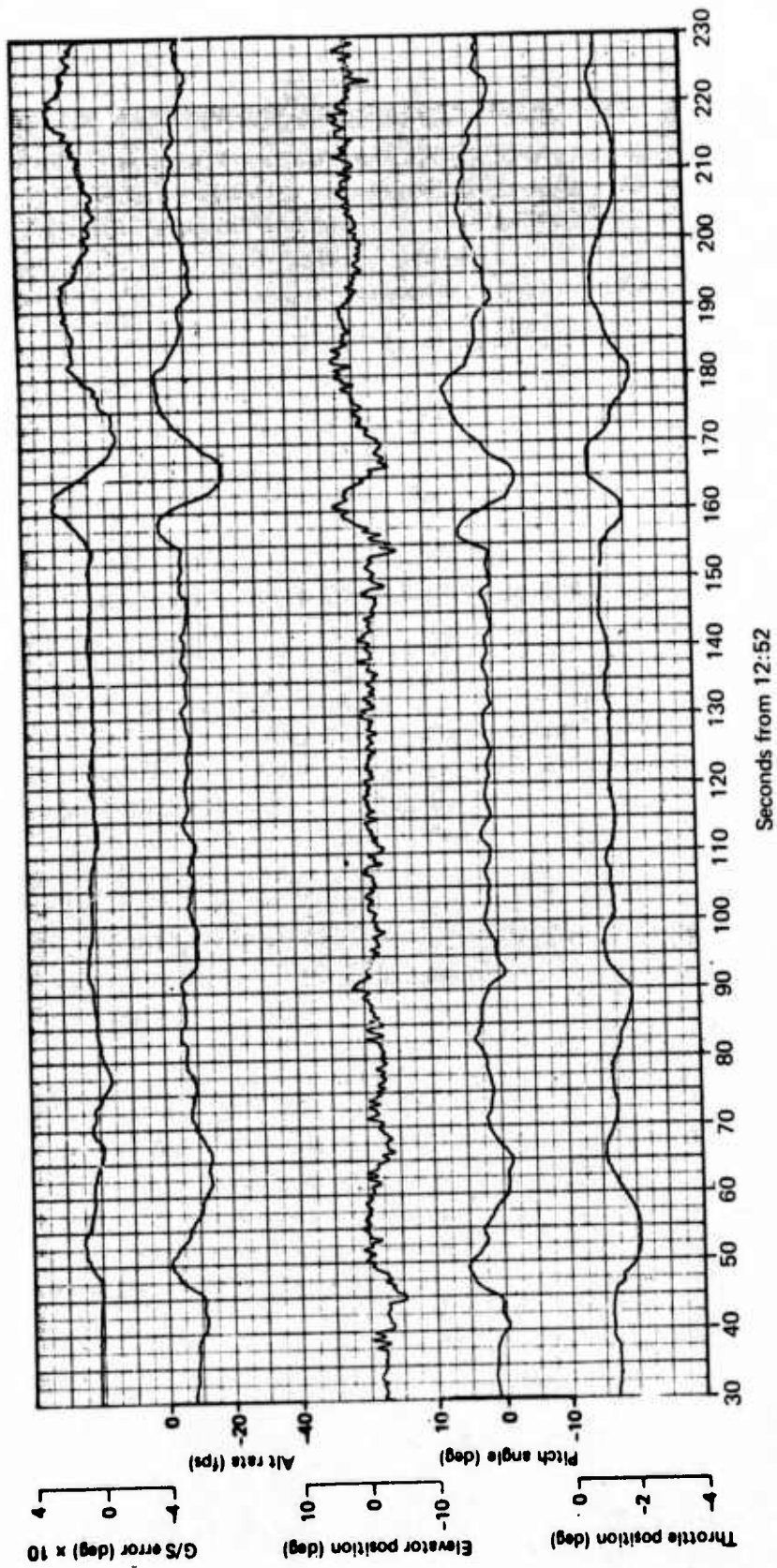


FIGURE 7-20.—PITCH AUTOLAND GLIDE SLOPE FLY-OFF MANEUVER

