

NASA CR-183465

FDD/554-89/002
CSC/TR-89/6075



Gamma Ray Observatory (GRO)

Dynamics Simulator

Requirements and Mathematical Specifications

Revision 1

March 1990

(NASA-CR-183465) GAMMA RAY OBSERVATORY
(GRO) DYNAMICS SIMULATOR REQUIREMENTS AND
MATHEMATICAL SPECIFICATIONS, REVISION 1
(Computer Sciences Corp.) 443 p CSCL 145

N90-23447

Unclass

63/14 0267611

NASA

National Aeronautics and
Space Administration

Goddard Space Flight Center
Flight Dynamics Division / Code 550
Greenbelt, Maryland 20771

1. 2. 3. 4.

1. 2.

FDD/554-89/002
CSC/TR-89/6075

GAMMA RAY OBSERVATORY (GRO)
DYNAMICS SIMULATOR
REQUIREMENTS AND MATHEMATICAL SPECIFICATIONS
REVISION 1

March 1990

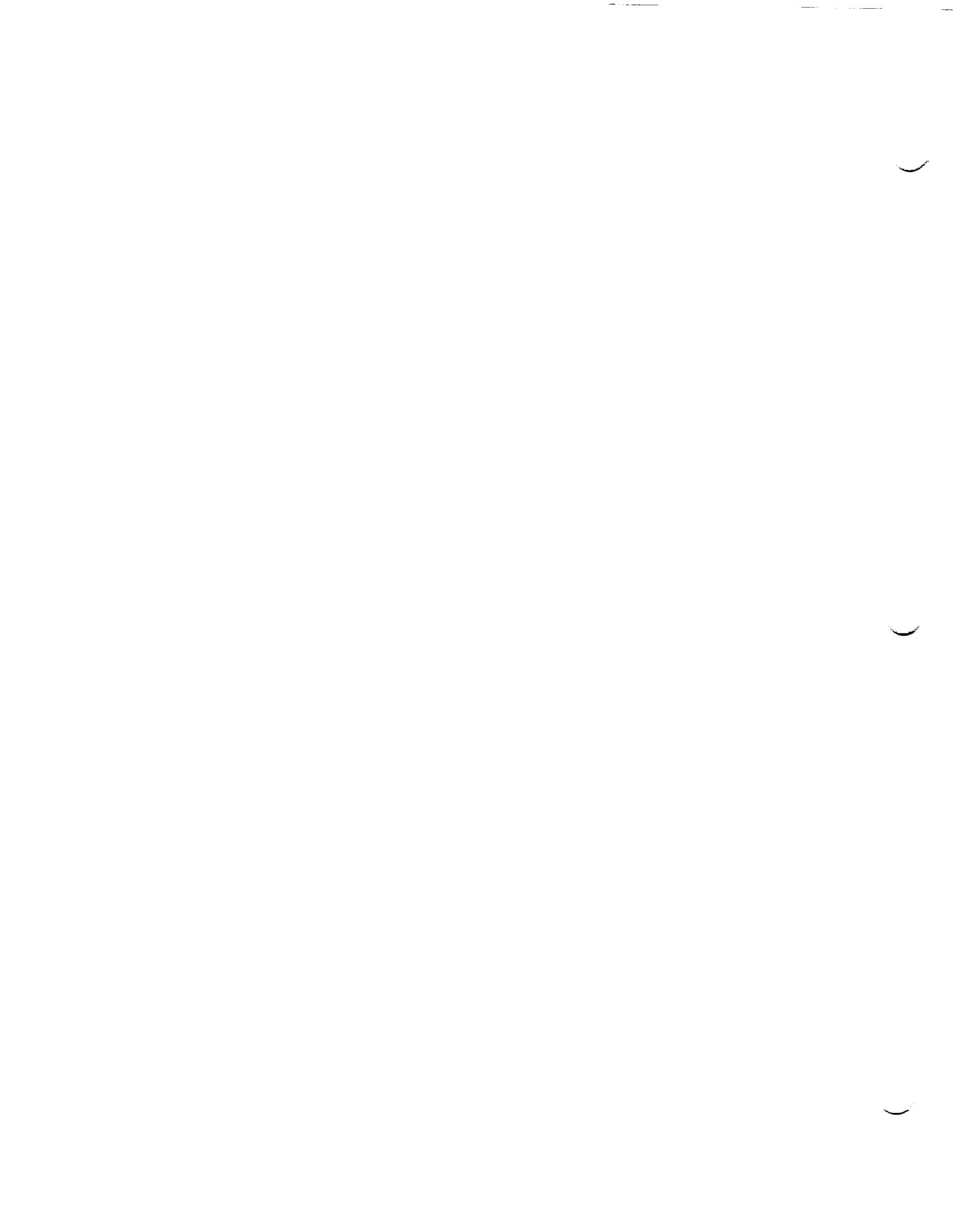
FLIGHT DYNAMICS DIVISION
CODE 550

Developed Under
the Direction of

Through

S. Greatorex
GSFC/Code 554

V. Fernandes
Task 470
Contract NAS 5-31500
Computer Sciences Corporation



**GAMMA RAY OBSERVATORY (GRO)
DYNAMICS SIMULATOR
REQUIREMENTS AND MATHEMATICAL SPECIFICATIONS
REVISION 1**

March 1990

The primary contributor to this document is

R. Harman

GSFC/Code 554

The other contributor is

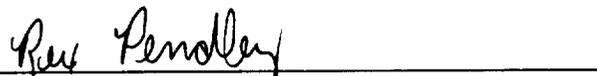
D. Blejer

CSC

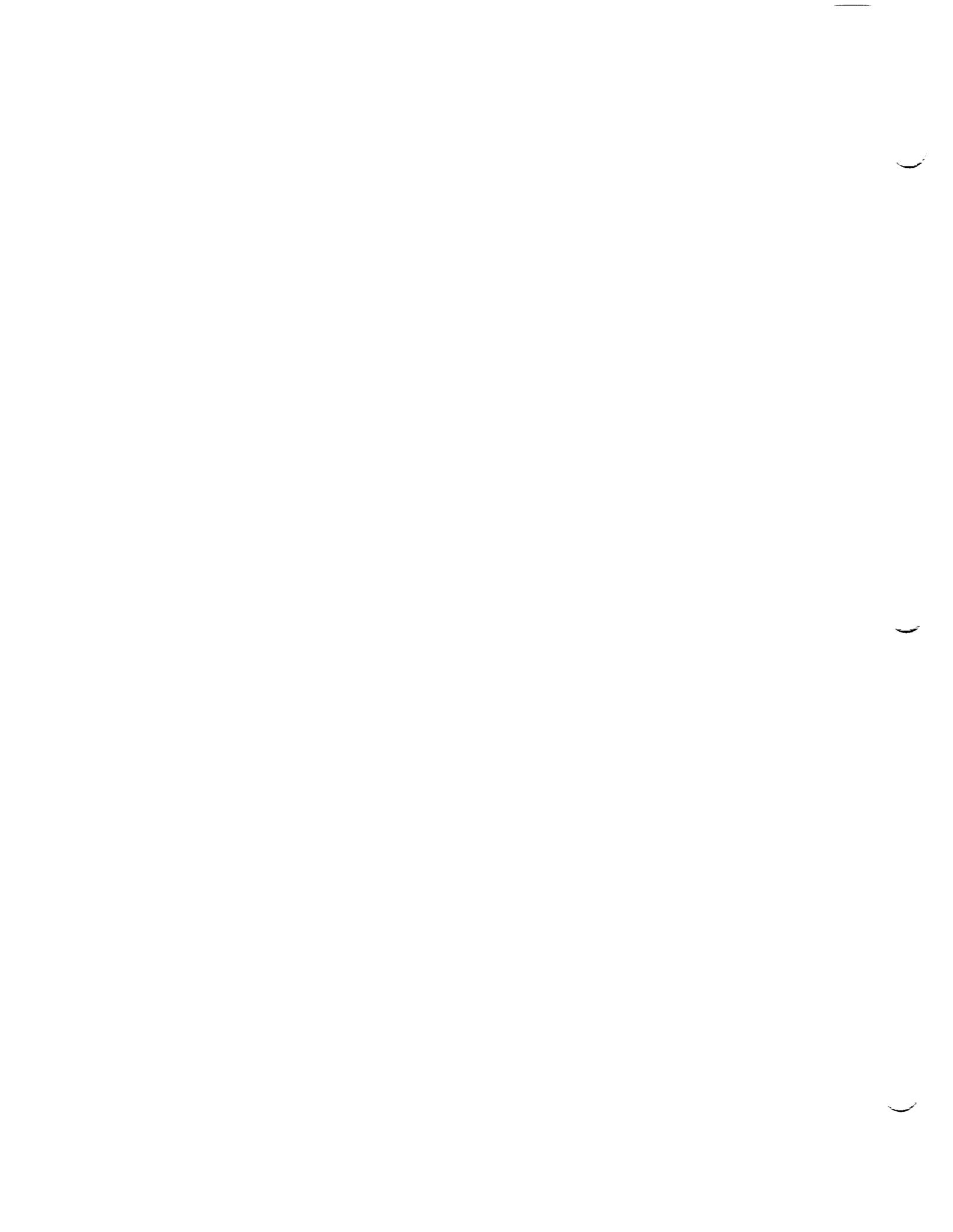
APPROVED FOR PUBLICATION BY:



S. Greatorax, ATR
Flight Dynamics Analysis
Branch

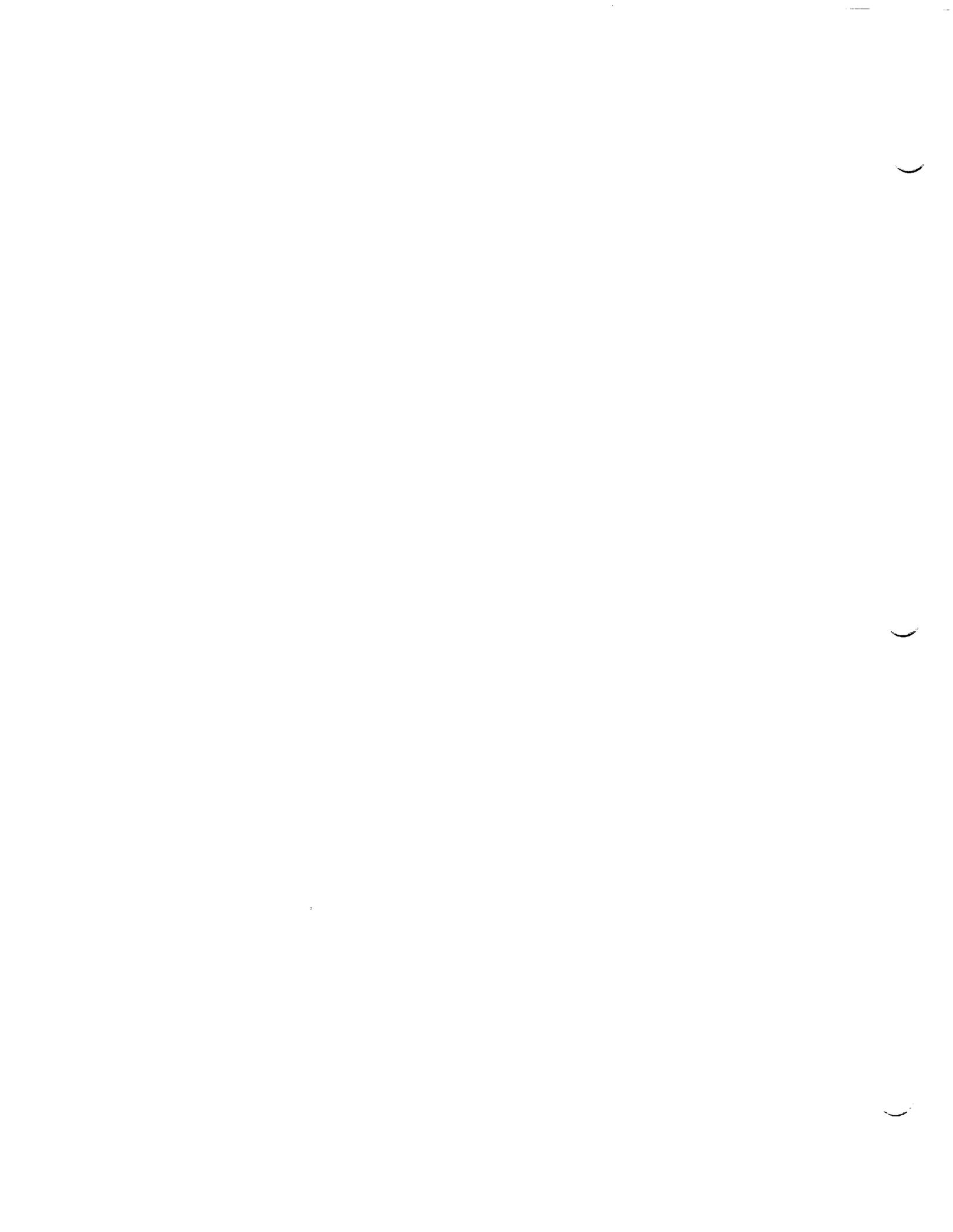


R. Pendley, Manager
Launch Support Analysis
Department



PREFACE

This document was originally issued as CSC/SD-85/6016 in April 1985. Sections 5.2 and 6 contain the new material necessitating the revision.



ABSTRACT

The requirements and mathematical specifications for the Gamma Ray Observatory (GRO) Dynamics Simulator are presented. The complete simulator system, which consists of the Profile Subsystem, Simulation Control and Input/Output Subsystem, Truth Model Subsystem, Onboard Computer Model Subsystem, and Postprocessor, is described. The simulator will be used to evaluate and test the attitude determination and control models to be used on board GRO under conditions that simulate the expected in-flight environment.

1

2

3

TABLE OF CONTENTS

<u>Section 1 - Introduction</u>	1-1
1.1 Mission Overview	1-1
1.1.1 Mission Objectives.	1-2
1.1.2 Spacecraft Configuration.	1-2
1.2 Attitude Control and Determination Subsystem	1-6
1.2.1 Attitude Hardware	1-6
1.2.1.1 FHST	1-9
1.2.1.2 IRU.	1-13
1.2.1.3 TAM.	1-15
1.2.1.4 FSS.	1-16
1.2.1.5 CSS.	1-17
1.2.1.6 MT	1-17
1.2.1.7 RW	1-18
1.2.1.8 Thrusters.	1-19
1.2.2 Modes of Operation.	1-20
1.2.2.1 Standby Mode	1-20
1.2.2.2 Normal Pointing Mode	1-23
1.2.2.3 Normal Maneuver Mode	1-23
1.2.2.4 Sun-Referenced Pointing Mode	1-24
1.2.2.5 Safe-Hold Mode	1-25
1.2.2.6 Thruster Maneuver Mode	1-25
1.2.2.7 Velocity Control Mode.	1-26
1.2.2.8 Thruster Command Mode.	1-26
1.2.2.9 Backup Orbit Maintenance Mode.	1-27
1.3 Dynamics Simulator Overview.	1-27
<u>Section 2 - Profile Subsystem</u>	2-1
2.1 Requirements	2-1
2.1.1 Processing Requirements	2-1
2.1.2 Input Requirements.	2-2
2.1.3 Output Requirements	2-3
2.2 Mathematical Specifications.	2-5
2.2.1 Initialization of Spacecraft Attitude	2-5
2.2.2 Ephemerides and Aberration.	2-7

TABLE OF CONTENTS (Cont'd)

Section 2 (Cont'd)

2.2.2.1	Spacecraft, Solar, and Lunar Ephemerides	2-7
2.2.2.2	Star Ephemeris.	2-8
2.2.2.3	Aberration.	2-8
2.2.3	Magnetic Field.	2-9
2.2.4	Angular Momentum of Internal Motion Computation	2-9
2.2.4.1	Solar Arrays and High-Gain Antenna Angular Rates.	2-10
2.2.4.2	Center of Mass	2-11
2.2.4.3	Moment of Inertia.	2-11
2.2.4.4	Angular Momentum of Internal Motion	2-12
2.2.5	External Torque Computations.	2-13
2.2.5.1	Aerodynamic Torque Model	2-13
2.2.5.2	Solar Pressure Torque Model.	2-15
2.2.5.3	Gravity-Gradient Torque.	2-17
2.2.5.4	Residual Dipole Magnetic Torque	2-18
2.2.5.5	Resultant External Torque.	2-18
2.2.6	Maneuver Simulation	2-18
2.2.7	High-Gain Antenna	2-21

Section 3 - Simulation Control and I/O Subsystem. 3-1

3.1	Requirements	3-1
3.1.1	Processing Requirements	3-1
3.1.2	Input Requirements.	3-2
3.1.3	Output Requirements	3-3
3.2	Mathematical Specifications.	3-4

Section 4 - Truth Model Subsystem 4-1

4.1	Requirements	4-1
4.1.1	Processing Requirements	4-1
4.1.2	Input Requirements.	4-3
4.1.3	Output Requirements	4-4

TABLE OF CONTENTS (Cont'd)

Section 4 (Cont'd)

4.2	Mathematical Specifications	4-5
4.2.1	Dynamics Integrator	4-5
4.2.1.1	Adams-Moulton-Bashforth Integrator	4-5
4.2.1.2	Derivation of AMB Equations.	4-6
4.2.1.3	Changing Interval Size	4-12
4.2.1.4	Partial Step Equation.	4-13
4.2.1.5	Stepsize Considerations.	4-14
4.2.1.6	Equations of Motion.	4-17
4.2.2	Simulator Interpolation	4-20
4.2.3	Internal Motion	4-25
4.2.4	Sensor Models	4-25
4.2.4.1	IRU Model.	4-27
4.2.4.2	FSS Model.	4-28
4.2.4.3	CSS Model.	4-33
4.2.4.4	FHST Model	4-36
4.2.4.5	Wheel Tachometer Model	4-42
4.2.4.6	TAM Model.	4-43
4.2.5	Actuator Models	4-48
4.2.5.1	Reaction Wheel Assembly.	4-48
4.2.5.2	Magnetic Torquer Assembly.	4-51
4.2.5.3	Attitude Control Thrusters	4-56
4.2.5.4	Orbit Adjust Thrusters	4-59

Section 5 - Onboard Computer Model Subsystem

5.1	Requirements	5-1
5.1.1	Processing Requirements	5-1
5.1.2	Input Requirements.	5-2
5.1.3	Output Requirements	5-2
5.2	Mathematical Specifications	5-3
5.2.1	Sensor Processing	5-3
5.2.1.1	Inertial Reference Unit Data Processing	5-3
5.2.1.2	Fine Sun Sensor Data Proc- essing	5-13

TABLE OF CONTENTS (Cont'd)

Section 5 (Cont'd)

5.2.1.3	Fixed-Head Star Tracker Data Processing	5-15
5.2.1.4	Wheel Tachometer Data Processing	5-24
5.2.1.5	Magnetometer Data Processing	5-26
5.2.1.6	Star Data Averaging Function	5-32
5.2.2	Observatory Support	5-36
5.2.2.1	Ephemeris Computation Function	5-36
5.2.2.2	Spacecraft Orientation Function	5-37
5.2.2.3	Earth Occultation Function	5-40
5.2.3	Attitude Determination.	5-42
5.2.3.1	System Overview.	5-42
5.2.3.2	Kinematic Integration.	5-48
5.2.3.3	Attitude Error Computation Function	5-53
5.2.3.4	Attitude Estimation Function	5-57
5.2.4	Attitude Control.	5-106
5.2.4.1	High-Level Thrust Control Law Function	5-106
5.2.4.2	Low-Level Thrust Control Law Function	5-113
5.2.4.3	Thruster Command Processing Function	5-122
5.2.4.4	Magnetic Control Law Function	5-127
5.2.4.5	Wheel Control and Distribution Function.	5-133
5.2.4.6	Mode Control Function.	5-143
5.2.4.7	Safe Contingency Mode Function	5-166
5.2.5	Antenna Pointing Command Processing	5-181

TABLE OF CONTENTS (Cont'd)

<u>Section 6 - Attitude Control Electronics.</u>	6-1
6.1 Sensor Data Processing	6-1
6.1.1 Gyro Data Processing	6-1
6.1.1.1 Input/Output	6-1
6.1.1.2 Algorithm	6-2
6.1.2 Coarse Sun Sensor Data Processing	6-9
6.1.2.1 Input/Output	6-9
6.1.2.2 Algorithm	6-9
6.1.3 Fine Sun Sensor Data Processing	6-11
6.1.3.1 Input/Output	6-11
6.1.3.2 Algorithm	6-11
6.1.4 Reaction Wheel Momentum Computation	6-13
6.1.4.1 Input/Output	6-13
6.1.4.2 Algorithm	6-13
6.1.5 Three-Axis Magnetometer Data Processing	6-14
6.1.5.1 Input/Output	6-14
6.1.5.2 Algorithm	6-15
6.2 Attitude Error Computations	6-16
6.2.1 Input/Output	6-16
6.2.2 Algorithm	6-17
6.3 Actuator Command Generation	6-22
6.3.1 Thruster Control Function	6-22
6.3.1.1 Input/Output	6-22
6.3.1.2 Algorithm	6-23
6.3.2 Reaction Wheel Control Function	6-29
6.3.2.1 Input/Output	6-29
6.3.2.2 Algorithm	6-30

TABLE OF CONTENTS (Cont'd)

Section 6 (Cont'd)

6.3.3	Reaction Wheel Command Processing	6-34
6.3.3.1	Input/Output	6-34
6.3.3.2	Algorithm.	6-35
6.3.4	Magnetic Control Law.	6-39
6.3.4.1	Input/Output	6-39
6.3.4.2	Algorithm.	6-40
6.4	ACE Control.	6-44
6.4.1	OBC/ACE Transitions	6-44
6.4.2	Computational Scheduling.	6-45
6.4.2.1	Input/Output	6-45
6.4.2.2	Main Loop.	6-50
6.5	ACE/CPE Modes.	6-53
6.5.1	Sun Reference Pointing Mode Control Law	6-53
6.5.2	Roll Scan Mode Control Law.	6-54
6.5.3	Reaction Wheel Attitude Hold Mode	6-57
6.5.4	Safe Hold Mode.	6-61
6.5.5	Thruster Attitude Hold Mode	6-64
6.5.6	Contingency Orbit Maintenance Mode.	6-64
6.5.7	Thruster Command Mode	6-70
6.6	Momentum Management.	6-70

Appendix A - Coordinate Systems

Appendix B - GRO Surface Area Model

Appendix C - Utility and Initialization Subroutines

Appendix D - OBC Ground Command Description

Glossary

References

LIST OF ILLUSTRATIONS

Figure

1-1	View of GRO Spacecraft From +Z Axis.	1-3
1-2	View of GRO Spacecraft From -Z Axis.	1-4
1-3	GRO Subsystems	1-7
1-4	ACADS Configuration.	1-8
1-5	ACADS Components (View From +X Axis)	1-10
1-6	ACADS Components (View From -X Axis)	1-11
1-7	ACAD Control Mode Transitions.	1-22
1-8	GRO Dynamics Simulator Data Flow	1-28
4-1	General Form of an n-Point Lagrangian Interpolating Polynomial	4-22
4-2	Angular Momentum Versus Time for a Scanning Instrument.	4-26
4-3	Spacecraft Shadow Testing Geometry	4-30
4-4	Sun Sensor FOV Angle Definitions in the FSS Reference Frame.	4-32
4-5	Coarse Sun Sensor Assembly	4-34
4-6	Camera Focal Plane	4-37
4-7	Star Tracker Angular Coordinates	4-38
4-8	Accept All Star Unit Vectors Contained Within the Cone of Angle θ	4-40
4-9	Torquer Bar in the Reference Frame	4-45
4-10	Duty Cycle as a Function of the Commanded Input Voltage.	4-50
4-11	Wheel Torque Versus Wheel Speed.	4-52
4-12	MMS Model of Hysteresis Loop	4-53
4-13	Thruster Location and Thruster Angle Convention	4-57
5-1	Gyro Data Acquisition Functional Block Diagram	5-4
5-2	Gyro Data Compensation Functional Block Diagram.	5-5
5-3	FSS Coordinate Frame and Sun Angle Compo- nent Definition.	5-14
5-4	Magnetic Data Processing Flow.	5-30
5-5	GRO Normal Pointing Mode Attitude Deter- mination Functional Block Diagram.	5-43
5-6	Attitude Estimation Algorithm Functional Block Diagram.	5-45
5-7	Attitude Estimation Software Function.	5-46
5-8	Attitude Estimation Overview Flow Chart.	5-69
5-9	ATTEST Measurement Processing.	5-82
5-10	Multiple Firing Algorithm.	5-118

LIST OF ILLUSTRATIONS (Cont'd)

Figure

5-11	Thruster Firing Logic	5-120
5-12	ACAD Mode 1: Standby Mode	5-144
5-13	ACAD Mode 2: Normal Pointing Mode	5-145
5-14	ACAD Mode 3: Normal Maneuver Mode	5-146
5-15	ACAD Mode 6: Thruster Maneuver Mode	5-147
5-16	ACAD Mode 7: Velocity Control Mode	5-148
6-1	GRO Reaction Wheel Configuration	6-41
6-2	Magnetic Torquer Location and Orientation	6-43
6-3	SRPM Control Flow	6-55
6-4	SRPM Attitude Error Determination	6-56
6-5	SRPM Reaction Wheel Control	6-57
6-6	RSM Control Flow	6-58
6-7	RSM Attitude Error Determination	6-59
6-8	RSM Reaction Wheel Control	6-60
6-9	RWAHM Control Flow	6-62
6-10	RWAHM Attitude Error Determination	6-63
6-11	RWAHM Reaction Wheel Control	6-63
6-12	SHM Control Flow	6-65
6-13	SHM Attitude Error Determination	6-66
6-14	SHM Thruster Control	6-67
6-15	TAHM Control Flow	6-68
6-16	TAHM Attitude Error Determination	6-69
6-17	TAHM Thruster Control	6-69
6-18	Momentum Management	6-72

LIST OF TABLES

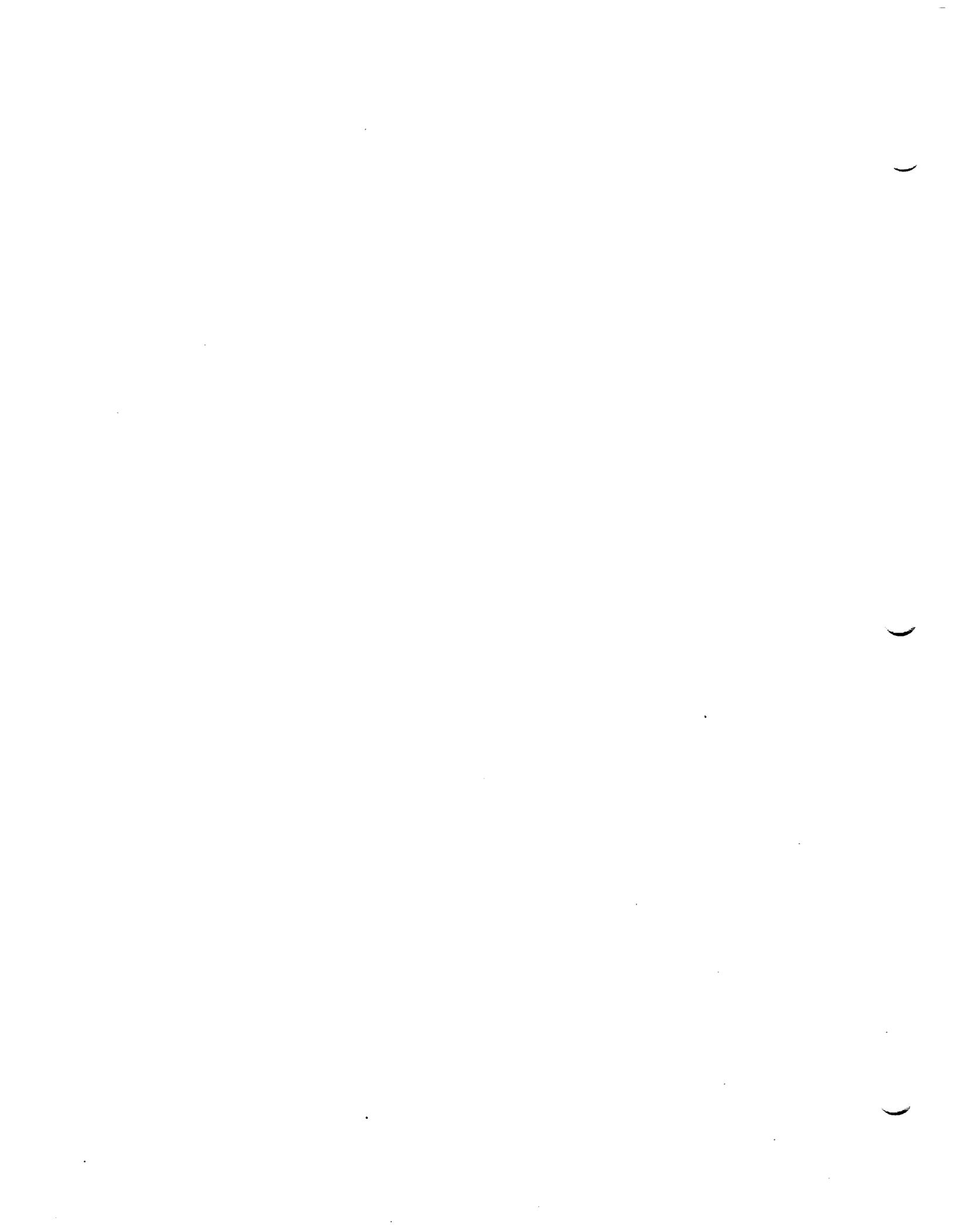
Table

1-1	ACAD Modes and Equipment Usage	1-21
4-1	Thruster Locations	4-60
5-1	Gyro Channel Definitions	5-6
5-2	Magnetometer Data Processing	5-27
5-3	Saved Attitude Estimation Data	5-74
5-4	Flags and Counters Used by ATTEST	5-85
5-5	Functions Executed by ACAD Mode, Frequency	5-149
5-6	ACAD Flags and Constants Set on Mode Initiation	5-150
5-7	ACAD Mode Transition Matrix	5-155
5-8	Commands Issued on Mode Initiation	5-158

LIST OF TABLES (Cont'd)

Table

5-9	Gyro Channel Definitions	5-170
5-10	Gyro Selection for Two-Axis Miscompares.	5-172
5-11	Gyro Selection for Single-Axis Miscompares	5-173
6-1	Thruster Control Gain Definitions.	6-24
6-2	Values for Scaling the Components of TRW_d	6-38
6-3	RAM Variables To Be Initialized During Power-On Initialization.	6-46



SECTION 1 - INTRODUCTION

This document presents the requirements, functional specifications, and mathematical specifications for the Gamma Ray Observatory (GRO) Dynamics Simulator. The GRO Dynamics Simulator will provide an analytical tool for testing and evaluating the attitude determination and control system to be used on board the GRO spacecraft. The system will be tested and evaluated under conditions that simulate the expected in-flight environment as closely as possible.

1.1 MISSION OVERVIEW

GRO will constitute a major advance in gamma ray astronomy by offering the first opportunity for comprehensive observations that cover a wide energy range from 0.1 to 30,000 mega-electronvolts (MeV). Relatively long exposures using extremely large and heavy instruments in orbit above the absorbing atmosphere are required to make these observations.

The GRO spacecraft is tentatively scheduled to be launched by the Space Transportation System (STS) from the Eastern Test Range in June 1990. The operational orbit will be circular, at an altitude between 350 and 450 kilometers (km) and an inclination of 28.5 degrees (deg). GRO will carry a hydrazine propulsion system to achieve a Hohmann orbit transfer from the nominal shuttle orbit at an altitude of 296 km to the mission orbit. Science requirements necessitate an undisturbed inertial orientation for at least 2 weeks for each target.

GRO will have a 2-year science mission lifetime, with two possible 6-month extensions. After completing the science mission, GRO will be returned to Earth during a 3-month

phase. Either it will be retrieved with the STS or its re-entry will be controlled.

1.1.1 MISSION OBJECTIVES

The study of cosmic gamma rays is the primary mission objective of GRO. The observatory will carry the following four scientific instruments for this study:

1. The Oriented Scintillation Spectrometer Experiment (OSSE) will observe line and continuum gamma ray sources in the 0.1- to 10-MeV range and solar gamma rays and neutrons above 10 MeV.
2. The Imaging Compton Telescope (COMPTEL) will perform a sensitive survey of gamma rays in the 1- to 30-MeV range with good angular resolution and low background.
3. The Energetic Gamma Ray Experiment Telescope (EGRET) will search for discrete gamma ray sources in the 20- to 30,000-MeV range. EGRET will measure gamma ray intensity, energy spectrum, position, and time variations.
4. The Burst and Transient Source Experiment (BATSE) will monitor the entire unocculted sky for transient gamma ray events and bursts. BATSE is optimized for gamma rays in the 0.06- to 6-MeV range and will provide burst signals to the other instruments.

1.1.2 SPACECRAFT CONFIGURATION

Views of the GRO spacecraft from the +Z axis and the -Z axis are shown in Figures 1-1 and 1-2, respectively. The physical parameters of interest are given in the following tabulation. Numbers are approximate and are current as of December 1988.

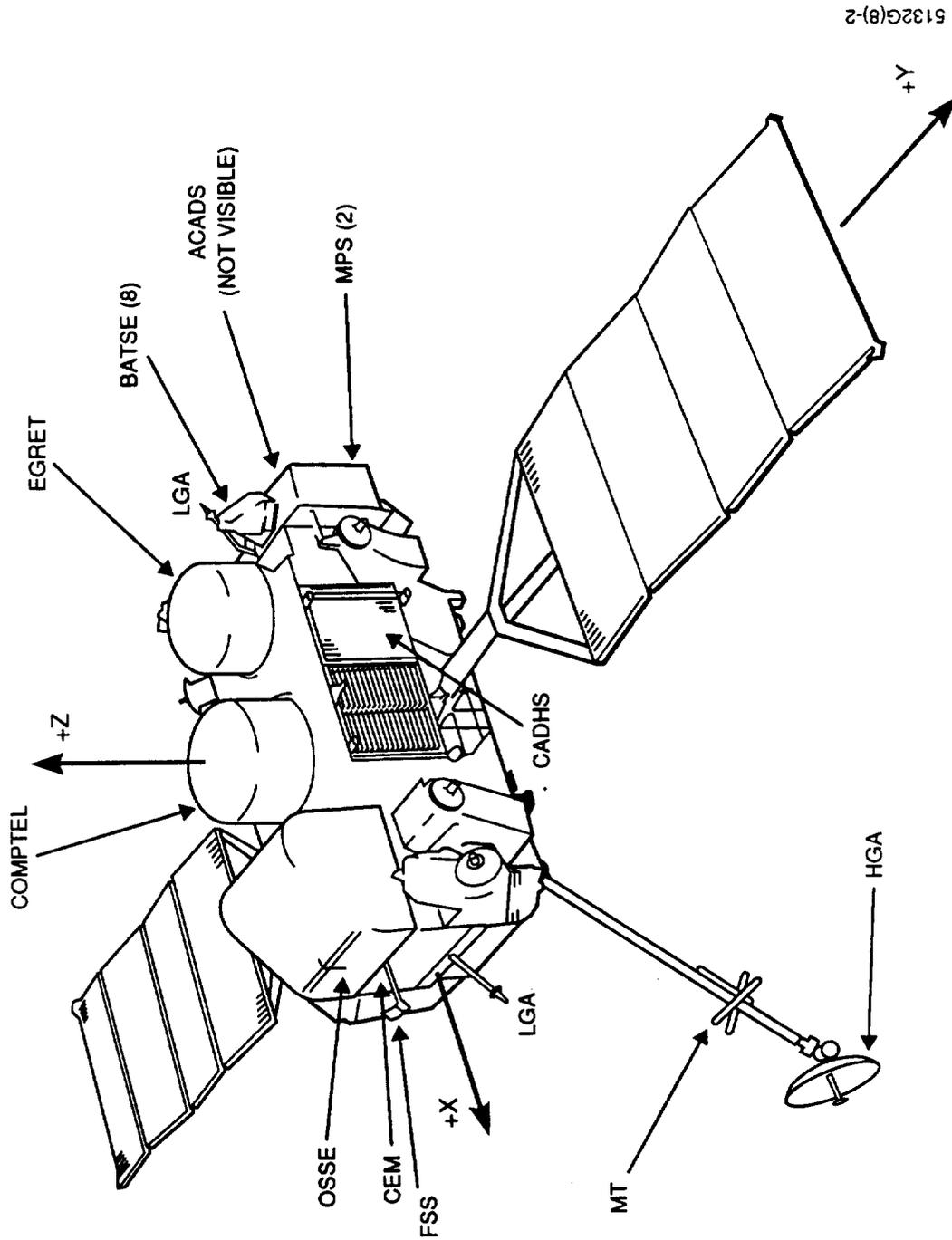


Figure 1-1. View of GRO Spacecraft From +Z Axis

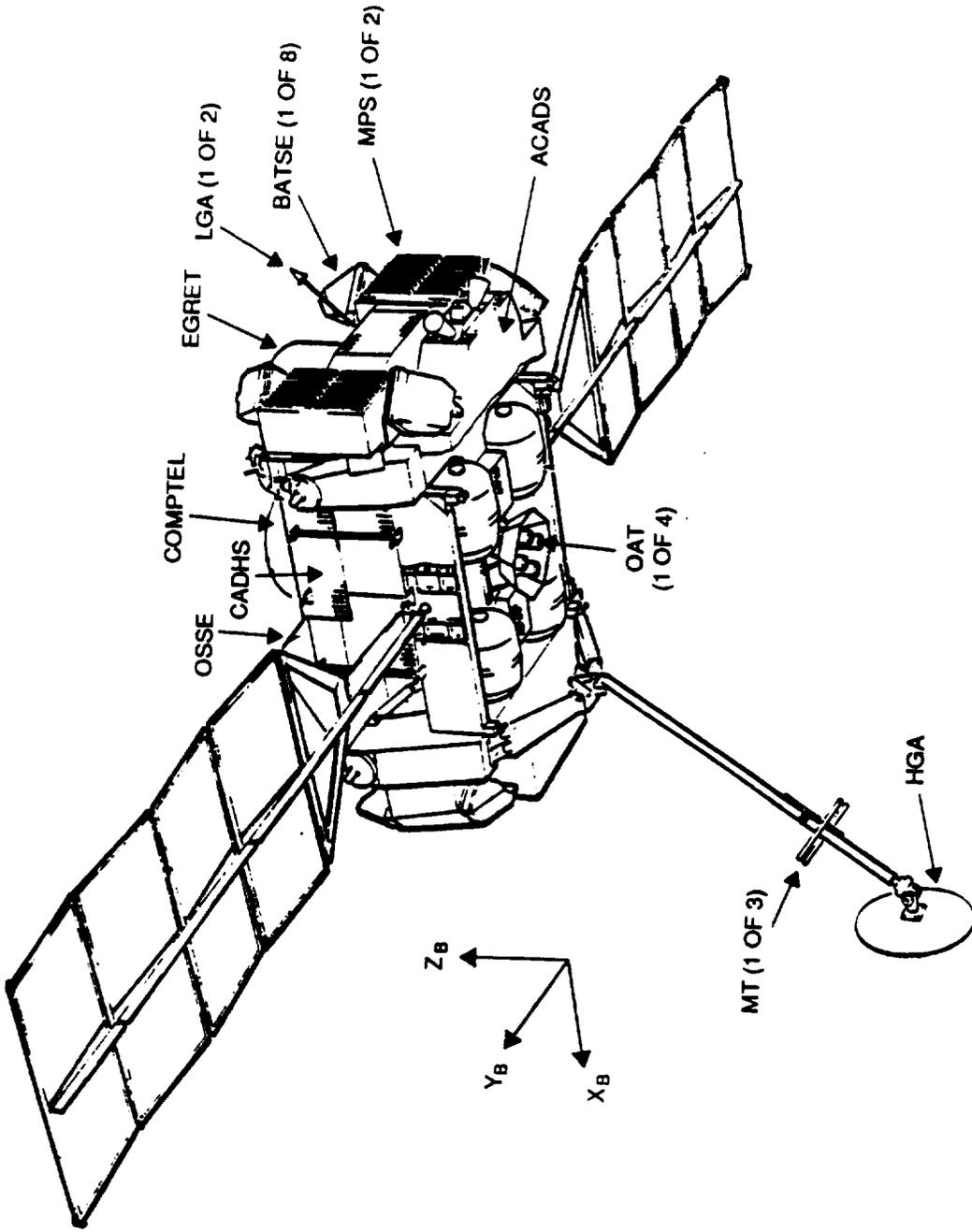


Figure 1-2. View of GRO Spacecraft From -Z Axis

Parameter	Value
Total mass (end of life (EOL), empty)	29,820 pounds mass (lbm) (13,555 kilograms (kg))
Length	9.2 meters (m)
Diameter	4.5 m
Total power	1,900 watts (W)
Hydrazine propellant mass	4,248 lbm (1,931 kg)
Moments of inertia (beginning of life (BOL)) (Maximum limits):	
Maximum weight	15,694 kg
I_{xx}	54,696 kg•m ²
I_{yy}	82,440 kg•m ²
I_{zz}	100,991 kg•m ²
I_{xy}, I_{yx}	32.5 kg•m ²
I_{xz}, I_{zx}	-3,974 kg•m ²
I_{yz}, I_{zy}	-6.8 kg•m ²

The major spacecraft subsystems and their capabilities are as follows:

1. Scientific Payload--OSSE, COMPTEL, EGRET, BATSE.
2. Thermal Control Subsystem (TCS)--Provides the capability to meet thermal requirements of all subsystems.
3. Propulsion Subsystem (PS)--Provides orbit transfer and orbit trim maneuver capabilities.
4. Electrical Power Distribution Subsystem (EPDS)--Uses the Multimission Modular Spacecraft (MMS) Modular Power Subsystems (MPSs) and the steerable solar arrays to provide electrical power to the subsystems.
5. Communications and Data Handling Subsystem (CADHS)--Provides capability for telecommunications and commands by using a high-gain antenna (HGA), two low-gain antennas (LGAs), and a CADH module similar to the MMS module. CADHS also provides an onboard computer (OBC). A GRO-unique feature is the time

transfer unit (TTU), which permits the synchronization of spacecraft clock time to universal coordinated time (UTC).

6. Attitude Control and Determination Subsystem (ACADS)--Provides the capabilities for attitude maneuvers and stability for science instrument viewing. ACADS is of the MMS heritage.

A component block diagram is shown in Figure 1-3.

1.2 ATTITUDE CONTROL AND DETERMINATION SUBSYSTEM

ACADS has five main functions:

1. Attitude control--Provides three-axis attitude control capabilities for inertial pointing, stabilization maneuvers, safehold, and thruster activation during velocity control.
2. Attitude determination--Determines attitude to support the control function and to provide definitive attitude annotation in science telemetry.
3. Ephemeris computation--Computes GRO and Tracking and Data Relay Satellite (TDRS) ephemerides by using data uplinked from the ground.
4. HGA pointing--Commands HGA to point to the TDRS based on computer ephemerides and on information uplinked from the ground on HGA/TDRS contact times.
5. Solar array control--Provides the capability to position the solar arrays in any required orientation about its rotational axis.

ACADS specifications are given in Reference 1.

1.2.1 ATTITUDE HARDWARE

All spacecraft attitude hardware is contained in the ACADS. The basic ACADS configuration is shown in Figure 1-4. It

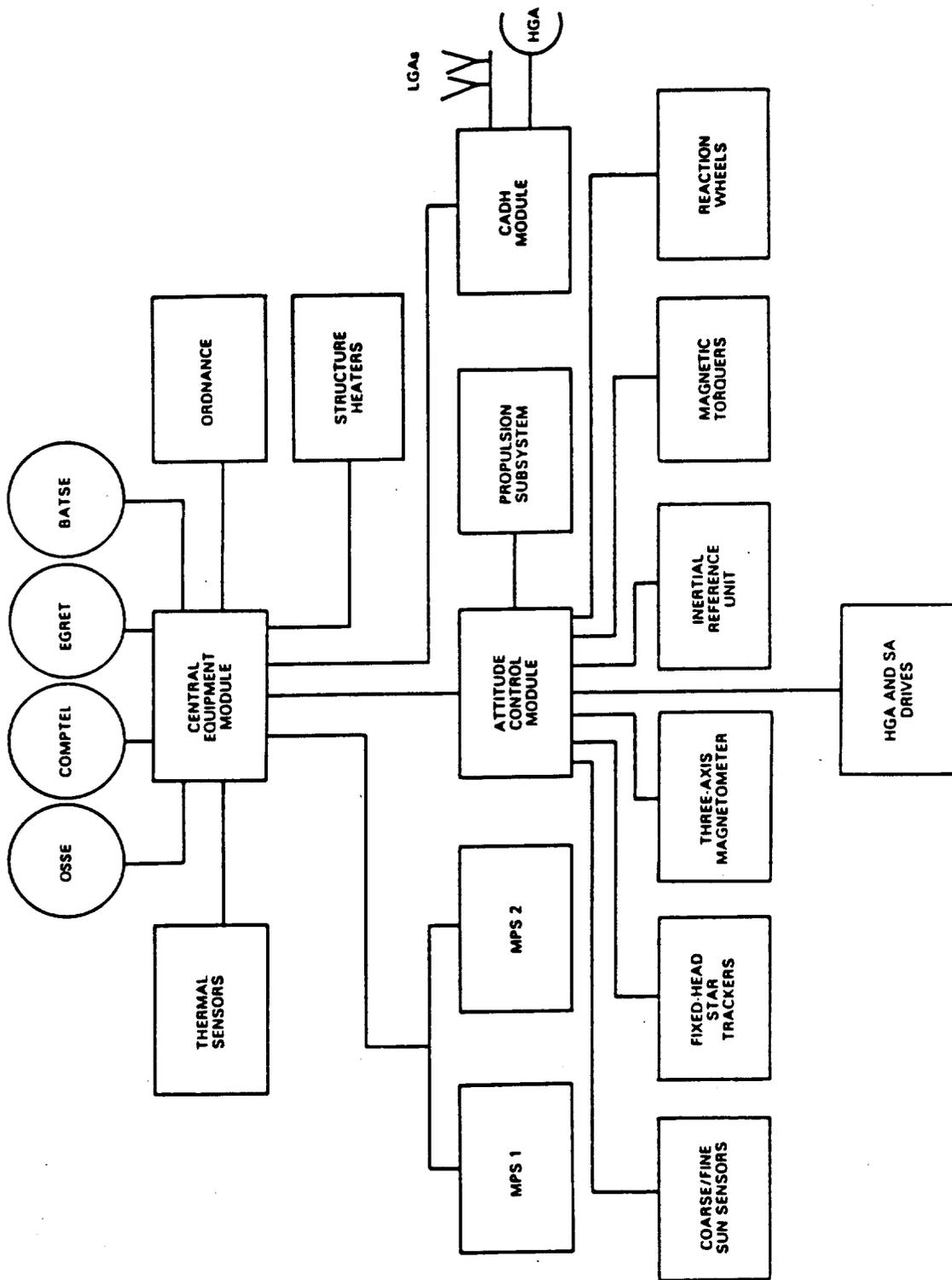


Figure 1-3. GRO Subsystems

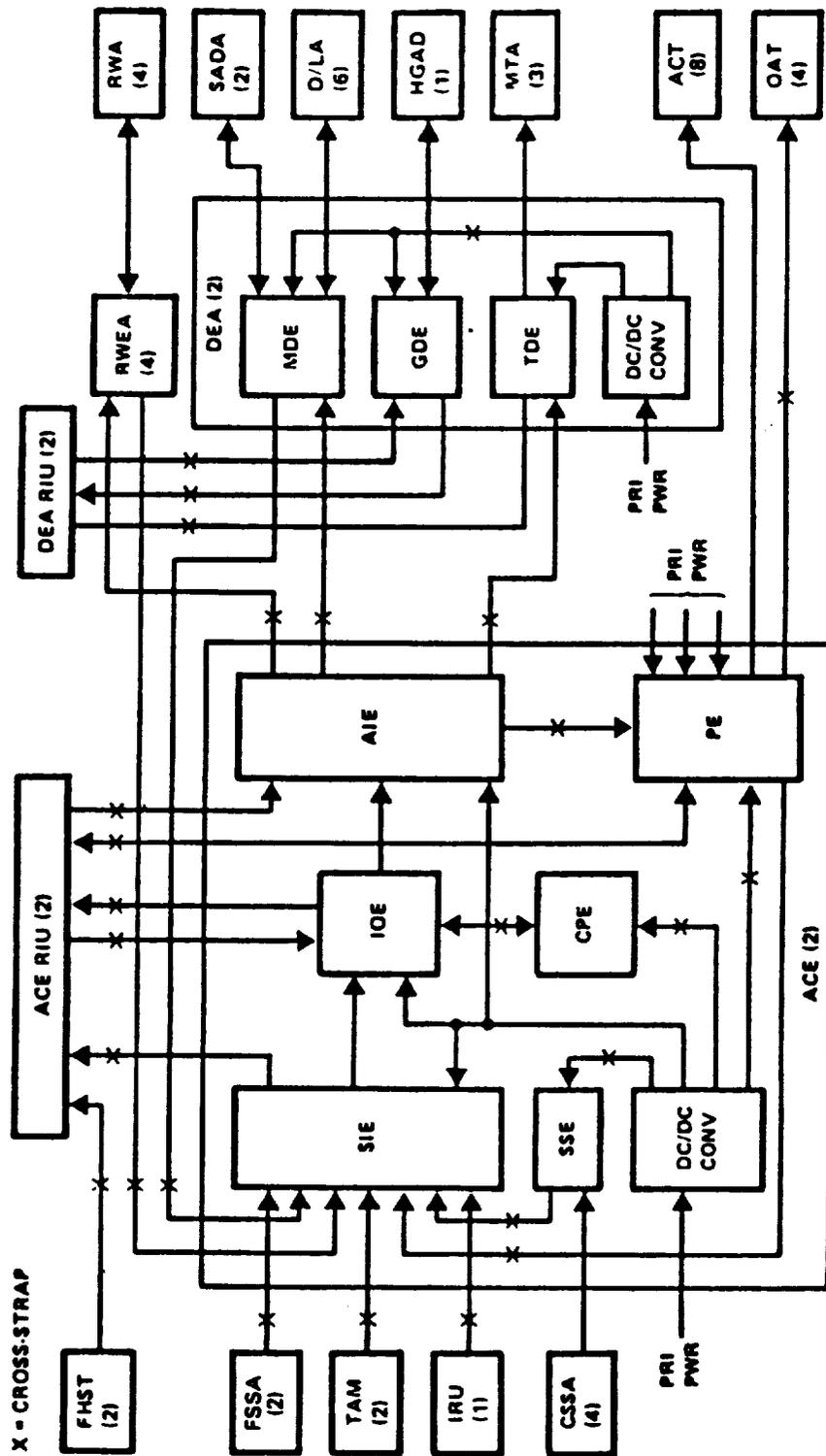


Figure 1-4. ACADS Configuration

shows the various sensor, electronics, and actuator assemblies that constitute ACADS and identifies the interfaces between them and other subsystems, such as CADHS, EPDS, and PS. The subassemblies that constitute the attitude control electronics (ACE) and the drive electronics assembly (DEA) are shown within larger boxes that represent the ACE and DEA, respectively. Arrows indicate power input. See Appendix C for acronym definitions.

ACADS components viewed from the +X axis and -X axis are shown in Figures 1-5 and 1-6, respectively. The OBC located in the CADHS is an integral part of ACADS. Other major components of ACADS are as follows:

- Fixed-head star trackers (FHSTs) (2)
- Inertial reference unit (IRU) (1)
- Three-axis magnetometers (TAMs) (2)
- Fine Sun sensors (FSSs) (2)
- Coarse Sun sensors (CSSs) (4)
- Magnetic torquers (MTs) (3)
- Reaction wheels (RWs) (4)
- Attitude control thrusters (ACTs) (8)
- Orbit adjust thrusters (OATs) (4)

The following component descriptions use information from References 2 through 9.

1.2.1.1 FHST

The FHST is an attitude sensor that searches for, detects, and tracks stars; provides accurate position and intensity information on stars in its field of view (FOV); and generates status flags and parameters that characterize the sensor operation. Specifications are as follows:

- Manufacturer--FHST is made by Ball Aerospace Systems Division (BASD); see Reference 3 for FHST details.

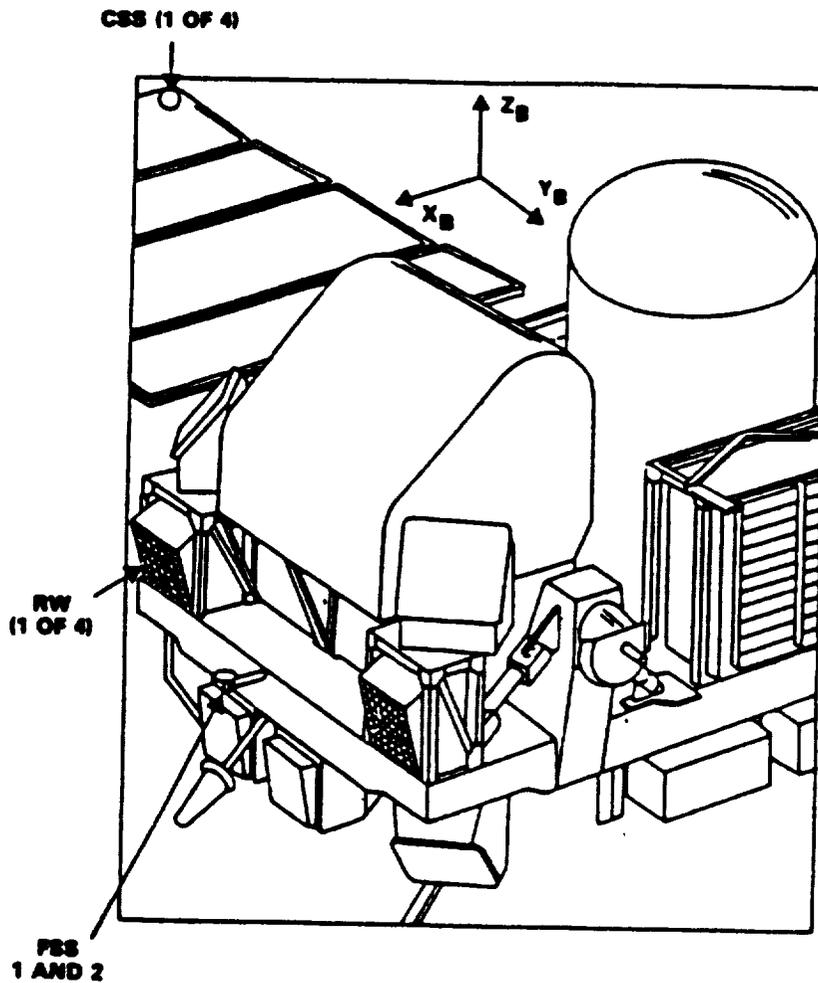


Figure 1-5. ACADS Components (View From +X Axis)

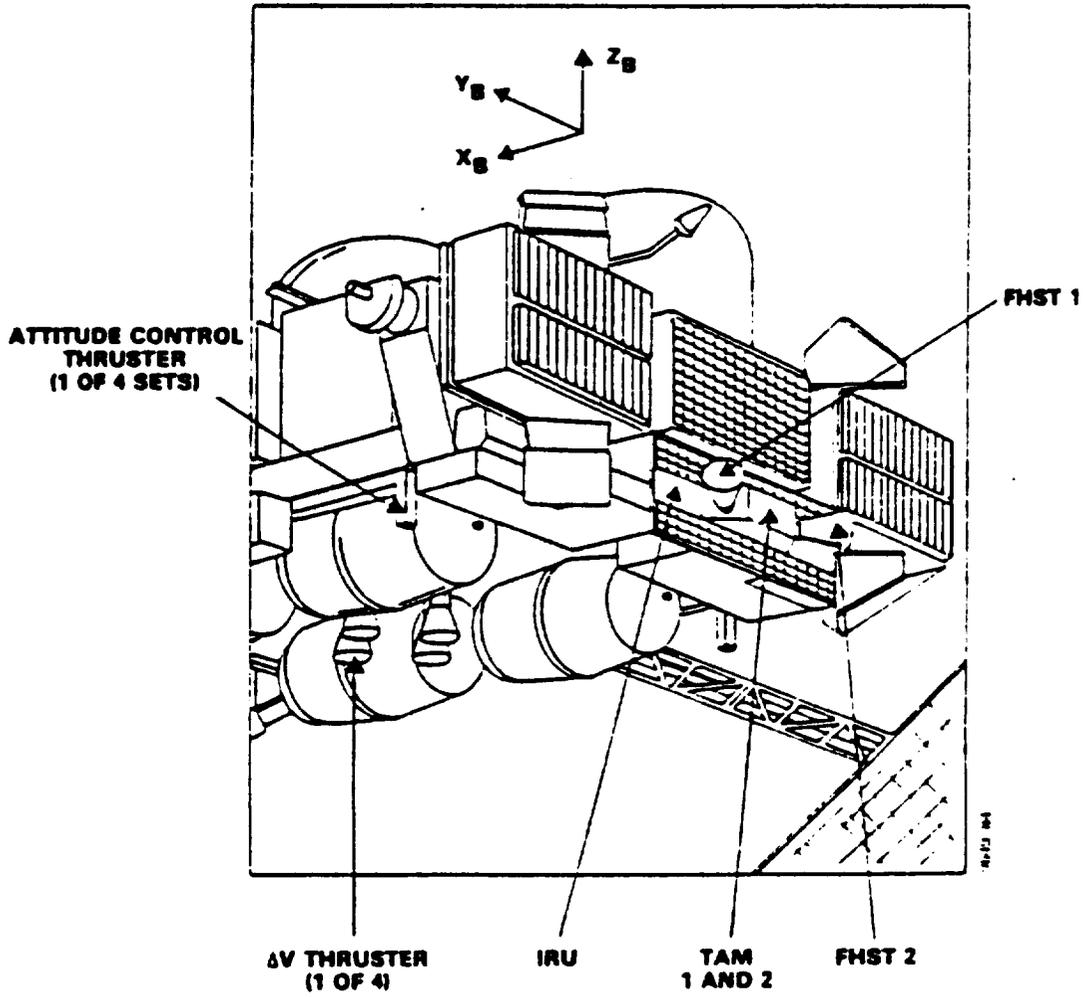


Figure 1-6. ACADS Components (View From -X Axis)

● Operation--Light from the object to be tracked enters an optical lens. The image of this object is focused on the photocathode of an image dissector tube (IDT), which magnetically deflects and focuses the electrons onto an aperture in a plate. The corresponding signal is amplified and processed to provide intensity and position information. The FHST operates in three modes:

- Search mode
- Track mode
- Offset mode

In the total field of view (TFOV), a search mode consists of a horizontal scan pattern with appropriate vertical shifts at the ends (raster). Four commandable thresholds set the minimum sensitivity for acquiring a star. Position and intensity output during the search mode do not convey meaningful information. When a star is acquired, the track mode begins.

A small cross pattern (in the form of a figure 8) is rapidly executed at the center of the star image. A STAR PRESENT flag is generated to indicate that star position and intensity data are valid for the tracked star. The track pattern remains locked on the star during attitude changes. If the star leaves the TFOV, if its intensity falls below the commanded threshold, or if a BREAK TRACK command is received from the user, then search mode resumes. During the track mode, other stars in the TFOV have no influence on the FHST or on its output.

With the optional offset mode capability, a small offset raster scan can be commanded in a reduced field of view (RFOV). If a star is acquired, it will be tracked throughout the TFOV. If the star is lost, a reduced scan will continue at the original position in the RFOV.

- Parameters and values--These are as follows:

Parameter	Value
TFOV	8 deg by 8 deg
RFOV	1.5 deg by 1.5 deg
Range of star visual magnitudes	+5.7 to -7
Number of threshold settings	4
Maximum tolerable vehicle rate	0.3 deg/second (sec)
Search mode:	
Scan type	Raster
Number of lines in TFOV	70
Maximum acquisition time	10 sec (TFOV); 1.5 sec (RFOV)
Track mode:	
Scan type	Unidirectional cross-scan
Scan period	100 milliseconds (msec)
Output rate	10/sec (each axis)
Accuracy	10 arc-sec (1σ) calibrated over 8-deg diameter circular FOV
Nominal data resolution	7 arc-sec

- GRO-specific features--FHST options for GRO include the following:

- Stray-light shade
- Bright object sensor
- Shutter housing assembly
- Self-test feature

GRO has two FHSTs arranged symmetrically with respect to the -X spacecraft body axis in the X-Y plane so that their bore-sights point to +45 degrees and -45 degrees from the -X axis. The approximate locations of the FHSTs are indicated in Figure 1-6.

1.2.1.2 IRU

The IRU is an attitude rate sensor that consists of a gyro package that measures inertial vehicle rates about the

sensor axes. Output consists of analog rates, accumulated angles, range status, and temperature. Specifications are as follows:

- Manufacturer--IRU is made by Teledyne; see Reference 4 for IRU details.

- Operation--The IRU consists of three spinning wheels or rotors. Each rotor is mounted on two gimbals to provide 2 deg of freedom and, therefore, rate information along two body axes (two-channel output). The six-channel IRU configuration provides dual redundancy along each body axis. The IRU assembly is fixed in the spacecraft (strap-down). The current required to magnetically torque a gimbal to maintain null deflection (torque rebalancing) is proportional to the accumulated rotation angle about the corresponding body axis (rate integrating mode). Torque current is differenced after small intervals of time to generate analog rates (rate mode). The IRU can operate either in a high-rate or a low-rate mode (range status).

- Parameters and values--These are as follows:

<u>Parameter</u>	<u>Value</u>
Scale factor stability	± 0.01 percent per month (low rate) ± 0.1 percent per month (high rate)
Alignment stability	TBD
Acceleration-insensitive drift rate (AIDR)	± 0.04 arc-sec/sec for 30 days of start-stop operation (low rate) ± 0.0003 arc-sec/sec for 6 hours of continuous operation (low rate) ± 0.001 deg/sec for 30 days (high rate)
Nominal data resolution	0.8 arc-sec/count (high rate); 0.05 arc-sec/count (low rate)
High-rate range	± 2.0 deg/sec
Low-rate range	± 400 arc-sec/sec

- GRO-specific features--These are as follows:
 - Low-power, radiation-resistant electronics
 - No heaters or other temperature controls (therefore, wide-range temperature performance is expected)

The approximate location of the IRU in the spacecraft is indicated in Figure 1-6.

1.2.1.3 TAM

The TAM is an attitude sensor that measures the magnetic field at the TAM position. Output along each orthogonal sensor axis is the measured magnetic field component along that axis. Specifications are as follows:

- Manufacturer--TAM is made by Schoenstedt; see Reference 5 for TAM details.
- Operation--The TAM uses a saturable-core transformer with a high-permeability core, an excitation winding, and a signal pickup winding. The excitation winding is driven with a 10-kilohertz (kHz) sine wave alternatively saturating the core in opposite directions. If there is an external magnetic field component parallel to the core, each time the saturable core makes a transition from one saturated extreme to the other, a pulse is induced into the signal pickup winding. The pulse is proportional to the component of the external field in that direction.

- Parameters and values--These are as follows:

<u>Parameter</u>	<u>Value</u>
Scale factor	10 volts (direct current)/1600 milligauss (mG)
Linearity	±0.02 percent full scale
Noise	<6 millivolts (mV) peak to peak

- GRO-specific features--The approximate location of TAMs in the spacecraft is indicated in Figure 1-6.

1.2.1.4 FSS

The FSS is an attitude sensor that provides two-axis Sun direction information with respect to the sensor axes. Output consists of the angles between the boresight and the projections of the Sun vector on the two orthogonal planes containing the boresight. Specifications are as follows:

- Manufacturer--FSS is made by Adcole; see Reference 6 for FSS details.

- Operation--Each two-axis FSS consists of two single-axis Sun sensors mounted orthogonally. Each single-axis Sun sensor consists of two reticles: a fine reticle and a coarse reticle. Each reticle is composed of two thin, fused silica plates separated by a fused silica spacer. Reticle patterns are located on the insides of the plates. Silicon photocell arrays are located beneath the reticles.

The coarse reticle pattern is gray coded and encodes the coarse angle over the total FOV. The fine reticle patterns and the resultant photocell currents are used to generate fine-angle data.

- Parameters and values--These are as follows:

<u>Parameter</u>	<u>Value</u>
FOV	64 deg by 64 deg for each FSS (126 deg by 64 deg total)
Nominal accuracy	0.019 deg
Nominal data resolution	0.004 deg
Stray light FOV	140 deg by 140 deg for each FSS

- GRO-specific features--These include two FSS units (heads) with one electronics unit. The approximate location of FSSs in the spacecraft is indicated in Figure 1-5.

1.2.1.5 CSS

The CSS is an attitude sensor that provides coarse Sun direction information with respect to the sensor surface normal. Specifications are as follows:

- Manufacturer--CSS is made by TRW; see Reference 7 for CSS details.

- Operation--There are four CSS units, each of which produces an output proportional to the cosine of the angle between the Sun direction and the sensor surface normal. This output can be combined to provide Sun unit vector information in body coordinates. The CSS will be used by the OBC and ground systems for attitude determination when the Sun is outside the FSS FOV.

- Parameters and values--These are as follows:

<u>Parameter</u>	<u>Value</u>
FOV	2π steradian (sr) for each CSS (4π sr total)
Nominal resolution	TBD deg
Scale factor variation	12.6 percent

- GRO-specific features--Four units are located at the corners of the solar panels (see Figure 1-5), with their surface normals in the plane of the solar panel rotated 45 deg with respect to the solar panel edges.

1.2.1.6 MT

The MT is an electromagnet used as an attitude actuator, which, when energized, generates a torque to effect changes in spacecraft angular momenta. Specifications are as follows:

- Manufacturer--MT is made by Ithaco; see Reference 8 for MT details.

- Operation--Each MT consists of two coils connected in a series and wound on a ferromagnetic rod. A current applied through the MT generates a magnetic dipole moment, which interacts with the magnetic field to produce a torque.

- Parameters and values--These are as follows:

<u>Parameter</u>	<u>Value</u>
Maximum magnetic dipole moment	2000 amperes (A)•m ²
Nominal residual magnetic dipole moment	<20 A•m ²
Nominal data resolution	TBD

- GRO-specific features--The MTs are located on the HGA boom (see Figure 1-2).

1.2.1.7 RW

The RW stores angular momentum and is an attitude actuator that can be used to change the angular momentum of the spacecraft. Specifications are as follows:

- Manufacturer--RW is made by Sperry; see Reference 9 for RW details.

- Operation--Each RW consists of an inertia wheel, a brushless direct current motor, and a tachometer enclosed in a sealed housing. A command issued to speed up or slow down the wheel speed results in a change in spacecraft angular momentum; this occurs because the wheel assembly is rigidly fixed in the spacecraft and because the total angular momentum of the spacecraft plus wheel system has to be conserved in the absence of external torques.

- Parameters and values--These are as follows:

<u>Parameter</u>	<u>Value</u>
Maximum angular momentum	540 newtons (N)•m•s
Angular velocity	-6000 revolutions per minute (rpm) to +6000 rpm

- GRO-specific features--These are similar to those built for the Space Telescope. Four wheels are arranged in a pyramid for redundancy. The location of the RWs is indicated in Figure 1-5.

1.2.1.8 Thrusters

Thrusters can be used as attitude actuators. The reaction forces on the spacecraft caused by the ejection of the gas propellant generate the needed torques to maneuver the spacecraft. Specifications are as follows:

- Manufacturer--The orbit adjust thrusters (OATs) are made by Hamilton Standard; the attitude control thrusters (ACTs) are made by TRW.

- Operation--The propulsion system consists of four OATs for orbit transfer and attitude maneuvers and four pairs of ACTs for attitude maneuvers only. The OATs are fired simultaneously for orbit transfer. They are fired in an off-modulation mode for attitude maneuvers during orbit transfer. The ACTs are operated in pairs to produce control torques along the desired axis.

- Parameters and values--These are as follows:

<u>Parameter</u>	<u>Value</u>
OATs	
Effective distance from spacecraft center line (moment arm)	33 centimeters (cm) each
Thrust level	444.8 N (100 pound-force (lbf))(BOL) 155.7 N (35 lbf) (end of life (EOL))
Thrust level uncertainty (imbalance)	5 percent (3 σ)
ACTs	
Effective distances from spacecraft center line (moment arm)	TBD

Parameter	Value
Thrust level	22.2 N (5 lbf) (BOL), 8.9 N (2 lbf) (EOL)
Thruster misalignment	1 deg (3 σ) in X- and Y-axes
Center-of-gravity offset	1.9 cm (3 σ) on each axis
Disturbance torques by OATs	1 - 14 N•m

- GRO-specific features--The location of the OATs and ACTs is indicated in Figure 1-6.

1.2.2 MODES OF OPERATION

All ACAD control modes, conditions, and automatic sequences will be subject to override by command, even in the presence of a single failure.

OBC-controlled modes requiring thruster firings include a commandable thruster drive inhibit capability by which all normal control law operations are performed and telemetered but no actuation signals are applied to the thruster valves.

The ACAD provides capabilities for operating in the control modes defined below. Equipment usage is as given in Table 1-1, and allowable transitions are as shown in Figure 1-7.

1.2.2.1 Standby Mode

The standby mode places attitude control torquers in a quiescent state. The entry to and exit from the standby mode will be by external command. The capability for operating the motor drive electronics (MDE) is provided.

While in the standby mode, the ACAD will accept and process commands for execution upon transfer to other modes. Command storage capability will be provided for the modes controlled by the ACAD electronics.

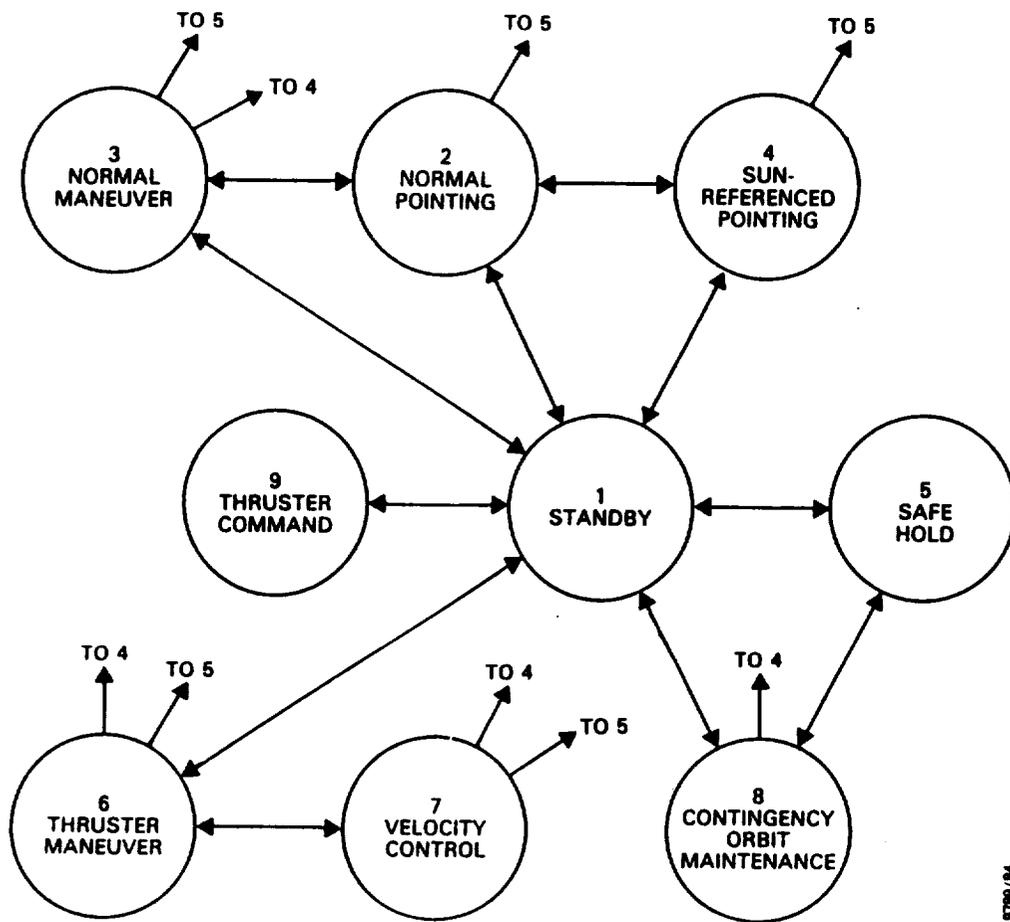
Table 1-1. ACAD Modes and Equipment Usage

EQUIPMENT USAGE	FIXED-HEAD STAR TRACKER	INERTIAL REFERENCE UNIT	COARSE SUN SENSOR	FINE SUN SENSOR	TRIAXIAL MAGNETOMETER	MAGNETIC TORQUERS	REACTION WHEELS	LOW-LEVEL THRUSTERS	AV THRUSTERS	ONBOARD COMPUTER	BACKUP CONTROL ELECTRONICS	SOLAR ARRAY DRIVE	ANTENNA DRIVE
1. STANDBY	T	T	T	T	T					A		A	
2. NORMAL POINTING	A	A	B		A	A	A			A		A	A
3. NORMAL MANEUVER	T	A			T		A			A		A	
4a. SUN-REFERENCED POINTING	T	A	A	A	A	A	A				A	A	
4b. REACTION WHEEL ATTITUDE HOLD		A			A	A	A				A	A	
5a. SAFE-HOLD		A	A	A	T			A			A	A	
5b. THRUSTER ATTITUDE HOLD		A			T			A			A	A	
6. THRUSTER MANEUVER		A			T			A		A		A	
7. VELOCITY CONTROL		A			T			A	A	A		A	
8. CONTINGENCY ORBIT MAINTENANCE		A			T			A	A	A		A	
9. THRUSTER COMMAND		T	T	T	T			A	A		A	A	A

NOTES:

- A ACTIVE
- T USED FOR TELEMETRY ONLY
- B USED FOR BACKUP ONLY

978977-1/84



9780/84

Figure 1-7. ACAD Control Mode Transitions

1.2.2.2 Normal Pointing Mode

The normal pointing mode allows the GRO to be held in any desired orientation, using attitude error data provided by a stellar/inertial attitude reference. Attitude stabilization and control are accomplished by reaction torques produced by a set of four reaction wheels. Transfers to and from the normal pointing mode will be by external command only, except in the case of a detected anomaly, in which case a transfer to either the Sun-referenced pointing or the safe-hold modes will take place automatically. Wheel momentum unloading is accomplished by using magnetic torquers operating in a closed-loop control configuration with Earth magnetic field data provided by TAMs.

While in the normal pointing mode, the solar array position is adjustable as required by external command.

High-gain antenna pointing can be controlled by either ACAD subsystem or external commands. HGA pointing commands are derived in the OBC by using GRO and TDRS ephemerides computed on board with the use of data provided regularly by ground command.

1.2.2.3 Normal Maneuver Mode

Attitude maneuvers at rates within the capability of the GRO reaction wheel system are performed in the normal maneuver mode. The attitude control laws used in this mode minimize excitations of flexible modes. Transfers to and from the normal maneuver mode will be external command only, except in the case of a detected anomaly, in which case transfers to either the Sun-referenced pointing or the safe-hold modes will be made automatically.

Attitude information will be provided by the IRU without stellar updates, and control torques will be developed by the reaction wheels. TAM measurements will provide data

to telemetry for post facto attitude determination on the ground.

1.2.2.4 Sun-Referenced Pointing Mode

The main objective of the Sun-referenced pointing mode is to allow the GRO +X axis to be pointed at the Sun by using attitude error data supplied by the coarse and fine Sun sensors and control torques developed by the reaction wheels. Momentum unloading is accomplished as described in Section 1.2.2.2 for the normal pointing mode. Operation in the Sun-referenced pointing mode will be initiated by external command or by the occurrence of a detected anomaly. Mode termination will be by external command or by the occurrence of a detected anomaly that requires a transfer to the safe-hold mode.

Transfer to the Sun-referenced pointing mode automatically initiates an attitude hold condition in which the GRO attitude is stabilized by reaction wheels, using rate information provided by the IRU. During this attitude hold condition, the solar array will be slewed to a preselected index position. Upon completion of the solar array slew, a Sun acquisition will be initiated automatically.

The control and steering laws for the Sun-referenced pointing mode are mechanized in the ACE. In the absence of valid IRU rate data, the control laws have provisions to operate with a derived rate.

Stabilization about the X-axis is accomplished by using reaction wheel torques and attitude data derived from IRU measurements. The capability is provided for inducing an adjustable rotation rate about the X-axis that will be used for star identification on the ground and for star acquisition.

1.2.2.5 Safe-Hold Mode

The primary objectives of the safe-hold mode are to place GRO in Sun-referenced pointing orientation and to position the solar arrays for maximum power in the event of detected anomaly. The safe-hold mode will be initiated by external command or by the occurrence of a detected anomaly. The mode will be terminated by external command only. Transfer to the safe-hold mode automatically initiates an attitude hold condition in which the attitude of the observatory is stabilized by means of thrusters operating with rate information provided by the IRU. During this attitude hold condition, the solar array will be slewed to a preselected index position. Upon completion of the solar array slew maneuver, a Sun acquisition will be performed by means of ACTs, using attitude data obtained from the coarse and fine Sun sensors and rate information provided by the IRU.

Provisions are made for holding attitude automatically during Sun eclipse and by external command at any time.

Control laws for the safe-hold mode will be implemented in the ACE. In the absence of valid IRU rate data, the control laws have provisions for operating with a derived rate.

1.2.2.6 Thruster Maneuver Mode

Attitude maneuvers at rates that exceed reaction wheel momentum storage capabilities can be executed in the thruster maneuver mode. Control torques are provided by the ACTs, and attitude data are derived from IRU information. Control laws are implemented in the OBC. The thruster maneuver mode will be initiated and terminated by ground command except on detection of a confirmed anomaly, which will cause a direct transfer to either the Sun-referenced pointing or the safe-hold modes. While in the thruster maneuver mode, the momentum unloading function is disabled. Magnetometer data will

be telemetered for post facto attitude determination on the ground.

1.2.2.7 Velocity Control Mode

The velocity control mode will be used to execute velocity correction maneuvers and to control the attitude of GRO during these maneuvers. In the velocity control mode, the ACADS controls the firings of the velocity control thrusters to produce the required firing durations and to control GRO attitude about the pitch and roll axes by off modulation. The ACTs are used to control the attitude about the yaw axis. Firing durations are selectable by external command. The required initial orientations for velocity correction maneuvers are attainable by means of either the normal or the thruster maneuver modes. Means are provided to automatically rotate the GRO Z-axis for maintaining the thrust vector along the orbit velocity direction. Attitude error information is provided by the IRU, operating without celestial updates. Magnetometer data will be telemetered for ground attitude determination. Control laws are implemented in the OBC. The velocity control mode will be initiated and terminated by external command only, except in the event of a detected OBC failure or other selected abnormal observatory conditions, which will cause automatic turnoff of the ΔV thrusters and a transfer to either the Sun-referenced pointing or the safe-hold modes.

1.2.2.8 Thruster Command Mode

The thruster command mode allows any selected low-level thruster to be fired directly, in real time, by external commands. The firings will be in impulses of fixed duration. Two impulse durations are selectable by external command. Each command will produce only one impulsive firing. Activation of the thruster command mode will be by ground command only, by using fault tolerant commands.

The control logic for the thruster command mode is mechanized in the ACE, separately from the OBC.

1.2.2.9 Backup Orbit Maintenance Mode

This mode provide capabilities to orient the observatory and to control the propulsion subsystem thrusters with logic that is independent of the CADH Subsystem OBCs in such a manner that the GRO orbit can be maintained for a period of up to 3 months.

Attitude maneuvers for attaining the orientation required for ΔV firings will be made with control torques provided by the low-level thrusters and attitude data provided by the IRU.

Velocity increments will be attained by firing either one of the available pairs of ΔV thrusters. Three-axis attitude control during ΔV firings will be provided by the low-level thrusters.

1.3 DYNAMICS SIMULATOR OVERVIEW

The purpose of the GRO Dynamics Simulator is to test and evaluate GRO flight software under conditions that simulate the expected in-flight environment as closely as possible. The complete simulator system, shown in Figure 1-8, consists of five logical subsystems (which may or may not correspond to separate programs when developed):

- Profile
- Simulation Control and Input/Output (SCIO) Subsystem
- Truth Model
- OBC Model
- Postprocessor

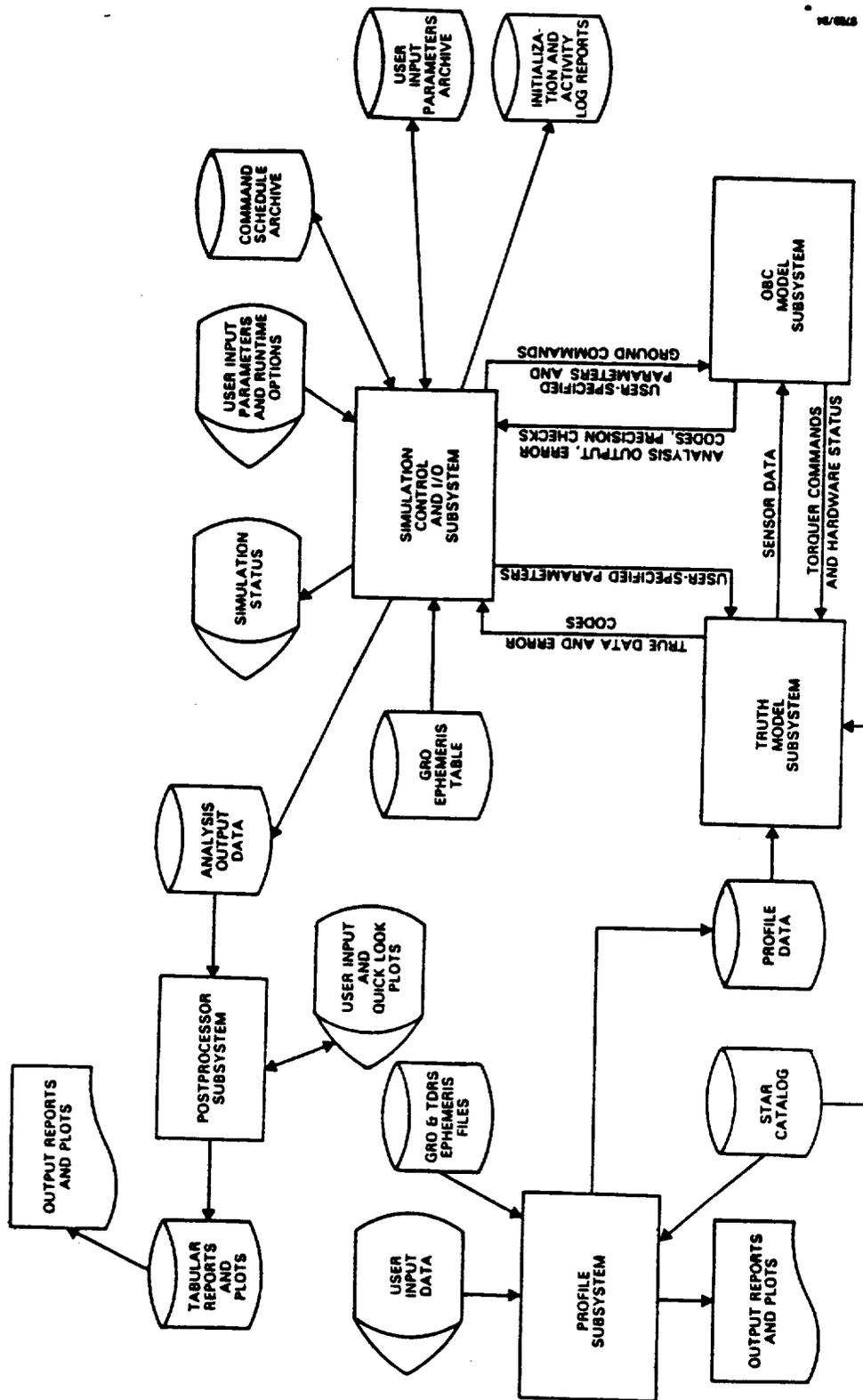


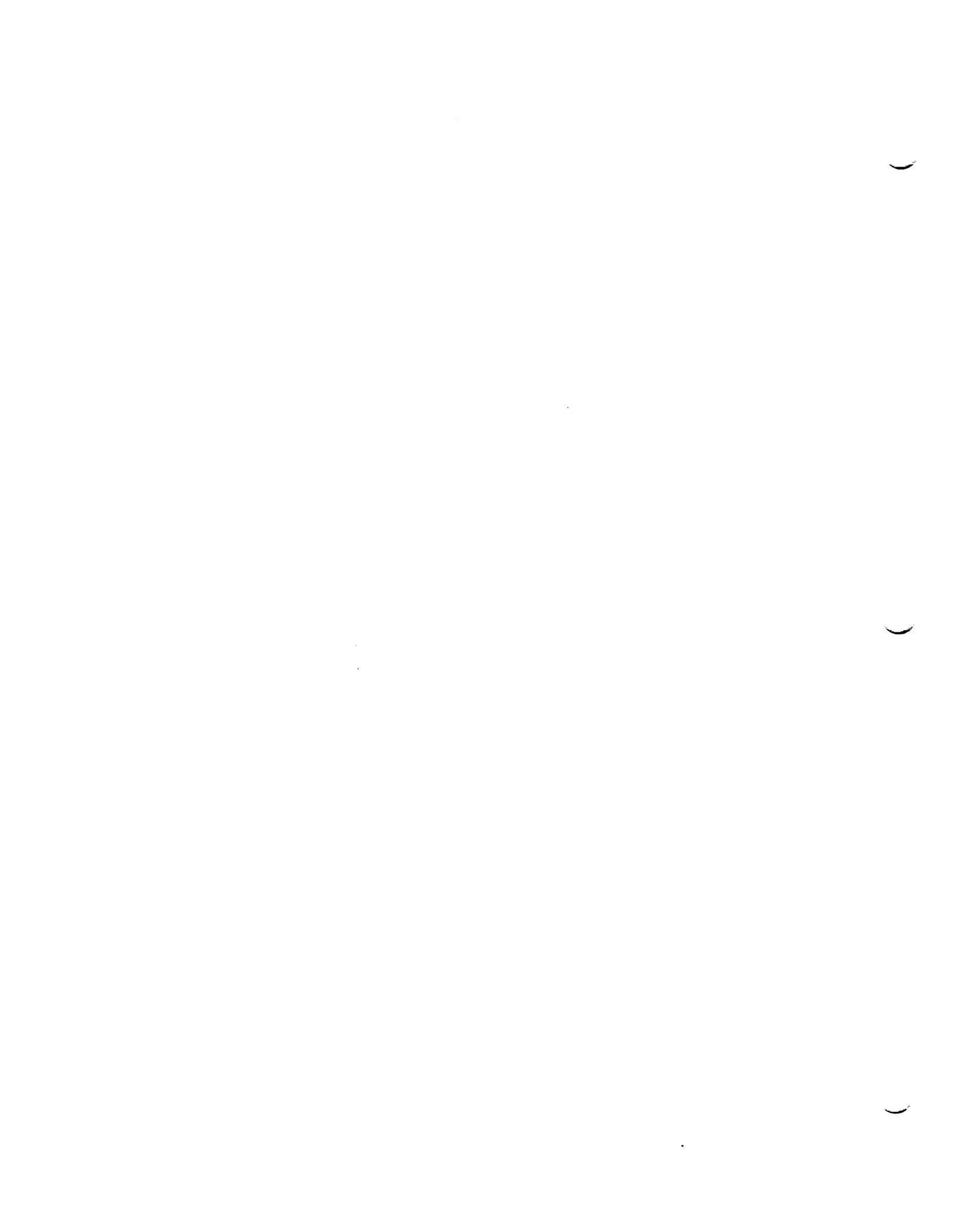
Figure 1-8. GRO Dynamics Simulator Data Flow

The Profile Subsystem will generate the environmental torque profile for GRO. In addition, it will generate the spacecraft, solar, and lunar ephemerides; the Earth's magnetic field vector; time-varying moments of inertia; and star position vectors. The Profile Subsystem is discussed in detail in Section 2.

The SCIO Subsystem will be the executive driver for the simulator. It will accept user input, initialize parameters, control simulation, and generate output data files and a simulation status display. The SCIO Subsystem is discussed in detail in Section 3.

The Truth Model Subsystem uses the ephemerides, magnetic field, star position vectors, and disturbance torque information supplied by the Profile Subsystem and the control torques to obtain the current spacecraft state and sensor measurements to simulate the response of the attitude hardware. The Truth Model Subsystem is discussed in detail in Section 4.

The OBC Model Subsystem uses sensor data provided by the Truth Model, a Kalman filter for attitude determination, and control laws to compute attitude control commands that are provided as input to the Truth Model. The OBC Model Subsystem is discussed in detail in Section 5. The attitude control electronics (ACE), which is implemented as part of the OBC Subsystem, is discussed in Section 6.



SECTION 2 - PROFILE SUBSYSTEM

The Profile Subsystem for GRO will calculate the external environmental torque profile for the spacecraft on the bases of a normal orbit and attitude, mass properties data, and geometric configuration. The environmental torques modeled will include aerodynamic, solar pressure, gravity-gradient, and residual magnetic dipole torques. The program will also generate spacecraft and solar ephemerides, Earth's magnetic field vector, and star position vectors.

2.1 REQUIREMENTS

2.1.1 PROCESSING REQUIREMENTS

- R1.1.1 Calculate external environmental torque profile for GRO on the bases of a nominal orbit and attitude, mass properties data, and geometric configuration.
- R1.1.2 Model environmental torques, including gravity-gradient, aerodynamic, solar pressure, and residual magnetic dipole torques.
- R1.1.3 Generate spacecraft, solar, lunar, and TDRS ephemerides and the Earth's magnetic field vector.
- R1.1.4 Compute the angular momentum contributed by the rotation of the solar arrays and TDRS antenna.
- R1.1.5 Calculate apparent spacecraft-to-star unit vector for each star in the FHST fields of view.
- R1.1.6 Provide the following three methods of calculating the initial attitude:
 - Using the calculated Sun vector and input coordinates of a point on the celestial sphere

- Using input coordinates of the body X-axis and the target point
 - Using the input attitude quaternion
- R1.1.7 Model a nominal maneuver.
- R1.1.8 Model TDRS tracking by the high-gain antenna.
- R1.1.9 Perform all internal computations in the mks system of units and express all angular quantities in radians
- R1.1.10 Use only righthanded orthogonal coordinate systems
- R1.1.11 Compute the time varying inertia properties due to movement of the HGA and solar arrays
- R1.1.12 Compute the orientation angles (separately) for the solar arrays
- R1.1.13 Compute the pointing angles for the HGA
- R1.1.14 Compute spacecraft velocity

2.1.2 INPUT REQUIREMENTS

- R1.2.1 The Profile Subsystem will provide defaults for all user input parameters.
- R1.2.2 The Profile Subsystem will support interactive user modification or input of the following information:
- Start and stop times of the run
 - Interval stepsize
 - Magnetic field parameters (order of field, constant magnetic field bias, and residual magnetic dipole bias)
 - Epoch time and initial Keplerian orbital elements

- Options for generating Sun, Moon, TDRS, and spacecraft ephemerides
- Star tracker parameters, target star criteria, and star catalog
- Debug option
- Parameters to describe a maneuver
- Order of magnetic field model
- Components of moment of inertia tensor
- Drag coefficients (other than default)

R1.2.3 The Profile Subsystem will not support user to input or modification of the following permanent information that relates to the satellite geometry:

- Number and types of elements in satellite decomposition
- Area of individual spacecraft element
- Normal-to-spacecraft surface vectors
- Solar radiation pressure and aerodynamic torque constants
- Vector from spacecraft center of mass to center of pressure for each element
- Solar radiation pressure values
- Aberration coefficients

2.1.3 OUTPUT REQUIREMENTS

R1.3.1 The Profile Subsystem will output the following information at a user-specified interval in a

form that can be input to the Truth Model Subsystem portion of the simulator:

- Start time and stop time
- Number of records
- Current time
- Earth's magnetic field
- Unit vector from spacecraft to Sun
- Vectors from spacecraft to Earth and Moon
- HGA drive (HGAD) pointing angles (one pair of angles for each TDRS)
- Velocity vector
- Individual external torque vectors
- Components of moment of inertia tensor
- Aerodynamic drag coefficient
- Star unit vectors
- TDRS Fourier coefficients
- Spacecraft center of mass

R1.3.1.1 The output data types (R*8, R*4, etc.) should remain the same as in the existing ERBS Profile Program (Reference 10).

R1.3.1.2 All output should be in the meter-kilogram-second (mks) system of units, with all angular quantities in radians.

R1.3.2 The Profile Subsystem will print a profile initialization report containing the following information:

- Start and end times
- Interval stepsize

- Magnetic field variables
- Input orbital elements
- Methods for generating solar, lunar, TDRS, and spacecraft ephemerides

R1.3.3 The Profile subsystem will provide printed output at each time step that contains the following information:

- Current time
- Earth magnetic field
- Unit vector from spacecraft to Sun
- Vectors from spacecraft to Earth and Moon
- Velocity vector
- Total torque vector
- Components of moment of inertia tensor
- Individual torque vectors
- Star unit vectors

2.2 MATHEMATICAL SPECIFICATIONS

2.2.1 INITIALIZATION OF SPACECRAFT ATTITUDE

Since the spacecraft is to remain inertially fixed in space for at least 14 days, the attitude matrix [A] will remain fixed for a simulator run unless a maneuver is executed (see later discussion). Provision should be made to initialize the attitude in any one of the following three ways:

1. Input the coordinates of the point on the celestial sphere, right ascension and declination, and generate an attitude matrix consistent with Sun-spacecraft constraints.
 - a. For epoch time, obtain the Sun vector, \vec{R}_S , in geocentric Cartesian inertial coordinates, as described in Section 2.2.2.1.
 - b. Form the Sun unit vector, \hat{R}_S .

c. Generate the attitude matrix [A] such that

$$[A] = \begin{bmatrix} X_X & X_Y & X_Z \\ Y_X & Y_Y & Y_Z \\ Z_X & Z_Y & Z_Z \end{bmatrix} = [\hat{X} | \hat{Y} | \hat{Z}] \quad (2-1)$$

where $\hat{Z} = [Z_X, Z_Y, Z_Z]$ is the spacecraft body Z-axis, which is the scientific instrument observing direction, and is therefore given by

$$\hat{Z} = [\cos \delta \cos \alpha, \cos \delta \sin \alpha, \sin \delta]^T \quad (2-2)$$

and

$$\hat{Y} = (\hat{Z} \times \hat{R}_S) / |\hat{Z} \times \hat{R}_S| \quad (2-3)$$

$$\hat{X} = \hat{Y} \times \hat{Z} \quad (2-4)$$

This method puts the body X-axis in the spacecraft-Sun-target point plane.

2. Input the right ascension and declination of both the body X-axis and the target point. Generate the attitude matrix as

$$[A] = [\hat{X} | \hat{Y} | \hat{Z}]$$

where

$$\hat{Z} = [\cos \delta_T \cos \alpha_T, \cos \delta_T \sin \alpha_T, \sin \delta_T] \quad (2-5)$$

$$X = [\cos \delta_X \cos \alpha_X, \cos \delta_X \sin \alpha_X, \sin \delta_X] \quad (2-6)$$

$$Y = Z \times X \quad (2-7)$$

3. Input the attitude quaternion, q. Generate the attitude matrix as in the utility EULERC (Reference 10):

$$[A] = \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix} \quad (2-8)$$

2.2.2 EPHEMERIDES AND ABERRATION

2.2.2.1 Spacecraft, Solar, and Lunar Ephemerides

The spacecraft, solar, and lunar ephemerides are obtained by a call to a FORTRAN ephemeris package, EPHEMS. EPHEMS is a modified version of subroutine EPHEMX, which is described in Reference 11). Options for obtaining spacecraft, solar, and lunar ephemerides have been modified. All subroutines referenced, except subroutine FASTOX, are documented in Reference 11. FASTOX is described in Reference 12 and in Appendix C.

The subroutines available for obtaining the GRO and TDRS spacecraft position and velocity are

1. FASTOX (Earth orbit)
2. DTAPRE (Earth orbit) (entry point of ROLTAP)

The subroutine for obtaining the solar and/or lunar ephemeris is SMPOS. SMPOS is described in Reference 11 and Appendix C.

2.2.2.2 Star Ephemeris

The star ephemeris will not originate in the Profile Program. The star data to be used will be obtained from a subset of the star catalog generated by the SKYMAP Program. The star coordinates will be converted to a unit vector in geocentric inertial (GCI) coordinates by

$$\widehat{\text{STAR}} = [\cos \delta \cos \alpha, \cos \delta \sin \alpha, \sin \delta]^T \quad (2-11)$$

where $\widehat{\text{STAR}}$ is the unit vector from the spacecraft to the star in GCI coordinates, and α and δ are the celestial coordinates for the star.

2.2.2.3 Aberration

During the time that light travels from the Sun or a star to the spacecraft, the position of the Sun or star relative to the spacecraft changes. Therefore, the Sun or star position, as viewed from the spacecraft, is not the true Sun or star position. Since the Sun and star positions provided to the Profile Subsystem are the true positions, they should be corrupted by aberration effects.

The apparent position of the Sun and stars is computed as follows:

$$\widehat{\mu} = \frac{\widehat{\mu}_0 + (\vec{V}_E/c) + (\vec{V}_S/c)}{|\widehat{\mu}_0 + (\vec{V}_E/c) + (\vec{V}_S/c)|} \quad (2-12)$$

where $\widehat{\mu}$ is the apparent unit position vector of the Sun or star in GCI coordinates, $\widehat{\mu}_0$ is the true unit position vector of the Sun or stars in GCI coordinates (from ephemeris), \vec{V}_E is the Earth velocity vector in GCI coordinates, \vec{V}_S is the spacecraft velocity vector in GCI coordinates, and c is the speed of light.

The unit vector in the direction of the Earth velocity vector is given by

$$\hat{V}_E = \frac{\vec{R} \times \hat{W}}{|\vec{R} \times \hat{W}|} \quad (2-13)$$

where \vec{R} is the Earth-to-Sun vector in GCI coordinates (from ephemeris) and \hat{W} is the ecliptic north pole (in GCI coordinates).

The spacecraft velocity vector, \vec{V}_s , is obtained from the ephemeris routines identified in Section 2.2.2.1.

2.2.3 MAGNETIC FIELD

The magnetic field is modeled by an eighth-order spherical harmonic expansion, the International Geomagnetic Reference Field (IGRF). (See Appendix H of Reference 11.) This expansion is computed by subroutine MAGFLD (Reference 10).

2.2.4 ANGULAR MOMENTUM OF INTERNAL MOTION COMPUTATION

The coordinates of the center of mass (CM) of the spacecraft components (main body, solar array, and antenna) are required for computing the component inertia tensors, which in turn are required for determining the angular momentum attributable to internal motions and the gravity-gradient torque.

For Sections 2.2.4.1 and 2.2.4.2 let

- B_j represent spacecraft component j , $j = 0$ (main body), 1, 2, 3
- \vec{R}_c = CM of the total system with respect to the origin of the body coordinate system BCS (OBCS)
- $\vec{r}_j(t)$ = CM of B_j with respect to OBCS
- \vec{p}_j = CM of B_j with respect to \vec{R}_c
- \vec{h}_j = hinge point of B_j with respect to OBCS
- m_j = mass of B_j

$\vec{\omega}_j$ = angular velocity vector of B_j in inertial space
 $\vec{\omega}_j^i$ = angular velocity vector of B_j with respect to B_0
 ($j = 1, 2, 3; \vec{\omega}_j = \vec{\omega}_0 + \vec{\omega}_j^i$)

2.2.4.1 Solar Arrays and High-Gain Antenna Angular Rates

The high-gain antenna angular rates, $\dot{\alpha}$, are obtained from the high-gain antenna module, where they are computed as indicated in Section 2.2.7.

Let

$$\hat{R}_{SB} = [A] \hat{R}_S$$

and

$$a = \tan^{-1} R_{SB_Z} / R_{SB_X} \quad (2-14)$$

where α is the angular displacement of the solar array normal from the X-Y body coordinate plane, which is limited to the range $-90^\circ \leq \alpha \leq 90^\circ$ and is positive toward the body Z-axis. The angular rate is then computed by

$$\dot{\alpha} = (\alpha_c - \alpha_p) / \Delta t \quad (2-15)$$

where subscripts c and p indicate current and previous values. If $|\dot{\alpha}| > \dot{\alpha}_M$, then $\dot{\alpha} = \dot{\alpha}_M$ with proper sign and $\dot{\alpha}_M$ is TBD, where $\dot{\alpha}_M$ is the maximum solar array angular rate.

TRW is considering plans to fix the solar arrays at a particular cant angle and hold it fixed for each view period. Provision should therefore be made to input a specific angle and a flag used to bypass the computation of the cant angle and set its angular rate, $\dot{\alpha}$, to zero.

2.2.4.2 Center of Mass

The CM of B_j is given by

$$\vec{r}_j(t) = C_j^T (\vec{r}_j(t_0) - \vec{h}_j) + \vec{h}_j \quad (2-16)$$

where $j = 0, 1, 2, 3$; $[C_0] = I$ (unit matrix); and $[C_1]$, $[C_2]$, and $[C_3]$ represent the transformation matrices for the antenna and two solar arrays, respectively.

Then

$$\vec{R}_c = \frac{\sum_{j=0}^3 m_j \vec{r}_j(t)}{\sum_{j=0}^3 m_j} \quad (2-17)$$

and

$$\vec{p}_j = \vec{r}_j(t) - \vec{R}_c \quad (2-18)$$

2.2.4.3 Moment of Inertia

The total moment of inertia (MOI) tensor of the spacecraft, $[I_T]$, relative to axes parallel to the BCS frame and passing through \vec{R}_c is given by

$$[I_T] = \sum_{j=0}^3 [J_j] \quad (2-19)$$

where

$$[J_j] = [C_j]^T [J_j^*] [C_j] + m_j \left(\vec{\rho}_j^2 [I] - \vec{\rho}_j \circ \vec{\rho}_j^T \right) \quad (2-20)$$

where $[J_j^*]$ is the MOI tensor of B_j in the BCS frame but relative to the CM of B_j , and $\vec{\rho}_j \circ \vec{\rho}_j^T$ is an outer product.

2.2.4.4 Angular Momentum of Internal Motion

The angular momentum contributed by the solar array panels and the TDRS antenna is given by

$$\vec{L}_{INT} = \sum_{j=1}^3 ([J_j] + [K_j]) \vec{\omega}_j \quad (2-21)$$

where

$$[K_j] = m_j \left[\vec{\rho}_j \cdot (\vec{R}_c - \vec{h}_j) [I] - (\vec{R}_c - \vec{h}_j) \circ \vec{\rho}_j^T \right] \quad (2-22)$$

Hence, Equations (2-20) and (2-22) can be substituted into Equation (2-21) to obtain

$$\vec{L}_{INT} = \sum_{j=1}^3 \left\{ [C_j]^T [J_j^*] [C_j] + m_j \left[\vec{\rho}_j \cdot (\vec{\rho}_j + \vec{R}_c - \vec{h}_j) [I] - (\vec{\rho}_j + \vec{R}_c - \vec{h}_j) \circ \vec{\rho}_j^T \right] \right\} \vec{\omega}_j \quad (2-23)$$

where \vec{L}_{INT} is internal angular momentum.

2.2.5 EXTERNAL TORQUE COMPUTATIONS

2.2.5.1 Aerodynamic Torque Model

The atmospheric density for a given time and spacecraft position is computed by using the Jacchia model described in Reference 13. The values for the 10.7-centimeter (cm) flux and the geomagnetic index used in the Profile Subsystem are from the 95 percentile range of the Marshall Space Flight Center (MSFC) tables dated October 25, 1977.

The aerodynamic pressure, QOC, is computed from the following equation:

$$QOC = \frac{1}{2} (\text{RHO}) (\text{VMAG}^2) \quad (2-24)$$

where RHO is the atmospheric density at the current spacecraft position and VMAG is the magnitude of the spacecraft velocity from the spacecraft ephemeris.

To simplify the calculation of the solar and aerodynamic torques, the spacecraft is modeled as an assembly of cylinders, flat plates, and spheres. For each of the components, a surface unit normal vector and moment arm is needed. The unit normal for a flat plate is the vector perpendicular to the surface. The unit normal for a cylinder is the cylinder's axis of symmetry. A sphere has no unit normal. For a cylinder, the area entered is the cylinder height times the diameter. For a sphere or flat plate, the area entered is its surface area. The unit normal of each component is computed in the Profile Program if the component is Sun pointing or anti-Sun pointing, Earth pointing or anti-Earth pointing. This is done by picking up the spacecraft-to-Sun and spacecraft-to-Earth vectors from the ephemeris modules. If the component is rigidly fixed to the body, the unit normal in the spacecraft frame of reference

is input to the program. For the GRO mission, the latter option is used. The GRO spacecraft body model is described in Appendix B. Shadowing by the solar panels is not addressed.

The next step is to compute the angle of attack and projected area for each component. The angle of attack is defined as the angle between the wind vector and the unit normal vector.

$$\text{ANGATK} = \cos^{-1} [-(\widehat{\text{ANV}} \cdot \widehat{\text{WIND}})] \quad (2-25)$$

where ANGATK is the angle of attack, $\widehat{\text{ANV}}$ is the unit normal of a flat plate in spacecraft coordinates or the unit vector along a cylinder axis in spacecraft coordinates, and $\widehat{\text{WIND}}$ is the unit vector of the wind in spacecraft coordinates. This is the negative of the spacecraft velocity unit vector.

For a sphere, the angle of attack is assumed to be zero.

If the magnitude of the angle of attack is greater than 90 deg for a flat plate, the torque contribution for that element is set to zero.

The projected area is defined as the surface area the wind is hitting. For a sphere, the angle of attack has no meaning, and the projected area is calculated as

$$\text{PAREA} = \text{AREA}/4 \quad (2-26)$$

For a cylinder, the projected area is

$$\text{PAREA} = (\text{AREA}) \sin (\text{ANGATK}) \quad (2-27)$$

For a flat plate, the projected area is

$$\text{PAREA} = (\text{AREA}) \cos (\text{ANGATK}) \quad (2-28)$$

where PAREA is the projected area and AREA is the component's area (input). The aerodynamic force along $\widehat{\text{WIND}}$ is

$$\text{Force} = (\text{CD})(\text{QOC})(\text{PAREA}) \quad (2-29)$$

where CD is the aerodynamic drag coefficient (an input constant) (Reference 11, Section 17.2.3).

The torque on the spacecraft from each component is computed from

$$\vec{T} = \vec{R} \times \vec{F} \quad (2-30)$$

where \vec{R} is the vector to component in body coordinates (input) and \vec{F} is the force vector (= FORCE $\widehat{\text{WIND}}$).

To compute the total aerodynamic torque, the torque for each component is added vectorally.

2.2.5.2 Solar Pressure Torque Model

Before the solar pressure torque is computed, a check is performed to see if the spacecraft is in the Earth's shadow. This is done as follows:

$$\text{SHADOW} = (\hat{R} \cdot \widehat{\text{PSUN}}) - \left[1 - (R_e / \overline{\text{SCTE}})^2 \right]^{1/2} \quad (2-31)$$

where \hat{R} is the spacecraft-to-Earth unit vector (GCI coordinates), $\widehat{\text{PSUN}}$ is the spacecraft-to-Sun unit vector (GCI coordinates), R_e is the radius of the Earth (km), and $\overline{\text{SCTE}}$ is the spacecraft-to-Earth vector (km GCI).

If SHADOW is greater than zero, the spacecraft is in shadow and no solar pressure torque is computed. If it is less than zero, the spacecraft is in Sun, and the solar pressure torque is computed.

The model of the spacecraft as an assembly of simple geometric shapes, as described in the computation of the aerodynamic torque (Section 2.2.5.1), is also used to compute the solar pressure torque. First, the angle of attack is computed:

$$\theta = \text{SOLCOS} = \cos^{-1} (\widehat{\text{ANV}} \cdot \widehat{\text{SUN}}) \quad (2-32)$$

where SOLCOS is the angle of attack, $\widehat{\text{ANV}}$ is the unit normal of a flat plate in spacecraft coordinates or the unit vector along a cylinder axis in spacecraft coordinates, and $\widehat{\text{SUN}}$ is the unit vector from the spacecraft to the Sun in spacecraft coordinates.

For a sphere, the angle of attack is assumed to be zero. If the magnitude of the angle of attack for a flat plate is greater than 90 deg, the torque contribution for that element is set to zero.

Next, the solar radiation force is computed. For a flat plate, it is calculated as follows (Reference 11):

$$\vec{F} = (-\text{SOLC1}) (\cos \theta) (\text{AREA}) \{ (1 - \text{SOLC2}) \widehat{\text{SUN}} + 2[(\text{SOLC2} \cos \theta) + \text{SOLC3}] \widehat{\text{ANV}} \} \quad (2-33)$$

where SOLC1 is the solar radiation pressure, AREA is the flat plate's surface area (input value), SOLC2 is the coefficient of specular reflection, $\widehat{\text{SUN}}$ is previously defined, and SOLC3 is one-third the coefficient of diffuse reflection.

For a cylinder, the solar radiation force is computed by the following:

$$\vec{F} = (-\text{SOLC1})(\text{AREA}) \left\{ \left[\left(1 + \frac{\text{SOLC2}}{3} \right) \sin \theta + \frac{\pi}{2} \text{SOLC3} \right] \widehat{\text{SUN}} - \cos \theta \left(\frac{4}{3} \text{SOLC2} \sin \theta + \frac{\pi}{2} \text{SOLC3} \right) \widehat{\text{ANV}} \right\} \quad (2-34)$$

For a sphere, it is computed as follows:

$$\vec{F} = (-\text{SOLC1})(\text{AREA}) \left(\frac{1}{4} + \frac{\text{SOLC3}}{3} \right) \widehat{\text{SUN}} \quad (2-35)$$

The torque on the spacecraft from each component is

$$\vec{T} = \vec{R} \times \vec{F} \quad (2-36)$$

where \vec{R} is the vector to component in body coordinates (input). The torque for each component is added vectorially to obtain the total solar pressure torque.

2.2.5.3 Gravity-Gradient Torque

The gravity-gradient torque is computed from the following equation (Reference 11):

$$\vec{T} = \frac{3\mu}{R_s^3} (\widehat{R}_s \times [I] \widehat{R}_s) \quad (2-37)$$

where μ is the Earth's gravitational constant (3.986005×10^{14} cubic meters per squared second), R_s is the Earth-to-spacecraft distance, $[I]$ is the spacecraft moment of inertia tensor, and \widehat{R}_s is the unit vector from Earth to spacecraft in BCS coordinates.

2.2.5.4 Residual Dipole Magnetic Torque

The torque on the spacecraft due to a residual dipole is

$$\vec{T} = \vec{m} \times [A] \vec{B} \quad (2-38)$$

where \vec{m} is the residual dipole in BCS coordinates (an input value), [A] is the rotation matrix from GCI to BCS coordinates (see Section 2.2.1 for the method of computing [A]), and \vec{B} is the magnetic field vector in GCI coordinates from the magnetic field model (Section 2.2.3).

2.2.5.5 Resultant External Torque

The individual external torques are included in the output records of the Profile Subsystem to facilitate their inclusion or exclusion in the truth model if the option is exercised to compute some of them by using the actual spacecraft attitude instead of a nominal orientation.

2.2.6 MANEUVER SIMULATION

A nominal maneuver is modeled in the Profile Subsystem to provide a continuous definition of environmental torques over the interval of the maneuver. Being nominal, it does not represent the actual irregular motions of the spacecraft about its axes. Nor will it represent the overshoot and oscillations as the spacecraft settles in its new orientation. Nevertheless, it should provide a better representation than if it were omitted.

TRW, the prime contractor, currently plans to maneuver the spacecraft to effect an attitude change by a single Euler axis rotation, that is, by simultaneous rotations about each body axis.

To model the maneuver, the start time, t_s , and the end time, t_e , of the maneuver are required, as is the specification of the old and new spacecraft orientations, expressed

as quaternions \bar{q}_0 and \bar{q}_n , respectively. The quaternion \bar{q}_t that will map \bar{q}_0 to \bar{q}_n is obtained from

$$\bar{q}_t = \bar{q}_n * \bar{q}_0^{-1}$$

where * denotes quaternion multiplication and \bar{q}_0^{-1} is the conjugate with \bar{q}_0 in the form

$$\bar{q}_t = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \quad (2-39)$$

which is expanded to the form

$$\bar{q}_t = \begin{bmatrix} \alpha_1 \sin \frac{\theta}{2} \\ \alpha_2 \sin \frac{\theta}{2} \\ \alpha_3 \sin \frac{\theta}{2} \\ \cos \frac{\theta}{2} \end{bmatrix} \quad (2-40)$$

with $\theta = 2 \cos^{-1} q_4$, if $|q_4| \leq 0.707$ and

$$\theta = \sin^{-1} \sqrt{1 - q_4^2}$$

if $|q_4| > 0.707$, for better accuracy; and

$$\alpha_k = \frac{q_k}{\sin \frac{\theta}{2}} \quad k = 1, 2, 3$$

where $\alpha_1, \alpha_2, \alpha_3$ define the Euler axis and Θ the angle of positive rotation about that axis. Consider two cases:

Case 1

By default, it will be assumed that a simulated maneuver will be by the shortest positive rotation about the axis, i.e., $0 < \Theta \leq 180$. If $q_4 < 0$, ($\cos \Theta/2 < 0$) indicating $\Theta > 180$, the negative of \bar{q}_t is used, forcing $\Theta < 180$ and switching the positive direction of the Euler axis.

Case 2

A flag will be used to simulate a rotation greater than 180 deg (and less than 360 deg). Therefore, the negative of \bar{q}_t will be used if $q_4 > 0$ to force $\Theta > 180$ and to switch the positive direction of the Euler axis.

The angular rate, \dot{w} , is then obtained by

$$\dot{w} = \frac{\Theta}{T}$$

where $T = t_\Theta - t_s$.

Then the uniform angle of increment per time increment is

$$\Theta_\Delta = \dot{w}\Delta t$$

and Δt TBD with respect to total maneuver distance and time T . By default, Δt will equal 2 min but is left as a user option.

This then determines the quaternion at each Δt , i.e.,

$$\Delta \bar{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} \alpha_1 \sin \frac{\theta \Delta}{2} \\ \alpha_2 \sin \frac{\theta \Delta}{2} \\ \alpha_3 \sin \frac{\theta \Delta}{2} \\ \cos \frac{\theta \Delta}{2} \end{bmatrix} \quad (2-41)$$

Thus, the quaternion characterizing a slew about the Euler axis from \bar{q}_0 to \bar{q}_n at increments of time = Δt are defined

$$q_1 = \Delta \bar{q} * \bar{q}_0$$

$$q_2 = \Delta \bar{q} * q_1 = (\Delta \bar{q})^2 * \bar{q}_0$$

$$q_3 = (\Delta \bar{q})^3 * \bar{q}_0$$

⋮

$$q_n = (\Delta \bar{q})^n * \bar{q}_0$$

2.2.7 HIGH-GAIN ANTENNA

The high-gain antenna is used for tracking and communication purposes through the TDRS. The GRO software simulator is to be capable of tracking (pointing to) one of two TDRSs (east or west), switching on alternate passes. The angles defining antenna orientation are azimuth and elevation angles in the antenna coordinate system (ACS) (see Appendix A). The

antenna is capable of moving at a tracking rate of 0.06 deg/sec and at a slew rate of 2.81 deg/sec. Input requirements are specifications of which TDRS is to be tracked first and whether switching is to be done on successive orbits, as well as unit vectors to the TDRSs, T_E and T_W , in BCS coordinates. Also required is the cant angle, α , between the body X-axis and the boom, as well as the spacecraft altitude, h , and a unit vector to the Earth, \hat{R}_E .

Calculations are as follows:

1. Check for TDRS visibility. If $\theta > \theta_E$, TDRS is visible where $\theta = \hat{R}_E \cdot \hat{T}_i$ and $\hat{T}_i = \hat{T}_E$ or \hat{T}_W and $\theta_E = 71.44 \text{ deg} - 0.0244 (h - 350 \text{ km})$.

2. If TDRS is visible, transform its unit vector to ACS, as follows:

$$\hat{T}_{iA} = [R] \hat{T}_i \quad (2-42)$$

where

$$[R] = \begin{bmatrix} -\sin \alpha & 0 & -\cos \alpha \\ 0 & 1 & 0 \\ \cos \alpha & 0 & \sin \alpha \end{bmatrix} \quad (2-43)$$

3. Compute observations, as follows:

$$a = \tan^{-1} [T_{iY}/(-T_{iX})] \quad (2-44)$$

$$e = \cos^{-1} (T_{iZ}) \quad (2-45)$$

Azimuth is measured positive from the -X axis (ACS) going toward the +Y axis (ACS), and is bounded by mechanical gimbal stops to $-200 \leq a \leq 200 \text{ deg}$. Elevation is measured

positive from the +Z axis (ACS) going toward the -X axis (ACS) for an azimuth of 0, and is bounded by mechanical stops to $-110 \leq e \leq 110$ deg. In computing the pointing angles, use the antenna tracking rates and the respective gimbal limits.

4. If the current TDRS is occulted by the Earth and if switching is to be done, proceed as in item 2, but with the other TDRS unit vector, and use the antenna slewing rates. Antenna slewing to acquire a new TDRS should thus occur when the spacecraft is occulted by the Earth from the previous TDRS.

5. The antenna angular rates are computed as follows:

$$\dot{a} = (a_c - a_p) / \Delta t \quad (2-46)$$

$$\dot{e} = (e_c - e_p) / \Delta t \quad (2-47)$$

If $|\dot{a}| > \dot{a}_M$, $\dot{a} = \dot{a}_M$ with proper sign. If $|\dot{e}| > \dot{e}_M$, $\dot{e} = \dot{e}_M$ with proper sign. Subscripts c and p indicate current and previous values, Δt is the computing interval, and subscript M indicates the maximum rate for the tracking mode of the antenna.



SECTION 3 - SIMULATION CONTROL AND I/O SUBSYSTEM

The SCIO Subsystem for GRO will control the execution of the Truth Model and the OBC Model during simulation. The SCIO Subsystem will also be responsible for handling all communications with the user during simulation and will perform the following major functions:

- Initialize simulation parameters and allow the user to interactively set user input parameters and to input a ground command schedule
- Control simulation on the basis of a cycle time equal to the real-time cycle time and allow the user the option of inputting an overriding cycle time
- Send ground commands to the OBC Model on the basis of a command schedule
- Output analysis data generated by the Truth Model and the OBC Model during each simulation cycle

3.1 REQUIREMENTS

3.1.1 PROCESSING REQUIREMENTS

- R2.1.1 The SCIO Subsystem will control interaction and communication between the Truth Model, the OBC Model, and the user during simulation.
- R2.1.2 The SCIO Subsystem will control the execution of the Truth Model and the OBC Model on the basis of the real-time cycle time or a user-specified cycle time. One pass through the Truth Model and OBC Model will be one simulation cycle.
- R2.1.3 The SCIO Subsystem will send a user-specified time-ordered set of commands to the OBC Model.

R2.1.4 The SCIO Subsystem will provide the user with the following run time options during simulation:

- Start (restart) simulation
- Stop simulation
- Modify user input parameters
- Interrupt simulation at specified time and modify parameters as specified by user at run initiation

R2.1.4.1 The SCIO Subsystem will interrupt simulation, at the direction of the user, only at the end of a simulation cycle.

R2.1.4.2 The SCIO Subsystem will log all changes to user input parameters made during simulation to provide traceability.

R2.1.5 The SCIO Subsystem will always generate an initialization report containing the values of all user input parameters at the beginning of every simulation.

3.1.2 INPUT REQUIREMENTS

R2.2.1 The SCIO Subsystem will provide defaults for all user input parameters.

R2.2.2 The SCIO Subsystem will allow the user to interactively override the defaults for all user input parameters.

R.2.2.3 The SCIO Subsystem will allow the user to optionally input the following sets of parameters:

- Sensor parameters
- Initial conditions and integration parameters
- Simulation control parameters

- OBC Model parameters
 - Output control parameters
- R2.2.4 The SCIO Subsystem will, at a user-specified time and with user-specified values, modify any selected subset of the following:
- Sensor parameters
 - Control parameters
- R2.2.5 The SCIO Subsystem will allow the user to interactively create a set of time-scheduled ground commands for the OBC Model. Appendix D contains a list of the ground commands accepted by the SCIO Subsystem.
- R2.2.6 Simulations will be repeatable.
- R2.2.6.1 The SCIO Subsystem will store the user input parameters set and command schedule associated with each simulation case to allow repeatability.
- R2.2.6.2 The SCIO Subsystem will allow the user to use a prestored set of user input parameters and command schedules to initialize simulation.
- 3.1.3 OUTPUT REQUIREMENTS
- R2.3.1 The SCIO Subsystem will output simulation status information to the user terminal at a user-specified time interval.
- R2.3.2 The SCIO Subsystem will collect and store output data from the Truth Model and the OBC Model during simulation for later analysis after simulation has completed. These data will contain the following information:
- True state vector
 - Spacecraft position and velocity
 - Sensor status flags and output
 - Time tag

- OBC Model status and control voltages
- OBC Model attitude error

3.2 MATHEMATICAL SPECIFICATIONS

To obtain for output display purposes the spacecraft roll, pitch, and yaw errors, an Euler rotation sequence of 3-1-2 has been assumed. To date, no definitive word has been received on this matter. Unless such word is received, the following formulation is tentative, and the actual Euler rotation sequence should be considered TBD. Given the attitude matrix, [A], and the spacecraft body rate vector, $\vec{\omega}_B$, the spacecraft roll, pitch, and yaw errors are obtained as follows:

$$\text{roll error} = \phi = \sin^{-1} [A_{23}] \quad (3-1)$$

$$\text{pitch error} = \theta = \tan^{-1} [-A_{13}/A_{33}] \quad (3-2)$$

$$\text{yaw error} = \psi = \tan^{-1} [-A_{21}/A_{22}] \quad (3-3)$$

The spacecraft roll, pitch, and yaw error rates are then obtained as follows:

$$\text{roll error rate} = \dot{\phi} = \omega_x \cos \theta + \omega_z \sin \theta \quad (3-4)$$

$$\text{pitch error rate} = \dot{\theta} = \omega_y - \omega_u \tan \phi \quad (3-5)$$

$$\text{yaw error rate} = \dot{\psi} = \omega_u / \cos \phi \quad (3-6)$$

where ω_u is $\omega_z \cos \theta - \omega_x \sin \theta$.

SECTION 4 - TRUTH MODEL SUBSYSTEM

The Truth Model Subsystem will use the ephemeris data, external torques, and magnetic field information as well as the current spacecraft state to simulate or encode the response of the attitude hardware. The Truth Model Subsystem will communicate with the user through the simulation control and I/O portion (Section 3) and will be completely controlled by the SCIO Subsystem. The Truth Model Subsystem will perform the following functions:

- Decode spacecraft commands from the OBC Model Subsystem
- Effect actuators as commanded by the OBC Model Subsystem
- Update and interpolate the spacecraft ephemeris and environmental torques
- Integrate the spacecraft equations of motion with respect to time
- Generate true data at a specified rate for output

The derived software requirements and mathematical specifications for the Truth Model Subsystem are presented in the following sections.

4.1 REQUIREMENTS

4.1.1 PROCESSING REQUIREMENTS

R3.1.1 The Truth Model Subsystem will use profile data generated by the Profile Subsystem.

R3.1.1.1 The Truth Model Subsystem will interpolate the profile data, using a five-point Lagrangian interpolator when the data intervals are larger than the simulation time step (cycle time).

- R3.1.2 On option, the environmental torque profile will be calculated in the Truth Model Subsystem.
- R3.1.3 The Truth Model Subsystem will compute torques resulting from OBC Model Subsystem commands.
- R3.1.4 The Truth Model Subsystem will compute the net external torque on the spacecraft.
- R3.1.5 The Truth Model Subsystem will integrate the spacecraft attitude equations of motion by using a fifth-order, variable-step Adams-Moulton-Bashforth (AMB) predictor-corrector.
- R3.1.6 The Truth Model Subsystem will model the attitude sensors to provide realistic sensor data for input to the OBC Model Subsystem.
- R3.1.7 The Truth Model Subsystem will prepare true data for standard output.
- R3.1.8 The Truth Model Subsystem will operate under the control of the SCIO Subsystem.
- R3.1.9 In the Truth Model Subsystem, all coordinate systems will be right-handed orthogonal systems.
- R3.1.10 The Truth Model Subsystem will perform all internal computations in the mks system of units and will express all angular quantities in radians.
- R3.1.11 The Truth Model Subsystem will be designed and built to facilitate incorporation into the Goddard GRO Simulator (GGS).
- R3.1.12 On option, the Truth Model Subsystem will calculate the effect on angular movement caused by the scanning motion of the two tape recorders and the OSSE payload.

4.1.2 INPUT REQUIREMENTS

R3.2.1 For each time step, the Truth Model Subsystem will obtain the following profile data from the Profile Subsystem:

- Earth's magnetic field
- Unit vector from spacecraft to Sun
- Vectors from spacecraft to Earth and Moon
- HGAD pointing angles
- Velocity vector
- Total torque vector
- Moment of inertia tensor
- Star unit vectors

R3.2.2 The Truth Model Subsystem will receive the following user input parameters from the SCIO Subsystem:

- Initial attitude angles and rates
- Sensor errors, biases, and misalignments
- Sensor noise switch and noise specification (that is, mean and standard deviation)
- Maneuver indicator
- Thruster information (performance indicator, configuration, misalignment)
- Magnetic torquer and reaction wheel information (performance indicator, configuration, misalignment)
- Flags to include gravity-gradient torque and magnetic dipole moment
- Integration stepsize (maximum)
- Star tracker information

- R3.2.3 The Truth Model Subsystem will receive the following control information from the SCIO Subsystem: current time.
- R3.2.4 The Truth Model Subsystem will accept the following commands from the OBC Model Subsystem:
- To change the rate of rotation of the reaction wheels
 - To change pointing angles of HGAD and solar array
 - To charge or change the polarity of the magnetic torque bars
 - To execute the thrusters (that is, to burn the thrusters); if thruster burn is commanded, the command will specify which thrusters are to be turned on or off and the duration of the burn
 - To modify the current state attitude of the attitude hardware (turn off/on gyro, switches, etc.)
- R3.2.5 The Truth Model Subsystem will receive all input parameter uncertainties as standard deviations.

4.1.3 OUTPUT REQUIREMENTS

- R3.3.1 The Truth Model Subsystem will output the following true data at every time step to the SCIO Subsystem for inclusion in the output data set:
- True attitude (quaternions)
 - True body rates
 - True wheel speeds
 - True Sun unit vector
 - Spacecraft position and velocity
 - Sun sensor angles

R3.3.2 The Truth Model Subsystem will output the following information to the OBC Model Subsystem at each time step:

- Star tracker status
- IRU status
- Sun sensor status
- IRU rates
- Wheel speeds
- Magnetometer measurements
- HGA angles

4.2 MATHEMATICAL SPECIFICATIONS

4.2.1 DYNAMICS INTEGRATOR

4.2.1.1 Adams-Moulton-Bashforth Integrator

The equations of motion for the GRO spacecraft may be written in the compact form

$$\vec{y}' = \frac{d\vec{y}}{dt} = \vec{f}(\vec{y}(t), t) \quad (4-1)$$

The simulator within the Truth Model Subsystem employs an AMB integrator to solve the above system of differential equations at a discrete set of points in time, given a set of initial conditions. The method of solution developed here is sometimes referred to as a predictor-corrector.

The fifth-order AMB predictor at the nth step uses the current value and four previous values of the function to predict the state at the (n + 1) step, i.e.,

$$y_{n+1}^{(p)} = y_n + h \left(1 + \frac{1}{2} \nabla + \frac{5}{12} \nabla^2 + \frac{3}{8} \nabla^3 + \frac{251}{720} \nabla^4 \right) y_n' \quad (4-2)$$

where h is the stepsize and ∇ is a backward difference operator such that

$$y'_n = f(y(t_n), t_n)$$

$$\nabla y'_n = y'_n - y'_{n-1}$$

$$\nabla^2 y'_n = y'_n - 2y'_{n-1} + y'_{n-2} \quad (4-3)$$

$$\nabla^3 y'_n = y'_n - 3y'_{n-1} + 3y'_{n-2} - y'_{n-3}$$

$$\nabla^4 y'_n = y'_n - 4y'_{n-1} + 6y'_{n-2} - 4y'_{n-3} + y'_{n-4}$$

(The vector notation has been dropped for convenience.) The derivative is then evaluated at step $(n + 1)$

$$y'_{n+1} = f\left(y_{n+1}^{(p)}, t_{n+1}\right) \quad (4-4)$$

and the AMB corrector uses the current difference and three previous differences to correct the function as follows:

$$\begin{aligned} y_{n+1}^{(c)} &= y_n + \frac{h}{720} (469 + 109\nabla + 49\nabla^2 + 19\nabla^3) y'_n + \frac{251}{720} h y'_{n+1} \\ &= y_{n+1}^{(p)} - \frac{251}{720} \cdot h \left[\left(\sum_{i=0}^4 \nabla^i \right) y'_n - y'_{n+1} \right] \end{aligned} \quad (4-5)$$

4.2.1.2 Derivation of AMB Equations

The basic concept of the Adams integration is to integrate forward, using the value of the function at the last integrated point and the value of the derivatives at that point

and previous points. The value y_{n+1} is desired in terms of y_n and $y'_n, y'_{n-1}, y'_{n-2},$ etc. The derivatives could be represented by backward differences as $y'_n, \nabla y'_n, \nabla^2 y'_n,$ etc.

Before expressions are derived for the predictor or corrector equations, expressions for two operators, E and hD, are derived. By definition,

$$\nabla y_{n+1} = y_{n+1} - y_n$$

so that

$$y_n = (1 - \nabla) y_{n+1}$$

or

$$y_{n+1} = (1 - \nabla)^{-1} y_n \quad (4-6)$$

Equation (4-6) can be rewritten as an operator equation,

$$y_{n+1} = E y_n \quad (4-7a)$$

where, by definition, $E = (1 - \nabla)^{-1}$. This relation shows that the operator E can be expressed in terms of backward differences as

$$\begin{aligned} E &= (1 - \nabla)^{-1} = 1 + \nabla + \nabla^2 + \dots + \nabla^n + \dots \\ &= \sum_{k=0}^{\infty} \nabla^k \end{aligned} \quad (4-7b)$$

The operator D is defined as equivalent to d/dt . Thus,
 $y'(t) = Dy(t)$.

Expressing $y(t)$ in terms of a Taylor's series using the notation of the operator D gives

$$\begin{aligned}
 y(t) &= y(0) + tD y(0) + \frac{t^2}{2!} D^2 y(0) + \dots + \frac{t^n}{n!} D^n y(0) + \dots \\
 &= \sum_{k=0}^{\infty} \left[\frac{(tD)^k}{k!} y(0) \right] \qquad (4-8a)
 \end{aligned}$$

or

$$\begin{aligned}
 y(t) &= \left(1 + tD + \frac{t^2}{2!} D^2 + \dots + \frac{t^n}{n!} D^n + \dots \right) y(0) \\
 &= \sum_{k=0}^{\infty} \left[\frac{(tD)^k}{k!} \right] y(0) \qquad (4-8b)
 \end{aligned}$$

and, finally,

$$y(t) = \text{EXP}(tD) y(0) \qquad (4-8c)$$

where the usual definition of the exponential operator has been used.

If $y_n \equiv y(t)$, Equation (4-8b) may be used to obtain a convenient expression for y_{n+1} ,

$$y_{n+1} \equiv y(t + h) = \text{EXP}(hD) \cdot y(t) = \text{EXP}(hD) \cdot y_n \qquad (4-9)$$

Combining Equations (4-6) and (4-9) gives

$$e^{hD} y_n = (1 - \nabla)^{-1} y_n$$

or, in operator form,

$$e^{hD} = (1 - \nabla)^{-1} \quad (4-10)$$

Hence, the operator hD can be written in terms of backward differences as

$$hD = \ln [(1 - \nabla)^{-1}] \quad (4-11a)$$

or

$$hD = -\ln (1 - \nabla) = \nabla + \frac{\nabla^2}{2} + \dots + \frac{\nabla^n}{n} + \dots \quad (4-11b)$$

$$= \sum_{k=1}^{\infty} \frac{\nabla^k}{k}$$

The predictor and corrector formulas can be derived using the operators E and hD . From Equation (4-7),

$$Y_{n+1} = E Y_n$$

$$Y_{n+1} = E Y_n + Y_n - Y_n$$

$$Y_{n+1} = Y_n + (E - 1) Y_n$$

or

$$Y_{n+1} = Y_n + (E - 1) \frac{hD}{hD} Y_n$$

and, finally,

$$Y_{n+1} = Y_n + \frac{h(E - 1)}{hD} Y_n' \quad (4-12)$$

Using Equations (4-7) and (4-11b), Equation (4-12) becomes

$$Y_{n+1} = Y_n + h \left(\frac{\nabla + \nabla^2 + \nabla^3 + \dots + \nabla^n + \dots}{\nabla + \frac{\nabla^2}{2} + \frac{\nabla^3}{3} + \dots + \frac{\nabla^n}{n} + \dots} \right) Y_n' \quad (4-13)$$

The division may be carried out to yield any desired number of terms. Thus,

$$Y_{n+1} = Y_n + h \left(1 + \frac{1}{2} \nabla + \frac{5}{12} \nabla^2 + \frac{3}{8} \nabla^3 + \frac{251}{720} \nabla^4 + \frac{95}{288} \nabla^5 + \frac{19087}{60480} \nabla^6 + \frac{5257}{17280} \nabla^7 + \dots \right) Y_n' \quad (4-14)$$

The predictor at the nth step, Equation (4-2), is obtained by truncating Equation (4-14) to six terms, i.e.,

$$Y_{n+1}^{(p)} = Y_n + h \left(1 + \frac{1}{2} \nabla + \frac{5}{12} \nabla^2 + \frac{3}{8} \nabla^3 + \frac{251}{720} \nabla^4 \right) Y_n' \quad (4-15)$$

The corrector equation is computed as follows:

$$Y_{n+1} = Y_{n+1} - Y_n + Y_n = Y_n + \nabla Y_{n+1} \quad (4-16a)$$

or

$$Y_{n+1} = Y_n + \nabla \left(\frac{hD}{hD} \right) Y_{n+1} \quad (4-16b)$$

or

$$Y_{n+1} = Y_n + h \left(\frac{\nabla}{hD} \right) Y'_{n+1} \quad (4-16c)$$

Using Equation (4-11b), it follows that

$$Y_{n+1} = Y_n + h \frac{\nabla}{\nabla + \frac{\nabla^2}{2} + \frac{\nabla^3}{3} + \dots} Y'_{n+1} \quad (4-16d)$$

Expanding Equation (4-16d) yields

$$Y_{n+1} = Y_n + h \left(1 - \frac{1}{2} \nabla - \frac{1}{12} \nabla^2 - \frac{1}{24} \nabla^3 - \frac{19}{720} \nabla^4 - \frac{3}{160} \nabla^5 - \frac{863}{60480} \nabla^6 - \frac{275}{24192} \nabla^7 - \dots \right) Y'_{n+1} \quad (4-17)$$

To obtain Equation (4-5), Equation (4-17) is truncated to six terms, i.e.,

$$Y_{n+1} = Y_n + h \left(1 - \frac{1}{2} \nabla - \frac{1}{12} \nabla^2 - \frac{1}{24} \nabla^3 - \frac{19}{720} \nabla^4 \right) Y'_{n+1} \quad (4-18)$$

After expanding the backward differences (Equation (4-3)), substituting these into Equation (4-18), and simplifying,

the corrector at the nth step is given by the following relation:

$$\begin{aligned}
 y_{n+1}^{(c)} &= y_n + \frac{h}{720} \left(469 + 109\nabla + 49\nabla^2 + 19\nabla^3 \right) y_n' \\
 &\quad + \frac{251}{720} h y_{n+1}' \\
 &= y_{n+1}^{(p)} - \frac{251}{720} h \left[\left(\sum_{i=0}^4 \nabla^i \right) y_n' - y_{n+1}' \right] \quad (4-19)
 \end{aligned}$$

4.2.1.3 Changing Interval Size

To maintain accuracy between preset bounds, doubling and halving the interval size are often necessary. Because the integration is performed by using a difference table, a new difference table must be constructed in terms of the old difference table. Let the parameter p be defined by the ratio

$$p = \frac{\text{new interval}}{\text{old interval}} \quad (4-20)$$

For example, if the interval is halved, $p = 1/2$; if it is doubled, $p = 2$.

If $\nabla_p y_n'$ equals the first partial difference in terms of the new interval,

$$\nabla_p y_n' = y_n' - y_{n-p}' = y_n' - E^{-p} y_n' = (1 - E^{-p}) y_n' \quad (4-21a)$$

or

$$\nabla_p y_n' = [1 - (1 - \nabla)^p] y_n' \quad (4-21b)$$

and, finally,

$$\nabla_p y'_n = \left[p\nabla - \frac{p(p-1)}{2!} \nabla^2 + \frac{p(p-1)(p-2)}{3!} \nabla^3 - \dots \right] y'_n \quad (4-21c)$$

The mth partial difference is readily obtained as

$$\nabla_p^m y'_n = \left[1 - (1 - \nabla)^p \right]^m y'_n \quad (4-22a)$$

or

$$\nabla_p^m y'_n = \left[p\nabla - \frac{p(p-1)}{2!} \nabla^2 + \frac{p(p-1)(p-2)}{3!} \nabla^3 - \dots \right]^m y'_n \quad (4-22b)$$

The cases of doubling and halving are frequently encountered and may be represented, in abbreviated form, by a matrix multiplication (Equations (4-27) and (4-28)).

4.2.1.4 Partial Step Equation

An equation for integrating a partial step forward using backward differences is developed here by using operator methods. Let $p = (t_{n+p} - t_n)/h$. Then,

$$\begin{aligned} Y_{n+p} &= E^p Y_n = E^p Y_n - Y_n + Y_n \\ &= y_n + (E^p - 1) y_n \\ &= y_n + (E^p - 1) \frac{hD}{hD} Y_n \end{aligned} \quad (4-23)$$

where E and hD are previously defined. Thus,

$$Y_{n+p} = Y_n + h \left[\frac{(E^p - 1)}{hD} \right] Y'_n \quad (4-24)$$

This is evaluated in terms of backward differences as follows:

$$Y_{n+p} = Y_n + h \left[\frac{(1 - \nabla)^{-p} - 1}{-\ln(1 - \nabla)} \right] Y'_n \quad (4-25a)$$

or

$$Y_{n+p} = Y_n + h \left[\frac{p\nabla + \frac{p(p+1)}{2!} \nabla^2 + \frac{p(p+1)(p+2)}{3!} \nabla^3 + \dots}{\nabla + \frac{\nabla^2}{2} + \frac{\nabla^3}{3} + \frac{\nabla^4}{4} + \dots} \right] Y'_n \quad (4-25b)$$

Equation (4-25b) is evaluated to the desired number of differences:

$$\begin{aligned} Y_{n+p} = Y_n + h & \left[\left(pY'_n + \frac{p^2}{2} \nabla Y'_n + \frac{2p^3 + 3p^2}{12} \nabla^2 Y'_n \right) \right. \\ & + \left(\frac{p^4 + 4p^3 + 4p^2}{24} \right) \nabla^3 Y'_n \\ & + \left(\frac{6p^5 + 45p^4 + 110p^3 + 90p^2}{720} \right) \nabla^4 Y'_n \\ & \left. + \left(\frac{2p^6 + 24p^5 + 105p^4 + 200p^3 + 144p^2}{1440} \right) \nabla^5 Y'_n \right] \end{aligned} \quad (4-26)$$

4.2.1.5 Stepsize Considerations

Throughout any given simulation, the integration stepsize will change constantly. It will be decreased to satisfy certain accuracy requirements but increased to save computer

time spent for arithmetic calculations. There are some restrictions on how much and when the current stepsize can be changed.

For GRO applications, there will be a maximum integration stepsize of 1000 milliseconds (1 second), the frequency at which the control torques are to be updated. During their application, the control torques will be constant and will significantly affect the dynamics of the spacecraft. Hence, to ensure an accurate simulation, the maximum stepsize will be used.

The stepsize will be halved or doubled for two reasons. First, recomputation of the finite differences is simplified. Second, halving or doubling the current stepsize only at certain times ensures that the integration will terminate at a time when the control torques are updated. Two step-sizes will be used:

- H--For GRO, this is equal to the minor frame time. Assume that a minor frame begins at the time the control torques are applied.
- h_c --This is the current stepsize.
--If there is a maximum stepsize,

$$h_c \leq H \text{ and } h_c = \frac{H}{2^K}; K = 0, 1, \dots, 30$$

--If there is no maximum stepsize,

$$h_c = \frac{H}{2^K} \text{ where } K = 0, \pm 1, \pm 2, \dots, \pm 30$$

Thus, if a minor frame begins with a stepsize of $H/2^K$, there are 2^K integration steps remaining in the minor frame.

There is also one other restriction. If there are 2^K steps remaining, 2^K must be less than or equal to $2^{31} - 1$, because $2^{31} - 1$ is the largest integer that can be stored in an I*4 variable. If $2^K \geq 2^{31} - 1$, stepsize halving is terminated, even if the accuracy requirements have not been met, and the simulation continues.

In the case of halving and doubling stepsizes, let

$$\begin{bmatrix} \nabla y' \\ \nabla^2 y' \\ \nabla^3 y' \\ \nabla^4 y' \end{bmatrix}$$

be the backward differences for the current stepsize, h_c . Then, if h_c is to be halved, the new divided differences $\nabla_{1/2}^i y'$ are given by

$$\begin{bmatrix} \nabla_{1/2} y' \\ \nabla_{1/2}^2 y' \\ \nabla_{1/2}^3 y' \\ \nabla_{1/2}^4 y' \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{8} & \frac{1}{16} & \frac{5}{128} \\ 0 & \frac{1}{4} & \frac{1}{8} & \frac{5}{64} \\ 0 & 0 & \frac{1}{8} & \frac{3}{32} \\ 0 & 0 & 0 & \frac{1}{16} \end{bmatrix} \begin{bmatrix} \nabla y' \\ \nabla^2 y' \\ \nabla^3 y' \\ \nabla^4 y' \end{bmatrix} \quad (4-27)$$

If h_c is to be doubled, the new differences $\nabla_2^i y'$ are

$$\begin{bmatrix} \nabla_2 y' \\ \nabla_2^2 y' \\ \nabla_2^3 y' \\ \nabla_2^4 y' \end{bmatrix} = \begin{bmatrix} 2 & -1 & 0 & 0 \\ 0 & 4 & -4 & 1 \\ 0 & 0 & 8 & -12 \\ 0 & 0 & 0 & 16 \end{bmatrix} \begin{bmatrix} \nabla y' \\ \nabla^2 y' \\ \nabla^3 y' \\ \nabla^4 y' \end{bmatrix} \quad (4-28)$$

4.2.1.6 Equations of Motion

The attitude state vector for dynamics simulation will be a 11-component vector, X , consisting of the following components:

$$X_1 = q_1$$

$$X_2 = q_2$$

$$X_3 = q_3$$

$$X_4 = q_4$$

$$X_5 = L_{T_1}$$

$$X_6 = L_{T_2}$$

$$X_7 = L_{T_3}$$

$$X_8 = L_{mw_1}$$

$$X_9 = L_{mw_2}$$

$$X_{10} = L_{mw_3}$$

(4-29)

$$X_{11} = L_{mw_4}$$

where $\bar{q} = (q_1, q_2, q_3, q_4)^T$ is the attitude quaternion representing the transformation from inertial to body coordinates

$\vec{L}_T = (L_{T_1}, L_{T_2}, L_{T_3})^T$ is the total spacecraft angular momentum in Joule-seconds in body coordinates

L_{mw_i} = angular momentum of the momentum wheels in Joule-seconds along the wheel axes

The equations of motion define the time rate of change of the components of the state vector. The equations of motion are as follows:

$$\dot{\bar{q}} = \frac{1}{2} [\Omega] \bar{q} \quad (4-30)$$

$$\dot{\vec{L}}_T = \vec{N}_e + \vec{L}_T \times \vec{\omega}_B \quad (4-31)$$

$$\dot{L}_{mw_1} = N_{mw_1} \quad (4-32)$$

$$\dot{L}_{mw_2} = N_{mw_2} \quad (4-33)$$

$$\dot{L}_{mw_3} = N_{mw_3} \quad (4-34)$$

$$\dot{L}_{mw_4} = N_{mw_4} \quad (4-35)$$

where

$$[\Omega] = \begin{bmatrix} 0 & \omega_3 & -\omega_2 & \omega_1 \\ -\omega_3 & 0 & \omega_1 & \omega_2 \\ \omega_2 & -\omega_1 & 0 & \omega_3 \\ -\omega_1 & -\omega_2 & -\omega_3 & 0 \end{bmatrix} \quad (4-36)$$

$\vec{\omega}_B = (\omega_1, \omega_2, \omega_3)$ = body angular velocity (radians per second) in body coordinates

\vec{N}_e = net external torque on the spacecraft newton-meters) ($N \cdot m$) in body coordinates

N_{mw_i} = torque on the i th momentum wheel ($N \cdot m$) applied along the wheel axis

The body angular velocity is computed as follows:

$$\vec{\omega}_B = \left[I_T \right]^{-1} \cdot \vec{L}_B \quad (4-37)$$

where $\left[I_T \right]^{-1}$ = inverse total spacecraft moment of inertial tensor ($kg \cdot m^2$) in body coordinates

\vec{L}_B = body angular momentum

The body angular momentum is computed as follows:

$$\vec{L}_B = \vec{L}_T - \vec{L}_{INT} \quad (4-38)$$

where \vec{L}_{INT} = total angular momentum due to motion of paths internal to the spacecraft and

$$\vec{L}_{INT} = \vec{L}_{mw} + \vec{L}_{Si} + \vec{L}_{TR} + \vec{L}_{HGA} + \vec{L}_{SA}$$

and $\vec{L}_{mw} = \vec{L}_{mw_1} + \vec{L}_{mw_2} + \vec{L}_{mw_3} + \vec{L}_{mw_4}$

\vec{L}_{Si} = angular momentum of scanning instrument, if used

\vec{L}_{TR} = angular momentum of the tape recorders, if used

\vec{L}_{HGA} = angular momentum attributable to the motion of the high-gain antenna

\vec{L}_{SA} = angular momentum attributable to the motion of the solar arrays

The use of effects of scanning instruments such as the OSSE and the tape recorders should be user selected as an option.

The wheel speeds are then obtained from the following:

$$\vec{\omega}_{mw_i} = \vec{L}_{mw_i} \left[I_{mw_i} \right]^{-1} \quad i = 1, 2, 3, 4$$

where $\left[I_{mw_i} \right]$ is the moment of inertia for each wheel.

4.2.2 SIMULATOR INTERPOLATION

From data available only at discrete times, a five-point Lagrangian interpolating polynomial is evaluated within the Truth Model Subsystem. As a simulation progresses in time, the five grid points for interpolation change to ensure that the request time lies between at least two of the grid points during a simulation. If the request time lies outside the data available for the grid, an error flag is set, and simulation is terminated.

The general form of an n-point Lagrangian interpolating polynomial is as follows:

$$p_n(t) = \sum_{i=1}^n L_i(t) \cdot f_i \quad (4-39)$$

where

$$L_i(t) = \prod_{\substack{j=1 \\ j \neq i}}^n \frac{(t - t_j)}{(t_i - t_j)} \quad (4-40)$$

and where the ordered pairs (t_i, f_i) $i = 1, \dots, n$ (Figure 4-1) define data generated within the Profile Subsystem (Section 2) that will be interpolated, and f_i is a grid coordinate, such as the x-component of the Sun vector in GCI coordinates at time t .

Equations (4-39) and (4-40) can be used to expand $p_n(t)$ with respect to t :

$$p_n(t) = \sum_{i=0}^n B_n \cdot t^i \quad (4-41)$$

where $B_n = B_n(t_i, f_i)$. The form of the interpolator in Equation (4-41) is used in the simulator because the time (2 min) between two consecutive grid points is very large compared with the maximum integration interval (1.0 sec). If the values of B_n are computed for each 2-min interval and used in Equation (4-41), less time is expended compared with the use of Equations (4-39) and (4-40) over that same interval. However, if the value of t is very large (10^9 , as in the simulator), an overflow will occur when using Equation (4-41). To avoid this problem, the request times are scaled before the interpolating polynomial is evaluated.

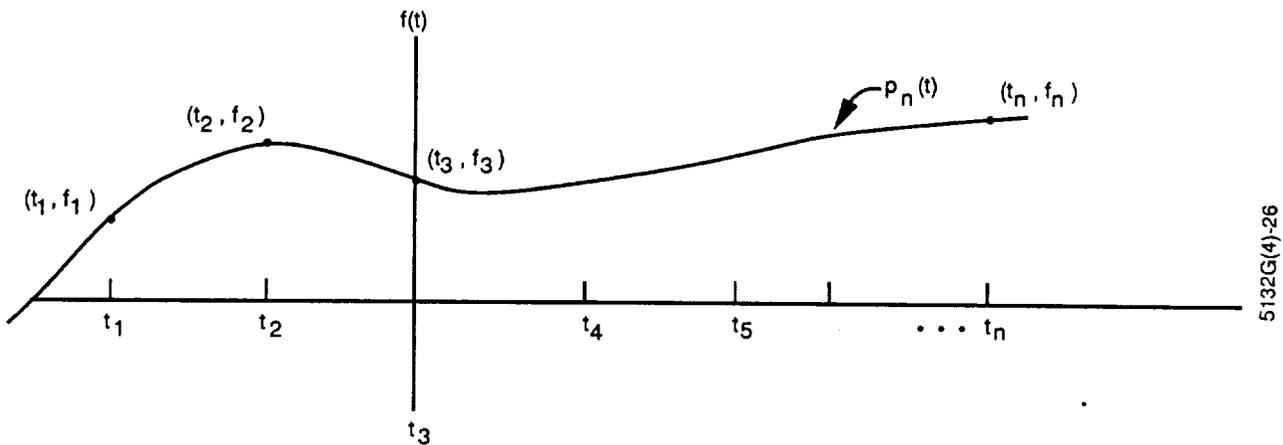


Figure 4-1. General Form of an n-Point Lagrangian Interpolating Polynomial

Suppose that data are requested at time t and that the current $n = 5$ grid points are (t_i, f_i) , with $i = 1$ to 5 , and $t_1 < t < t_5$. The five-point Lagrange interpolating polynomial $P_5^*(t_s)$ is given by

$$P_5^*(t_s) = \sum_{i=1}^5 L_i^*(t_s) \cdot f_i \quad (4-42)$$

where t_s is the scaled request time

$$t_s = \frac{t - t_3}{\Delta t} \quad |t_s| \leq 2 \quad (4-43)$$

where t_3 is the middle value of the time coordinates t_1 to t_5 , inclusive; Δt is the grid point spacing presently defined such that $\Delta t = t_{j+1} - t_j$, with $j = 1$ to 4 , and is computed

by the software every time the grid points are updated (presently $\Delta t = 2$ minutes); and

$$L_i^*(t_s) = \prod_{\substack{j=1 \\ i \neq j}}^5 \frac{t_s - t_j^{(s)}}{t_i^{(s)} - t_j^{(s)}} \quad (4-44)$$

The scaled grid points are given by

$$t_j^{(s)} = \frac{t_j - t_3}{\Delta t} = -2, -1, 0, 1, 2 \quad (4-45a)$$

$j = 1, \dots, 5$

$$t_i^{(s)} = \frac{t_i - t_3}{\Delta t} = -2, -1, 0, 1, 2 \quad (4-45b)$$

$i = 1, \dots, 5$

The equivalence of Equations (4-39) and (4-42) is proven as follows:

$$P_5^*(t_s) = \sum_{i=1}^5 \left(\prod_{\substack{j=1 \\ i \neq j}}^5 \frac{t_s - t_j^{(s)}}{t_i^{(s)} - t_j^{(s)}} \right) \cdot f_i \quad (4-46a)$$

$$P_5^*(t_s) = \sum_{i=1}^5 \left(\prod_{\substack{j=1 \\ i \neq j}}^5 \frac{\frac{t - t_3}{\Delta t} - \frac{t_j - t_3}{\Delta t}}{\frac{t_i - t_3}{\Delta t} - \frac{t_j - t_3}{\Delta t}} \right) \cdot f_i \quad (4-46b)$$

$$= \sum_{i=1}^5 \left(\prod_{\substack{j=1 \\ i \neq j}}^5 \frac{t - t_j}{t_i - t_j} \right) \cdot f_i \quad (4-46c)$$

$$= \sum_{i=1}^5 L_i(t) \cdot f_i \quad (4-46d)$$

The scaling, however, will ensure that $|t_s| \leq 2$. The time, t , in Equation (4-39) is of the order of 10^9 . A loss of significant digits will occur if Equation (4-39) is evaluated as is.

Expanding Equation (4-42) with respect to t_s and grouping like terms yields

$$P_5^*(t_s) = \sum_{i=0}^4 A_i t_s^i \quad (4-47)$$

where

$$A_0 = f_3$$

$$A_1 = \frac{(f_1 - 8f_2 + 8f_4 - f_5)}{12}$$

$$A_2 = \frac{(-f_1 + 16f_2 - 30f_3 + 16f_4 - f_5)}{24} \quad (4-48)$$

$$A_3 = \frac{(-f_1 + 2f_2 - 2f_4 + f_5)}{12}$$

$$A_4 = \frac{f_1 - 4f_2 + 6f_3 - 4f_4 + f_5}{24}$$

The numerical coefficients in each expression for A_i are independent of Δt .

4.2.3 INTERNAL MOTION

There are two instruments on the GRO spacecraft that affect angular movement and may therefore be included in a simulation at user option. These are the two tape recorders and the OSSE payload instrument. Both are modeled as scan type instruments.

The angular momentum attributable to a scanning instrument along its scan axis is modeled as represented in Figure 4-2.

The angular momentum attributable to a scanning instrument, \vec{L}_{S_i} , along its scan axis, \hat{U}_{SB_i} , expressed in body coordinates is given by

$$\vec{L}_{S_i} = (S) \left(L_{S_i} \right) \left(U_{SB_i} \right)$$

where $S = \begin{cases} +1 & \text{if } X \text{ is odd} \\ -1 & \text{if } X \text{ is even} \end{cases}$

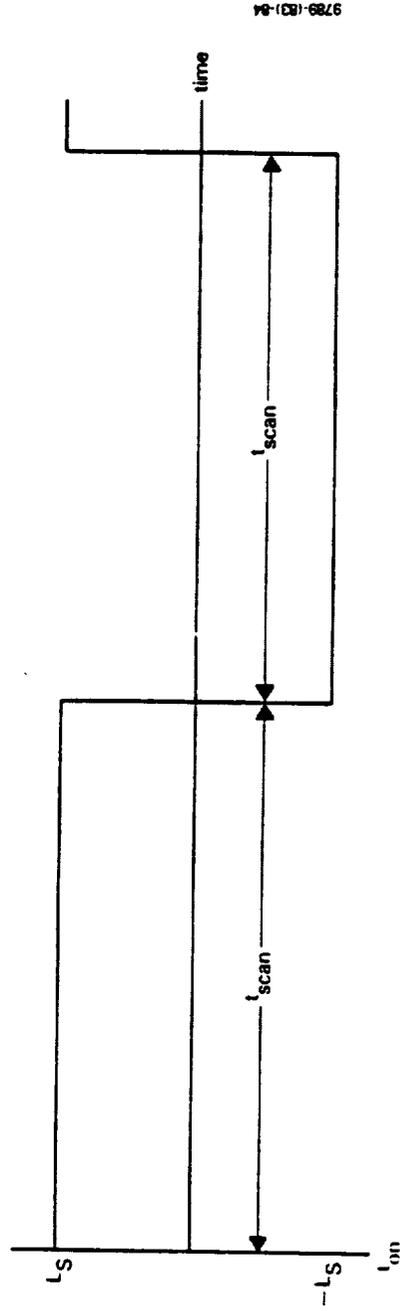
and $X = \text{MOD} (t_i - t_{on}, t_{scan}) + 1$.

\vec{L}_{S_i} is the angular momentum for each of the tape recorders and the OSSE and is TBD.

\hat{U}_{S_i} is the scan unit vector for each of the tape recorders and is TBD; \hat{U}_S for the OSSE is $[0 \ 1 \ 0]^T$; t_0 is the time the instrument is turned on, t_{scan} is the length of time of one complete scan, and t_i is the time for which the angular momentum is being calculated.

4.2.4 SENSOR MODELS

The attitude sensor models for IRU, FSS, CSS, FHST, wheel tachometer, and TAM are described in the following subsections.



9789-1031-94

Figure 4-2. Angular Momentum Versus Time for a Scanning Instrument

4.2.4.1 IRU Model

The IRU consists of three rate-integrating gyros and has six channels that provide measurements of angular displacement along the three spacecraft body axes. One channel is primary and one is backup for each axis. For each channel, gyro data generation involves the following two steps: calculation of angular displacement and modeling gyro noise to add on to the angular displacement.

4.2.4.1.1 Angular Displacement Calculation

Calculate angular displacement as follows:

1. Input angular velocity vector, $\vec{\omega}$.
2. Project $\vec{\omega}$ along the channel input axis, \vec{G} , to get rate, r , measured by that channel (i.e., $r = \vec{G} \cdot \vec{\omega}$) in radians per second.
3. Compute angular displacement by integrating rate.
 - a. If spacecraft is inertially fixed (low rate mode),

$$\Delta\theta = (r + b) \Delta t$$

where $\Delta\theta$ = angular displacement in radians

r = rate (computed in step 2) in radians per second

b = component of rate bias along axis in radians per second

Δt = gyro cycle time in seconds

- b. If maneuver is under way (high rate mode), the integrator described in Section 4.2.1 should be used to integrate $(r + b)$. Two successive values of θ are then differenced to get $\Delta\theta$, i.e., $\Delta\theta = \theta(n\Delta t) - \theta((n - 1) \Delta t)$.

4.2.4.1.2 Gyro Noise Modeling

The gyro noise in GRO comes from two sources:

1. Measurement noise (n_v) on angular rate. This is modeled as once-integrated white Gaussian noise and can be modeled as a sequence of random variables that are (a) Gaussian, (b) zero mean, and (c) variance of $\sigma_v^2 \Delta t$. The value of such a random variable is added to $\Delta\theta$ at each step, i.e., $\Delta\theta_v = \sigma_v \sqrt{\Delta t} \cdot Z$, where Z is standard normal random variable.
2. A random walk (n_u), which is Gaussian noise integrated twice before being added to $\Delta\theta$. The model for computing $\Delta\theta_u$ is TBD.

The noise from both sources are added to $\Delta\theta$ to yield

$$\Delta\theta_{\text{NOISY}} = \Delta\theta_u + \Delta\theta_v + \Delta\theta$$

where $\Delta\theta$ is from the angular displacement calculation and $\Delta\theta_{\text{NOISY}}$ is the noisy angular displacement value.

4.2.4.2 FSS Model

The modeling of an FSS consists of the following steps:

1. With the Sun in body coordinates, test to see if the spacecraft is in shadow.
2. If the spacecraft is not in shadow, rotate the Sun vector to the sensor frame.
3. If the spacecraft is not in shadow, compute Sun angles α , β (adding noise optionally).

4. If the spacecraft is not in shadow, test to see if the Sun is in the sensor field of view.

These steps are described in the following subsections.

4.2.4.2.1 Spacecraft Shadow Testing

The geometry used to determine if the spacecraft is in the shadow is shown in Figure 4-3. If the Sun vector lies in the shaded area between the two spacecraft-Earth tangent vectors, the spacecraft is in the Earth's shadow.

The angle θ_{SE} between the Sun unit vector, \hat{s} , and Earth unit vector, \hat{e} , is computed as

$$\hat{s} \cdot \hat{e} = \cos \theta_{SE}$$

or

$$\theta_{SE} = \cos^{-1} (\hat{s} \cdot \hat{e})$$

The angle θ_c between \hat{e} and a vector from the spacecraft and tangent to the Earth is computed from

$$\cos \theta_c = \sqrt{R^2 - R_e^2} / R$$

or

$$\theta_c = \cos^{-1} \left(\sqrt{R^2 - R_e^2} / R \right)$$

where R is the distance from the spacecraft to the center of the Earth and R_e is the Earth radius.

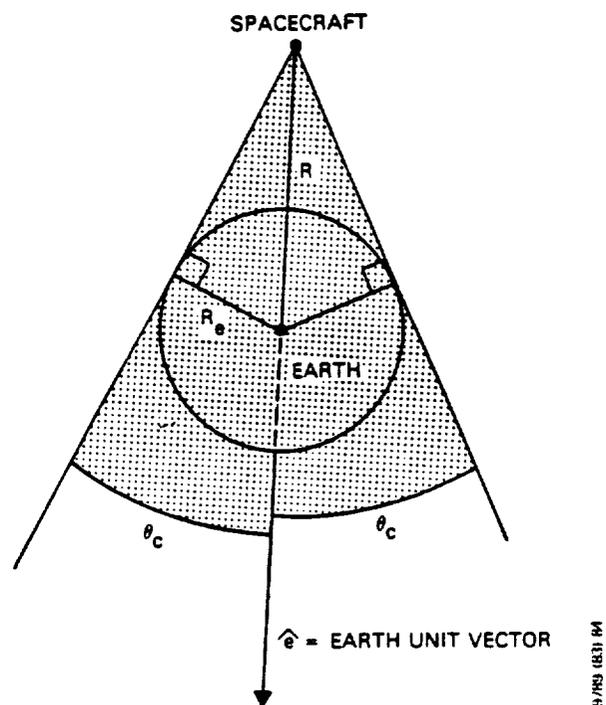


Figure 4-3. Spacecraft Shadow Testing Geometry

The Earth is blocking the Sun if

$$\theta_{SE} \leq \theta_c$$

or

$$\cos \theta_{SE} \geq \cos \theta_c$$

or

$$\hat{s} \cdot \hat{e} \geq \sqrt{R^2 - R_e^2} / R$$

4.2.4.2.2 Rotating Sun Vector to Sensor Frame

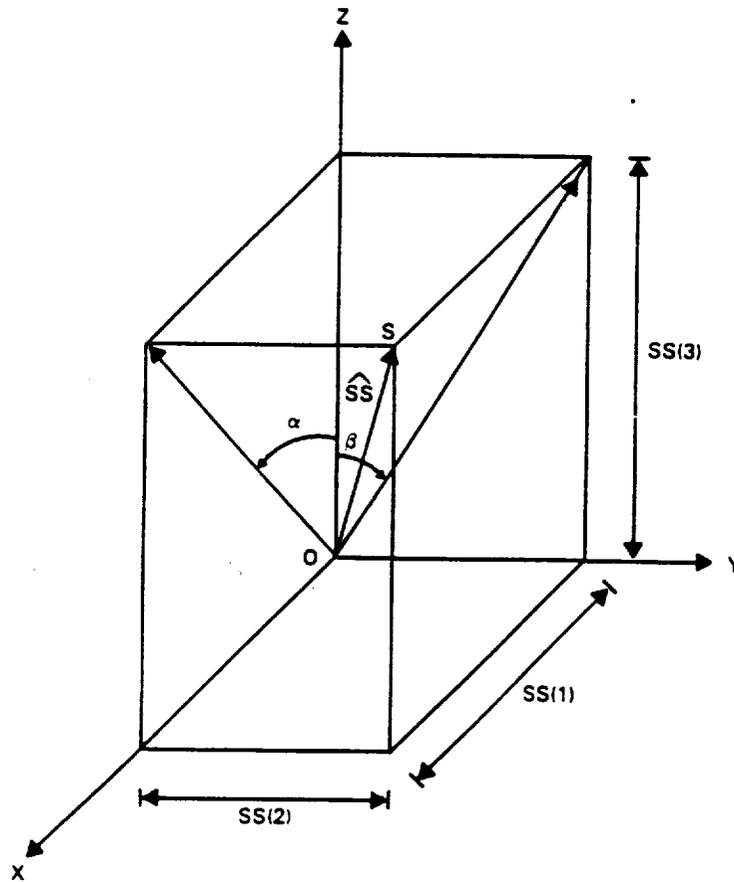
The Sun vector in sensor frame \hat{SS} can be computed from the ephemeris Sun vector, \hat{S}_I , expressed in an inertial frame by the equation

$$\hat{SS} = [A]_{SB} [A]_{BI} \hat{S}_I$$

where $[A]_{SB}$ is the body-to-sensor frame rotation matrix and $[A]_{BI}$ is the GCI-to-BCS attitude matrix.

4.2.4.2.3 Computing Sun Angles With Noise Optionally Added

Figure 4-4 shows how the two Sun angles (α and β) and the sensor frame are defined. (Nominally, the FSS reference frame is aligned with the BCS reference frame.) The +Z axis is the boresight vector. From this figure, $\tan \alpha = SS1/SS3$ and $\tan \beta = SS2/SS3$, where SS1, SS2, SS3 are the X, Y, Z



9/FEB/83 44

Figure 4-4. Sun Sensor FOV Angle Definitions in the FSS Reference Frame

components of the Sun vector in the sensor frame. The noise is added to the angle; thus, with noise added

$$\alpha = \tan^{-1} (SS1/SS3) + N_{\alpha}$$

$$\beta = \tan^{-1} (SS2/SS3) + N_{\beta}$$

where N_{α} , N_{β} are Gaussian noise with means M_{α} and M_{β} and standard deviations D_{α} and D_{β} . M_{α} , M_{β} , D_{α} , D_{β} are user input via SCIO Subsystem.

4.2.4.2.4 Testing if Sun in Sensor Field of View

The Sun is in the sensor FOV if and only if $|\alpha| \leq A_{\alpha}$ and $|\beta| \leq A_{\beta}$, where A_{α} and A_{β} are maximum half-angles for the FOV for angles α and β . If the FOV is square, A_{α} and A_{β} can be replaced by a single parameter A .

4.2.4.3 CSS Model

The CSSs are located on the four outer corners of the solar arrays and are shown in Figure 4-5. These units provide an output current that is a measure of the angle between the normal to the unit surface and the Sun line. The data provided are used to derive spacecraft roll and pitch attitude for Sun acquisition and/or reacquisition. For this mode of operation, the solar arrays are set to zero reference; that is, they are in the Y-Z plane, and the nominal orientation of outward unit normal vectors are as indicated in Figure 4-5.

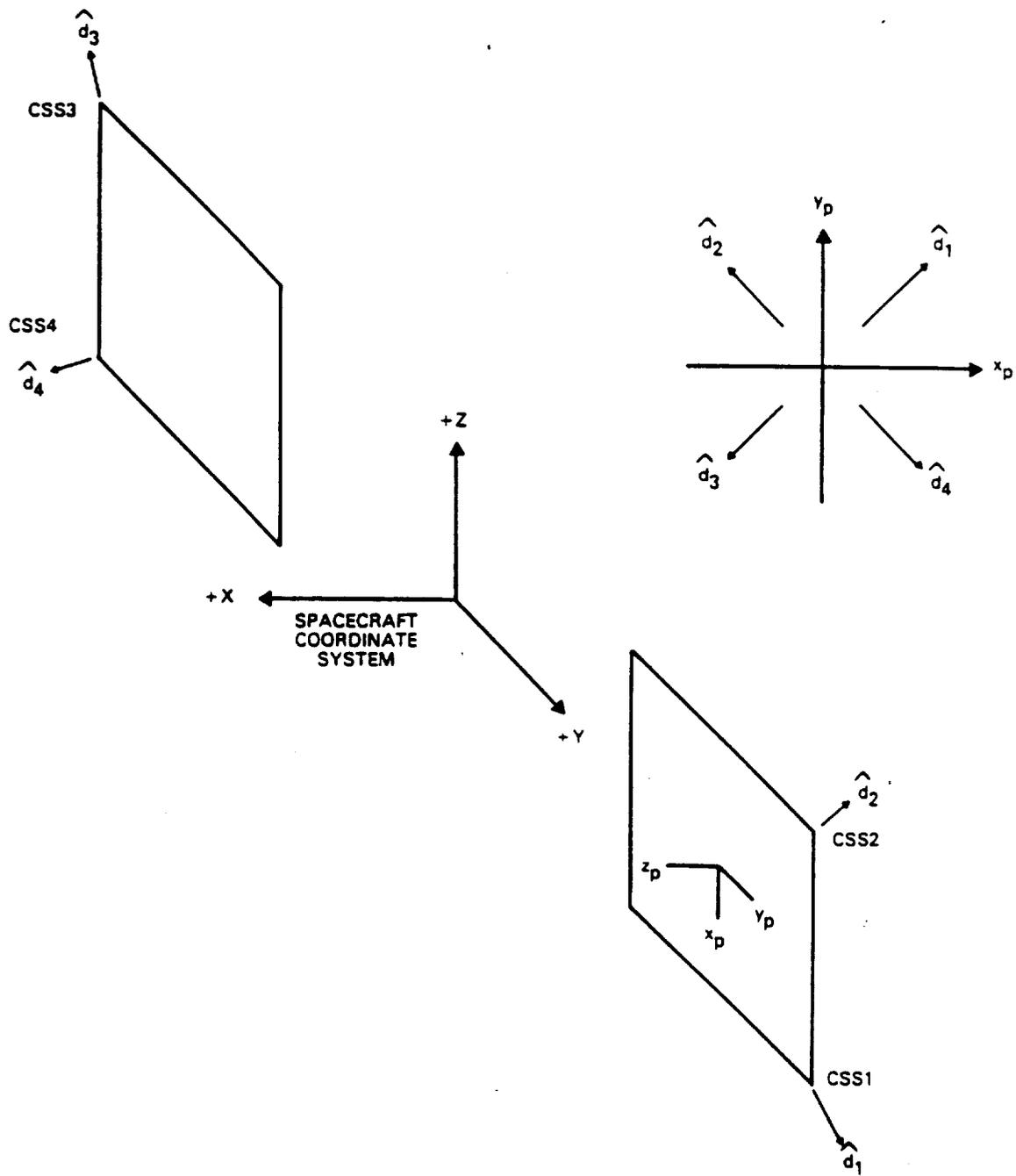


Figure 4-5. Coarse Sun Sensor Assembly

9789 (87) 84

The unit vectors to the CSS faces are given by the following, in body coordinates:

$$\hat{C}_{Bi} = \begin{bmatrix} -\cos \alpha \Delta E + \sin \alpha \sin B + \sin \alpha \cos B \Delta A \\ \cos B - \sin B \Delta A \\ \sin \alpha \Delta E + \cos \alpha \sin B + \cos \alpha \cos B \Delta A \end{bmatrix}$$

where α is the angle between the body X-axis and the unit vector normal to the active solar array face, and B is -45, 45, 135, and 225 degrees for sensors 1, 2, 3, and 4, respectively, and ΔA and ΔE are misalignments in the up/down and lateral directions when the sensor is viewed from outside the spacecraft.

With the unit vector to the Sun in body coordinates, \hat{R}_{SB} , the sensor measurement, O_{Ci} , is obtained as follows:

$$\alpha_i = \cos^{-1} [\hat{R}_{SB} \cdot \hat{C}_{SBi}] + n_i$$

and

$$O_{Ci} = K\alpha_i$$

where K is a scale factor nominally equal to 3.9 milliamperes per degree and n_i is Gaussian noise, to be added on user option, and is given as

$$n_i = \xi_i \text{STD}_i + \text{AM}_i$$

where ξ_i is a random number between (-1, 1), and STD_i and AM_i are the standard deviation and mean associated with the ith sensor.

4.2.4.4 FHST Model

The FHST Model generates the star camera data. Output from a star camera consists of a (u, v) coordinate measured in the camera's focal plane (see Figures 4-6 and 4-7) and a star intensity. The camera is commanded by the OBC to "search" an RFOV. All stars contained in the star catalog are searched to determine if they are within the RFOV. (The catalog consists of the GCI coordinates of 12 selected stars for a period of time spanning the requested simulation.)

Once a star is found within the RFOV, the corresponding (u, v) coordinates and intensity are digitized. If no stars are found in either camera FOV, no data is generated.

If a star camera is turned on, data will be generated as follows. Subroutine STCENT checks for blockage of the camera by the Earth, Moon, or Sun. Let [C] be the matrix that rotates vectors from GCI coordinates to spacecraft coordinates. Let \hat{S} , \hat{E} , \hat{M} be the Sun, Earth, and Moon unit vectors in GCI coordinates. These are rotated into body coordinates as follows:

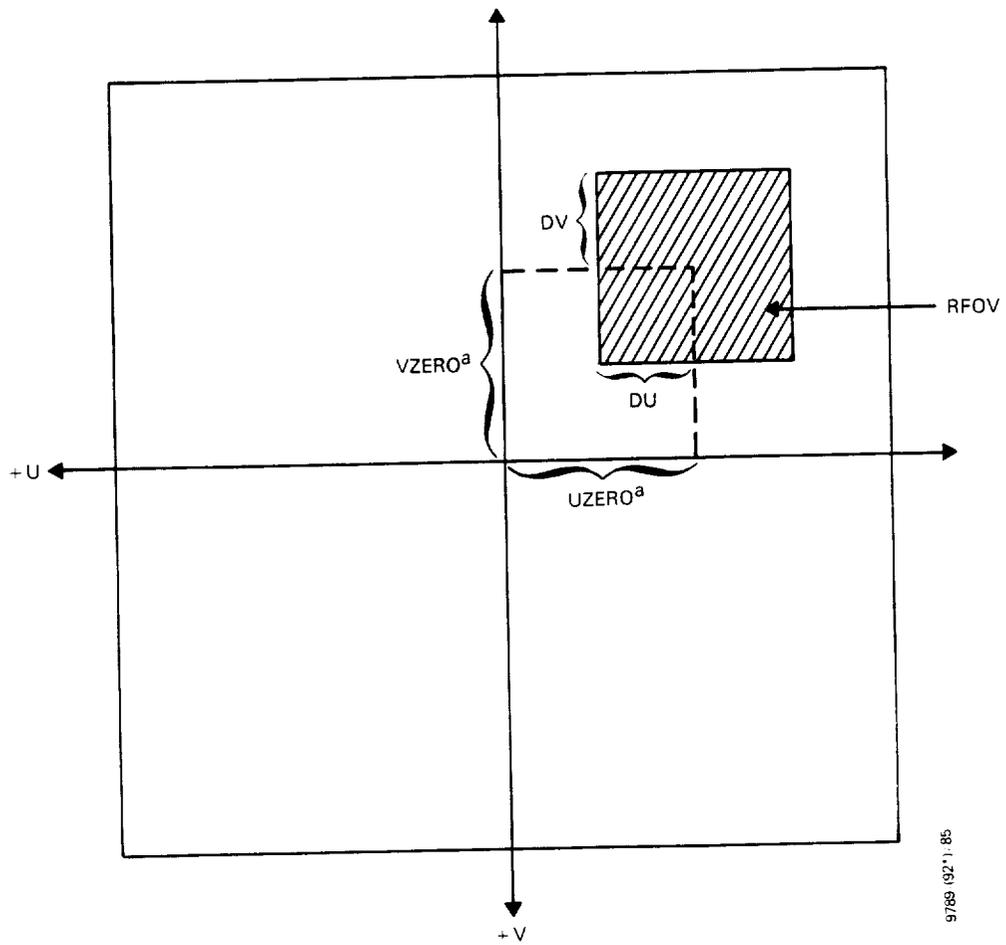
$$\hat{S}_{\text{body}} = [C] \hat{S}$$

$$\hat{E}_{\text{body}} = [C] \hat{E}$$

$$\hat{M}_{\text{body}} = [C] \hat{M}$$

The vectors in body coordinates are then rotated into camera coordinates via the transformation matrix [SC2ST]:

$$\hat{S}_{\text{camera}} = [\text{SC2ST}] \hat{S}_{\text{body}}$$



9789 (32*) 85

^a($UZERO$, $VZERO$) IS A COMMANDED OFFSET.

Figure 4-6. Camera Focal Plane

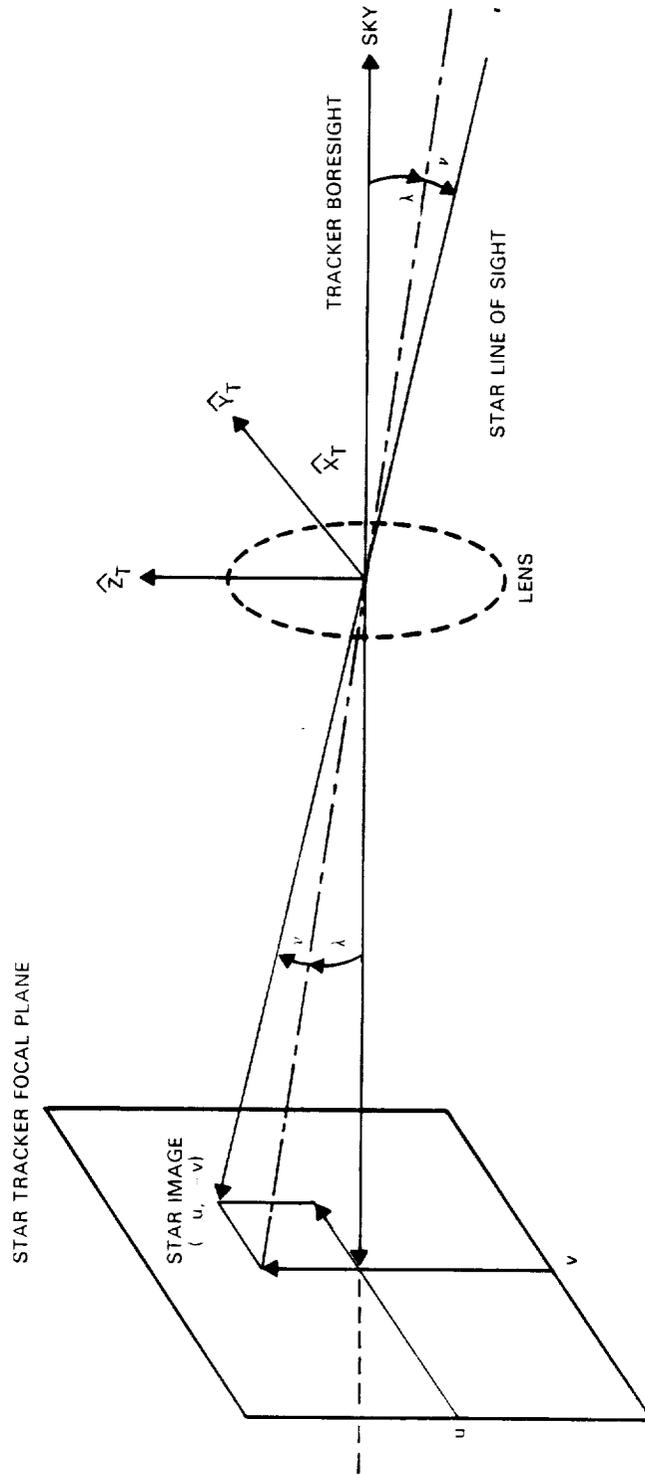


Figure 4-7. Star Tracker Angular Coordinates

$$\hat{E}_{\text{camera}} = [\text{SC2ST}] \hat{E}_{\text{body}}$$

$$\hat{M}_{\text{camera}} = [\text{SC2ST}] \hat{M}_{\text{body}}$$

Let C_S = cosine of Sun subtended angle, C_E = cosine of Earth subtended angle, and C_M = cosine of Moon subtended angle.

No output from the star camera will be generated if $\hat{S}_{\text{camera}}(1) > C_S$ or $\hat{E}_{\text{camera}}(1) > C_E$ or $\hat{M}_{\text{camera}}(1) > C_M$, where $\hat{S}_{\text{camera}} = \hat{E}_{\text{camera}} = \hat{M}_{\text{camera}} = 1$. If the camera is not occulted, a search is initiated for stars within the RFOV.

Let $\hat{\text{STAR}}$ be a unit vector of a star in GCI coordinates. ($\hat{\text{STAR}}$ is obtained by interpolating the discrete ephemeris information.) To obtain $\hat{\text{STAR}}$ in camera coordinates, the following transformations are performed in the following order:

$$\hat{\text{STR}} = \begin{bmatrix} \text{STR}_1 \\ \text{STR}_2 \\ \text{STR}_3 \end{bmatrix} = [\text{SC2ST}] [\text{C}] \hat{\text{STAR}}$$

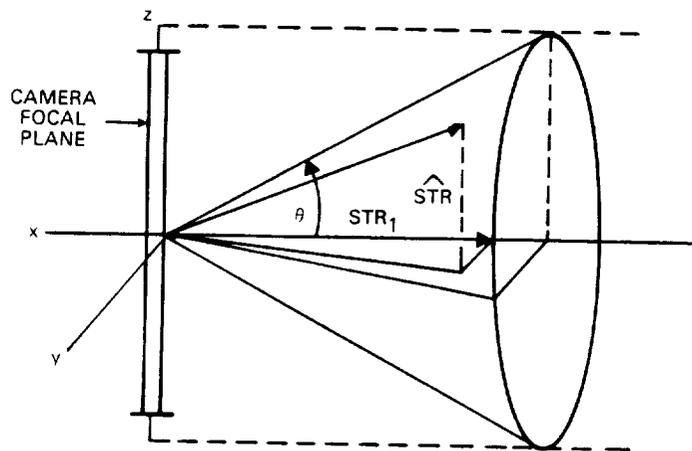
where matrices $[\text{SC2ST}]$ and $[\text{C}]$ are previously defined.

Before the associated (u, v) coordinates are generated for a star, its intensity must be above a fixed threshold, and the unit vector defining it must lie within a circular cone with a vertex angle of measure θ as shown in Figure 4-8.

If $\hat{\text{STR}}$ passes these checks, the (u, v) coordinates are generated as follows.

In terms of the angles λ, ν (see Figure 4-7):

$$\hat{\text{STR}} = \begin{bmatrix} \cos \nu \cos \lambda \\ -\sin \nu \\ -\cos \nu \sin \lambda \end{bmatrix}$$



9789 192 1 85

NOTES: $\theta \approx 6$ DEGREES.
 \widehat{STR}_1 MUST BE GREATER THAN $\cos \theta$ BEFORE \widehat{STR} IS GENERATED.