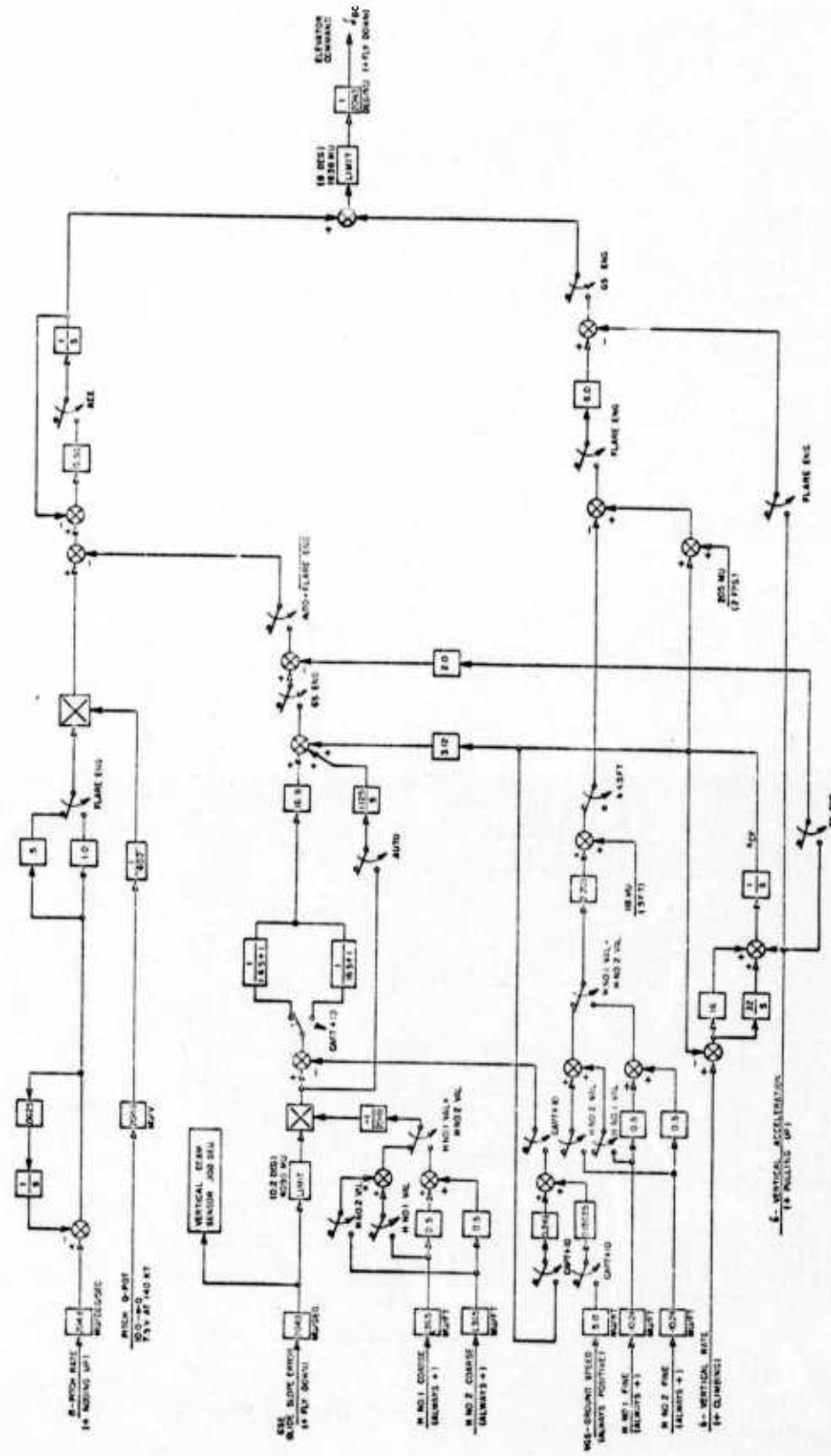


FIGURE 7-21.—737 AGCS PITCH AXIS AUTOLAND

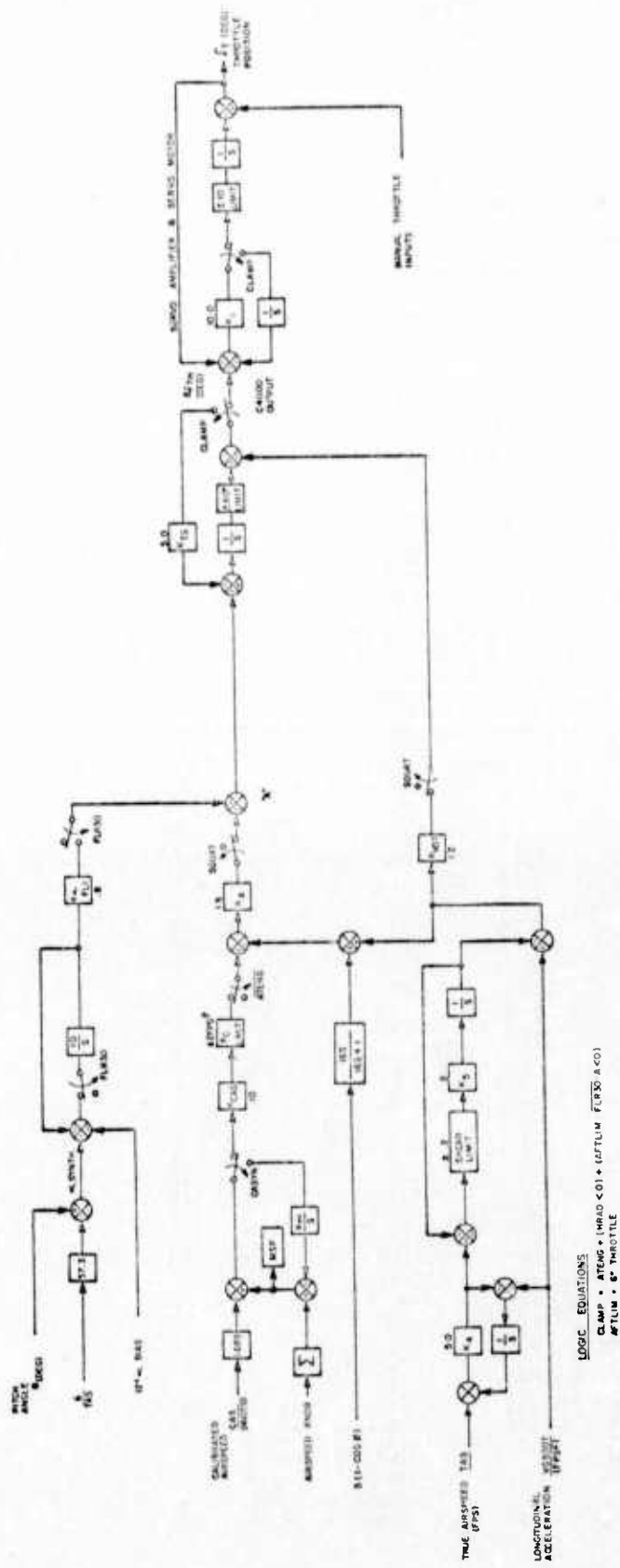