Figure 4-8. Accept All Star Unit Vectors Contained Within the Cone of Angle θ

From Figure 4-7, it can be seen that

$$v = -\tan \lambda$$

$$u = -\frac{\tan v}{\cos \lambda}$$

Solving u , v in terms of the star observation vector yields

$$v = -\tan \lambda = -\frac{\cos v \sin \lambda}{\cos v \cos \lambda} = \text{STR}_3 / \text{STR}_1$$

$$u = -\frac{\tan v}{\cos \lambda} = \frac{-\sin v}{\cos v \cos \lambda} = \text{STR}_2 / \text{STR}_1$$

Gaussian noise can be added to the (u, v) coordinates. The angular coordinates of $\widehat{\text{STR}}$, CU , and CV , in the camera frame of reference, are computed as follows:

$$\text{CU} = \text{STR}_2 / \text{STR}_1 + \text{UNOISE}$$

$$\text{CV} = \text{STR}_3 / \text{STR}_1 + \text{VNOISE}$$

where UNOISE , VNOISE are Gaussian noise to be added to the (u, v) coordinates.

The next test determines if (CU, CV) lies within the RFOV defined by the following rectangle (see Figure 4-6):

$$\text{UPLUS} = \text{UZERO} + \text{DU}$$

$$\text{UMINUS} = \text{UZERO} - \text{DU}$$

$$\text{VPLUS} = \text{VZERO} + \text{DV}$$

$$\text{VMINUS} = \text{VZERO} - \text{DV}$$

(CU, CV) lies within the RFOV if $CU \leq UPLUS$ and $CU \geq UMINUS$, and $CV \leq VPLUS$ and $CV \geq VMINUS$.

If any of the above conditions are violated, the next star in the star catalog is checked. This process continues until a star is found to satisfy all of the above conditions or the star catalog is exhausted. No data will be generated if there are no stars in either camera's RFOV.

4.2.4.5 Wheel Tachometer Model

Wheel tachometer readings are generated by computing the wheel speed from angular momentum and moment of inertia and computing the number of revolutions over a fixed time interval Δt . The model is as follows.

For each wheel ($i = 1, 2, 3, 4$)

1. Compute wheel speed

$$S_i = H_{W_i} / I_{W_i}$$

where H_{W_i} is the angular momentum of reaction wheel i , I_{W_i} is the moment of inertia of reaction wheel i , and S_i is the speed in radians per second of reaction wheel i .

The angular momentum, H_{W_i} is measured along the reaction wheel spin axis and is an element of the state vector.

2. The number of counts accumulated over time Δt is computed as

$$C_i = S_i \cdot K_{wh} \cdot \Delta t$$

where C_i is the number of counts over Δt , K_{wh} is the conversion factor from radians per second to counts, and Δt is the cycle time in seconds.

4.2.4.6 TAM Model

The TAM Model provides digital output from the magnetometers. The magnetometers measure the X, Y, Z components of the Earth's magnetic field in the magnetometer reference frame corrupted by biases caused by spacecraft residual fields (assumed to be constant). The final output from the TAM Model reflects the effects of any constant bias fields and fields caused by the torquer coils.

Let \vec{H} be the magnetometer measurement in the sensor frame of reference (FOR), \vec{B}_{mag} be the Earth's magnetic field (in the sensor FOR), \vec{B}_{bias} be the constant magnetic field bias, and \vec{B}_{coils} be the magnetic field due to the electromagnetic coils measured at the magnetometers. Then,

$$\vec{H} = \vec{B}_{mag} + \vec{B}_{bias} + \vec{B}_{coils}$$

The Earth's magnetic field \vec{B}_{mag} is obtained as follows. The Earth's B-field in GCI coordinates, \vec{B}_{GCI} , is obtained from the profile data set. If the transformation matrix [C] rotates vectors from GCI to spacecraft coordinates, the B-field in spacecraft coordinates, \vec{B}_{SC} , is given by

$$\vec{B}_{SC} = [C] \vec{B}_{GCI}$$

The B-field is then rotated into the magnetometer FOR using the matrix [SC2MAG]

$$\vec{B}_{mag} = [SC2MAG] \vec{B}_{SC}$$

where \vec{B}_{mag} is the Earth's magnetic field in the magnetometer FOR.

The term \vec{B}_{coils} is the field generated by the torque coils measured at the magnetometers:

$$\vec{B}_{\text{coils}} = [\text{ETQCPL}] \vec{m} \quad (4-49)$$

where m is the vector of the torquer coil commands and $[\text{ETQCPL}]$ is the torquer coil/magnetometer coupling matrix. The matrix $[\text{ETQCPL}]$ is derived as follows.

Consider the X-axis torquer coil. The magnetic field at the magnetometer attributable to the torquer coil is modeled as the sum of the two poles, which have a strength of $+M$ and $-M$, respectively (see Figure 4-9):

$$\vec{B}_x = M \left(\frac{\vec{r}_m - \vec{r}_+}{|\vec{r}_m - \vec{r}_+|^3} - \frac{\vec{r}_m - \vec{r}_-}{|\vec{r}_m - \vec{r}_-|^3} \right) \quad (4-50)$$

where M is the magnetic pole strength of the torquer bar (pole-centimeters (cm)), \vec{r}_m is the vector from the origin to the magnetometer, \vec{r}_+ is the vector from the origin to the positive pole of the torquer coil (cm), and \vec{r}_- is the vector from the origin to the negative pole of the torquer coil (cm). Here, \vec{r}_m , \vec{r}_+ , and \vec{r}_- are measured in an arbitrary reference frame. The magnetometers on board GRO are nominally aligned with the body coordinate system.

By definition, a dipole is equal to the pole strength times the distance between the positive and negative poles. Since

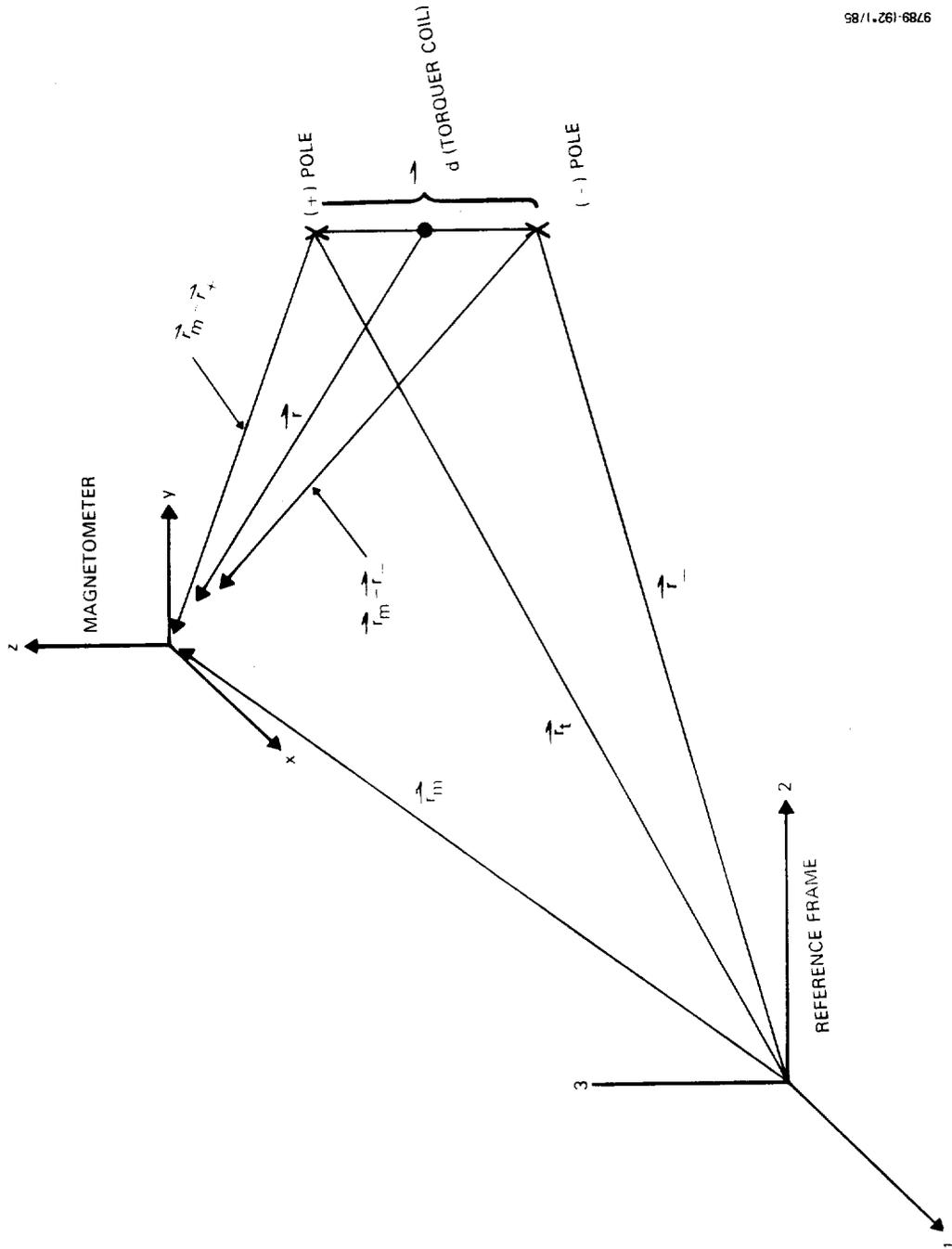


Figure 4-9. Torquer Bar in the Reference Frame

the commands to the torquer coils are in terms of dipoles, the following relationship is defined for convenience:

$$K \cdot M_{cx} = d \cdot M$$

where M_{cx} is the dipole commanded to the X-axis coil (pole-cm), K is a unitless constant, d is the distance between the positive and negative poles (cm), and M is the strength of the poles (pole).

Therefore, Equation (4-50) can be written in terms of the commanded dipole, M_{cx} :

$$\vec{B}_x = \frac{K}{d} \left(\frac{\vec{r} - \frac{\vec{d}}{2}}{|\vec{r} - \frac{\vec{d}}{2}|^3} - \frac{\vec{r} + \frac{\vec{d}}{2}}{|\vec{r} + \frac{\vec{d}}{2}|^3} \right) M_{cx} \quad (4-51)$$

where \vec{d} is the vector from the negative pole to the positive pole.

Simplifying Equation (4-51) will yield Equation (4-52a):

$$\vec{B}_x = \begin{bmatrix} Q_{x1} \\ Q_{x2} \\ Q_{x3} \end{bmatrix} M_{cx} \quad (4-52a)$$

where Q_{xi} ($i = 1, 3$) is in units of $1/\text{length}^3$.

Similar relationships can be written for the magnetic fields generated by the y and z torquer bars:

$$\vec{B}_y = \begin{bmatrix} Q_{y1} \\ Q_{y2} \\ Q_{y3} \end{bmatrix} M_{cy} \quad (4-52b)$$

$$\vec{B}_z = \begin{bmatrix} Q_{z1} \\ Q_{z2} \\ Q_{z3} \end{bmatrix} M_{Cz} \quad (4-52c)$$

The fields attributable to the x, y, z torquer coils measured at the magnetometer are given by \vec{B}_x , \vec{B}_y , and \vec{B}_z . The dipoles commanded to the x, y, and z torquer coils are given by M_{Cx} , M_{Cy} , and M_{Cz} , respectively.

By the principle of superposition, \vec{B}_{coils} , the total magnetic field due to the x, y, and z torquer coils, is given by

$$\vec{B}_{\text{coils}} = \vec{B}_x + \vec{B}_y + \vec{B}_z$$

Equations (4-52a-c) can be written as a matrix product:

$$\vec{B}_{\text{coils}} = \begin{bmatrix} Q_{x1} & Q_{y1} & Q_{z1} \\ Q_{x2} & Q_{y2} & Q_{z2} \\ Q_{x3} & Q_{y3} & Q_{z3} \end{bmatrix} \begin{bmatrix} M_{Cx} \\ M_{Cy} \\ M_{Cz} \end{bmatrix}$$

Therefore, from Equation (4-49),

$$[\text{ETQCPL}] = \begin{bmatrix} Q_{x1} & Q_{y1} & Q_{z1} \\ Q_{x2} & Q_{y2} & Q_{z2} \\ Q_{x3} & Q_{y3} & Q_{z3} \end{bmatrix}$$

and

$$\vec{m} = \begin{bmatrix} M_{Cx} \\ M_{Cy} \\ M_{Cz} \end{bmatrix}$$

where [ETQCPL] is the torquer coil/magnetometer coupling matrix and m is the torquer coil commands (dipoles in pole-cm).

Ideally, the magnetometer frame is parallel to the spacecraft frame. Any misalignment is assumed to be a rigid body rotation of the TAM, which will be represented internally by a direction cosine matrix and to the user as a Euler rotation sequence. The order of rotations is TBD, but should be chosen so that no singularities occur for 0-deg rotations.

4.2.5 ACTUATOR MODELS

4.2.5.1 Reaction Wheel Assembly

Torques imparted by the reaction wheel assemblies (RWAs) are modeled as follows. The wheel torque is provided by an alternating current two-phase induction motor, which is driven by square pulses provided by a reaction wheel drive electronics package. The torque level is controlled by varying the duty cycle or fraction of each half-cycle in which voltage is nonzero.

The torque equation is

$$N_{\text{wheel}} = X_{\text{DC}} \cdot N_{\text{EM}} - N_{\text{friction}} \quad (4-53)$$

where N_{wheel} = net torque on wheel
 X_{DC} = fraction of each half-cycle that the voltage is nonzero (the duty cycle)
 N_{EM} = electromagnetic torque when the duty cycle is unity
 N_{friction} = torque created by bearing friction

The value N_{EM} is proportional to wheel speed and the duty cycle X_{DC} is proportional to a control voltage provided by the OBC. Both of these functions are generated by piecewise linear interpolation on a table of calibration parameters.

The duty cycle, X_{DC} , is varied between +1 and -1 (inverted voltage polarity) by a control voltage, as shown in Figure 4-10). The OBC first computes the control voltage, I_{CMD} (counts). Then the duty cycle is obtained by

$$X'_{DC} = X_j + (|I_{CMD}| - I_j) \cdot \left(\frac{X_{j-1} - X_j}{I_{j-1} - I_j} \right), \quad (4-54a)$$

$$I_1 \leq I_{j-1} \leq |I_{CMD}| \leq I_j < I_M$$

or

$$X'_{DC} = X_M \quad \text{if } |I_{CMD}| \geq I_M$$

If

$$I_{CMD} < 0, X_{DC} = -X'_{DC} \quad (4-54b)$$

or if

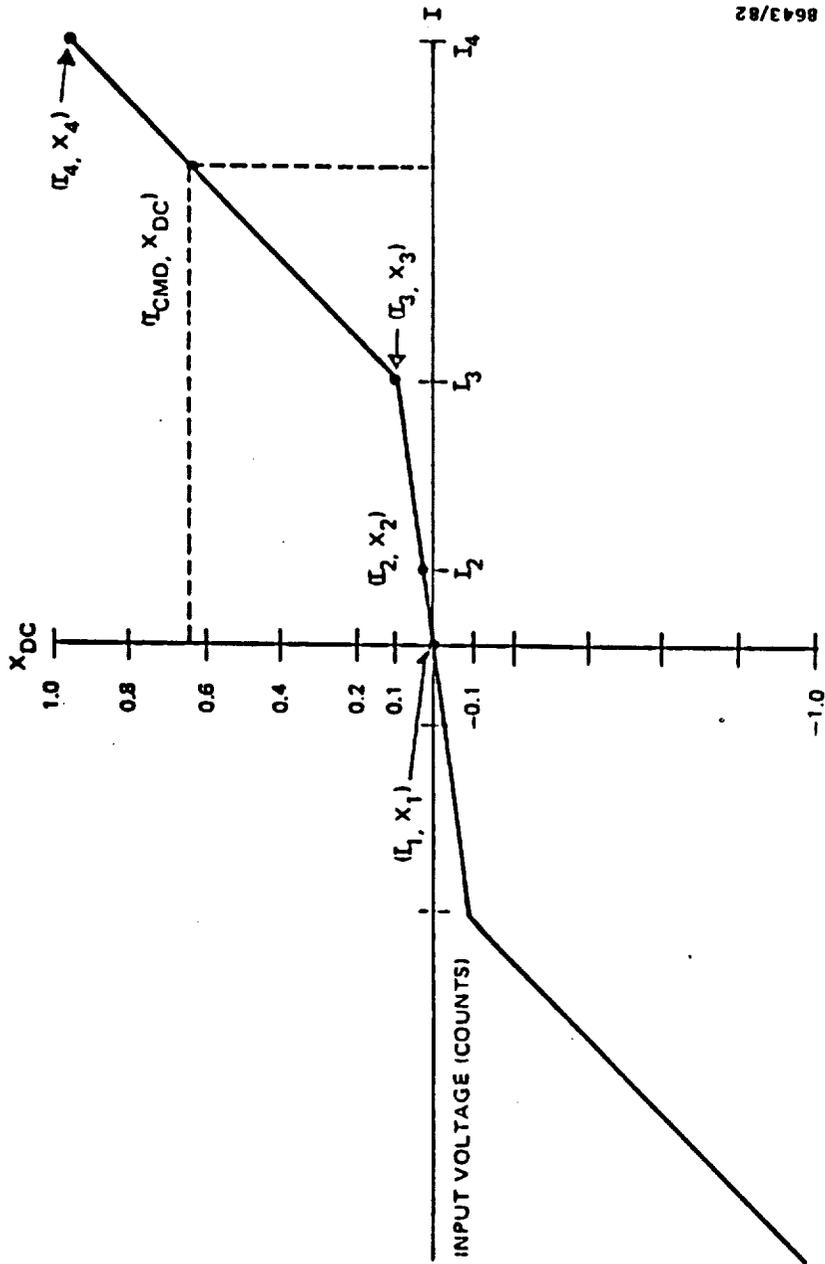
$$I_{CMD} \geq 0, X_{DC} = X'_{DC}$$

where $(I_1, X_1), (I_2, X_2), \dots, (I_M, X_M)$ are calibration parameters.

N_{EM} is obtained by

$$N_{EM} = \alpha_i + (s_c - s_i) \cdot \left(\frac{\alpha_{i-1} - \alpha_i}{s_{i-1} - s_i} \right), \quad s_1 < s_{i-1} \leq s_c \quad (4-55a)$$

$$\leq s_i < s_n$$



8643/82

Figure 4-10. Duty Cycle as a Function of the Commanded Input Voltage (Example From Landsat-D)

$$N_{EM} = \alpha_8 \quad \text{if } s_c \geq s_n \quad (4-55b)$$

$$N_{EM} = \alpha_1 \quad \text{if } s_c \leq s_1 \quad (4-55c)$$

The $(s_1, \alpha_1), (s_2, \alpha_2), \dots, (s_n, \alpha_n)$ are calibration parameters (see Figure 4-11).

$N_{friction}$ is modeled as the sum of a Coulomb term and a viscous term

$$N_{friction} = N_{coul} \cdot \text{sign}(s_c) + fv \quad (4-56)$$

where N_{coul} is the Coulomb friction coefficient, fv is the viscous friction coefficient, and $\text{sign}(s_c)$ gives the direction of the wheel momentum.

4.2.5.2 Magnetic Torquer Assembly

When current is applied to a ferromagnetic core electromagnet (also called a magnetic torquer assembly (MTA)), the resulting relationship between magnetic intensity (related to current) and magnetic induction (related to dipole strength) is multivalued. There are two possible dipoles that can be generated for a given current (provided that the magnet is driven to saturation at all times). Therefore, the resultant dipole depends on the applied current and whether the current is increasing or decreasing.

The hysteresis loop describing the MMS torquer coils is modeled by a parallelogram as shown in Figure 4-12. The demagnetization of the coil is not modeled.

When wheel momentum unloading is required, the OBC will command a dipole so that excess angular momentum will be dissipated. Eventually, the commanded dipole will exceed the absolute value of the constant residual dipole, D_r , in

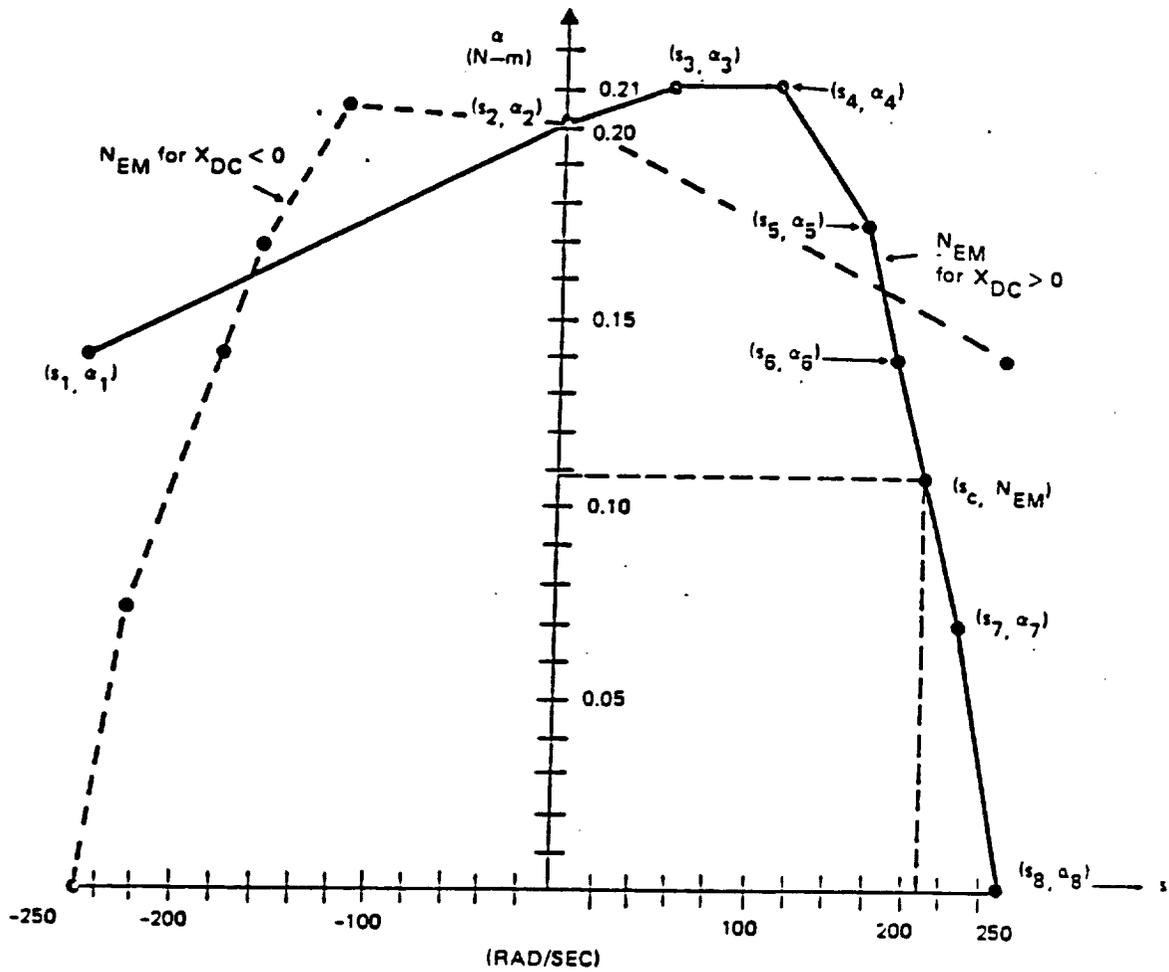


Figure 4-11. Wheel Torque Versus Wheel Speed
(From Landsat-D)

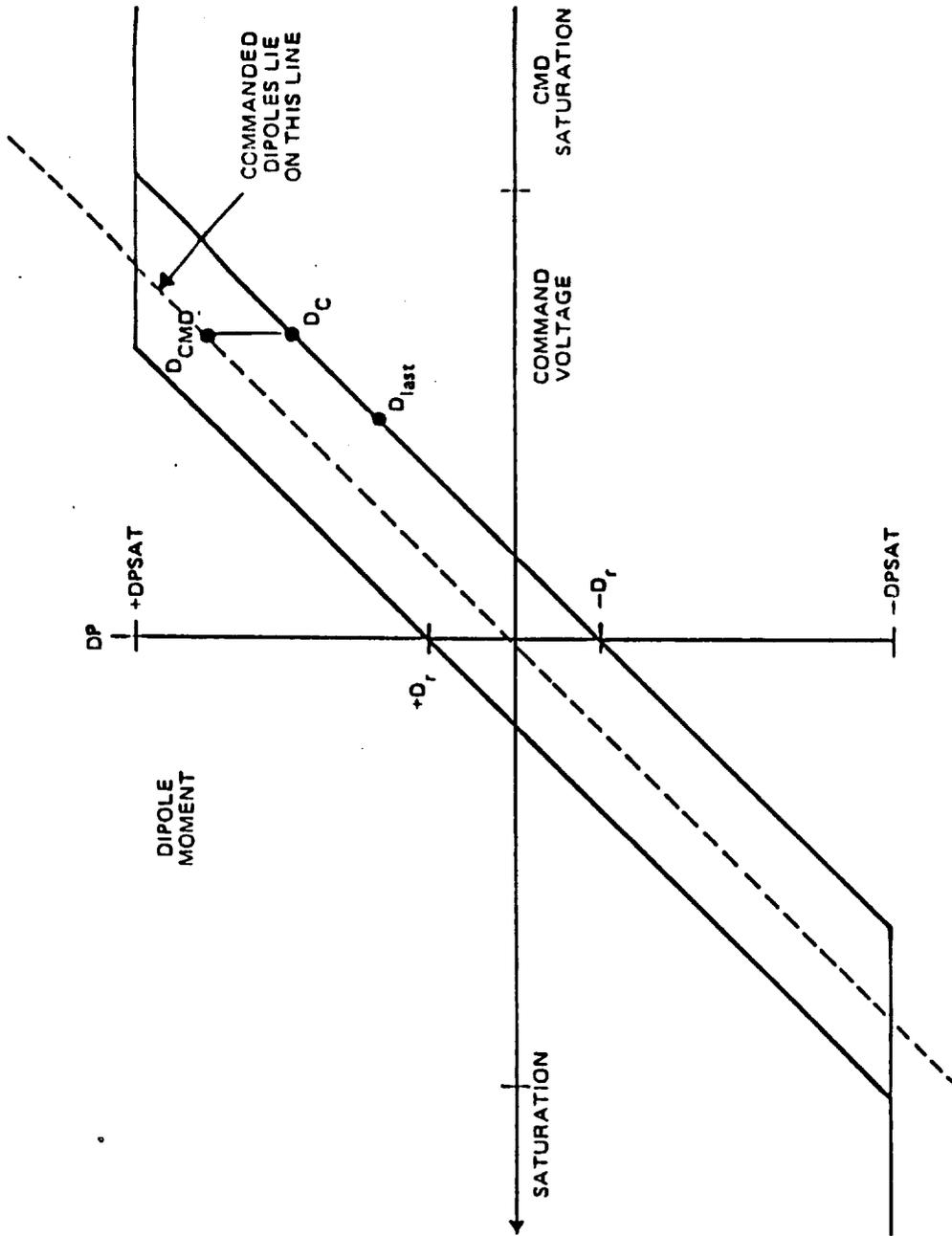


Figure 4-12. MMS Model of Hysteresis Loop

Figure 4-12. For any time thereafter, a dipole moment resulting from an OBC command can be represented by a point on the parallelogram in Figure 4-12.

The resultant magnetic dipole corresponding to an OBC command is computed for each of the three coils onboard, as follows.

Consider one coil. First the OBC commanded dipole (I_{CMD} in counts) is converted to ampere-meters² by the following equation:

$$D_{\text{CMD}} = S(I_{\text{CMD}} - Z) \quad (4-57)$$

where D_{CMD} is the commanded dipole in ampere-meters², S is a scaling factor, and Z is the command value for a zero-commanded dipole.

The dipole caused by this coil will change only if the difference between the dipole at the last sampling, D_{last} , and the current commanded dipole, D_{CMD} , is greater than the residual dipole moment, D_r .

Therefore, the current dipole, D_C , taking into account the current command, is determined as follows:

$$D_C = D_{\text{CMD}} - D_r \quad \text{if } D_{\text{CMD}} - D_{\text{last}} > D_r \quad (4-58a)$$

$$D_C = D_{\text{CMD}} + D_r \quad \text{if } D_{\text{CMD}} - D_{\text{last}} < -D_r \quad (4-58b)$$

$$D_C = D_{\text{last}} \quad \text{if } |D_{\text{CMD}} - D_{\text{last}}| \leq D_r \quad (4-58c)$$

If the applied current exceeds a certain value, the magnitude of the dipole produced by the ferromagnetic coil reaches a maximum. This is modeled in the MMS simulator as

shown in Figure 4-12, i.e., $|D_C| \leq DP_{SAT}$, where DP_{SAT} is the saturation value for the coils.

Once the dipole moment, \overrightarrow{DP} , is determined, the torque produced by its interaction with the Earth's magnetic field can be computed. The net torque, $\overrightarrow{T}_{S/C}$, applied to the spacecraft is

$$\overrightarrow{T}_{S/C} = \overrightarrow{DP}_{S/C} * \overrightarrow{B} \quad (4-59)$$

where \overrightarrow{B} is the Earth's magnetic field in Tesla (1 tesla = 10^4 gauss) in the spacecraft coordinate system, and $\overrightarrow{DP}_{S/C}$ is the vector sum of dipoles resulting from the OBC commands in ampere-meters² (in spacecraft coordinates).

The vector of current dipoles, $\overrightarrow{DP}_{acs}$, is obtained by using Equation (4-58). To obtain $\overrightarrow{DP}_{S/C}$, the following transformation is applied:

$$\overrightarrow{DP}_{S/C} = [T_{coils}] \overrightarrow{DP}_{acs} \quad (4-60)$$

where $[T_{coils}]$ is the matrix whose columns contain the unit vectors of each coil axis in the BCS frame.

The Earth's magnetic field in GCI coordinates, B_{GCI} , at the current time is available by interpolating the ephemeris information (see subroutine INTERP in Reference 10). Let $[C]$ be the matrix that rotates a vector from GCI to spacecraft coordinates. Then,

$$\overrightarrow{B}_{S/C} = [C] B_{GCI} \quad (4-61)$$

Attitude information in terms of the quaternion at the current time is available from ADMPAR. The attitude matrix $[C]$ is constructed from the quaternion via subroutine EULERC.

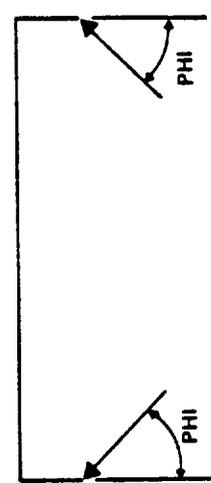
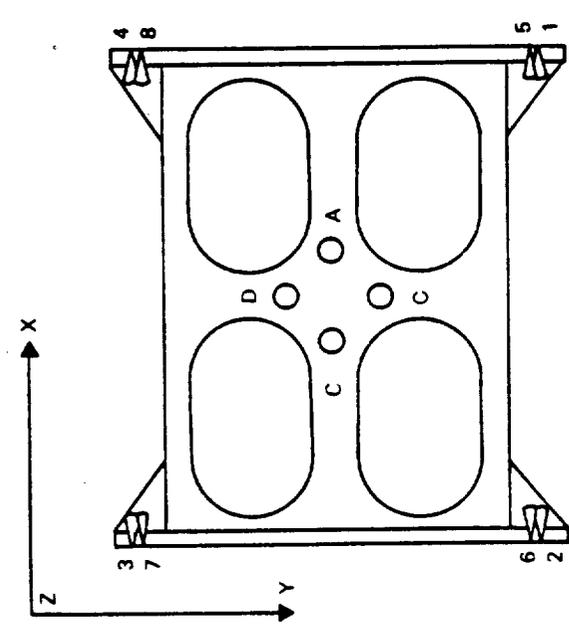
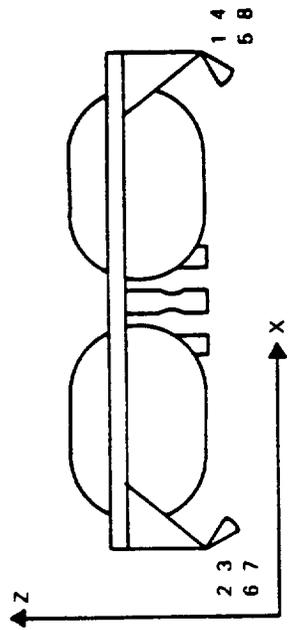
The torque on the spacecraft in the spacecraft frame due to the commanded dipole is then given by

$$\vec{T}_{S/C} = \overline{DP}_{S/C} * \vec{B}_{S/C} \quad (4-62)$$

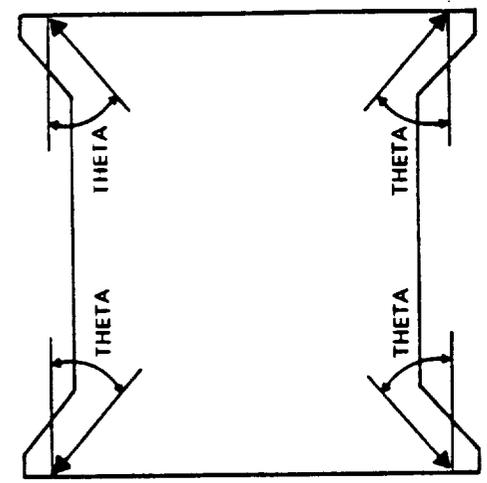
4.2.5.3 Attitude Control Thrusters

The ACTs generate a nominal thrust level of 22.241 newtons (5 lbf). There is a redundant set of thrusters located on legs at the four corners of a rectangular structural frame, as shown in Figure 4-13. They are numbered from 1 to 8, with the primary set given as 1 to 4. The thruster model employed is a trapezoidal model. The full thrust level is 22.241 newtons at steady state with a thrust buildup from t_s , the start time, to t_1 , the time steady thrust is reached. The thrust tail-off is from t_2 , the time the motor is commanded off, to t_f , the final time or the time the thrust reaches zero. It is assumed that the control system will command the proper motors to fire and when to fire them, thus defining t_s and t_2 . The intervals t_s to t_1 and t_2 to t_f are functions of the motor and need to be defined. Representative values may be used if they are not defined by the prime contractor. The setting of these time intervals should be controlled by the user. The system user may choose to set these time intervals equal to zero, simulating instantaneous on and off operation of the thrusters. Determination of the thrust depends on which of the three regions the time of evaluation is in:

$$F'_m = F_i [(t - t_s)/(t_1 - t_s)] \quad \text{when } t_s \leq t < t_1$$



THRUSTER ANGLE DEFINITION



5783-103-04

Figure 4-13. Thruster Location and Thruster Angle Convention

$$F'_m = F_i \quad \text{when } t_1 \leq t \leq t_2$$

$$F'_m = F_i \left[\frac{(t_f - t)}{(t_f - t_2)} \right] \quad \text{when } t_2 < t \leq t_f$$

where F_i is the motor thrust level and F'_m is the motor thrust along the motor axis.

The motor thrust is then scaled as follows:

$$F_m = S_i F'_M$$

where S_i is the scale factor for the i th motor. The scale factor allows the simulation of different performance levels of a motor, such as hot, under performance, or failure mode.

The thrust vector for each motor in body coordinates is given by

$$\vec{F}_{B_i} = \begin{bmatrix} \cos \theta \sin \phi - \sin \theta \Delta a + \cos \theta \cos \phi \Delta E \\ \sin \theta \sin \phi + \cos \theta \Delta A + \sin \theta \cos \phi \Delta E \\ \cos \phi - \sin \phi \Delta E \end{bmatrix} F_M$$

where θ in the column matrix is really θ_i and, if nonzero, will be given from the following:

i	1/5	2/6	3/7	4/8
θ_i	θ	$180 - \theta$	$180 + \theta$	$-\theta$

5132G(4)-24

A nonzero value of θ is TBD. The cant angle, ϕ , is currently equal to 29 deg. The misalignments for each thruster, ΔA and ΔE , are expressed in radians and should be available to the system user as input. They represent misalignments in lateral and up/down directions, respectively, i.e., in directions orthogonal to the motor axis.

With the thrusts determined for each firing motor, the resulting torque is obtained from the following:

$$\vec{N}_T = \vec{R} \times \vec{F}_B$$

where \vec{R} is the vector from the spacecraft center of mass and is given in Table 4-1. The torque for each motor fired is added vectorially to obtain the total torque due to thrusting.

4.2.5.4 Orbit Adjust Thrusters

The OATs generate a nominal thrust level of 444.822 newtons (100 lbf). They are located on the underside of the spacecraft symmetrically about the $-Z_B$ axis between the fuel tanks. Their locations are shown in Figure 4-13, where they are identified as thrusters A, B, C, and D. The modeling of the thrust for each motor is the same as that of the ACTs as given in Section 4.2.5.3. But once the motor thrust, F_M , is obtained, the thrust vector for each motor in body coordinates is given by

$$F_{B_i} = \begin{bmatrix} -\sin \alpha \Delta A + \cos \alpha \Delta E \\ \cos \alpha \Delta A + \cos \alpha \Delta E \\ 1 \end{bmatrix} F_M$$

where α is 0, 180, 90, 270 for motors A, B, C, and D, respectively, and ΔA and ΔE are motor misalignments as mentioned in Section 4.2.5.3. The torque for each motor fired is obtained, and the total torque generated is obtained as mentioned therein.

Table 4-1. Thruster Locations

MOTOR	LOCATION (m)		
	X_B	Y_B	Z_B
ORBIT			
A	0.381	0	-0.475
C	0	0.381	-0.475
B	-0.381	0	-0.475
D	0	-0.381	-0.475
ATTITUDE			
5	1.659	1.463	-0.475
6	-1.659	1.463	-0.475
7	-1.659	-1.463	-0.475
8	1.659	-1.463	-0.475
1	1.659	1.546	-0.475
2	-1.659	1.546	-0.475
3	-1.659	-1.546	-0.475
4	1.659	-1.546	-0.475

9789 (83) 04

NOTE: ATTITUDE THRUSTERS ARE IN SETS, 1, 2, 3, AND 4 AND 5, 6, 7, AND 8. ONE SET IS PRIMARY AND THE OTHER IS SECONDARY. UNTIL OTHERWISE INFORMED, ASSUME SET 1, 2, 3, AND 4 IS PRIMARY. SUBSCRIPT B DENOTES THE BODY COORDINATE SYSTEM.

SECTION 5 - ONBOARD COMPUTER MODEL SUBSYSTEM

The OBC Model Subsystem simulates functions normally performed on board the spacecraft. The functions include responding to ground commands, processing sensor data, and generating commands to the actuators.

5.1 REQUIREMENTS

5.1.1 PROCESSING REQUIREMENTS

- R4.1.1 Simulate the GRO attitude determination algorithms and the control laws of the OBC (NSSC-1).
- R4.1.2 Respond to ground commands issued by the Simulation Control and Input/Output (SCIO) Subsystem.
- R4.1.3 Process the following sensor data:
- IRU
 - FSS
 - CSS
 - FHST
 - Wheel Tachometer
 - TAM
- R4.1.4 Simulate the backup control laws of the attitude control electronics.
- R4.1.5 Generate commands to the following actuators:
- RWA
 - MTA
 - Thrusters
- R4.1.6 Simulate the following modes of operation (see Section 1.2.2 for descriptions):
- Normal pointing mode
 - Normal maneuver mode
 - Sun-referenced pointing mode
 - Safehold mode

- Thruster maneuver mode
- Velocity control mode
- Thruster command mode
- Backup orbit maintenance mode

R4.1.7 Determine commands required for pointing high-gain antenna at TDRS.

R4.1.8 Determine solar array position.

R4.1.9 Interpolate GRO and TDRS ephemerides.

5.1.2 INPUT REQUIREMENTS

R4.2.1 Accept the ground commands (listed in Appendix D) from the SCIO Subsystem.

R4.2.2 Receive user-specified input parameters from the SCIO Subsystem.

R4.2.3 Accept the following types of sensor data from the Truth Model Subsystem:

- IRU
- FSS
- CSS
- FHST
- Wheel tachometer
- TAM

5.1.3 OUTPUT REQUIREMENTS

R4.3.1 Output actuator commands to the Truth Model.

R4.3.2 Output error codes, precision checks, and analysis data to the SCIO Subsystem.

R4.3.3 Send the following telemetry to the SCIO Subsystem:

- OBC Subsystem attitude error (radians)
- OBC Subsystem compensated rates (radians per second)

- Break track commands
- Commands to enable/disable OATs
- Commands for positive and negative rotation about each axis (ACTs)
- Commanded magnetic dipole moment components
- Wheel torque commands
- HGA azimuth and elevation slew commands (radians)
- Attitude control and determination mode flag

5.2 MATHEMATICAL SPECIFICATIONS

5.2.1 SENSOR PROCESSING

5.2.1.1 Inertial Reference Unit Data Processing

The IRU data processing function is divided into two sub-functions: gyro data acquisition and gyro data compensation.

The functions of gyro data acquisition are

- Gyro counter overflow detection and correction
- Gyro prefilter

Figure 5-1 is a functional block diagram of gyro data acquisition. The gyro prefilter consists of two first-order filters in series. The bandwidth is set to 0.1 Hz, which is lower than the first bending mode bandwidth (0.36 Hz) and higher than the control system bandwidth (0.013 Hz).

The function of gyro data compensation is to compensate gyro data for scale factors and alignment and bias errors. Figure 5-2 is a functional block diagram of the gyro data compensation algorithm.

5.2.1.1.1 Gyro Data Acquisition Function

The gyro data acquisition function (GYROD) uses the gyro data counts to perform overflow compensation and prefiltering, as shown in the following pages.

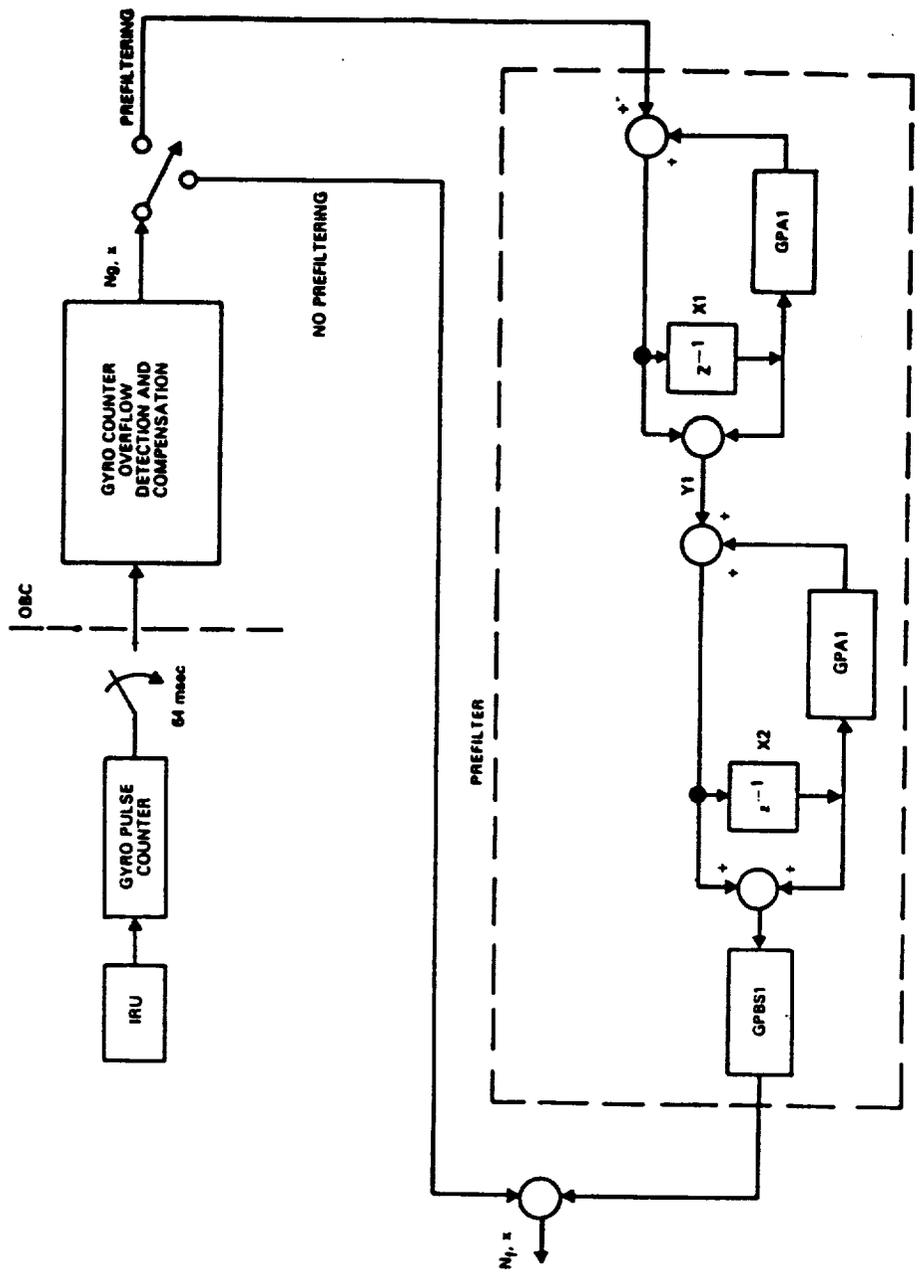


Figure 5-1. Gyro Data Acquisition Functional Block Diagram (X-Axis Channel)

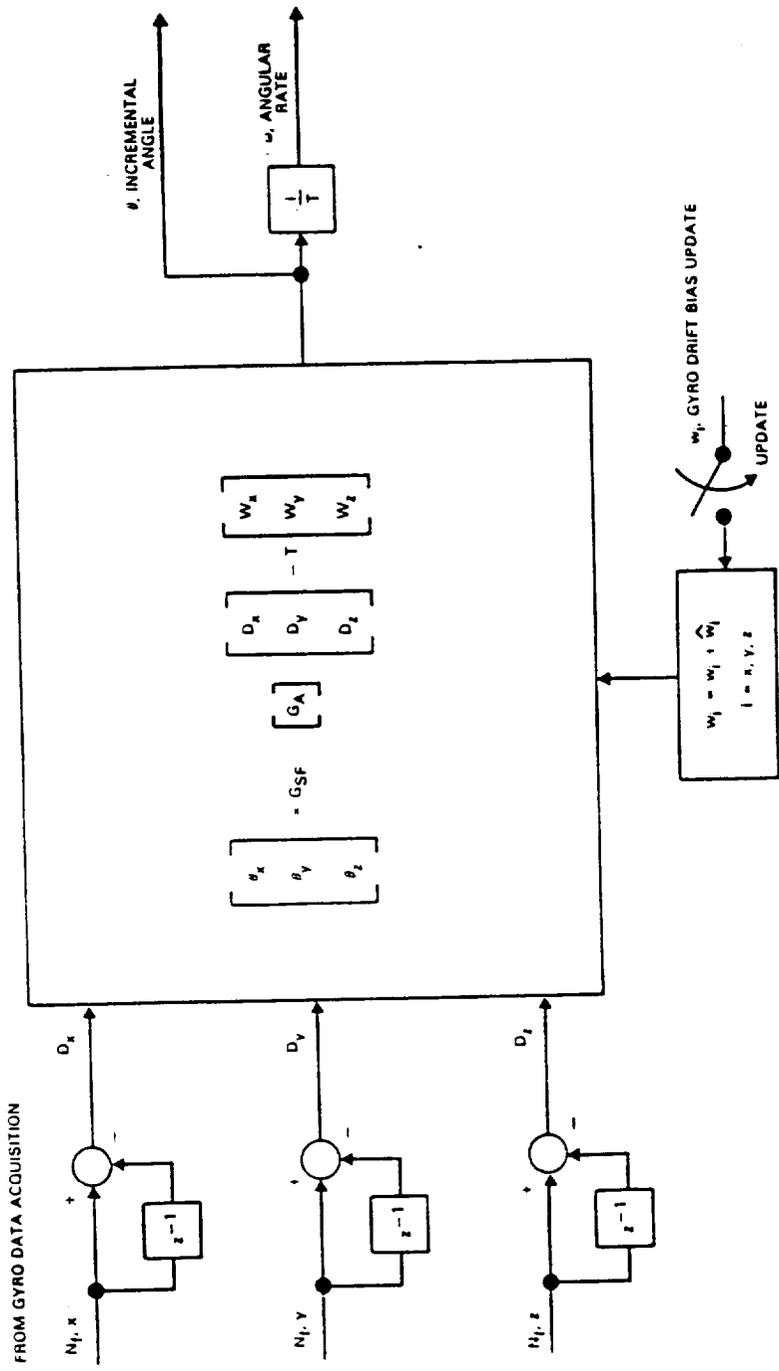


Figure 5-2. Gyro Data Compensation Functional Block Diagram

There are six gyro channels. Three are labeled primary, and the other three are backup channels. Table 5-1 gives the gyro channel definitions. The gyro channel data (NG_{ij} , $i=1$ to 3 and $j=1$ to 2) are sampled at 64 msec in all modes.

Table 5-1. Gyro Channel Definitions

<u>Axis (i)</u>		<u>Channel (j)</u>		<u>Gyro (k)</u>
Roll (X)	1	A	1	1
	1	B	2	3
Pitch (Y)	2	A	1	2
	2	B	2	3
Yaw (Z)	3	A	1	1
	3	B	2	2

- Input--GYROD uses the following input:

<u>Name</u>	<u>Description</u>
FLTRON	Gyro prefilter data base flag (data base constant (DBC)): = False, disable prefilter = True, enable prefilter
GPA	Gyro processing prefilter coefficient (related to GPB1) (DBC)
GPB	Gyro processing prefilter coefficient, which determines prefilter bandwidth (DBC)
GPBS	Gyro processing prefilter coefficient (GPB1) (DBC)
GYRINIT	GYRODAT initialization flag (from MODECON)
IGY _i	Gyro channel selection flag where $i = X, Y, Z$ body axes gyros (DBCs): = 0, channel A = 1, channel B = -1, gyro data are bad for axis i
PNGF _i	Past value of filtered gyro data for $i = X, Y, Z$ gyro sensor axes (variables from gyro data compensation function (GYROCOMP)/GYROD)

Name	Description
NG _{ij}	Raw gyro data pulse counts where i = X, Y, Z gyro sensor axes; j = 1, 2 for channel A or B (output of IRU pulse counters)
NGLC	Gyro accumulator overflow test limit (DBC)
NGS	Gyro accumulator saturation level (DBC)

● Processing--The gyro data acquisition execution interval is 256 msec, and execution is performed before gyro data compensation.

For a given 256-msec time interval, four separate sets of gyro information have been gathered. At any one time, five complete sets of gyro data are saved. Four of these are for the current 256-msec cycle, and the last set is from the end of the previous 256-msec cycle. In this section, the previous 256-msec cycle set of gyro channel A and B data is subtracted from the current set of gyro channel A and B data for the current 256-msec cycle.

If the GYRODAT initialization flag is set (GYRINIT=1), the following is performed:

```

For i = 1 to 3 and j = 1 to 2
  PNGij = NGij           | Past value of gyro pulse counters
  TNGij = 0             | Cumulative gyro counter
  NGIi = 0             | Prefilter state variable
  NGLi = 0             | Prefilter state variable
  PNGFi = 0           | Past filtered counter
  GYRINIT = 0

```

The data are first converted from radians (from the Truth Model) to counts:

$$NG_{ij} = NG_{ij} / GSFH_{ij} \quad (\text{for gyro high rate})$$

$$NG_{ij} = NG_{ij} / GSFL_{ij} \quad (\text{for gyro low rate})$$

```

For i = 1 to 3 and j = 1 to 2

```

Once the data conversion is completed, the following computations are performed for both primary and backup channels (i=1 to 3, j=1 to 2):

$$DNG = NG_{ij} - PNG_{ij}$$

$$PNG_{ij} = NG_{ij}$$

where DNG = difference in gyro counts during last 64 msec
 PNG_{ij} = Past values for gyro counts from last 64 msec
 or last frame (i = X, Y, Z and j = 1, 2 for
 channels A and B)

If |DNG| > NGLC, the gyro pulse counter for i = X, Y, or Z has overflowed, and the gyro data require compensation, as follows:

$$DNG = DNG - \text{sign}(DNG) * NGS$$

The differences are then added up follows:

$$TNG_{ij} = TNG_{ij} + DNG \quad \text{for } i=1 \text{ to } 3 \text{ and } j=1 \text{ to } 2$$

where TNG_{ij} = cumulative gyro counter (OBC gyro counter)

The prefilter step is performed next. The filter step computations are not performed for a particular axis if the proper flag is set (IGY_i = -1). If the prefilter flag (FLTRON) is set to TRUE, the filter state variables are updated to minimize the filter transient if the prefiltering is ever performed. If the prefilter flag is set to FALSE, the following computations are performed:

$$j = IGY_i + 1$$

$$NGT1_i = TNG_{ij} + GPA * NGI_i$$

$$\text{NGT2}_i = \text{NGT1}_i + \text{NGI}_i + \text{GPA} * \text{NGL}_i$$

$$\text{NGI}_i = \text{NGT1}_i$$

$$\text{NGF}_i = \text{GPBS} * (\text{NGT2}_i + \text{NGL}_i)$$

$$\text{NGL}_i = \text{NGT2}_i$$

(where j indicates the prime gyro channel (j=1 to 2))

If the filter is off, the following computations are performed:

$$j = \text{IGY}_i + 1$$

$$\text{NGF}_i = \text{TNG}_{ij}$$

$$\text{NGI}_i = \text{TNG}_{ij} / (2 * \text{GPB})$$

$$\text{NGL}_i = \text{TNG}_{ij} / (2 * \text{GPBS})$$

(j indicates the prime gyro channel (j=1 or 2))

where NGT1_i = temporary variable

NGT2_i = temporary variable

NGI_i = filter state variable

NGL_i = filter state variable

NGF_i = filtered gyro data

- Output--GYROD produces the following output:

<u>Name</u>	<u>Description</u>
NGF_i	Filtered values of gyro data for i = X, Y, Z gyro sensor axes (to GYROCOMP)
NGH_i	Past value of filtered gyro data for i = X, Y, Z gyro sensor axes (to GYROCOMP)
TNG_{ij}	Cumulative OBC gyro counters for all six gyro channels (i = 1 to 3 and j = 1 to 2) (to GYRODAT, SAFECON)

5.2.1.1.2 Gyro Data Compensation Function

The gyro data compensation function (GYROCOMP) uses gyro count increments to produce compensated body roll, pitch, and yaw angular rates, as shown in the following tabulation. It performs compensation for scale factors, alignment errors, and gyro bias compensation terms as part of this process.

- Input--GYROCOMP uses the following input:

<u>Name</u>	<u>Description</u>
ACSINT	ACS sample period (depends on mode) (from MODECON)
GC _{ij}	Elements of gyro alignment calibration matrix from gyro sensor axes (j = 1, 2, 3) to ACAD body axes (i = 1, 2, 3) (DBC's)
GSFH _i	Gyro high-rate mode scale factor for i = X, Y, Z gyro sensor axes (DBC's)
GSFL _i	Gyro low-rate mode scale factor for i = X, Y, Z gyro sensor axes (DBC's)
IGY _i	Gyro channel selection flag where i = X, Y, Z gyro sensor axes (DBC's): = 0, channel A = 1, channel B = -1, bad gyro axis data
NGF _i	Filtered gyro data for i = X, Y, Z gyro sensor axes (variables from GYROD)
PNGF _i	Past values of filtered gyro data for i = X, Y, Z gyro sensor axes (variables from GYROD/GYROCOMP)
SENSTA _i	Gyro rate status word (high = 0 or low = 1 modes); (i = X, Y, Z gyro axes) (status word from IRU)
THETB _i	Estimated bias gyro rate where i = X, Y, Z ACAD body axes (from ESTIMAT via KININT or DBC/GND)

- Processing--GYROCOMP is executed every 512 msec in the normal pointing and normal maneuver modes and every 256 msec in the velocity control and the thruster maneuver modes. This module is performed before kinematic integration.

The processing first computes the change in gyro counts over the past processing cycle and saves the last computed filtered gyro counts for all three active gyro channels. This computation is performed only for those gyro axes that have been labeled good ($IGY_i \neq -1$):

$$CNF_i = NGF_i - PNGF_i$$

$$PNGF_i = NGF_i$$

where i is X, Y, Z (gyro sensor axes) and CNF_i is the difference in filtered gyro counts over the past processing cycle.

The processing then determines the rate mode for each gyro channel and converts gyro counts to radians. The rate (high or low) of the gyro channel currently in use is determined from the gyro rate status word $SENSTA_i$. If $SENSTA_i = 0$, the gyro is in the high-rate mode, and if the gyro channel is good ($IGY_i \neq -1$), the following computations are performed:

$$j = IGY_i + 1$$

$$THETC_i = GSFH_{ij} * CNF_i$$

If $SENSTA_i = 1$, the gyro is in the low-rate mode, and if the gyro channel is good ($IGY_i \neq -1$), the following computations are performed:

$$j = IGY_i + 1$$

$$THETC_i = GSFL_{ij} * CNF_i$$

where i is X, Y, Z; $THETC_i$ is the gyro angular increment in radians; and j is the prime channel indicator ($j = 1$ or 2).

Normally all three gyro channels will have the same rate mode; however, the processing allows for mixed-mode operation. The converted gyro angular increment data are next transformed from gyro sensor axis coordinates to ACAD coordinates:

$$WG_X = (GC_{11} * THETC_X) + (GC_{12} * THETC_Y) + (GC_{13} * THETC_Z)$$

$$WG_Y = (GC_{21} * THETC_X) + (GC_{22} * THETC_Y) + (GC_{23} * THETC_Z)$$

$$WG_Z = (GC_{31} * THETC_X) + (GC_{32} * THETC_Y) + (GC_{33} * THETC_Z)$$

The increments are then corrected for estimated gyro rate bias:

$$THETA_i = WG_i - THETB_i * ACSINT$$

where WG_i = uncompensated gyro angular increments in roll, pitch, and yaw ($i = X, Y, Z$ ACAD body axes)

$THETA_i$ = compensated gyro angular increments in roll, pitch, and yaw ($i = X, Y, Z$ ACAD body axes)

The compensated angular rate components of the spacecraft are then computed as

$$W_i = THETA_i / ACSINT$$

where W_i represents compensated spacecraft body angular rates in roll, pitch, and yaw ($i = X, Y, Z$ body axes).

- Output--GYROCOMP produces the following output:

<u>Name</u>	<u>Description</u>
PNGF _i	Past values of filtered gyro data for $i = X, Y, Z$ ACAD body axes (to GYROD)

<u>Name</u>	<u>Description</u>
THETA _i	Compensated angular increments in roll, pitch, and yaw where i = X, Y, Z ACAD body axes (to KINIT)
W _i	Compensated spacecraft body rates in roll, pitch, and yaw where i = X, Y, Z ACAD body axes (to WHECON, ATTEST, HLTCON, LLTCON, and SAFECON)

5.2.1.2 Fine Sun Sensor Data Processing

The FSS data processing function (FSSPROC) converts and compensates raw data from the selected FSS head to form tangents of the angular components of the Sun vector in FSS head coordinates (Figure 5-3).

- Input--FSSPROC uses the following input:

<u>Name</u>	<u>Description</u>
AF _{ijk}	FSS data calibration coefficients where i = 1,2, j = 1,2, ..., 9, k=1,2 (DBC's)
FSSA	Raw FSS alpha angle in radians (from Truth Model)
FSSB	Raw FSS beta angle in radians (from Truth Model)
HN	FSS head number; HN=0: Head 1; HN=1: Head 2
SUNPRS	Sun presence flag

- Processing--FSS data processing is scheduled every 32.768 seconds in all modes. FSS data processing first looks at SUNPRS to determine the Sun presence. If SUNPRS is FALSE, no further processing of the raw data is performed. The remainder of the processing occurs if SUNPRS is set to TRUE.

The next step is to take the raw FSS alpha and beta angles from the Truth Model and convert them to counts. This is performed as follows:

$$C_FSSA = (FSSA - B_CONV) / A_CONV$$

$$C_FSSB = (FSSB - B_CONV) / A_CONV$$

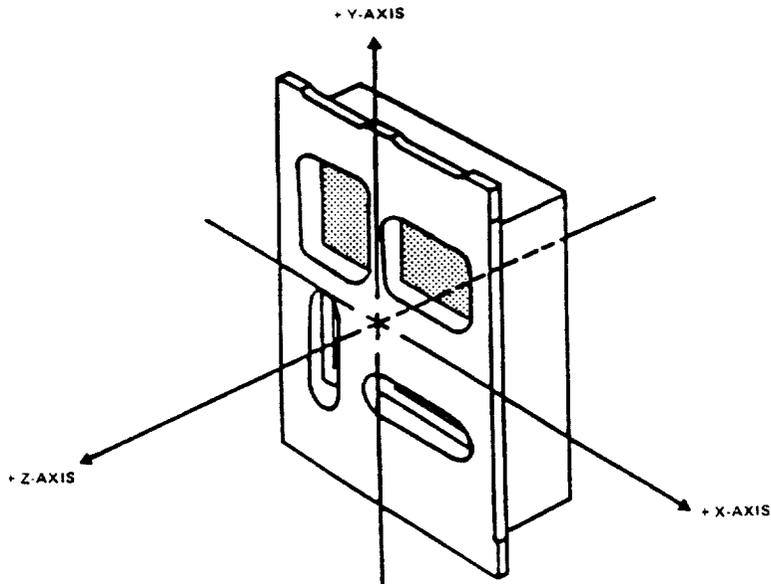
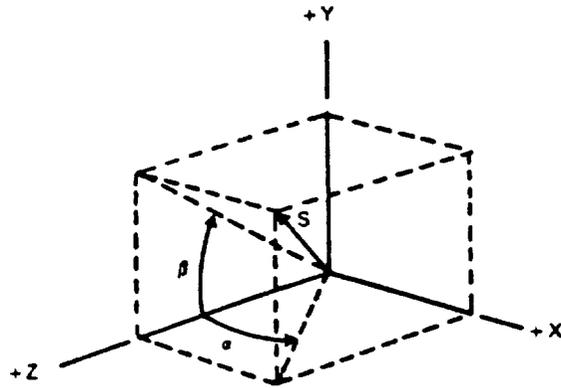


Figure 5-3. FSS Coordinate Frame and Sun Angle Component Definition

The tangents of the alpha and beta angles in FSS head coordinates are computed using the input values of alpha and beta compensated for known calibration errors as follows:

$$k = HN + 1$$

$$X = AF_{11k} + AF_{12k} * C_{FSSA} + AF_{13k} * \text{SIN}(AF_{14k} * C_{FSSA} + AF_{15k}) + AF_{16k} * \text{SIN}(AF_{17k} * C_{FSSA} + AF_{18k})$$

$$Y = AF_{21k} + AF_{22k} * C_{FSSB} + AF_{23k} * \text{SIN}(AF_{24k} * C_{FSSB} + AF_{25k}) + AF_{26k} * \text{SIN}(AF_{27k} * C_{FSSB} + AF_{28k})$$

$$TA = (AF_{19k} + X) / (1.0 - AF_{19k} * X)$$

$$TB = (AF_{29k} + Y) / (1.0 - AF_{29k} * Y)$$

- Output--FSSPROC produces the following output:

<u>Name</u>	<u>Description</u>
HN	Sun sensor select status (to ATTEST)
SUNPRS	Sun presence flag (to ATTEST, MODECON)
TA	Tangent of the alpha Sun angle in FSS head coordinates (to ATTEST)
TB	Tangent of the beta Sun angle in FSS head coordinates (to ATTEST)

5.2.1.3 Fixed-Head Star Tracker Data Processing

The FHST data processing function converts the raw FHST data to a form suitable for further processing and compensates the FHST measurements for the effects of known sensor measurement errors. It compensates for measurement error sources of geometric distortion, magnetic fields, temperature variations, and star intensity variations and converts the compensated angular data to FHST position coordinates. This function is performed every 32.768 seconds in the standby and normal pointing modes.

- Input--FHST processing uses the following input:

<u>Name</u>	<u>Description</u>
A _{ij}	Flat field and temperature calibration coefficients for H where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
BES _i	Saved components of Earth's magnetic field vector along spacecraft axes where i = X, Y, Z (variables from magnetometer processing)
B _{ij}	Flat field and temperature calibration coefficients for V where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
C _{ij}	Star intensity correction coefficients for H where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
D _{ij}	Star intensity correction coefficients for V where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
EB _{ij}	Coefficients used to compensate H for effects of ZT axis magnetic field where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
EH _{ij}	Coefficients used to compensate H for effects of XT axis magnetic field where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
EV _{ij}	Coefficients used to compensate H for effects of YT axis magnetic field where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
FB _{ij}	Coefficients used to compensate V for effects of ZT axis magnetic field where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
FHST	FHST selection flag, set during attitude estimation process (from ATTEST): = 1, selects FHST 1 = 2, selects FHST 2
FH _{ij}	Coefficients used to compensate V for effects of XT axis magnetic field where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
FV _{ij}	Coefficients used to compensate V for effects of YT axis magnetic field where i = 1 to 19 (coefficient number) and j = 1, 2 for FHST 1 or 2 (DBC's)
FHHAS _i	Saved raw FHST horizontal measurements where i = 1, 2 for FHST 1 or 2 (from ATTEST)

<u>Name</u>	<u>Description</u>
KF _{ijk}	Coefficients that relate magnetic torquer bar coil currents to magnetic field at FHST where i = X, Y, Z (FHST axes), j = X, Y, Z (antenna boom axes), and k = 1, 2 for FHST 1 or 2 (DBC's)
KST	FHST measurement unit conversion (DBC)
MDS _i	Saved components of torquer bar generated dipole moment in antenna boom coordinates (from ATTEST)
FHTS _i	Saved FHST temperature data from FHST i where i = 1, 2 (from ATTEST)
FHMS _i	Saved FHST intensities from FHST i where i = 1, 2 (from ATTEST)
XT _{ij}	Components of FHST X-axis unit vectors along j ORC axis for FHST i where i = 1, 2 and j = X, Y, Z (DBC's)
FHVAS _i	Saved raw FHST vertical measurements where i = 1, 2 for FHST 1 or 2 (from ATTEST)
YT _{ij}	Components of FHST Y-axis unit vector along j ORC axis for FHST i where i = 1, 2, j = X, Y, Z (DBC's)
ZT _{ij}	Components of FHST Z-axis unit vector along j ORC axis for FHST i where i = 1, 2, j = X, Y, Z (DBC's)

• Processing--First it must be determined whether the processing is for FHST 1 or FHST 2. Let m denote the FHST chosen, and let

$$H = FHHAS_m$$

$$V = FHVAS_m$$

$$TS = FHTS_m$$

$$IS = FHMS_m$$

$$XT'_i = XT_{mi} \quad \text{for } i = X, Y, Z$$

$$YT'_i = YT_{mi} \quad \text{for } i = X, Y, Z$$

$$ZT'_i = ZT_{mi} \quad \text{for } i = X, Y, Z$$

$$KF'_{ij} = KF_{ijm} \quad \text{for } i = X, Y, Z \text{ and } j = X, Y, Z$$

- where
- TS = temperature measurement from selected FHST
 - IS = intensity measurement from selected FHST
 - XT'_i = components of selected FHST X-axis unit vector along i ORC axis where i = X, Y, Z
 - YT'_i = components of selected FHST Y-axis unit vector along i ORC axis where i = X, Y, Z
 - ZT'_i = components of selected FHST Z-axis unit vector along i ORC axis where i = X, Y, Z
 - KF'_{ij} = coefficients that relate magnetic field at selected FHST to the magnetic torquer bar coil currents where i = X, Y, Z (FHST axes), j = X, Y, Z (antenna boom coordinates)
 - m = FHST 1 or 2

The raw FHST position data are calibrated in six steps. Each step requires the evaluation of a polynomial of the following form:

$$\begin{aligned}
 F(H, V, X, \vec{A}_i) = & A_{1i} + A_{2i} * V + A_{3i} * H + A_{4i} * X \\
 & + A_{5i} * V^2 + A_{6i} * V * H + A_{7i} * V * X \\
 & + A_{8i} * H^2 + A_{9i} * H * X + A_{10i} * X^2 \\
 & + A_{11i} * V^3 + A_{12i} * V^2 * H + A_{13i} * V^2 * X \\
 & + A_{14i} * V * H^2 + A_{15i} * V * H * X \\
 & + A_{16i} * V * X^2 + A_{17i} * H^3 + A_{18i} * H^2 * X \\
 & + A_{19i} * H * X^2
 \end{aligned}$$

where \vec{A}_i denotes a vector composed of the coefficients A_{1i}, \dots, A_{19i} and H, V, X are scalars. In this polynomial, i is either 1 or 2, depending on whether the FHST is 1 or 2. $\vec{B}_i, \vec{C}_i,$ and \vec{D}_i are defined similarly. The algorithm is as follows:

1. Compensate horizontal and vertical measurements for flat field and temperature effects:

$$\text{HHT} = F(H, V, \text{TS}, \vec{A}_i) + H$$

$$\text{VVT} = F(H, V, \text{TS}, \vec{B}_i) + V$$

where HHT and VVT are partially calibrated FHST horizontal and vertical data.

For these equations,

$$A_{3i} = A_{3i}^* - 1.0$$

and

$$B_{2i} = B_{2i}^* - 1.0$$

where A_{3i}^* and B_{2i}^* are the BASD calibration coefficients.

2. Compensate for star intensity variations:

$$\text{HHI} = F(\text{HHT}, \text{VVT}, \text{IS}, \vec{C}_i) + \text{HHT}$$

$$\text{VVI} = F(\text{HHT}, \text{VVT}, \text{IS}, \vec{D}_i) + \text{VVT}$$

where HHI and VVI are partially calibrated FHST horizontal and vertical data.

For these equations,

$$C_{3i} = C_{3i}^* - 1.0$$

and

$$D_{2i} = D_{2i}^* - 1.0$$

where C_{3i}^* and D_{2i}^* are the BASD calibration coefficients.

3. Compute the Earth's magnetic field components along the FHST axes:

$$BHF = BES_X * XT'_X + BES_Y * XT'_Y + BES_Z * XT'_Z$$

$$BVF = BES_X * YT'_X + BES_Y * YT'_Y + BES_Z * YT'_Z$$

$$BBF = BES_X * ZT'_X + BES_Y * ZT'_Y + BES_Z * ZT'_Z$$

where BHF, BVF, and BBF are components of the Earth's magnetic field along the FHST XT, YT, and ZT axes, respectively.

Also, compute the total magnetic field at the FHST:

$$BHT = BHF + KF'_{XX} * MDS_X + KF'_{XY} * MDS_Y + KF'_{XZ} * MDS_Z$$

$$BVT = BVF + KF'_{YX} * MDS_X + KF'_{YY} * MDS_Y + KF'_{YZ} * MDS_Z$$

$$BBT = BBF + KF'_{ZX} * MDS_X + KF'_{ZY} * MDS_Y + KF'_{ZZ} * MDS_Z$$

where BHT, BVT, and BBT are components of the total magnetic field along the FHST XT, YT, and ZT axes, respectively.

Also, compensate for magnetic field variations along the ZT axis:

$$\text{HHM1} = F(\text{HHI}, \text{VVI}, \text{BBT}, \vec{\text{EB}}_i) + \text{HHI}$$

$$\text{VVM1} = F(\text{HHI}, \text{VVI}, \text{BBT}, \vec{\text{FB}}_i) + \text{VVI}$$

where $\vec{\text{EB}}_i, \vec{\text{FB}}_i$ = coefficients describing FHST i

HHM1, VVM1 = partially calibrated FHST horizontal and vertical data

For these equations,

$$\text{EB}_{3i} = \text{EB}_{3i}^* - 1.0$$

and

$$\text{FB}_{2i} = \text{FB}_{2i}^* - 1.0$$

where EB_{3i}^* and FB_{2i}^* are the BASD calibration coefficients.

4. Compensate for magnetic field variations along the XT axis:

$$\text{HHM2} = F(\text{HHM1}, \text{VVM1}, \text{BHT}, \vec{\text{EH}}_i) + \text{HHM1}$$

$$\text{VVM2} = F(\text{HHM1}, \text{VVM1}, \text{BHT}, \vec{\text{FH}}_i) + \text{VVM1}$$

where $\vec{\text{EH}}_i, \vec{\text{FH}}_i$ = coefficients describing FHST i

HHM2, VVM2 = partially calibrated FHST horizontal and vertical data

For these equations,

$$EH_{3i} = EH_{3i}^* - 1.0$$

and

$$FH_{2i} = FH_{2i}^* - 1.0$$

where EH_{3i}^* and FH_{2i}^* are the BASD calibration coefficients.

5. Compensate for magnetic field variations along the YT axis:

$$HHM3 = F(HHM2, VVM2, BVT, EV_i) + HHM2$$

$$VVM3 = F(HHM2, VVM2, BVT, FV_i) + VVM2$$

$$PHIM = KST * HHM3$$

$$THETAM = KST * VVM3$$

where EV_i, FV_i = coefficients describing FHST i
 $HHM3, VVM3$ = calibrated FHST horizontal and vertical data in counts, respectively
 $PHIM, THETAM$ = calibrated FHST horizontal and vertical data in radians

For these equations,

$$EV_{3i} = EV_{3i}^* - 1.0$$

and

$$FV_{2i} = FV_{2i}^* - 1.0$$

where EV_{3i}^* and FV_{2i}^* are the BASD calibration coefficients.

6. Convert the compensated angles (PHIM, THETAM) to position coordinates:

$$TPHI = PHIM * \{1 + PHIM^2[(1/3) + PHIM^2 * 2/15]\}$$

$$TTHETA = THETAM * \{1 + THETAM^2[(1/3) + THETAM^2 * 2/15]\}$$

$$X = TTHETA^2 + TPHI^2$$

$$ZSC = 1 - X/2 + (3/8)X^2 - (5/16)X^3$$

$$XSC = -ZSC * TTHETA$$

$$YSC = ZSC * TPHI$$

where TPHI, TTHETA = calibrated FHST data in position coordinates
(equations shown are suitable approximations for tan PHIM and tan THETAM)

X = intermediate parameter

XSC = compensated FHST line-of-sight (LOS) unit vector component along the FHST X-axis

YSC = compensated FHST LOS unit vector component along the FHST Y-axis

ZSC = compensated FHST LOS unit vector component along the FHST Z-axis

- Output--STAR DAT produces the following output:

<u>Name</u>	<u>Description</u>
IS	Tracked star intensity (magnitude) (to ATTEST)
XSC	Compensated FHST LOS unit vector component along the FHST X-axis (to ATTEST)
XT _i	Components of FHST X-axis unit vector along i ACAD axis where i = X, Y, Z (to ATTEST)
YSC	Compensated FHST LOS unit vector component along the FHST Y-axis (to ATTEST)
YT _i	Components of FHST Y-axis unit vector along i ACAD body axis where i = X, Y, Z (to ATTEST)
ZT _i	Components of FHST Z-axis unit vector along i ACAD body axis where i = X, Y, Z (to ATTEST)

5.2.1.4 Wheel Tachometer Data Processing

The wheel tachometer data processing function computes the stored wheel momentum for each reaction wheel using the position data for each wheel from the reaction wheel electronics assembly (RWEA).

- Input--Wheel tachometer processing uses the following input:

<u>Name</u>	<u>Description</u>
KWS _i	Reaction wheel tachometer count-to-momentum conversion factor for wheel i for i=1 to 4 (db)
WHL _i	Reaction wheel i tachometer data (from Truth Model)
MINIT	Momentum computation initialization flag (= 0, do not initialize, = 1, initialize) (from MODECON)

- Processing--Wheel tachometer processing must be performed every 256 msec but is used in the standby, normal pointing, and normal maneuver modes every 512 msec. It is performed prior to the computation of the reaction wheel control and distribution laws software function and the magnetic control software function.

If initialization is required, the following code is performed:

```
If MINIT = 1 Then
    OWHLi = WHLi
    MINIT = 0
Endif
```

The tachometer data from the Truth Model are converted from radians to counts. This is performed as indicated:

$$\text{CONRW}_i = \text{KWS}_i * 0.256 \text{ sec}/I_{\text{rw}} \quad (I_{\text{rw}} = 0.62547892 \text{ slug-ft}^2)$$

where I_{rw} is the moment of inertia of a reaction wheel and 0.256 sec is the cycle time. This was the calculated TRW stated capability of 393 ft-lbf-sec at 6000 RPM (Reference 14).

The data from the Truth Model are then converted to counts:

$$\text{WHL}_i = \text{WHL}_i / \text{CONRW}_i$$

Next, the angular position change from the last computation cycle is computed for each reaction wheel, as follows:

$$\text{NW}_i = \text{WHL}_i - \text{OWHL}_i$$

where NW_i = change in reaction wheel i position tachometer counts and $i = 1$ to 4

OWHL_i = value of reaction wheel i tachometer counts from last cycle (see the following paragraphs) and $i = 1$ to 4

These data are then limited to 2047 counts.

If $|NW_i| > 2047$,

$$NW_i = NW_i - 4096 * \text{sign}(WHL_i)$$

where 4096 is the total range of the tachometer register plus 1.

Otherwise, there is no change in NW_i .

The stored reaction wheel angular momentum for each reaction wheel i is then computed as

$$HW_i = KWS_i * NW_i$$

where HW_i is the stored angular momentum for reaction wheel i , and $i = 1$ to 4.

Finally, the values of the reaction wheel i tachometer counts are saved for use in the next computation cycle

$$OWHL_i = WHL_i \quad \text{for } i = 1 \text{ to } 4$$

• Output--Wheel tachometer processing has the following output:

<u>Name</u>	<u>Description</u>
HW_i	Stored angular momentum for reaction wheel i where $i = 1$ to 4 (to MAGCON, WHECON)
MINIT	Momentum computation initialization flag; reset to 0 by TACPRO (to MODECON)

5.2.1.5 Magnetometer Data Processing

The magnetometer data processing function accepts data from the TAMs and the MTAs and computes the net magnetic field at the TAM attributable to the Earth's magnetic field.

• Input--The input for the magnetometer data processing module is described in Table 5-2.

Table 5-2. Magnetometer Data Processing (1 of 2)

DEFINITION	VECTOR	NOMINAL VALUE ^a	UNITS ^b	SOURCE
DATA FROM MAGNETOMETER 1 TAM 1X TAM 1Y TAM 1Z	\vec{TAM}	N/A	TESLAS	TRUTH MODEL
DATA FROM MAGNETOMETER 2 TAM 2X TAM 2Y TAM 2Z	\vec{TAM}	N/A	TESLAS	TRUTH MODEL
TORQUER BAR DIPOLE MOMENT MEASUREMENTS, TWO COILS PER AXIS DPM 1X DPM 1Y DPM 1Z	\vec{DPM}	N/A	AMP-M**2	TORQUE DRIVERS
TORQUER BAR DIPOLE MOMENT MEASUREMENTS, TWO COILS PER AXIS DPM 2X DPM 2Y DPM 2Z	\vec{DPM}	N/A	AMP-M**2	TORQUE DRIVERS
MAGNETOMETER SCALE FACTOR KMS		-0.003315	GAUSS/ COUNT	DATA BASE
MAGNETOMETER DATA ZERO OFFSET COMPENSATION B_0			COUNTS	DATA BASE
COORDINATE TRANSFORMATION AND CONVERSION CONSTANT OF THE TORQUER BARS EFFECT ON THE MAGNETOMETER KB _{XX} KB _{XY} KB _{XZ}	\vec{KB}	-5.662E-5 -4.426E-6 -3.901E-6	GAUSS/ COUNT	DATA BASE
COORDINATE TRANSFORMATION AND CONVERSION CONSTANT OF THE TORQUER BARS EFFECT ON THE MAGNETOMETER KB _{YX} KB _{YY} KB _{YZ}	\vec{KB}	0.0 1.923E-5 -1.835E-5	GAUSS/ COUNT	DATA BASE
COORDINATE TRANSFORMATION AND CONVERSION CONSTANT OF THE TORQUER BARS EFFECT ON THE MAGNETOMETER KB _{ZX} KB _{ZY} KB _{ZZ}	\vec{KB}	1.656E-5 2.184E-5 2.083E-5	GAUSS/ COUNT	DATA BASE

5132G(4)-21

Table 5-2. Magnetometer Data Processing (2 of 2)

DEFINITION	VECTOR	NOMINAL VALUE ^a	UNITS ^b	SOURCE
MAGNETIC TORQUER SCALE FACTOR TDE _{i=3}	\overline{XTDE}	3x1.02228	COUNT/ AMP-M**2	DATA BASE
TORQUER BAR SELECTION FLAG MAGFLD 1 = TDE1 2 = TDE2		1	N/A	DATA BASE
TELSA TO GAUSS CONVERSION TTOO		1.0E+4	GAUSS/ TESLA	DATA BASE
TORQUER BAR DIPOLE MOMENT ZERO OFFSET COMPENSATION M _O		0	COUNTS	DATA BASE
COMPONENTS OF MAGNETIC FIELD BIAS VECTOR DUE TO NONTORQUER BAR ELEMENTS OF THE SPACECRAFT BB X BB Y BB Z	\overline{BB}	0	GAUSS	DATA BASE
MAGNETOMETER SELECTION FLAG SELMAG (1 = TAM 1) (2 = TAM 2)		1	N/A	DATA BASE
DEFINITION	VECTOR	NOMINAL VALUE ^a	UNITS	DESTINATION
COMPONENTS OF EARTH'S MAGNETIC FIELD VECTOR ALONG SPACECRAFT AXES BE X BE Y BE Z	\overline{BE}	< 0.25	GAUSS	MAGNETIC CONTROL LAW, ATTITUDE ESTIMATION
X, Y, Z TORQUER BAR GENERATED DIPOLE MOMENTS MD X MD Y MD Z	\overline{MD}	N/A	A/D	ATTITUDE ESTIMATION

5132G(4)-22

^a NOMINAL VALUE IS SINGLE PRECISION.
^b D/A IS DIGITAL TO ANALOG; A/D IS ANALOG TO DIGITAL.

- Processing--After it is determined which magnetometer is to be used, the zero offset of the raw measurements is removed, and the result is appropriately scaled to give the measured magnetic field. Figure 5-4 is the functional flow diagram. Next, the magnetic dipole moments for each axis generated by the magnetic torquer bars are computed. Finally, the measured magnetic field is compensated by subtracting these dipole moment contributions, along with any other spacecraft magnetic biases, to yield the net magnetic field of the Earth at the TAM.

This software function receives input from magnetometer electronics and the torque drivers. It sends output to the attitude estimation algorithm (FHST calibration) and the magnetic control law (momentum management) for further processing. These output data are the X-, Y-, and Z-components of the magnetic field at the magnetometer sensor being used. Magnetometer data processing is performed every 512 msec in modes that are TBD.

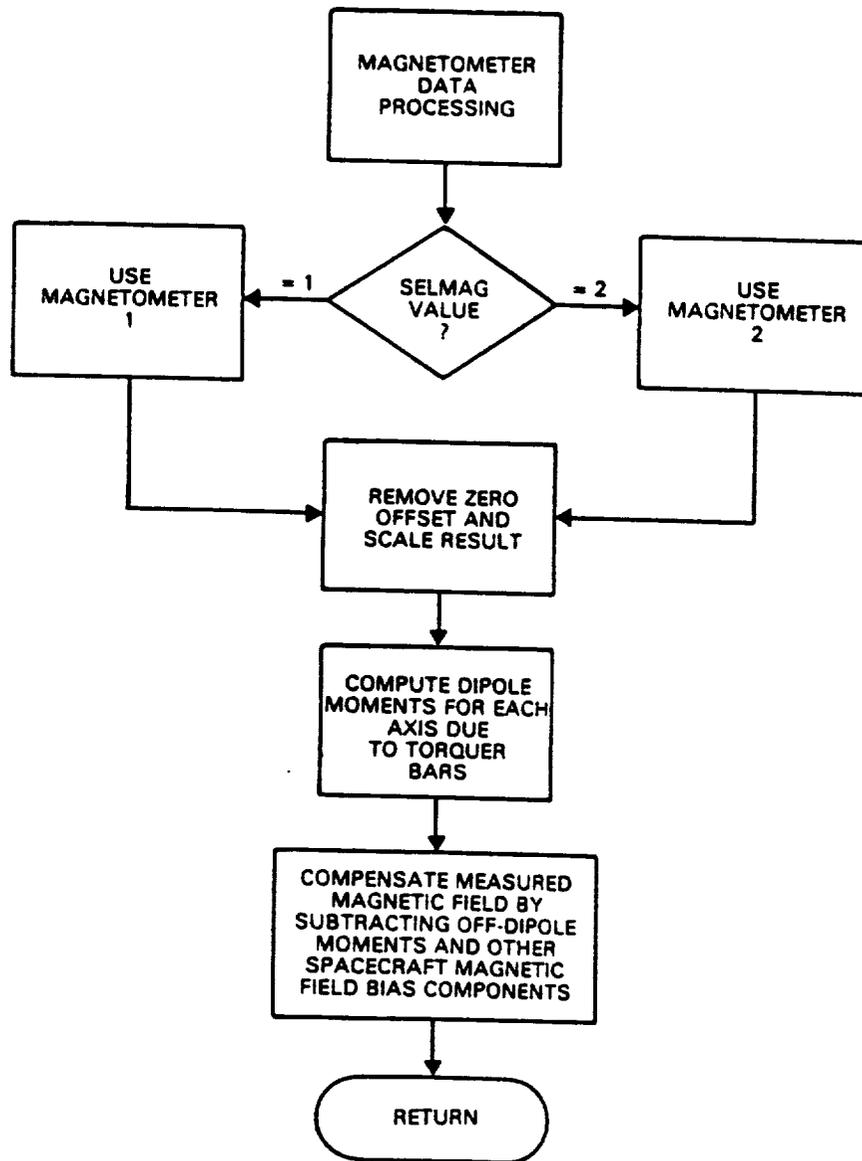
Figure 5-4 shows how magnetometer data processing works. A flag, SELMAG, determines which magnetometer is to be used. The variables used in the following equations are further described in Table 5-2.

B_0 , a zero offset, is converted from counts to gauss, and the TAM data from the Truth Model are converted from teslas to gauss. B_0 is then subtracted from the TAM data:

$$B_0 = B_0 * KMS$$

$$B_j = TTOG * TAM_j - B_0; j = X, Y, Z$$

where KMS = conversion from counts to gauss
 TTOG = conversion from tesla to gauss



9/85 (R/1) 94

Figure 5-4. Magnetic Data Processing Flow

TAM_{i,j} = raw magnetometer measurements for magnetometer i where i = 1 or 2, component j = X, Y, Z

B_j = components of the measured magnetic field j where j = X, Y, Z

The torquer bar contribution to the measured magnetic field is then calculated. Using the MAGFIG flag, the proper torquer bar output is chosen and converted from amp-m² to counts; a torquer bar dipole moment zero offset is then subtracted:

$$MD_i = XTDE_i * DPM(MAGFIG,i) - M_0$$

where XTDE_i = conversion from amp-m² to counts

M₀ = torquer bar dipole moment zero offset (counts)

DPM(i,j) = torquer bar dipole moment measurement in MTA frame; i = X, Y, Z coordinate; j = TDE A/B

In the following three equations, the net magnetic field of the Earth is computed by compensating the measured magnetic field (B_X, B_Y, B_Z) for the effects attributable to the magnetic torquer bars (MD_X, MD_Y, and MD_Z multiplied by a coordinate transformation matrix KB_{XX}, ..., KB_{ZZ} to give the actual contributions with respect to the magnetometer coordinates and converted from counts to gauss) and the magnetic field bias attributable to all other nontorquer bar elements of the spacecraft (BB_X, BB_Y, and BB_Z):

$$BE_X = B_X + (KB_{XX} * MD_X + KB_{XY} * MD_Y + KB_{XZ} * MD_Z) + BB_X$$

$$BE_Y = B_Y + (KB_{YX} * MD_X + KB_{YY} * MD_Y + KB_{YZ} * MD_Z) + BB_Y$$

$$BE_Z = B_Z + (KB_{ZX} * MD_X + KB_{ZY} * MD_Y + KB_{ZZ} * MD_Z) + BB_Z$$

The magnetometer data processing function is updated every 512 msec and is performed prior to updates for either the magnetic control law function or the FHST function.

- Output--The output for the magnetometer data processing module is described in Table 5-2.

5.2.1.6 Star Data Averaging Function

The star data averaging function (STARAVG) either (1) converts the raw FHST data to a form suitable for further processing and computes the average of the four most current star position measurements for each of the two FHSTs or (2) sends out break-track commands to the FHST periodically.

- Input--STARAVG uses the following input:

<u>Name</u>	<u>Description</u>
SCPRS _i	Star presence data from FHST i where i = 1 to 2 (from Truth Model)
SCM _i	Star magnitudes from FHST i where i = 1 to 2 (from Truth Model)
STT _i	FHST temperature data from FHST i where i = 1 to 2 (constant in GROSS OBC but not in flight software)
SCU _i	Raw FHST horizontal measurements, where i = 1 to 2 (from Truth Model)
SCV _i	Raw FHST vertical measurements, where i = 1 to 2 (from Truth Model)
STRAVG _i	Averaging function execution flag for FHST where i = 1 to 2 (from STARAVG, MODECON)
STLIM	Star data comparison threshold (DBC)
TFSCAN _i	Total FOV scan flag for FHST i, where i = 1, 2 (from DBC, MODECON)
TSH _{ij}	Three previous raw FHST horizontal measurements where i = 1 to 2, j = 1 to 3, where j = 3 is the oldest (from STARAVG)
TSPRS _{ij}	Three previous raw FHST i star presence bits where i = 1 to 2, j = 1 to 3, where j = 3 is the oldest (from STARAVG)
TSV _{ij}	Three previous raw FHST vertical measurements where i = 1 to 2, j = 1 to 3, where j = 3 is the oldest (from STARAVG)

● Processing--This function is performed every 256 msec in the standby mode and the normal pointing mode. It is performed before the attitude estimation function (ATTEST).

If $TFSCAN_i = 0$, the data averaging function is performed as follows ($i = 1$ to 2 in following discussions):

$H = SCU_i$

$V = SCV_i$

If $SPRS_i$ is true, $PRS = 1$; otherwise, $PRS = 0$

$FHT_i = STT_i$

$FHM_i = STM_i$

where H = horizontal measurement from selected FHST
 V = vertical measurement from selected FHST
 PRS = star presence flag from selected FHST
 FHT_i = temperature measurement from selected FHST
 FHM_i = star magnitude measurement from selected FHST

The previous star data values are set to the most current star data values if STARAVG was not performed in the previous cycle:

If $STRAVG_i = 0$

$TSPRS_{ij} = PRS$

$TSH_{ij} = H$

$TSV_{ij} = V$

$STRAVG_i = 1$

for $j = 1$ to 3

where $STRAVG_i$ = averaging function execution flag
 $TSPRS_{ij}$ = three previous star presence flags
 TSH_{ij} = three previous horizontal measurements
 TSV_{ij} = three previous vertical measurements

The averages of the four most current tracker H- and V-axis measurements are computed if all four most current star presence flags are set to 1:

If $PRS = 1$ and $TSPRS_{i1} = 1$ and $TSPRS_{i2} = 1$ and $TSPRS_{i3} = 1$
 $FHHA_i = (H + TSH_{i1} + TSH_{i2} + TSH_{i3})/4$
 $FHVA_i = (V + TSV_{i1} + TSV_{i2} + TSH_{i3})/4$
 $IVASD_i = 1$
Otherwise $IVASD_i = 0$

where $FHHA_i$ = averaged horizontal measurement for FHST i
 $FHVA_i$ = averaged vertical measurement for FHST i
 $IVASD_i$ = valid averaged star data flag for FHST i
(0=invalid, 1=valid)

Once the averaging has been performed, the averaged H- and V-axis data are compared to the four most current star H- and V-axis data, respectively. If the difference between the averaged data and any one of the four most current star data exceeds the threshold, the valid averaged star data flag is reset to 0; otherwise, the flag remains set to 1.

If $IVASD_i = 1$
then compare average to the four measurements as shown:

If $| FHHA_i - H | > STLIM$ or
If $| FHHA_i - TSH_{i1} | > STLIM$ or
If $| FHHA_i - TSH_{i2} | > STLIM$ or
If $| FHHA_i - TSH_{i3} | > STLIM$ or
If $| FHVA_i - V | > STLIM$ or
If $| FHVA_i - TSV_{i1} | > STLIM$ or
If $| FHVA_i - TSV_{i2} | > STLIM$ or
If $| FHVA_i - TSV_{i3} | > STLIM$
Then $IVASD_i = 0$

where $STLIM$ = star data comparison threshold

The next step is to update the previous star data values and save the current star data. The variables with $j = 3$ will have the oldest star data:

$$\begin{aligned} \text{TSPRS}_{ij} &= \text{TSPRS}_i^{(j-1)} \text{ for } j = 3, 2 \\ \text{TSH}_{ij} &= \text{TSH}_i^{(j-1)} \text{ for } j = 3, 2 \\ \text{TSV}_{ij} &= \text{TSV}_i^{(j-1)} \text{ for } j = 3, 2 \\ \text{TSPRS}_{i1} &= \text{PRS} \\ \text{TSH}_{i1} &= \text{H} \\ \text{TSV}_{i1} &= \text{V} \end{aligned}$$

This completes the star-averaging portion of STARAVG. If a break track is commanded ($\text{TFSCAN}_i = 1$), the following processing is performed. A total field of view (TFOV) scan with break-track command to the selected FHST is issued every IBKTK computation cycle, and the corresponding valid averaged star data flag is set to zero ($\text{IVASD}_i = 0$):

$$\begin{aligned} \text{ISTCNT}_i &= \text{ISTCNT}_i + 1 \\ \text{If } \text{ISTCNT}_i &= \text{IBKTK} \text{ Then} \\ \text{CBT}(i) &= \text{TRUE} \\ \text{ISTCNT}_i &= 0 \end{aligned}$$

where ISTCNT_i = counter for TFOV function
 IBKTK = break-track command period
 $\text{CBT}(i)$ = break-track command to FHST i

The valid averaged star data flag and the averaging function execution flag are then reset:

$$\begin{aligned} \text{IVASD}_i &= 0 \\ \text{STRAVG}_i &= 0 \end{aligned}$$

- Output--STARAVG produces the following output:

<u>Name</u>	<u>Description</u>
CBT _i	Break-track with TFOV scan commands to FHST i where i = 1 to 2 (to Truth Model)
FHHA _i	Averaged FHST i horizontal measurement where i = 1 to 2 (to ATTEST)
FHM _i	Tracked star intensity (magnitude) where i = 1 to 2 (to ATTEST)
FHT _i	FHST temperature where i = 1 to 2 (to ATTEST)
FHVA _i	Averaged FHST i vertical measurement where i = 1 to 2 (to ATTEST)
ISTCNT _i	TFOV function counter for FHST i where i = 1 to 2 (to STARAVG)
IVASD _i	Valid averaged star data flag for FHST i where i = 1 to 2 (to ATTEST)
STRAVG _i	Averaging function execution flag for FHST i where i = 1 to 2 (to STARAVG)
TSH _{ij}	Three previous raw measurements for FHST i where i = 1 to 2, j = 1 to 3, where j = 3 oldest (to STARAVG)
TSPRS _{ij}	Three previous raw FHST i presence bits where i = 1 to 2, j = 1 to 3, where j = 3 oldest (to STARAVG)
TSV _{ij}	Three previous raw vertical measurements for FHST i where i = 1 to 2, j = 1 to 3, where j = 3 oldest (to STARAVG)

5.2.2 OBSERVATORY SUPPORT

The functions in the observatory support section compute the spacecraft ephemeris used by the antenna pointing command processing function to generate the HGA gimbals angles and gimbals slew commands as well as the spacecraft ephemeris and orbital geometry data used by the attitude estimation function to determine the FHST occultation by the Earth.

5.2.2.1 Ephemeris Computation Function

The ephemeris computation function (EPHEM) computes the current position and velocity of GRO and the positions of two TDRS satellites every 2.048 sec, on the basis of GRO and

TDRS ephemerides computed on the ground and uploaded to the OBC.

- Input--Input for EPHEM is as follows:

<u>Name</u>	<u>Description</u>
GROP _i	GRO current position in ECI coordinates where i = X, Y, Z (from Truth Model)
GROV _i	GRO current velocity in ECI coordinates where i = X, Y, Z (from Truth Model)
TDREP _i	TDRS-East current position in ECI coordinates where i = X, Y, Z (from Truth Model)
TDRWP _i	TDRS-West current position in ECI coordinates where i = X, Y, Z (from Truth Model)

- Processing--No processing is performed in EPHEM.
- Output--Output for EPHEM is as follows:

<u>Name</u>	<u>Description</u>
GROP _i	GRO current position in ECI coordinates where i = X, Y, Z (to ORIENT, OCCULT, ATTEST, ANTCON)
GROV _i	GRO current velocity in ECI coordinates where i = X, Y, Z (to ATTEST)
TDREP _i	TDRS-East current position in ECI coordinates where i = X, Y, Z (to ANTCON)
TDRWP _i	TDRS-West current position in ECI coordinates where i = X, Y, Z (to ANTCON)

5.2.2.2 Spacecraft Orientation Function

The spacecraft orientation function (ORIENT) computes the orientation of the spacecraft +X and +Z axes in astronomic coordinates (right ascension and declination). The computation is simply a coordinate transformation of the spacecraft attitude determined by the ACAD subsystem to right ascension and declination.

The orientation function also computes the elevation and azimuth of the Earth center in spacecraft body coordinates. This calculation is based on the position of the spacecraft in XYZ ECI coordinates from the ephemeris function combined with the spacecraft attitude data from ACAD.

- Input--ORIENT uses the following input:

<u>Name</u>	<u>Description</u>
EPA _i	Euler parameters specifying the attitude of the ACAD body axes with respect to ECI coordinates where i = 1 to 4 (from KININT, DBC/GND)
GROP _i	GRO position in ECI coordinates where i = X, Y, Z (from EPHEM)
QOA _i	Euler parameters (quaternion) specifying the misalignment of the observatory axes from the ACAD optical cube where i = 1 to 4 (data base constant)

- Processing--The computation is performed once every 2.048 sec before execution of the occultation function and after execution of the ephemeris function.

The right ascension and declination of the spacecraft +X and +Z axes are computed from the input ACAD Euler parameters, EPA_i, as follows.

First, the ACAD Euler parameters, EPA_i, are transformed to the observatory Euler parameters, EPO_i, using the ACAD misalignment quaternion, QOA_i:

$$EPO_1 = +QOA_4 * EPA_1 + QOA_3 * EPA_2 - QOA_2 * EPA_3 + QOA_1 * EPA_4$$

$$EPO_2 = -QOA_3 * EPA_1 + QOA_4 * EPA_2 + QOA_1 * EPA_3 + QOA_2 * EPA_4$$

$$EPO_3 = +QOA_2 * EPA_1 - QOA_1 * EPA_2 + QOA_4 * EPA_3 + QOA_3 * EPA_4$$

$$EPO_4 = -QOA_1 * EPA_1 - QOA_2 * EPA_2 - QOA_3 * EPA_3 + QOA_4 * EPA_4$$

The elevation and azimuth of the Earth center in spacecraft coordinates are calculated as follows: Use the observatory Euler parameters, EPO_i , to transform the input spacecraft position vector, $GROP_i$, in ECI coordinates to E_i in spacecraft body Cartesian coordinates (D_i are intermediate variables) and then transform $-E_i$ to spacecraft body spherical coordinates:

$$P_1 = EPO_2 * GROP_3 - EPO_3 * GROP_2$$

$$P_2 = EPO_3 * GROP_1 - EPO_1 * GROP_3$$

$$P_3 = EPO_1 * GROP_2 - EPO_2 * GROP_1$$

$$Q_1 = GROP_1 + 2 * (EPO_4 * P_1 - EPO_3 * P_2 - EPO_2 * P_3)$$

$$Q_2 = GROP_2 + 2 * (EPO_3 * P_1 + EPO_4 * P_2 - EPO_1 * P_3)$$

$$Q_3 = GROP_3 + 2 * (-EPO_2 * P_1 + EPO_1 * P_2 + EPO_4 * P_3)$$

$$GEOAZ = ARCTAN (-Q_2 / -Q_1)$$

$$MAG(Q) = (Q_1 * Q_1 + Q_2 * Q_2 + Q_3 * Q_3) ** 1/2$$

$$GEOEL = ARCCOS (-Q_3 / MAG(Q))$$

The least significant bit for GEOAZ and GEOEL is $1E-4$ radians.

- Output--ORIENT produces the following output:

<u>Name</u>	<u>Description</u>
GEOEL	Elevation of the Earth center measured from the +Z spacecraft axis (to OCCULT)
GEOAZ	Azimuth of the Earth center measured right-handed about the spacecraft +Z axis from the +X axis. (to OCCULT)

5.2.2.3 Earth Occultation Function

The occultation function (OCCULT) determines whether the FHST fields of view are occulted by the Earth. The determination is based on a calculation of the angular difference between the direction to the center of the Earth and the target direction combined with the angles subtended by the Earth and the FHST field of view. It is assumed that the fields of view are conical and the Earth is spherical. Thus, a field of view is occulted when the angular differences are less than the sum of the half-angles subtended by the Earth and the field of view.

The calculations are performed in spacecraft body coordinates. The direction to the Earth center is supplied by the spacecraft orientation function (ORIENT). The angle subtended by the Earth is computed from the orbital GRO XYZ position supplied by the ephemeris computation function. The target directions (in spacecraft body coordinates) and field-of-view angles are data base constants.

- Input--OCCULT uses the following input:

<u>Name</u>	<u>Description</u>
GEOAZ	Azimuth of the Earth center about the spacecraft Z-axis measured from the X-axis (from ORIENT)
GEOEL	Elevation of the Earth center measured from the +Z spacecraft axis (from ORIENT)
GEORAD	Radius of the Earth, defined for this computation to be 6500.0 km (DBC)
GROP _i	GRO position in ECI coordinates where i = X, Y, Z (from EPHEM)
IFOV _i	Half-angle field of view for FHST i where i = 1 to 2 (DBC)
INSAZ _i	Azimuth of the pointing direction of FHST i where i = 1 to 2 (DBC)
INSEL _i	Elevation of the pointing direction of FHST i where i = 1 to 2 (DBC)

● Processing--These computations are done once every 2.048 sec, following execution of the ephemeris and orientation functions. The computational accuracy of this function is 1E-2 radians.

Occultation is defined as an overlap of the solid angles subtended by the Earth and the FHST fields of view. The half-angle subtended by the Earth at the spacecraft (EANG) is computed using the spacecraft orbit radius (ORBRAD) and the radius of the Earth (GEORAD):

$$\text{ORBRAD} = (\text{X} ** 2 + \text{Y} ** 2 + \text{Z} ** 2) ** 1/2$$

where X, Y, Z are the components of GRO position (GROP_i) and

$$\text{EANG} = \text{ARCSIN}(\text{GEORAD}/\text{ORBRAD})$$

Then, for each FHST, the angular difference, ANG_i, between the instrument pointing direction, ANG_i, between the FHST pointing direction (INSEL_i, INSAZ_i), and between the direction to the Earth center (GEOEL, GEOAZ) is computed using the cosine law:

$$\begin{aligned} \text{COS}(\text{ANG}_i) &= \text{COS}(\text{INSEL}_i) * \text{COS}(\text{GEOEL}) + \text{SIN}(\text{INSEL}_i) \\ &* \text{SIN}(\text{GEOEL}) * \text{COS}(\text{INSAZ}_i - \text{GEOAZ}) \end{aligned}$$

for i = 1 to 2

Occultation occurs if ANG_i is less than the sum of the half-angle subtended by the Earth and the half-angle field of view of the instrument:

If ANG_i < EANG + IFOV_i for i = 1 to 2

Then set status flag OCSTAT_i = 1, indicating that FHST i is occulted

Else set status flag OCSTAT_i = 0, indicating that FHST i is not occulted

- Output--OCCULT produces the following output:

<u>Name</u>	<u>Description</u>
OCSTAT ₁	Occultation status of FHST 1 (to ATTEST)
OCSTAT ₂	Occultation status of FHST 2 (to ATTEST)

5.2.3 ATTITUDE DETERMINATION

The GRO attitude determination system consists of the following functions: kinematic integration (Section 5.2.3.2), attitude error computation (Section 5.2.3.3), and attitude estimation (Section 5.2.3.4). Section 5.2.3.1 is a brief system overview of the processing that occurs in each mode of operation.

5.2.3.1 System Overview

5.2.3.1.1 Normal Pointing Mode

The GRO attitude determination system is based on the concept of using precision gyros for high-bandwidth attitude data, with FHSTs or the FSS giving low-bandwidth attitude data that provide periodic attitude and gyro drift updates. A functional block diagram of the attitude determination system is shown in Figure 5-5.

Between updates, the spacecraft attitude is determined from gyro data. The IRU output pulses are accumulated in the gyro pulse counter. They are sampled and are processed by the gyro data acquisition function to detect and compensate for gyro counter overflow, if required, and to prefilter the gyro data (Section 5.2.1.1.1). The data are then processed in the gyro data compensation function to compensate for known drift biases, scale factor, and alignment errors. The spacecraft angular increments and angular rates are also computed (Section 5.2.1.1.2). The spacecraft angular increments are used by the kinematic integration function to propagate a set of

Euler symmetric parameters that specify the attitude of GRO reference axes relative to the ECI coordinate frame (Section 5.2.3.2). The reference attitude is specified in terms of the Euler symmetric parameters that define the orientations of the mission reference frame relative to the ECI coordinate frame. The reference Euler symmetric parameters are then compared with the spacecraft Euler symmetric parameters in the attitude error computation function to compute the attitude error (for roll, pitch, and yaw) vector (Section 5.2.3.3). The attitude error vector is provided to various control processing functions, along with the spacecraft angular rates.

The attitude estimation function uses measurements from FHSTs and the FSS to update the spacecraft attitude and gyro drift biases (Section 5.2.3.4). The normal mode of operation employs measurements from two FHSTs; however, in the event of one FHST failure, the measurements from the other FHST and the FSS are used.

Figure 5-6 shows the attitude estimation algorithm functional block diagram. A functional flow chart is shown in Figure 5-7. The attitude estimation algorithm starts by propagating the estimation error covariance matrix from the time of previous update to the current time. Propagation of the estimation error covariance matrix requires the computation of a Kalman filter state transition matrix and a state noise covariance matrix.

After covariance matrix propagation, the algorithm implements a sensor selection logic to determine the proper sensor to provide the measurements. As in the Landsat-D update filter, the measurements are processed from one sensor at a time. The preferred sensor is the one that has gone the longer time without providing an update. If valid data are

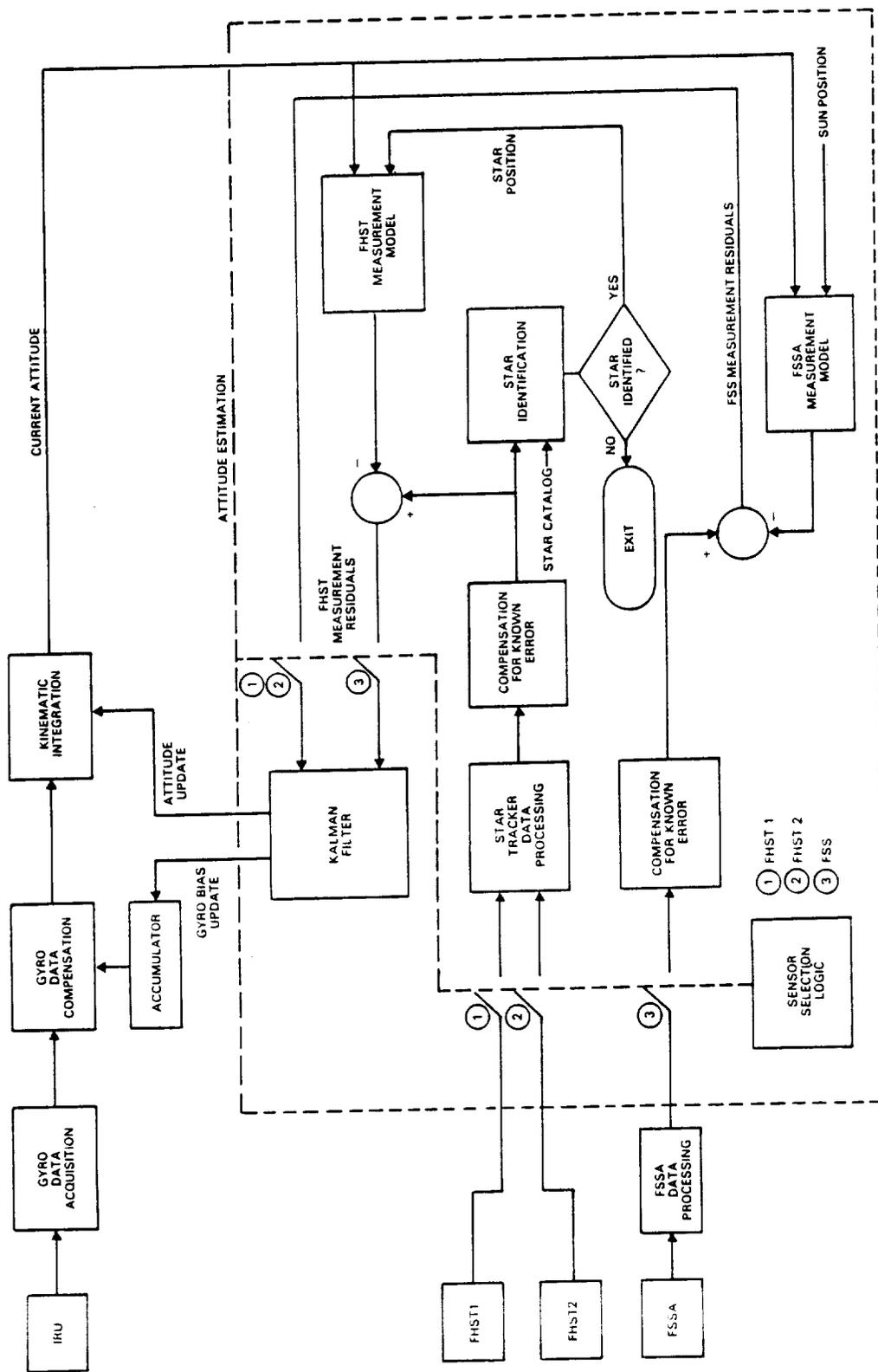


Figure 5-6. Attitude Estimation Algorithm Functional Block Diagram

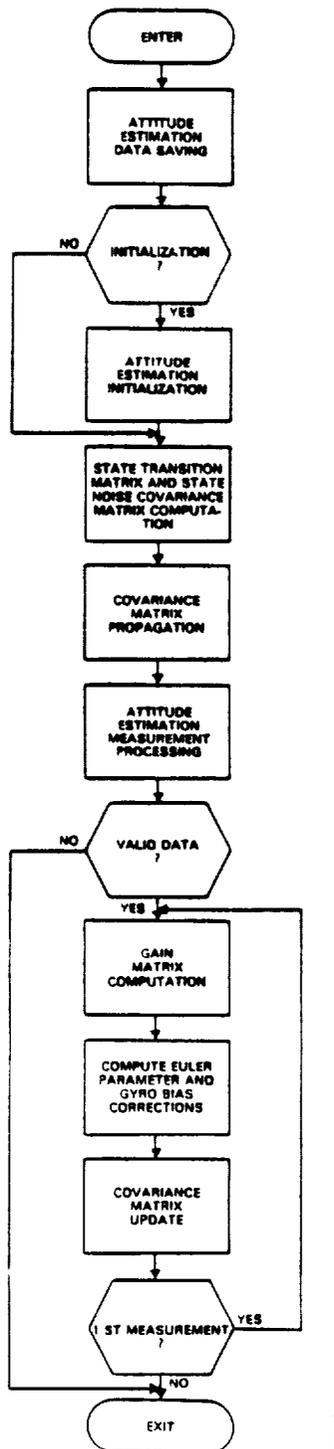


Figure 5-7. Attitude Estimation Software Function

not available from the preferred sensor, the logic checks to see if valid data are available from the other sensor. If so, those data are used in an update; if not, attitude estimation processing is terminated.

If an update is to be performed, the optimal gain matrix is computed; sensor data are used to estimate the attitude and gyro drift bias errors; and the estimation error covariance matrix is updated to reflect the effects of processing the sensor data. The position data from a sensor are considered to consist of a measurement vector containing two elements that are processed sequentially as two scalar measurements. Thus, two passes through the updating cycle are required.

5.2.3.1.2 Standby Mode

During the standby mode, attitude estimation (Section 5.2.3.4) is performed at intervals of 32.768 sec. Kinematic integration (Section 5.2.3.2) and attitude error computation (Section 5.2.3.3) are performed at the cycle time of 512 msec.

5.2.3.1.3 Normal Maneuver Mode

During normal maneuver mode, no attitude estimation (Section 5.2.3.4) is performed. Kinematic integration (Section 5.2.3.2) and attitude error computation (Section 5.2.3.3) are performed at the cycle time of 512 msec.

5.2.3.1.4 Thruster Maneuver Mode

During thruster maneuver mode, no attitude estimation (Section 5.2.3.4) is performed. The gyro data prefilter is disabled, and kinematic integration (Section 5.2.3.2) and attitude error computation (Section 5.2.3.3) are performed at the cycle time of 256 msec.

5.2.3.1.5 Velocity Control Mode

During velocity control mode, no attitude estimation (Section 5.2.3.4) is performed. The gyro prefilter is disabled, and kinematic integration (Section 5.2.3.2) and attitude error computation (Section 5.2.3.3) are performed at the cycle time of 256 msec.

5.2.3.2 Kinematic Integration

The kinematic integration function (KININT) uses the compensated gyro data to propagate the Euler parameters that specify vehicle attitude relative to the ECI coordinate frame. It also uses the results of the attitude estimation function to update the Euler parameters and the gyro bias compensation.

The Euler symmetric parameters are propagated by the algorithms

$$q(t_{m+1}) = \left(\cos B I + \frac{1}{2B} \sin B \Omega_m \right) q(t_m)$$

where q = Euler symmetric parameters

$$B = \theta/2$$

$$\theta = \left(\theta_x^2 + \theta_y^2 + \theta_z^2 \right)^{1/2}$$

θ_i = compensated gyro angular increments where $i = 1$ to 3

I = 4-by-4 unit matrix

$$\Omega_m = \begin{bmatrix} 0 & \theta_3 & -\theta_2 & \theta_1 \\ -\theta_3 & 0 & \theta_1 & \theta_2 \\ \theta_2 & -\theta_1 & 0 & \theta_3 \\ -\theta_1 & -\theta_2 & -\theta_3 & 0 \end{bmatrix}$$

This equation is implemented by expanding the cosine and sine functions in a Taylor series, which is truncated after three terms.

- Input--KININT uses the following input:

<u>Name</u>	<u>Description</u>
EPA _i	ACAD attitude quaternion in ECI coordinates, where i = 1 to 4 (from KININT, Ground)
SE _i	Estimated roll, pitch, and yaw attitude determination errors where i = 1, 2, 3 ACAD body axes (variables from attitude estimation function)
SE _i	Estimated roll, pitch, and yaw drift compensation errors where i = 4, 5, 6 ACAD body axes (variables from attitude estimation function)
THETA _i	Compensated gyro angular increments in roll, pitch, yaw where i = X, Y, Z ACAD body axes (variables from gyro data compensation function)
THETB _i	Gyro drift errors, where i = 1 to 3 ACAD axes (from KININT, ground)
UPDATE	Update filter flag where 0 = no update, 1 = update (flag from attitude estimation function)

- Processing--KININT is executed every 512 msec in the normal pointing and normal maneuver modes and every 256 msec in the velocity control and thruster maneuver modes. It is executed prior to the attitude error computation, attitude and rate command processing, attitude estimation, and antenna pointing command processing functions.

One-half the total gyro increment is computed as follows:

$$B = 0.5 * \left[\left(\text{THETA}_X^2 + \text{THETA}_Y^2 + \text{THETA}_Z^2 \right) ** 0.5 \right]$$

where B = one-half the total gyro angular increment.

One-half $(\sin B)/B$ is computed using an approximation as follows:

$$S = 0.5 * (1.0 - B^2/6 + B^4/120)$$

where S is the approximation to one-half $(\sin B)/B$.

The cosine of B is approximated as follows:

$$CB = (1.0 - B^2/2 + B^4/24)$$

where CB is the approximation to $\cos B$.

Euler parameter increments are computed from the compensated gyro angular increment inputs $THETA_i$ and the values of Euler parameters computed the last cycle in $KININT$ as follows:

$$DA_1 = S * (THETA_Z * EPA_2 - THETA_Y * EPA_3 + THETA_X * EPA_4)$$

$$DA_2 = S * (-THETA_Z * EPA_1 + THETA_X * EPA_3 + THETA_Y * EPA_4)$$

$$DA_3 = S * (THETA_Y * EPA_1 - THETA_X * EPA_2 + THETA_Z * EPA_4)$$

$$DA_4 = S * (-THETA_X * EPA_1 - THETA_Y * EPA_2 - THETA_Z * EPA_3)$$

where DA_i = Euler parameter update increments and $i = 1$
to 4

EPA_i = Euler parameters that specify the ACAD body axis
orientation with respect to the ECI frame and
 $i = 1$ to 4

The Euler parameters are then updated by using these increments after first multiplying by cos B:

$$EPA_i = CB * EPA_i + DA_i$$

for i = 1 to 4.

These Euler parameters are adjusted in sign depending on EPA_4 as follows:

If $EPA_4 < 0$,

$$EPA_i = -EPA_i \quad \text{for } i = 1 \text{ to } 4$$

Otherwise, EPA_i is not changed.

The Euler parameters are normalized:

$$E = 0.5 * \left(3.0 - EPA_1^2 - EPA_2^2 - EPA_3^2 - EPA_4^2 \right)$$

$$EPA_i = E * EPA_i \quad \text{for } i = 1 \text{ to } 4$$

where E is the factor used to normalize the Euler parameters.

Following the above process, KININT checks the update filter flag (UPDATE) from the attitude estimation module.

If UPDATE is zero, the processing in KININT is complete, and the module is exited.

Otherwise, new Euler parameter increments (DAU_i , $i = 1$ to 4) are computed, and the Euler parameters are updated using the above equations, with $THETA_i$ replaced by SE_i as follows:

$$B = 0.5 * \left(SE_1^2 + SE_2^2 + SE_3^2 \right)^{1/2}$$

$$S = 0.5 * (1.0 - B^2/6 + B^4/120)$$

$$CB = (1. - B^2/2 + B^4/24)$$

$$DAU_1 = S * (SE_3 * EPA_2 - SE_2 * EPA_3 + SE_1 * EPA_4)$$

$$DAU_2 = S * (-SE_3 * EPA_1 + SE_1 * EPA_3 + SE_2 * EPA_4)$$

$$DAU_3 = S * (SE_2 * EPA_1 - SE_1 * EPA_2 + SE_3 * EPA_4)$$

$$DAU_4 = S * (-SE_1 * EPA_1 - SE_2 * EPA_2 - SE_3 * EPA_3)$$

where SE_i = estimated roll, pitch, and yaw attitude determination errors and $i = 1$ to 3 ACAD body axes

$$EPA_i = CB * EPA_i + DAU_i \text{ for } i = X, Y, Z$$

If $EPA_4 < 0$,

$$EPA_i = -EPA_i \text{ for } i = 1, 2, 3, 4$$

Otherwise, EPA_i will not change.

$$E = 0.5 * \left(3.0 - EPA_1^2 - EPA_2^2 - EPA_3^2 - EPA_4^2 \right)$$

$$EPA_i = E * EPA_i \text{ for } i = 1 \text{ to } 4$$

Gyro drift errors are then updated as follows:

$$THETB_X = THETB_X + SE_4$$

$$\text{THETB}_Y = \text{THETB}_Y + \text{SE}_5$$

$$\text{THETB}_Z = \text{THETB}_Z + \text{SE}_6$$

where SE_i = estimated roll, pitch, and yaw rate bias compensation errors and $i = 4$ to 6 ACAD body axes

THETB_i = gyro rate compensation for roll, pitch, yaw where $i = X, Y, Z$ ACAD body axes

Finally, the update filter flag is set to zero ($\text{UPDATE} = 0$), and processing is complete.

- Output--KININT produces the following output:

<u>Name</u>	<u>Description</u>
EPA_i	Updated Euler parameters that specify the ACAD body axis orientation with respect to the ECI frame where $i = 1$ to 4 (to ATTEST, ATTERR, ANTCON, ORIENT, KININT)
THETB_i	Gyro drift error in roll, pitch, yaw where $i = X, Y, Z$ ACAD body axes (to GYRODAT, KININT)
UPDATE	Update filter flag; =0, no, =1, yes (to KININT)

5.2.3.3 Attitude Error Computation Function

The attitude error computation function (ATTERR) uses the Euler symmetric parameters and Euler symmetric parameter commands to compute the attitude errors as shown below.

- Input--ATTERR uses the following input:

<u>Name</u>	<u>Description</u>
ACADM	ACAD mode flag where 1 = standby, 2 = normal point, 3 = normal maneuver, 4 = Sun reference point, 5 = safehold, 6 = thruster maneuver, 7 = velocity control (flag set by mode module)
D_i	Euler symmetric parameters that specify the commanded observatory axes with respect to the ECI frame for normal pointing/maneuver modes where $i = 1$ to 4 (DBC's)

<u>Name</u>	<u>Description</u>
DTCP _i	Euler symmetric parameters that specify the commanded observatory axes with respect to the ECI frame for velocity/thruster control modes where i = 1 to 4 (variables from THRUSTER)
EPA _i	Updated Euler parameters that specify the ACAD body axis orientation with respect to the ECI frame where i = 1 to 4 (variables from KININT)
QOA _i	Euler symmetric parameters that specify the observatory axes in the ACAD body axis frame where i = 1-4 (DBC's)

● Processing--ATTERR is executed every 512 msec in the standby, normal pointing, and normal maneuver modes and every 256 msec in the velocity control and thruster maneuver modes. This software function is performed prior to the reaction wheel control and distribution, delta-V thruster control law, and low-level thruster control law software functions.

The attitude commands used by ATTERR are first determined as a function of the ACADMD indicator as follows:

If ACADMD = 1, 2, 3, 4, or 5 (standby, normal pointing, normal maneuver, Sun reference, or safehold modes), then

$$\text{CMDTMP}_i = D_i \quad \text{for } i = 1 \text{ to } 4$$

If ACADMD = 6 or 7 (velocity control or thruster control mode), then

$$\text{CMDTMP}_i = \text{DTCP}_i \quad \text{for } i = 1 \text{ to } 4$$

where CMDTMP_i represents the Euler symmetric parameters (commanded values) used by ATTERR that relate the attitude reference frame to the ECI frame and i = 1 to 4.

The attitude commands are transformed from observatory to ACAD coordinates as follows, i.e., by performing the following quaternion product:

$$q_{\text{Target}}^{\text{ACAD}} = q_{\text{Target}}^{\text{Observatory}} * q_{\text{Observatory}}^{\text{ACAD}}$$

or, expressed as a matrix product:

$$\begin{bmatrix} \text{TARATT}_1 \\ \text{TARATT}_2 \\ \text{TARATT}_3 \\ \text{TARATT}_4 \end{bmatrix} = \begin{bmatrix} -\text{QOA}_4 & \text{QOA}_3 & -\text{QOA}_2 & \text{QOA}_1 \\ -\text{QOA}_3 & -\text{QOA}_4 & \text{QOA}_1 & \text{QOA}_2 \\ \text{QOA}_2 & -\text{QOA}_1 & -\text{QOA}_4 & \text{QOA}_3 \\ -\text{QOA}_1 & -\text{QOA}_2 & -\text{QOA}_3 & -\text{QOA}_4 \end{bmatrix} \begin{bmatrix} \text{CMDTMP}_1 \\ \text{CMDTMP}_2 \\ \text{CMDTMP}_3 \\ \text{CMDTMP}_4 \end{bmatrix}$$

NOTE: A question remains outstanding about the accuracy of this matrix.

If $\text{TARATT}_4 < 0$,

$$\text{TARATT}_i = -\text{TARATT}_i \quad \text{for } i = 1 \text{ to } 4$$

Otherwise, TARATT_i does not change.

Once the target quaternion is referenced with respect to ACAD coordinates, the error quaternion to transform the OBC quaternion to the commanded quaternion is derived as follows:

$$q_{\text{Target}}^{\text{ACAD}} = q_{\text{OBC}}^{\text{ACAD}} * q_E$$

therefore,

$$q_E = q_{\text{OBC}}^{\text{ACAD}^{-1}} * q_{\text{Target}}^{\text{ACAD}}$$

or, expressed as a matrix product:

$$\begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix} = \begin{bmatrix} \text{TARATT}_4 & \text{TARATT}_3 & -\text{TARATT}_2 & \text{TARATT}_1 \\ -\text{TARATT}_3 & \text{TARATT}_4 & \text{TARATT}_1 & \text{TARATT}_2 \\ \text{TARATT}_2 & -\text{TARATT}_1 & \text{TARATT}_4 & \text{TARATT}_3 \\ -\text{TARATT}_1 & -\text{TARATT}_2 & -\text{TARATT}_3 & \text{TARATT}_4 \end{bmatrix} \begin{bmatrix} \text{EPA}_1 \\ \text{EPA}_2 \\ \text{EPA}_3 \\ -\text{EPA}_4 \end{bmatrix}$$

If $C_4 < 0$,

$$C_i = -C_i \quad \text{for } i = 1 \text{ to } 4$$

Otherwise, C_i does not change.

where C_i represents the error quaternion between the observed quaternion and the commanded quaternion, where $i = 1$ to 4.

Finally, the spacecraft body axes errors are computed as

$$E_X = 2 * C_1$$

$$E_Y = 2 * C_2$$

$$E_Z = 2 * C_3$$

where E_i represents the attitude errors in roll, pitch, and yaw, i.e., the commanded-minus-actual angles about the ACAD body, and $i = X, Y, Z$ axes.

- Output--ATTERR produces the following output:

<u>Name</u>	<u>Description</u>
E_i	Attitude errors in roll, pitch, and yaw, i.e., commanded-minus-actual angles about the ACAD body, and $i = X, Y, Z$ axes (to LLTCN, HLTCON, WHECON, SAFECON)

<u>Name</u>	<u>Description</u>
TARATT _i	Euler symmetric parameters that specify the commanded ACAD body axes with respect to the ECI frame for normal pointing/maneuver modes (to ANTCON)

5.2.3.4 Attitude Estimation Function

The attitude estimation function (ATTEST) uses data from the FHST or FSS to update the spacecraft attitude and gyro biases. The normal mode of operation employs data from two FHSTs; in the event of one FHST failure, the data from the other FHST and the FSS will be used. Every 32.768 sec, ATTEST generates roll, pitch, and yaw errors. These errors are fed into the kinematic integration function in place of the normal gyro data that are used between 32.768-sec updates.

The attitude estimation function consists of an extended Kalman filter (KF) that is implemented in two steps. First, the internal statistics are propagated, based on the Dynamics Model; second, the state vector is updated based on the Observation Model and the internal statistics.

Dynamics Model

The gyro rate measurement is assumed to have the following form:

$$\begin{aligned}\dot{\underline{Q}} &= \underline{w} - \underline{b}_o - \underline{b} + \underline{n}_v \\ \underline{\dot{b}} &= \underline{n}_u\end{aligned}$$

where $\dot{\underline{Q}}$ = gyro rate measurement
 \underline{w} = true spacecraft rate
 \underline{b} = gyro random walk error
 \underline{b}_o = gyro bias
 \underline{n}_v = float torque noise (Gaussian white noise)
 \underline{n}_u = float torque derivative noise (Gaussian white noise)

Since \underline{b} is the integral of a white noise, it becomes a random walk. The gyro drift error $\dot{\underline{e}} = \dot{\underline{\Theta}} - \underline{w}$, is formed as follows:

$$\dot{\underline{e}} = -\underline{b}_0 - \underline{b} + \underline{n}_v$$

The gyro bias, \underline{b}_0 , is assumed to be known and can be taken out of the above equation. The attitude error $\underline{\psi}$ is computed as follows: $\dot{\underline{\psi}} + \underline{w} \times \underline{\psi} = \dot{\underline{e}}$. However, since \underline{w} is negligible, the dynamic model is reduced to the following form:

$$\dot{\underline{\psi}} = -\underline{b} + \underline{n}_v$$

$$\dot{\underline{b}} = \underline{n}_u$$

If these two equations are put into a linear state space formulation, the following equations are derived:

$$\dot{\underline{X}}(t) = F \underline{X}(t) + \underline{W}(t)$$

$$\dot{\underline{X}}(t) = \begin{bmatrix} \dot{\underline{\psi}} \\ \dot{\underline{b}} \end{bmatrix} = \begin{bmatrix} 0_{3 \times 3} & -I_{3 \times 3} \\ 0_{3 \times 3} & 0_{3 \times 3} \end{bmatrix} \begin{bmatrix} \underline{\psi} \\ \underline{b} \end{bmatrix} + \begin{bmatrix} \underline{n}_v(3 \times 1) \\ \underline{n}_u(3 \times 1) \end{bmatrix}$$

where $\underline{\psi}$ = attitude error

\underline{b} = gyroscope random walk error

\underline{n}_v = float torque noise (Gaussian)

\underline{n}_u = float torque derivative noise (Gaussian)

The state equation is discretized to the following form:

$$\underline{X}(t_k) = \phi_k \underline{X}(t_{k-1}) + \underline{W}(t_k)$$

where $\phi_k = e^{A T_k}$ and $T_k = t_k - t_{k-1}$.

The two characteristics of $\underline{W}(t)$ are the mean

$$E[\underline{W}(t)] = 0$$

and the covariance

$$E[W(t) \underline{W}^T(t)] = \begin{bmatrix} \underline{n}_v \underline{n}_v^T & 0_{3 \times 3} \\ 0_{3 \times 3} & \underline{n}_u \underline{n}_u^T \end{bmatrix} \delta(t-t')$$

where T denotes the transpose.

Because it is assumed that there is no correlation between \underline{n}_u and \underline{n}_v , the off-diagonal elements in the above equation are zero.

The spectral density matrix is defined as follows:

$$Q(t) = E[W(t) \underline{W}^T(t)]$$

Thus, the covariance is given as

$$Q(t) \delta(t-t')$$

The discrete dynamics noise covariance matrix, Q_k , is obtained using the state transition matrix, ϕ_k , and the spectral density matrix, $Q(t)$, as follows:

$$Q_k = \int_{t_{k-1}}^{t_k} \phi(t_k, t') Q(t') \phi^T(t_k, t') dt'$$

$$= \begin{bmatrix} p_1 & 0 & 0 & p_3 & 0 & 0 \\ 0 & p_1 & 0 & 0 & p_3 & 0 \\ 0 & 0 & p_1 & 0 & 0 & p_3 \\ p_3 & 0 & 0 & p_2 & 0 & 0 \\ 0 & p_3 & 0 & 0 & p_2 & 0 \\ 0 & 0 & p_3 & 0 & 0 & p_2 \end{bmatrix}$$

$$\text{where } p_1 = n_v^2 T_k + 1/3 n_u^2 T_k^3$$

$$p_2 = n_u^2 T_k$$

$$p_3 = -1/2 n_k^2 T_k^2$$

Q_k is used in the propagation of the state covariance matrix, P_k , as follows:

$$P_k(-) = \phi_k P_{k-1}(+) \phi_k^T + Q_k$$

where $P_k(-)$ = propagated covariance matrix at time k
 $P_{k-1}(+)$ = updated covariance matrix at time $k-1$

Observation Model

FHST Model

In the GRO flight software, the FHST measurements are used to create an observed star unit vector, OS, in the sensor coordinate frame. The identified star position in the star catalog is used to create an expected or computed unit star vector, CS, in the sensor coordinate frame. The following is then defined:

$$z_k(i) = OS_k(i) - CS_k(i) \quad \text{for } i = 1 \text{ to } 2$$

where i is the i th coordinate of the vectors and z_k is the measurements residuals.

From this definition of z_k , H_k is shown to be

$$H_k = \begin{bmatrix} (X \times S_k)^T & 0_{1 \times 3} \\ (Y \times S_k)^T & 0_{1 \times 3} \end{bmatrix}$$

where \underline{S}_k = observed star in spacecraft body frame
 \underline{X} = FHST X-coordinate axis in the spacecraft frame
 \underline{Y} = FHST Y-coordinate axis in the spacecraft frame

In the observation model,

$$\underline{Z}_k = H_k \underline{X} + \underline{V}_k$$

where \underline{Z}_k is the observation and \underline{V}_k is the sensor noise (Gaussian).

The sensor noise characteristics are as follows:

$$E[\underline{V}_k] = 0$$

$$R_k = E[\underline{V}_k \underline{V}_k^T] = \begin{bmatrix} R_{11} & 0 \\ 0 & R_{22} \end{bmatrix}$$

It is further assumed that the initial state vector, \underline{X}_0 , is Gaussian and that \underline{X}_0 , W , and \underline{V}_k are independent of each other. Because all are assumed zero mean and Gaussian, this is equivalent to assuming they are uncorrelated with each other.

FSS Model¹

As with the FHST, the FSS model uses an observed Sun position, OS, and a computed Sun position, CS, to compute measurement residuals, Z, as follows:

$$\underline{Z}_k(i) = \underline{OS}_k(i) - \underline{CS}_k(i) \quad \text{for } i = x \text{ and } y$$

¹Reference 15 describes the FSS measurement model in more detail.

The measurement equation is the same as that for the FHST equation. For the FSS, the H_k is shown to be

$$H_k = \begin{bmatrix} (\underline{X}_{MP} \times \underline{S}_k)^T & 0_{1 \times 3} \\ (\underline{Y}_{MP} \times \underline{S}_k)^T & 0_{1 \times 3} \end{bmatrix}$$

where \underline{S}_k is the computed Sun vector.

\underline{X}_{MP} and \underline{Y}_{MP} are defined as follows:

$$\begin{aligned} X_{MPx} &= (X_{Fx} - Z_{Fx} XP) / (\hat{Z}_F \cdot \underline{S}_k) \\ X_{MPy} &= (X_{Fy} - Z_{Fy} XP) / (\hat{Z}_F \cdot \underline{S}_k) \\ X_{MPz} &= (X_{Fz} - Z_{Fz} XP) / (\hat{Z}_F \cdot \underline{S}_k) \\ Y_{MPx} &= (Y_{Fx} - Z_{Fx} YP) / (\hat{Z}_F \cdot \underline{S}_k) \\ Y_{MPy} &= (Y_{Fy} - Z_{Fy} YP) / (\hat{Z}_F \cdot \underline{S}_k) \\ Y_{MPz} &= (Y_{Fz} - Z_{Fz} YP) / (\hat{Z}_F \cdot \underline{S}_k) \end{aligned}$$

where \underline{X}_F = FSS X-coordinate axis in the spacecraft frame
 \underline{Y}_F = FSS Y-coordinate axis in the spacecraft frame
 \underline{Z}_F = FSS Z-coordinate axis in the spacecraft frame
 XP and YP = FSS expected measurements

The sensor noise characteristics for the FSS are the same as those for the FHST.

Update Algorithms

The state vector is updated by processing the following equation with the inputs $P_k(-)$, H_k , R_k , and the observation vector, Z_k :

$$K_k = P_k(-) H_k^T \left[H_k P_k(-) H_k^T + R_k \right]^{-1}$$

where K_k is the Kalman gain matrix:

$$P_k(+)= (I - K_k H_k) P_k(-)$$

where $P_k(+)$ is the updated covariance matrix:

$$X_k(+)= X_k(-) + K_k(Z_k - H_k X_k(-))$$

where $X_k(+)$ is the updated state vector.

The GRO flight software employs a scalar implementation method that requires the above sequence of equations to be executed twice. In the first pass, the following substitutions are made for the FHSTs (or similarly the FSS):

$$H_k = H_{k,1} = \left[(\underline{X} \times \underline{S}_k)^T \quad 0_{1 \times 3} \right]$$

$$R_k = R_{k,1} = R_{11}$$

The resulting Kalman gain matrix, $\underline{K}_{k,1} = K_k$, is used to update the covariance matrix, where $P_{k,1} = P_k$, and its update, where $\underline{X}_k(-) = 0$. The equations are as follows:

$$\underline{K}_{k,1} = P_k(-) H_{k,1}^T / \left[H_{k,1} P_k(-) H_{k,1}^T + R_{k,1} \right]$$

$$P_{k,1}(+) = [I - \underline{K}_{k,1} H_{k,1}] P_k(-)$$

$$\underline{X}_{k,1}(+) = \underline{K}_{k,1} Z_k$$

In the second pass, the following substitutions are made:

$$H_k = H_{k,2} = \left[(\underline{Y} \times \underline{S}_k)^T \quad 0_{1 \times 3} \right]$$

$$R_k = R_{k,2} = R_{22}$$

$$K_k = \underline{K}_{k,2}$$

$$P_k(-) = P_{k,1}$$

$$\underline{X}_k(-) = \underline{X}_{k,1}(+)$$

where $\underline{X}_{k,1}(+)$ is the state vector update from the first pass.

The final Kalman gain matrix, $\underline{K}_k = \underline{K}_{k,2}$, is used to update the covariance matrix, $P_k(+) = P_{k,2}(+)$, and the state $\underline{X}_k(+) = \underline{X}_{k,2}(+)$. The equations are as follows:

$$\underline{K}_{k,2} = P_{k,1}(+) H_{k,2}^T / \left[H_{k,2} P_{k,1}(+) H_{k,2}^T + R_{k,2} \right]$$

$$P_{k,2}(+) = [I - \underline{K}_{k,2} H_{k,2}] P_{k,1}(+)$$

$$\underline{X}_k(+) = \underline{X}_k(-) + \underline{K}_{k,2} [Z_{k,2} - H_{k,2} \underline{X}_k(-)]$$

where $Z_{k,2}$ is the Y-component of Z_k .

• Input--The attitude estimation function uses the following data:

<u>Name</u>	<u>Description</u>
AEOFF	Attitude estimation override flag; = 0, ATTEST on, = 1, ATTEST off (variable from DBC/GND)
BE _i	Components of Earth's magnetic field vector along spacecraft axes where i = X, Y, Z (variables from MAGPROC)
CI	Inverse of the speed of light (DBC's)
CMDBKT	Flag that determines if break track commands are allowed; = 0, commands not allowed, = 1, commands allowed (from MODECON)
EPA _i	Updated Euler parameters that specify the ACAD body axis orientation with respect to ECI frame where i = 1 to 4 (variables from KININT, DBC/GND)

<u>Name</u>	<u>Description</u>
FHHA _i	Averaged FHST i horizontal measurement where i = 1 to 2 (from STARAVG)
FHM _i	FHST i star intensity measurement where i = 1 to 2 (from STARAVG)
FHT _i	FHST i temperature measurement where i = 1 to 2 (from STARAVG)
FHVA _i	Averaged FHST i vertical measurement where i = 1 to 2 (from STARAVG)
FHST	FHST selection; 1 = FHST 1, 2 = FHST 2 (from DBC/GND, ATTEST)
FLTINIT	Filter initialization flag (= 0, no initialization, = 1, initialization) (variable flag from MODECON, ATTEST)
GROP _i	GRO position in ECI coordinates where i = X, Y, Z) (variables from EPHEM)
GROV _i	GRO velocity in ECI coordinates where i = X, Y, Z) (variables from EPHEM)
HN	FSS head select, bit 16 of FSS 2; = 0, sensor 1, = 1, sensor 2 (variable flag from FSSPROC)
HRFOV _{ij}	Horizontal offset pointing coordinates for FHST i where i = 1 to 2, j = 1 to 5 (DBC)
IS	Tracked star intensity (magnitude) (variable from STARDAT)
ISL _i	Star catalog data, ith star magnitude lower limit where i = 1 to NCAMAX (DBCs)
ISU _i	Star catalog data, ith star magnitude upper limit where i = 1 to NCAMAX (DBCs)
IVASD _i	Valid averaged star data flag for FHST i where i = 1 to 2 (from STARAVG)
KS	Nominal standard deviation for FHST acceptable accuracy (DBCs)
LSIX _i	Star catalog data, X-component of ith star unit vector in ECI frame where i = 1 to NCAMAX (DBCs)
LSIY _i	Star catalog data, Y-component of ith star unit vector in ECI frame where i = 1 to NCAMAX (DBCs)
LSIZ _i	Star catalog data, Z-component of ith star unit vector in ECI frame where i = 1 to NCAMAX (DBCs)
MAXFOV _i	Maximum value of pointer NFOV _i for FHST i where i = 1 to 2 (DBC)

<u>Name</u>	<u>Description</u>
MD _i	Components of torquer bar generated dipole moment in antenna boom coordinates where i = X, Y, Z (variables from MAGPROC)
NCAMAX	Number of stars in the OBC star catalog (DBC)
NFALIM	Star identification failure threshold (DBC)
NFOVO _i	Initial setting for pointer NFOV _i where i = 1 to 2 (DBC)
OCSTAT _i	Occultation status for FHSTs where i = 1 for FHST 1, i = 2 for FHST 2 (from OCCULT)
OVRFOV _i	Override flag for FHST offset pointing coordinate selection for FHST i where i = 1 to 2 (from DBC/GND)
PA ₀	Initial attitude error variance (DBC)
PG ₀	Initial gyro bias variance (DBC)
RF	FSS measurement error variance (DBC)
RL	FHST measurement error variance for stars near the tracker origin (DBC)
RLM	Measured star radius squared from FHST boresight for computation of measurement error variance (DBC)
RU	FHST measurement error variance for stars far from the FHST origin (DBC)
RSN	Inverse of the mean distance from the Earth to the Sun (DBC)
SCP11 _i	Nominal value of maximum exponent of attitude covariance elements PM11 _i where i = 1 to 9 (DBC)
SI _i	Components of the Sun pointing unit vector from Earth in ECI coordinates where i = X, Y, Z (variables from EPHEM)
ST _j	FHST/FSS configuration flags where j = 1 to 4 (data base flags) (see Table 5-4 for definition of bits) (DBC/GND)
SUNPRS	Sun presence flag, bit 32 of FSS4 (= 0, not present, = 1, present) (flag from FSS processor)
TB	Tangent of beta Sun angle in FSS coordinates of head in use (variable from FSS PROC)
TA	Tangent of alpha Sun angle in FSS coordinates of head in use (variable from FSS PROC)
TF17	OBC clock time (variable from EXEC)

<u>Name</u>	<u>Description</u>
TFMOD	OBC clock time modulus (DBC)
TFSCAN _i	Total FOV scan flag for FHST i where i = 1 to 2; = 1, TFOV scan, = 0, no TFOV scan (DBC/GND)
TMD	Average FHST measurement time delay (DBC)
TPD	Maximum allowed change in propagation interval without recomputing state transition matrix (DBC)
TPSNOM	Nominal value of propagation time interval (DBC)
VE _i	Components of Earth orbital velocity in ECI coor- dinates where i = X, Y, Z (variables from EPHEM)
VG _i	Gyro white noise drift for i = X, Y, Z axes (DBC)
VLOW	Minimum allowed value of FHST measurement resid- ual acceptable accuracy (DBC)
VHIGH	Maximum allowed value of FHST measurement resid- ual acceptable accuracy (DBC)
VR _i	Gyro random walk drift variances where i = X, Y, Z (DBCs)
W _i	Compensated spacecraft body rates in roll, pitch, yaw where i = X, Y, Z body axes (variables from GYRODAT)
XF _{ij}	Components of ith FSS head X-axis unit vector along ACAD body axes where j = 1 to 3, i = 1, 2 (DBC)
XPC _L	Lower FSS field-of-view limit for tangent of beta angle (DBC)
XPC _U	Upper FSS field-of-view limit for tangent of beta angle (DBC)
XSC	Compensated FHST LOS unit vector component along FHST X-axis (variables from star data processing)
XT _i	Components of FHST X-axis unit vector along i ACAD body axis where i = X, Y, Z (variables from star data processing)
YF _{ij}	Components of ith FSS head Y-axis unit vector along ACAD body axes where j = 1 to 3, i = 1, 2 (DBCs)
YPC _L	Lower FSS FOV limit (tangent of angle) (DBC)
YPC _U	Upper FSS FOV limit (tangent of angle) (DBC)
YSC	Compensated FHST LOS unit vector component along FHST Y-axis (variable from star data processing)

<u>Name</u>	<u>Description</u>
YT _i	Components of FHST Y-axis unit vector along ith ACAD body axis where i = X, Y, Z (variables from star data processing)
ZF _{ij}	Components of ith FSS head Z-axis unit vector along ACAD body axes where j = 1 to 3, i = 1, 2 (DBC's)
ZT _i	Components of FHST Z-axis unit vector along ith ACAD body axis where i = X, Y, Z (variables from star data processing)

● Processing--The attitude estimation function is executed every 32.768 sec. Attitude estimation data saving is accomplished at such a time that a consistent set of ACS data is saved. Figure 5-8 shows the processing flow of the attitude estimation function.

Notations for Attitude Estimation--This subsection defines notation used in the remaining subsections and does not contain any specific functional requirements to be implemented. In addition to general notation definitions, the state transition, error covariance, and state noise covariance matrices and their elements are defined.

Matrix [X] denotes a general 3-by-3 matrix. The elements of the matrix are defined as follows:

$$[X] = \begin{bmatrix} X_1 & X_4 & X_7 \\ X_2 & X_5 & X_8 \\ X_3 & X_6 & X_9 \end{bmatrix}$$

It is assumed that the elements of the matrices are stored columnwise.

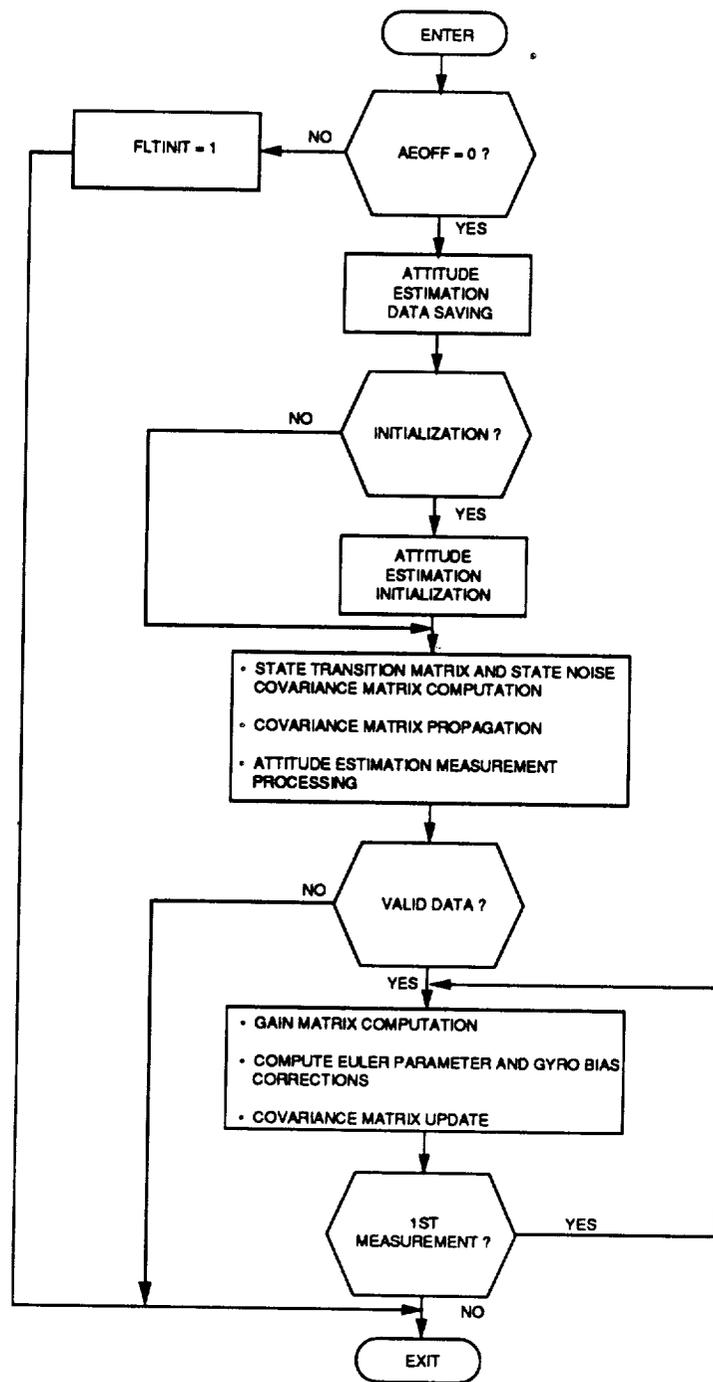


Figure 5-8. Attitude Estimation Overview Flow Chart

Vector--A general three-dimensional vector is denoted as V. The elements of the vector are defined as follows:

$$\vec{V} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

or

$$\vec{V} = (v_x, v_y, v_z)$$

Transpose

XPOSE [X] = Transpose of the matrix [X].

Dot Product

$$\text{DOT } (\vec{A}, \vec{B}) = \vec{A} \cdot \vec{B}$$

Cross Product

$$\text{CROSS } (\vec{A}, \vec{B}) = \vec{A} \times \vec{B}$$

Vector Coordinate Transformation--The vector coordinate transformation transforms a vector from one coordinate frame to another when the Euler symmetric parameters that related the two frames are known. Let \vec{A} be the vector to be transformed; \vec{B} the resulting transformed vector; $\vec{Q} = [Q_1, Q_2, Q_3, Q_4]$ the Euler symmetric parameters; and J, a flag indicating whether the direct or inverse transformation is to be used.

The function is denoted as

$$\vec{B} = \text{XFORM } (\vec{A}, \vec{Q}, J)$$

The processing is defined as follows: Save Q_4 in a temporary location:

$$X = Q_4$$

If $J = -1$, set $X = -X$; otherwise, there is no change to X .

The following computations are performed:

$$D_1 = Q_2 * A_3 - Q_3 * A_2$$

$$D_2 = Q_3 * A_1 - Q_1 * A_3$$

$$D_3 = Q_1 * A_2 - Q_2 * A_1$$

$$B_1 = A_1 + 2 * (X * D_1 - Q_3 * D_2 + Q_2 * D_3)$$

$$B_2 = A_2 + 2 * (X * D_2 - Q_1 * D_3 + Q_3 * D_1)$$

$$B_3 = A_3 + 2 * (X * D_3 - Q_2 * D_1 + Q_1 * D_2)$$

State Transition Matrix--The state transition matrix is partitioned into three 3-by-3 matrices, defined as follows:

$$\text{State transition matrix} = \begin{bmatrix} [FM_{11}] & [FM_{12}] \\ 0 & [I] \end{bmatrix}$$

$$\text{where } [FM_{11}] = \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix}$$

$$[FM_{12}] = \begin{bmatrix} F_{14} & F_{15} & F_{16} \\ F_{24} & F_{25} & F_{26} \\ F_{34} & F_{35} & F_{36} \end{bmatrix}$$

[I] = a 3-by-3 identity matrix

F_{ij} = elements of the state transition matrix that relate the attitude errors to the initial attitude errors for $i = 1$ to 3; $j = 1$ to 3. For $j = 4$ to 6, these relate attitude errors to the gyro bias type of errors.

Error Covariance Matrix--The error covariance matrix is partitioned into four 3-by-3 matrices. The following diagram defines the notation

$$\text{Error covariance matrix} = \begin{bmatrix} [PM_{11}] & [PM_{12}] \\ [PM_{21}] & [PM_{22}] \end{bmatrix}$$

$$\text{where } [PM_{11}] = \begin{bmatrix} PM_{111} & PM_{114} & PM_{117} \\ PM_{112} & PM_{115} & PM_{118} \\ PM_{113} & PM_{116} & PM_{119} \end{bmatrix}$$

$$[PM_{12}] = \begin{bmatrix} PM_{121} & PM_{124} & PM_{127} \\ PM_{122} & PM_{125} & PM_{128} \\ PM_{123} & PM_{126} & PM_{129} \end{bmatrix}$$

$$[PM_{21}] = \text{XPOSE } [PM_{12}]$$

$$[PM_{22}] = \begin{bmatrix} PM_{221} & PM_{224} & PM_{227} \\ PM_{222} & PM_{225} & PM_{228} \\ PM_{223} & PM_{226} & PM_{229} \end{bmatrix}$$

- PM_{11i} = elements of the attitude error covariance matrix, where i = 1 to 9
- PM_{12i} = elements of the cross covariance matrix between the attitude and gyro bias errors, where i = 1 to 9
- PM_{22i} = elements of the gyro bias error covariance matrix, where i = 1 to 9

State Noise Covariance Matrix--The state noise covariance matrix is a symmetric matrix whose elements are defined as follows:

$$\text{State noise covariance matrix} = \begin{bmatrix} W_{111} & W_{112} & W_{113} & W_{121} & W_{124} & W_{127} \\ W_{112} & W_{114} & W_{115} & W_{122} & W_{125} & W_{128} \\ W_{113} & W_{115} & W_{116} & W_{123} & W_{126} & W_{129} \\ W_{121} & W_{124} & W_{127} & W_{221} & 0 & 0 \\ W_{122} & W_{125} & W_{128} & 0 & W_{222} & 0 \\ W_{123} & W_{126} & W_{129} & 0 & 0 & W_{223} \end{bmatrix}$$

where W_{ijk} represents elements of state noise covariance matrix.

Attitude Estimation Data Saving--Table 5-3 lists the input data variables that are saved at the beginning of ATTEST and are then used during the computation cycle of ATTEST. The acronyms under which the variables are saved and used in ATTEST are also shown in the table. These data will be obtained at times that result in a consistent data set. The saved data that are outputs of KININT, MAGPROC, EPHEM, GYROCOMP, FSSPROC, and OBCEXEC are sampled after these modules have executed. The FHST data is sampled at a consistent time near the sampling time of the other data.

Table 5-3. Saved Attitude Estimation Data

DATA DESCRIPTION	DATA SOURCE	INPUT DATA SYMBOL	SAVED DATA SYMBOL
FHST DATA: HORIZONTAL MEASUREMENT VERTICAL MEASUREMENT FHST INTENSITY FHST TEMPERATURE VALID AVERAGED STAR DATA FLAG	STARAVG	FHHA _i FHVA _i FHM _i FHT _i IVASO _i	FHHAS _i FOR i = 1, 2; j = 1, 2 FHVAS _i FOR i = 1, 2; j = 1, 2 FHMS _i FOR i = 1, 2 FHTS _i FOR i = 1, 2 STARU _i FOR i = 1, 2
ECI EULER PARAMETERS	KININT	EPA _i	SEPA _i FOR i = 1 to 4
MAGNETIC FIELD COMPONENTS	MAGPROC	BE _i	BES _i FOR i = X, Y, Z
MAGNETIC TORQUER BAR MOMENTS	MAGPROC	MD _i	MDS _i FOR i = X, Y, Z
ORBIT POSITION	EPHEM	GROP _i	RXS _i FOR i = 1 TO 3
ORBIT RATE	EPHEM	GROV _i	VXS _i FOR i = 1 TO 3
SPACECRAFT BODY RATES	GYRCOMP	W _i	WU _i FOR i = X, Y, Z
FSS DATA	FSSPROC	SUNPRS HN TA TB	SUNPS HEAD XPC YPC
SUN UNIT VECTOR	EPHEM	SI _i	SIXS _i FOR i = 1 TO 3
EARTH ORBITAL VELOCITY	EPHEM	VE _i	VEXS _i FOR i = TO 3
OCCULTATION STATUS	OCCULT	OCSTAT _i	OCSTATS _i FOR i = 1, 2; j = 1, 2
OBC CLOCK DATA	OBCEXEC	TU	

5132G(4)-23

Attitude Estimation Override by Ground Command--The ATTEST function is not to be executed if the AEOFF flag is set by ground command:

```
If AEOFF = 1
    FLTINIT = 1
    Exit ATTEST
```

where AEOFF = attitude estimation off-flag
FLTINIT = update filter initialization flag

Otherwise, process ATTEST function.

Attitude Estimation Initialization--If the attitude estimation initialization flag FLTINIT is set to one, the following attitude estimation initialization computations will be performed:

$$PM_{11i} = 0 \quad i = 1, 2, \dots, 9$$

$$PM_{111} = PM_{115} = PM_{119} = PA_0$$

$$PM_{12i} = 0 \quad i = 1, 2, \dots, 9$$

$$PM_{22i} = 0 \quad i = 1, 2, \dots, 9$$

$$PM_{221} = PM_{225} = PM_{229} = PG_0$$

$$TPS = 0$$

$$FLTINIT = 0$$

$$TUS = TU - TPSNOM$$

$$\text{NFAIL}_i = 0 \quad i = 1,2$$

$$\text{LSTID}_i = 0 \quad i = 1,2$$

$$\text{NFOV}_i = \text{NFOVO}_i \quad i = 1,2$$

$$\text{FSS} = 0$$

where TPS = past value of covariance matrix propagation interval

TUS = value of TU at last covariance matrix propagation

TU = saved OBC clock time (see Table 5-3)

If FLTINIT is not equal to one, no initialization computations are performed.

State Transition and State Noise Matrices--The state transition and state noise covariance matrices are a function of the propagation time interval, which is computed as follows:

$$\text{TPI} = \text{TU} - \text{TUS}$$

where TP is the covariance matrix propagation interval.

If $|\text{TP} - \text{TPS}| \leq \text{TPD}$, the change of the propagation interval is small, and the state noise covariance matrix is not recomputed. The remainder of the processing in this subsection is not performed in this case. In the above, TPD is the maximum allowed change in propagation interval (without recomputing the state transition matrix).

If $|\text{TP} - \text{TPS}| > \text{TPD}$, the change of the propagation interval is large, and the remaining computations in this subsection are performed:

The current propagation interval is saved as follows:

$$\text{TPS} = \text{TP}$$

where TPS is the past value of covariance matrix propagation interval.

Intermediate quantities used in computing the state transition and state noise covariance matrices are computed:

$$T_2 = \text{TP}^2/2$$

$$T_3 = \text{TP}^3/3$$

The elements of the state transition matrix are computed:

$$F_{11} = 1$$

$$F_{12} = 0$$

$$F_{13} = 0$$

$$F_{21} = 0$$

$$F_{22} = 1$$

$$F_{23} = 0$$

$$F_{31} = 0$$

$$F_{32} = 0$$

$$F_{33} = 1$$

$$F_{14} = -\text{TP}$$

$$F_{15} = 0$$

$$F_{16} = 0$$

$$F_{24} = 0$$

$$F_{25} = -TP$$

$$F_{26} = 0$$

$$F_{34} = 0$$

$$F_{35} = 0$$

$$F_{36} = -TP$$

The elements of the state noise covariance matrix are computed:

$$W_{111} = VG_X * TP + VR_X * T_3$$

$$W_{112} = 0$$

$$W_{113} = 0$$

$$W_{114} = VG_Y * TP + VR_Y * T_3$$

$$W_{115} = 0$$

$$W_{116} = VG_Z * TP + VR_Z * T_3$$

$$W_{121} = -VR_X * T_2$$

$$W_{122} = 0$$

$$W_{123} = 0$$

$$W_{124} = 0$$

$$W_{125} = -VR_Y * T_2$$

$$W_{126} = 0$$

$$W_{127} = 0$$

$$W_{128} = 0$$

$$W_{129} = -VR_Z * T_2$$

$$W_{221} = VR_X * TP$$

$$W_{222} = VR_Y * TP$$

$$W_{223} = VR_Z * TP$$

where VR_i represents gyro random walk drift variances and $i = X, Y, Z$.

Covariance Matrix Propagation--The attitude estimation error covariance matrix propagation algorithm is specified as follows:

Compute various intermediate matrices:

$$[X_1] = [FM_{11}] [PM_{11}]$$

$$[X_2] = XPOSE [FM_{11}]$$

$$[PM_{11}] = [X_1] [X_2]$$

$$[X_1] = [FM_{11}] [PM_{12}]$$

$$[X_2] = [FM_{12}] [PM_{22}]$$

Propagate elements of $[PM_{12}]$:

$$PM_{12i} = X_{1i} + X_{2i} + W_{12i} \quad i = 1 \text{ to } 9$$

Compute various intermediate matrices:

$$[X_3] = XPOSE [FM_{12}]$$

$$[X_4] = [X_1] [X_3]$$

$$[X_1] = [X_2] [X_3]$$

Propagate elements of $[PM_{11}]$:

$$PM_{111} = PM_{111} + 2 * X_{41} + X_{11} + W_{111}$$

$$PM_{115} = PM_{115} + 2 * X_{45} + X_{15} + W_{114}$$

$$PM_{119} = PM_{119} + 2 * X_{49} + X_{19} + W_{116}$$

$$PM_{112} = PM_{112} + X_{42} + X_{44} + X_{12} + W_{112}$$

$$PM_{113} = PM_{113} + X_{43} + X_{47} + X_{13} + W_{113}$$

$$PM_{116} = PM_{116} + X_{46} + X_{48} + X_{16} + W_{115}$$

$$PM_{114} = PM_{112}$$

$$PM_{117} = PM_{113}$$

$$PM_{118} = PM_{116}$$

Propagate elements of $[PM_{22}]$:

$$PM_{221} = PM_{221} + W_{221}$$

$$PM_{225} = PM_{225} + W_{222}$$

$$PM_{229} = PM_{229} + W_{223}$$

$$PM_{224} = PM_{222}$$

$$PM_{227} = PM_{223}$$

$$PM_{228} = PM_{226}$$

Measurement Processing--The measurement processing requirements are presented in Figure 5-9 and the paragraphs of this subsection as referenced by Figure 5-9. Measurement processing also calls the STARDAT module.

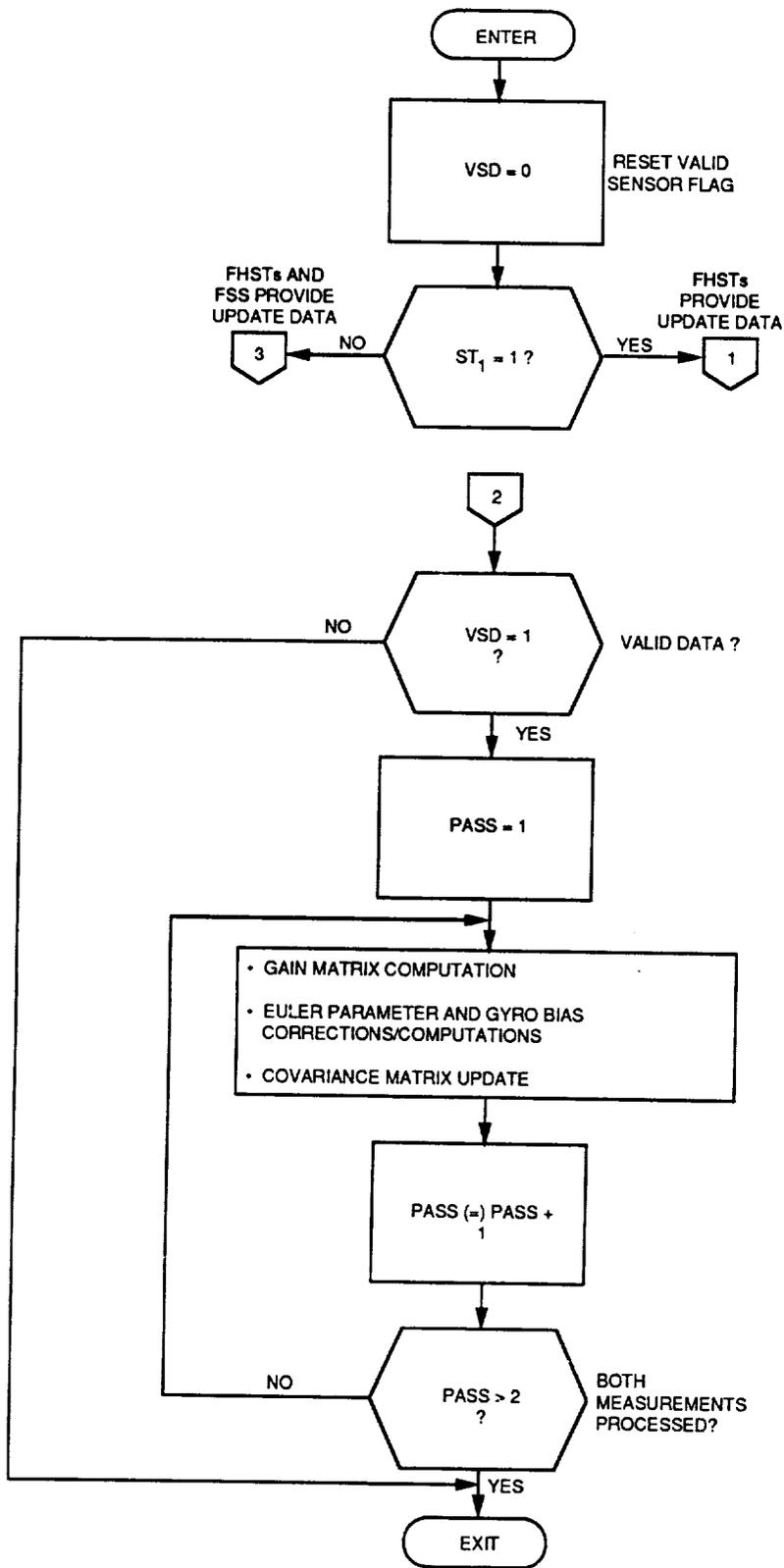
The measurement processing logic determines if valid sensor data are available, and if so, selects the proper sensor to provide the update data. Two attitude estimation sensor configurations are provided for as indicated by the sensor configuration flag, ST. (Table 5-4 defines the bits of ST and of the other flags and counters used in ATTEST.) The logic determines if valid data are available from the sensor that has gone the longest time without providing update data. If valid data from this sensor are available, they are used in the update. If valid data from the preferred sensor are not available, the logic checks to see if valid data are available from the other sensor. If so, these data are used in the update. If neither sensor has valid data, measurement processing is terminated.

FHST Data Correction--The FHST data correction function compensates the FHST measurements for the effects of velocity aberration and measurement time delay. The input to this function consists of the compensated FHST measurements from STARDAT.

The required processing is as follows:

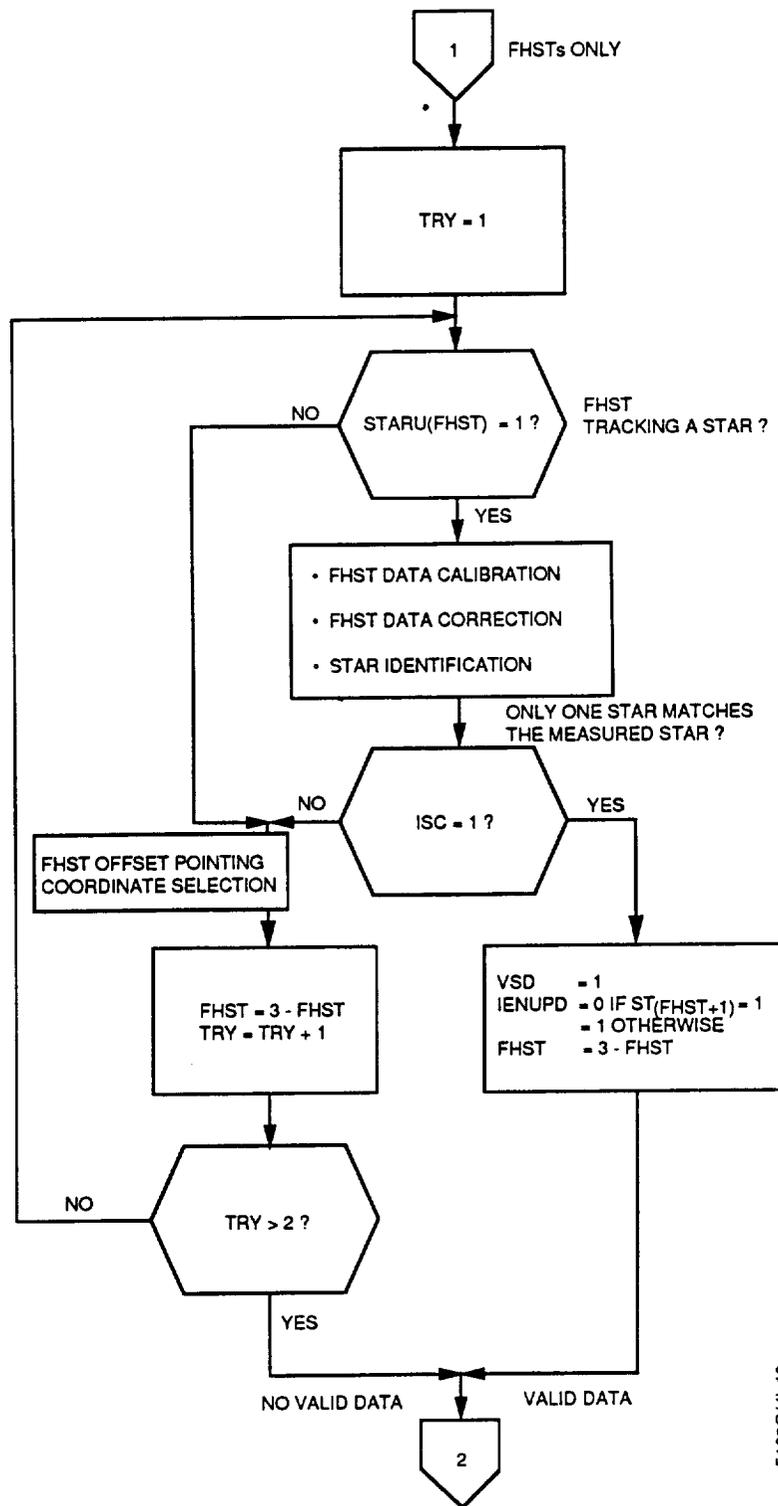
The total aberration velocity is computed:

$$\vec{VA} = \vec{VXS} + \vec{VEXS}$$



5132G(4)-18

Figure 5-9. ATTEST Measurement Processing (1 of 3)



5132G(4)-19

Figure 5-9. ATTEST Measurement Processing (2 of 3)

Table 5-4. Flags and Counters Used by ATTEST (1 of 2)

Flags	Counters
FLTINIT	Update filter initialization flag (external flag from MODECON): = 1, initialize update filter = 0, do not initialize filter
FHST	FHST pointer (internal flag): = 1, use data from FHST 1 = 2, use data from FHST 2
FSS	Flag that indicates if the FSS is the preferred update sensor (internal flag to switch usage between FSS and FHST): = 1, FSS is the preferred sensor = 0, FSS is not the preferred sensor
IENUPD	Update inhibited/not inhibited for sensor being process (internal flag): = 0, update is inhibited according to ST bit corresponding to sensor in question = 1, update is not inhibited according to ST bit corresponding to sensor in question
ISC	Counter of the number of stars that pass the star identification criteria (internal counter)
PASS	Sensor measurement processing counter (internal flag): = 1, first measurement processed = 2, second measurement processed
ST _j	Sensor configuration flag (OBC set by ground): ST ₁ = 0, use FSS and one FHST (as indicated by flag "FHST") = 1, use two FHSTs ST ₂ = 0, FHST 1 not inhibited = 1, FHST 1 inhibited ST ₃ = 0, FHST 2 not inhibited = 1, FHST 2 inhibited ST ₄ = 0, FSS not inhibited = 1, FSS inhibited
STARU _i	Star presence flag for ith FHST (renamed from IVASD _i , which comes from STARAVG): = 1, star data available = 0, no star data available
SUNPS	FSS Sun presence flag (external flag): = 1, Sun in the FSS FOV = 0, Sun not in the FSS FOV

Table 5-4. Flags and Counters Used by ATTEST (2 of 2)

<u>Flags</u>	<u>Counters</u>
AEOFF	Attitude estimation flag (from DBC/GND) = 0, proceed with ATTEST processing = 1, do not execute ATTEST
TRY	Update sensor sequence counter (internal flag): = 1, data from first (i.e., preferred) update sensor has been checked = 2, data from both update sensors has been checked
UPDATE	Update ready flag (external flag set by ATTEST for use by KININT): = 1, updates are ready = 0, updates are not ready
VSD	Valid sensor data flag (internal flag): = 1, valid sensor measurement available = 0, no valid sensor measurement available

where $\overrightarrow{VXS} = (VXS_1, VXS_2, VXS_3)$
 $\overrightarrow{VEXS} = (VEXS_x, VEXS_y, VEXS_z)$
 $VXS_i =$ saved orbit rates (Table 5-3)
 $VEXS_i =$ Earth orbital velocities (input to ATTEST),
 $i = X, Y, Z$

This computation is transformed to ACAD body axis coordinates:

$$\overrightarrow{VAV} = \text{XFORM}(\overrightarrow{VA}, \overrightarrow{SEPA}, -1)$$

where $\overrightarrow{VAV} =$ total star aberration velocity vector in ACAD body axes

$\overrightarrow{SEPA} =$ vector of saved Euler parameters (see Table 5-3)

The projections of the aberration velocity are computed along the FHST X- and Y-axes:

$$XSA = \text{DOT}(\hat{XT}, \overrightarrow{VAV})$$

$$YSA = \text{DOT}(\hat{YT}, \overrightarrow{VAV})$$

where $\hat{XT} = (XT_x, XT_y, XT_z)$
 $\hat{YT} = (YT_x, YT_y, YT_z)$
 $XT_i, YT_i =$ input from STARDAT

The aberration corrections are computed along the FHST X- and Y-axes:

$$XSA = CI * XSA$$

$$YSA = CI * YSA$$

where CI is the inverse of speed of light (DBC).

The FHST measurements are compensated for aberration error:

$$XS = XSC - XSA$$

$$YS = YSC - YSA$$

where XS, YS = components of compensated FHST measurements
(LOS unit vector) corrected for velocity
aberration (in FHST axes)

XSC, YSC = components of compensated FHST calculated
in STARDAT

The Z-component of the compensated/corrected measured star
is computed:

$$ZS = \text{SQRT}(1 - XS * XS - YS * YS)$$

The compensated/aberration corrected star unit vector is
transformed to ACAD body coordinates:

$$LS_1 = XT_X * XS + YT_X * YS + ZT_X * ZS$$

$$LS_2 = XT_Y * XS + YT_Y * YX + ZT_Y * ZS$$

$$LS_3 = XT_Z * XS + YT_Z * YS + ZT_Z * ZS$$

The measurement matrix components are computed for later use:

$$\vec{H1} = \text{CROSS}(\hat{XT}, \vec{LS})$$

$$\vec{H2} = \text{CROSS}(\hat{YT}, \vec{LS})$$

where $\vec{H1}$ = measurement vector (matrix component) for
X-measurement

$\vec{H2}$ = measurement vector (matrix component) for
Y-measurement

Compute the star LOS rate vector in ACAD body coordinates:

$$\vec{LR} = \text{CROSS}(\vec{LS}, \vec{WU})$$

where $\vec{LS} = (LS_1, LS_2, LS_3)$

$\vec{WU} = (WU_X, WU_Y, WU_Z)$

WU_i = saved body rates, $i = X, Y, Z$

Measurement rates are computed along the FHST X- and Y-axes:

$$XSD = \text{DOT}(\hat{X}T, \vec{L}R)$$

$$YSD = \text{DOT}(\hat{Y}T, \vec{L}R)$$

Compute time delay corrections:

$$XSD = XSD * TMD$$

$$YSD = YSD * TMD$$

where TMD is the average measurement time delay (DBC).

Time delay errors are compensated for as follows:

$$XS = XS + XSD$$

$$YS = YS + YSD$$

where XS, YS are components of compensated star position corrected for velocity aberration and measurement time delay (in FHST coordinates).

Star Identification--The star identification module determines if the star currently being tracked is in the active star catalog, and, if so, exactly which star it is. First, the measurement noise variance and an acceptance accuracy for the measurement residuals are computed, followed by a transformation of the FHST axis unit vectors to ECI coordinates. Next, a star catalog is searched for stars satisfying both of the following two criteria:

- The magnitude of the tracked star is within the upper and lower magnitude limit of the star in the catalog

- The measurement residuals are within an acceptable value

For a valid star to have been found, only one star in the catalog must pass the above criteria. More than one star passing the tests results in no valid star and the issuance of the break track command with offset pointing coordinates. First, the measurement noise variances are computed, depending on the measured star location in the FHST FOV:

$$RTP = XS * XS + YS * YS$$

If $RTP > RLM$ set $R = R_U$; otherwise, set $R = R_L$:

$$R_1 = R$$

$$R_2 = R$$

where

- R = FHST measurement variance
- R_1 = measurement variance for X-measurement
- R_2 = measurement variance for Y-measurement
- RTP = radius squared of star position measurement from the tracker null
- R_L, R_U = measurement variances for stars near or far, respectively, from origin (DBC)

Next, the acceptance accuracy is computed as a function of attitude error covariance elements:

$$SCP_{11} = \text{MAXIMUM} \{ \text{EXPONENT}[PM_{11i}] \text{ for } i = 1, 2, \dots, 9 \} + 1$$

$$VS_1 = KS * 2^{[(SCP_{11} - SCP_{11I})/2]}$$

Limit VS_1 as follows:

$$V_{LOW} \leq VS_1 \leq V_{HIGH}$$

$$VS_2 = VS_1$$

where

- SCP_{111} = maximum exponent of attitude covariance elements PM_{11i} , $i = 1, 2, \dots, 9$
- $MAXIMUM \{ \}$ = maximum value of the elements in $\{ \}$
- $EXPONENT[X_i]$ = exponent of X_i when X_i is expressed as a mantissa and exponent with the exponent in powers of 2. The exponent is the same as the NSSC-1 scaling stored in the exponent word for floating-point values in the NSSC-1
- SCP_{111I} = nominal value of SCP_{111} (DBC)
- VS_1 = measurement residual acceptable accuracy criteria for the X-axis FHST measurement
- VS_2 = measurement residual acceptable accuracy criteria for the Y-axis FHST measurement
- V_{LOW}, V_{HIGH} = minimum, maximum acceptable accuracy for measurement residuals (DBCs)

The FHST unit vectors are then transformed to ECI coordinates:

$$\overrightarrow{XTI} = XFORM(\hat{XT}, \overrightarrow{SEPA}, +1)$$

$$\overrightarrow{YTI} = XFORM(\hat{YT}, \overrightarrow{SEPA}, +1)$$

where \overrightarrow{XTI} = FHST X-axis unit vector in terms of ECI components

\overrightarrow{YTI} = FHST Y-axis unit vector in terms of ECI components

The onboard star catalog contains the number of stars in the catalog (NCAMAX) and the following information for each star i in the catalog for $i = 1$ to NCAMAX:

Star magnitude upper limit	ISU _{i}
Star magnitude lower limit	ISL _{i}
Star unit vector in ECI coordinate frame	LSIX _{i} LSIY _{i} LSIZ _{i}

The star catalog search processing is as follows:

Step 1: Initialization:

$$ISC = 0$$

$$n = 1$$

where ISC = counter of number of stars that pass the star identification criteria

n = star catalog search counter

Step 2: Star intensity is checked for:

$$\text{If } (IS \geq ISL_n) \text{ and } (IS \leq ISU_n)$$

Step 3 follows; otherwise, step 5 is next.

Step 3: Measurement residuals are computed:

$$SX = \text{DOT}(\overrightarrow{XTI}, \overrightarrow{LSI}_n)$$

$$SSX = XS - SX$$

$$SX = \text{ABS}(SSX)$$

$$SY = \text{DOT}(\overrightarrow{YTI}, \overrightarrow{LSI}_n)$$

$$SSY = YS - SY$$

$$SY = ABS(SSY)$$

where $\overrightarrow{LSI}_n = (LSIX_n, LSIY_n, LSIZ_n)$

SSX, SSY = components of FHST measurement residuals
(along FHST X- and Y-axes)

SX, SY = components of the nth catalog star initially
and then the absolute values of the measurement residuals

Step 4: Acceptable measurements residuals are tested for:

$$\text{If } (SX < VS_1) \text{ and } (SY < VS_2)$$

$$ISC = ISC + 1$$

$$ISTR = n$$

$$Z_1 = SSX$$

$$Z_2 = SSY$$

where ISTR = star catalog sequence number of last accepted star

Z₁, Z₂ = updated sensor measurement residuals (used in a later subsection on Euler Symmetric Parameter and Gyro Bias Correction)

Step 5:

$$n = n + 1$$

If (n > NCAMAX), step 6 follows; otherwise, step 2 follows.

Step 6: If $ISC \neq 1$ (that is, other than one star passes the test), no valid star has been identified, and the following is performed:

$$NFAIL(FHST) = 1 + NFAIL(FHST)$$

$$LSTID(FHST) = -1$$

where $NFAIL(i)$ = number of star identifications failed and
 $i = 1, 2$ for FHST 1 or 2

$LSTID(i)$ = catalog number of last star identified and
 $i = 1, 2$ for FHST 1 or 2

If $ISC = 1$ (only one star passes the test), a valid star has been found, and the following is performed:

$$NFAIL(FHST) = 0$$

If $ISTR$ is not equal to $LSTID(FHST)$,

$$LSTID(FHST) = ISTR$$

$$ISC = -1$$

FHST Offset Pointing Coordinates Selection--The function of the FHST offset pointing coordinates selection module is to provide updated RFOV offset pointing coordinates to the FHST for the purpose of acquiring and tracking a valid guide star. The offset pointing coordinates are selected from an onboard data table. The pointer that determines the coordinates to be used is incremented each time this module is executed.

The module shall be executed

- When no star is being tracked ($STARU(FHST) = 0$) or
- When the star being tracked is not a valid guide star ($ISC < > 1$ and $NFAIL(FHST) < > 0$)

The module is not executed

- During FHST occultation (OCSTATS(FHST) = 1 or
- If the override flag (OVRFOV) is set by ground command or
- If the FHST is operating in the TFOV scan mode or
- If valid averaged star data exists for the FHST and the number of star identification failures is less than the limit

The FHST offset pointing coordinate selection processing is as follows:

Step 1

If (OCSTATS (FHST) = 1) or
(OVRFOV (FHST) = 1) or
(TFSCAN (FHST) = 1) or
(STARU (FHST) = 1 AND NFAIL (FHST) < = NFALIM)

Then do not execute the module

where OCSTAT_j = FHST occultation status for j=FHST+10
OVRFOV = override flag
TFSCAN = total FOV scan flag
NFALIM = star identification failure threshold

Otherwise, proceed on to Step 2.

Step 2

NFOV(FHST) = NFOV(FHST) + 1

If NFOV(FHST) > MAXFOV(FHST) Then NFOV(FHST) = 1

where NFOV_i = pointer for RFOV offset coordinates table
i = 1 to 2

MAXFOV_i = maximum value of pointer NFOV_i i = 1 to 2

Step 3

Send offset pointing coordinates with the break track commands to the FHST if break track commands are enabled:

If CMDBKT = 1 Then

Send break track command to FHST i with

HRFOV_{ij} = horizontal offset pointing coordinate and

VRFOV_{ij} = vertical offset pointing coordinate

where i = FHST and j = NFOV(FHST)

FSS Data Compensation--The FSS data compensation module compensates FSS data for velocity aberration errors and computes the FSS measurement residuals, the FSS measurement error variances, and the elements of the FSS measurement matrix. The input to the FSS data compensation module consists of the FSS measurements calibrated for the internal FSS error sources (i.e., the FSS data after processing by the FSS data calibration module).