FIGURE 7-23. -737 AGCS AIRSPEED CONTROL LAW

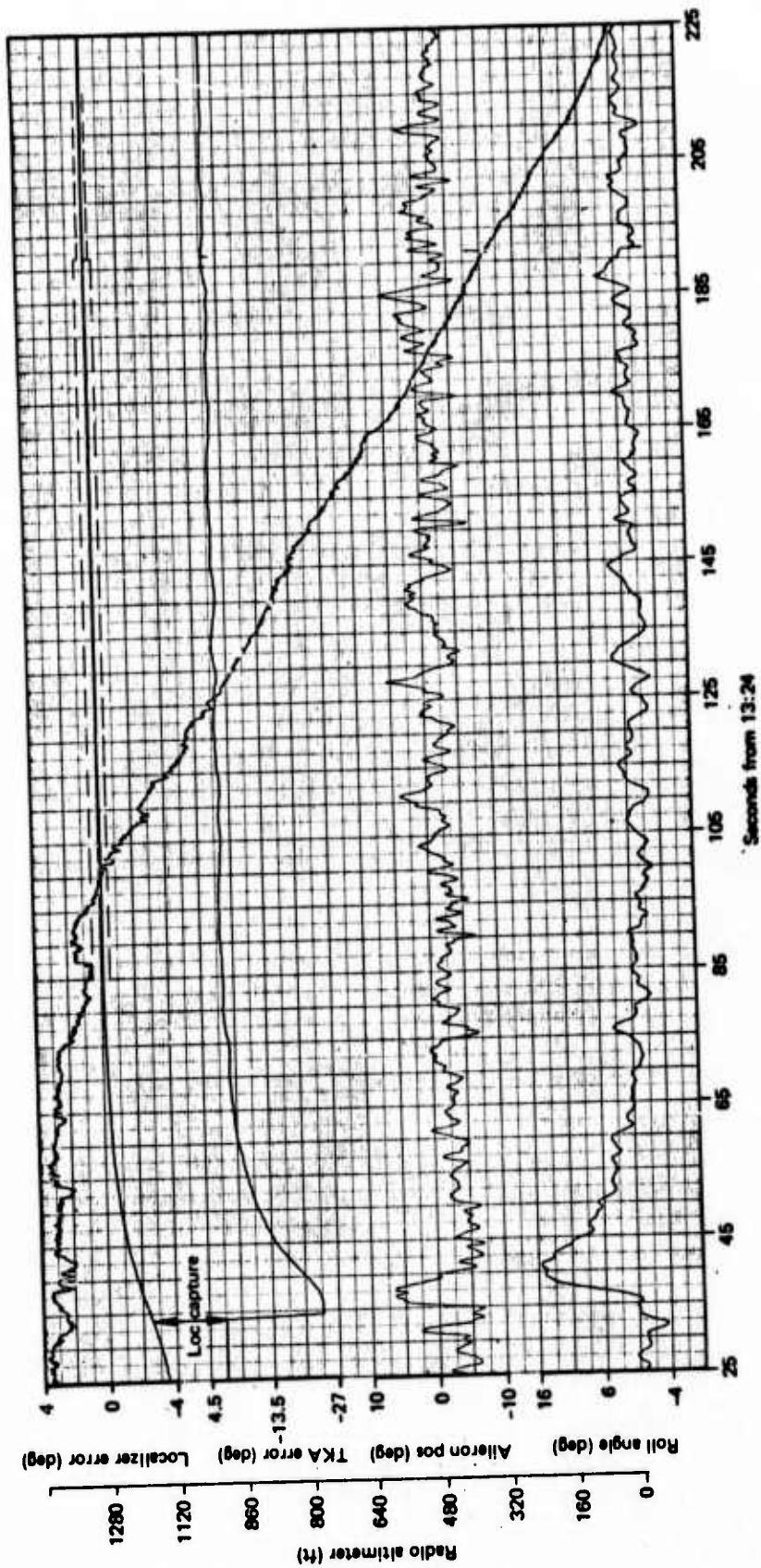