The required processing is as follows:

The total FSS aberration velocity is computed:

$$\vec{VA} = \vec{VE} + \vec{VXS}$$

where $\vec{VXS} = (VXS_1, VXS_2, VXS_3)$

$\vec{VE} = (VE_X, VE_Y, VE_Z)$

The computation is transformed to vehicle coordinations:

$$\vec{VAV} = \text{XFORM}(\vec{VA}, \vec{SEPA}, -1)$$

The sensor axes are selected:

If HEAD = 0,

$$XF_i = XF_{1i} \quad i = 1 \text{ to } 3$$

$$YF_i = YF_{1i} \quad i = 1 \text{ to } 3$$

$$ZF_i = ZF_{1i} \quad i = 1 \text{ to } 3$$

Otherwise,

$$XF_i = XF_{2i} \quad i = 1 \text{ to } 3$$

$$YF_i = YF_{2i} \quad i = 1 \text{ to } 3$$

$$ZF_i = ZF_{2i} \quad i = 1 \text{ to } 3$$

The aberration corrections are computed:

$$EF_X = \text{DOT}(\vec{XF}, \vec{VAV}) * CI$$

$$EF_Y = \text{DOT}(\vec{YF}, \vec{VAV}) * CI$$

$$EF_Z = \text{DOT}(\vec{ZF}, \vec{VAV}) * CI$$

where $\vec{XF} = (XF_1, XF_2, XF_3)$, etc.

FSS measurements are compensated for aberration:

$$XP = XPC - SZN * (EF_X - XPC * EF_Z)$$

$$YP = YPC - SZN * (EF_Y - YPC * EF_Z)$$

where $SZN = (1 + XPC * XPC + YPC * YPC)^{1/2}$

XPC, YPC = saved FSS data (see Table 5-3)

The sunline unit vector is computed at the vehicle in ECI coordinates:

$$\hat{S}_A = \hat{S}_{IXS} - \overrightarrow{RXS} * RSN$$

where $\hat{S}_{IXS} = (SIXS_X, SIXS_Y, SIXS_Z)$
 $SIXS_i$ = components of Sun-pointing unit vector and
 $i = X, Y, Z$ (from EPHEM)
 $\overrightarrow{RXS} = (RXS_1, RXS_2, RXS_3)$
 RXS_i = saved orbit position data (see Table 5-3)
 RSN = inverse of mean distance from Earth to Sun
 (data base input)

This computation is transformed to ACAD coordinates:

$$\overrightarrow{S}_V = XFORM(\hat{S}_A, \overrightarrow{SEPA}, -1)$$

where \overrightarrow{S}_V is the sunline vector at the vehicle in ACAD body axes.

The expected FSS measurements are computed:

$$SP_X = DOT(\hat{XF}, \hat{S}_V)$$

$$SP_Y = DOT(\hat{YF}, \hat{S}_V)$$

$$SP_Z = DOT(\hat{ZF}, \hat{S}_V)$$

$$XPE = SP_X / SP_Z$$

$$YPE = SP_Y / SP_Z$$

where SP_i = expected sunline components along FSS axes
 and $i = X, Y, Z$

XPE, YPE = expected FSS measurements

The measurement residuals are computed:

$$Z_1 = (XP - XPE)/2$$

$$Z_2 = (YP - YPE)/2$$

where Z_1, Z_2 represent updated sensor measurement residuals.

The measurement matrix components are computed:

$$FM_{11} = (XF_1 - ZF_1 * XPE)/(SP_Z * 2)$$

$$FM_{12} = (XF_2 - ZF_2 * XPE)/(SP_Z * 2)$$

$$FM_{13} = (XF_3 - ZF_3 * XPE)/(SP_Z * 2)$$

$$FM_{21} = (YF_1 - ZF_1 * YPE)/(SP_Z * 2)$$

$$FM_{22} = (YF_2 - ZF_2 * YPE)/(SP_Z * 2)$$

$$FM_{23} = (YF_3 - ZF_3 * YPE)/(SP_Z * 2)$$

$$\vec{H}_1 = \text{CROSS}(\vec{FM}_1, \hat{SV})$$

$$\vec{H}_2 = \text{CROSS}(\vec{FM}_2, \hat{SV})$$

where $\vec{FM}_i = (FM_{i1}, FM_{i2}, FM_{i3})$ = intermediate vectors used in computing $\vec{H}_1, \vec{H}_2, i = 1, 2$

\vec{H}_i = measurement vectors (matrix components) for X- and Y-measurements (corresponding to $i = 1, 2$)

The measurement error variances are computed:

$$R1 = \left[\left(1 + XPE^2 \right)^2 \right] * RF/4$$

$$R2 = \left[\left(1 + YPE^2 \right)^2 \right] * RF/4$$

where R_1 = measurement variance for X-measurement

R_2 = measurement variance for Y-measurement

RF = FSS measurement error variance (DBC)

Gain Matrix Computation--The gain matrix computation module computes the elements of the attitude estimation gain matrix. Since the sensor measurements are processed sequentially, the gain matrix computation module will be executed twice during an attitude estimation processing cycle. The path through the gain matrix computation module depends on which measurement is being processed (i.e., the first or the second). If the first measurement is being processed (PASS = 1 for X-measurement), set the measurement vector, \vec{H} , equal to \vec{H}_1 (the measurement vector for the first measurement) and R equal to R_1 . If the second measurement is being processed (PASS = 2 for Y-measurement), set \vec{H} equal to \vec{H}_2 and R equal to R_2 .

After the proper value of \vec{H} has been set, the following processing is performed:

$[PM_{11}] \vec{H}$ is computed:

$$X_{11} = \text{DOT}(\vec{H}, \vec{VP}_1)$$

$$X_{12} = \text{DOT}(\vec{H}, \vec{VP}_2)$$

$$X_{13} = \text{DOT}(\vec{H}, \vec{VP}_3)$$

where $\vec{VP}_1 = (PM_{111}, PM_{112}, PM_{113})$

$\vec{VP}_2 = (PM_{114}, PM_{115}, PM_{116})$

$\vec{VP}_3 = (PM_{117}, PM_{118}, PM_{119})$

$U = \vec{H}[PM_{11}] \vec{H} + R$ is computed:

$$U = |\text{DOT}(\vec{H}, \vec{X}_1)| + R$$

where $\vec{X}_1 = (X_{11}, X_{12}, X_{13})$

The first three elements of the gain matrix are computed:

$$K_i = X_{1i}/U \quad i = 1 \text{ to } 3$$

where K_i represents Kalman filter gain matrix elements and $i = 1$ to 6 .

$[PM_{12}] \vec{H}$ is computed:

$$X_{21} = \text{DOT}(\vec{H}, \vec{VP}_4)$$

$$X_{22} = \text{DOT}(\vec{H}, \vec{VP}_5)$$

$$X_{23} = \text{DOT}(\vec{H}, \vec{VP}_6)$$

where $\vec{VP}_4 = (PM_{121}, PM_{122}, PM_{123})$

$\vec{VP}_5 = (PM_{124}, PM_{125}, PM_{126})$

$\vec{VP}_6 = (PM_{127}, PM_{128}, PM_{129})$

The last three elements of the gain matrix are computed:

$$K(i + 3) = X_{2i}/U \quad i = 1 \text{ to } 3$$

Euler Symmetric Parameter and Gyro Bias Correction Computation--The Euler symmetric parameter and gyro bias correction computation combines the ATTEST estimates of the Euler symmetric parameter and gyros bias compensation errors into a

vector that can be used to update these quantities. The two sensor measurements must be processed before the total correction vector is computed. A flag, UPDATE, is set to indicate that update data are available.

If the first measurement (PASS = 1 for X-measurement) is being processed, the error state vector is computed as follows:

$$SE_i = K_i * Z_1 \quad i = 1 \text{ to } 6$$

If the second measurement (PASS = 2 for Y-measurement) is being processed, the error state vector is computed as follows:

$$X = \text{DOT}(\vec{H}_2, \vec{S}_1)$$

$$Z_2 = Z_2 - X$$

$$SE_i = SE_i + K_i * Z_2 \quad i = 1 \text{ to } 6$$

where S_1 is the vector formed from the first three components of SE_i .

The update ready flag is set after second measurement processing is complete:

$$\text{UPDATE} = \text{IENUPD}$$

Covariance Matrix Update--The covariance matrix update module adjusts the elements of the attitude estimation error covariance matrix to account for the error reduction achieved by processing the update sensor data. The following processing is performed if IENUPD = 1. Otherwise, no further processing in this subsection is performed.

The following processing is performed (using data from the gain matrix computation module):

$$PM_{111} = PM_{111} - X_{11} * X_{11}/U$$

$$PM_{112} = PM_{112} - X_{11} * X_{12}/U$$

$$PM_{113} = PM_{113} - X_{11} * X_{13}/U$$

$$PM_{114} = PM_{112}$$

$$PM_{115} = PM_{115} - X_{12} * X_{12}/U$$

$$PM_{116} = PM_{116} - X_{12} * X_{13}/U$$

$$PM_{117} = PM_{113}$$

$$PM_{118} = PM_{116}$$

$$PM_{119} = PM_{119} - X_{13} * X_{13}/U$$

The elements of $[PM_{22}]$ are updated as follows:

$$PM_{221} = PM_{221} - X_{21} * X_{21}/U$$

$$PM_{222} = PM_{222} - X_{21} * X_{22}/U$$

$$PM_{223} = PM_{223} - X_{21} * X_{23}/U$$

$$PM_{224} = PM_{222}$$

$$PM_{225} = PM_{225} - X_{22} * X_{22}/U$$

$$PM_{226} = PM_{226} - X_{22} * X_{23}/U$$

$$PM_{227} = PM_{223}$$

$$PM_{228} = PM_{226}$$

$$PM_{229} = PM_{229} - X_{23} * X_{23}/U$$

The elements of $[PM_{12}]$ are updated as follows:

$$PM_{121} = PM_{121} - X_{11} * X_{21}/U$$

$$PM_{122} = PM_{122} - X_{12} * X_{21}/U$$

$$PM_{123} = PM_{123} - X_{13} * X_{21}/U$$

$$PM_{124} = PM_{124} - X_{11} * X_{22}/U$$

$$PM_{125} = PM_{125} - X_{12} * X_{22}/U$$

$$PM_{126} = PM_{126} - X_{13} * X_{22}/U$$

$$PM_{127} = PM_{127} - X_{11} * X_{23}/U$$

$$PM_{128} = PM_{128} - X_{12} * X_{23}/U$$

$$PM_{129} = PM_{129} - X_{13} * X_{23}/U$$

If the first measurement is being processed, the gain matrix computation module is executed again to initiate processing of the second measurement. If the second measurement was just processed, ATTEST processing is exited.

- Output--ATTEST produces the following output:

<u>Name</u>	<u>Description</u>
BES _i	Save components of Earth's magnetic field vector along spacecraft axes where i = X, Y, Z (to STARDAT)
CBT _i	Break track commands to FHST i where i = 1, 2 (to FHST)
FHHAS _i	Save raw FHST horizontal measurements and FHST status where i = 1, 2 for FHST 1 or 2 (to STARDAT)
FHMS _i	Save FHST intensities from FHST i where i = 1, 2 (to STARDAT)
FHST	Set FHST selection flag set during attitude estimation process to select FHST 1 (FHST = 1) or 2 (FHST = 2) (to STARDAT, ATTEST)
FHTS _i	Save FHST temperature data from FHST i where i = 1, 2 (to STARDAT)
FHVAS _i	Save raw FHST vertical measurements and FHST status where i = 1, 2 for FHST 1 or 2 (to STARDAT)
FLTINIT	Filter initialization flag. This is reset by ATTEST (to ATTEST)
MDS _i	Save components of torquer bar generated dipole moment in antenna boom coordinates where i = X, Y, Z (to STARDAT)
SE _i	Estimate roll, pitch, yaw attitude determination errors where i = 1 to 3, and estimated roll, pitch, and yaw gyro drift errors where i = 4 to 6; these errors are in ACAD body axes (to KININT)
UPDATE	Update filter flag (0 = no update, 1 = update) (to KININT)

5.2.4 ATTITUDE CONTROL

5.2.4.1 High-Level Thrust Control Law Function

The high-level thrust control law function (HLTCON) uses commanded and actual angular rates and attitude errors to control the roll and pitch attitude/rate errors, as shown below.

- Input--HLTCON uses the following input:

<u>Name</u>	<u>Description</u>
ACSINT	ACAD sample period (from MODECON)
B1ON	Thruster bank 1 (pitch) on flag (0 = off, 1 = on) (DBC, set by ground)
B2ON	Thruster bank 2 (roll) on flag (0 = off, 1 = on) (DBC, set by ground)
DELVK ₁	Time bias for OBC thrusting time counter (DBC)
DELVK ₂	Time bias for hardware backup timer setting (DBC)
DELVT	Velocity control time counter (from HLTCON)
DELVTG	Commanded thrusting time (DBC set by ground)
DW _i	Estimate of rate due to past thruster firing where i = 1 to 2 (DBC)
D1 _x	Roll dead-zone limit for switching on thrusters (DBC)
D1 _y	Pitch dead-zone limit for switching on thrusters (DBC)
D3 _x	Roll dead-zone limit for switching off thrusters (DBC)
D3 _y	Pitch dead-zone limit for switching off thrusters (DBC)
E _i	Attitude errors in roll, pitch, i.e., commanded-minus-actual angles about the ACAD body axes for i = X, Y (variables from ATTERR)
HLCNT _i	High-level thruster cumulative counters (from HLTCON)
HKP _x	Roll position gain (DBC)
HKP _y	Pitch position gain (DBC)

<u>Name</u>	<u>Description</u>
HKR _X	Roll rate gain (DBC)
HKR _Y	Pitch rate gain (DBC)
HLTINIT	Flag indicating velocity control parameters are to be initialized (= 0, do not initialize, = 1, initialize) (variable flag from MODECON, HLTCON)
IH _i	Thruster control flag where i = 1 to 2 (from HLTCON)
KIRE _i	Integral estimated rate error scale factor where i = 1 to 2 (DBC)
KRE _i	Estimated rate error scale factor where i = 1 to 2 (DBC)
RL1 _i	Estimated rate error limit where i = 1 to 2 (DBC)
RL2 _i	Integral estimated rate error limit where i = 1 to 2 (DBC)
RS2 _i	Integral rate error estimate where i = 1 to 2 (from HLTCON)
RS _i	Rate estimate where i = 1 to 2 (from HLTCON)
W _i	Compensated spacecraft body rates in roll and pitch (i = X, Y ACAD body axes) (variables from GYRCOMP)
WC _X	Commanded spacecraft roll rate about ACAD body X-axis (variable from THRUSTR)
WC _Y	Commanded spacecraft pitch rate about ACAD body Y-axis (variable from THRUSTR)

● Processing--HLTCON is executed every 256 msec during the velocity control mode only. It is performed after ATTERR, GYRCOMP, and THRUSTR.

Initialization--Initialization of this module occurs when the velocity control mode is initialized as follows:

If HLTINIT = 1

For i = 1 to 2 (roll and pitch axes)

RS_i = W_i (rate estimate)

IH_i = 0 (thruster control flags)

RS2_i = 0 (integral rate error estimate)

HLTINIT = 0

DELVT = DELVTG + DELVK1

DELVTB = DELVTG + DELVK2

where IH_x, IH_y = internal control flags for the roll and pitch axes, respectively

Thruster Control Flags--Each time HLTCON is executed, all thruster control flags are set to enable both roll and pitch control:

ICTL_i = 1 for i = 1 to 4

If B1ON = 0 (no pitch control)

ICTL₁ = ICTL₂ = 0 (pitch control orbit adjust thrusters (OATs) disabled)

If B2ON = 0 (no roll control)

ICTL₃ = ICTL₄ = 0 (roll control OATs disabled)

where ICTL_i = internal flags for indicating high-level thruster use for thruster i, where i = 1 to 4;
= 0, not used, = 1, used

Control Law Computation--The high-level thruster control law computations are performed for roll and pitch axes (i = 1, 2) if the associated thruster bank is on (B2ON = 1 for roll, B1ON = 1 for pitch). If B2ON and/or B1ON = 0, no control is performed about the respective axis and the program skips to the Velocity Control Countdown Section.

For i = 1 to 2

If (i = 1 and B2ON = 1) or (i = 2 and B1ON = 1)

A rate estimate (RS_i) is computed based on the past firing history of the high-level thrusters (IH_i):

$$RS_i = RS_i + DW_i * IH_i$$

where DW_i = estimate of the rate the spacecraft will undergo about axis i due to an OAT being off-pulsed (DBC)

IH_i = past off-pulsing flag (= 1, positive torque applied about axis i ; = 0, no torque about axis i ; = -1, negative torque applied about axis i)

This estimate is used to compute a (limited) estimated rate error ($RS1_i$):

$$RS1_i = KRE_i * (W_i - RS_i)$$

$$\text{If } ABS(RS1_i) > RL1_i \text{ Then } RS1_i = SIGN(RS1_i) * RL1_i$$

where KRE_i = estimated rate error scale factor about axis i

$RL1_i$ = estimated rate error limit about axis i

This estimate is then used to compute a (limited) integral rate error ($RS2_i$):

$$RS2_i = RS2_i + KIRE_i * RS1_i$$

$$\text{If } ABS(RS2_i) > RL2_i \text{ Then } RS2_i = SIGN(RS2_i) * RL2_i$$

where $KIRE_i$ = integral estimated rate error scale factor

$RL2_i$ = integral estimated error limit

The rate estimate (RS_i), estimated rate error ($RS1_i$), and integral rate error ($RS2_i$) are then combined to produce an updated rate estimate:

$$RS_i = RS_i + RS1_i + RS2_i$$

The rate error (commanded-estimated) is then combined with the position error (E_i) to give the control law error (ERR_i):

$$ERR_i = HKP_i * E_i - HKR_i * (WC_i - RS_i)$$

where HKP_i = position gain

HKR_i = rate gain

WC_i = commanded rate

Control torques are produced by off-modulating the appropriate high-level thruster. The control torques are achieved by sending the appropriate enable/disable command to the thrusters.

Figure 4-13 and Table 4-1 give the OAT locations relative to the spacecraft body axes.

The control law error, ERR_i , is processed through a dead-band with hysteresis computation as follows for both the roll axis ($i = 1$) and the pitch axis ($i = 2$). The control flags ($ICTL_i$) are set as follows.

Case 1: Previous Positive Torque ($IH_i = 1$)

If the spacecraft was previously performing a positive torque ($IH_i = 1$), the following is performed: If the control law error is within the OAT on limit $D3_i$, the control torque is discontinued and the spacecraft's 0 torque flag is set:

$$\text{If } ERR_i > -D3_i \quad IH_i = 0$$

If the control law error is outside the OAT on limit $D3_i$, the appropriate OAT for a positive torque about axis i is

off-pulsed. In this case, ERR_i is negative and a positive torque was applied to counter a negative pointing error:

```
Else ( $ERR_i = < -D3_i$ )
```

```
If  $i = 1$  (roll)
```

```
    ICTL4 = 0
```

```
Else (pitch)
```

```
    ICTL1 = 0
```

Case 2: Previous Negative Torque ($IH_i = -1$)

If the spacecraft was previously performing a negative torque ($IH_i = -1$), the following is performed: If the control law error is within the on-band $D3_i$ limit, IH_i is set to a 0 torque. In this case, ERR_i is positive and a negative torque is needed to counter a positive pointing error:

```
If  $ERR_i < D3_i$  Then  $IH_i = 0$ 
```

If the control error is outside the on-band limit ($D3_i$), the appropriate OAT is offpulsed to perform a torque about axis i :

```
Else ( $ERR_i > = D3_i$ )
```

```
If  $i = 1$  (roll)
```

```
    ICTL3 = 0
```

```
Else (pitch)
```

```
    ICTL2 = 0
```

```
Endif
```

Case 3: No Previous Torque ($IH_i = 0$)

If the spacecraft was performing no previous torque ($IH_i = 0$), the following is performed: If the control law error

(ERR_i) is outside the off-to-control limit (Dl_i), the following is performed:

```
If ABS(ERRi) > = Dli Then
  If ERRi < 0 Then           ! (a positive torque is needed)
    IHi = 1                 ! (flag for positive torque)
    If i = 1 THEN           ! (roll)
      ICTL4 = 0
    Else                     ! (pitch)
      ICTL1 = 0
    Endif
  Else                       ! (a negative torque is needed)
    IHi = -1               ! (flag for negative torque)
    If i = 1 Then           ! (roll)
      ICTL3 = 0
    Else                     ! (pitch)
      ICTL2 = 0
    Endif
  Endif
Endif
```

Velocity Control Countdown--The thrusting time counter, DELVT, is counted down to zero, at which time shutdown is accomplished by disabling all high-level thrusters and requesting a mode transition to the thruster maneuver mode by MODECON:

```
DELVT = DELVT - ACSINT
If DELVT ≤ 0
  ICTLi = 0   i = 1 to 4   (disable all OATs)
  ACADMDT = 6 (MODECON request flag to transition to the
               thruster maneuver mode)
```

- Output--HLTCON produces the following output:

<u>Name</u>	<u>Description</u>
ACADMDT	ACAD mode request flag to transition to thruster maneuver mode (to MODECON)
DELVT	OBC thrusting time counter (to HLTCON)
HLCNT _i	High-level thruster firing cumulative counters where i = 1 to 4 (to HLTCON)
HLTINIT	HLTCON initialization flag (to HLTCON)
ICNTL _i	Commands to PE for enabling/disabling high-level thrusters where i = 1 to 4 for thrusters 1 to 4 (to truth model)
IH _i	Thruster control flags for axis i where i = 1 (roll) to 2 (pitch) (to HLTCON)
RS2 _i	Integral rate error estimate where i = 1 (roll) to 2 (pitch) (to HLTCON)
RS _i	Rate estimate where i = (roll) to 2 (pitch) (to HLTCON)

5.2.4.2 Low-Level Thrust Control Law Function

The low-level thrust control law function (LLTCON) uses the attitude errors and rates to perform a thruster maneuver or to maintain attitude control during a velocity correction, as shown below.

- Input--LLTCON uses the following input:

<u>Name</u>	<u>Description</u>
CSL _i	Count of number of cycles to control (fire) low-level thrusters for each axis where i = X, Y, Z ACAD body axes (DBC's)
E _i	Attitude errors in roll, pitch, and yaw, i.e., commanded-minus-actual angles about ACAD body axes where i = X, Y, Z (variables from ATTERR)
KPV _i	Position gains, velocity control mode for roll, pitch, and yaw where i = X, Y, Z ACAD body axes (DBC's)
KRV _i	Rate gains, velocity control mode for roll, pitch, and yaw where i = X, Y, Z ACAD body axes (DBC's)

<u>Name</u>	<u>Description</u>
LCA _i	Rate filter coefficient A where i = 1 to 3 ACAD axes (from MODECON)
LCB _i	Rate filter coefficient B where i = 1 to 3 ACAD axes (from MODECON)
LCC _i	Rate filter coefficient C where i = 1 to 3 ACAD axes (from MODECON)
LH _i	Past value of hysteresis state flag where i = 1 to 3 ACAD axes (from LLTCON)
LHY _i	Hysteresis constants for velocity control mode in roll, pitch, and yaw where i = X, Y, Z ACAD body axes (DBC's)
LKP _i	Position gains in roll, pitch, and yaw where i = X, Y, Z ACAD body axes, thruster maneuver mode (DBC's)
LKR _i	Rate gains in roll, pitch, and yaw where i = X, Y, Z ACAD body axes, thruster maneuver mode (DBC's)
LLCNT _i	Low-level thruster controlled direction and axis counter (from LLTCON)
LLF _i	Axis-controlled flags where i = 1 to 3 (from MODECON)
LLTINIT	Low-level thruster initialization flag (from MODECON)
LPL _i	Position limits in roll, pitch, and yaw where i = X, Y, Z ACAD body axes for thruster maneuver mode (DBC's)
PRE1 _i	Past value of rate error (commanded-actual) where i = 1 to 3 ACAD axes (from LLTCON)
PRE2 _i	Previous past value of rate error (PRE1 _i) where i = 1 to 3 ACAD axes (from LLTCON)
PWF1 _i	Past value of filtered rate error where i = 1 to 3 (from LLTCON)
PWF2 _i	Previous past value of filtered rate error where i = 1 to 3 (from LLTCON)
SS _i	Control axis selection parameters where i = 1 to 3 for axis, i = 4 for transition to new axis (from LLTCON)
U, V	Sequential firing logic state parameters (from LLTCON)
W _i	Compensated spacecraft body rates in roll, pitch, and yaw where i = X, Y, Z ACAD body axes (variables from GYRCOMP)

Name	Description
WC _i	Commanded spacecraft rates in roll, pitch, and yaw where i = X, Y, Z ACAD body axes (variables from THRUSTER)

● Processing--LLTCON is executed every 256 msec during the velocity control and thruster maneuver modes.

Initialization--If the low-level thruster initialization flag is set (LLTINIT = 1), the following initialization is performed:

```

LHi = 0    for i = X, Y, Z
PRE1i = PRE2i = PWF1i = PWF2i = WCi - Wi
           for i = X, Y, Z
SS4 = 1
U = 0
V = 0
LLCNTi = 1    for i = 1 to 6
LLTINIT = 0

```

where

LH_i = intermediate value of control flags CN_i, CP_i (see below) for roll, pitch, and yaw where i = X, Y, Z ACAD body axes

PRE1_i, PRE2_i, PWF1_i, and PWF2_i = rate filter state variables

SS₄ = control axis selection parameter

U, V = sequential logic parameters

At each pass through LLTCON, the control signals for each axis are reset as follows:

$$CP_i = CN_i = 0 \quad \text{for } i = X, Y, Z$$

where CP_i = control (firing) flags for positive control about roll, pitch, and yaw axes where i = X, Y, Z;
(= 0, no control, = 1, control)

CN_i = control (firing) flags for negative control about roll, pitch, and yaw axes where $i = X, Y, Z$;
(= 0, no control, = 1, control)

$i = X, Y, Z$ ACAD body axes

Control Law Computation--The axes controlled by the low-level thrusters depend on the ACAD mode and the high-level thruster bank activated. Control flags LLF_i are set in MODECON to indicate which axes LLTCON controls: If $LLF_i = 1$, LLTCON controls axis i and the control law is computed only for these axes. The attitude error (E_i) is first multiplied by a gain (LKP_i) and limited to $\pm LPL_i$:

For $i = 1$ to 3

If $LLF_i = 1$ (if control is requested about axis i)

$SP_i = LKP_i * E_i$

If $ABS(SP_i) > LPL_i$ Then $SP_i = sign(SP_i) * LPL_i$

The rate error (commanded - actual) is filtered, and the past filter state values are updated. Filter coefficients are mode-dependent and set in MODECON:

$$WF_i = LCC_i * (WC_i - W_i + 2 * PRE1_i + PRE2_i) - LCB_i * PWF2_i - LCA_i * PWF1_i$$

$$PRE2_i = PRE1_i$$

$$PWF2_i = PWF1_i$$

$$PRE1_i = WC_i - W_i$$

$$PWF1_i = WF_i$$

The filtered rate is then added to the position limiter output to obtain the control law output (S_i):

$$S_i = SP_i - LKR_i * WF_i$$

The control law output (S_i) is fed to a deadband plus hysteresis computation that resets the firing flags for positive (CP_i) or negative (CN_i) control. LH_i is the past value of the hysteresis state flag and is reset if S_i is outside the hysteresis band:

```
If ABS( $S_i$ ) > = 1 Then  $LH_i$  = 1
If ABS( $S_i$ ) < (1.0 -  $LHY_i$ ) Then  $LH_i$  = 0
If [( $LH_i$  = 1) and ( $S_i$  > 0)] Then
     $CN_i$  = 1 (command negative control about axis i)
Else
     $CN_i$  = 0 (command no negative control about axis i)
Endif
If [( $LH_i$  = 1) and ( $S_i$  < 0)]
     $CP_i$  = 1 (command positive control about axis i)
Else
     $CP_i$  = 0 (command no positive control about axis i)
Endif
```

CP_i and CN_i are the control (firing) flags for positive and negative control about the roll, pitch, and yaw axes ($i = 1$ to 3).

Sequential Logic/Firing Computations--The multiple firing algorithm is shown in Figure 5-10. It allows multiple firing about a single axis or a combination of axes. The same axis is controlled until variable C is 1 or control is no longer desired about that axis (i.e., CP_i or CN_i are 0). C is the current firing counter and allows for more than one successive firing on each axis.

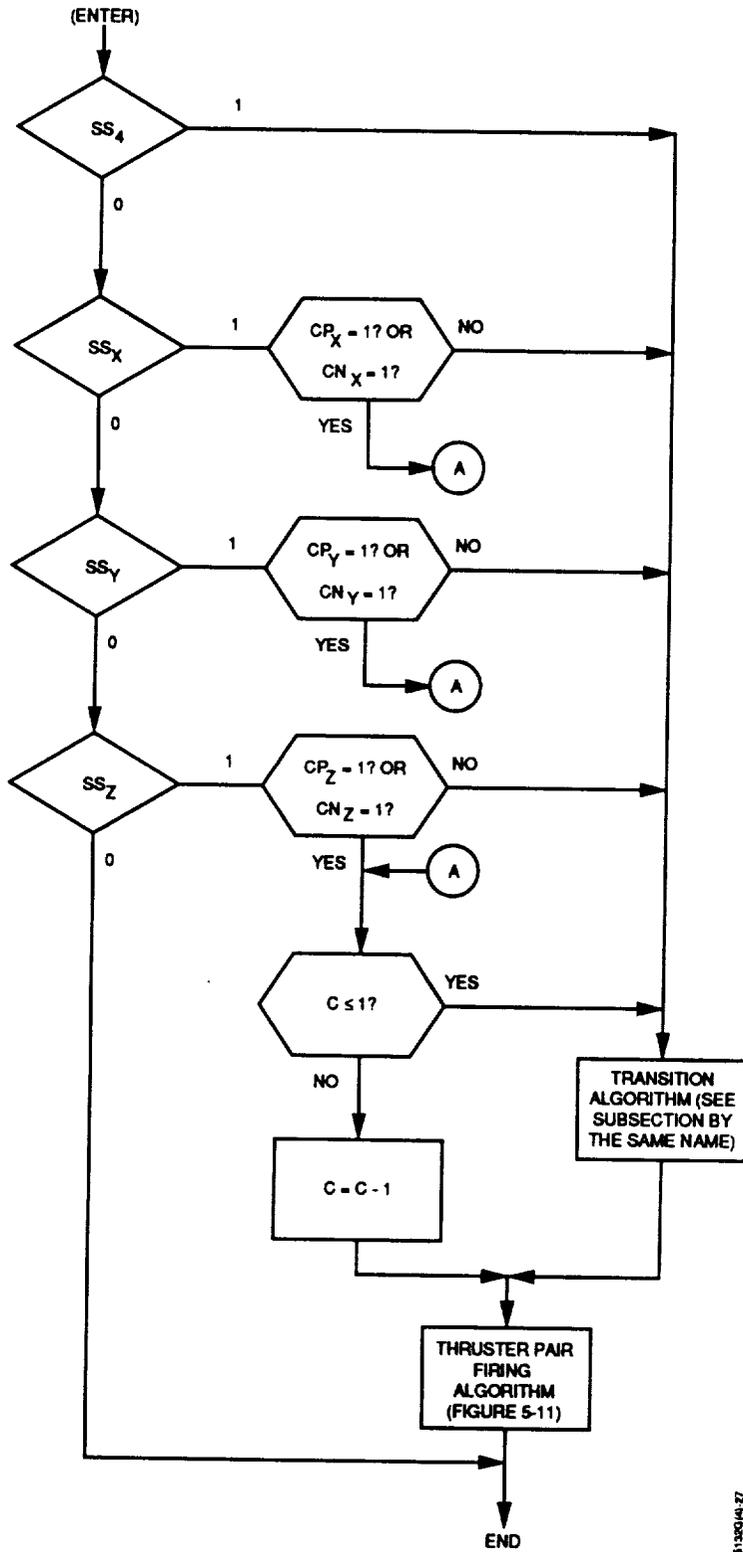


Figure 5-10. Multiple Firing Algorithm

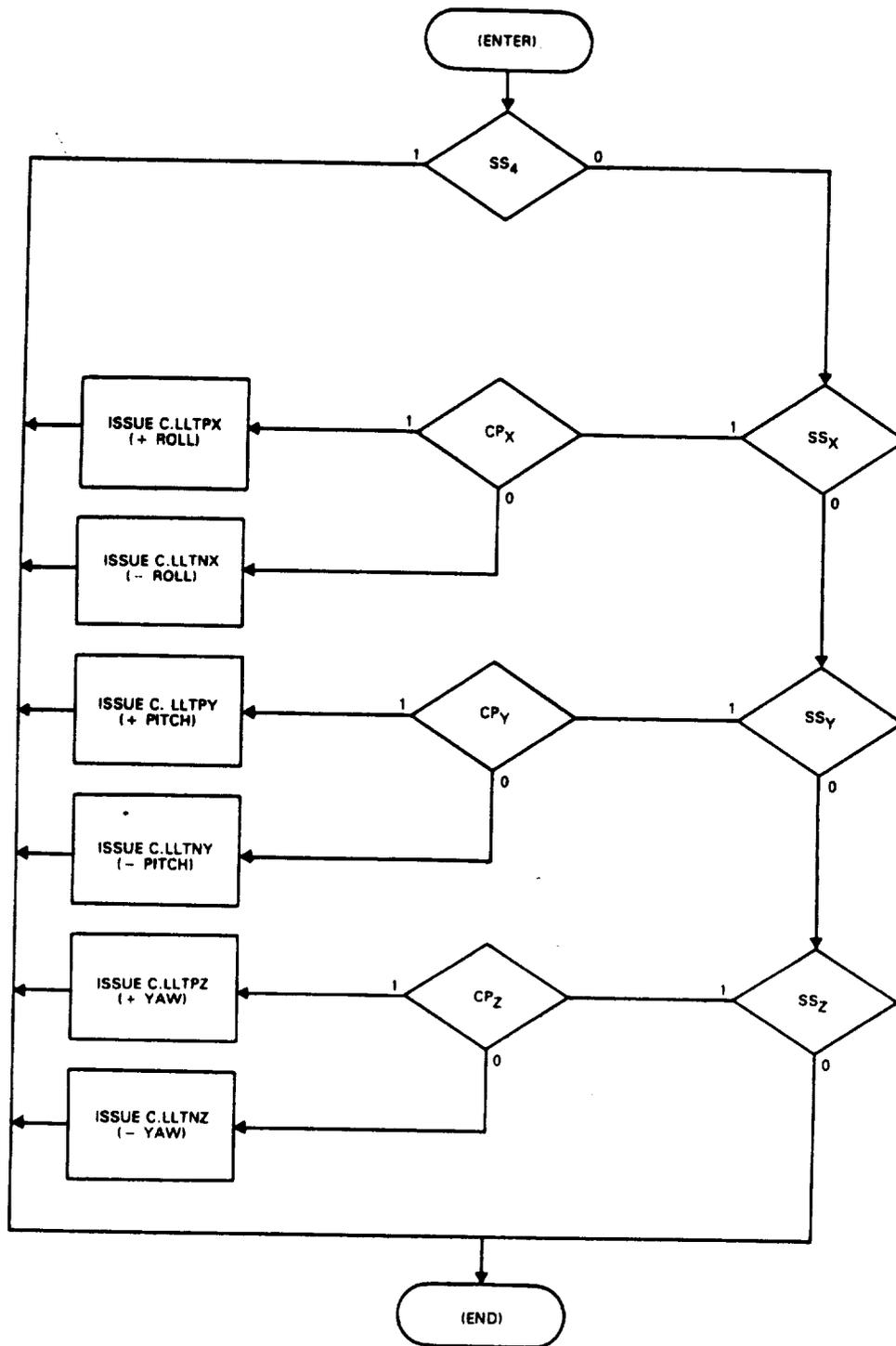
Figure 5-10 provides for all the necessary logic for transitioning between control axes and firing the thrusters by reference to the transition algorithm of the following subsection and to the thruster firing algorithm of Figure 5-11.

On the first pass through Figure 5-10 after the velocity control or thruster maneuver modes are initialized, SS_4 is 1, so that the transition algorithm will be used to set appropriate values of the SS_i control flags ($i = X, Y, Z, 4$).

Transition Algorithm--The current thruster firing scheme allows control along only one axis at a time. If control is required about one axis, it is performed about that axis. When control is required about two axes, it is alternated between the two. If control is required about three axes, it is alternated in the order of roll, pitch, and then yaw. The state transition algorithm uses information about the current controlled axis and the requested axis or axes. The algorithm selects only one axis, even if three-axis control is desired.

The state transition algorithm is as follows: (The symbol-ology and explanation of logical operators is shown after item 13.)

1. $UP = U$
2. $VP = V$
3. $R = CP_X + CN_X$
4. $P = CP_Y + CN_Y$
5. $Y = CP_Z + CN_Z$
6. $U = \text{NOT} (\bar{P} * \bar{Y} + \overline{UP * VP * R} + UP * VP * R + UP * R * \bar{Y})$
7. $V = \bar{P} * Y + R * \bar{P} + \overline{UP * VP * R} + UP * R + UP * \overline{VP * Y}$
8. $SS_X = \bar{U} * V$
9. $SS_Y = U * \bar{V}$
10. $SS_Z = U * V$
11. $SS_4 = \bar{U} * \bar{V}$



9780 05

Figure 5-11. Thruster Firing Logic

12. If $(SS_i = 0)$, $CP_i = CN_i = 0$ for $i = X, Y, Z$

13. If $(SS_i = 1)$, $C = CSL_i$ for $i = X, Y, Z$

where C = current multiple firing counter

CN_i = negative control about roll, pitch, and yaw axes
where $i = X, Y, Z$

CP_i = positive control about roll, pitch, and yaw axes
where $i = X, Y, Z$

P = pitch control desired command

R = roll control desired command

Y = yaw control desired command

SS_i = control axis selection parameters where $i = X, Y, Z$ (roll, pitch, and yaw axes) ($= 0$, no control; $= 1$, control) and $i = 4$ for transition ($= 0$, controlling about any one of the axes; $= 1$, no control on any of the three axes)

U = sequential logic state parameter

UP = previous U parameter state

V = sequential logic state parameter

VP = previous V parameter state

$+$ = logical operator denoting or. Therefore, if either CP_x or $CN_x = 1$, $R = 1$; otherwise, $R = 0$

$*$ = logical operator denoting and; e.g., $UP*VP*R$ is 1 if each of UP , VP , and R are 1; otherwise, it is 0

\bar{X} = denotes complement of X or same as NOT (X), as defined earlier

The thruster pair firing logic is specified in Figure 5-11 and is called according to the logic of Figure 5-10. During each computation cycle, firing commands will be issued unless the transition parameter SS_4 is 1, indicating no control is required about any axis. If $SS_4 = 0$, one of six firing commands will be issued to the propulsion electronics (PE), depending on the control flags SS_i and CP_i for $i = X, Y, Z$. These firing commands ($C.LLTP_i$ and $C.LLTN_i$ for $i = X, Y, Z$) result in low-level thruster firings for positive or negative rotations about the X, Y, Z axes. These correspond to \pm roll, \pm pitch, or \pm yaw commands. The following subsection, Output, defines these commands.

For the current computation cycle time of 256 msec, the above commands are issued within 128 msec after the input of the last 64-msec gyro data sample used for the current computation cycle.

- Output--LLTCON produces the following output:

<u>Name</u>	<u>Description</u>
C.LLTP _i	Commands to PE to fire low-level thrusters for a plus rotation about i-axis where i = X, Y, Z for roll, pitch, and yaw (to Truth Model)
C.LLTN _i	Commands to PE to fire low-level thrusters for a negative rotation about i-axis where i = X, Y, Z for roll, pitch, and yaw (to Truth Model)
LH _i	Past value of hysteresis state flag where i = 1 to 3 (to LLTCON)
LLTINIT	Low-level thruster control initialization flag reset to zero by LLTCON (to LLTCON)
PRE _{1i}	Past value of rate error where i = 1 to 3 (to LLTCON)
PRE _{2i}	Previous past value of rate error where i = 1 to 3 (to LLTCON)
PWF _{1i}	Past value of filtered rate error where i = 1 to 3 (to LLTCON)
SS _i	Control axis selection parameters where i = 1 to 3 for roll, pitch, and yaw axes; i = 4 for transition to a new axis (to LLTCON)
U, V	Sequential logic state parameters (to LLTCON)

5.2.4.3 Thruster Command Processing Function

The thruster command processing function (THRUSTER) computes the command rates and command attitudes (Euler symmetric parameters) that specify the desired spacecraft attitude and rate relative to the ECI frame for the thruster maneuver and velocity control modes. Initially, the command attitudes are the same as those specified by a ground command, and the command rates are set to zero. When the current flight software time (TF17) is greater than a specific value, the command rates are set to the values specified by ground control, and the command attitudes are turned at the command rates.

- Input--THRUSTR uses the following input:

<u>Name</u>	<u>Description</u>
DINIT _i	Initial attitude command where i = 1 to 4 from the GROUND (DBC)
DTCP _i	Updated commanded attitude quaternion where i = 1 to 4 (from THRUSTR)
ITURN	Commanded attitude turning flag (0 = do not rotate, 1 = rotate)
TCPINIT	Thruster command processing initialization flag, from the GROUND (DBC)
TF	Current flight software time from the executive
TSTH	Sampling period (computation interval) for THRUSTR (DBC)
TTURN	Time to start turning the command attitude, from the GROUND (DBC)
TTURNS	Saved time to start turning the command attitude (from THRUSTR)
WCI _i	Spacecraft roll, pitch, yaw turning rates where i = X, Y, Z ACAD body axes from the GROUND (DBCs)
WCS _i	Saved commanded turning rates where i = 1 to 3 ACAD axes (from THRUSTR)

● Processing--The thruster command processing function performs the following every 256 msec during the thruster maneuver and velocity control modes and before the performance of the attitude error computation function.

It determines if initialization is needed by discerning the value of the initialization flag TCPINIT.

If TCPINIT = 1

$$DTCP_i = DINIT_i \quad \text{for } i = 1 \text{ to } 4$$

$$TCPINIT = 0$$

$$WCS_i = WCI_i \quad i = 1 \text{ to } 3$$

$$TURNS = TTURN$$

$$ITURN = 0$$

where $DTCP_i$ is the updated commanded attitude (Euler parameters) for $i = 1$ to 4.

$TURNS$ is the saved turning time, and WCS_i is the saved commanded rate. Otherwise, the above values do not change.

The turning flag $ITURN$ is set in the following manner:

If ($ITURN = 0$) Then $ITURN = 1$

Otherwise, $ITURN = 0$.

The commanded rates used by $THRUSTR$ are set as follows:

If ($ITURN = 0$) Then

$WC_i = WCS_i$ for $i = 1$ to 4

EXIT $THRUSTR$ since the target quaternion does not need to be changed

Endif

If ($ITURN = 1$) Then

PROCEED WITH $THRUSTR$ since the target quaternion needs to be rotated at the ground command rate

Endif

One-half the total commanded angular increment is computed as follows:

$$B = 0.5 * TSTH * \left(WCS_X^2 + WCS_Y^2 + WCS_Z^2 \right)^{1/2}$$

where B is one-half the total commanded angular increment.

One-half $\sin(B)/B$ is computed using an approximation as follows:

$$S = 0.5 * (1.0 - B^2/6 + B^4/120)$$

where S represents approximations to $\sin(B)/B$.

The cosine of B is approximated as follows:

$$CB = (1.0 - B^2/2 + B^4/24)$$

where CB is an approximation to $\cos B$.

Intermediate parameters C_i are computed as follows:

$$C_1 = S * WCS_X * TSTH$$

$$C_2 = S * WCS_Y * TSTH$$

$$C_3 = S * WCS_Z * TSTH$$

Euler parameter increments are computed as follows:

$$DD_1 = C_3 * DTCP_2 - C_2 * DTCP_3 + C_1 * DTCP_4$$

$$DD_2 = -C_3 * DTCP_1 + C_1 * DTCP_3 + C_2 * DTCP_4$$

$$DD_3 = C_2 * DTCP_1 - C_1 * DTCP_2 - C_3 * DTCP_4$$

$$DD_4 = -C_1 * DTCP_1 - C_2 * DTCP_2 - C_3 * DTCP_3$$

where DD_i = Euler parameter update increments where $i = 1$ to 4

$DTCP_i$ = updated commanded Euler parameters for the velocity control mode where $i = 1$ to 4

The commanded Euler parameters are updated as follows:

$$DTCP_i = CB * DTCP_i + DD_i \quad \text{for } i = 1 \text{ to } 4$$

The sign of $DTCP_i$ is corrected as follows:

If $DTCP_4 < 0$

$$DTCP_i = -DTCP_i \quad \text{for } i = 1 \text{ to } 4$$

These parameters are normalized as follows:

$$E = 0.5 * \left(3 - DTCP_1^2 - DTCP_2^2 - DTCP_3^2 - DTCP_4^2 \right)$$

$$DTCP_i = E * DTCP_i \quad \text{for } i = 1 \text{ to } 4$$

Finally, the commanded rates are set equal to the values set by the ground:

$$WC_i = WCSI_i \quad \text{for } i = 1 \text{ to } 3$$

- Output--THRUSTR produces the following output:

<u>Name</u>	<u>Description</u>
DTCP _i	Updated command attitude Euler parameters, where $i = 1$ to 4, to attitude error computation function
ITURN	Command attitude turning flag (0 = do not turn; 1 = turn) (to THRUSTR)
TCPINIT	Thruster command processing initialization flag from ground (this is reset to zero by THRUSTR) (to THRUSTR)
TTURNS	Saved time to start turning the command attitude (to THRUSTR)
WC _i	Commanded spacecraft roll, pitch, and yaw rates, where $i = X, Y, Z$, to low-level thruster control law and high-level thruster control law functions
WCS _i	Saved commanded turning rates where $i = 1$ to 3 ACAD axes (to THRUSTR)

5.2.4.4 Magnetic Control Law Function

The magnetic control law function (MAGCON) uses the calculated angular momenta and Earth's magnetic field vector to compute the commands sent to the magnetic torquer drive electronics.

- Input--MAGCON uses the following input:

<u>Name</u>	<u>Description</u>
BE _i	Components of Earth's magnetic field vector along ACAD body axes (parameters from MAGPROC)
CALPHAW	Cosine of reaction wheel configuration base angle relative to ACAD body coordinates (DBC)
CBETAW	Cosine of reaction wheel configuration pyramidal inclination angle (DBC)
CENFLAG	Control flag for bias momentum loop (DBC)
CENT	Magnetic control law centering scheme constant (DBC)
CMDMAG	Flag that determines if MAGCON commands are allowed (= 0, commands are not allowed; = 1, commands are allowed) (from MODECON)
DCYLMT	Decay rate for bias momentum loop (DBC)
EADT ₁	Bias momentum decay rate constant (DBC)
EADT ₂	Peak momentum decay rate constant (DBC)
FAILW	Failed wheel indicator (DBC)
HW _i	Stored angular momentum for reaction wheel i where i = 1 to 4 (variables from MOMCOMP)
KBT _{ij}	Elements of 3 to 3 coordinate transformation from ACAD body coordinates where j = X, Y, Z to magnetic torquer coordinates where i = X, Y, Z (DBCs)
KMD	Maximum magnetic torquer capability constant (DBCs)
KM _i	Magnetic control law gain where i = X, Y, Z ACAD body axes (DBCs)
KMTS	Magnetic torquer bar (input data) scale factor (DBCs)
MAGINIT	Magnetic control law initialization flag (= 0, do not initialize; = 1, initialize) (from MODECON and reset to zero by MAGCON)

<u>Name</u>	<u>Description</u>
SALPHAW	Sine of reaction wheel configuration base angle relative to ACAD body coordinates (DBC)
SBETAW	Sine of reaction wheel configuration pyramidal inclination angle (DBC)
XLIM	Momentum bias limit (DBC)
XTDE _i	Magnetic torquer bar scale factor where i = X, Y, Z, magnetic torquer coordinates (DBC)
TSML	Sampling period (computation interval) for magnetic control law (DBC)

● Processing--MAGCON is executed every 512 msec during normal pointing mode and standby mode (commands not sent).

Initialization--The following initialization is performed if MAGINIT = 1. Otherwise, this section is skipped.

```

PEAKPi = 0
PEAKNi = 0
HCENTi = 0
COUNTERi = 0
MAGINIT = 0

```

where PEAKP_i = component i of positive momentum peak
where i = X, Y, Z

PEAKN_i = component i of negative momentum peak
where i = X, Y, Z

HCENT_i = momentum bias vector component in ACAD body axes where i = X, Y, Z

COUNTER_i = counter for i axis momentum loop delay
where i = X, Y, Z

The failed wheel flag (FAILW) is first examined, and the momentum input associated with a failed wheel is set to zero.

The momentum is first copied to a temporary variable as follows:

$$THW_i = HW_i \quad \text{for } i = 1 \text{ through } 4$$

IF FAILW = 0, no adjustment occurs.

Otherwise,

$$THW_j = 0 \quad \text{for } j = FAILW$$

NOTE: This algorithm may be modified to include additional failure options.

Body Momentum Computation--The reaction wheel momenta from each wheel are combined and transformed into ACAD body coordinate components:

$$H_X = (THW_1 + THW_2 + THW_3 + THW_4) * CBETAW$$

$$H_Y = [(THW_4 - THW_1) * CALPHAW + (THW_2 - THW_3) * SALPHAW] * SBETAW$$

$$H_Z = [(THW_1 - THW_4) * SALPHAW + (THW_2 - THW_3) * CALPHAW] * SBETAW$$

where H_i represents reaction wheel momentum components in ACAD body coordinates for $i = X, Y, Z$.

Peak Momentum Bias Computation--Momentum bias vector components (H_{CENT}_i for $i = X, Y, Z$) are calculated to center the angular momentum accumulation about each ACAD body axis. In this calculation, positive and negative momentum peaks are first identified and decayed exponentially. When an instantaneous momentum component exceeds the positive or negative

peak, it becomes the new peak. This is accomplished as follows for $i = X, Y, Z$ (ACAD body axes):

If $H_i > \text{PEAKP}_i$ counter $i = \text{counter } i + 1$

$$\text{PEAKP}_i = H_i$$

where $\text{PEAKP}_i =$ component i of positive momentum peak
and if $H_i < \text{PEAKN}_i$

$$\text{PEAKN}_i = H_i$$

where $\text{PEAKN}_i =$ component i of negative momentum peak

Peak Momentum Bias Decay

If $\text{COUNTER}_i > \text{DCYLMT}$
 $\text{PEAKP}_i = \text{PEAKP}_i * \text{EADT2}$
 $\text{PEAKN}_i = \text{PEAKN}_i * \text{EADT2}$
 $\text{HCENT}_i = \text{HCENT}_i * \text{EADT1}$
 $\text{COUNTER}_i = 0$

The bias vector is then computed from its previous value and a constant times the sum of the positive and negative momentum peaks, as follows:

$$\text{HCENT}_i = \text{HCENT}_i + \text{TSML} * \text{CENT} * (\text{PEAKP}_i + \text{PEAKN}_i) * \text{DCYLMT}$$

Finally, the momentum bias vector components are limited as follows (for each i):

If $|\text{HCENT}_i| > \text{XLIM}$

$$\text{HCENT}_i = \text{XLIM} * \text{sign}(\text{HCENT}_i)$$

Otherwise, there is no change in HCENT_i .

where $\text{HCENT}_i =$ momentum bias vector components in ACAD body axes where $i = X, Y, Z$

TSML = MAGCON sampling period (512 msec)
 DCYLMT = decay rate for momentum bias loop
 CENT = magnetic control law centering constant

Dumped Momentum Computation--The momentum that the magnetic control law attempts to "dump" is computed based on the value of the momentum bias flag (CENFLAG). If CENFLAG is 0, the momentum bias is ignored and the momentum to be dumped is simply the sensed reaction wheel momentum in body coordinates. If CENFLAG is 1, the momentum to be dumped is the vector sum of the sensed reaction wheel momentum and the momentum bias in body coordinates. This is accomplished as follows:

If CENFLAG = 1

$$HD_i = (H_i + HCENT_i) * (1.35582) \quad \text{for } i = X, Y, Z$$

Otherwise (CENFLAG \neq 1)

$$HD_i = H_i * (1.35582) \quad \text{for } i = X, Y, Z$$

where HD_i = instantaneous momentum components to be zeroed by the magnetic control law

The 1.35582 term is used to convert angular momentum from foot-pound-seconds to newton-meter-seconds.

Torquer Bars Magnetic Moment Computation--The cross-product vector between the Earth's magnetic field vector and the momentum vector to be dumped is computed as

$$\vec{M} = \vec{B} \times \vec{HD}$$

where $\vec{M} = (M_x, M_y, M_z)$ = cross-product vector of Earth's magnetic field vector and momentum vector to be dumped in newton-meter-seconds-gauss

$\vec{B} = (BE_x, BE_y, BE_z)$ = Earth's magnetic field measured by the magnetometers in gauss

$\vec{HD} = (HD_x, HD_y, HD_z)$ = excess momentum to be dumped in newton-meter-seconds

The commanded magnetic moment vector components are defined as

$$M_i = -KM_i * M_i \quad i = X, Y, Z$$

where KM_i = magnetic control low gain; units:
1/second-gauss-tesla

M_i = commanded magnetic moment components in ACAD body
ACAD body axes for $i = X, Y, Z$, in ampere-meters²

This vector is transformed into magnetic torquer coordinates as follows:

$$\begin{bmatrix} \text{MTORQ}_X \\ \text{MTORQ}_Y \\ \text{MTORQ}_Z \end{bmatrix} = \begin{bmatrix} \text{KBT}_{XX} & \text{KBT}_{XY} & \text{KBT}_{XZ} \\ \text{KBT}_{YX} & \text{KBT}_{YY} & \text{KBT}_{YZ} \\ \text{KBT}_{ZX} & \text{KBT}_{ZY} & \text{KBT}_{ZZ} \end{bmatrix} \begin{bmatrix} M_X \\ M_Y \\ M_Z \end{bmatrix}$$

where MTORQ_i are the commanded magnetic moment components in magnetic torquer coordinates in ampere-meters².

The commands are converted to counts, which become the commands sent to the magnetic torquers as follows:

$$\text{MTORQ}_i = \text{Rnd}(\text{MTORQ}_i * \text{XTDE}_i) \quad \text{for } i = X, Y, Z$$

where $\text{RND}(A) = \text{sign of } A \text{ times the largest integer } \leq \text{ABS}(A + (0.5) * \text{sign}(A))$

These commands are limited as follows:

If $|\text{MTORQ}_i| > \text{KMD}$

$$\text{MTORQ}_i = \text{KMD} * \text{sign}(\text{MTORQ}_i) \quad \text{for } i = X, Y, Z$$

Otherwise, there is no change in MTORQ_i for $i = X, Y, Z$.

The torquer bar commands are then converted from counts to ampere-meters² so they can be shipped to the Truth Model:

$$C.ML_i = MTORQ_i / XTDE_i \quad \text{for } i = X, Y, Z \text{ magnetic torquer axis}$$

If CMDMAG = 1, the commands are sent to the torquer bars.

If CMDMAG ≠ 1, the commands are not sent to the torquer bars.

- Output--MAGCON produces the following output:

<u>Name</u>	<u>Description</u>
C.ML _i	Commanded magnetic moment components in counts sent to the TDE each computation cycle where i = X, Y, Z magnetic torquer axes (to Truth Model)
MAGINIT	Magnetic control law (MAGCON) initialization flag; reset to zero by MAGCON (to MAGCON)

5.2.4.5 Wheel Control and Distribution Function

The wheel control and distribution function (WHECON) computes reaction wheel torque commands required to maneuver, maintain spacecraft attitude, and distribute momentum among the wheels during the normal maneuver and pointing modes as shown below. It uses attitude error, gyro data compensation, and stored momentum data in performing these computations.

- Input--WHECON uses the following input:

<u>Name</u>	<u>Description</u>
ACSINT	Sampling period (computation interval) for WHECON (from MODECON)
DW _{ijk}	Elements for transforming requested torques in ACAD body axes to reaction wheel torque commands where i = 0 to 4 (i = FAILW), j = 1 to 4 (wheel), and k = 1 to 3 (ACAD body axes) (DBC's)
E _i	Attitude errors in roll, pitch, and yaw, i.e., commanded-minus-actual angles about the ACAD body axes where i = X, Y, Z (variables from ATTERR)

<u>Name</u>	<u>Description</u>
ELO _i	Default angular error limits where i = X, Y, Z ACAD body axes (DBC)
ETHRESH	Threshold for default angular error limits (DBC)
FAILW	Failed wheel indicator (= 0, no wheel failed; = i, wheel i failed where i = 1 to 4) (DBC/GND)
HBIAS	Momentum distribution bias (DBC)
HW _i	Stored angular momentum for reaction wheel i where i = 1 to 4 (variables from MOMCOMP)
IEL _i	Integral error limits for roll, pitch, and yaw where i = X, Y, Z ACAD body axes (DBC)
KD	Momentum distribution time constant (DBC)
KWC _i	Reaction wheel torque command scale factors where i = 1 to 4 (DBC)
PEE _i	Past value of wheel momentum estimate where i = 1 to 4 (from WHECON)
PHWS _i	Past value of wheel momentum estimate where i = 1 to 4 (from WHECON)
PIE _i	Past value of limited integral error where i = 1 to 3 (from WHECON)
PTW _i	Past value of wheel torque commands where i = 1 to 4 (from WHECON)
TEX _i	Reaction wheel friction estimate for wheel i where i = 1 to 4 (DBC)
TE1 _i	Torque filter state 1 where i = 1 to 3 (from WHECON)
TE2 _i	Torque filter state 2 where i = 1 to 3 (from WHECON)
TF1 _i	Filtered torque state 1 where i = 1 to 3 (from WHECON)
TF2 _i	Filtered torque state 2 where i = 1 to 3 (from WHECON)
TL _i	Torque limit in roll, pitch, and yaw where i = X, Y, Z ACAD body axes (DBC)
TRWINT	Time interval for updating estimated wheel momentum (DBC)
VEL _i	Variable error limit constants, where i = X, Y, Z (DBC)

<u>Name</u>	<u>Description</u>
W _i	Compensated spacecraft body rates in roll, pitch, and yaw where i = X, Y, Z body axes (variable from GYRCOMP)
WCA _i	Filter constant A where i = 1 to 3 (DBC)
WCA1 _i	Filter constant A1 where i = 1 to 3 (DBC)
WCA2 _i	Filter constant A2 where i = 1 to 3 (DBC)
WCB1 _i	Filter constant B1 where i = 1 to 3 (DBC)
WCB2 _i	Filter constant B2 where i = 1 to 3 (DBC)
WHEINIT	WHECON initialization flag; = 0, no; = 1, yes (from MODECON, WHECON)
WKI _i	Wheel control law integral gain where i = 1 to 3 ACAD axes (DBC)
WKP _i	Wheel control law proportional gain where i = 1 to 3 ACAD axes (DBC)
WKR _i	Wheel control law rate gain where i = 1 to 3 ACAD axes (DBC)
WL _i	ACAD body axes rate limits where i = X, Y, Z ACAD body axes (roll, pitch, yaw) (DBCs)
WTL _i	Wheel torque limits where i = 1 to 4 for wheels 1 to 4 (DBCs)

● Processing--WHECON is executed every 512 msec during the normal pointing and normal maneuver modes. It is performed after ATTERR and MOMCOMP.

Initialization--If WHEINIT = 1, the following initialization is performed:

$PIE_i = 0$ for i = 1 to 3 (past integral error)
 $WOK_i = 1$ for i = 1 to 4 (wheel health status flag)
 $PHWS_i = HW_i$ for i = 1 to 4
 $PTW_i = TRW_i = 0$ for i = 1 to 4
 $ICNT = TRWINIT/ACSINT$

Variable Error Limits--Angular error limits for roll, pitch, and yaw are calculated as follows:

$$EMAG = (E_X * E_X + E_Y * E_Y + E_Z * E_Z)^{1/2}$$

where EMAG is the magnitude of the total angular position error.

If EMAG > ETHRESH,

$$EL_i = |E_i| * VEL_i / EMAG \quad \text{for } i = X, Y, Z$$

where EL_i = variable error limits for $i = X, Y, Z$
 $i = X, Y, Z$ ACAD body axes

Otherwise,

$$EL_i = ELO_i \quad \text{for } i = X, Y, Z$$

where ELO_i are the default values.

• Control Law--The following is a proportional plus integral plus derivative control scheme that calculates torque commands along the ACAD body axes. EE_i , WW_i , and IE_i are local variables, and PEE_i and PIE_i are the past values of EE_i and IE_i , respectively. The control law error output (TE_i) is filtered and limited for $i = X, Y, Z$.

If $EL_i < |E_i|$,

$$EE_i = EL_i * \text{sign}(E_i)$$

where EE_i = limited angular errors in roll, pitch, and yaw for $i = X, Y, Z$ ACAD body axes

$\text{sign}(a) = 1$ if $a \geq 0$, $\text{sign}(a) = -1$, if $a < 0$

Otherwise,

$$EE_i = E_i$$

IF WHENINIT = 1

$$\text{Then } PEE_i = EE_i$$

where PEE_i is the past value of the limited angular errors (see below).

The ACAD body rates in roll, pitch, and yaw are limited by performing the following successively for $i = X, Y, Z$:

$$\text{If } WL_i < |W_i|$$

$$WW_i = WL_i * \text{sign}(W_i)$$

where WW_i represents limited body axes rates and $i = X, Y, Z$ for roll, pitch, and yaw.

Otherwise,

$$WW_i = W_i \quad \text{for } i = X, Y, Z$$

The integral error in roll, pitch, and yaw is computed as

$$IE_i = PIE_i + ACSINT * (EE_i + PEE_i)/2 \quad \text{for } i = X, Y, Z$$

where IE_i = integral error in roll, pitch, and yaw
 PIE_i = past value of integral error in roll, pitch, and yaw (see below)
 PEE_i = past value of limited angular error in roll, pitch, and yaw (see below)
 $i = X, Y, Z$ ACAD body axes

The integral errors are limited by performing the following successively for $i = X, Y, Z$:

If $|IE_i| > IEL_i$

$$IE_i = IEL_i * \text{sign}(IE_i)$$

Otherwise, there is no change in I_i .

The past values are updated as follows:

$$PEE_i = EE_i \quad \text{for } i = X, Y, Z$$

$$PIE_i = IE_i \quad \text{for } i = X, Y, Z$$

The control law error output is computed as follows:

$$TE_i = WKP_i * EE_i + WKI_i * IE_i - WKR_i * WW_i \quad \text{for } i = X, Y, Z$$

where TE_i are requested torques about the X, Y, Z ACAD body axes (roll, pitch, yaw).

If WHECON is being initialized (WHEINIT = 1), the following is performed:

$$TF2_i = TF1_i = TE2_i = TE1_i = TE_i \quad \text{for } i = X, Y, Z$$

where $TE1_i$ = torque filter state 1
 $TE2_i$ = torque filter state 2
 $TF1_i$ = filtered torque state 1
 $TF2_i$ = filtered torque state 2

TFK_i is limited as follows:

If $ABS(TFK_i) > TL_i$

Then $TFK_i = TL_i * \text{sign}(TFK_i)$ for $K = 1, 2$

Otherwise, TFK_i is not changed.

The filter control law error (TF_i) is then computed in ACAD coordinates:

$$TF_i = WCA_i * TE_i + WCA1_i * TE1_i + WCA2_i * TE2_i \\ - WCB1_i * TF1_i - WCB2_i * TF2_i$$

where WCA, WCA1, WCA2, WCB1, and WCB2 are the filter constants.

The filter control law error is then limited as follows:

$$\text{If } ABS(TF_i) > TL_i \\ TF_i = TL_i * SIGN(TF_i)$$

Otherwise, TF_i remains unchanged.

The torque states are then saved for the next iteration:

$$TF2_i = TF1_i \\ TF1_i = TF_i \\ TE2_i = TE1_i \\ TE1_i = TE_i$$

The WHECON initialization flag is then set to zero:

$$WHEINIT = 0$$

Otherwise, there is no change in T_i .

• Steering Law, Momentum Distribution, and Command Output--The steering law computes the torques for each

reaction wheel as a function of which wheel (if any) is disabled. The torque for each axis is computed as follows:

$$\begin{bmatrix} TW_1 \\ TW_2 \\ TW_3 \\ TW_4 \end{bmatrix} = \begin{bmatrix} DW_{i11} & DW_{i12} & DW_{i13} \\ DW_{i21} & DW_{i22} & DW_{i23} \\ DW_{i31} & DW_{i32} & DW_{i33} \\ DW_{i41} & DW_{i42} & DW_{i43} \end{bmatrix} \begin{bmatrix} TF_X \\ TF_Y \\ TF_Z \end{bmatrix}$$

where TW_j = torque for wheel j , $j = 1$ to 4
 i = FAILW = failed wheel indicator where $i = 0$ for no failure, $i = 1$ to 4 for wheel 1 to 4 failure

Momentum distribution is accomplished as follows:

If FAILW = 0, the momentum distribution law is implemented:

$$HDIS = [1/4 * (HW_1 - HW_2 - HW_3 + HW_4) - HBIAS] * KD$$

$$TW_1 = TW_1 - HDIS$$

$$TW_2 = TW_2 + HDIS$$

$$TW_3 = TW_3 + HDIS$$

$$TW_4 = TW_4 - HDIS$$

where TW_j represents torque commands to wheels 1 to 4, $j = 1$ to 4.

Otherwise (FAILW \neq 0), and no momentum distribution computations are made.

The wheel torques, TW_i , are then limited as follows:

If $ABS(TW_i) > WTL_i$ for $i = 1$ to 4

Then $TW_i = WTL_i * SIGN(TW_i)$

Otherwise, TW_i remains unchanged.

An estimate for wheel friction is then added:

$$TCOM_i = TW_i + TEX_i * SIGN(HW_i) \quad \text{for } i = 1 \text{ to } 4$$

The torque command is rounded down to an integer value. If a particular wheel has been identified as failed, the torque command for that particular wheel is set to 0:

```
For i = 1 to 4
  If i ≠ FAILW Then
    C.WCTi = RND (TCOMi * KWCi)
  Else
    C.WCTi = 0
Endif
```

The wheel torque commands, $C.WCT_i$, are formed from TW_i for $i = 1$ to 4 and are output to the Truth Model each computational cycle:

$$C.WCT_i = C.WCT_i / KWC_i$$

Wheel Failure Detection--Wheel failures are determined by comparing the estimated wheel momentum based on integration of the reaction wheel torque commands (HWS_i) with the measured momentum (HW_i) from MOMCOMP. If the difference exceeds the error threshold, HWET, the health status flag, WOK_i , is set off:

```
For i = 1 to 4
  HWSi = PHWSi + ACSINT * (TWi + PTWi)/2.0
  PWSi = HWSi
  PTWi = TWi
  If ABS(HWi - HWSi) > HWET and CMDWHE = 1 Then WOKi = 0
```

At the end of each TRWINT computation period, the estimated wheel momentum is reset to the actual momentum, HW.

```

ICNT = ICNT - 1
IF ICNT ≤ 0 Then
    ICNT = TRWINT/ACSINT
    For i = 1 to 4
        PHWSi = HWi
    Enddo
Endif

```

- Output--WHECON produces the following output:

<u>Name</u>	<u>Description</u>
C.WCT _i	Torque commands to wheels 1 to 4 where i = 1 to 4 (ft-lbf) (to Truth Model)
ICNT	Momentum update time counter (to WHECON)
PEE _i	Past value of limited error where i = 1 to 3 (to WHECON)
PHWS _i	Past value of estimated wheel momentum where i = 1 to 4 (to WHECON)
PIE _i	Past value of limited integral error where i = 1 to 3 (to WHECON)
PTW _i	Past value of torque command to wheel where i = 1 to 4 (to WHECON)
TE1 _i	Torque filter state 1 where i = 1 to 3 (to WHECON)
TE2 _i	Torque filter state 2 where i = 1 to 3 (to WHECON)
TF1 _i	Filtered torque state 1 where i = 1 to 3 (to WHECON)
TF2 _i	Filtered torque state 2 where i = 1 to 3 (to WHECON)
WHEINIT	WHECON initialization flag (to WHECON)
WOK _i	Wheel health status flag where i = 1 to 4; = 0, bad; = 1, good (to SAFECON)

5.2.4.6 Mode Control Function

The mode control function (MODECON) controls the transitions between the various modes of operation of the ACAD subsystem. Input consists of the mode transition request from the ground and from the SAFECON, HLTCON, and MODECON software functions.

MODECON examines the various requests in priority order and transitions to the new mode if the requested transition is valid. MODECON also implements the test for Sun acquisition in the Sun reference pointing mode (SRPM). If the test fails, MODECON requests itself to make the transition to the safe hold mode (SHM).

The ACAD modes of operation controlled by the OBC are defined as follows:

<u>ACAD Mode Number</u>	<u>Definition</u>
1	Standby mode (SM)
2	Normal pointing mode (NPM)
3	Normal maneuver mode (NMM)
4*	Sun reference pointing mode (SRPM)
5*	Safe-hold mode (SHM)
6	Thruster maneuver mode (TMM)
7	Velocity control mode (VCM)

The asterisks (*) indicate modes that are not controlled by the OBC but by the control processing electronics (CPE) firmware. Upon transition to a CPE-controlled mode from an OBC-controlled mode, the OBC informs the CPE which actuators are in use and the status of the gyro and gyro channels.

Figures 5-12 through 5-16 describe the processes for each of the above OBC-controlled modes. Table 5-5 lists the various module processing times, and Table 5-6 lists the initial MODECON variables for each OBC mode.

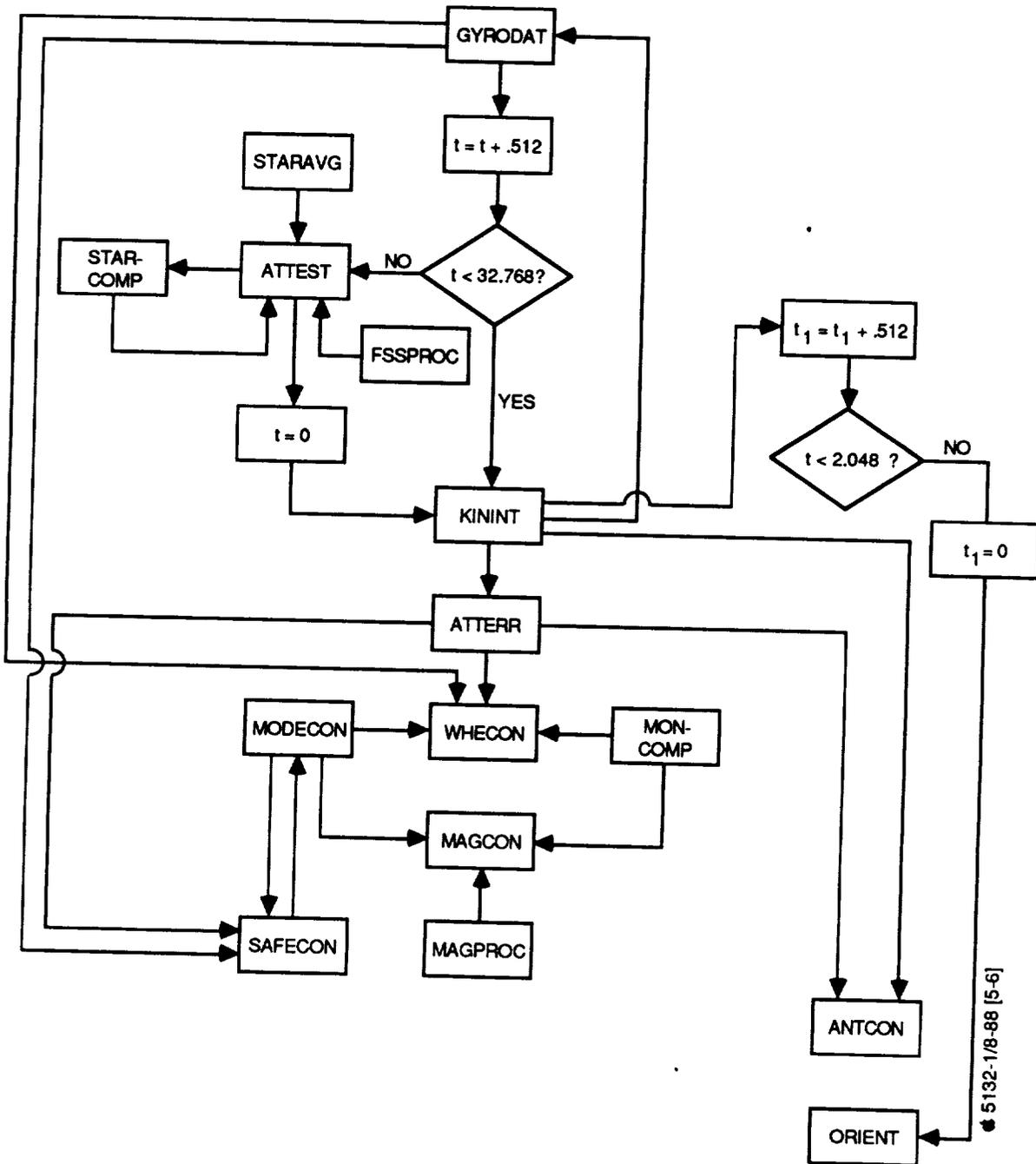


Figure 5-12. ACAD Mode 1: Standby Mode

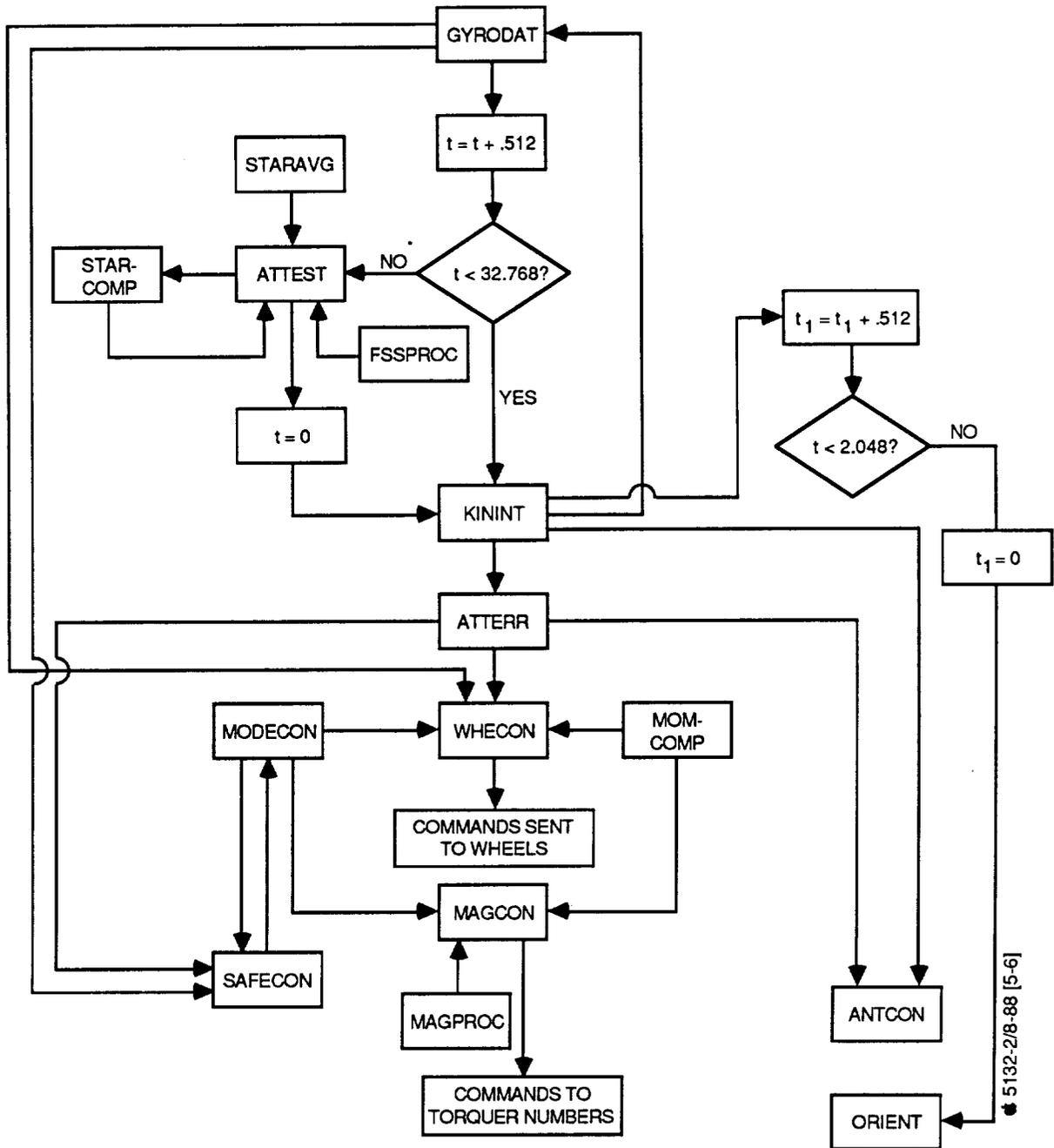


Figure 5-13. ACAD Mode 2: Normal Pointing Mode

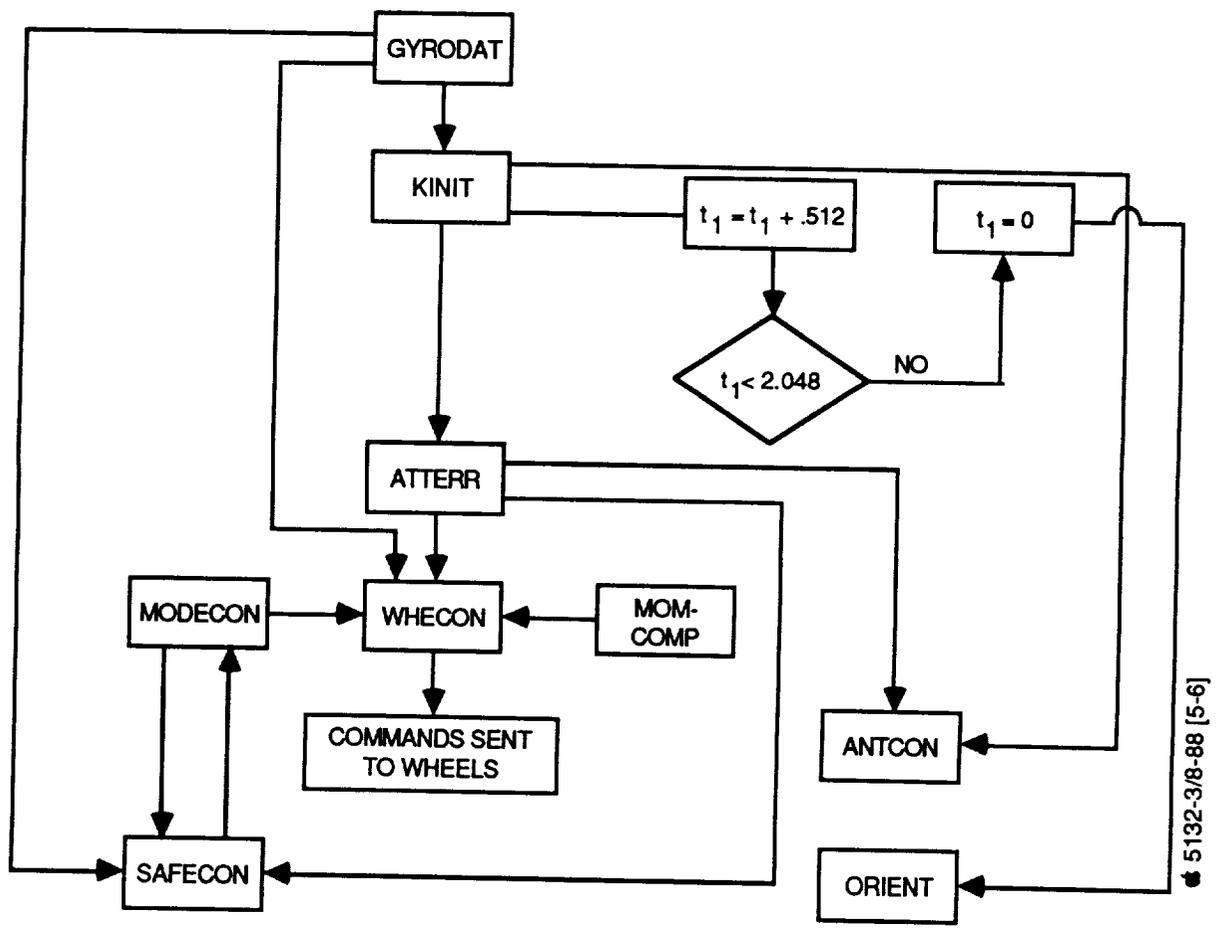


Figure 5-14. ACAD Mode 3: Normal Maneuver Mode

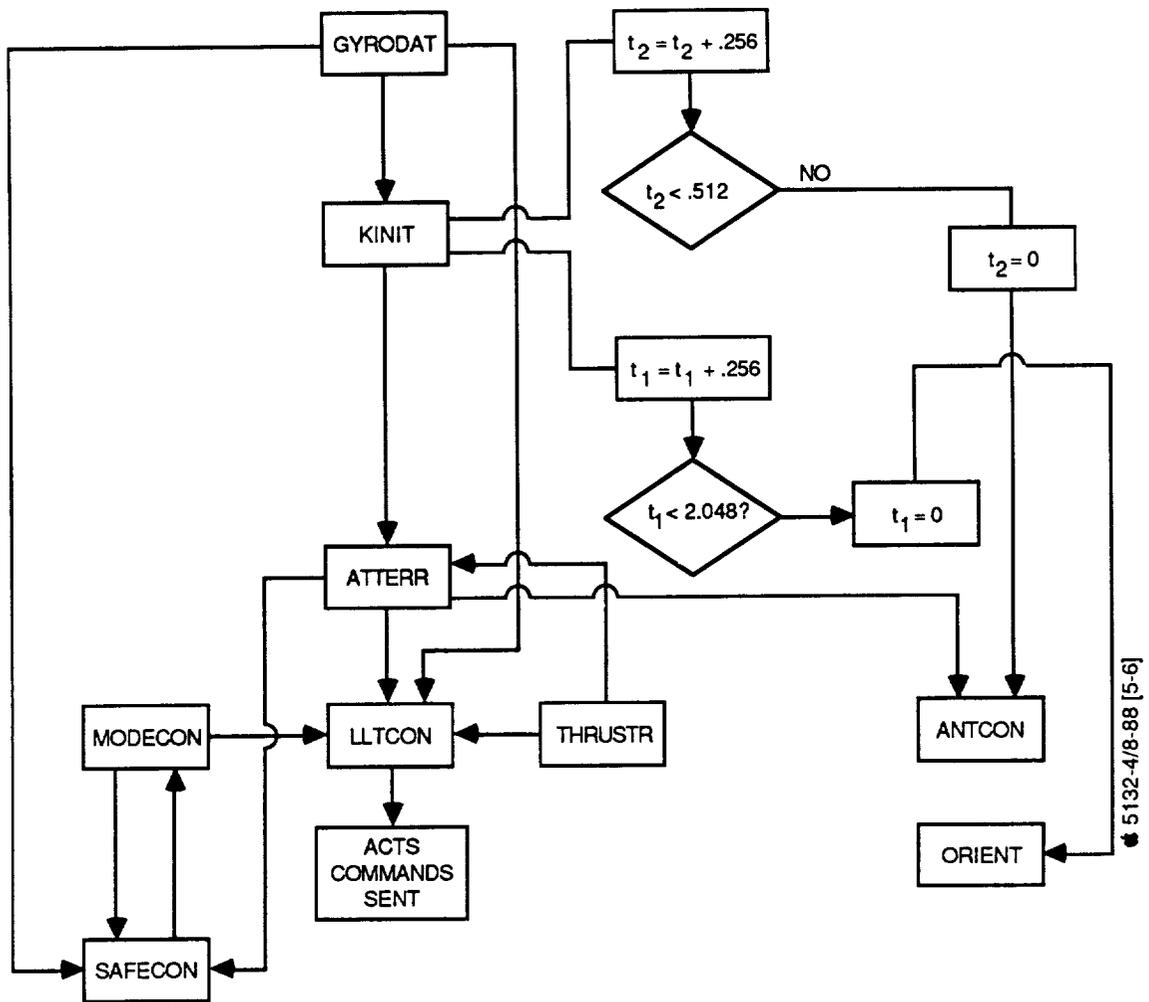


Figure 5-15. ACAD Mode 6: Thruster Maneuver Mode

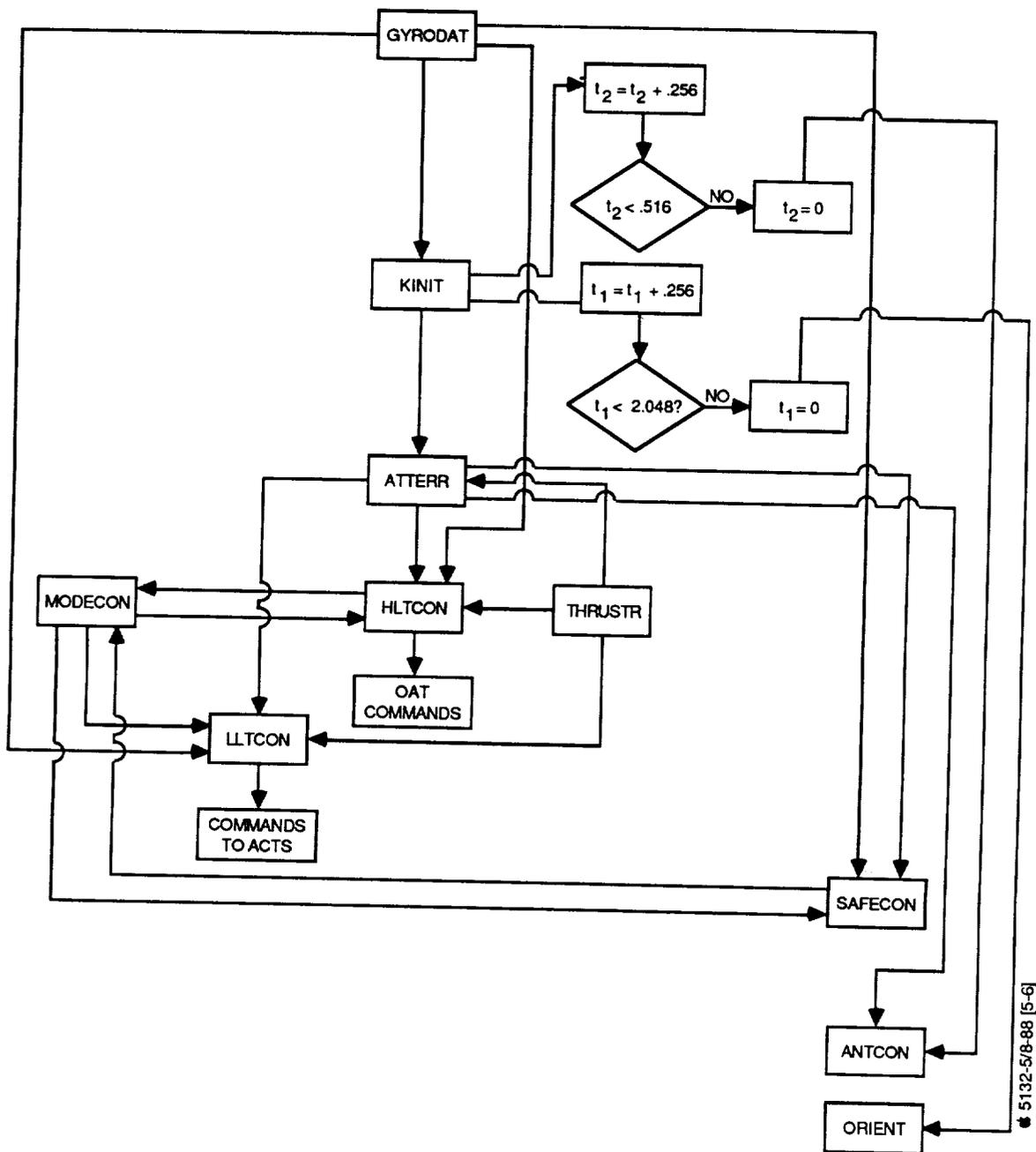


Figure 5-16. ACAD Mode 7: Velocity Control Mode

Table 5-5. Functions Executed by ACAD Mode, Frequency

Mode Number Mode Name	1 SM	2 NPM	3 NMM	4 SRPM	5 SHM	6 TMM	7 VCM
<u>Function</u>							
MODECON	0.256	0.256	0.256	0.256	0.256	0.256	0.256
GYRODAT	0.256	0.256	0.256	0.256	0.256	0.256	0.256
FSSPROC	32.768	32.768		32.768	32.768		
MAGPROC	0.512	0.512		0.512	0.512		
MOMCOMP	0.256	0.256	0.256	0.256	0.256	0.256	0.256
STARAVG	0.256	0.256		0.256	0.256		
KININT	0.512	0.512	0.512	0.512	0.512	0.256	0.256
(1)EPHEM	2.048	2.048	2.048	2.048	2.048	2.048	2.048
ORIENT	2.048	2.048	2.048	2.048	2.048	2.048	2.048
OCCULT	2.048	2.048	2.048	2.048	2.048	2.048	2.048
ATTEST	32.768	32.768		32.768	32.768		
STARDAT	32.768	32.768		32.768	32.768		
THRUSTR						0.256	0.256
ATTERR	0.512	0.512	0.512	0.512	0.512	0.256	0.256
HLTCON							0.256
LLTCON						0.256	0.256
MAGCON	0.512	0.512		0.512	0.512		
(2)ANTCON	0.128	0.128	0.128	0.128	0.128	0.128	0.128
WHECON	0.512	0.512	0.512	0.512	0.512		
(3)SAFECON		0.512	0.512	65.536	65.536	0.256	0.256

- NOTES:
1. EPHEM also has long period processors that execute every 1200.128 sec (about 20 min) in all modes.
 2. ANTCON initialization, TDRS selection, and gimbal angle generation functions are executed every 0.512 sec in all modes.
 3. ACAD monitor functions of SAFECON are performed every 0.512 sec in modes 2 and 3 and every 0.256 sec in modes 6 and 7.
 4. This table reflects dependencies spelled out in some of the processing sections of various modules (e.g., ANTCON is executed after ATTERR and before WHECON).

Table 5-6. ACAD Flags and Constants Set on Mode Initiation
(1 of 2)

<u>Mode Number</u> <u>Mode Name</u>	<u>1</u> <u>SM</u>	<u>2</u> <u>NPM</u>	<u>3</u> <u>NMM</u>	<u>4</u> <u>SRPM</u>	<u>5</u> <u>SHM</u>	<u>6</u> <u>TMM</u>	<u>7</u> <u>VCN</u>
GYRINIT	1	0	0	0	0	0	0
FLTINIT	1	1	0	1	1	0	0
MAGINIT	1	1	0	1	1	0	0
WHEINIT	1	1	1	1	1	0	0
ANTINIT	1	1	0	0	0	0	0
LLTINIT	0	0	0	0	0	1	1
HLTINIT	0	0	0	0	0	0	1
SCMINIT	1	0	0	0	0	0	0
MOMINIT	1	0	0	0	0	0	0
CMDANT	0	1	0	0	0	0	0
CMDWHE	0	1	1	0	0	0	0
CMDMAG	0	1	0	0	0	0	0
CMDBKT	0	1	0	0	0	0	0
ACADM	1	2	3	4	5	6	7
ACSINT	512	512	512	512	512	256	256
LLF _i						1	1
ISTCNT _j	0	0		0	0		
STRAVG _j	0	0		0	0		
TFSCAN _j				1	1		
AECT		1	2			3	4
ARCT		1	2			3	4
GCCT		1	2	3	4	5	6
SGCL		1	2	3	4	5	6
TGCL		1	2	3	4	5	6
THGCL		1	2	3	4	5	6
LIMKEY	1	2	3	4	5	6	

Table 5-6. ACAD Flags and Constants Set on Mode Initiation
(2 of 2)

Mode Number <u>Mode Name</u>	<u>1</u> <u>SM</u>	<u>2</u> <u>NPM</u>	<u>3</u> <u>NMM</u>	<u>4</u> <u>SRPM</u>	<u>5</u> <u>SHM</u>	<u>6</u> <u>TMM</u>	<u>7</u> <u>VCM</u>
LCA _i						1	2
LCB _i						1	2
LCC _i						1	2
LKP _i						1	2
LKR _i						1	2
LPL _i						1	2
LHY _i						1	2

- NOTES:
1. ACSINT is in units of milliseconds.
 2. The numbers given for AECT through LHY_i are the values of j to use in setting these constants from the corresponding data base values AECT_j through LHY_{ij}, respectively.

- Input--MODECON uses the following input:

<u>Name</u>	<u>Description</u>
ACADMD	Current ACAD mode (from MODECON)
ACADMDA	Request for mode transition (from MODECON)
ACADMDG	Request for mode transition (from DBC/GND, MODECON)
ACADMDS	Request for mode transition (from SAFECON, MODECON)
ACADMDT	Request for mode transition (from HLTCN, MODECON)
ACSINT	ACS sample period (from MODECON)
AECT _j	Attitude error comparison thresholds where j = 1 to 6 (DBC)
AQTAK	Sun acquisition time counter A (from MODECON)
AQTAL	Sun acquisition time limit A (DBC)
AQTBK	Sun acquisition time counter B (from MODECON)
AQTBL	Sun acquisition time limit B (DBC)
ARCT _j	Attitude rate comparison thresholds where j = 1 to 6 (DBC)
B1ON	High-level thruster bank 1 (pitch) on-flag; = 0, off; = 1, on (DBC set by the ground)
B2ON	High-level thruster bank 2 (roll) on-flag; = 0, off; = 1, on (DBC set by the ground)
DW _{i,j,k}	Coordinate transformation from ACAD axes (where i = 1 to 3) to reaction wheel axes (where j = 1 to 4, k = FAILW = 0 to 4 DBC)
FAILW	Failed reaction wheel flag (DBC reset by ground and SAFECON)
GCCT _j	Gyro channel comparison thresholds where j = 1 to 6 (DBC)
IGY _i	Gyro channel selection flags where i = 1 to 3 (DBC reset by ground and SAFECON)
LCA _{i,j}	LLTCN rate filter constant A where i = 1 to 3 ACAD axes; j = 1 to 2 for TMM, VCM (DBC)
LCB _{i,j}	LLTCN rate filter constant B where i = 1 to 3 ACAD axes; j = 1 to 2 for TMM, VCM (DBC)
LCC _{i,j}	LLTCN rate filter constant C where i = 1 to 3 ACAD axes; j = 1 to 2 for TMM, VCM (DBC)

<u>Name</u>	<u>Description</u>
LHY _{i,j}	LLTCON hysteresis constants where i = 1 to 3; j = 1 to 2 (DBC)
LIMKEY _j	ANTCON keyhole where j = 1 to 6 (DBC)
LKP _{i,j}	LLTCON position gains where i = 1 to 3; j = 1 to 2 (DBC)
LKR _{i,j}	LLTCON rate gains where i = 1 to 3; j = 1 to 2 (DBC)
LPL _{i,j}	LLTCON position error limits where i = 1 to 3; j = 1 to 2 (DBC)
MODEX	Mode transition control flag (from MODECON)
RECREQ	Request for reconfiguration of ACAD hardware (from SAFECON)
SBTIMEK	Standby mode time counter
SBTIMEL	Time unit for standby mode (DBC)
SGCL _j	Single-axis gyro compare limits where j = 1 to 6 (DBC)
SYNCK	ACAD hardware resynchronization time counter (from MODECON)
SYNCTM	ACAD hardware synchronization time (DBC)
SUNPRS	Sun present flag (from FSSPROC)
TGCL _j	Two-axis gyro compare limits where j = 1 to 6 (DBC)
THGCL _j	Three gyro compare limits where j = 1 to 6 (DBC)

● Processing--MODECON is performed every 256 msec in all modes.

Initialization

```

ACADM = 0
ACADMG = 0
ACAMDT = 0
ACAMDS = 0
RECREQ = 0
MODEX = 0
ACAMDA = 1

```

Mode Transition Validity Test--Ground-requested mode transitions are performed only when the requested mode is different from the current mode, precluding reinitializations in the ACS software functions without an explicit mode change. The allowable mode transitions are shown in Table 5-7.

Concurrent mode transition requests are executed in the following priority order:

1. ACADMDG > 0 Ground request
2. ACADMDA > 0 MODECON request
3. ACADMDS > 0 SAFECON request
4. ACADMDT > 0 HLTCO request

If more than one request is pending, only the highest priority is honored. (Lower priority pending requests are canceled.) If a valid transition is requested, the MODEXIN and MODEX flags are set to control mode initiation and mode transition as described in Table 5-7 and the safe contingency mode transition processing. Invalid requests and canceled requests are telemetered to the ground.

The processing in this section is performed only if MODEX = 0 (i.e., no mode transition is currently in progress).

If MODEX = 0

If the ground commands the spacecraft to transition to another mode, this portion of the code is used:

```
Then if (ACADMDG = (1-6) and ACADMD = (1-6) and
      ACADMDG < > ACADMD or
      ACADMDG = (1, 5, or 6) and ACADMD = 7 or
      ACADMDG = 7 and ACADMD = 6)
Then MODEX = ACADMDG
```

Table 5-7. ACAD Mode Transition Matrix

	1	2	3	4	5	6	7
SBM	G,S	G	G	G	G	G	
G	NPM	G	G,S	G,S	G		
G	G	NMM	G,S	G,S	G		
G	G	G	SRPM	G,A	G		
G	G	G	G	SHM	G		
G	G	G	G	G,S	TMM	G	
G				G,S	G,T	VCM	

5132G(5-6)-06

- NOTES:
1. G = Ground command
S = SAFECON request
T = HLTCON request
A = MODECON request
 2. The ACAD modes are shown on the diagonal matrix elements and allowable transactions are indicated by an entry in the off-diagonal matrix element connecting the two modes in question. Blank off-diagonal elements indicate unallowable transitions. Transitions take place clockwise.
 3. Hardware configurations are not shown, since the mode definition does not depend on configuration.
 4. The initial transition to SBM at OFS initialization is not shown.

If MODECON is commanding a mode transition, this portion of the code is used:

```
Else if (ACADMDA = 1 and ACADMD = 0 or
         ACADMDA = 5 and ACADMD = 4)
Then MODEX = ACADMDA
```

If SAFECON is commanding a transition to either SRPM or SHM, the following code is used:

```
Else if (ACADMDS = 4 and ACADMD = (2 or 3) or
         ACADMDS = 5 and ACADMD = (2, 3, 6, or 7))
Then MODEX = ACADMDS
```

If the spacecraft is in the VCM and the mode is finished, the transition to the TMM is performed in this portion of the code:

```
Else if (ACADMDT = 6 and ACADMD = 7)
Then MODEX = ACADMDT
Endif      !(MODEX = 0)
```

If a mode transition is in progress already, this code is used:

```
If MODEX ≠ 0
Then MODEXIN = 1, PREVMODE = ACADMD
Else
    MODEXIN = 0
Endif
ACADMDG = 0
ACADMDA = 0
ACADMDS = 0
ACADMDT = 0
```

The MODEX flag < > 0 indicates that a transition to ACAD mode MODEX is in progress. MODEX = 0 means that the previous transition is completed.

Mode Initiation Processing--This portion of the code is performed in MODEXIN = 1. This flag indicates that a new mode has been commanded. Table 5-5 indicates the functions executed by the various modes as well as their frequency. Table 5-6 describes the various variables that are initialized upon mode transition. Table 5-8 describes the various commands sent upon mode initialization.

Additional Mode Transition Processing--In the VCM, the following commands are issued and flags set depending on high-level thruster bank enablement. If an OAT bank is enabled, the ACTs are not used to control that particular axis.

```
If MODEX = 7 (VCM)
    Then if B2ON = 1 (roll high-level thruster bank on)
        Then issue command C.LLTDR to disable low-level
        roll thrusters
        LLF1 = 0 (disable LLTCON roll control)
    If B1ON = 1 (pitch high-level thruster bank on)
        Then issue command C.LLTDP to disable low-level
        pitch thrusters
        LLF2 = 0 (disable LLTCON pitch control)
    Endif
    Endif
Endif
```

High-level thrusters are enabled in the VCM by the HLTCON function.

Upon entry to SM, the standby counter is reset to 0 as follows:

```
If (MODEX = 1) SBTIMEK = 0
```

Table 5-8. Commands Issued on Mode Initiation

Mode Number	1	2	3	4	5	6	7
Mode Name	SM	NPM	NMM	SRPM	SHM	TMM	VCM
Command Name ¹							
C.LLTD	X	X	X	X	X		
C.LLTE						X	X
C.HLTD	X	X	X	X	X	X	
C.RWTZ	X			X	X	X	X
C.MGTZ	X		X	X	X	X	X
C.GHR	X					X	X
C.FWRWS	X	X	X	X			
C.FWTHR					X	X	X

¹Command Names:

- C.LLTD Command low-level thrusters disabled
- C.LLTE Command low-level thrusters enabled
- C.HLTD Command high-level thrusters disabled
- C.RWTZ Command zero reaction wheel torque commands to the reaction wheels
- C.MGTZ Command zero magnetic torquer commands to the magnetic torquers
- C.GHR Command the gyros to the high rate
- C.FWRWS Command to inform the firmware that the reaction wheels are the actuators in use (to ACE)
- C.FWTHR Command to inform the firmware that the thrusters are the actuators in use (to ACE)

NOTE: The gyros must be commanded to the low rate by the ground when the ground commands the spacecraft to the normal pointing mode.

The mode control for the OBC-controlled modes is now finished (i.e., SM, NPM, NMM, TMM, VCM), and the mode transitions are thus complete. The mode transition control flag is reset to 0:

```
If MODEX = (1, 2, 3, 6, or 7)
    ACADMD = MODEX
    MODEX = 0
Endif
```

On transition to the backup modes SRPM and SHM, additional processing is needed. If ACAD hardware reconfiguration has been requested (RECREQ = 1) from SAFECON, the appropriate command must be issued. In the SRPM, the Sun acquisition test counters must be reset. In the SHM, a command must be issued to switch to the B-side low-level thrusters.

```
If MODEX = (4 or 5)
    If RECREQ = 1
        Use B-side electronics
        If MODEX = 5
            Use B-side low-level thrusters
        Endif
        SYNCK = SYNCTM/0.256
        RECREQ = 0
    Else (If RECREQ < > 1)
        SYNCK = 0
    Endif
```

The Sun acquisition test counters are set to 0 if the spacecraft is transitioning to the SRPM (i.e., MODEX = 4).

```
If MODEX = 4
    AQTAK = 0
    AQTBK = 0
Endif
If MODEX = (4 or 5) Then
    ACADMD = MODEX
    MODEXIN = 0
Endif
```

Safe Contingency Mode Transition Processing--This processing is done only if MODEX = 4 or 5. The OBC mode flag ACADMD has previously been set to the new mode. Therefore, the OBC will be operating in mode 4 or 5 while the following processing is taking place.

Mode transition to the SRPM and SHM requires a series of commands to the ACAD firmware to reset the gyro channel selection flags and to identify the axes that must use derived rate. On entry to the SRPM, if a reaction wheel failure has been indicated (FAILW > 0), another series of firmware commands is needed to reset the failed wheel indicator and the reaction wheel transformation matrix. Normally, the ACAD electronics will have been reconfigured, and these commands must be delayed until the ACAD electronics is resynchronized.

```
C.FWGYR = 0
If SYNCK > 0 (ACAD reconfiguration in progress)
    Then SYNCK = SYNCK - 1
```

If the reconfiguration is finished, the ACAD firmware gyro channel selection and derived rate axis flag indicator, C.FWGYR, is set to 1, and the reaction wheel reconfiguration

flags are set (provided a wheel has failed and the spacecraft is transitioning to the SRPM).

```
C.FWGYR = 0
If SYNCK ≤ 0 Then
    C.FWGYR = 1 (reset the ACAD firmware gyro channel selection and derived rate axis flags)
    C.FWRWX = 0 (do not reset reaction wheel matrix)
    If MODEX = 4 Then
        If FAILW > 0 Then
            C.FWRWX = 1 (reset reaction wheel matrix)
            Do for i = 1 to 4
                If FAILW < > i Then
                    C.RWONi = 1 (reaction wheel i power on)
                Endif
            Enddo
            C.FWSRP = 1 (initiate operation in SRPM)
        Endif
    If MODEX = 5
        C.FWSH = 1 (initiate operation in SHM)
    Endif
Endif
```

This logic uses the value of the OBC failed wheel indicator, FAILW, as set by the ground or as reset by SAFECON, to set the corresponding flag in the firmware. FAILW is passed to the ACE.

Computing the Valid Gyro Flags--The valid gyro flags that are sent to the firmware are set here provided the reset flag, C.FWGYR, is set. The values come from the OBC gyro

flag array, IGY_i , and are assigned to the firmware array, IGF_i .

```
If C.FWGYR = 1
Do for i = 1 to 3
  If  $IGY_i = -1$  Then
     $IGF(i) = 1$  (derived rate required about axis i)
     $IGF(i + 3) = 0$ 
  Else
     $IGF(i) = 0$  (good gyro data about axis i)
     $IGF(i + 3) = IGY_i$ 
Endif
```

Reconfiguring the Reaction Wheel--The 3-by-4 reaction wheel command transformation matrix, TRWD, is set by this function. The values are obtained from the OBC reaction wheel command transformation matrix, DW_{ijk} , which contains five possible scenarios depending on which reaction wheel has failed, if any. The proper matrix is given to TRWD.

```
If C.FWRWX = 1 Then
Do for j = 1 to 4 and k = 1 to 3
   $TRWD(j, k) = DW_{ijk}$ 
Enddo
```

where $i = \text{FAILW}$
 $TRWD(j, k) =$ firmware reaction wheel transformation matrix where $j = 1$ to 4 and $k = 1$ to 3
 $DW_{ijk} =$ OBC flight software reaction wheel transformation matrix where $i = 0$ to 4, $j = 1$ to 4 and $k = 1$ to 3

Sun Acquisition Test in SRPM--In the Sun reference pointing mode, a test for Sun acquisition is performed. If the Sun is not acquired before the time limit is exceeded, transition to safe hold mode is requested.

C4

```

If ACADMD = 4 and MODEX = 0 (transition to SRPM complete)
  Then if SUNPRS = 1
    Then AQTAK = AQTAK + 1
      If AQTAK > AQTAL
        Then AQTBK = 0
      Else AQTAK = 0
        AQTBK = AQTBK + 1
        If AQTBK > AQTBL
          Then set the MODECON request flag ACADMDA = 5
            to transition to SHM RECREQ = 1

```

Standby Mode Timer--At entry to SM, a timer begins incrementing. If the timer exceeds the specified time limit, transition to SRPM is requested:

```

If ACADMD = 1 and MODEX = 0 (transition to SM complete)
  Then
    SBTIMEK = SBTIMEK + 1
  Endif
  If SBTIMEK > SBTIMEL
    Then
      Set the MODECON request flag ACADMDA = 4 to request
      transition to SRPM
    Endif

```

- Output--MODECON produces the following:

<u>Name</u>	<u>Description</u>
ACADMD	ACAD mode status flag data base flag (to ATTERR, MODECON, ANTCON)
ACADMDA	Mode transition request flag from MODECON (to MODECON)
ACADMDG	Mode transition request flag from ground (to MODECON)
ACADMDS	Mode transition request flag from SAFECON (to MODECON)

<u>Name</u>	<u>Description</u>
ACADMDT	Mode transition request flag from HLTCON (to MODECON)
ACSINT	ACS sample period DBC (to GYRODAT, EXEC, SAFECON, HLTCON, WHECON, MODECON)
AECT	Attitude error comparison threshold (to SAFECON)
ANTINIT	ANTCON initialization flag (to ANTCON)
AQTAK	Sun acquisition time counter A (to MODECON)
AQTBK	Sun acquisition time counter B (to MODECON)
ARCT	Attitude rate comparison threshold (to SAFECON)
C.FWRWS	Command to inform firmware that reaction wheels are the actuators in use (to ACE, Truth Model)
C.FWSH	Commands to ACAD firmware to initiate backup SHM (to ACE)
C.FWSRP	Commands to ACAD firmware to initiate backup SRPM (to ACE)
C.FWTHR	Command to inform firmware that thrusters are the actuators in use (to ACE, Truth Model)
C.GHR	Command to the IRU to select gyro high-rate mode (to Truth Model)
C.HLTD	Commands to the propulsion electronics (PE) to disable all high-level thrusters (to Truth Model)
C.LLTD	Commands to the PE to disable all low-level thrusters (to Truth Model)
C.LLTDP	Command to the PE to disable low-level pitch thruster firing (to Truth Model)
C.LLTDR	Command to the PE to disable low-level roll thruster firing (to Truth Model)
C.LLTE	Commands to the PE to enable all low-level thrusters (to Truth Model)
C.MGTZ	Zero magnetic torquer commands to the torquer drive electronics (TDE) (to Truth Model)
C.RWON _i	Reaction wheel i power up command where i = 1, 2, 3, or 4 (to Truth Model)
C.RWTZ	Zero reaction wheel torque commands to the RWEA (to Truth Model)
CMDANT	Enablement flag for antenna commands (to ANTCON)
CMDBKT	Enablement flag for break track commands (to ATTEST)

<u>Name</u>	<u>Description</u>
CMDMAG	Enablement flag for magnetic torquer commands (to MAGCON)
CMDWHE	Enablement flag for reaction wheel commands (to WHECON)
FLTINIT	ATTEST filter initialization flag (to ATTEST)
GCCT	Gyro channel comparison threshold (to SAFECON)
GYRINIT	GYRODAT initialization flag (to GYRODAT)
HLTINIT	Velocity control initialization flag (to HLTCON)
LCA _i	Low-level thruster rate filter constant A where i = 1, 2, 3 (to LLTCON)
LCB _i	Low-level thruster rate filter constant B where i = 1, 2, 3 (to LLTCON)
LCC _i	Low-level thruster rate filter constant C where i = 1, 2, 3 (to LLTCON)
LHY _i	Low-level thruster hysteresis constants where i = 1, 2, 3 (to LLTCON)
LIMKEY	Keyhole region limit for current mode (to ANTCON)
LKP _i	Low-level thruster position gains where i = 1, 2, 3 (to LLTCON)
LKR _i	Low-level thruster rate gains where i = 1, 2, 3 (to LLTCON)
LLF _i	Low-level thruster axis controlled flag where i = 1, 2, 3 (to LLTCON)
LLTINIT	LLTCON initialization flag (to LLTCON)
LPL _i	Position error limits where i = 1 to 3 (to LLTCON)
MAGINIT	MAGCON initialization flag (to MAGCON)
MODEX	Mode transition control flag (to MODECON)
MOMINIT	Momentum comparison initialization flag (to MOMCOM)
RECREQ	Request for reconfiguration of ACAD hardware (to MODECON)
SBTIMEK	Standby time counter (to MODECON)
SCMINIT	SAFECON initialization flag (to SAFECON)
SGCL	Single-axis gyro channel miscompare limit (to SAFECON)
STRAVG _i	Averaging function execution flag for star tracker i where i = 1, 2 (to STARAVG)

<u>Name</u>	<u>Description</u>
SYNCK	ACAD hardware resynchronization counter (to MODECON)
TGCL	Two-axis gyro channel miscompare limit (to SAFECON)
THGCL	Three-axis gyro channel miscompare limit (to SAFECON)
WHEINIT	WHECON error initialization flag (to WHECON)

5.2.4.7 Safe Contingency Mode Function

The safe contingency mode function (SAFECON) provides a failure detection and response capability for the EPDS power and ACADS. If out-of-tolerance conditions are detected, commands are issued to spacecraft subsystems, and requests are made to the mode control function (MODECON) for transition to safe contingency mode.

- Input--SAFECON uses the following input:

<u>Name</u>	<u>Description</u>
ACADM	Current ACAD mode (from MODECON)
ACSINT	ACS sample interval (from MODECON)
AECT	Attitude error comparison threshold (from MODECON)
AERK	Excessive attitude error or rate counter (from SAFECON)
AERL	Successive attitude error or rate limit (DBC)
ARCT	Attitude rate comparison threshold (from MODECON)
E_i	Attitude errors (commanded minus actual) where $i = 1, 2, 3$ (from ATERR)
ENATTER	Enablement flag for attitude error and rate comparison; = 1, enabled (from DBC/GND, SAFECON)
ENGYBAK	Enablement flag for gyro channel comparison in backup mode; = 1 enabled (from DBC/GND, SAFECON)
ENGYCOMP	Enablement flag for gyro channel comparison; = 1, enabled (from DBC/GND, SAFECON)
FAILW	Failed reaction wheel indicator (data base reset by MODECON)
GCCT	Gyro channel comparison threshold (from MODECON)

<u>Name</u>	<u>Description</u>
GRB _{ij}	Nominal gyro rate biases for axis i (where i = 1, 2, 3), channel j (where j = 1, 2) (DBC's)
GSFH _{ij}	Gyro high-rate mode scale factors for axis i (where i = 1, 2, 3), channel j (where j = 1, 2) (DBC's)
GSFL _{ij}	Gyro low rate mode scale factors for axis i (where i = 1, 2, 3), channel j (where j = 1, 2) (DBC's)
PTNG _{ij}	Past value of OBC gyro counters for axis i (where i = 1, 2, 3), channel j (where j = 1, 2) (from SAFECON)
REGERR	Gyro register error flag (from GYRODAT)
REGK	Gyro register error counter (from SAFECON)
REGREQ	Hardware reconfiguration request flag (from MODECON)
REGREQP	Flag indicating that firmware reconfiguration has occurred
SCMINIT	SAFECON initialization flag (from MODECON)
SENSTA	Gyro rate status (from GYRODAT)
SGCK _i	Single-axis gyro miscompare counters for axis; where i = 1, 2, 3 (from SAFECON)
SGCL	Successive single-axis gyro miscompare limit (from MODECON)
SIND _i	Indicator that a single-axis failure has taken place on axis i where i = 1 to 3 (from SAFECON)
TGCK _i	Two-axis gyro miscompare counters for axis i where i = 1, 2, 3 (from SAFECON)
TGCL	Successive two-axis gyro miscompare limit (from MODECON)
THGCK	Three-axis miscompare counter
THGCL	Successive three-axis miscompare limit (from MODECON)
THIND	Indicator that a three-axis gyro failure has taken place (from SAFECON)
TIND _i	Indicator that a two-axis miscompare has taken place, with i being the good axis, where i = 1 to 3 (from SAFECON)
TNG _{ij}	Cumulative OBC gyro counters for axis i (where i = 1, 2, 3), channel j (where j = 1, 2) (from GYRODAT)
WOK _i	Reaction wheel health flags where i = 1, 2, 3, 4 (from WHECON)

<u>Name</u>	<u>Description</u>
W_i	Spacecraft body rates where $i = 1, 2, 3$ (from GYRODAT)
$ZWHOFF_i$	Flag that indicates if wheel i is good ($= 1$) or bad ($= 0$) (from DBC/GND) where $i = 1$ to 4

• Processing--ACAD monitoring is performed once every ACS sample period (ACSINT) in ACAD modes NPM, NMM, TMM, and VCM following GYRODAT. The SAFECON execution interval and ACAD modes in which it executes are shown in Table 5-6.

If transition to a safe contingency mode (SRPM or SHM) is needed, the ACAD mode control function (MODECON) is requested to implement the transition. MODECON must know which gyros to use for which axes and which axis must use the derived rate. This information is specified by resetting the channel select flags, IGY_i . In this respect, the requirement to use the derived rate (or in roll, to use magnetometer control) is specified by setting the channel select flag $IGY_i = -1$.

In responding to attitude and rate errors, SAFECON also controls which reaction wheels to use. This is specified by resetting the failed wheel indicator, FAILW.

If reconfiguration of the ACAD hardware, nominally switching to B-side electronics, is required, the RECREQ request flag is set for implementation by MODECON.

Each of the following limit-check functions and the associated commanding and mode transition responses is separately enabled and disabled by ground command.

Initialization--If initialization is requested (SCMINIT = 1), the gyro register error flags, the ACAD out-of-limit counters, and the powered-off wheel status flag are reset:

```

Do for i = 1 to 3
    SGCKi = 0
    TGCKi = 0

```

THGCK = 0
AERK = 0
ZWHOFF_i = 0

Also, the past values of the OBC gyro counters are set:

Do for i = 1 to 3 and j = 1 to 2

PTNG_{ij} = TNG_{ij} - cumulative OBC gyro miscompare in
last value of OBC gyro counter

The initialization flag SCMINIT is then reset to 0. Upon flight software initialization, attitude error and rate comparisons are enabled:

ENATTER = 1
ENGYCOMP = 1
ENGYBAK = 1

Gyro Channel Comparison

Gyro Channel Differences--The gyro channel compare function is performed by first differencing the current and past values of the OBC gyro counters and then resetting the past values:

Do for i = 1 to 3 and j = 1 to 2 (all channels)

GI_{ij} = TNG_{ij} - PTNG_{ij}
PTNG_{ij} = TNG_{ij}

GI_{ij} is the angular increment in counts, and PTNG_{ij} is the past value of the OBC gyro counter.

The gyro angular increments are then corrected for scale factor and nominal gyro rate bias. The rate status for each gyro (not axis) is given by the corresponding bit of the rate status word, SENSTA. Although on GRO all gyros are set to the same rate, the possibility of different rates on different gyros is provided for.

Do for i = 1 to 3 and j = 1 to 2 (all channels)

If BIT_k = 0

Then $GI_{ij} = GI_{ij} * GSFH_{ij} - GRB_{ij} * ACSINT$
 Else $GI_{ij} = GI_{ij} * GSFL_{ij} - GRB_{ij} * ACSINT$

BIT_k is the kth bit of SENSTA, k is determined as a function of i,j from Table 5-9, and GRB is the nominal gyro rate bias. GI_{ij} is the gyro increment, in radians corrected for nominal gyro rate bias.

Table 5-9. Gyro Channel Definitions

<u>Axis (i)</u>		<u>Channel (j)</u>		<u>Gyro (k)</u>
Roll (X)	1	A	1	1
	1	B	2	3
Pitch (Y)	2	A	1	2
	2	B	2	3
Yaw (Z)	3	A	1	1
	3	B	2	2

Next, the difference between the gyro increments on each axis is compared with the gyro comparison threshold. If the difference exceeds the threshold, the axis miscompare flag, GMC_i , is set:

Do for i = 1 to 3 (all axes)

$DI_i = GI_{i1} - GI_{i2}$

If $ABS(DI_i) > GCCT$

Then $GMC_i = 1$

Else $GMC_i = 0$

The axis miscompare flags, GMC_i , are temporary variables. $GMC_i = 1$ indicates that a miscompare has occurred on axis i. DI_i is not a true rate because GI_{ij} is not corrected for the sample period value, ACSINT. This must be considered in selecting values for GCCT.

Miscompare Type Determination--Next, it is determined whether a one-, two-, or three-axis miscompare has occurred, and successive occurrences are counted:

```

If GMC1 + GMC2 + GMC3 = 2 (two-axis miscompare)
Then THGC = 0
  Do for i = 1 to 3
    SGCKi = 0
    If GMCi = 0
      Then TGCKi = TGCKi + 1
    Else TGCKi = 0
  Else if GMC1 + GMC2 + GMC3 = 3 (three-axis miscompare)
Then THGCK = THGCK + 1
  Do for i = 1 to 3
    SGCKi = 0
    TGCKi = 0
  Else If GMC1 + GMC2 + GMC3 = 1 (single-axis miscompare)
Then THGCK = 0
  Do for i = 1 to 3
    TGCKi = 0
    If GMCi = 1
      Then SGCKi = SGCKi + 1
    Else SGCKi = 0
  Else THGCK = 0 TGCKi = SGCKi = 0 for i = 1 to 3 (no
    miscompare)

```

TGCK_i counts two-axis miscompares in which axis "i" is good, and SGCK_i counts single-axis miscompares in which axis "i" is bad. The miscompare counters are reset to zero if sequential miscompares of the same type (single or two axis) on the same axis do not occur.

Gyro Channel Definition--To implement the failure response for different types of miscompares, the relationship between axes (i), channels (j), and gyros (k) is required. This is shown in Table 5-9.

Failure Declaration and Response--The last steps are to determine if a failure should be declared. If the sequential miscompare counters exceed their limits, transition to SRPM or SHM is requested.

For two-axis miscompares:

Do for $i = 1$ to 3

If $TGCK_i > TGCL$ and $ENGYCOMP = 1$

Then the gyro channel select flags, IGY_i , are reset to select the good gyro channels, as shown in Table 5-10.

Table 5-10. Gyro Selection for Two-Axis Miscompares

Good Axis <u>(i)</u>	Bad Gyro	Gyro to Use for			Channel Selection		
		Roll	Pitch	Yaw	<u>IGY1</u>	<u>IGY2</u>	<u>IGY3</u>
1	2	3	3	1	1	1	0
2	1	3	2	2	1	0	1
3	3	1	2	1	0	0	0

Set the ACAD hardware reconfiguration request flags $RECREQ = 1$, $RECREQP = 1$

Set the two-axis miscompare flag to indicate that a failure has occurred and the good axis is i ; $TIND_i = 1$

If $ACADMD = 6$ or 7 (thruster modes)

Then set mode transition request flag $ACADMDS = 5$ to transition to SHM

If $ACADMD = 2$ or 3 (wheel modes)

Then set mode transition request flag $ACADMDS = 4$ to transition to SRPM

The good gyro channels for two-axis miscompares are selected by noting that "i" indicates the good axis and the other two axes are bad. Therefore, the presumption is that the gyro

common to the other two axes is the bad one and should not be used. This logic is stated in Table 5-10.

The channel selection in this case is somewhat arbitrary because four good channels are available and only three are required.

For single-axis miscompares:

```

Do for i = 1 to 3
  If  $SGCK_i > SGCL$  and  $ENGYCOMP = 1$ 
    Then reset gyro channel select flags,  $IGY_i$ , to select
      the good gyro and derived rate on the third axis as
      shown in Table 5-11
    Set ACAD hardware reconfiguration request flags  $RECREQ = 1$ ,  $RECREQP = 1$ 

    Set the single-axis failure flag to indicate that a failure
    on axis i has taken place;  $SIND_i = 1$ 

    If  $ACADMD = 6$  or  $7$  (thruster modes)
      Then set mode transition request flag  $ACADMDS = 5$  to
      transition to SHM

    If  $ACADMD = 2$  or  $3$  (wheel modes)
      Then set mode transition request flag  $ACADMDS = 4$  to
      transition to SRPM
  
```

Table 5-11. Gyro Selection for Single-Axis Miscompares

Bad Axis (i)	Bad Gyro	Gyro to Use for			Channel Selection		
		Roll	Pitch	Yaw	IGY1	IGY2	IGY3
1	1,3		2	2	-1	0	1
2	2,3	1		1	0	-1	0
3	1,2	3	3		1	1	-1

For single-axis mismatches, "i" indicates the bad axis. This means that one or both channels used for axis i are bad, that is, the derived rate must be used on that axis (or in the case of the roll axis, magnetometer control). It also implies (but does not prove) that the two gyros on axis i may be bad and should not be used. Table 5-11 shows this logic, where $IGY_i = -1$ indicates use of the derived rate.

The channel selection in this case is unique because of the assumption that both gyros on axis i are bad, leaving only one gyro for the other two axes.

For three-axis mismatches:

If $THGCK > THGCL$ and $ENGYCOMP = 1$

Then

Set gyro channel select flags $IGY_i = -1$, ($i = 1, 3$) to select derived rate on all axes

Set hardware reconfiguration request flags $RECREQ = 1$, $RECREQP = 1$

Set three-axis mismatch indicator $THIND = 1$

Disable gyro checking in backup mode by setting $ENGYBAK = 0$

Request transition to SHM by setting $ACADMDS = 5$

The following is done when the spacecraft is in a backup mode (SHM or SRPM).

For two-axis mismatches: When a two-axis mismatch occurs in the SHM or SRPM, the gyros will be considered to be "degraded" (1) if a prior two-axis mismatch has occurred with a different good axis or (2) if a prior single-axis mismatch has occurred on the axis that is now considered to be good. The response in the SRPM is to wait until Sun presence time-out, then go to SHM using three-axis derived

rate. The response in the SHM is to change gyros immediately to three-axis derived rate.

Do for i = 1 to 3

If $TGCK_i > TGCL$ and $ENGYBAK = 1$

Then

Do for k = 1 to 3

If $(TIND_k \neq 0$ and $i \neq k)$ or $SIND_i = 1$ or

$SIND_k = TIND_k$ for all k

Then

Set two-axis miscompare indicator, $TIND_i = 1$

Reset gyro channel select flags

$IGY_i = -1$ ($i = 1, 3$) to select derived rate on all axes

Disable gyro checks in backup modes

$ENGYBAK = 0$

If $ACADM = 4$

Then

Request hardware reconfiguration

$RECREQ = RECREQP = 1$

Else if $ACADM = 5$

Then

Request hardware configuration if not already done

If $RECREQP = 0$

Then $RECREQP = RECREQ = 1$

Request SHM

$ACADMDS = 5$

Endif

Endif

Enddo

Endif

Enddo

For single-axis mismatches: When a single-axis mismatch occurs in a backup mode, the gyros will be considered to be degraded (1) if a prior single-axis mismatch has occurred on a different axis, (2) if any prior two-axis mismatch has occurred, or (3) if no prior mismatch has occurred.

```
Do for i = 1 to 3
  If SGCKi > SGCL and ENGYBAK = 1
  Then
    If SINDk <> 0 and k<>i or
      TINDk <> 0 or
      (SINDk = TINDk = 0 for all k)
    Then
      Set single-axis mismatch indicator, SINDi
      = 1
      Reset gyro channel select flags IGYi = -1,
      (i = 1, 3) to select derived rate on all axes
      Disable gyro checks in backup modes
      ENGYBAK = 0
      If ACADMD = 4
      Then
        Request hardware reconfiguration
        RECREQ = RECREQP = 1
      Else if ACADMD = 5
      Then
        Request hardware reconfiguration if not
        already done
        If RECREQP = 0
        Then RECREQ = RECREQP = 1
        Request SHM
        ACADMDS = 5
      Endif
    Endif
  Endif
Endif
Enddo
```

For three-axis miscompares: Any new three-axis miscompare in a backup mode is considered to be a gyro degradation.

```
If   THGCK > THGCL and
     ENGYBAK = 1 and
     THIND = 0
Then
     Set three-axis miscompare indicator
     THIND = 1
     Reset gyro channel select flags  $IGY_i = -1$ , ( $i = 1, 3$ ) to select derived rate on all axes
     Disable gyro checks in backup modes
     ENGYBAK = 0
     If ACADMD = 4
     Then
         Request hardware reconfiguration
         RECREQ = RECREQP = 1
     Else if ACADMD = 5
     Then
         Request hardware configuration if not already done
         If RECREQP = 0
         Then RECREQP = RECREQ = 1
         Request SHM
         ACADMDS = 5
     Endif
Endif
```

Attitude Error and Attitude Rate Comparison--If any attitude error (E_i) or body rate (W_i) exceeds its threshold consecutively for more intervals than the specified limit (AERL), transition to SRPM or SHM is requested. The thresholds (AECT and ARCT) are set by MODECON before attitude maneuvers to preclude undesired transitions during maneuver transients. This is necessary because the initial attitude error during

a maneuver will be equal to the commanded attitude change, which may be as large as 180 degrees.

If mode transition is to be requested, a check of reaction wheel health is made before requesting mode transition. If ACAD is not in a thruster mode and at least three wheels are good, transition to SRPM is requested. Otherwise, transition to SHM is requested.

```
    If ABS(E1) > AECT or ABS(W1) > ARCT or
       ABS(E2) > AECT or ABS(W2) > ARCT or
       ABS(E3) > AECT or ABS(W3) > ARCT
```

```
    Then AERK = AERK + 1
```

```
    Else AERK = 0
```

```
    If AERK > AERL and ENATTER = 1
```

```
    Then set ACAD hardware reconfiguration request flag
```

```
        RECREQ = 1, RECREQP = 1
```

```
    If ACADM = 2 or 3 (wheel modes)
```

```
    Then
```

```
        If ZWHOFF1 + ZWHOFF2 + ZWHOFF3 + ZWHOFF4 < 3
```

```
            Request SHM
```

```
            ACADMDS = 5
```

```
        Else
```

```
            Request SRPM
```

```
            ACADMDS = 4
```

```
    Else
```

```
        Request SHM
```

```
        ACADMDS = 5
```

This logic uses a single counter (AERK) for all errors, and sequential errors on different axes are all counted. The failed wheel indicator, FAILW, may be reset based on the results of the reaction wheel tests conducted in SAFECON if previously it was set to zero.

Reaction Wheel Correction--Responses to reaction wheel failures are based on information regarding reaction wheel failures reported by WHECON (WOK_i) and information regarding reaction wheel health ($ZWHOFF_i$). All failures should be flagged; failed wheels should be turned off; and, if possible, a good wheel should be turned on if it has been turned off by the ground. To turn off a good wheel from the ground, commands must be sent to set $FAILW = 1$ and $ZWHOFF_i = 1$ for wheel i .

To operate with fewer than three wheels, the ground must load an appropriate steering matrix (DW_{ijk}), set $ZWHOFF_i = 0$ for each of the failed wheels, and set $FAILW =$ the highest numbered failed wheel ($i = 1,4$).

If $ZWHOFF_i \neq WOK_i$ for any i and

ACADM = 2 or 3 and

AMCONTROL = OBC

Then

$WHEOK = WOK_1 + WOK_2 + WOK_3 + WOK_4$

If $WHEOK = 3$

Then

If $FAILW = 0$

Then

Do for $i = 1,4$

If $WOK_i = 0$, then $FAILW = 1$

Send command $C.RWOFF_i$ to turn off wheel i

Set $ZWHOFF_i = 0$

Else if $ZWHOFF(FAILW) = 1$

Send command $C.RWON(FAILW)$ to turn on wheel $FAILW$

Set $WHEINIT = 1$ to reset failure detection logic in WHECON

Do for $i = 1,4$

If $WOF_i = 0$, then $FAILW = 1$

Send command $C.RWOFF_i$ to turn off wheel i

Set $ZWHOFF_i = 0$

```

Else
  Do for i = 1,4
    If WOKi = 0, then FAILW = i
    Send command C.RWOFFi to turn off wheel i
    Set ZWHOFFi = 0
  Else if WHEOK < 3
    Do for i = 1,4
      If WOKi = 0, then FAILW = i
      Send command C.RWOFFi to turn off wheel i
      Set ZWHOFFi = 0

```

- Output--SAFECON produces the following output:

<u>Name</u>	<u>Description</u>
ACADMDS	Mode transition request (to MODECON)
AERK	Attitude error or rate counter (to SAFECON)
ENATTER	Enablement flag for attitude error and rate comparison; = 1, enabled (to SAFECON)
ENGYBAK	Enablement flag for gyro channel comparison in backup mode; = 1 enabled (from DBC/GND, SAFECON)
ENGYCOMP	Enablement flag for gyro channel comparison; = 1, enabled (to SAFECON)
FAILW	Failed reaction wheel indicator (to WHECON, MAGCON, MODECON, and SAFECON)
IGY _i	Gyro channel selection flags for axis i where i = 1, 2, 3 (to GYRODAT and MODECON)
PTNG _{ij}	Past value of cumulative gyro counters for axis i (where i = 1, 2, 3), channel j (where j = 1, 2) (to SAFECON)
RECREQ	ACAD reconfiguration request (to MODECON)
REGERR	Gyro register error flag (to SAFECON, GYRODAT)
REGK	Gyro register error counter (to SAFECON)
REGREQP	Flag indicating that firmware reconfiguration has occurred
SGCK _i	Single-axis miscompare counter for axis i where i = 1, 2, 3 (to SAFECON)
SIND _i	Indication that a single-axis failure has taken place on axis where i = 1 - 3 (from SAFECON)
TGCK _i	Two-axis miscompare counter for axis i where i = 1, 2, 3 (to SAFECON)

Name	Description
THGCK	Three-axis miscompare counter
THGCL	Successive three-axis miscompare limit (from MODECON)
THIND	Indicates that a three-axis gyro failure has taken place (from SAFECON)
TIND _i	Two-axis miscompare file where i = 1 to 3 (from SAFECON)
ZWHOFF _i	Flag that indicates if wheel i is good (=1) or bad (=0) (to SAFECON)

5.2.5 ANTENNA POINTING COMMAND PROCESSING

The antenna pointing command processing function (ANTCON) uses current antenna gibal angles and ephemeris and attitude data to compute desired gibal angles and gibal slew commands as shown below.

- Input--ANTCON uses the following input:

Name	Description
ACADMD	ACAD status flag (from MODECON)
ALIMIT1	Antenna azimuth limit 1 (DBC)
ALIMIT2	Antenna azimuth limit 2 (DBC)
ANT _{ij}	Elements of 3-by-3 coordinate transformation from ACAD body axes where j = X, Y, Z to antenna body axes where i = X, Y, Z (DBC)
AZBIAS	Azimuth resolver output bias (DBC)
AZRES	Current gibal azimuth angle resolver output (from Truth Model)
BIAZ	Data base flag (set by ground) used to prevent azimuth gibal limiting (= 0, normal; = 1, 180-deg flip) (DBC)
CMDANT	Enable flag for antenna commanding; = 0, no; = 1, yes (from DBC/GROUND, MODECON)
ELBIAS	Elevation resolver output bias (DBC)
ELIMIT1	Antenna elevation limit 1 (DBCs)
ELIMIT2	Antenna elevation limit 2 (DBCs)
ELRES	Current gibal elevation angle resolver output (output of gibal drive electronics every 128 msec)

<u>Name</u>	<u>Description</u>
EPA _i	Updated Euler parameters that specify the ACAD body axis orientation with respect to the ECI frame where i = 1 to 4 (parameters from KININT, DBC/GROUND)
GROP _i	GRO position in ECI coordinates where i = X, Y, Z (parameters from EPHEM)
ITDRS	Data base flag set by ground command to indicate which TDRS to use (= 0, TDRS-East; = 1, TDRS-West) (DBC)
KI	Integral gain (DBC)
KP	Proportional gain (DBC)
LIM1	Integral error limit (DBC)
LIM2	Linear proportional error limit (DBC)
LIM3	Nonlinear proportional gain slope factor (DBC)
LIM4	Maximum slew rate limit (DBC)
LIM5	Slew acceleration limit (DBC)
LIM6	Stepper motor scale factor (DBC)
LIMKEY	Keyhole region (from MODECON)
LIMKEY2	Keyhole region 2 (DBC)
ANTINIT	Initialization flag for ANTCON (= 1, initialize; = 0, do not initialize) (DBC)
RPC	Gimbal resolver scale factor (DBC)
TARATT _i	Euler symmetric parameters that specify commanded ACAD body axes with respect to the ECI frame for i = 1 to 4 (parameters from ATTITUDE ESTIMATION)
TDREP _i	TDRS-East current position in ECI coordinates where i = X, Y, Z (from ephem)
TDRWP _i	TDRS-West current position in ECI coordinates where i = X, Y, Z (from ephem)
TSAN	Sampling period (computation interval) of ANTCON (DBC)

• Processing--Initialization of ANTCON is executed within 128 msec. TDRS selection and gimbal angle generation are executed every 0.512 sec. ANTCON is executed after attitude error and before wheel control.

Initialization--If ANTINIT = 1, the following initialization takes place:

$$\text{OLDAZ} = (\text{AZRES} - \text{AZBIAS}) * \text{RPC}$$

$$\text{OLDEL} = (\text{ELRES} - \text{ELBIAS}) * \text{RPC}$$

$$\text{ELOUT1} = \text{AZOUT1} = \text{ELERR1} = \text{AZERR1} = 0$$

$$\text{IACCEL} = \text{IACCAZ} = \text{ANTINIT} = 0$$

where AZERR1 = old value of azimuth error signal (see below)

AZOUT1 = old value of integration for azimuth (see below)

ELOUT1 = old output of integration for elevation (see below)

ELERR1 = old value of elevation error signal (see below)

IACCAZ = feedback value of acceleration limited steps for azimuth

IACCEL = feedback value of acceleration limited steps for elevation

If ANTINIT = 0, no initialization is performed, and the remainder of ANTCON is executed with the old values of the above parameters.

TDRS Selection--The position data from TDRS-East or TDRS-West are selected as follows:

If ITDRS = 0, then

$$\text{TDREPH}_i = \text{TDREP}_i \quad \text{for } i = X, Y, Z$$

where TDREPH_i is ephemeris position data for selected TDRS.

Otherwise (ITDRS = 1)

$$\text{TDREPH}_i = \text{TDRWP}_i \quad \text{for } i = X, Y, Z$$

Gimbal Angle Generation--The gimbal angles needed for proportional and integral control are computed. The LOS vector components between the GRO and TDRS spacecraft are computed by the following equation:

$$RLOS_i = TDREPH_i - GROP_i \quad \text{for } i = X, Y, Z$$

where $RLOS_i$ is the LOS vector from GRO to TDRS.

The LOS vector is then normalized to a unit vector:

$$RLOS_i = RLOS_i / RLOSM \quad \text{for } i = X, Y, Z$$

where RLOSM is $\left(RLOSX^2 + RLOS Y^2 + RLOS Z^2 \right)^{1/2}$.

Using the commanded Euler symmetric parameters EPA_i , $RLOS_i$ is transformed from the ECI coordinate system to ACAD body coordinates as follows:

$$RLOS B_i = XFORM(\overrightarrow{RLOS}, \overrightarrow{EPA}, -1)$$

where $RLOS B_i$ = normalized LOS vector in spacecraft coordinates for $i = X, Y, Z$

XFORM = defined in Section 5.2.3.4, Attitude Estimation Function, Processing subsection, paragraph entitled Vector Coordinate Transformation

$$\overrightarrow{EPA} = (EPA_1, EPA_2, EPA_3, TARATT_4)$$

and \overrightarrow{RLOS} is defined above.

The $RLOS B$ vector is then converted from spacecraft coordinates to antenna coordinates by the following:

$$RLOSA_X = ANT_{XX} * RLOS B_X + ANT_{XY} * RLOS B_Y + ANT_{XZ} * RLOS B_Z$$

$$RLOSA_Y = ANT_{YX} * RLOSB_X + ANT_{YY} * RLOSB_Y + ANT_{YZ} * RLOSB_Z$$

$$RLOSA_Z = ANT_{ZX} * RLOSB_X + ANT_{ZY} * RLOSB_Y + ANT_{ZZ} * RLOSB_Z$$

where $RLOSA_i$ represents normalized LOS vector components in antenna coordinates for $i = X, Y, Z$.

The desired elevation gimbal angle is now computed as follows:

$$ELANG = ARCCOS(RLOSA3)$$

The antenna elevation angle is then adjusted as a function of BIAZ (bidirectional azimuth external command).

If BIAZ = 1, then

$$ELANG = -ELANG$$

where ELANG is the desired (commanded) elevation angle of the GRO antenna.

In normal pointing mode (ACADMD = 2), if the LOS is in the antenna keyhole region, that is, the elevation angle ELANG is near zero, the coordinate transformation to antenna axes is redone using the commanded attitude, $TARATT_i$, rather than the actual attitude, EPA_i . The keyhole region test is performed as follows:

```

If ACADMD = 2 and |ELANG| < LIMKEY
Then          KEY = 1
Else          KEY = 0
If KEY = 1

```

Then perform antenna transformations over:

$$\text{RLOSB} = \text{XFORM}(\overrightarrow{\text{RLOS}}, \overrightarrow{\text{TARATT}}, -1)$$

$$\text{RLOSA}_x = \text{ANT}_{xx} * \text{RLOSB}_x + \text{ANT}_{xy} * \text{RLOSB}_y + \text{ANT}_{xz} * \text{RLOSB}_z$$

$$\text{RLOSA}_y = \text{ANT}_{yx} * \text{RLOSB}_x + \text{ANT}_{yy} * \text{RLOSB}_y + \text{ANT}_{yz} * \text{RLOSB}_z$$

$$\text{RLOSA}_z = \text{ANT}_{zx} * \text{RLOSB}_x + \text{ANT}_{zy} * \text{RLOSB}_y + \text{ANT}_{zz} * \text{RLOSB}_z$$

where RLOSA_i represents the new LOS vector components in antenna coordinates for $i = X, Y, Z$.

The antenna elevation and azimuth angles are updated as follows:

If $|\text{ELANG}| > \text{ELIMIT2}$,

$\text{ELANG} = \text{OLDEL}$

$\text{AZANG} = \text{OLDAZ}$

where AZANG = desired (commanded) azimuth angle of GRO antenna

OLDAZ = previous cycle's (old) value of AZANG

OLDEL = previous cycle's (old) value of ELANG

If $\text{ABS}(\text{ELANG}) \leq \text{ELIMIT2}$, perform the following:

If $\text{ACADMD} = 2$ (NPM) Then

If $|\text{ELANG}| < \text{LIMKEY2}$

$\text{ELANG} = 0$

$\text{AZANG} = 0$

$\text{FLAG} = 0$

Otherwise, $|\text{ELANG}| \geq \text{LIMKEY2}$

FLAG = 1

Otherwise, ACADMD is not equal to 2 (NPM)

Then

If $\text{ABS}(\text{ELANG}) < \text{LIMKEY}$

ELANG = OLDEL

AZANG = OLDAZ

FLAG = 0

Else FLAG = 1

If FLAG = 1, then perform the following:

If BIAZ = 0

Then $\text{AZANG} = \text{ATAN2}(\text{RLOSA}_Y, -\text{RLOSA}_X)$

If BIAZ = 1

Then $\text{AZANG} = \text{ATAN2}(-\text{RLOSA}_Y, \text{RLOSA}_X)$

where ATAN2 is a two-argument arc tangent defined over the four quadrants with an output between $-\pi$ and π and defined as follows:

$\text{ATAN2}(y/x)$	= $-\pi + \arctan(y/x)$	for $x < 0, y < 0$
	= $-\pi/2$	for $x = 0, y < 0$
	= $\arctan y/x$	for $x > 0$
	= $\pi/2$	for $x = 0, y > 0$
	= $\pi + \arctan(y/x)$	for $x < 0, y \geq 0$
	= undefined	for $x = 0, y = 0$

Arctan (y/x) is a one-argument arc tangent with an output between $\pm\pi/2$.

The difference between the new and old azimuth angles is then computed:

$\text{DAZ} = \text{AZANG} - \text{OLDAZ}$

The azimuth angle is adjusted using this difference:

If $DAZ > \pi$, $AZANG = AZANG - 2\pi$

If $DAZ < -\pi$, $AZANG = AZANG + 2\pi$

Otherwise, there is no change in AZANG.

The azimuth angle is further constrained as follows:

If $AZANG > ALIMIT2$, $AZANG = AZANG - 2\pi$

If $AZANG < -ALIMIT2$, $AZANG = AZANG + 2\pi$

Otherwise, there is no change in AZANG.

If $FLAG = 0$, do not change AZANG.

The present values of AZANG and ELANG are then saved for the next iteration:

$OLDAZ = AZANG$

$OLDEL = ELANG$

The following constraints are then applied to the azimuth angle:

If $AZANG > ALIMIT1$, $AZANG = ALIMIT1$

If $AZANG < -ALIMIT1$, $AZANG = -ALIMIT1$

Otherwise, there is no change in AZANG.

Similar constraints are applied to the elevation angle:

If $ELANG > ELIMIT1$, $ELANG = ELIMIT1$

If $ELANG < -ELIMIT1$, $ELANG = -ELIMIT1$

Otherwise, there is no change in ELANG.

Compensation/Slew Computations--Azimuth and elevation errors are generated by differencing the desired angles calculated

in the previous subsection, Gimbal Angle Generation, with the resolver output:

$$AZERR2 = AZANG - (AZRES - AZBIAS) * RPC$$

$$ELERR2 = ELANG - (ELRES - ELBIAS) * RPC$$

where AZERR2 and ELERR2 are new values of azimuth and elevation errors.

Trapezoidal integration is used as follows:

$$AZOUT2 = AZOUT1 + (AZERR2 + AZERR1) * TSAN * (0.5)$$

$$ELOUT2 = ELOUT1 + (ELERR2 + ELERR1) * TSAN * (0.5)$$

where AZOUT2, ELOUT2 = new output of integration for azimuth and elevation

AZOUT1, ELOUT1 = old output of integration for azimuth and elevation

AZERR1, ELERR1 = old values of azimuth and elevation errors as shown below

This output is then limited as follows:

If AZOUT2 > LIM1, AZOUT2 = LIM1

If AZOUT2 < -LIM1, AZOUT2 = -LIM1

Otherwise, there is no change in AZOUT2.

If ELOUT2 > LIM1, ELOUT2 = LIM1

If ELOUT2 < -LIM1, ELOUT2 = -LIM1

Otherwise, there is no change in ELOUT2.

A nonlinear proportional gain is implemented as follows:

$$AZKP = KP * AZERR2$$

If AZERR2 > LIM2

$$AZKP = LIM2 * KP + (AZERR2 - LIM2) * KP * LIM3$$

If AZERR2 < -LIM2

$$AZKP = -LIM2 * KP + (AZERR2 + LIM2) * KP * LIM3$$

Otherwise, there is no change in AZKP, where AZKP is the nonlinear proportional gain output in azimuth:

$$ELKP = KP * ELERR2$$

If ELERR2 > LIM2,

$$ELKP = LIM2 * KP + (ELERR2 - LIM2) * KP * LIM3$$

If ELERR2 < -LIM2,

$$ELKP = -LIM2 * KP + (ELERR2 + LIM2) * KP * LIM3$$

Otherwise, there is no change in ELKP, where ELKP is the nonlinear proportional gain output in elevation.

The new values of the azimuth and elevation errors and output of integration are saved for the next computation time:

$$AZERR1 = AZERR2$$

$$ELERR1 = ELERR2$$

$$\text{AZOUT1} = \text{AZOUT2}$$

$$\text{ELOUT1} = \text{ELOUT2}$$

The PI control output is computed as

$$\text{AZCON} = \text{AZKP} + \text{AZOUT2} * \text{KI}$$

$$\text{ELCON} = \text{ELKP} + \text{ELOUT2} * \text{KI}$$

where AZCON and ELCON are azimuth and elevation PI control output.

This output is then converted to steps:

$$\text{AZSTP} = \text{LIM6} * \text{AZCON}$$

$$\text{ELSTP} = \text{LIM6} * \text{ELCON}$$

where AZSTP and ELSTP represent azimuth and elevation PI control output in steps.

The azimuth commanded steps are rounded to the nearest integer.

If AZSTP < 0, IX = -1

Otherwise, IX = +1

$$\text{IRTAZ} = \text{IX} * \text{INT}(|\text{AZSTP}| + 0.5)$$

where IX = intermediate sign parameter

IRTAZ = commanded step in azimuth (integer)

INT(a) means take the integer portion of a.

Similarly, the elevation commanded steps are rounded to the nearest integer.

If ELSTP < 0, IX = -1

Otherwise, IX = +1,

$$\text{IRTEL} = \text{IX} * \text{INT}(|\text{ELSTP}| + 0.5)$$

where IRTEL is the commanded step in elevation (integer).

The commanded azimuth step is limited to the stepper motor maximum slew rate as follows:

If IRTAZ > LIM4, IRTAZ = LIM4

If IRTAZ < -LIM4, IRTAZ = -LIM4

Otherwise, there is no change in IRTAZ.

Similarly, the commanded elevation step is limited

If IRTEL > LIM4, IRTEL = LIM4

If IRTEL < -LIM4, IRTEL = -LIM4

Otherwise, there is no change in IRTEL.

The commanded azimuth step is acceleration-limited as follows:

$$\text{IXAZ} = \text{IRTAZ} - \text{IACCAZ}$$

where IXAZ = intermediate value of acceleration-limited azimuth step command

IACCAZ = value of azimuth stepper motor slew command from last cycle (see below)

If IXAZ > LIM5, IXAZ = LIM5

If IXAZ < -LIM5, IXAZ = -LIM5

Otherwise, there is no change in IXAZ.

Similarly, the commanded elevation step is acceleration-limited as follows:

$$\text{IXEL} = \text{IRTEL} - \text{IACCEL}$$

where IXEL = intermediate value of acceleration-limited
elevation step command

IACCEL = value of elevation stepper motor slew command
from last cycle (see below)

If IXEL > LIM5, IXEL = LIM5

If IXEL < -LIM5, IXEL = -LIM5

Otherwise, there is no change in IXEL.

The final step commands to be sent to the stepper motors are
computed as follows:

$$\text{AZSLEW} = \text{IACCAZ} + \text{IXAZ}$$

$$\text{ELSLEW} = \text{IACCEL} + \text{IXEL}$$

where AZSLEW and ELSLEW represent azimuth and elevation slew
command output to the stepper motors.

The azimuth and elevation slew commands, C.ANAZ and C.ANEL,
are formed from AZSLEW and ELSLEW and are output to the
stepper motors each computation cycle.

The above commands are issued before the next computation
interval for WHECON is started.

The commanded step values are saved for the next computation
frame, as follows:

$$\text{IACCAZ} = \text{AZSLEW}$$

$$\text{IACCEL} = \text{ELSLEW}$$

If commanding is prohibited, the commands to the antenna drive are set to zero as follows:

```
If COMDANT ≠ 1
    C.ANAZ = 0
    C.ANEL = 0
Endif
```

- Output--ANTCON produces the following output:

<u>Name</u>	<u>Description</u>
C.ANAZ	Azimuth slew command output to the azimuth stepper motor (to Truth Model)
C.ANEL	Elevation slew command output to the elevation stepper motor (to Truth Model)

SECTION 6 - ATTITUDE CONTROL ELECTRONICS

6.1 SENSOR DATA PROCESSING

6.1.1 GYRO DATA PROCESSING

The gyro data consist of six gyro channels (X,Y,Z-prime and X,Y,Z-backup). The prime and backup channel selection is made in the OBC and given to the ACE in the form of gyro channel flags. In the case of an OBC failure, the ACE has its own defaults for prime and backup channel flags. The gyro data input to this function represent the total motion detected by the gyro. This function is performed every 128 msec. The entry (DBC) beside a variable indicates a data base constant that the ground can update.

6.1.1.1 Input/Output

- From the Truth Model

GYRANG₁--gyro roll axis data, gyro 1 (rad)
GYRANG₂--gyro roll axis data, gyro 3 (rad)
GYRANG₃--gyro pitch axis data, gyro 2 (rad)
GYRANG₄--gyro pitch axis data, gyro 3 (rad)
GYRANG₅--gyro yaw axis data, gyro 1 (rad)
GYRANG₆--gyro yaw axis data, gyro 2 (rad)

- From the OBC (or Ground)

IGF₁ = 0, good roll axis data; = 1, bad roll axis data (DBC)

IGF₂ = 0, good pitch axis data; = 1, bad pitch axis data (DBC)

IGF₃ = 0, good yaw axis data; = 1, bad yaw axis data (DBC)

IGF₄ = 0, use channel 1 data; = 1, use channel 2 (DBC)

IGF₅ = 0, use channel 3 data; = 1, use channel 4 (DBC)

IGF₆ = 0, use channel 5 data; = 1, use channel 6
(DBC)

- From Computational Scheduling

INITT--initialize thruster control flag

INITW--initialize reaction wheel control flag

IAH--attitude hold flag

ISCAN--roll scan flag

- From Attitude Error Computation

IECL--eclipse flag

- To Attitude Error Computation

G_x--gyro roll rate (rad/sec)

G_y--gyro pitch rate (rad/sec)

G_z--gyro yaw rate (rad/sec)

THETA_x--gyro roll position error (rad)

THETA_y--gyro pitch position error (rad)

THETA_z--gyro yaw position error (rad)

- Initialization

REF_i = 0; i=1 to 6 as the default

6.1.1.2 Algorithm

1. Convert radians to counts

Since the gyro data from the Truth Model are in radians, these data first need to be converted to counts:

$$GD_i = (GYRANG_i - B_{gyro}) / A_{gyro} \quad i=1 \text{ to } 6$$

where for i=1 to 6

GD_i = gyro output in counts (i=1 to 6)

GYRANG_i = gyro output from the Truth Model (rad)

B_{gyro} = conversion factor (rad) (DBC)

A_{gyro} = conversion factor (rad/count) (DBC)

2. Process data

Subtract the past value of GD (GDP) from the current value and store the current value of GD into GDP:

$$\begin{aligned} \text{DNG}_i &= \text{GD}_i - \text{GDP}_i \quad i=1 \text{ to } 6 \\ \text{GDP}_i &= \text{GD}_i \end{aligned}$$

Each gyro shall be compensated (CBIAS_i) for drift every CNBIAS_i (i=1 to 6) cycles according to the following:

$$\text{DNGC}_i = \text{DNG}_i + \text{CBIAS}_i \quad i=1 \text{ to } 6$$

Sum the computed differences according to

$$\text{SUM}_i = \text{SUM}_i + \text{DNGC}_i \quad i=1 \text{ to } 6$$

Limit DNGC_i to ± 512 counts:

$$\text{GPER}_i = \text{GREF}_i - \text{SUM}_i \quad i=1 \text{ to } 6$$

Limit GPER_i to $\pm 2^{15}$ counts for i=1 to 6.

where for i=1 to 6:

<u>Parameter</u>	<u>Range</u>
DNGC _i = raw gyro data differences (counts)	± 1350 counts
SUM _i = accumulated count differences (counts)	$\pm 2^{23}$ counts
CBIAS _i = gyro drift estimate (counts) (DBC)	± 50 counts
CNBIAS _i = drift update period (cycles) (DBC)	$\pm 2^7$ cycles
GREF _i = gyro count reference (counts) (DBC)	$\pm 2^{15}$ counts
GPER _i = gyro count accumulated (counts)	$\pm 2^{15}$ counts
GDP _i = past value of GD _i	

3. Select gyro channels to be used

a. Roll

- (1) If the roll data are bad, set the attitude error and rates to 0 and continue with the pitch selection.

If IGF_1 is not equal to 0

$$G_{roll} = 0$$

$$D_{roll} = 0$$

Else

- (2) Using the IGF_i flag, the proper gyro channel data are used.

If $IGF_4 = 0$

$$G_{roll} = GPER_1$$

$$D_{roll} = DNGC_1$$

Else

$$G_{roll} = GPER_2$$

$$D_{roll} = DNGC_2$$

Endif

where G_{roll} = selected roll count sum (counts)

D_{roll} = selected roll count difference (counts)

$IGF_1 = 0$, good roll axis data; $= 1$, bad roll axis data (DBC)

$IGF_4 = 0$, use channel 1 data; $= 1$, use channel 2 data (DBC)

b. Pitch

- (1) If the pitch gyro data are bad, the pitch and rate error are set to 0.

If IGF_2 is not equal to 0

$$G_{pitch} = 0$$

$$D_{pitch} = 0$$

Else

- (2) Using the IGF flag, the proper pitch gyro channel is selected for the attitude and rate error.

```

If IGF5 = 0
    Gpitch = GPER3
    Dpitch = DNGC3
Else
    Gpitch = GPER4
    Cpitch = DNGC4
Endif

```

where G_{pitch} = selected pitch count sum (counts)
D_{pitch} = selected pitch count difference (counts)
IGF₂ = 0, good pitch data; = 1, bad pitch data (DBC)
IGF₅ = 0, use channel 3 data; = 1, use channel 4 data (DBC)

c. Yaw

- (1) If the yaw gyro data are bad, the yaw and rate error are set to 0.

```

If IGF3 is not equal to 0

```

```

    Gyaw = 0
    Dyaw = 0

```

```

Else

```

- (2) The proper yaw gyro channel is selected.

```

If IGF6 = 0

```

```

    Gyaw = GPER5
    Dyaw = DNGC5

```

```

Else

```

```

    Gyaw = GPER6
    Dyaw = DNGC6

```

```

Endif

```

where G_{yaw} = selected yaw count sum (counts)
 D_{yaw} = selected yaw count difference (counts)
 $IGF_3 = 0$, good yaw axis data; = 1, bad yaw axis data (DBC)
 $IGF_6 = 0$, use channel 5 data; = 1, use channel 6 data (DBC)

4. Calculate gyro attitude error estimate

$$\begin{bmatrix} \theta_x \\ \theta_y \\ \theta_z \end{bmatrix} = CGP \cdot \begin{bmatrix} CTG_{11} & CTG_{12} & CTG_{13} \\ CTG_{21} & CTG_{22} & CTG_{23} \\ CTG_{31} & CTG_{32} & CTG_{33} \end{bmatrix} \cdot \begin{bmatrix} G_{roll} \\ G_{pitch} \\ G_{yaw} \end{bmatrix}$$

where CGP = gyro position scale factor (rad/count) (DBC)

CTG_{ij} = gyro coordinate-to-control axis coordinates transformation matrix where $i, j = 1$ to 3 (DBC)

$THETA_x$ = roll error (rad)

$THETA_y$ = pitch error (rad)

$THETA_z$ = yaw error (rad)

5. Calculate gyro rate estimate

$$\begin{bmatrix} XNG_x \\ XNG_y \\ XNG_z \end{bmatrix} = CGR \cdot \begin{bmatrix} CTG_{11} & CTG_{12} & CTG_{13} \\ CTG_{21} & CTG_{22} & CTG_{23} \\ CTG_{31} & CTG_{32} & CTG_{33} \end{bmatrix} \cdot \begin{bmatrix} D_{roll} \\ D_{pitch} \\ D_{yaw} \end{bmatrix}$$

$$G_x = A_{fg} * G_x + B_{fg} * XNG_x$$

$$G_y = A_{fg} * G_y + B_{fg} * XNG_y$$

$$G_z = A_{fg} * G_z + B_{fg} * XNG_z$$

where CGR = gyro rate scale factor (rad/sec/count) (DBC)

XNG_x = unfiltered roll rate (rad/sec)

XNG_y = unfiltered pitch rate (rad/sec)

XNG_z = unfiltered yaw rate (rad/sec)

A_{fg} = rate filter constant (DBC)

B_{fg} = rate filter gain (DBC)

G_x = roll rate (rad/sec)

G_y = pitch rate (rad/sec)
 G_z = yaw rate (rad/sec)

6. Determine gyro count reference (G_{ref})

This section sets the value of G_{ref} in the following conditions:

- Initialization of reaction wheel control
- Initialization of thruster control
- Initialization of a attitude hold mode
- Spacecraft going into shadow (no Sun)
- Spacecraft leaving roll scan mode

The code required to implement the G_{ref} computation is shown below. The determination of SUMU(i) where i=1 to 6 is described on the next page.

```
ISET=0
If (INITW.OR.INITT) = 1
    ISET = 1
Endif
If (OLDIAH = 0.AND.IAH) = 1
    ISET = 1
Endif
If (OLDIECL = 0.AND.IECL = 1.AND.IAH = 0)
    ISET = 1
Endif
If (OLDISCAN = 1.AND.ISCAN = 0)
    ISET = 1
Endif
```

```

OLDIAH = IAH
OLDIECL = IECL
OLDISCAN = ISCAN

If ISET = 1
    Do for i = 1 to 6
        GREF(i) = SUMU(i)
    Enddo
Endif

```

where ISET = flag to set GREF
 OLDIAH = past value of IAH
 OLDISCAN = past value of ISCAN
 OLDIECL = past value of IECL

To determine SUMU, SUM must be converted to a 24-bit binary number. Bits 1 through 3 and 20 through 24 must be set to 0. This is then converted to a decimal number and its value is SUMU(i), where i=1 to 6. The following routine is used to accomplish this task:

```

    Do 2000 i=1,6
        SUMUP=SUM(i)
        Do 1000 j=1,24
            IBINRY(j)=0
            IBIN=SUMUP/2
            RBIN=SUMUP/2
            RDIF=RBIN-REAL(IBIN)
            If(RDIF.NE.0.0) IBINRY(j)=1
            SUMUP=IBIN
        1000 Continue
        SUMU(i)=0
    
```

```

        Do 3000 k=1,16
        SUMU(i)=IBINRY(k+5)*(2(k+4)) + SUMU(i)
3000 Continue
2000 Continue

```

where SUMU = 16 bits from SUM, least significant bit=32
gyro counts

IBINRY = binary representation of SUM

6.1.2 COARSE SUN SENSOR DATA PROCESSING

The CSS data processing function converts raw CSS data from the Truth Model into pitch and yaw angular errors between the spacecraft +X axis and the line of sight to the Sun. This function is performed every 256 msec.

6.1.2.1 Input/Output

- From the Truth Model
 - CSS_i = CSS i output where i=1 to 4 (rad)
- To Attitude Error Computation
 - ECS_y = CSS pitch axis error (rad)
 - ECS_z = CSS yaw axis error (rad)
 - CSS_{prs} = CSS Sun presence flag (1=presence)

6.1.2.2 Algorithm

1. Convert data

Since the CSS data from the Truth Model are in radians, these data first need to be converted to counts:

$$IDCS_i = (CSS_i - BCSS)/ACSS \quad i=1 \text{ to } 4$$

where $IDCS_i$ = CSS i output (counts) where i=1 to 4

BCSS = conversion factor (rad) (DBC)

ACSS = conversion factor (rad/count) (DBC)

Although the conversion from radians to counts and back to radians (step 2.a) seems superfluous, it is specified here

and in the other sensors as well, to model the round-off errors introduced by the ACE hardware and to permit the use of the same conversion coefficients used by the ACE.

2. Calculate pitch and yaw errors

The nomenclature used in the following step, defined in Reference 14, is confusing at best. It is reproduced here as specified to avoid further confusion, but with added comments to clarify its meaning.

a. Conversion of counts to radians

$$SC_1 = IDCS_2 - IDCS_4$$

$$SC_2 = IDCS_1 - IDCS_3$$

Compute yaw component of Sun vector:

$$SC_y = (SC_1 + SC_2) \cdot CSF$$

Compute pitch component of Sun vector:

$$SC_z = (SC_1 - SC_2) \cdot CSF$$

where CSF = CSS scale factor (rad/count) (DBC)

Limit SC_y and SC_z to ± 1.0 radians.

b. Calculation of yaw and pitch errors

$$\text{Yaw error: } ECS_z = SC_y \cdot (1 + (SC_y^2) \cdot (1/6 + (3/40) \cdot SC_y^2))$$

$$\text{Pitch error: } ECS_y = -SC_z \cdot (1 + (SC_z^2) \cdot (1/6 + (3/40) \cdot SC_z^2))$$

c. Sun presence test

$$CSS_{prs} = 1$$

$$CSS_t = IDCS_1 + IDCS_2 + IDCS_3 + IDCS_4$$

$$\text{If } (CSS_t \text{ .lt. } CTHRS) \text{ } CSS_{prs} = 0$$

where CTHRS = CSS Sun presence threshold (counts) (DBC)

CSS_t = sum of CSS output

CSS_{prs} = CSS Sun presence flag (1=presence)

6.1.3 FINE SUN SENSOR DATA PROCESSING

In the backup modes under control of the ACE, the FSS data are used to calculate an angular yaw and pitch error between the spacecraft +X axis and the line of sight to the Sun. FSS 2 is the default head. If a mode changes, the default head is reset to 2. This function is performed every 256 msec.

6.1.3.1 Input/Output

- From the Truth Model
 - FSS_a = raw alpha angle (rad)
 - FSS_b = raw beta angle (rad)
 - FSS_{prs} = FSS Sun presence indicator (1=Sun presence)
- From Ground
 - HN = FSS head number (default is head 2)
- To Attitude Error Computation
 - EFS_y = FSS pitch axis error (rad)
 - EFS_z = FSS yaw axis error (rad)
 - FSS_{prs} = FSS Sun presence indicator
- To Thruster Control
 - FSS_{prs} = FSS Sun presence indicator

6.1.3.2 Algorithm

1. Convert to counts

FSS data for the proper head (HN) are chosen. Since the data from the Truth Model are in radians, these data must be converted to counts from the FSS:

$$N_{ac} = (FSS_a - B_{fss})/A_{fss}$$

$$N_{bc} = (FSS_b - B_{fss})/A_{fss}$$

where N_{ac} = alpha angle (counts)
 N_{bc} = beta angle (counts)

FSS_a = uncalibrated alpha angle from Truth Model
(rad)

FSS_b = uncalibrated beta angle from Truth Model
(rad)

B_{fss} = conversion factor (rad) (DBC)

A_{fss} = conversion factor (rad/count) (DBC)

2. Perform data compensation

The alpha and beta angles in the FSS head coordinates must be computed using the sensor data (N_{ac} and N_{bc}) and compensated for known calibration errors as follows:

a. Conversion to counts

$$X_1 = AF_{11} + AF_{12} \cdot N_{ac}$$

$$Y_1 = AF_{21} + AF_{22} \cdot N_{bc}$$

b. Calibration

$$X_2 = X_1 - (X_1^3)/3$$

$$Y_2 = Y_1 - (Y_1^3)/3$$

$$A = AF_{19} + X_2$$

$$B = AF_{29} + Y_2$$

where AF_{11} = alpha angle conversion factor (rad) (DBC)
 AF_{12} = alpha angle conversion factor (rad/count) (DBC)
 AF_{21} = beta angle conversion factor (rad) (DBC)
 AF_{22} = beta angle conversion factor (rad/count) (DBC)
 AF_{19} = alpha angle calibration factor (rad) (DBC)
 AF_{29} = beta angle calibration factor (rad) (DBC)

c. Pitch and yaw attitude error computation

$$EFS_y = -B + FSYB_2$$

$$EFS_z = A$$

where EFS_y = pitch error (rad)
 EFS_z = yaw error (rad)
 $FSYB_2$ = FSS head 2 pitch bias (rad) (DBC)

6.1.4 REACTION WHEEL MOMENTUM COMPUTATION

The reaction wheel momentum computation function derives the stored reaction wheel angular momentum by processing the data from each of the four tachometers. This function is performed every 256 msec.

6.1.4.1 Input/Output

- From the Truth Model
 WHL_i = cumulative wheel position (rad) where $i=1$ to 4
 $AOWHL_i$ = cumulative wheel position in previous cycle (rad) where $i=1$ to 4
- To Magnetic Control Law
 HW_i = reaction wheel stored angular momentum (ft-lbf-sec) where $i=1$ to 4
- To Reaction Wheel Command Processing
 HW_i = reaction wheel stored angular momentum (ft-lbf-sec) where $i=1$ to 4
- At Initialization
 $WHLP_i = WHL_i$ where $i=1$ to 4

6.1.4.2 Algorithm

1. Convert Truth Model data to counts

$$KWHL_i = (WHL_i - BRW_a) / ARW_a$$
$$KWHLP_i = (WHLP_i - BRW_a) / ARW_a$$

$i=1$ to 4

where BRW_a = conversion factor (rad) (DBC)

ARW_a = conversion factor (rad/count) (DBC)

2. Calculate count difference

The old cumulative count is subtracted from the present cumulative count to find count change:

$$NW_i = KWHL_i - KWHLP_i \quad i=1 \text{ to } 4$$

The new cumulative count is assigned to the past cumulative count.

$$WHLP_i = WHL_i \quad i=1 \text{ to } 4$$

3. Compute stored reaction wheel angular momentum

$$HW_i = CWS_i \cdot NW_i \quad i=1 \text{ to } 4$$

where CWS_i = tachometer momentum conversion factor
(ft-lbf-sec/count) (DBC) $i=1$ to 4

6.1.5 THREE-AXIS MAGNETOMETER DATA PROCESSING

This function receives data from one of two TAMs and calculates the value of the Earth's magnetic field at the location of the TAM compensating for magnetic effects of the spacecraft. This function is performed every 256 msec.

6.1.5.1 Input/Output

- From the Truth Model

TAM_{ij} = TAM data for each magnetometer where
 i =magnetometer (1,2) and $j=1$ to 3 (x,y,z
components of the magnetic field) (tesla)

- To Reaction Wheel Control

BE_y = Earth magnetic field along spacecraft Y-axis
(gauss)

BE_z = Earth magnetic field along spacecraft Z-axis
(gauss)

- To Magnetic Control Law

BE_x = Earth magnetic field along spacecraft X-axis
(gauss)

BE_y = Earth magnetic field along spacecraft Y-axis
(gauss)

BE_z = Earth magnetic field along spacecraft Z-axis
(gauss)

● To Thruster Control

BE_y = Earth magnetic field along spacecraft Y-axis
(gauss)

BE_z = Earth magnetic field along spacecraft Z-axis
(gauss)

6.1.5.2 Algorithm

1. Convert data from Truth Model to counts

$$ACETAM_j = (TAM_{i1} - B_{tam}) / A_{tam} \quad j=1 \text{ to } 3, \quad i=1 \text{ or } 2 \text{ (TAM number)}$$

where $ACETAM_j$ = TAM data (counts) where $j=1$ to 3

B_{tam} = conversion factor (tesla) (DBC)

A_{tam} = conversion factor (tesla/count) (DBC)

2. Convert data to counts

A zero offset, CBO, is subtracted from the input data $ACETAM(j)$, and this difference is multiplied by the appropriate scale factor (CMS):

$$B_j = CMS * (ACETAM_j - CBO)$$

where CMS = magnetometer scale factor (gauss/count) (DBC)

CBO = magnetometer data zero offset (counts)

3. Calibrate data

The magnetic field bias due to nontorquer bar elements of the spacecraft is added to the measured magnetic field:

$$B_{ex} = B_x + CB_{xb}$$

$$B_{ey} = B_y + CB_{yb}$$

$$B_{ez} = B_z + CB_{zb}$$

where CB_{jb} = components of the magnetic field bias vector
where $j=1$ to 3 (gauss) (DBC)

The maximum magnitude of the magnetic field vector is
0.9 gauss.

6.2 ATTITUDE ERROR COMPUTATIONS

The attitude error computation function generates three-axis attitude error signals and three-axis angular rate signals for the reaction wheel control function and thruster control function. The attitude errors and rates can be determined by the gyros, FSS, and CSS depending on the attitude hold flag (IAH) and the eclipse flag (IECL). This function is performed every 256 msec.

6.2.1 INPUT/OUTPUT

- From Computational Scheduling
 - IAH = attitude hold flag
 - CONTROL = control mode flag
 - INIT_t = initialize thruster control flag
 - INIT_w = initialize wheel control flag
- From Gyro Data Processing
 - G_x = gyro roll rate
 - G_y = gyro pitch rate
 - G_z = gyro yaw rate
 - THETA_x = gyro attitude roll error
 - THETA_y = gyro attitude pitch error
 - THETA_z = gyro attitude yaw error
- From Coarse Sun Sensor Data Processing
 - ECS_y = CSS pitch axis attitude error
 - ECS_z = CSS yaw axis attitude error
 - CSS_{prs} = CSS Sun presence flag

- From Fine Sun Sensor Data Processing
 - EFS_y = FSS pitch axis attitude error
 - EFS_z = FSS yaw axis attitude error
- To Gyro Data Processing
 - IECL = eclipse flag (1 = eclipse)
- To Reaction Wheel Control
 - W_x = compensated roll rate
 - W_y = compensated pitch rate
 - W_z = compensated yaw rate
 - E_x = roll axis attitude pointing error
 - E_y = pitch axis attitude pointing error
 - E_z = yaw axis attitude pointing error
- To Thruster Control
 - W_x = compensated roll rate
 - W_y = compensated pitch rate
 - W_z = compensated yaw rate
 - E_x = roll axis attitude pointing error
 - E_y = pitch axis attitude pointing error
 - E_z = yaw axis attitude pointing error
 - IECL = eclipse flag
- To Reaction Wheel Command Processing
 - IECL = eclipse flag (1 = eclipse)

6.2.2 ALGORITHM

1. Set eclipse flag. If the Sun is present in at least one of the Sun sensors, the IECL flag is set to 0. Otherwise, it is set to 1.

IECL=1

If(CSS_{prs} = 1 or (FSS_{prs} = 1 and IFSF=0))

IECL = 0

Endif

where IFSF = FSS failure flag (1=failure) (DBC)

2. Determine pitch and yaw errors. First use the FSS-derived pitch and yaw attitude errors. If the FSS data are not available, the CSS pitch and yaw errors are used. In the event of no Sun in either of the Sun sensors, the eclipse flag is set, and the spacecraft uses pitch and yaw data from the gyros. If the gyros are used, the spacecraft pitch and yaw attitude errors are at zero, so essentially the spacecraft is in an attitude hold condition.

If IFSF = 0 and FSS_{prs} = 1

ES_y = EFS_y

ES_z = EFS_z

Else

ES_y = ECS_y

ES_z = ECS_z

Endif

The next step is to determine the spacecraft body rates. The straightforward process to determine rates is to use the rates from the gyros. However, if a gyro channel has failed (and the failure flag has been set), the function derives the rate from the Sun sensors. If the spacecraft is also in eclipse, it will drift because no attitude information is available.

The derived rate code for the reaction wheel modes is as follows:

If CONTROL = 0 or 1

If INIT_w = 1 .OR. EXIT = 1 (Initialize filter)

EXIT = 0

UY=YY=ES_y

```

SY10=SY9=SY8=SY7=SY6=SY5=SY4=SY3=SY2=SY1=ESy
WDy = 0
UZ=ZZ=ESz
SZ10=SZ9=SZ8=SZ7=SZ6=SZ5=SZ4=SZ3=SZ2=SZ1=ESz
WDz = 0
Else
(Pitch filter)
YY = AFSR*YY + BFSR*(UY + ESy)
UY = ESy
WDy = -(YY - SY10)/TS10
LIMIT WDy TO ±DRLIM
SY10 = SY9
SY9 = SY8
SY8 = SY7
SY7 = SY6
SY6 = SY5
SY5 = SY4
SY4 = SY3
SY3 = SY2
SY2 = SY1
SY1 = YY
(Yaw filter)
ZZ = AFSR**ZZ + BFSR*(UZ + ESz)
UZ = ESz
WDz = -(ZZ - SZ10)/TS10
LIMIT WDz TO ±DRLIM
SZ10 = SZ9
SZ9 = SZ8
SZ8 = SZ7
SZ7 = SZ6
SZ6 = SZ5
SZ5 = SZ4
SZ4 = SZ3
SZ3 = SZ2
SZ2 = SZ1
SZ1 = ZZ
Endif
Endif

```

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
UY,UZ	Pointing error saved	±1.7 rad
YY,ZZ	Filtered pointing error	±1.7 rad
SY ₁₋₁₀	Previous values of YY	±1.7 rad
SZ ₁₋₁₀	Previous values of ZZ	±1.7 rad
WD _y ,WD _z	Derived rates (pitch and yaw)	±0.0155 rad/sec
TS ₁₀	10 cycle times	2.56 sec

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
AFSR	Filter constant for derived rate for RW control = $1 - 2 \text{ BFSR}$	0.7 to 1.0 n.d.
BFSR	Filter gain for derived rate for RW control = $(1 - \text{AFSR})/2$	0 to 0.3 n.d.
DRILM	Derived rate limit	0 to 0.0155 rad/sec

The derived rate code for the thruster modes is as follows:

```

If CONTROL = 2
  If INITt = 1 .OR. EXIT = 1 (Initialize filter)
    EXIT = 0
    UY=YY=ESy
    SY4=SY3=SY2=SY1=ESy
    WDy = 0
    UZ=ZZ=ESz
    SZ4=SZ3=SZ2=SZ1=ESz
    WDz = 0
  Else
    (Pitch filter)
    YY = AFSH*YY + BFSH*(UY + ESy)
    UY = ESy
    WDy = -(YY - SY4)/TS4
    LIMIT WDy TO ±DRLIM
    SY4 = SY3
    SY3 = SY2
    SY2 = SY1
    SY1 = YY
    (Yaw filter)
    ZZ = AFSH*ZZ + BFSH*(UZ + ESz)
    UZ = ESz
    WDz = -(ZZ - SZ4)/TS4
    LIMIT WDz TO ±DRLIM
    SZ4 = SZ3
    SZ3 = SZ2
    SZ2 = SZ1
    SZ1 = YY
  Endif
Endif

```

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
AFSH	Filter constant = $1 - 2 \text{ BFSH}$	0.0 to 1.0 n.d.
BFSH	Filter gain = $(1 - \text{AFSH})/2$	0.0 to 1.0 n.d.
TS ₄	Derived rate time difference	1.024 sec

The attitude errors used by the control system are determined as follows:

$$E_x = \text{THETA}_x$$
$$W_x = -G_x$$

If the spacecraft has been in an attitude hold or eclipse since the last cycle, the gyro data are used for attitude rates and errors. Otherwise, the Sun sensor data are chosen. If the spacecraft is not in a hold or eclipse mode and there is a gyro channel failure, the appropriate derived rate is chosen.

If (IAH = 1 and OLDIAH = 1) or (IECL = 1 and OLDIECL = 1)
Then

$$E_y = \text{THETA}_y$$
$$E_z = \text{THETA}_z$$
$$W_y = -G_y$$
$$W_z = -G_z$$

Else

$$E_y = ES_y, \text{ limited to } \pm \text{ PLIM}$$
$$E_z = ES_z, \text{ limited to } \pm \text{ PLIM}$$

If IGF₂ = 1 (pitch gyro channel failure)

$$W_y = -WD_y$$

If IGF₃ = 1 (yaw gyro channel failure)

$$W_z = -WD_z$$

Endif

where IGF₂ = pitch gyro channel health flag (0 = good,
1 = bad) (DBC)

IGF₃ = yaw gyro channel health flag (0 = good,
1 = bad) (DBC)

6.3 ACTUATOR COMMAND GENERATION

6.3.1 THRUSTER CONTROL FUNCTION

The thruster control function takes in attitude errors and rates from the attitude error computation function, processes them, determines which ACTs need to be fired, and sends the appropriate commands to the ACTs. This function is performed every 256 msec.

6.3.1.1 Input/Output

- From Magnetometers

BE_y = Earth magnetic field along spacecraft Y-axis

BE_z = Earth magnetic field along spacecraft Z-axis

- From Attitude Error Computation

E_x = spacecraft X-axis attitude error

E_y = spacecraft Y-axis attitude error

E_z = spacecraft Z-axis attitude error

W_x = spacecraft X-axis rate

W_y = spacecraft Y-axis rate

W_z = spacecraft Z-axis rate

IECL = eclipse flag

- From Gyro Data Processing

IGF_1 = roll valid gyro flag

IGF_2 = pitch valid gyro flag

IGF_3 = yaw valid gyro flag

- From FSS Data Processing

FSS_{prs} = FSS Sun presence flag

- From Computational Scheduling

IAH = attitude hold flag

- Initialization

```

MTNABL = 0
INITt = 0
WJx = WJy = WJz = 0
BMy = BMz = 0
Wroll = 0
PHI = 0
PSPx = PSPy = PSPz = 0
NSPx = NSPy = NSPz = 0
WFFy = WFFz = 0

```

6.3.1.2 Algorithm

The first step is to select the proper control gains. If the spacecraft is in an attitude hold condition (IAH=1), the following gains are selected:

```

CHi = CH2i
CPLi = CPL2i    i=1 to 3
CPi = CP2i
CRi = CR2i
AFJ = AFJ2
BFJ = BFJ2

```

If the spacecraft is not in an attitude hold condition (IAH=0), the following gains are assigned:

```

CHi = CH1i
CPLi = CPL1i    i=1 to 3
CPi = CP1i
CRi = CR1i
AFJ = AFJ1
BFJ = BFJ1

```

These gains are modified by ground command as needed to accommodate specific maneuvers. Table 6-1 defines all these DBCs.

Table 6-1. Thruster Control Gain Definitions

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
CH1 _{x,y,z}	Stored constant for roll, pitch, and yaw hysteresis for IAH = .F.	0.0 to 0.9 n.d.
CH2 _{x,y,z}	Stored constant for roll, pitch, and yaw hysteresis for IAH = .T.	0.0 to 0.9 n.d.
CH _{x,y,z}	Roll, pitch, and yaw hysteresis	0.0 to 0.9
CPL1 _{x,y,z}	Stored radian constant for roll, pitch, and yaw position limit for IAH = .F.	0.0175 to 0.175 rad
CPL2 _{x,y,z}	Stored radian constant for roll, pitch, and yaw position limit for IAH = .T.	0.0175 to 0.175 rad
CPL _{x,y,z}	Roll, pitch, and yaw radian position limit	0.0175 to 0.175 rad
CP1 _{x,y,z}	Stored constant for roll, pitch, and yaw error gain for IAH = .F.	0.0 to 50.0 1/rad
CP2 _{x,y,z}	Stored constant for roll, pitch, and yaw error gain for IAH = .T.	0.0 to 50.0 1/rad
CP _{x,y,z}	Roll, pitch, and yaw error gain	0.0 to 50.0 1/rad
AFJ ₁ ,BFJ ₁	Rate filter constants for IAH = .F.; AFJ ₁ + BFJ ₁ = 1.0	0 to 0.99 n.d.
AFJ ₂ ,BFJ ₂	Rate filter AFJ ₂ + BFJ ₂ = 1.0	0 to 0.99 n.d.
AFJ, BFJ	Rate filter AFJ + BFJ = 1.0	0 to 0.99 n.d.
CR1 _{x,y,z}	Stored constant for roll, pitch, and yaw rate gain for IAH = .F.	10.0 to 300.0 sec/rad
CR2 _{x,y,z}	Stored constant for roll, pitch, and yaw rate gain for IAH = .T.	10.0 to 300.0 sec/rad
CR _{x,y,z}	Roll, pitch, yaw rate gain	10.0 to 300.0 sec/rad

The attitude errors received from the attitude error computation function are now limited:

For $i = 1$ to 3
 If $(E_i > CPL_i)$ $E_i = CPL_i$
 If $(E_i < -CPL_i)$ $E_i = -CPL(i)$

These limited attitude errors are then multiplied by a gain and become nondimensional:

For $i = 1$ to 3
 $SP_i = CP_i \cdot E_i$

Rate filtering is then performed:

For $i = 1$ to 3
 $WJ_i = AF_j \cdot WJ_i + BF_j \cdot W_i$

Definitions

where SP_i = limited, scaled attitude errors where $i = 1$ to 3 (nondimensional)

WJ_i = filtered spacecraft rates where $i = 1$ to 3 (rad/sec)

The attitude error data from the Sun sensors is combined with the rate data from the gyros (Sun sensors in case of pitch or yaw gyro channel failure), which will be fed to a control relay:

$$\begin{aligned} S_x &= SP_1 + CR1 \cdot WJ_1 \\ S_y &= SP_2 + CR2 \cdot WJ_2 - WFF_y \\ S_z &= SP_3 + CR3 \cdot WJ_3 - WFF_z \end{aligned}$$

where S_i = spacecraft axis control status where $i = 1$ to 3

WFF_y = pitch modulator feedback signals used when pitch and/or yaw rates are derived from the Sun sensors

WFF_z = yaw modulator feedback signals used when pitch and/or yaw rates are derived from the Sun sensors

If the roll gyro channel are bad, the following algorithm is implemented to control the roll rate. This is performed by taking the y,z magnetometer readings and determining the rate of change of this vector. The control law attempts to limit this rate of change to 0. Since the Earth's magnetic field vector changes at a maximum rate of 2 revolutions per orbit (rpo), even if the spacecraft started with a 0 roll rate, it would develop up to a 2-rpo rate due to this change. The code and definitions are as follows:

```

If (IGF(1) .EQ. 1)
  BMy = AF1 * BMy + BF1 * BEy
  BMz = AF1 * BMz + BF1 * BEz
  OLDPHI = PHI
  MTB = SQRT (BMy*BMy + BMz*BMz)
  If (MTB .GE. TBLIM)
    PHI = BMy/MTB
  Else
    PHI = 0.0
  Endif
  PHIDIF = PHI - OLDPHI
  PHID = PHIDIF * SIGN (BMz)/TS
  If (PHIDIF .GT. RLIM) PHID = RLIM
  If (PHID .LT. -RLIM) PHID = -RLIM
  WROLL = AF2 * WROLL + BF2 * PHID
  WROLL = WROLL + DELW * TSPX
  SX = -WROLL * KRMAG
Endif

```

Symbol	Definition	Range
AF ₁ , BF ₁	Magnetic field filter gains	0 to 1 n.d.
AF ₂ , BF ₂	Derived rate filter gains	0 to 1 n.d.
BE _y , BE _z	Compensated magnetic field components	± 0.9 gauss
BM _y , BM _z	Filtered magnetic field components	± 1.12 gauss
MTB	YZ-plane magnetic field	0 to 0.9 gauss
PHI	Angle between Z-axis and TB	± 1.0 rad
PHIDIF	PHI minus previous PHI	± 2.0 rad
OLDPHI	Previous PHI	± 1.0 rad
PHID	Derived roll rate unfiltered	± 0.0155 rad/sec
WROLL	Derived roll rate filtered	± 0.0155 rad/sec

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
DELW	Modulator gain	± 0.002 n.d.
RLIM	Derived roll rate limit	0 to 0.0155 rad/sec
KRMAG	Rate gain	0 to 1000 1/sec
TS	Cycle time in seconds	0.256 sec
TBLIM	Lower limit on magnetic field magnitude	0.0001 to 0.1
TSPX	Roll axis firing direction	0, ± 1 n.d.