FIGURE 7-24.—ROLL AUTOLAND APPROACH

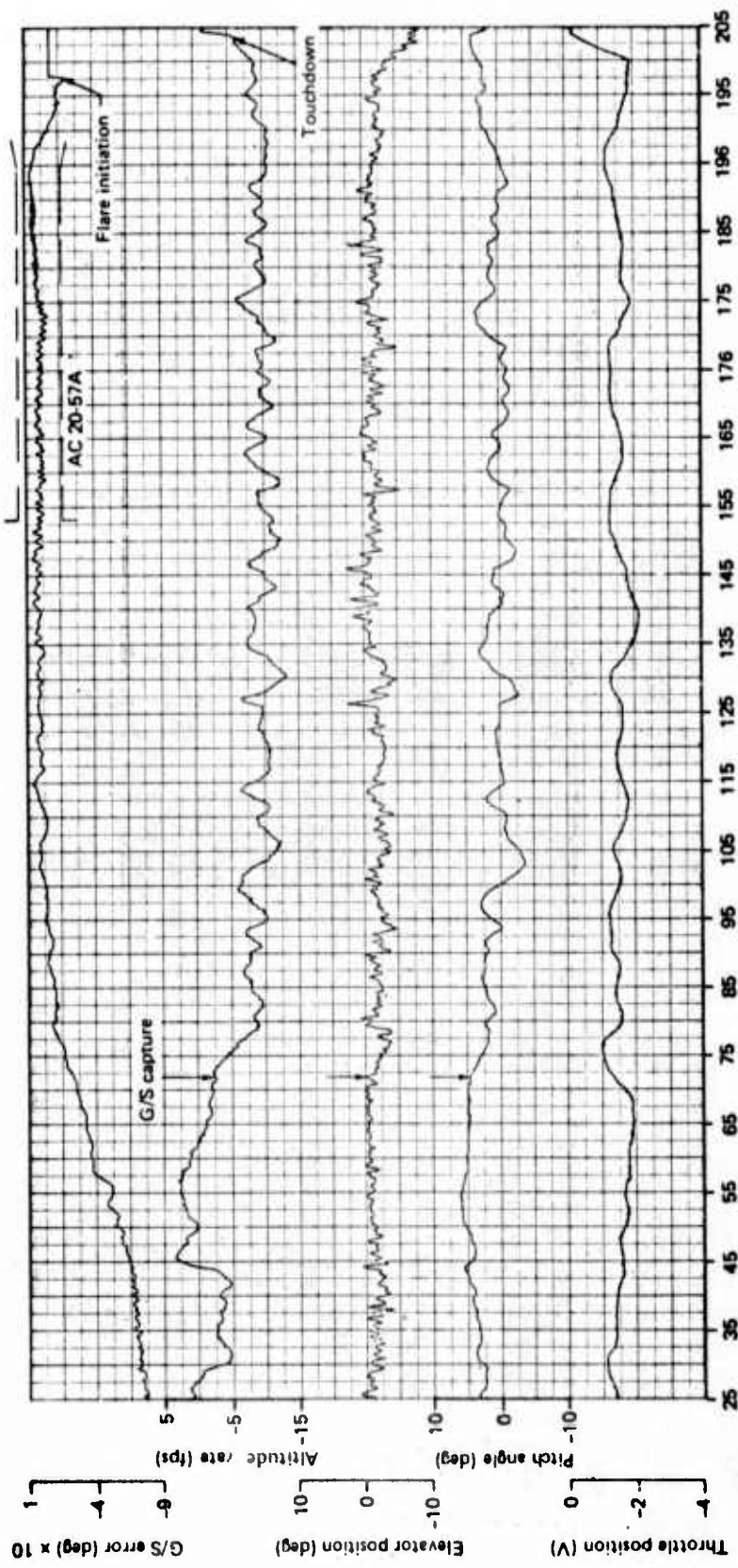


FIGURE 7-25.—PITCH AUTOLAND APPROACH