The control relay computation is now performed. The output of this section is roll, pitch, and yaw firing requests. The code and definitions are as follows:

```

Do For i = X,Y,Z
  If (Si ≥ 1)
    LHi = 1
  Else
    If (Si ≥ 1 - CHi)
      If (LHi = -1)
        LHi = 0
      Endif
    Else
      If (Si ≤ -1)
        LHi = -1
      Else
        If (Si ≤ -1 + CHi)
          If (LHi = 1)
            LHi = 0
          Endif
        Else
          LHi = 0
        Endif
      Endif
    Endif
  Endif
Enddo

```

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
LH _{x,y,z}	Roll, pitch, and yaw firing request flags	-1 or 0 or 1 n.d.

The firing requests are then scheduled, and the appropriate commands are sent to the ACTs. The code for this section is as follows:

```

    If (ISTATE .EQ. 1) GO TO 20
    If (ISTATE .EQ. 2) GO TO 30
    If (LHX .NE. 0) ISTATE = 1; GO TO 50
20  If (LHY .NE. 0) ISTATE = 2; GO TO 50
30  If (LHZ .NE. 0) ISTATE = 3; GO TO 50
    If (LHX .NE. 0) ISTATE = 1; GO TO 50
    If (LHY .NE. 0) ISTATE = 2; GO TO 50
    ISTATE = 0
50  Continue
    Do for i = x,y,z/1,2,3/roll,pitch,yaw
    TSPi = 0
        If (ISTATE .EQ. i)
            If (LHi .EQ. 1)
                TSPi = 1
                Set command CLLTPi = TRUE for positive
                i-axis firing
                PSPi = PSPi + 1
            Else
                TSPi = -1
                Set command CLLTNi = TRUE for negative
                i-axis firing
                NSPi = NSPi + 1
            Endif
        Endif
    Enddo

```

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
ISTATE	Thruster state of sequential logic	0, 1, 2, 3 n.d.
TSP _i	Thruster firing direction, i = x,y,z	0, ± 1 n.d.
CLLTP _i	Low-level thruster firing command for positive rotation about axis i = x,y,z (roll, pitch, yaw)	True of False
CLLTN _i	Low-level thruster firing command for negative rotation about axis i = x,y,z (R,P,Y)	

The last section of thruster command is the derived rate modulator. This is only used when the spacecraft is deriving pitch and/or yaw rates from the CSSs provided the rate is greater than MODRL.

```

For i = 2 to 3
If (IGFi = 0 or IECL = 1 or ABS( Wi < MODRL) or
FSSPRS = 1) then
WFFi = 0.0
Else
WFFi = AFFi • WFFi + BFFi • TSPi
limit WFFi to ± WFFLIM
Endif

```

where MODRL = modulator rate limit (rad/sec) (DBC)
AFF_i = rate modulator constant i=2 to 3 (DBC)
BFF_i = rate modulator gain i=2 to 3 (DBC)
WFFLIM = modulator output limit (DBC)

6.3.2 REACTION WHEEL CONTROL FUNCTION

The reaction wheel control function processes input from the attitude error computation function to generate spacecraft body control torques for the reaction wheels. All error and rate signals are limited before control law computation proper, from which spacecraft control torques are developed and then filtered and limited for output to the reaction wheel command processing function. If the roll scan flag (ISCAN) or the X-axis gyro failure flag (IGF(1)=1) is set, an alternate roll control law is executed. This function is performed every 256 msec.

6.3.2.1 Input/Output

- From Computational Scheduling

ISCAN = roll scan flag

INIT_w = reaction wheel initialization flag

- From the Magnetometer Data Processing

BE_y = Earth magnetic field along spacecraft Y-axis

BE_z = Earth magnetic field along spacecraft Z-axis

- From Attitude Error Computation

W_x = compensated roll rate

W_y = compensated pitch rate

W_z = compensated yaw rate

E_x = roll axis pointing error

E_y = pitch axis pointing error

E_z = yaw axis pointing error

- To Magnetic Control

MTNABL = magnetic torquer enable flag

- To Reaction Wheel Command Processing

T_x = roll axis control torque (ft-lbf)

T_y = pitch axis control torque (ft-lbf)

T_z = yaw axis control torque (ft-lbf)

I_{cage} = reaction wheel cage flag

6.3.2.2 Algorithm

The following variables are set during initialization:

If $INIT_w = 1$

$INIT_w = 0$

$I_{cage} = 1$

$HCW_i = 0$ $i = 1$ to 4

$Q_i = 0$ $i = x, y, z$

$EP_i = 0$ $i = x, y, z$

Endif

The attitude errors and rate signals are limited before they are inserted into the reaction wheel control law.

For $i=x, y, z$

 If $ABS(E_i) > ELIM_i$

$EC_i = ELIM_i \cdot sign(E_i)$

 Else

```

    ECi = Ei
Endif
If ABS(Wi) > WLIM
    Wi = WLIM * sign(Wi)
Endif

```

where ELIM_i = attitude error limits where i=x,y,z (rad)
(DBC)

WLIM = rate limit (rad/sec) (DBC)

The error signals are then integrated and limited:

```

For i=x,y,z
    Qi = Qi + (ECi + EPi) • TS2
    If ABS(Qi) > QLIMi
        Qi = QLIMi • sign (Qi)
    Endif
    EPi = ECi

```

where Q_i = integrated errors where i=x,y,z (rad/sec)
 TS₂ = half of the sample period (sec)
 QLIM_i = integrated error limits (rad/sec) (DBC)
 EP_i = past values of EC_i where i=x,y,z (rad)
 EC_i = limited error signal where i=x,y,z (rad)

The control torque computations for all three axes are then performed:

```

For i=x,y,z
    TEMPi = KPi • ECi + KWi • Wi + KQi • Qi

```

where KP_i = error gains where i=x,y,z (ft-lbf/rad) (DBC)
 KW_i = rate gains where i=x,y,z (ft-lbf/rad/sec) (DBC)
 KQ_i = integrated error gains where i=x,y,z (ft-lbf/rad-sec) (DBC)
 TEMP_i = Control torques where i=x,y,z (ft-lbf)

If the spacecraft is performing a roll scan (ISCAN=1), the following control law is used:

If ISCAN=1

$$\text{TEMP}_x = \text{KWW} \cdot (\text{WXSTAR} + \text{W}_x)$$

Endif

where KWW = roll scan roll rate gain (ft-lbf/rad/sec)
(DBC)

WXSTAR = roll scan rate command (rad/sec) (DBC)

In the case of bad gyro data (IGF(1)=1), the reaction wheel control function turns to control law, which limits the roll rate by controlling the rate of change of the measured magnetic field vector projected onto the spacecraft Y-Z plane. The maximum roll rate allowed by the control law is 0.3 deg/sec. The backup roll control law code and definitions for the various constants are as follows:

```
If(IGF(1) .EQ. 1) Then
  If(ABS(Ey) ≤ ETHR .AND. ABS(Ez) ≤ ETHR) Then
    TB = SQRT(BEy • BEy + BEz • BEz)
    IF(TB .GE. TBLIM) Then
      ER = BEy/TB
    Else
      ER = 0.0
    Endif
    BP = B
    B = ABF • BP + BBF • ER
    DELTB = (B-BP) • SIGN(BEz)
    If (ABS(DELTB) .GT. SRLIM)
      DELTB = SRLIM • SIGN(DELTB)
    Endif
    Y1 = B • PGAIN + DELTB • RGAIN
    If(ABS(Y1) .GT. XTLIM)
      Y1 = XTLIM • SIGN(Y1)
    Endif
    TEMPx = -Y1
  Else
    TEMPx = 0
  Endif
Endif
```

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
XTLIM	Roll control torque limit	0 to +0.625 ft-lbf
TBLIM	Lower limit on magnetic field magnitude	0.0001 to 0.1 gauss
TB	Magnetic field magnitude	± 0.9 gauss
ER	Normalized magnetic field in Y-direction	± 1.0 n.d.
B	Pseudo position signal	$\pm 1.$ rad
BP	Past value of B	$\pm 1.$ rad
ABF	Filter constant	$\pm 1.$ n.d.
BBF	Filter constant	$\pm 1.$ n.d.
DELTB	Pseudo rate signal	± 0.04 rad/sec
SRLIM	Rate limit value	0 to 0.0155 rad/sec
Y1	Unlimited control torque	± 0.625 ft-lbf
PGAIN	Position gain	± 1 ft-lbf/rad
RGAIN	Rate gain	± 300 ft-lbf/rad/sec
IGF(1)	Roll gyro data quality flag (1=bad, 0=good)	(DBC)

The control torque limiting and filtering code is as follows:

```

Do For i = x, y, z
  If ABS(TEMPi) > TLIM
    TEMPi = TLIM * sign(TEMPi)
  Endif
  UTPi = UTi
  UTi = TEMPi
  ATPi = ATi
  ATi = AF * ATPi + BF * (UTi + UTPi)
  TPi = Ti
  Ti = AF * TPi + BF * (ATi + ATPi)
Enddo

```

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
AF	Filter constant = $1.0 - 2 BF$	0.7 to 1.0 n.d.
BF	Filter constant = $(1.0 - AF)/2$	0.0 to 0.3 n.d.
UT _{y,z}	Buffered value of TEMP _{y,z}	± 0.7 ft-lbf
UTP _{y,z}	Past value of UT _{x,y,z}	± 0.7 ft-lbf
T _{x,y,z}	Control torques	± 0.5 ft-lbf

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
ATP _{x,y,z}	Past values of AT _{x,y,z}	±0.5 ft-lbf
TLIM	Control torque limit	0 to 0.625 ft-lbf

The last function of the reaction wheel control law is to enable/disable momentum unloading. If the pitch and yaw attitude errors are not excessive (<ETHR), the momentum unloading function is enabled; otherwise, this function is disabled. The Sun-pointing error threshold data base constant, ETHR, is currently set to 5 deg. This value ensures that no momentum unloading is performed during Sun acquisition maneuvers or when there is excessive pointing error. The code for this section is as follows:

```

If ABS(Ey) < ETHR .AND. ABS(Ez) < ETHR
  MTNABL = 1
Else
  MTNABL = 0
Endif

```

<u>Symbol</u>	<u>Definition</u>	<u>Range</u>
ETHR	Sun-pointing error threshold	0.00 to 0.20 rad

6.3.3 REACTION WHEEL COMMAND PROCESSING

The reaction wheel command processing function computes individual reaction wheel motor torquer commands from the spacecraft control torques determined in the reaction wheel control function. In addition, this function provides reaction wheel caging commands during eclipse portions of the orbit if pitch or yaw axis gyro failure has been indicated by ground command. This function is performed every 256 msec.

6.3.3.1 Input/Output

- From Computational Scheduling

IAH = attitude hold flag (1=attitude hold)

- From Attitude Error Computation
IECL = eclipse flag (1=eclipse)
- From Reaction Wheel Momentum Computation
HW_i = reaction wheel angular momentum where
i=1 to 4 (ft-lbf)
- From Reaction Wheel Control
T_x = roll axis control torque
T_y = pitch axis control torque
T_z = yaw axis control torque
ICAGE = reaction wheel cage flag (1=cage)
- To Truth Model
COM_i = reaction wheel command where i=1 to 4
(ft-lbf)

6.3.3.2 Algorithm

The first step is to choose between the caging logic (Section A) and the normal distribution logic (Section B). If there is either bad pitch or yaw gyro data and either an eclipse or attitude hold, the caging logic is used. Otherwise, the normal distribution logic is used.

```

If ((IGF2 = 1 or IGF3 = 1) and (IECL = 1 or IAH = 1))
  Caging Logic (Section A)
Else
  Normal Distribution Logic (Section B)
Endif

```

where IGF₂ = pitch axis valid gyro data flag (0=good,1=bad)
(DBC)
IGF₃ = yaw axis valid gyro data flag (0=good,1=bad)
(DBC)

6.3.3.2.1 Section A - Caging Logic

Magnetic torquer commands are disabled when the reaction wheels are caged. The first time through, the caging logic sets the individual reaction wheel control momenta, while

subsequent passes compute reaction wheel control torques to maintain constant reaction wheel momenta. This logic tries to keep the overall spacecraft momentum constant during eclipses and/or attitude hold conditions when the gyro data for the pitch and/or yaw axes are bad.

```

    MTNABL = 0
  If ICAGE = 1
    ICAGE = 0
    HCWi = HWi  i = 1 to 4
  Endif
  TRWi = AA • (HCWi - HWi)  i = 1 to 4
  For i = 1 to 4
    If ABS (TRWi) > TRWlim
      TRWi = TRWlim • sign (TRWi)
    Endif
  
```

where HWC_i = initial values of reaction wheel momenta
 where i=1 to 4
 AA = reaction wheel cage gain (1/sec) (DBC)
 TRW_{lim} = reaction wheel torque limit (ft-lbf) (DBC)

6.3.3.2.2 Section B - Normal Distribution Logic

The desired three-axis control torques are distributed to the four reaction wheels. Command redistribution to avoid single wheel saturation is then performed unless a wheel failure has been indicated by external commands.

```

    ICAGE = 1
    TRW(4x1 vector) = TRWd (4x3 matrix) * T(3x1 vector)
  If FAILW = 0
    Tδ = (HW1 - HW2 - HW3 + HW4) * Cτ
    TRW1 = TRW1 - Tδ
  
```

$$\begin{aligned} \text{TRW}_2 &= \text{TRW}_2 + T_\delta \\ \text{TRW}_3 &= \text{TRW}_3 + T_\delta \\ \text{TRW}_4 &= \text{TRW}_4 - T_\delta \end{aligned}$$

Endif

where FAILW = reaction wheel failure flag (=failed wheel number or 0)

C_T = Redistribution constant (1/sec)

TRW_i = reaction wheel command torques where $i = 1$ to 4 (ft-lbf)

T_δ = Torque adjustment factor (ft-lbf)

TRW_d = (4x3 matrix) = reaction wheel distribution matrix (see Table 6-2 for values)

6.3.3.2.3 Section C

Once Section A or B is finished, the command torque for each reaction wheel (TRW_i) is scaled due to reaction wheel friction, and the commands in counts to each reaction wheel are determined.

$$\text{COM}_i = (\text{TRW}_i + \text{TEX}_i \cdot \text{sign}(\text{HW}_i)) \cdot \text{CONV} \quad i=1 \text{ to } 4$$

These commands are then set to zero if the respective reaction wheel has failed and the values are limited.

For $i=1$ to 4

If FAILW = i

$$\text{COM}_i = 0$$

Else

Limit COM_i to ± 2047 counts

Endif

where TEX_i = reaction wheel friction parameters where $i=1$ to 4 (ft-lbf)

CONV = foot-pound-to-count conversions (counts/ft-lbf)

Table 6-2. Values for Scaling the Components of TRW_d

<u>TRW_d</u> <u>Component</u>	<u>No Wheel</u> <u>Disabled</u>	<u>Wheel 1</u> <u>Disabled</u>	<u>Wheel 2</u> <u>Disabled</u>	<u>Wheel 3</u> <u>Disabled</u>	<u>Wheel 4</u> <u>Disabled</u>
TRW _d (1,1) =	-0.5157	0	-1.031	-1.031	0
TRW _d (1,2) =	+0.4042	0	0	+0.8085	+0.8085
TRW _d (1,3) =	-0.4042	0	-0.8085	0	-0.8085
TRW _d (2,1) =	-0.5157	-1.031	0	0	-1.031
TRW _d (2,2) =	-0.4042	0	0	-0.8085	-0.8085
TRW _d (2,3) =	-0.4042	-0.8085	0	-0.8085	0
TRW _d (3,1) =	-0.5157	-1.031	0	0	-1.031
TRW _d (3,2) =	+0.4042	+0.8085	+0.8085	0	0
TRW _d (3,3) =	+0.4042	0	+0.8085	0	+0.8085
TRW _d (4,1) =	-0.5157	0	-1.031	-1.031	0
TRW _d (4,2) =	-0.4042	-0.8085	-0.8085	0	0
TRW _d (4,3) =	+0.4042	+0.8085	0	+0.8085	0

NOTE: The ACE/CPE does not set TRW_d. TRW_d comes from either the default ACE/CPE PROM (programmable read-only memory) value or the OBC.

This section takes the commands in counts, converts them to newton-meters, and ships the commanded torque values (COM_i where $i=1$ to 4) for each reaction wheel to the Truth Model.

$$COM_i = 1.3558179 \cdot (COM_i/CONV) \quad i=1 \text{ to } 4$$

6.3.4 MAGNETIC CONTROL LAW

The magnetic control law function performs the computations required to unload momentum stored in the reaction wheels. The actual unloading of the momentum is performed by sending the appropriate commands to the magnetic torquers, which interact with the Earth's magnetic field to produce a control torque. This function is performed every 512 msec.

6.3.4.1 Input/Output

- From Reaction Wheel Momentum Computation
 - HW_i = wheel i stored angular momentum (ft-lbf-sec)
wheel where $i=1$ to 4
- From Magnetometer Data Processing
 - BE_x = Earth magnetic field along spacecraft X-axis
(gauss)
 - BE_y = Earth magnetic field along spacecraft Y-axis
(gauss)
 - BE_z = Earth magnetic field along spacecraft Z-axis
(gauss)
- From Reaction Wheel Control
 - MTN_{abl} = magnetic torquer command enable
- To Torquer Bars
 - $MTORQ_i$ = commanded magnetic moment for torquer i
(counts) where $i=1$ to 3

6.3.4.2 Algorithm

If a particular wheel has been flagged as failed, the angular momentum for that wheel is set to 0:

For $i=1$ to 4

If($FAILW_i = 1$) $HW_i=0$

The angular momentum to be dumped is the angular momentum of all four reaction wheels projected onto the three body axes. Figure 6-1 shows the reaction wheel assembly configuration. From this figure, the reaction wheel axes for each wheel can be determined in body coordinates as follows:

$$R_1 = (CCBETA, -CSBETA*CCALPHA, CSBETA*CSALPHA)$$

$$R_2 = (CCBETA, CSBETA*CSALPHA, CSBETA*CCALPHA)$$

$$R_3 = (CCBETA, -CSBETA*CSALPHA, -CSBETA*CCALPHA)$$

$$R_4 = (CCBETA, CSBETA*CCALPHA, -CSBETA*CSALPHA)$$

Assuming that each reaction wheel has angular momentum HW_i ($i=1$ to 4) with the appropriate sign designating the direction of the spin (+/-), the angular momentum about each body axis is as follows:

$$H_1 = (HW_1 + HW_2 + HW_3 + HW_4) * CCBETA$$

$$H_2 = ((HW_4 - HW_1)) * CCALPHA + (HW_2 - HW_3) * CSALPHA \\ *CSBETA$$

$$H_3 = ((HW_1 - HW_4)) * CSALPHA + (HW_2 - HW_3) * CCALPHA \\ *CSBETA$$

The momentum is then converted from foot-pound-force-seconds to newton-meters:

For $i=1$ to 3

$$HD_i = H_i * 1.35582$$

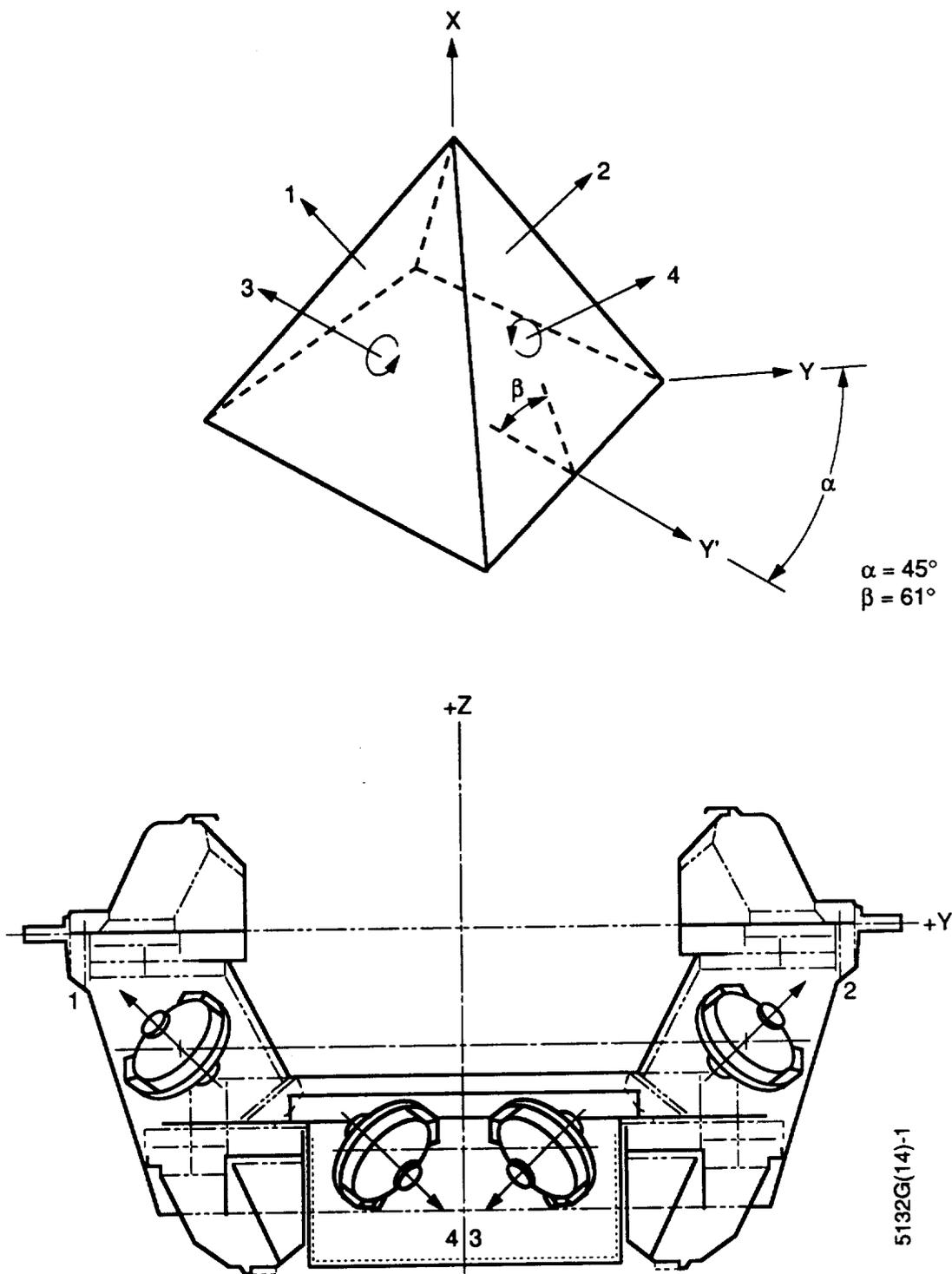


Figure 6-1. GRO Reaction Wheel Configuration

To determine the magnetic moment the spacecraft must develop to dump the reaction wheel momentum, the following equations must be examined:

$$\vec{T} = \vec{m} \times \vec{B}$$

$$\vec{T} = d \vec{H} / dt$$

where \vec{T} = torque vector
 \vec{m} = magnetic moment vector
 \vec{B} = magnetic field vector
 \vec{H} = angular momentum of the spacecraft

It should be noted that H is in the same direction as \vec{T} and the vector parallel to the magnetic moment vector (\vec{M}') is found by taking the following cross product:

$$\vec{M}' = \vec{B} \times \vec{H}$$

where \vec{M}' is in the same direction as \vec{M} .

Once \vec{M}' has been calculated, the vector is transformed from body coordinates to torquer bar coordinates (see Figure 6-2):

$$\begin{bmatrix} MTC_1 \\ MTC_2 \\ MTC_3 \end{bmatrix} = \begin{bmatrix} AMB_1 & AMB_4 & AMB_7 \\ AMB_2 & AMB_5 & AMB_8 \\ AMB_3 & AMB_6 & AMB_9 \end{bmatrix} \begin{bmatrix} M_1' \\ M_2' \\ M_3' \end{bmatrix}$$

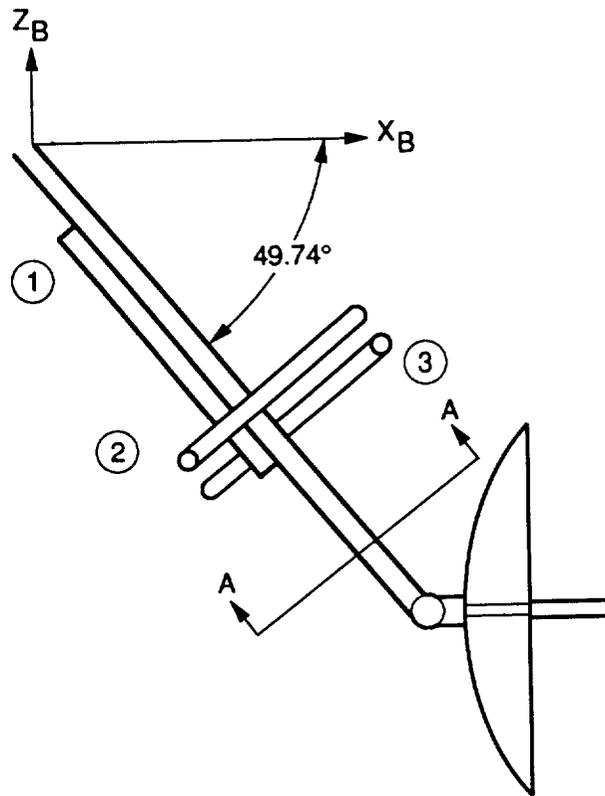
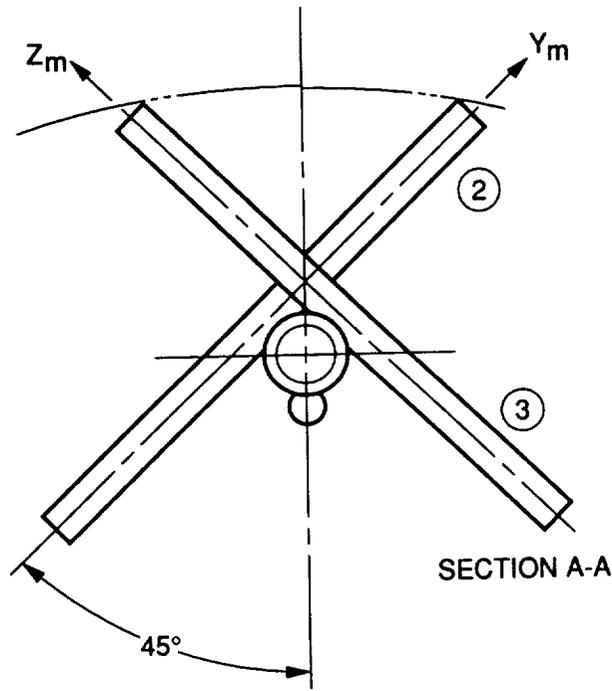
The vector MTC_i ($i=1$ to 3) is now converted to counts that will be sent to the torquer bars, and MTC_i is compared against the maximum allowable torquer command and limited if necessary. The minus creates a magnetic moment vector that, reacting with the Earth's magnetic field, will create a torque to decrease the excess angular momentum created by the reaction wheels.

For $i=1$ to 3

If ($ABS(MTC_i) < CMAXTQ_i$)

$MTORQ_i = -CM \cdot MTC_i$

$MTORQ_i = MTORQ_i \cdot CTDE_i$



5132G(14)-2

Figure 6-2. Magnetic Torquer Location and Orientation

```

Else
  MOTRQi = -(CMD • sign(MTCi))
Endif

```

If the momentum unloading function is disabled, the commands to the torquer bars are set to 0.

```

If(MTNABL = 0)
  MTORQi=0    i=1 to 3
Endif

```

Definitions

AMB(1-9) = transformation from body coordinates to magnetic torquer coordinates (DBC)

CTDE(1-3) = magnetic torquer scale factor (counts/A-m²) (DBC)

CM = cross-product control law gain (1/sec-gauss-tesla) (DBC)

CMD = magnetic torquer command limit (counts) (DBC)

CMAXTQ(1-3) = CMD/(CTDE(1-3) * CM) (N-m-gauss-sec) (DBC)

CCALPHA, CCBETA, CSALPHA, CSBETA = (DBC)

6.4 ACE CONTROL

6.4.1 OBC/ACE TRANSITIONS

When a transition occurs from the OBC to the ACE, the OBC passes the status parameters to the ACE and continues to perform sensor processing. These parameters are not handed over in the case of an OBC failure, in which case the ACE uses default values for these parameters. The actual control in this condition resides in the ACE, which does its own sensor processing and sends commands to the actuators. The main parameters handed over by the OBC are the gyro and reaction wheel status flags. The OBC has control in only one area: transitions between the SRPM to the SHM due to

loss of the Sun or failure to acquire the Sun. The OBC has a timer that starts up whenever the Sun signal from the FSS Sun presence sensor is lost. The Sun lost counter limit is a value that can be changed from the ground. If the counter exceeds the counter limit, the OBC directs the ACE to transition from the SRPM to the SHM, and Sun acquisition is tried in this mode. Table 6-3 lists the initial values of the various DBCs used in the ACE and those used to direct the IRU to use high-rate mode and to select FSS head 2 data. The power-on initialization function is defined by the following algorithm:

- Clear RAM from locations 21 to 7FF (hex) inclusive
- Initialize all RAM variables identified in Table 6-3
- Select IRU high-rate mode
- Select FSSA head 2

6.4.2 COMPUTATIONAL SCHEDULING

The various ACE control processing electronics (ACE/CPE) sensor and actuator functions are called by this portion of the firmware. The cycle time is 128 msec.

6.4.2.1 Input/Output

- To Computational Scheduler From Ground or OBC
 - BMC (1-5) = firmware mode from ground
 - = 1, Sun reference pointing
 - = 2, reaction wheel attitude hold
 - = 3, reaction wheel roll scan
 - = 4, safe hold mode
 - = 5, thruster attitude hold
 - ILATCH (1-2) = actuator control flag from OBC
 - = 0, reaction wheel control when spacecraft was on OBC
 - = 1, thruster control when spacecraft was on OBC

Table 6-3. RAM Variables To Be Initialized During Power-On Initialization (1 of 4)

<u>Name</u>	<u>DBC Group Definition</u>	<u>Value</u>
SPT	Solar Panel Timer	3056
CBIAS	Gyro Data Processing	0 for all elements
CNBIAS(i)	Gyro Data Processing	255 for all elements
IGF	Gyro Data Processing	0, 0, 0, 1, 0, 0
CTG	Gyro Data Processing	3 x 3 identify matrix
CGP	Gyro Data Processing	7.75×10^{-6}
CGR	Gyro Data Processing	3.03×10^{-5}
AFG	Gyro Data Processing	0.7132
BFG	Gyro Data Processing	0.2868
CSF	CSS Data Processing	2.1379×10^{-4}
CTHRESH	CSS Data Processing	240
AF11	FSS Data Processing	-0.62487
AF21	FSS Data Processing	-0.62487
AF12	FSS Data Processing	0.7628 E-4
AF22	FSS Data Processing	0.7628 E-4
AF19	FSS Data Processing	0
AF29	FSS Data Processing	0
FSYB2	FSS Data Processing	0.03491
CWS	Reaction Wheel Momentum Computation	0.260408
CMS	Magnetometer Data Processing	-3.315×10^{-4}
CBO	Magnetometer Data Processing	0
CBXB, CBYB, CBZB	Magnetometer Data Processing	0.022, 0.022, 0.022
AFSR	Attitude Error Computation	0.8511
BFSR	Attitude Error Computation	0.0744
AFSH	Attitude Error Computation	0.4264

Table 6-3. RAM Variables To Be Initialized During Power-On Initialization (2 of 4)

<u>Name</u>	<u>DBC Group Definition</u>	<u>Value</u>
BFSH	Attitude Error Computation	0.2868
TS4	Attitude Error Computation	1.024
.TS10	Attitude Error Computation	2.56
DRLIM	Attitude Error Computation	0.0155
PLIM	Attitude Error Computation	0.254
ELIM _{x,y,z}	Reaction Wheel Control	0.0622,0.0847,0.0894
WLIM	Reaction Wheel Control	0.00436
AF	Reaction Wheel Control	0.9182
ABF	Reaction Wheel Control	0.97993
BF	Reaction Wheel Control	0.0409
BBF	Reaction Wheel Control	0.02007
TS2	Reaction Wheel Control	0.128
QLIM _{x,y,z}	Reaction Wheel Control	0.4401,0.7800,0.8200
PGAIN	Reaction Wheel Control	0.01
RGAIN	Reaction Wheel Control	125.
SRLIM	Reaction Wheel Control	0.02
KP _{x,y,z}	Reaction Wheel Control	19.82,29.00,35.63
KW _{x,y,z}	Reaction Wheel Control	1479.0,2164.0,2659.0
KQ _{x,y,z}	Reaction Wheel Control	0.095,0.138,0.171
TLIM	Reaction Wheel Control	0.3646
TBLIM	Reaction Wheel Control	0.001
KWW	Reaction Wheel Control	1551.0
WXSTAR	Reaction Wheel Control	0.00087
ETHR	Reaction Wheel Control	0.087
XTLIM	Reaction Wheel Control	0.3646
CH1 _{x,y,z}	Thruster Control	0.00, 0.00, 0.00

Table 6-3. RAM Variables To Be Initialized During Power-On Initialization (3 of 4)

<u>Name</u>	<u>DBC Group Definition</u>	<u>Value</u>
CH2 _{x,y,z}	Thruster Control	0.0, 0.0, 0.0
CP1 _{x,y,z}	Thruster Control	11.12, 11.0, 10.10
CP2 _{x,y,z}	Thruster Control	11.12, 11.0, 10.10
CR1 _{x,y,z}	Thruster Control	22.06, 29.14, 85.8
CR2 _{x,y,z}	Thruster Control	22.06, 29.14, 85.8
CPL1 _{x,y,z}	Thruster Control	0.103, 0.101, 0.122
CPL2 _{x,y,z}	Thruster Control	0.103, 0.101, 0.122
AFJ ₁	Thruster Control	0.01
BFJ ₁	Thruster Control	0.99
AFJ ₂	Thruster Control	0.01
BFJ ₂	Thruster Control	0.99
AF ₁	Thruster Control	0.975
BF ₁	Thruster Control	0.025
RLIM	Thruster Control	0.008
KRMAG	Thruster Control	461
AF ₂	Thruster Control	0.994
BF ₂	Thruster Control	0.006
DELW	Thruster Control	0.0008
TS	Thruster Control	0.256
MODRL	Thruster Control	0.001745
AFF _y	Thruster Control	0.9974
BFF _y	Thruster Control	0.1023
AFF _z	Thruster Control	0.9987
BFF _z	Thruster Control	0.0512
WFFLIM	Thruster Control	3.0
CM	Magnetic Control Law	250.0
CCALPHA	Magnetic Control Law	0.70711
CSALPHA	Magnetic Control Law	0.70711
CCBETA	Magnetic Control Law	0.48481
CSBETA	Magnetic Control Law	0.87462

Table 6-3. RAM Variables To Be Initialized During Power-On Initialization (4 of 4)

<u>Name</u>	<u>DBC Group Definition</u>	<u>Value</u>
AMB ₁₋₉	Magnetic Control Law	0.66674, 0.52701, 0.52701, 0, 0.70711, -0.70711. -0.74529, 0.47145, 0.47145
CTDE _{1,2,3}	Magnetic Control Law	1.02228
CMD	Magnetic Control Law	2047
CMAXTQ _{1,2,3}	Magnetic Control Law	8.009
AA	Reaction Wheel Command Processing	0.3
TRWLIM	Reaction Wheel Command	0.5
CTAU	Reaction Wheel Command	0.00020825
TPHI	Reaction Wheel Command	0
TRWD	Reaction Wheel Command	-0.51567, 0.40424, -0.40424, -0.51567, -0.40424, -0.40424, -0.51567, 0.40424, 0.40424, -0.51567, -0.40424, 0.40424
CONV	Reaction Wheel Command	3208.556
TEX _{1,2,3,4}	Reaction Wheel Command	0.0528
LOMEM	Reaction Wheel Command	21 hex
HIMEM	Reaction Wheel Command	7FF (hex)

- To Attitude Error Computation
 - IAH = attitude hold flag (1=attitude hold)
 - CONTROL = control mode flag (1=reaction wheels,
2=ACTs)
 - INIT_t = initialize ACT firing (1=yes)
 - INIT_w = initialize reaction wheels (1=yes)
- To Reaction Wheel Control
 - ISCAN = roll scan flag (1=yes)
 - INIT_w = initialize reaction wheels (1=yes)
- To Thruster Control
 - INIT_t = initialize ACT firing (1=yes)
 - IAH = attitude hold flag

Initialization

SLOT=0

6.4.2.2 Main Loop

The following is a listing of the code contained in the main loop of the computational scheduler:

```

OLDBMC=BMC
OLDCONTROL=CONTROL
SLOT=SLOT+1
If(SLOT>4) SLOT=1

```

Read in the mode storage register and check to see if the mode is recognized. If not, determine when the actuator was last used under the OBC and switch to the same actuator under the ACE/CPE.

```

If(BMC not equal 1,2,3,4,or 5) Then
  If(OBC ok flag is false for 3 consecutive cycles) Then
    If(ILATCH = 0) Then
      BMC=1 (Sun reference pointing)
    Else
      BMC=4 (safe hold)
    Endif
  Endif
Endif

```

Perform mode control if the mode has changed this loop.

```
If(BMC not equal OLDBMC) Then
  If(BMC = 1) (Sun reference pointing mode)
    control=1
    IAH=0
    ISCAN=0
  Else if(BMC = 2) (reaction wheel attitude hold mode)
    CONTROL=1
    IAH=1
    ISCAN=0
  Else if(BMC = 3) (roll scan mode)
    CONTROL=1
    IAH=0
    ISCAN=1
  Else if(BMC = 4) (safe hold mode)
    CONTROL=2
    IAH=0
    ISCAN=0
  Else if(BMC = 5) (thruster attitude hold mode)
    CONTROL=2
    IAH=1
    ISCAN=0
  Else (No recognized mode)
    CONTROL=0
    IAH=0
    ISCAN=0

  Endif
```

The following code determines whether reaction wheel or thruster control needs to be initiated:

```
If(CONTROL = 1 and OLDCONTROL not equal 1) Then
  INITw=1
  Disable roll thruster firings
  Disable pitch thruster firings
  Disable yaw thruster firings

Else if(CONTROL = 2 and OLDCONTROL not equal 2) Then
  INITt=1
  Enable roll thruster firings
  Enable pitch thruster firings
  Enable yaw thruster firings
  Issue zero commands to RWA
  Issue zero commands to MTA
  Disable commands to OATs 1-4
  Issue OAT BANK1 and BANK2 power off
  Issue load OAT pulse timer with T = 1

Endif
```

Once mode control has been performed (if needed), the scheduler performs the solar array indexing for the SRPM and SHM:

```
If(BMC=srpm OR BMC=shm) Then
  If (splf .NE. 1 and sp2f .NE. 1) Then
    If (IAH = 1 and OLDIAH < > 1) Then
      splt = sp2t = 0      ! initialize counters
    Endif
    If (solar array 1 (SA1) is not indexed) Then
      Command SA1 to index
      splt = splt + 1
      If (splt > spt) splf = 1
    Else if (solar array 2 (SA2) is not indexed) Then
      Command SA2 to index
      sp2t = sp2t + 1
      If (sp2t > spt) sp2f = 1
      IAH=1
    Else
      IAH = 0
    Endif
  Else
    IAH = 0
  Endif
Endif
```

The sensor and external command processing is then performed:

```
Call external command processing
Call gyro data processing
If(SLOT = 2 or SLOT = 4) Then
  Call CSS data processing
  Call FSS data processing
  Call reaction wheel momentum computation
Else
  Call magnetometer data processing
Endif
```

Attitude error computation is now scheduled:

```
If(SLOT = 2 or SLOT = 4) Then
  Call attitude error computation
Endif
```

Actuator processing is now scheduled:

```
If(CONTROL=1) or (CONTROL = 0 and ACSF is set) Then
  If(SLOT = 2 or SLOT = 4) Then
    Call reaction wheel control
    Call reaction wheel command processing
  Else if(SLOT=3)
    Call magnetic control law
  Endif
Else if(CONTROL=2)
  If(SLOT = 2 or SLOT = 4)
    Call thruster control
  Endif
Endif
```

6.5 ACE/CPE MODES

6.5.1 SUN REFERENCE POINTING MODE CONTROL LAW

The SRPM uses the reaction wheels for control, pitch, and yaw attitude error data from the FSS; roll error data from the gyros; and rate information for all axes from the gyros to point the +X axis at the Sun during normal operations. In transitioning from an OBC-controlled mode to the ACE/CPE, the spacecraft first determines whether the solar arrays need to be indexed. If this is the case, the spacecraft is first put into an attitude hold mode using reaction wheels for control and gyros for attitude error and rate data. Once the spacecraft solar arrays are indexed, the spacecraft starts its Sun acquisition phase. The reaction wheels are used for control, the FSSs are used for pitch and yaw attitude error information, and the gyros are used for roll attitude error information and for rate information for all three axes. If the FSS registers no Sun, the spacecraft uses the CSSs for pitch and yaw attitude error information.

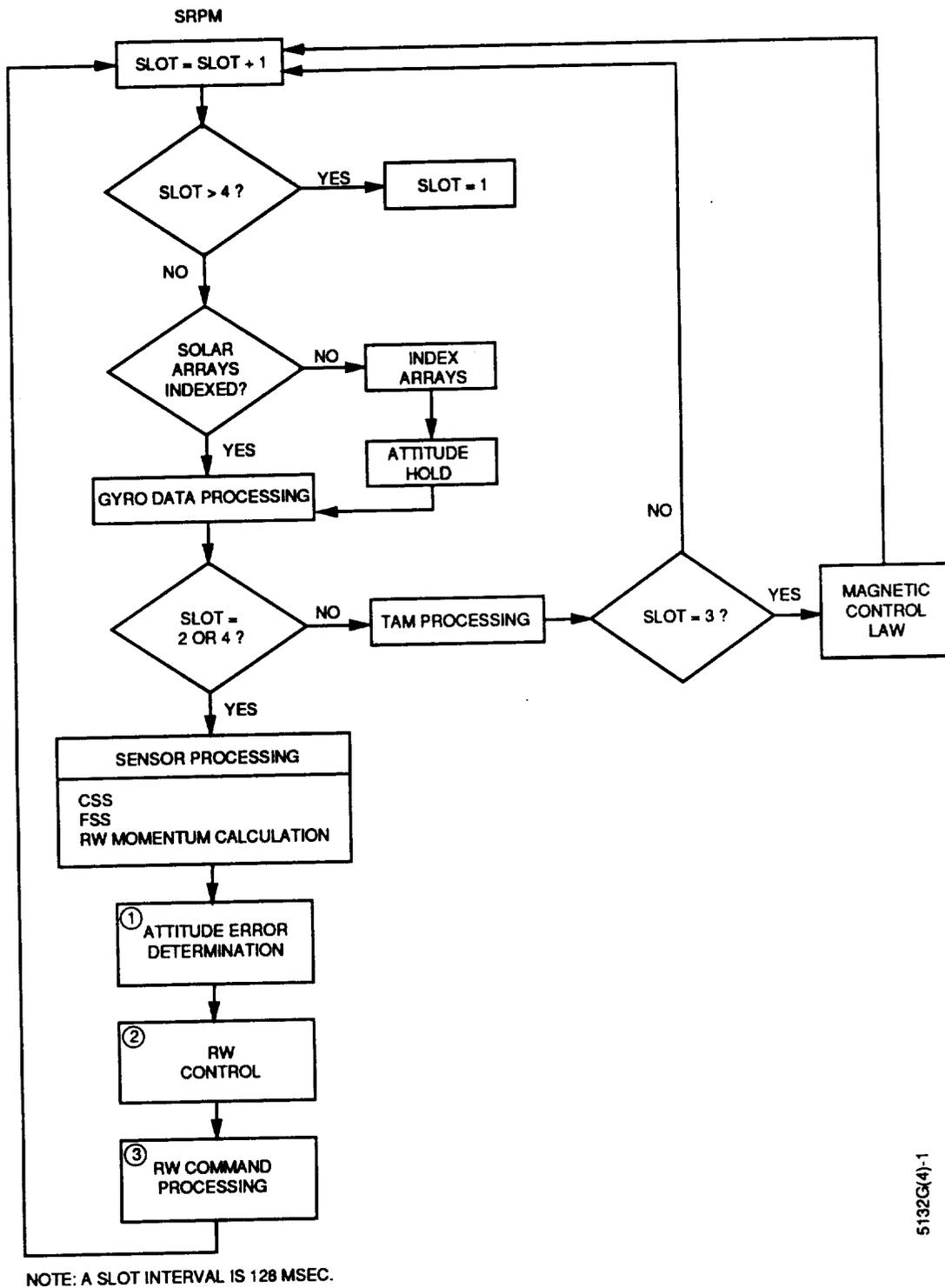
The SRPM has provisions for backup operation due to various failures. In case of an FSS failure, the ACE/CPE uses the CSSs for pitch and yaw attitude error information, provided that ground control personnel uplink an FSS failure flag. If a pitch or yaw gyro channel failed while the spacecraft

was in an OBC-controlled mode, the ACE has the ability to derive pitch and yaw rates by back-differencing FSS attitude error data (or CSS if there are no FSS data). In the case of roll gyro channel failure, the reaction wheel control function implements a control law that limits the roll rate by controlling the movement of the magnetometer Y- and Z-axis data. The maximum roll rate allowed by this control law is 0.3 deg/sec. In the case of no Sun in both the FSS and CSS FOVs, the ACE/CPE automatically places the spacecraft in an attitude hold mode using the gyros for attitude error and rate information and the reaction wheels for control. Figures 6-3 through 6-5 outline the various control loops used in the SRPM (Reference 15).

6.5.2 ROLL SCAN MODE CONTROL LAW

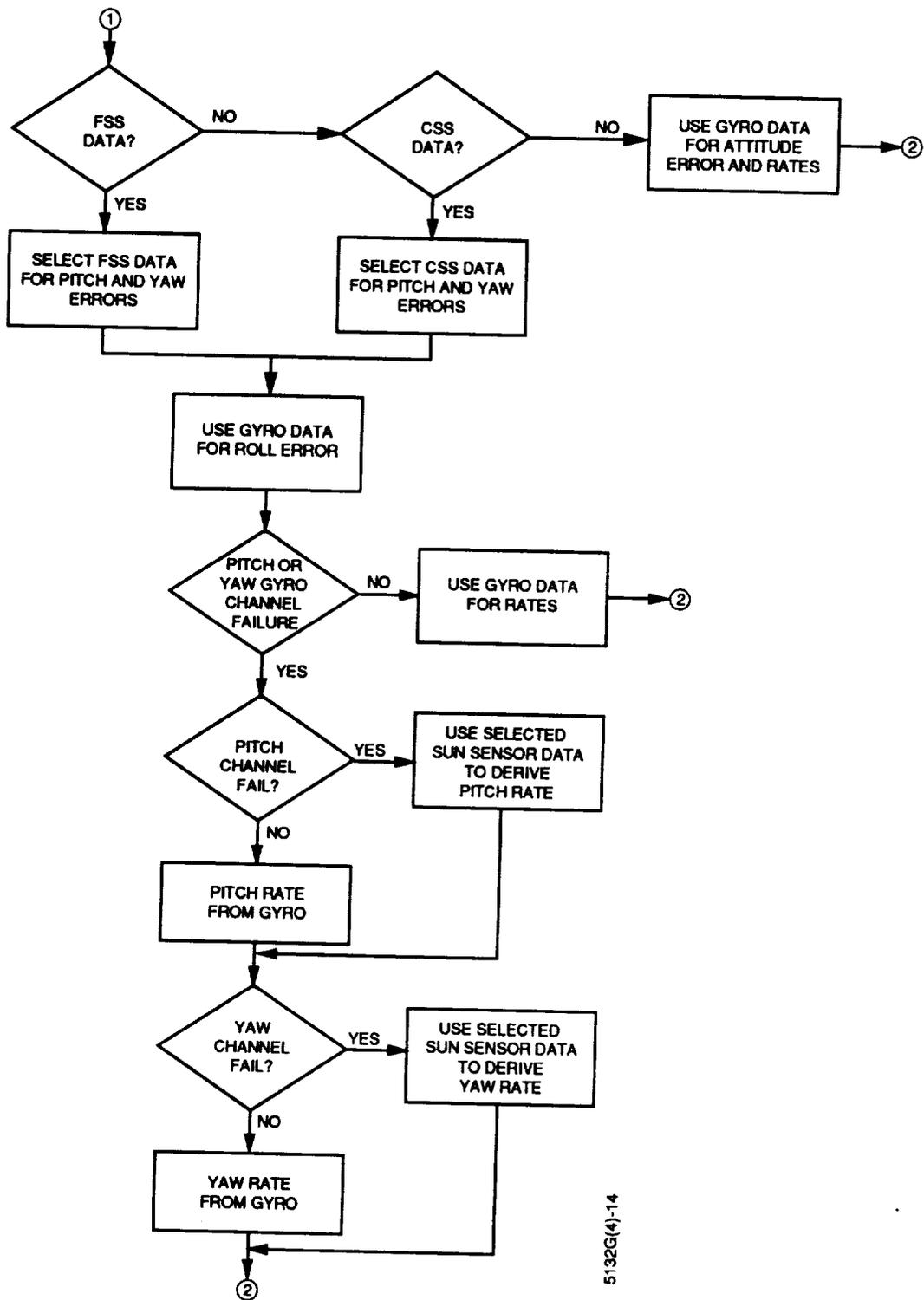
The roll scan mode (RSM) uses the reaction wheels for control, pitch and yaw error data from the FSSs, roll error data from the gyros, and rate information for all three axes from the gyros to point the +X axis at the Sun during normal operation. It also allows a rotation about the roll axis at a ground-selectable rate for star identification and attitude determination (References 15 and 16, p. 4-7). If the FSS registers no Sun, the spacecraft uses the CSSs for attitude error information.

The RSM has provisions for backup operation due to various failures. It is assumed that ground control personnel will have diagnosed all problems and set the proper flags, because this mode can only be reached by ground command. In the case of an FSS failure, the ACE/CPE uses the CSSs for attitude error information, provided that ground control personnel uplink an FSS failure flag. If a pitch or yaw gyro channel failure flag is set, the ACE has the ability to derive pitch and yaw rates by back-differencing FSS attitude error data (or CSS, if there are no FSS data). In the case



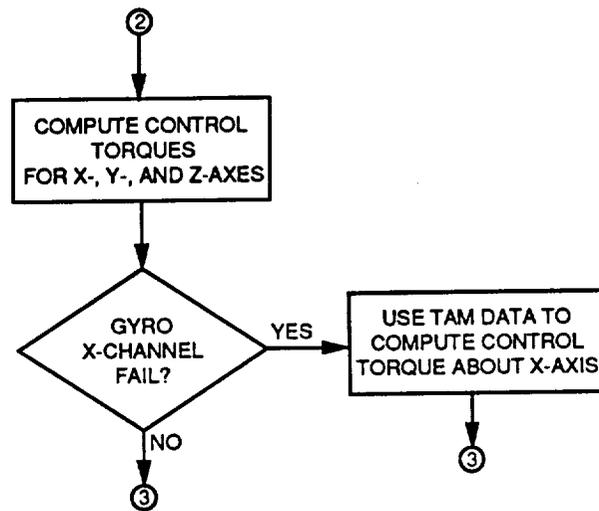
5132(4)-1

Figure 6-3. SRPM Control Flow



5132G(4)-14

Figure 6-4. SRPM Attitude Error Determination



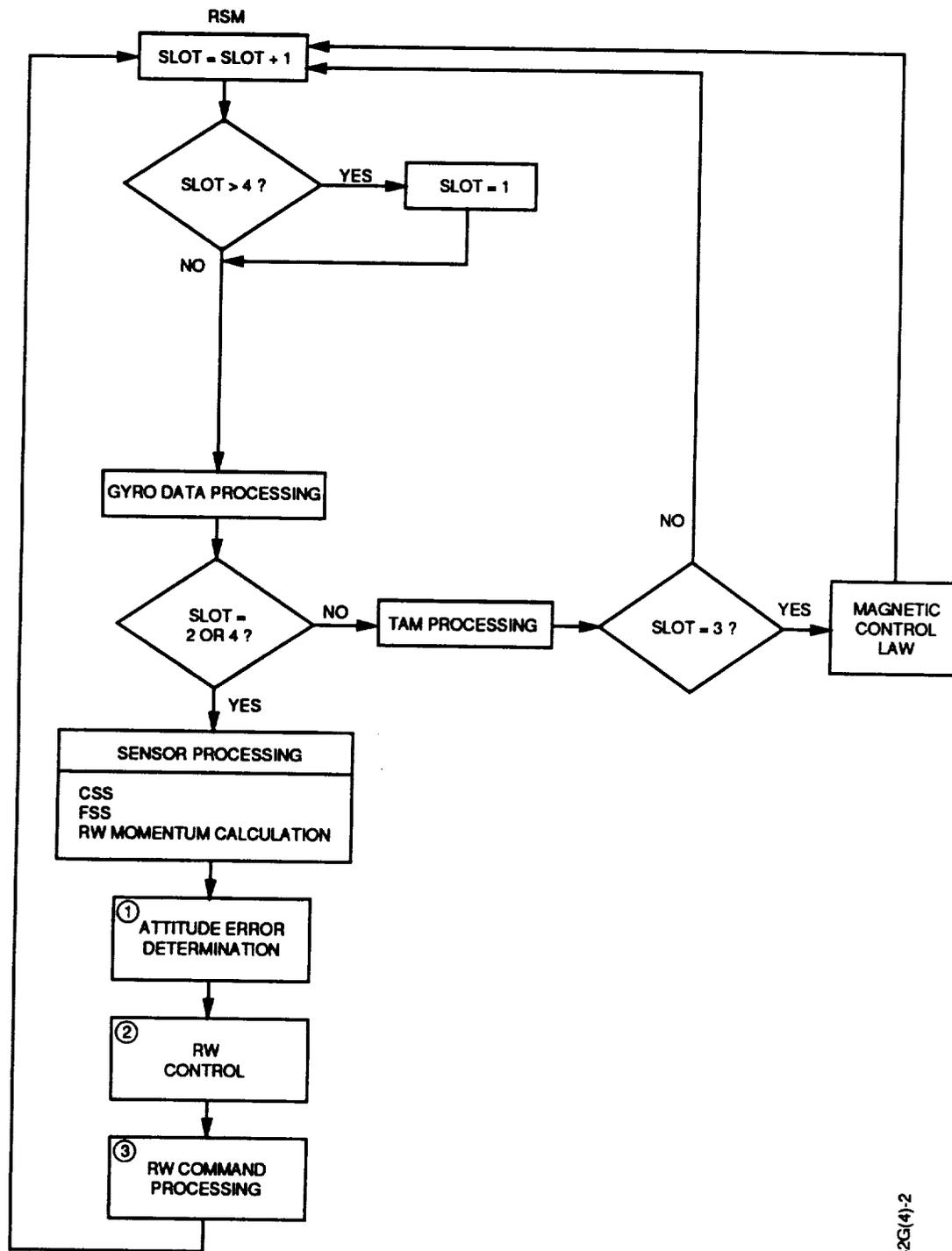
5132G(4)-9

Figure 6-5. SRPM Reaction Wheel Control

of roll channel gyro failure, ground control personnel would be required to set the roll channel selection flag to the proper roll channel so that the spacecraft could continue its roll rate. In the case of no Sun in both the FSS and CSS FOVs, the ACE/CPE automatically places the spacecraft in an attitude hold mode using the gyros for attitude error and rate information and the reaction wheels for control. The roll is not affected by the attitude hold. Figures 6-6 through 6-8 outline the various control loops for this mode.

6.5.3 REACTION WHEEL ATTITUDE HOLD MODE

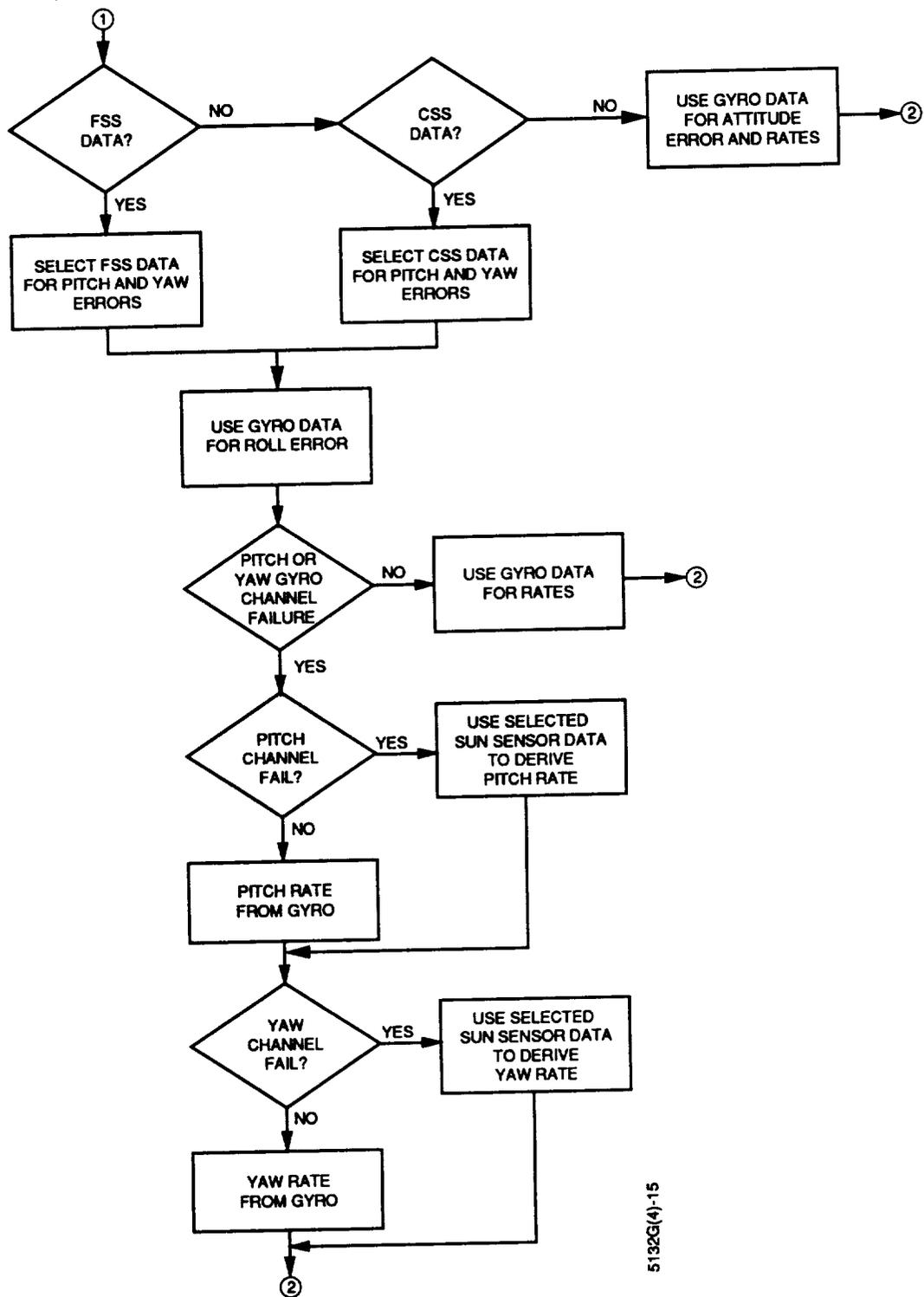
The reaction wheel attitude hold mode (RWAHM) uses the reaction wheels for control and the gyros for attitude error and rate data to hold the spacecraft attitude inertially fixed. Backup operation in this mode is limited because it is assumed that ground control personnel, who must command this mode, will identify and correct any problems before a transition to the RWAHM is initiated. If a roll gyro channel failure flag is set, the reaction wheel control function



NOTE: A SLOT INTERVAL IS 128 MSEC.

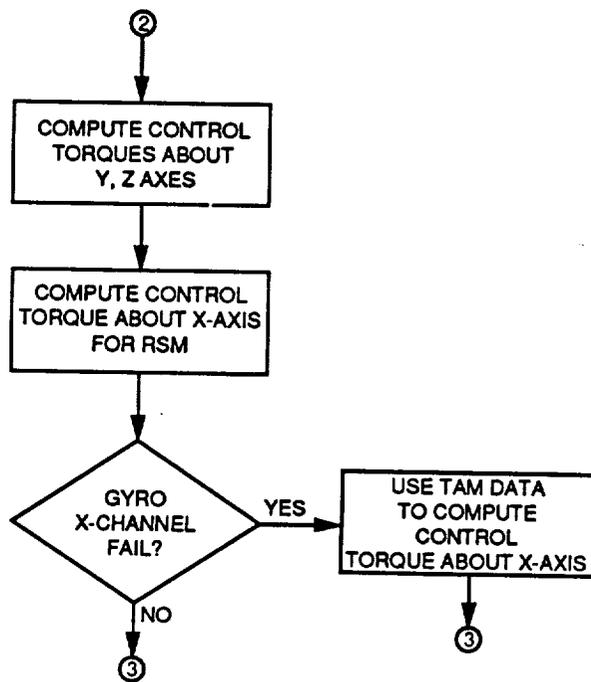
5132G(4)-2

Figure 6-6. RSM Control Flow



5132G(4)-15

Figure 6-7. RSM Attitude Error Determination



5132G(4)-12

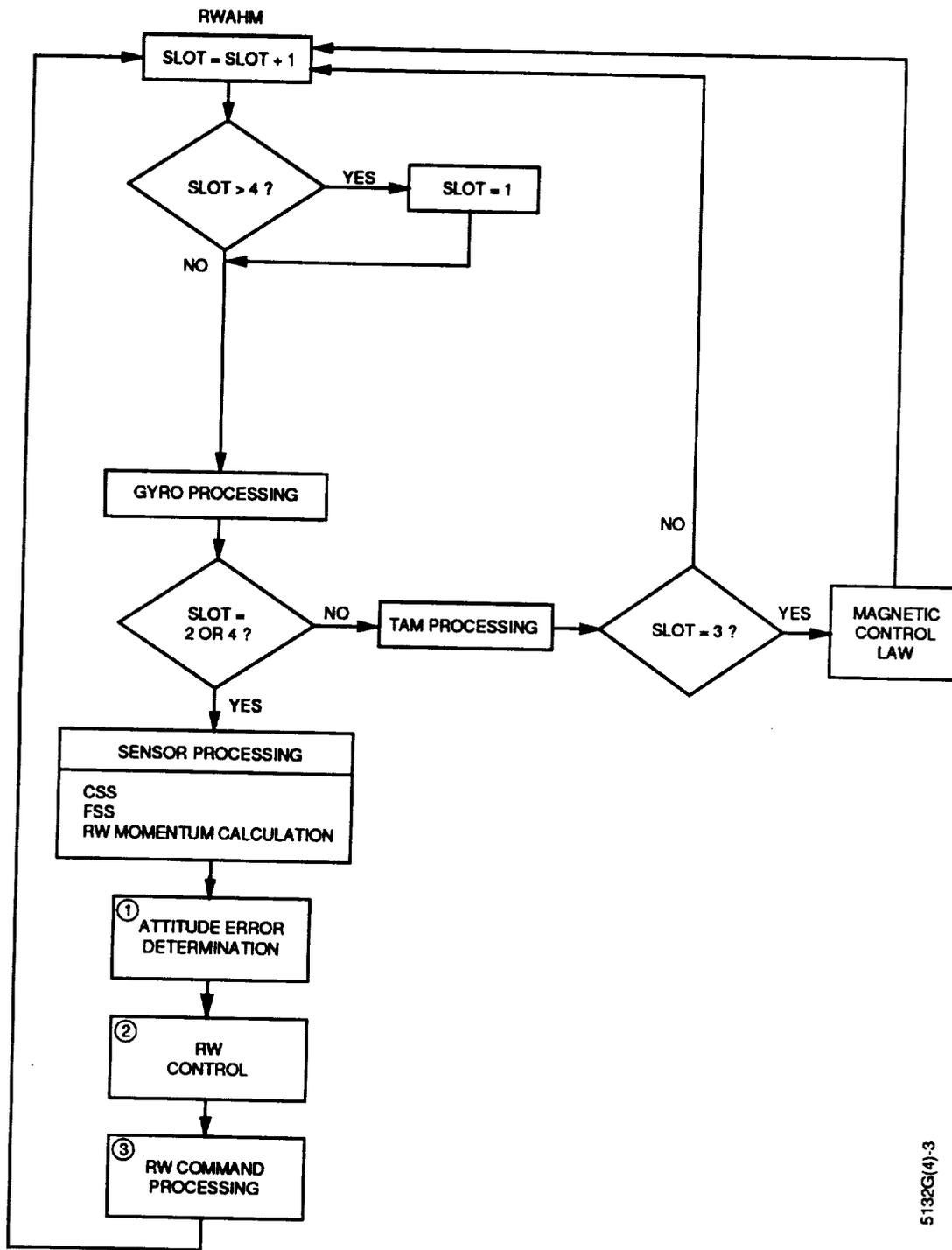
Figure 6-8. RSM Reaction Wheel Control

implements a control law that limits the roll rate by controlling the movement of the magnetometer Y- and Z-axis output. The minimum roll rate allowed by this control law is 0.3 deg/sec (Reference 15, pp. 7.4-27 to 7.4-30). Figures 6-9 through 6-11 outline the various control loops in the RWAHM.

6.5.4 SAFE HOLD MODE

The SHM uses the ACTs for control, pitch and yaw attitude error data from the FSS, roll error data from the gyros, and rate information for all axes from the gyros to point the +X axis at the Sun during normal operations. In transitioning from an OBC-controlled mode to the ACE/CPE, the spacecraft first determines whether the solar arrays need to be indexed. If this is the case, the spacecraft is first put into an attitude hold mode using the ACTs for control and the gyros for attitude error and rate data. Once the solar arrays are indexed, the spacecraft starts its Sun acquisition phase. The ACTs are used for control, the FSSs are used for pitch and yaw attitude error data, and the gyros are used for roll attitude error information and for rate information for all three axes. If the FSS registers no Sun, the spacecraft uses the CSSs for pitch and yaw attitude error information.

The SHM has provisions for backup operation due to various failures. In the case of FSS failure, the ACE/CPE uses the CSSs for pitch and yaw attitude error information, provided that ground control personnel uplink an FSS failure flag. If a pitch or yaw gyro channel failed while the spacecraft was in an OBC-controlled mode, the ACE has the ability to derive pitch and yaw rates by back-differencing FSS attitude error data (or CSS, if there are no FSS data). In the case of roll channel failure, the thruster control function implements a control law that limits the roll rate by



NOTE: A SLOT INTERVAL IS 128 MSEC.

5132G(4)-3

Figure 6-9. RWAHM Control Flow

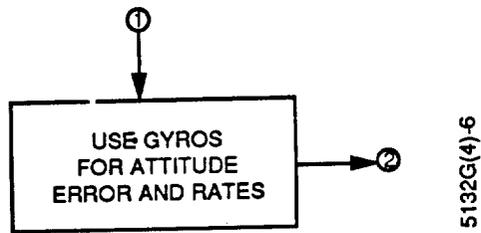


Figure 6-10. RWAHM Attitude Error Determination

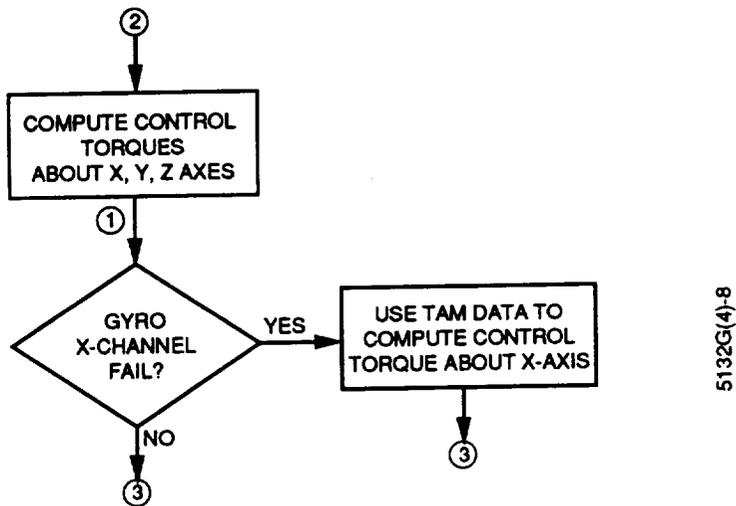


Figure 6-11. RWAHM Reaction Wheel Control

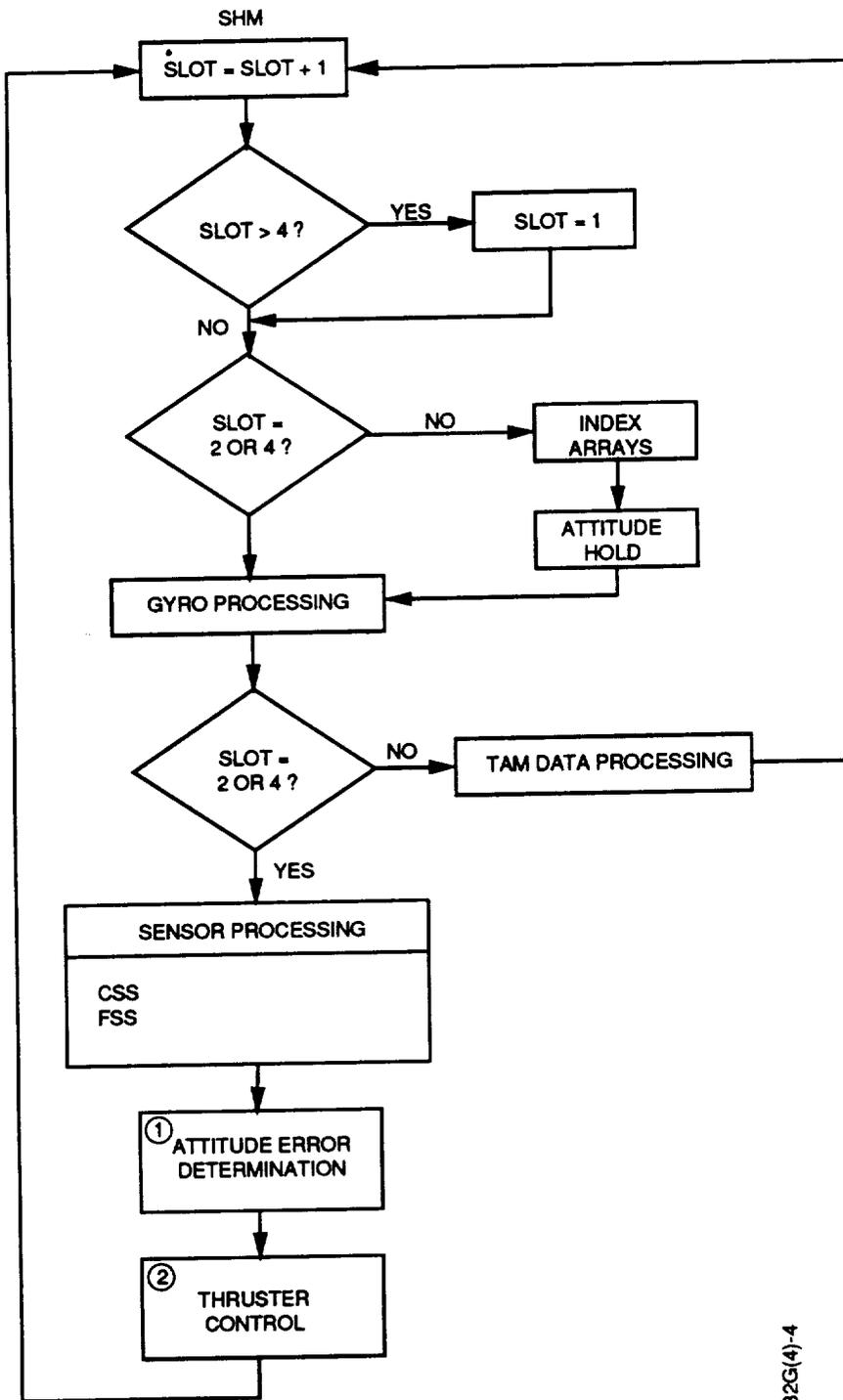
controlling the movement of the magnetometer Y- and Z-axis data. In the case of no Sun in both the FSS and CSS fields of view, the ACE/CPE automatically places the spacecraft in an attitude hold mode using the gyros for attitude error and rate information and the ACTs for control. Figures 6-12 through 6-14 outline the various control loops used in the SHM.

6.5.5 THRUSTER ATTITUDE HOLD MODE

The thruster attitude hold mode (TAHM) uses the ACTs for control and the gyros for attitude error and rate data to hold the spacecraft attitude inertially fixed. Backup operation in this mode is limited because it is assumed that ground control personnel, who must command this mode, will identify and correct any problems before a transition to the TAHM is initiated. If a roll gyro channel failure flag is set, the thruster control function implements a control law that limits the roll rate by controlling the movement of the magnetometer Y- and Z-axis output. Figures 6-15 through 6-17 outline the various control loops in the TAHM.

6.5.6 CONTINGENCY ORBIT MAINTENANCE MODE

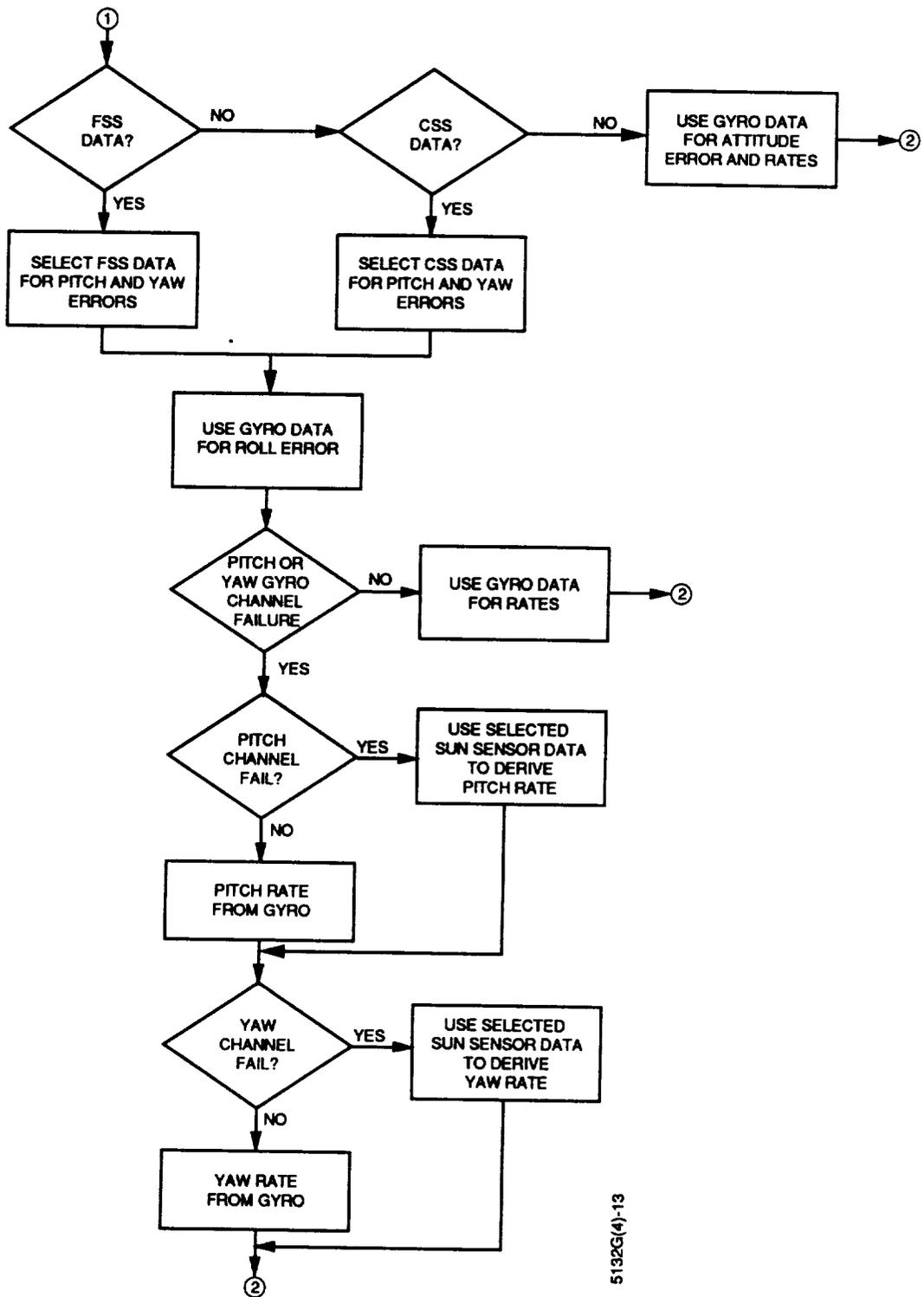
The contingency orbit maintenance mode (COMM) provides an orbit adjust capability to GRO while the spacecraft is under control of the ACE/CPE. This mode is a modified TAHM. In the TAHM, the ACTs are the actuators and the gyros provide attitude error and rate information. When the spacecraft is placed into the THM, the desired attitude is determined as well as the three-axis rotation needed to maneuver to the desired attitude. Each rotation is accomplished by uplinking a single-axis gyro error corresponding to the desired single-axis rotation. Once the spacecraft is in the desired attitude, ground control personnel uplink a command to fire one pair of OATs. Firing durations are selectable by ground control personnel. The OATs are not offpulsed for attitude



NOTE: A SLOT INTERVAL IS 128 MSEC.

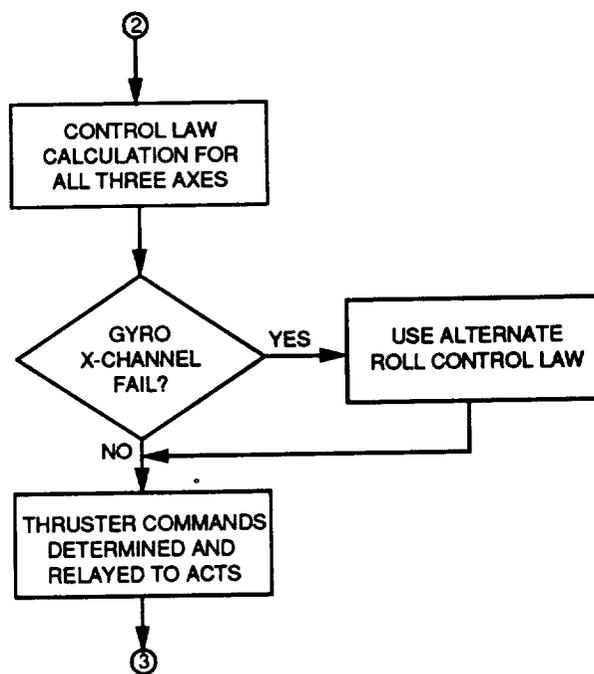
5132G(4)-4

Figure 6-12. SHM Control Flow



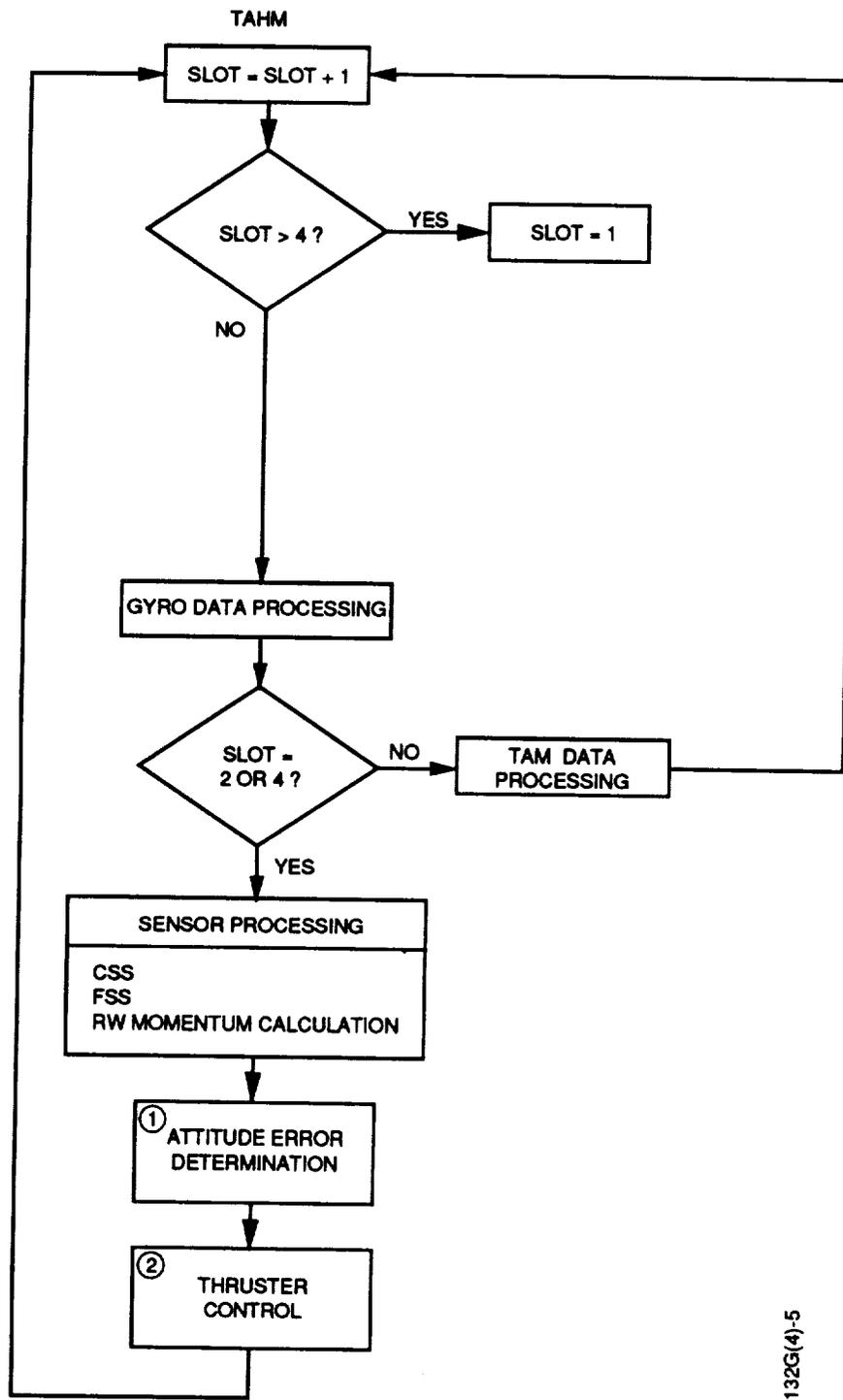
5132G(4)-13

Figure 6-13. SHM Attitude Error Determination



5132G(4)-10

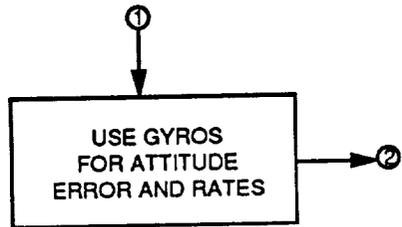
Figure 6-14. SHM Thruster Control



NOTE: A SLOT INTERVAL IS 128 MSEC.

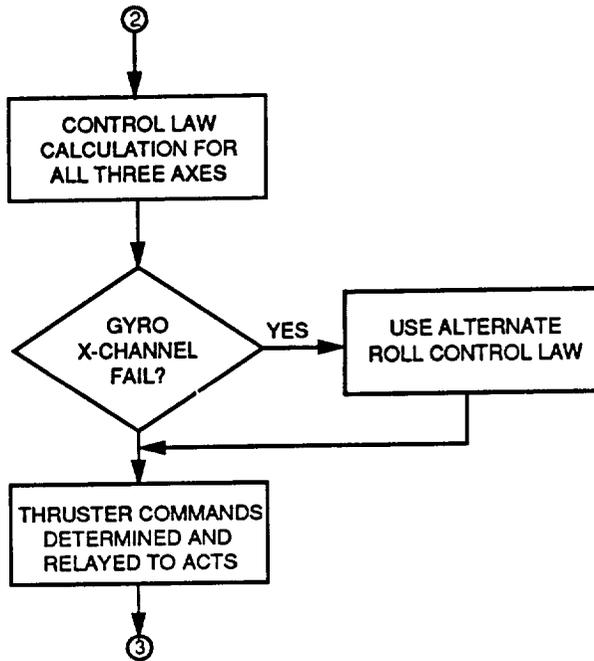
5132G(4)-5

Figure 6-15. TAHM Control Flow



5132G(4)-7

Figure 6-16. TAHM Attitude Error Determination



5132G(4)-11

Figure 6-17. TAHM Thruster Control

control. The ACTs provide all attitude control, and the gyros continue to provide attitude rate and error information. A 1-rpo rate capability exists. This would be accomplished by continually uplinking different GREF data base constants (Reference 15, pp. 7.4-4, 7.10-1 to 7.10-3). The different GREFs would change at a 1-rpo rate.¹

6.5.7 THRUSTER COMMAND MODE

The thruster command mode (TCM) provides the ability to fire any ACT or pair of ACTs in real time by external command. The firings are in impulses of fixed duration, selectable by ground control personnel. Two pulse durations are available (Reference 16): 256 msec and 2048 msec. The resolution of the pulses will be ± 1.0 percent. Thruster commands will be processed and executed one at a time up to a rate that produces continuous thruster operation. For this and higher command frequencies, the result will be continuous thruster operation, according to TRW (Reference 17, pp. 24, 25, 57). The TCM is not a mode per se but an ability to command the ACTs externally.

6.6 MOMENTUM MANAGEMENT

Momentum management is the process by which the spacecraft dumps the reaction wheel stored angular momentum while under control of the ACE/CPE. Momentum management is performed every 512 msec using data from the gyros, Sun sensors, TAMs, reaction wheel control function, and magnetic control law. The torquer bars produce a magnetic moment M that, acting

¹Personal communication, Terry Watson (TRW) at GSFC, August 1987.

with the Earth's magnetic field, opposes the excess angular momentum from the reaction wheels:

$$\vec{T} = \vec{M} \times \vec{B} \quad \vec{T} = d \vec{H}/dt$$

where \vec{T} = torque vector

\vec{M} = magnetic moment vector

\vec{B} = Earth's magnetic field vector

\vec{H} = angular momentum vector (reaction wheels)

$\vec{M}'_{\text{body}} = \vec{B} \times \vec{H}$; \vec{M}'_{body} parallel to \vec{M}

$\vec{M}'_{\text{bars}} = [AMB] \vec{M}'_{\text{body}}$

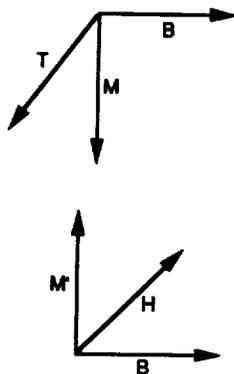
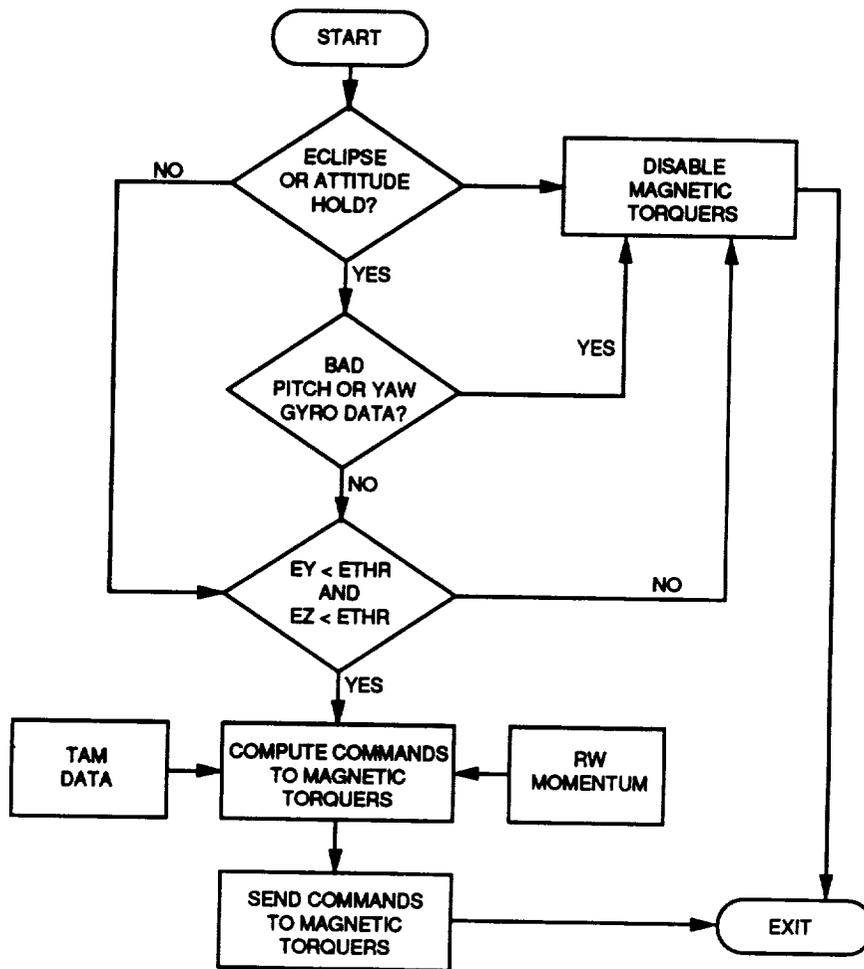
$\vec{M}_{\text{bars}} = -k \vec{M}'_{\text{bars}}$

where [AMB] = transformation matrix from body to bars coordinates

k = gauss-to-counts conversion factor

\vec{M}_{bars} = commands to the torquer bars

Figure 6-18 illustrates the momentum management function.



B = EARTH'S MAGNETIC FIELD
 M = MAGNETIC MOMENT VECTOR FROM TORQUER BARS
 H = REACTION WHEEL ANGULAR MOMENTUM VECTOR
 M' = INTERMEDIATE MAGNETIC MOMENT VECTOR

5132G(4)-16

Figure 6-18. Momentum Management (CPE)

APPENDIX A - COORDINATE SYSTEMS

A.1 GEOCENTRIC INERTIAL COORDINATE SYSTEM (X_G, Y_G, Z_G)

The origin of the geocentric inertial (GCI) coordinate system is at the geocenter, with the $+X_I$ axis in the equatorial plane pointed to the vernal equinox, $+Z_I$ along the Earth spin axis pointed North, and Y_I completing the righthand set.

A.2 BODY REFERENCE SYSTEMS

A.2.1 BODY COORDINATES (X_B, Y_B, Z_B)

The GRO body coordinate system (BCS) is centered on the center of mass of the observatory. The $+Z_B$ axis is normal to the plane of the trunnions and on the payload side of the spacecraft, the $+Y_B$ axis is parallel to the solar array drive axis, and the $+X_B$ axis completes a righthand orthogonal set and is on the Sun sensor side of the vehicle. Figure A-1 illustrates the GRO BCS.

A.2.2 SPACECRAFT COORDINATES (X, Y, Z)

Spacecraft coordinates are defined with the $+Z$ axis normal to the plane of the trunnions, located at the center line of the trunnions, and on the payload side of the observatory. The $+Y$ axis is parallel to the solar array drive axis, and the $+X$ axis forms a right-hand set on the Sun sensor side of the observatory. The center of the coordinates is at the intersection of the Z -axis with the plane of the trunnions.

A.2.3 ACAD REFERENCE COORDINATES (X_0, Y_0, Z_0)

Attitude control and determination (ACAD) reference coordinates are defined by the master reference cube installed on the optical bench. All attitude determination and pointing requirements are defined in terms of, or with reference to, this coordinate system. The ACAD reference system is

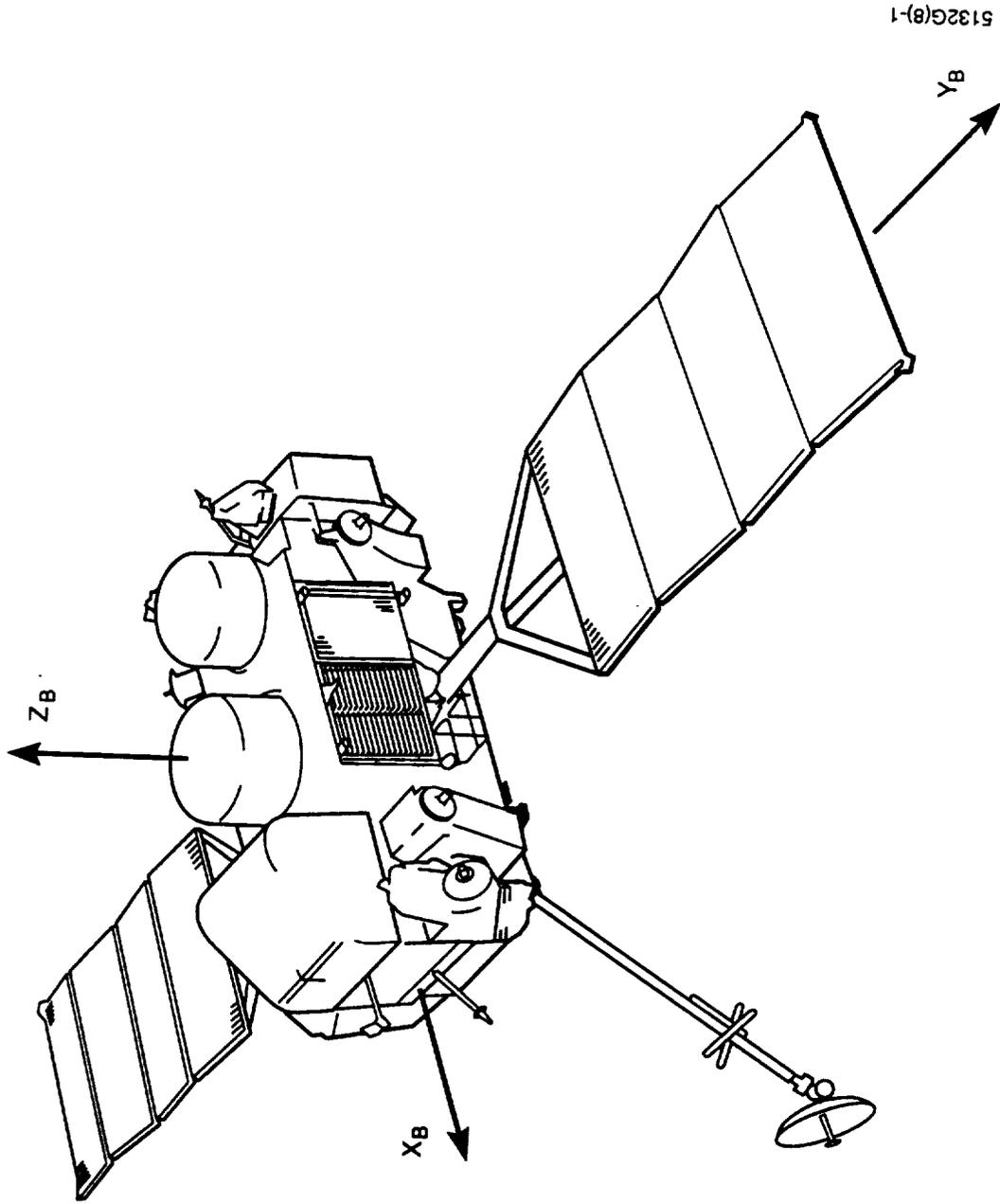


Figure A-1. GRO Spacecraft and Body Coordinate System

approximately parallel to the spacecraft coordinate system (within alignment tolerances).

A.3 ATTITUDE REFERENCE COORDINATE SYSTEM (X_R, Y_R, Z_R)

The attitude reference coordinate system is defined such that the origin is the spacecraft center of mass. The $+X_R$ axis is in the orbit plane pointed in the direction of spacecraft orbital motion, the $+Z_R$ axis is pointed to the geocenter, and the $+Y_R$ axis is normal to the orbit plane to complete the righthand set.

A.4 REACTION WHEEL ASSEMBLY COORDINATE SYSTEM ($X_{RWA}, Y_{RWA}, Z_{RWA}$)

The reaction wheel assembly (RWA) coordinate system is depicted in Figure A-2.

A.5 REACTION WHEEL COORDINATE SYSTEM (X_{RW}, Y_{RW}, Z_{RW})

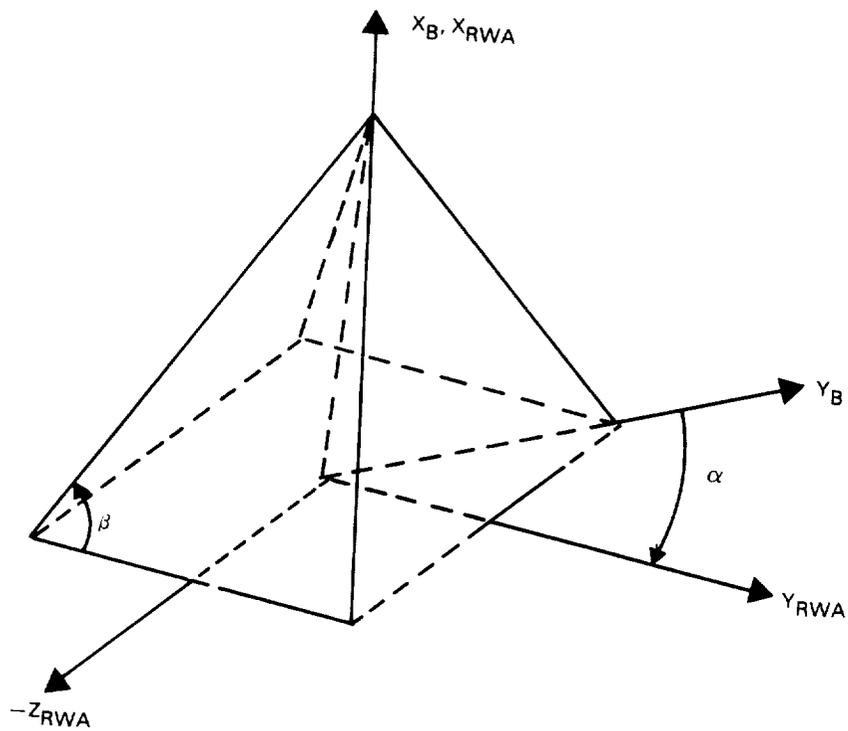
The reaction wheel (RW) coordinate system is defined such that the origin is centered on a face of the pyramidal assembly, the $+X_{RW}$ axis is directed along the pyramidal face pointed toward the apex, the $+Z_{RW}$ axis is directed normal to the pyramidal face, and the $+Y_{RW}$ axis is defined to complete the righthand set.

A.6 MAGNETIC TORQUER COORDINATE SYSTEM (X_{MT}, Y_{MT}, Z_{MT})

The magnetic torquer coordinate system is defined by a 2-1-3 Euler rotation relative to spacecraft body coordinates. The Euler angles are defined by θ about the $+Y_B$ axis, ϕ about the $+X_B$ axis, and ψ about the $+Z_B$ axis. Figure A-3 shows the Euler angle rotations.

A.7 INERTIAL REFERENCE UNIT INPUT AXES

The inertial reference unit (IRU) input axes are aligned with spacecraft body coordinates.



9789/85

Figure A-2. RWA Coordinates Relative to Spacecraft Body Coordinates ($\alpha = 45^\circ$, $\beta = 61^\circ$)

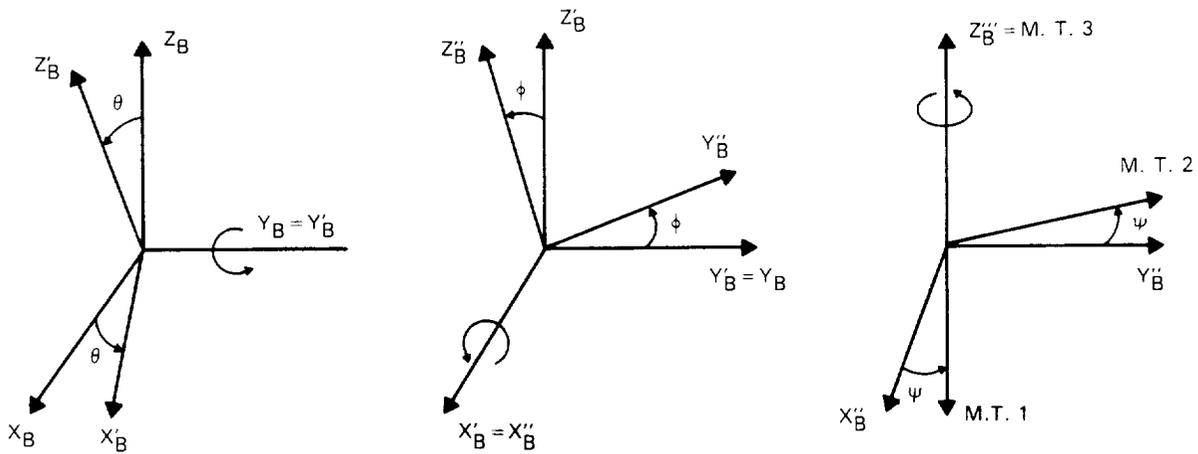


Figure A-3. Euler Rotations Defining Magnetic Torque Orientation for the Baseline Configuration $\Theta = 48.1842$, $\phi = 45.0$, and $\psi = 0.0$

9788 85

A.8 FIXED-HEAD STAR TRACKER COORDINATE SYSTEMS

Figure A-4 illustrates the FHST coordinate systems.

A.8.1 FHST NUMBER 1 (FHST1)

The FHST1 coordinate system is defined by the axes X_{T1} , Y_{T1} , and Z_{T1} , with the Z_{T1} axis along the boresight. The $+Z_{T1}$ axis is in the X_B - Y_B plane 45 deg from the $-X_B$ axis and 45 deg from the $+Y_B$ axis. The $+Y_{T1}$ axis is in the $+Z_B$ axis direction. The X_{T1} axis completes the righthand set.

A.8.2 FHST NUMBER 2 (FHST2)

The FHST2 coordinate system is defined by the axes X_{T2} , Y_{T2} , and Z_{T2} , with the Z_{T2} axis along the boresight. The $+Z_{T2}$ axis is in the X_B - Y_B plane 45 deg from the $-X_B$ axis and 45 deg from the $-Y_B$ axis. The $+Y_{T2}$ axis is in the $-Z_B$ axis direction. The X_{T2} axis completes the righthand set.

A.9 FINE SUN SENSOR COORDINATE SYSTEMS

Figures A-5 and A-6 illustrate the FSS coordinate systems.

A.9.1 FSS NUMBER 1 (FSS1)

The FSS1 coordinate system is defined by the axes X_{F1} , Y_{F1} , and Z_{F1} , with the Z_{F1} axis along the boresight. The $+Z_{F1}$ axis is in the X_B - Z_B plane 60 deg from the $+X_B$ axis and 30 deg from the $+Z_B$ axis. The $+X_{F1}$ axis is in the $+Y_B$ axis direction. The Y_{F1} axis completes the righthand set.

A.9.2 FSS NUMBER 2 (FSS2)

The FSS2 coordinate system is defined by the axes X_{F2} , Y_{F2} , and Z_{F2} , with the Z_{F2} axis along the boresight. The $+Z_{F2}$ axis is in the X_B - Z_B plane 2 deg from the $+X_B$ axis and 88 deg from the $-Z_B$ axis. The $+X_{F2}$ axis is in the $+Y_B$ axis direction. The Y_{F2} axis completes the righthand set.

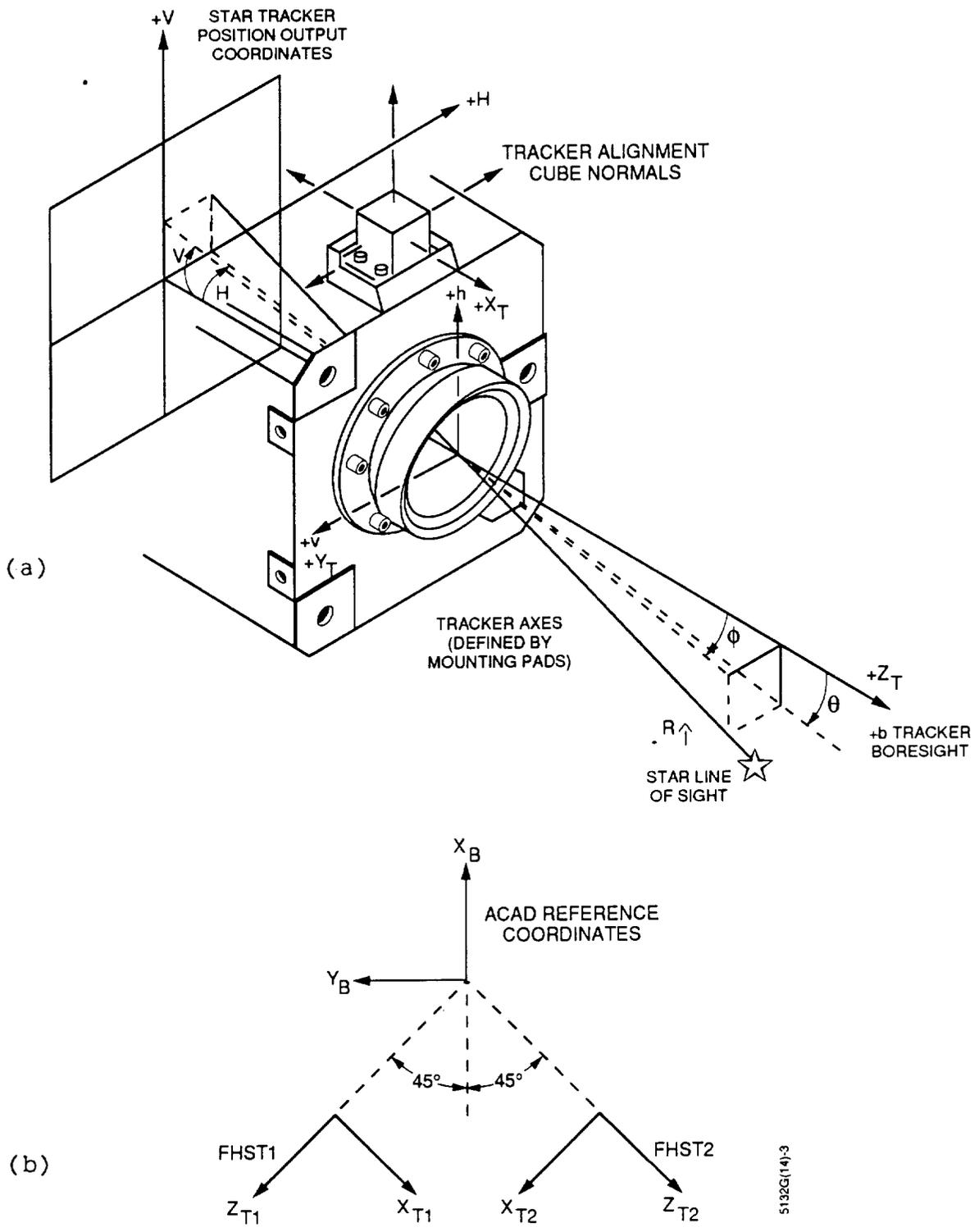


Figure A-4. (a) FHST Coordinate Systems and (b) FHST Orientation Relative to Spacecraft Body Coordinates

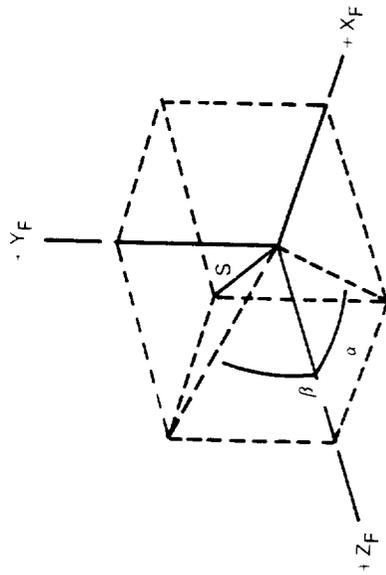
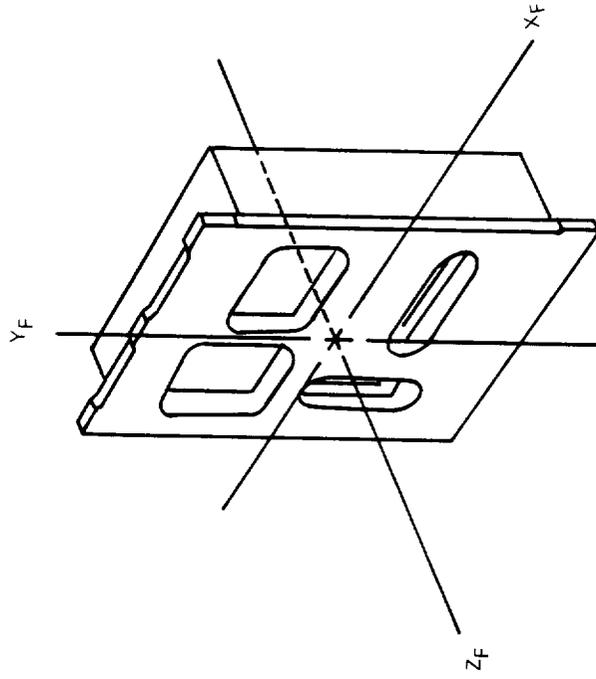
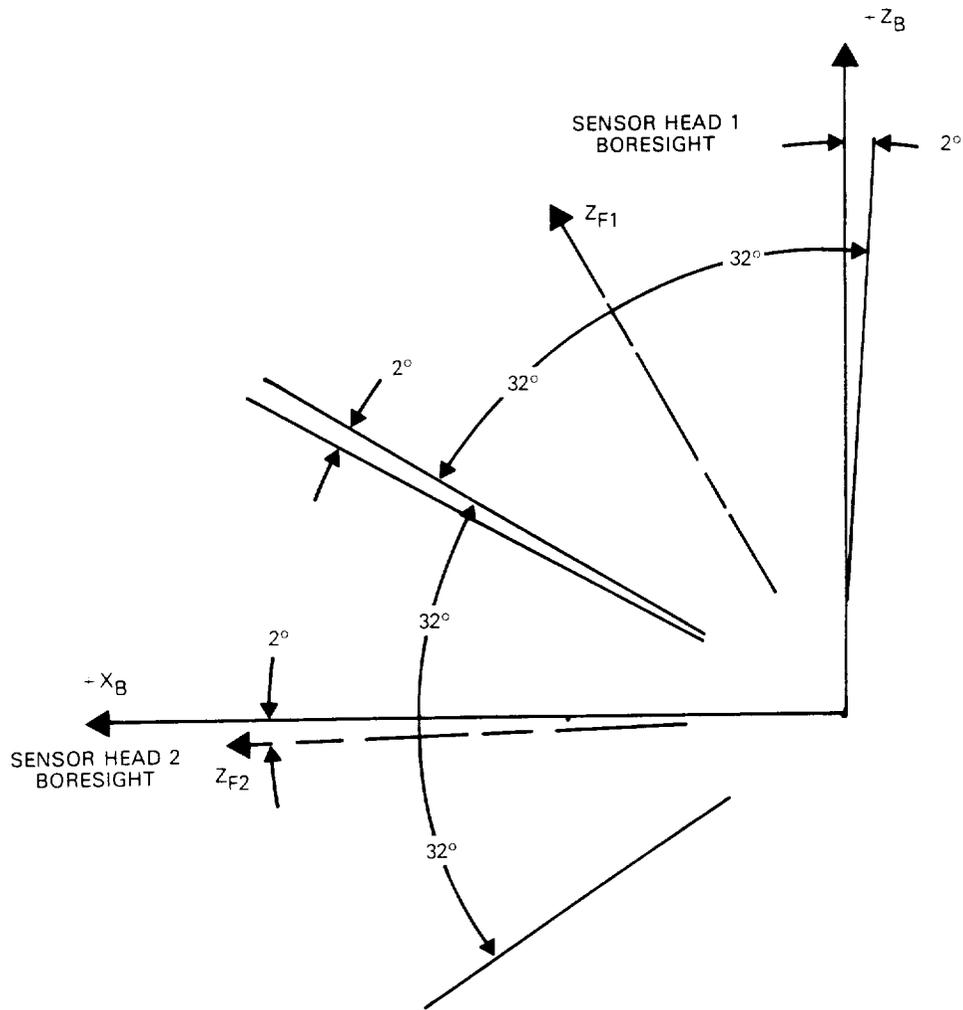


Figure A-5. Fine Sun Sensor Coordinate Frame and Sun Angle Component Definition



9789 85

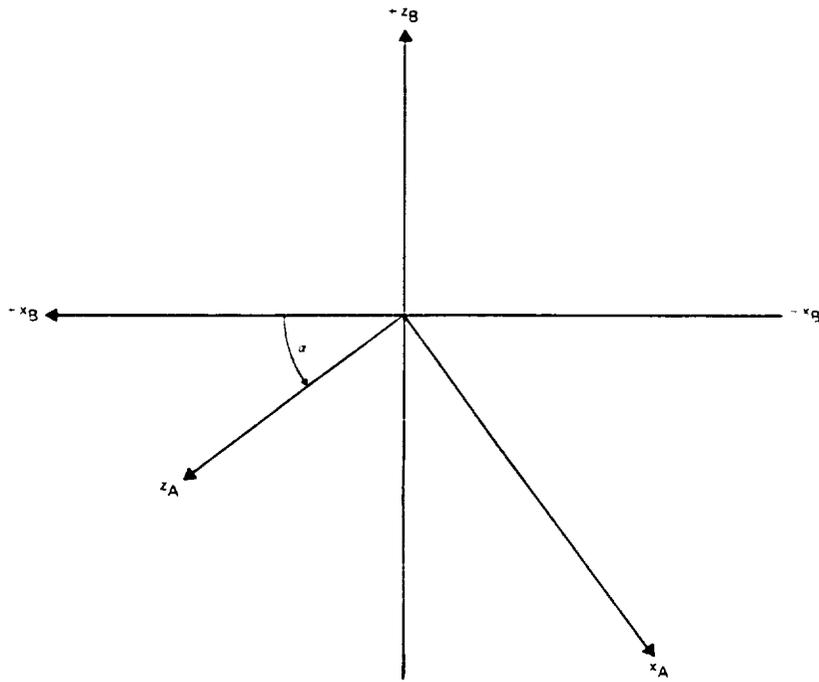
Figure A-6. FSS Orientation Relative to Spacecraft Body Coordinates

A.10 HIGH-GAIN ANTENNA REFERENCE SYSTEM (X_A, Y_A, Z_A)

The high-gain antenna (HGA) reference system is defined in Figure A-7, where it is specified to be a 49.7423-deg rotation about the spacecraft body pitch axis (Y_B).

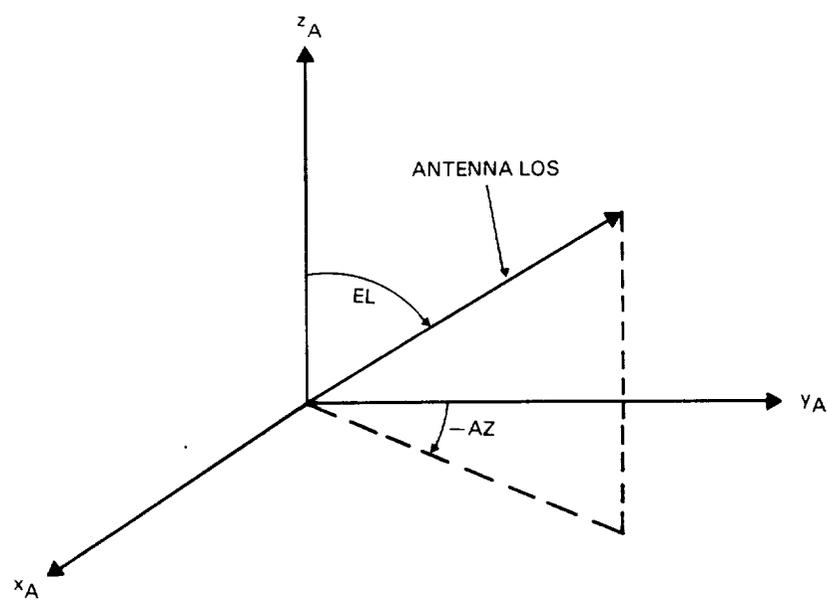
A.11 HIGH-GAIN ANTENNA LINE OF SIGHT

The HGA line of sight (LOS) is illustrated in Figure A-8, where the elevation angle (EL) and the azimuth angle (AZ) are defined.



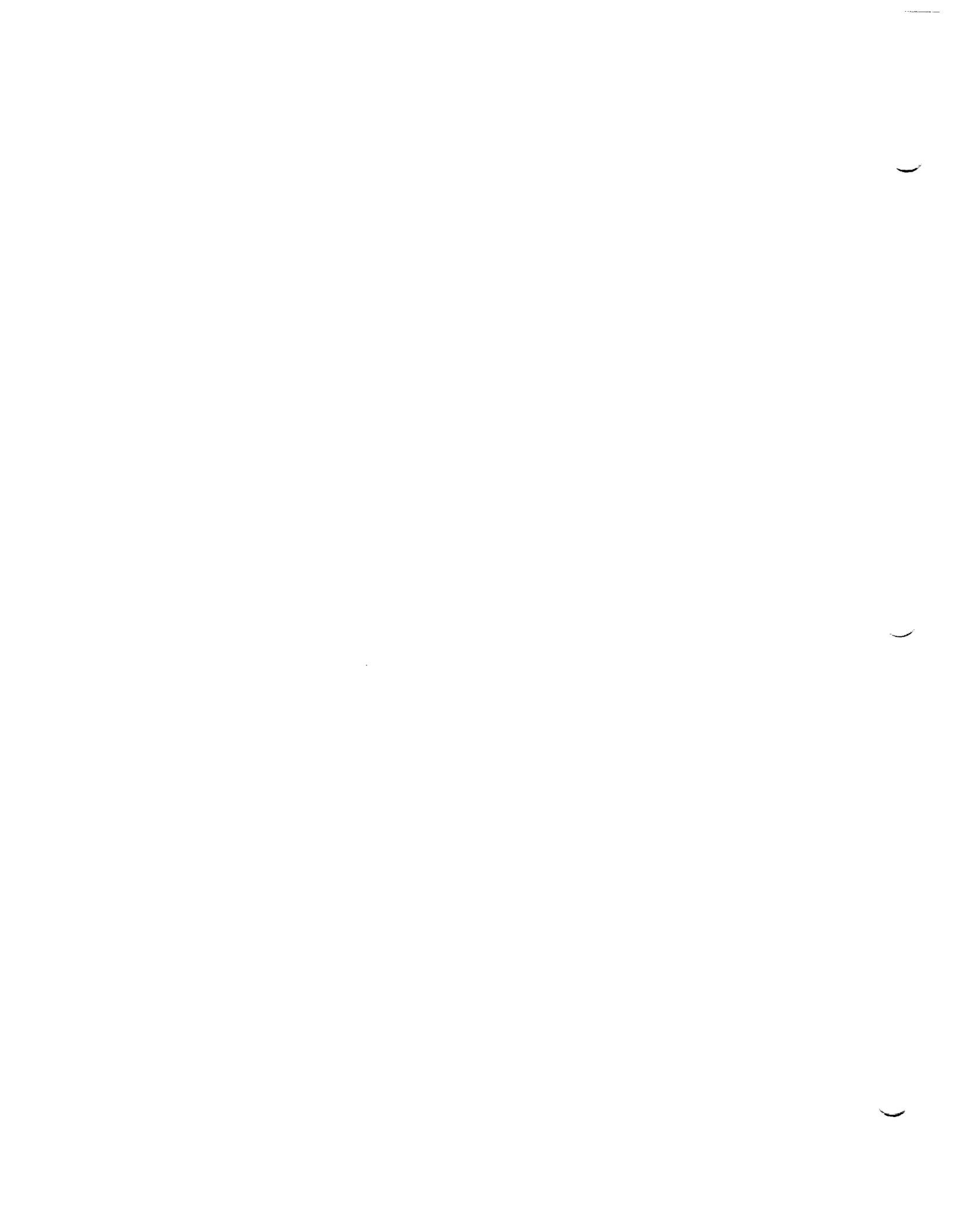
9789 85

Figure A-7. Spacecraft-to-Antenna Reference System



9789 85

Figure A-8. Antenna Reference System to Antenna LOS



APPENDIX B - GRO SURFACE AREA MODEL

The surface area model for the GRO spacecraft is composed of flat plates and cylinders. The model is represented in Figure B-1 and described in Table B-1.

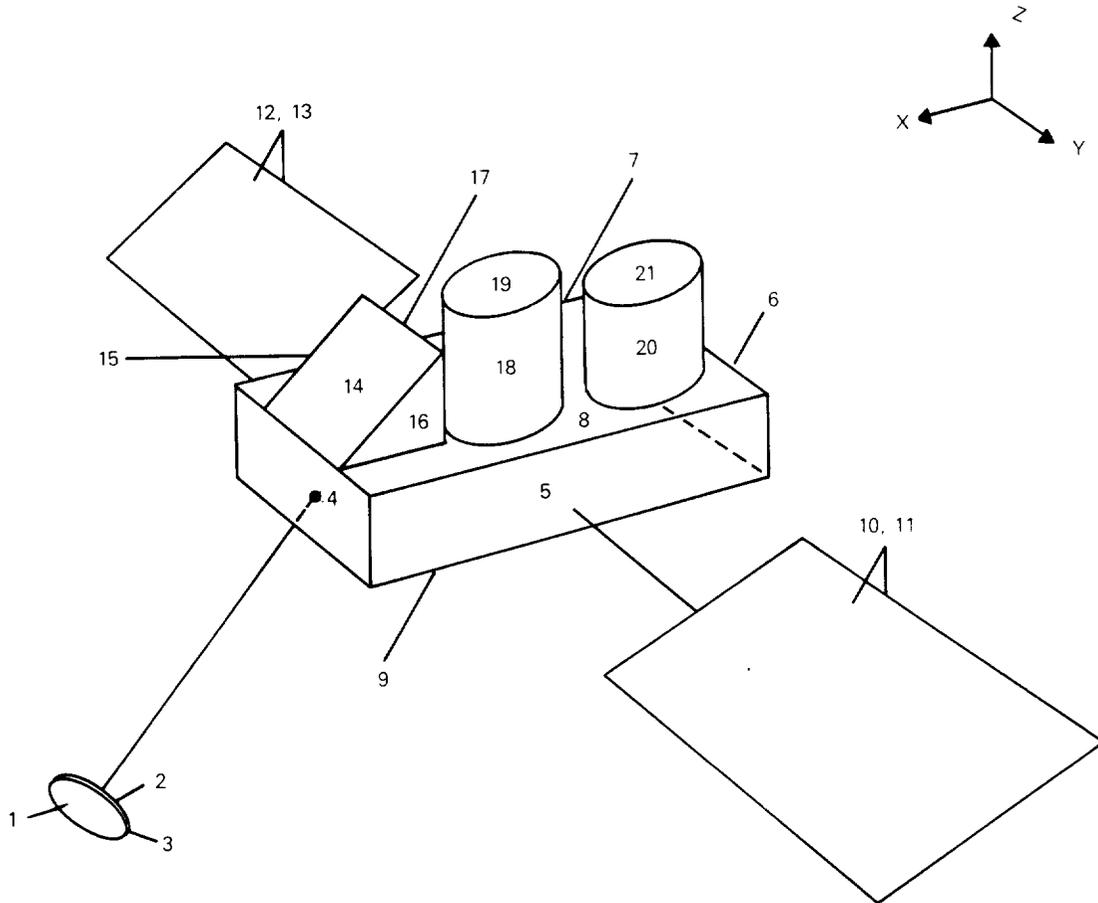


Figure B-1. GRO Surface Area Model

9788 82

Table B-1. Parameters for GRO Surface Area Model

NO.	SURFACE ELEMENT	AREA (M ²)	SURFACE NORMAL			CENTROID OF AREA (M)		
			X _B	Y _B	Z _B	X _B	Y _B	Z _B
1	HGA	1.81	SIN @	0.0	-COS @	1.71 + 5.98SINa	0.0	-1.07 - 5.98SINa
2	HGA	1.81	-SIN @	0.0	COS @	1.71 + 5.77SINa	0.0	-1.07 - 5.77SINa
3	HGA	1.02	-	-	-	1.71 + 5.87SINa	0.0	-1.07 - 5.87SINa
4	SC (+X)	7.79	1.0	0.0	0.0	3.27	0.0	0.0
5	SC (+Y)	13.97	0.0	1.0	0.0	0.0	1.82	0.0
6	SC (-X)	7.79	-1.0	0.0	0.0	-3.27	0.0	0.0
7	SC (-Y)	13.97	0.0	-1.0	0.0	0.0	1.82	0.0
8	SC (+Z)	15.27	0.0	0.0	1.0	0.0	0.0	1.07
9	SC (-Z)	23.77	0.0	0.0	-1.0	0.0	0.0	-1.07
10	SA1 (ACT)	19.00	COS ω1	0.0	SIN ω1	0.0	7.82	0.0
11	SA1 (PAS)	19.00	-COS ω1	0.0	-SIN ω1	0.0	7.82	0.0
12	SA2 (ACT)	19.00	COS ω2	0.0	SIN ω2	0.0	-7.82	0.0
13	SA2 (PAS)	19.00	-COS ω2	0.0	-SIN ω2	0.0	-7.82	0.0
14	OSSE (SLANTED)	4.82	0.66	0.0	0.75	2.41	0.0	1.82
15	OSSE (-Y)	1.28	0.0	1.0	0.0	2.13	1.07	1.57
16	OSSE (+Y)	1.28	0.0	1.0	0.0	2.13	1.07	1.57
17	OSSE (-X)	3.21	-1.0	0.0	0.0	1.56	0.0	1.82
18	COMPTTEL	6.96	-	-	-	0.0	0.0	1.69
19	COMPTTEL (+Z)	2.54	0.0	0.0	1.0	0.0	0.0	2.30
20	EGRET	4.62	-	-	-	-2.41	0.0	1.50
21	EGRET (+Z)	2.30	0.0	0.0	1.0	-2.41	0.0	1.93

8799 1961 886

B DENOTES SPACECRAFT COORDINATE SYSTEM
 @ INDICATES ANGLE FORMED BY HGA BOOM AND Z - BODY AXIS
 ω1, ω2 INDICATES ANGLE BETWEEN ACTIVE SIDE SURFACE NORMAL AND X+ BODY AXIS FOR SA1, SA2, RESPECTIVELY

APPENDIX C - UTILITY AND INITIALIZATION SUBROUTINES

This appendix contains algorithm descriptions of standard subroutine packages that will be used in the dynamics simulator. Section C.1 describes utilities CEULER, EULERC, DATE, DCROSS, DGMPRD, DGMTRA, DUNVEC, GAUSS, JD, INVERT, ORTHO, ROTMAT, ROTMT4, TCON20, and TCON40. Section C.2 describes the FASTOX orbit propagator and the Sun and Moon position routine, SMPOS.

C.1 UTILITY SUBROUTINE DESCRIPTIONS

C.1.1 CEULER

Subroutine CEULER converts a direction cosine matrix (or any orthogonal matrix) to a set of Euler symmetric parameters. The direction cosine matrix can be expressed in terms of the Euler symmetric parameters as follows (Reference 11):

$$[C] = \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix} \quad (C-1)$$

If the attitude matrix [C] is known, the Euler symmetric parameters q_1 , q_2 , q_3 , and q_4 can be solved for in terms of the components of [C].

Four distinct sets of equations can be derived from Equation (C-1). All are mathematically equivalent. However, numerical inaccuracy can be minimized by avoiding calculations in which the Euler symmetric parameter appearing in the denominator is close to zero. This is the criterion used for choosing a representation.

Subroutine CEULER performs the following:

1. If $1 + C_{11} + C_{22} + C_{33} > \epsilon$, * Equations (C-2a) through (C-2d) are used:

$$q_4 = \frac{1}{2} \left(1 + C_{11} + C_{22} + C_{33} \right)^{1/2} \quad (\text{C-2a})$$

$$q_1 = \frac{1}{4q_4} (C_{23} - C_{32}) \quad (\text{C-2b})$$

$$q_2 = \frac{1}{4q_4} (C_{31} - C_{13}) \quad (\text{C-2c})$$

$$q_3 = \frac{1}{4q_4} (C_{12} - C_{21}) \quad (\text{C-2d})$$

2. If $1 + C_{11} - C_{22} - C_{33} > \epsilon$, Equations (C-3a) through (C-3d) are used:

$$q_1 = \frac{1}{2} \left(1 + C_{11} - C_{22} - C_{33} \right)^{1/2} \quad (\text{C-3a})$$

$$q_2 = \frac{1}{4q_1} (C_{12} + C_{21}) \quad (\text{C-3b})$$

$$q_3 = \frac{1}{4q_1} (C_{31} + C_{13}) \quad (\text{C-3c})$$

$$q_4 = \frac{1}{4q_1} (C_{23} - C_{32}) \quad (\text{C-3d})$$

* ϵ is the tolerance for choosing a representation.

3. If $1 - C_{11} + C_{22} - C_{33} > \epsilon$, Equations (C-4a) through (C-4d) are used:

$$q_2 = \frac{1}{2} \left(1 - C_{11} + C_{22} - C_{33} \right)^{1/2} \quad (\text{C-4a})$$

$$q_1 = \frac{1}{4q_2} (C_{12} + C_{21}) \quad (\text{C-4b})$$

$$q_3 = \frac{1}{4q_2} (C_{23} + C_{32}) \quad (\text{C-4c})$$

$$q_4 = \frac{1}{4q_2} (C_{31} - C_{13}) \quad (\text{C-4d})$$

4. If tests 1 through 3 fail, Equations (C-5a) through (C-5d) are used:

$$q_3 = \frac{1}{2} \left(1 - C_{11} - C_{22} + C_{33} \right)^{1/2} \quad (\text{C-5a})$$

$$q_1 = \frac{1}{4q_3} (C_{31} + C_{13}) \quad (\text{C-5b})$$

$$q_2 = \frac{1}{4q_3} (C_{23} + C_{32}) \quad (\text{C-5c})$$

$$q_4 = \frac{1}{4q_3} (C_{12} - C_{21}) \quad (\text{C-5d})$$

C.1.2 EULERC

Subroutine EULERC computes the direction cosine matrix that corresponds to a set of Euler symmetric parameters. If q_1 ,

q_2 , q_3 , and q_4 are the four Euler symmetric parameters, the direction cosine matrix is given by Reference 11:

$$[C] = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}$$

$$= \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 - q_4^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix}$$

The matrix [C] may be normalized at the user's discretion by

$$[C]_{\text{NORMALIZED}} = [C_{ij}/E_{\text{norm}}]_{i,j}$$

where $i = 1-3$

$j = 1-3$

$$E_{\text{norm}} = q_1^2 + q_2^2 + q_3^2 + q_4^2$$

In theory,

$$\sum_{i=1}^4 q_i^2 \equiv 1$$

However, q_1 , q_2 , q_3 , and q_4 are obtained by integrating the equations of motion. Errors caused by approximating the state equations or by roundoff may cause numerical inaccuracies such that

$$\sum_{i=1}^4 q_i^2 \neq 1$$

C.1.3 DATE

Subroutine DATE converts a given Julian day into the corresponding calendar date.

C.1.4 DCROSS

Subroutine DCROSS computes the cross-product of two vectors, A and B, as $\vec{C} = \vec{A} \times \vec{B}$:

$$\begin{aligned}\vec{A} &= (a_1, a_2, a_3)^T \\ \vec{B} &= (b_1, b_2, b_3)^T \\ \vec{C} &= \begin{bmatrix} C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}\end{aligned}$$

C.1.5 DGMPRD

Subroutine DGMPRD computes the product of two input matrices [A] and [B] to form [C] = [A][B]. DGMPRD uses the fact that matrices are stored in columns as a linear array when forming the product [C].

C.1.6 DGMTRA

Subroutine DGMTRA computes the transpose of an input matrix A:

$$[A] = [a_{ij}]$$

$$[A]^T = [a_{ji}]$$

where i is 1, ..., m and j is 1, ..., n.

C.1.7 DUNVEC

Subroutine DUNVEC takes an input vector, \vec{V} , and returns a unit vector coincident with the input vector and the magnitude of the input vector.

C.1.8 GAUSS

Subroutine GAUSS generates a normally distributed random number, V , with a given mean, μ , and standard deviation, σ .

V is computed as follows. A random number, X , with a uniform distribution on $(0, 1)$ is generated by subroutine RANDU. A table of values $(Y_1, Y_2, \dots, Y_{201})$ describing a uniform distribution is available to subroutine GAUSS. The J th element is chosen from this table, where $J = 200 \cdot X + 1.5$. (J is converted to an integer such that $1 \leq J \leq 201$.)

$$\begin{aligned} V &= Y_J \cdot \sigma + \mu && \text{if } I^* \geq 0 \\ &= -Y_J \sigma + \mu && I^* < 0 \end{aligned}$$

where I^* is an integer calculated by subroutine RANDU and Y_J is the element chosen from the normal distribution table.

C.1.9 JD

Subroutine JD converts a calendar day to the corresponding Julian day (Reference 11).

C.1.10 INVERT

Subroutine INVERT computes the inverse of a matrix and solves the set of simultaneous linear equations $[A][X] = [C]$. INVERT computes $[A]^{-1}$ and the solution $[X] = [A]^{-1}[C]$.

C.1.11 ORTHO

Subroutine ORTHO constructs a 3 by 3 orthogonal matrix that rotates vectors from coordinate system S' to coordinate system S. The user specifies two vectors \hat{u} and \vec{v} that define the S frame, where only \hat{u} must be a unit vector. The user must also specify the order of the rotations. The S' axes are defined by the unit vectors \hat{i} , \hat{j} , and \hat{k} , where \hat{i} is along \hat{u} , \hat{j} is along $\vec{v} - (\hat{u} \cdot \vec{v})\hat{u}$, and \hat{k} is orthogonal to \hat{i} and \hat{j} .

The matrix [A] that transforms vectors from S' to S is constructed as follows. The column vectors of [A] are

$$\begin{bmatrix} A_{1,i} \\ A_{2,i} \\ A_{3,i} \end{bmatrix} = u$$

where i is 1-3

$$\begin{bmatrix} A_{1,j} \\ A_{2,j} \\ A_{3,j} \end{bmatrix} = \hat{u} \times \frac{\vec{v} \times \hat{u}}{|\vec{v} \times \hat{u}|}$$

where j is 1-3

$$\begin{bmatrix} A_{1,k} \\ A_{2,k} \\ A_{3,k} \end{bmatrix} = \pm(\vec{v} \times \hat{u})$$

where k is 1-3.

(The + or - sign is chosen so that the resulting coordinate system is right handed.) It should be noted that $i \neq j \neq k$.

C.1.12 ROTMAT AND ROTMT4*

Subroutine ROTMAT constructs a coordinate transformation matrix, given an ordered set of rotation angles and the axis about which the rotations are to be performed. A simple rotation about the Z-axis by an angle ϕ is illustrated in Figure C-1. The direction cosine matrix for this rotation is given by $A_3(\phi)$.

$$A_3(\phi) = \begin{bmatrix} \cos \phi & \sin \phi & 0 \\ -\sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

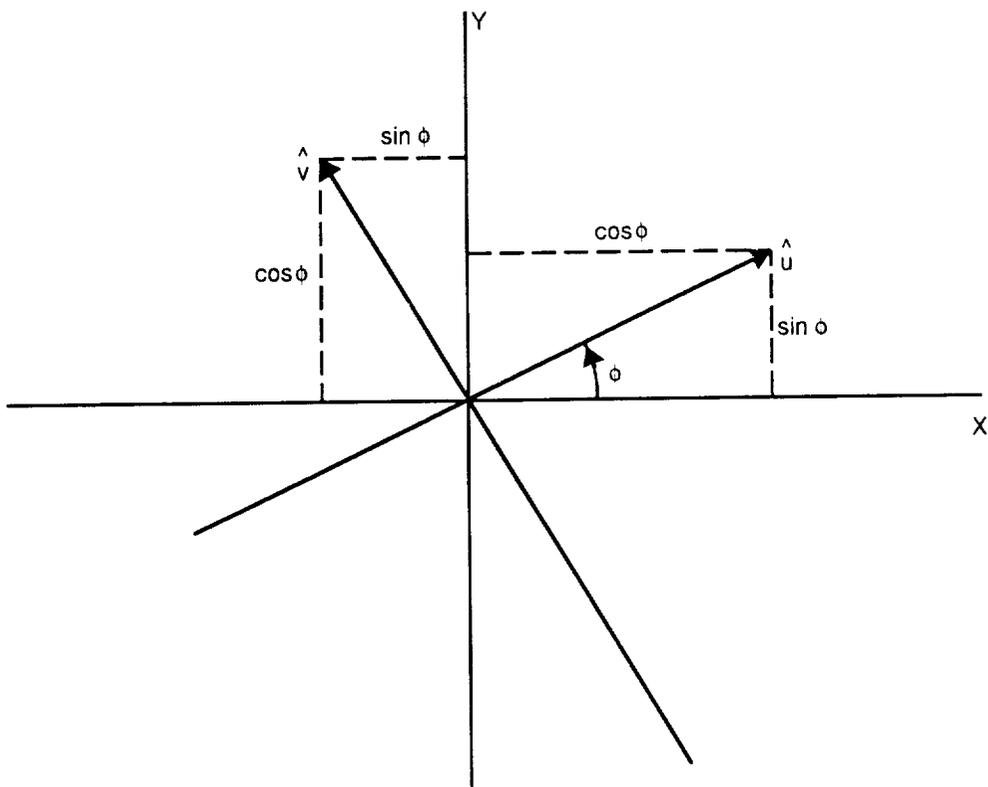
The direction cosine matrix for a rotation of an angle ϕ about the X-axis is given by $A_1(\phi)$:

$$A_1(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix}$$

The direction cosine matrix for a rotation of angle ϕ about the Y-axis is given by $A_2(\phi)$:

$$A_2(\phi) = \begin{bmatrix} \cos \phi & 0 & -\sin \phi \\ 0 & 1 & 0 \\ \sin \phi & 0 & \cos \phi \end{bmatrix}$$

*ROTMAT is used for double-precision computations, whereas ROTMT4 is used for single-precision computations.



5132G(5-6)-07

Figure C-1. Rotation About the Z-Axis by the Angle ϕ

If the X-, Y-, and Z-axes are denoted as the 1, 2, and 3 axes of rotation, the coordinate transformation matrix [T] can be computed, given the ordered set of rotation angles:

$$[T] = A_i(\phi) A_j(\phi) A_k(\phi)$$

where i, j, k is any permutation of the axis of rotation (1, 2, 3) as determined by the order in which the rotations are to be performed.

C.1.13 TCON20

Subroutine TCON20 converts time in seconds since 0 hours universal time (UT) September 1, 1957, to a calendar format (YYMMDD.HHMMSSMMM).¹

C.1.14 TCON40

Subroutine TCON40 converts a given date in a calendar format¹ to seconds since 0 hours UT September 1, 1957.

C.2 EPHEMERIS UTILITIES

C.2.1 FASTOX

Routine FASTOX is a two-body orbit generator that computes the position and velocity of a spacecraft given the Keplerian elements, the time from epoch, and the gravitational constant of the central body. The Keplerian elements include the semimajor axis, a; the eccentricity, e; the inclination, i; the right ascension of the ascending node, Ω ; the argument of perigee, ω ; and the mean anomaly at epoch, M_0 . A user option indicates if Earth perturbations are to be included in the calculations and if the velocity is to be computed.

¹Calendar format: Year-month-day-hour-minutes-second-milliseconds.

Using Kepler's equation to determine the position and velocity, the mean anomaly, M , at some time, t , is related to the eccentric anomaly, E , at the same time by

$$M = E - e \sin E \quad (C-6)$$

If t_0 is the epoch time of the elements, the mean anomaly at time t is found from the mean anomaly at epoch by

$$M = M_0 + n(t - t_0) \quad (C-7)$$

where $n \equiv 2\pi/\text{period}$ is the mean motion. FASTOX solves Kepler's equation numerically to find E at any time before or after the epoch using an iterative solution to Equation (C-6), obtained using Newton's method. Successive estimates of E are given by

$$E_i = E_{i-1} + \frac{M + e \sin (E_{i-1}) - E_{i-1}}{1 - e \cos (E_{i-1})} \quad (C-8)$$

where the starting value $E_0 = M$. If $M = 0$, the solution is trivial and therefore implies that $E = 0$. The iteration continues until either the correction term is $< 1 \times 10^{-7}$ or until 75 iterations have occurred. Once the eccentric anomaly is found, the true anomaly, v , and the distance, r , may be found from

$$\sin v = \frac{(1 - e^2)^{1/2} \sin E}{1 - e \cos E} \quad (C-9)$$

$$\cos v = \frac{\cos (E) - e}{1 - e \cos E} \quad (C-10)$$

$$r = a(1 - e \cos E) \quad (C-11)$$

Equations (C-9) through (C-11) thus give the position of the spacecraft in the orbit plane. The orientation of the orbit in space must be considered to find the position relative to an inertial system. Using spherical triangles,

$$x = r[\cos(\omega + \nu) \cos \Omega - \sin(\omega + \nu) \sin \Omega \cos i] \quad (C-12)$$

$$y = r[\cos(\omega + \nu) \sin \Omega + \sin(\omega + \nu) \cos \Omega \sin i] \quad (C-13)$$

$$z = r[\sin(\omega + \nu) \sin i] \quad (C-14)$$

where x , y , and z are resolved in the coordinate system in which the elements are defined. It is possible to use Equations (C-9) and (C-10) to remove the explicit true anomaly dependence in Equations (C-12) through (C-14) and therefore to compute the position directly.

The velocity at time t is found by applying the extended chain rule to Equations (C-12) through (C-14). Thus, for example,

$$\frac{dx}{dt} = \frac{\partial x}{\partial r} \frac{\partial r}{\partial E} \frac{\partial E}{\partial t} + \frac{\partial x}{\partial \nu} \frac{\partial \nu}{\partial E} \frac{\partial E}{\partial t} \quad (C-15)$$

assuming ω , Ω , and i are constant. The quantity dE/dt may be found by differentiating Kepler's equation to obtain

$$\frac{dM}{dt} = \frac{dE}{dt} - e \cos E \frac{dE}{dt} \quad (C-16)$$

so that

$$\frac{dE}{dt} = \frac{1}{1 - e \cos E} \frac{dM}{dt} \quad (C-17)$$

The equations for the velocity are as follows:

$$\frac{dx}{dt} = \left[\frac{na}{r} \quad a(1 - e^2)^{1/2} \quad k_1 \cos E - ak_2 \sin E \right] \quad (C-18)$$

$$\frac{dy}{dt} = \left[\frac{na}{r} \quad a(1 - e^2)^{1/2} \quad \ell_1 \cos E - a\ell_2 \sin E \right] \quad (C-19)$$

$$\frac{dz}{dt} = \left[\frac{na}{r} \quad a(1 - e^2)^{1/2} \quad m_1 \cos E - am_2 \sin E \right] \quad (C-20)$$

where

$$k_1 = -\cos \Omega \sin \omega - \sin \Omega \cos \omega \cos i$$

$$k_2 = \cos \Omega \cos \omega - \sin \Omega \sin \omega \cos i$$

$$\ell_1 = -\sin \Omega \sin \omega + \cos \Omega \cos \omega \cos i$$

$$\ell_2 = \sin \Omega \cos \omega + \cos \Omega \sin \omega \cos i$$

$$m_1 = \cos \omega \sin i$$

$$m_2 = \sin \omega \sin i$$

The above procedure is not coordinate-system dependent, i.e., the position and velocity will be in whatever coordinate system the elements of the orbit are defined.

C.2.2 SMPOS

Subroutine SMPOS computes the positions and distances of the Sun and Moon relative to the Earth. This method requires no input other than the time for which these quantities are desired and gives solar position errors up to about 0.01 deg and lunar position errors up to about 0.25 deg.

C.2.2.1 Solar Ephemeris

C.2.2.1.1 Mean Motion of the Sun

The mean motion of the Sun is given by Reference 18 as

$$L = 279.0696678 + 0.9856473354(d) + 2.267 \times 10^{-13}(d^2)$$

$$g = 358.0475845 + 0.985600267(d) - 1.12 \times 10^{-13}(d^2) - 7 \times 10^{-20}(d^3)$$

$$e = 0.016751$$

where L = mean longitude of the Sun, referred to the mean equinox of date

g = mean anomaly of the Sun

e = eccentricity of the terrestrial orbit

d = number of ephemeris days since 1900 January 0^d 12^h ephemeris time, which is Julian day 2 415 020

For the purpose of this ephemeris, the number of ephemeris days is taken to be equal to the number of Julian days, and the reduction of universal time to ephemeris time is not performed.

The motion of the Earth about the Sun is, to a good approximation, a two-body problem. The eccentricity of the orbit is significant and is considered in the next subsection. The apparent motion of the Sun is considered to lie entirely in the ecliptic plane.

C.2.2.1.2 Corrections to the Mean Motion

Reference 19 gives a sine series to correct the mean motion of the Sun for the eccentricity. The two largest terms have amplitudes exceeding 1' of arc and are (converted to degrees and for an epoch at the year 2000)

$$\delta L = 1.918^{\circ} \sin (g) + 0.0201^{\circ} \sin (2g)$$

The celestial longitude of the Sun is thus computed as

$$\lambda = L + \delta L$$

The celestial latitude of the Sun is 0 by the assumption that the apparent motion lies in the ecliptic plane, i.e.,

$$\beta = 0$$

C.2.2.1.3 Calculation of the Earth-Sun Distance

The distance from the center of the Earth to the center of the Sun, R_S , is computed from the expression for the distance in an elliptic orbit:

$$R_S = \frac{a(1 - e^2)}{1 + e \cos(f)}$$

where a = semimajor axis = 1 Au = 1.495×10^8 km

f = true anomaly = $g + \delta L$

C.2.2.2 Lunar Ephemeris

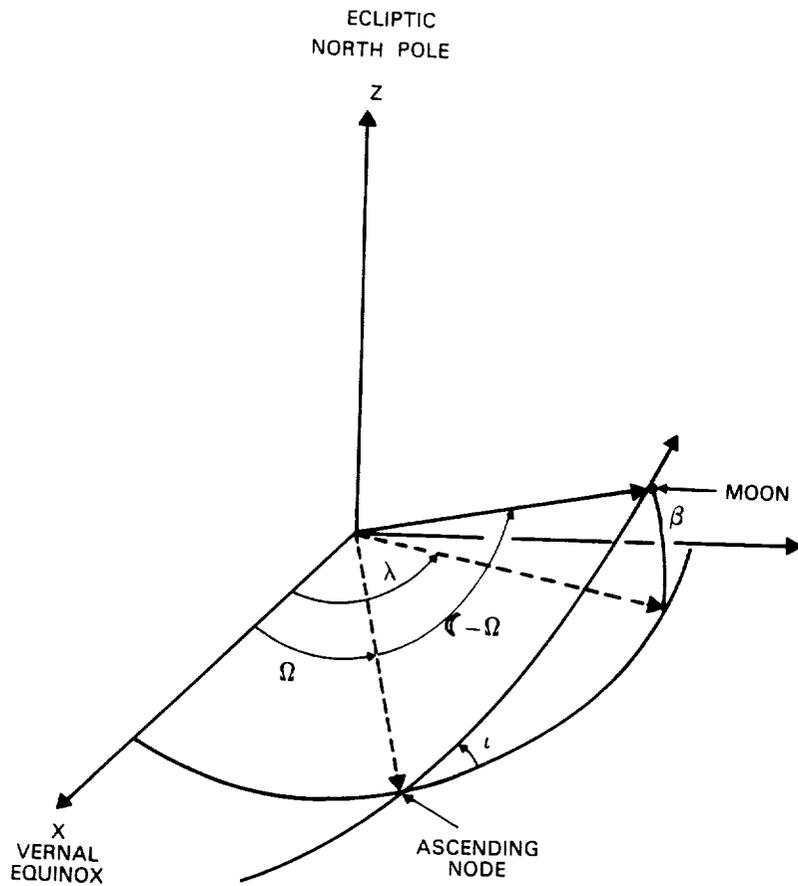
C.2.2.2.1 Mean Motion of the Moon

The mean motion of the Moon, as described in Reference 18, is (Figure C-2)

$$\begin{aligned} \zeta &= 270.^{\circ}434164 + 13.1763965268(d) - 8.5 \times 10^{-13}(d^2) \\ &\quad + 3.9 \times 10^{-20}(d^3) \end{aligned}$$

$$\begin{aligned} \Gamma' &= 334.^{\circ}329356 + 0.1114040803(d) - 7.739 \times 10^{-12}(d^2) \\ &\quad - 2.6 \times 10^{-19}(d^3) \end{aligned}$$

$$\begin{aligned} \Omega &= 259.^{\circ}183275 - 0.529539222(d) + 1.557 \times 10^{-12}(d^2) \\ &\quad + 5 \times 10^{-20}(d^3) \end{aligned}$$



9788/85

Figure C-2. Orientation of the Lunar Orbit in Space

$$D = 350.^\circ 737486 + 12.1907491914(d) - 1.076 \times 10^{-12}(d^2) + 3.9 \times 10^{-20}(d^3)$$

$$i = 5.145396374$$

where ζ = mean longitude of the Moon, measured in the ecliptic from the mean equinox of date to the mean ascending node, and then along the orbit

Γ' = mean longitude of the Moon's perigee, measured as above

Ω = longitude of the mean ascending node of the lunar orbit, measured in the ecliptic from the mean equinox of date

$D = \zeta - L$ = mean elongation of the Moon from the Sun

i = inclination of the lunar orbit to the ecliptic

The motion of the Moon about the Earth cannot be reasonably approximated as a two-body problem, even for coarse attitude systems. Corrections to the mean motion are discussed in the next section.

C.2.2.2.2 Corrections to the Mean Motion

Brown (Reference 20) gives corrections, $\delta\zeta$, to the Moon's mean longitude in the form

$$\delta\zeta = A \sin [\lambda(\zeta - \Gamma') + \lambda'(g) + F(\zeta - \Omega) + d(D)]$$

Only the 10 largest solar perturbations to the motion of the Moon and the 2 largest eccentric terms have been considered here. These are given in Table C-1.

Note that, while Brown identified approximately 1600 periodic terms in the motion of the Moon, only about 10 percent of these have amplitudes in excess of 1 minute of arc.

Table C-1. Periodic Terms in the Longitude of the Moon

<u>Term Number</u>	<u>A (degrees)</u>	<u>l</u>	<u>l'</u>	<u>F</u>	<u>d</u>
1	+6.289	1	0	0	0
2	-1.274	1	0	0	-2
3	+0.658	0	0	0	2
4	+0.213	2	0	0	0
5	-0.185	0	1	0	0
6	-0.114	0	0	2	0
7	-0.059	2	0	0	-2
8	-0.057	1	1	0	-2
9	+0.053	1	0	0	+2
10	-0.046	0	1	0	-2
11	+0.041	1	-1	0	0
12	-0.035	0	0	0	1

Once the corrected longitude $\zeta' = \zeta + \delta\zeta$ is obtained, the celestial latitude, β , and longitude, λ , are obtained from (Figure C-2)

$$\begin{cases} \tan (\lambda - \Omega) = \tan (\zeta' - \Omega) \cos (i) \\ \tan (\beta) = \sin (\lambda - \Omega) \tan (i) \end{cases}$$

C.2.2.2.3 Calculation of the Earth-Moon Distance

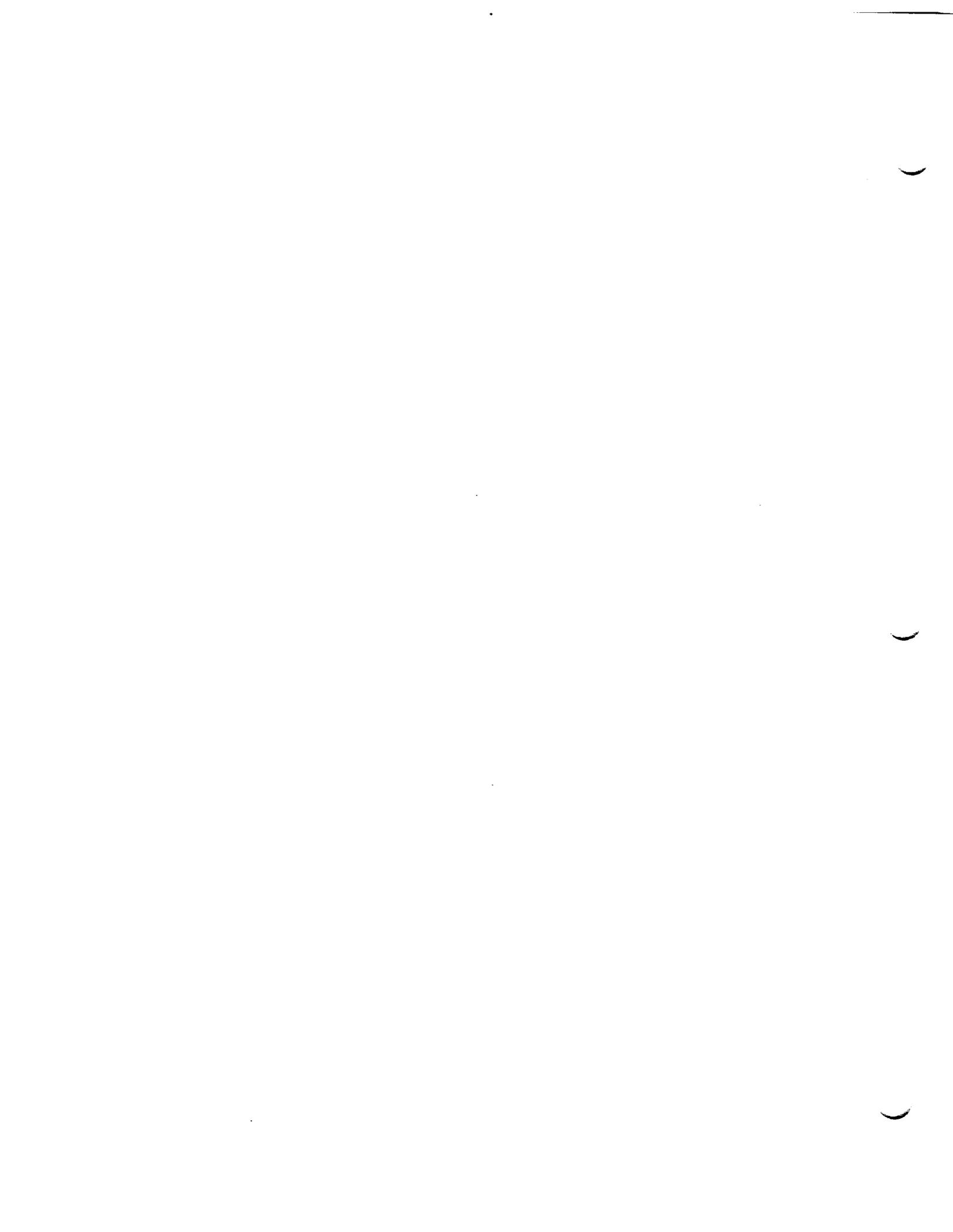
Brown gives a cosine series for the lunar parallax with the same arguments as given for $\delta\zeta$. The nine largest terms in this series (those with amplitudes exceeding 1 second of arc) were used (Table C-2).

The distance, R_m , from the center of the Earth to the center of the Moon may be found from the parallax, p , as

$$R_m = \frac{6378.388}{\sin (p)} \text{ (km)}$$

Table C-2. Periodic Terms in the Lunar Parallax

<u>Term Number</u>	<u>A arc seconds</u>	<u>l</u>	<u>l'</u>	<u>F</u>	<u>d</u>
1	3422.7	0	0	0	0
2	186.5398	1	0	0	0
3	34.3117	1	0	0	-2
4	28.2373	0	0	0	2
5	10.1657	2	0	0	0
6	3.0861	1	0	0	2
7	1.9178	0	1	0	-2
8	1.4437	1	1	0	-2
9	1.1528	1	-1	0	0



APPENDIX D - OBC GROUND COMMAND DESCRIPTION

<u>Mnemonic</u>	<u>Global COMMON Variable</u>	<u>Type</u>	<u>Value</u>	<u>Units</u>	<u>Description</u>
ATTHOLD	IAH	Byte	1	-	Attitude hold command (default)
ATTNO	IAH	Byte	0	-	No attitude hold command
AZBIAS	AZBIAS	R*4	0.0	rad	Set azimuth resolver output bias
BIAZO	BIAZ	Byte	0	-	Normal HGA azimuth limit (default)
BIAZPI	BIAZ	Byte	1	-	Perform 180-deg HGA azimuth flip
BO	BO	I*4	0	counts	Magnetometer data zero offset compensation
B1OFF	B1ON	L*1	F	-	Turn off OATs bank #1
B1ON	B1ON	L*1	T	-	Turn on OATs bank #1 (default)
B2OFF	B2ON	L*1	F	-	Turn off OATs bank #2 (default)
B2ON	B2ON	L*1	T	-	Turn on OATs bank #2
CSLX	CSL(1)	I*4	0	-	Number of cycles count to control low-level thrusters for X-axis
CSLY	CSL(2)	I*4	0	-	Number of cycles count to control low-level thrusters for Y-axis
CSLZ	CSL(3)	I*4	0	-	Number of cycles count to control low-level thrusters for Z-axis
CSS1DIS	CSSON(1)	L*1	F	-	Disable CSS 1
CSS1EN	CSSON(1)	L*1	T	-	Enable CSS 1 (default)
CSS2DIS	CSSON(2)	L*1	F	-	Disable CSS 2
CSS2EN	CSSON(2)	L*1	T	-	Enable CSS 2 (default)
CSS3DIS	CSSON(3)	L*1	F	-	Disable CSS 3
CSS3EN	CSSON(3)	L*1	T	-	Enable CSS 3 (default)

<u>Mnemonic</u>	<u>Global COMMON Variable</u>	<u>Type</u>	<u>Value</u>	<u>Units</u>	<u>Description</u>
CSS4DIS	CSSON(4)	L*1	F	-	Disable CSS 4
CSS4EN	CSSON(4)	L*1	T	-	Enable CSS 4 (default)
C1BREAK	CBT(1)	L*1	T	-	Break tracking mode command for FHST 1
C1TRACK	CBT(1)	L*1	F	-	Start tracking mode for FHST 1 (default)
C2BREAK	CBT(2)	L*1	T	-	Break tracking mode command for FHST 2
C2TRACK	CBT(2)	L*1	F	-	Start tracking mode for FHST 2 (default)
ELBIAS	ELBIAS	R*4	0.0	rad	Elevation resolver output bias
FLTROFF	FLTRI	L*1	F	-	Do not initialize the gyro prefilter
	FLTROFF	L*1	F	-	
FLTRON	FLTRI	L*1	T	-	Initialize the gyro prefilter
	FLTROFF	L*1	T	-	(default)
FSSDIS	FSSON	L*1	F	-	Disable FSSs
FSSEN	FSSON	L*1	T	-	Enable FSSs (default)
FSSOFF	IFSF	L*1	T	-	FSS failure flag (FSSOFF = failure)
FSSON	IFSF	L*1	F	-	FSS failure flag (FSSON = no failure, default)
GREFX1	GRAF(1)	R*4	0.0	rad	Gyro channel A reference angle X-axis
GREFY1	GRAF(2)	R*4	0.0	rad	Gyro channel A reference angle Y-axis
GREFZ1	GRAF(3)	R*4	0.0	rad	Gyro channel A reference angle Z-axis
GREFX2	GRAF(4)	R*4	0.0	rad	Gyro channel B reference angle X-axis

<u>Mnemonic</u>	<u>Global COMMON Variable</u>	<u>Type</u>	<u>Value</u>	<u>Units</u>	<u>Description</u>
GREFY2	GRAF(5)	R*4	0.0	rad	Gyro channel B reference angle Y-axis
GREFZ2	GRAF(6)	R*4	0.0	rad	Gyro channel B reference angle Z-axis
GHIHI	SENSTA(1)	I*4	1	-	Set high-rate mode for gyro 1
GYILO	SENSTA(1)	I*4	0	-	Set low-rate mode for gyro 1 (default)
GY2HI	SENSTA(2)	I*4	1	-	Set high-rate mode for gyro 2
GY2LO	SENSTA(2)	I*4	0	-	Set low-rate mode for gyro 2 (default)
GY3HI	SENSTA(3)	I*4	1	-	Set high-rate mode for gyro 3
GY3LO	SENSTA(3)	I*4	0	-	Set low-rate mode for gyro 3 (default)
HGAAZ	CANAZ	R*4	0.0	rad	HGA azimuth slew command
HGAEL	CANEL	R*4	0.0	rad	HGA elevation slew command
HGAOFF	HGAON	L*1	F	-	Turn off HGA
HGAON	HGAON	L*1	T	-	Turn on HGA (default)
HN1	HN	Byte	0	-	Select FSS head 1
HN2	HN	Byte	1	-	Select FSS head 2 (default)
ICOMEN	ICOM	Byte	1	-	Contingency orbit mode enable
ICOMNDIS	ICOM	Byte	0	-	Contingency orbit model disable
IRU1DIS	IRUON(1)	L*1	F	-	Disable gyro 1
IRU1EN	IRUON(1)	L*1	T	-	Enable gyro 1 (default)
IRU2DIS	IRUON(2)	L*1	F	-	Disable gyro 2
IRU2EN	IRUON(2)	L*1	T	-	Enable gyro 2 (default)
IRU3DIS	IRUON(3)	L*1	F	-	Disable gyro 3
IRU3EN	IRUON(3)	L*1	T	-	Enable gyro 3 (default)

Mnemonic	Global Variable	Type	Value	Units	Description
LONGPUL	ACTPULSE	R*4	2.048	sec	ACT long pulse time
MAG1DIS	MAGON(1)	L*1	F	-	Disable magnetometer 1
MAG1EN	MAGON(1)	L*1	T	-	Enable magnetometer 1 (default)
MAG2DIS	MAGON(2)	L*1	F	-	Disable magnetometer 2 (default)
MAG2EN	MAGON(2)	L*1	T	-	Enable magnetometer 2
MAGFIGA	MAGFIG	I*4	0	-	MTA TDE side A selected (default)
MAGFIGB	MAGFIG	I*4	1	-	MTA TDE side B selected
MO	MO	I*4	0	counts	Torquer bar dipole moment zero offset compensation
MTA1DIS	MTAON(1)	L*1	F	-	Disable MTA 1
MTA1EN	MTAON(1)	L*1	T	-	Enable MTA 1 (default)
MTA2DIS	MTAON(2)	L*1	F	-	Disable MTA 2
MTA2EN	MTAON(2)	L*1	T	-	Enable MTA 2 (default)
MTA3DIS	MTAON(3)	L*1	F	-	Disable MTA 3
MTA3EN	MTAON(3)	L*1	T	-	Enable MTA 3 (default)
NORMAN	ACADMD	I*4	3	-	Use normal maneuver ACAD mode
NORPNT	ACADMD	I*4	2	-	Use normal pointing ACAD mode (default)
NPITOFF	CLLTN(2)	L*1	F	-	No negative rotation about pitch axis using ACTs (default)
NPITON	CLLTN(2)	L*1	T	-	Negative rotation about pitch axis using ACTs
NROLLOFF	CLLTN(1)	L*1	F	-	No negative rotation about roll axis using ACTs (default)
NROLLON	CLLTN(1)	L*1	T	-	Negative rotation about roll axis using ACTs

<u>Mnemonic</u>	<u>Global COMMON Variable</u>	<u>Type</u>	<u>Value</u>	<u>Units</u>	<u>Description</u>
NYAWOFF	CLLTN(3)	L*1	F	-	No negative rotation about yaw axis using ACTs (default)
NYAWON	CLLTN(3)	L*1	T	-	Negative rotation about yaw axis using ACTs
OATDIS	OATEN	L*1	F	-	Disable OAT firing (default)
OATEN	OATEN	L*1	T	-	Enable OAT firing
OAT1OFF	CHL(1)	L*1	F	-	Disable OAT 1 (default)
OAT1ON	CHL(1)	L*1	T	-	Enable OAT 1
OAT2OFF	CHL(2)	L*1	F	-	Disable OAT 2 (default)
OAT2ON	CHL(2)	L*1	T	-	Enable OAT 2
OAT3OFF	CHL(3)	L*1	F	-	Disable OAT 3 (default)
OAT3ON	CHL(3)	L*1	T	-	Enable OAT 3
OAT4OFF	CHL(4)	L*1	F	-	Disable OAT 4 (default)
OAT4ON	CHL(4)	L*1	T	-	Enable OAT 4
OBCOFF	OBCOKAY	L*1	F	-	OBC status is not OK
OBCON	OBCOKAY	L*1	T	-	OBC status is OK (default)
ORBCON	ACADMD	L*1	9	-	Use orbit contingency ACAD mode
PITCHDIS	ACTPIT	L*1	F	-	Disable ACT pair firings - pitch axis (default)
PITCHEN	ACTPIT	L*1	T	-	Enable ACT pair firings - pitch axis
PM111	PM159(1)	R*4	0	-	Error covariance matrix initial value
PM115	PM159(2)	R*4	0	-	Error covariance matrix initial value
PM119	PM159(3)	R*4	0	-	Error covariance matrix initial value

<u>Mnemonic</u>	<u>Global COMMON Variable</u>	<u>Type</u>	<u>Value</u>	<u>Units</u>	<u>Description</u>
PPITOFF	CLLTP(2)	L*1	F	-	No positive rotation about pitch axis using ACTS (default)
PPITON	CLLTP(2)	L*1	T	-	Positive rotation about pitch axis using ACTS
PROLLOFF	CLLTP(1)	L*1	F	-	No positive rotation about roll axis using ACTS (default)
PROLLOON	CLLTP(1)	L*1	T	-	Positive rotation about roll axis using ACTS
PYAWOFF	CLLTP(3)	L*1	F	-	No positive rotation about yaw axis using ACTS (default)
PYAWON	CLLTP(3)	L*1	T	-	Positive rotation about yaw axis using ACTS
Q1	DINIT(1)	R*4	0.0	-	Initial attitude quaternion - Q1
Q2	DINIT(2)	R*4	0.0	-	Initial attitude quaternion - Q2
Q3	DINIT(3)	R*4	0.0	-	Initial attitude quaternion - Q3
Q4	DINIT(4)	R*4	1.0	-	Initial attitude quaternion - Q4
RAXDIS	ISCAN	L*1	F	-	Roll scan mode disable (default)
RAXEN	ISCAN	L*1	T	-	Roll scan mode enable
ROLLDIS	ACTROLL	L*1	F	-	Disable ACT pair firings - roll axis (default)
ROLLEN	ACTROLL	L*1	T	-	Enable ACT pair firings - roll axis
RW1DIS	RWON(1)	L*1	F	-	Disable RW 1
RW1EN	RWON(1)	L*1	T	-	Enable RW 1 (default)
RW1OSDIS	RWOS(1)	L*1	F	-	Disable RW 1 over speed (default)
RW1OSEN	RWOS(1)	L*1	T	-	Enable RW 1 over speed
RW2DIS	RWON(2)	L*1	F	-	Disable RW 2

Mnemonic	Global COMMON Variable	Type	Value	Units	Description
RW2EN	RWON(2)	L*1	T	-	Enable RW 2 (default)
RW2OSDIS	RWOS(2)	L*1	F	-	Disable RW 2 over speed (default)
RW2OSEN	RWOS(2)	L*1	T	-	Enable RW 2 over speed
RW3DIS	RWON(3)	L*1	F	-	Disable RW 3
RW3EN	RWON(3)	L*1	T	-	Enable RW 3 (default)
RW3OSDIS	RWOS(3)	L*1	F	-	Disable RW 3 over speed (default)
RW3OSEN	RWOS(3)	L*1	T	-	Enable RW 3 over speed
RW4DIS	RWON(4)	L*1	F	-	Disable RW 4
RW4EN	RWON(4)	L*1	T	-	Enable RW 4 (default)
RW4OSDIS	RWOS(4)	L*1	F	-	Disable RW 4 over speed (default)
RW4OSEN	RWOS(4)	L*1	T	-	Enable RW 4 over speed
SAFEH	ACADM	I*4	5	-	Use safehold ACAD mode
SALDIS	SAON(1)	L*1	F	-	Disable solar array drive 1
SALLEN	SAON(1)	L*1	T	-	Enable solar array drive 1 (default)
SALINDIS	MODESA(1)	I*4	0	-	Solar array 1 disable index mode
SALINEN	MODESA(1)	I*4	4	-	Solar array 1 enable index mode
SAINEG	SAMOVE(1)	R*4	0.0	rad	Move solar array 1 in negative direction to specified angle
SALPER	MODESA(1)	I*4	3	rad	Solar array 1 step period (default = 64)
SALPOS	SAPER(1)	R*4	4,8,16..512	msec	Move solar array 1 in positive direction to specified angle
	SAMOVE(1)	R*4	0.0 - 1.754	rad	
	MODESA(1)	I*4	3	-	

<u>Mnemonic</u>	<u>Global COMMON Variable</u>	<u>Type</u>	<u>Value</u>	<u>Units</u>	<u>Description</u>
SA1STOP	MODESA(1)	I*4	0	-	Solar array 1 enable stop mode
SA2DIS	SAON(2)	L*1	F	-	Disable solar array drive 2
SA2EN	SAON(2)	L*1	T	-	Enable solar array drive 2 (default)
SA2INDIS	MODESA(1)	I*4	0	-	Solar array 2 disable index mode
SA2INEN	MODESA(2)	I*4	4	-	Solar array 2 enable index mode
SA2NEG	SAMOVE(2)	R*4	0	rad	
SA2PER	MODESA(2)	R*4	0.0 - 1.754 3	rad	Move solar array 2 in negative direction to specified angle
SA2POS	SAPER(2)	R*4	4,8,16..512	msec	Solar array 2 step period (default = 64)
SA2STOP	SAMOVE(2)	R*4	0.0 - 1.754 3	rad	Move solar array 2 in positive direction to specified angle
SC1CLOSE	MODESA(2)	I*4	0	-	Solar array 2 enable stop mode
SC1OPEN	SCSHUT(1)	L*1	F	-	Close FHST 1 shutter
SC1THRES	SCSHUT(1)	L*1	T	-	Open FHST 1 shutter (default)
SC2CLOSE	SCSHUT(2)	L*1	F	-	Set FHST 1 visual star magnitude threshold
SC2OPEN	SCSHUT(2)	L*1	T	-	=0, 6th magnitude (default)
SC2THRES	SCTHRES(2)	I*4	0 - 3	-	=1, 5th magnitude =2, 4th magnitude =3, 3rd magnitude
SC2THRES	SCTHRES(2)	I*4	0 - 3	-	Close FHST 2 shutter
SC2THRES	SCTHRES(2)	I*4	0 - 3	-	Open FHST 2 shutter (default)
SC2THRES	SCTHRES(2)	I*4	0 - 3	-	Set FHST 2 visual star magnitude threshold
SC2THRES	SCTHRES(2)	I*4	0 - 3	-	=0, 6th magnitude (default)
SC2THRES	SCTHRES(2)	I*4	0 - 3	-	=1, 5th magnitude
SC2THRES	SCTHRES(2)	I*4	0 - 3	-	=2, 4th magnitude
SC2THRES	SCTHRES(2)	I*4	0 - 3	-	=3, 3rd magnitude

Mnemonic	Global COMMON Variable	Type	Value	Units	Description
SHRTPUL	ACTPULSE	R*4	0.256	sec	ACT short pulse time
STAND	ACADMD	I*4	1	-	Use standby ACAD mode
SUNREF	ACADMD	I*4	4	-	Use Sun-referenced pointing ACAD mode
TAM1	SELMAG	I*4	1	-	Select magnetometer 1 (default)
TAM2	SELMAG	I*4	2	-	Select magnetometer 2
TCPOFF	TCPINIT	L*1	F	-	Do not initialize thruster command processing
TCPON	TCPINIT	L*1	T	-	Initialize thruster command processing (default)
TDRSE	ITDRS	Byte	0	-	Select TDRS-E for HGA (default)
TDRSW	ITDRS	Byte	1	-	Select TDRS-W for HGA
THRCOM	ACADMD	I*4	8	-	Use thruster command ACAD mode
THRMAN	ACADMD	I*4	6	-	Use thruster maneuver ACAD mode
TTURN	TTURN	R*8	0	sec since 0 UT	Time to start turning the commanded attitude
VGX	VG(1)	R*4	0.0	counts	Input gyro white noise drift (X-axis)
VGY	VG(2)	R*4	0.0	counts	Input gyro white noise drift (Y-axis)
VGZ	VG(3)	R*4	0.0	counts	Input gyro white noise drift (Z-axis)
VRX	VR(1)	R*4	0.0	counts ²	Input gyro random walk drift (X-axis)
VRX	VR(2)	R*4	0.0	counts ²	Input gyro random walk drift (Y-axis)
VRZ	VR(3)	R*4	0.0	counts ²	Input gyro random walk drift (Z-axis)

Mnemonic	Global COMMON Variable	Type	Value	Units	Description
VELCON	ACADM	I*4	7	-	Use velocity control ACAD mode
WCX	WCI(1)	R*4	0.0	rad/sec	Spacecraft roll rate (X-axis)
WCY	WCI(2)	R*4	0.0	rad/sec	Spacecraft pitch rate (Y-axis)
WCZ	WCI(3)	R*4	0.0	rad/sec	Spacecraft yaw rate (Z-axis)
XCHANA	IGY(1)	Byte	0	-	Select gyro channel A, X-axis (default)
XCHANB	IGY(1)	Byte	1	-	Select gyro channel B, X-axis
XGYDIS	IGF(1)	L*1	T	-	Fail gyro X-axis
XGYEN	IGF(1)	L*1	F	-	No failure for gyro X-axis (default)
YAWDIS	ACTYAW	L*1	F	-	Disable ACT pair firings - yaw axis (default)
YAWEN	ACTYAW	L*1	T	-	Enable ACT pair firings - yaw axis
YCHANA	IGY(2)	Byte	0	-	Select gyro channel A, Y-axis (default)
YCHANB	IGY(2)	Byte	1	-	Select gyro channel B, Y-axis
YGYDIS	IGF(2)	L*1	T	-	Fail gyro Y-axis
YGYEN	IGF(2)	L*1	F	-	No failure for gyro Y-axis (default)
ZCHANA	IGY(3)	Byte	0	-	Select gyro channel A, Z-axis (default)
ZCHANB	IGY(3)	Byte	1	-	Select gyro channel B, Z-axis
ZGYDIS	IGF(3)	L*1	T	-	Fail gyro Z-axis
ZGYEN	IGF(3)	L*1	F	-	No failure for gyro Z-axis (default)

GLOSSARY

ACAD	attitude control and determination
ACADS	Attitude Control and Determination Subsystem
ACE	attitude control electronics
ACS	antenna coordinate system
ACT	attitude control thruster
AIDR	acceleration-insensitive drift rate
AMB	Adams-Moulton-Bashforth
BASD	Ball Aerospace Systems Division
BATSE	Burst and Transient Source Experiment
BCS	body coordinate system
BOL	beginning of life
CADHS	Communications and Data Handling Subsystem
CM	center of mass
COMM	contingency orbit maintenance mode
COMPTEL	Imaging Compton Telescope
CPE	control processing electronics
CSS	coarse Sun sensor
DBC	data base constant
DEA	drive electronics assembly
ECI	Earth-centered inertial
EGRET	Energetic Gamma Ray Experiment Telescope
EOL	end of life
EPDS	Electrical Power Distribution Subsystem
FHST	fixed-head star tracker
FOV	field of view
FSS	fine Sun sensor
GCI	geocentric inertial
GGG	Goddard GRO Simulator
GRO	Gamma Ray Observatory
HGA	high-gain antenna
HGAD	high-gain antenna drive
IDT	image dissector tube

IGRF	International Geomagnetic Reference Field
IRU	inertial reference unit
LGA	low-gain antenna
LOS	line of sight
MDE	motor drive electronics
MMS	Multimission Modular Spacecraft
MOI	moment of inertia
MPS	Modular Power Subsystem
MSFC	Marshall Space Flight Center
MTA	magnetic torquer assembly
NMM	normal maneuver mode
NPM	normal pointing mode
NSSC-1	NASA Standard Spacecraft Computer, Model 1
OAT	orbit adjust thruster
OBC	onboard computer
OBCS	origin of the BCS frame
OSSE	Oriented Scintillation Spectrometer Experiment
PE	propulsion electronics
PROM	programmable read-only memory
PS	Propulsion Subsystem
RFOV	reduced field of view
RSM	roll scan mode
RW	reaction wheel
RWA	reaction wheel assembly
RWAHM	reaction wheel attitude hold mode
RWEA	reaction wheel electronics assembly
SCIO	simulation control and input/output
SHM	safe-hold mode
SM	standby mode
SRPM	Sun reference pointing mode
STS	Space Transportation System
TAHM	thruster attitude hold mode

TAM	three-axis (triaxial) magnetometer
TBD	to be determined
TCM	thruster command mode
TCS	Thermal Control Subsystem
TDE	torquer drive electronics
TDRS	Tracking and Data Relay Satellite
TFOV	total field of view
TM	Truth Model
TMM	thruster maneuver mode
TTU	time transfer unit
UTC	universal time coordinated
VCM	velocity control mode



REFERENCES

1. TRW, Inc., SS7-39, GRO Attitude Control and Determination Subsystem Specifications, J. H. Decanini, January 1982
2. --, GRO Technical Proposal (Appendix 8), June 1982
3. Ball Aerospace Systems Division (BASD), TM79-04, User's Guide for NASA Standard Star Tracker, 1979; TRW, EQ7-144, Specifications for FHST, R. E. Edwards, September 1983
4. TRW, Inc., EQ2-675, Specifications for IRU, A. Spadoni, August 1983
5. --, EQ4-2822, Specifications for TAM, R. K. Schisler, August 1983
6. --, EQ7-145, Specifications for FSS, R. K. Schisler, August 1983
7. --, 40420-85-231-008, GRO Critical Design Audit Package, Attitude Control and Determination Subsystem, Volume 2, May 1985
8. --, EQ12-14, Specifications for MT, J. Lee, November 1983
9. --, EQ2-676, Specifications for RW, R. L. Wattenbarger, March 1983
10. Computer Sciences Corporation, CSC/SD-84/6029, Earth Radiation Budget Satellite (ERBS) Dynamics Simulator User's Guide and System Description, D. Shank, August 1984
11. J. R. Wertz, ed., Spacecraft Attitude Determination and Control. Dordrecht, Holland: D. Reidel, 1978
12. Computer Sciences Corporation, 3000-1740-01TR, MAPS/GEOS-C System Description and Operating Guide, Gerald M. Lerner et al., November 1974
13. National Aeronautics and Space Administration, NASA SP-8021, Goddard Space Flight Center, Models of the Earth's Atmosphere (120 to 1000 km), D. K. Weidner, C. L. Hasseltine, and R. E. Smith, May 1969
14. TRW, 40420-85-231-008, GRO Critical Design Audit Package, Attitude Control and Determination Subsystem, Volume 1, May 1985

15. Goddard Space Flight Center, GRO Attitude Error Analysis, GRO Mission Flight Dynamics Analysis Report A1.2, R. Harman, September 1989
16. TRW, XR-SDA-002, Rev. F, Firmware Requirements Specification for Control Processor Electronics, Attitude Control and Determination Subsystem, Gamma Ray Observatory, June 1986
17. --, SS7-39 Rev. C, Subsystem Specification, Attitude Control and Determination Subsystem, Gamma Ray Observatory, March 1985
18. Explanatory Supplement to the Astronomical Ephemeris and the American Ephemeris and Nautical Almanac. London: H. M. Stationary Office, 1961
19. Tables of the Motion of the Earth on Its Axis and Around the Sun, Washington: Bureau of Equipment, Navy Department, S. Newcomb, 1898 (vol. IV of Astronomical Papers of American Ephemeris and Nautical Almanac)
20. E. W. Brown, Tables of the Motion of the Moon. New Haven: Yale University Press, 1919, 3 volumes

DISTRIBUTION LIST

GSEC

C. Woodyard
J. Teles
T. Stengle
J. Jackson
R. Harman

