Appendix F

Instrument Software Management Plan

LMSAL-2H00005
23 June 2003

Stanford University Hansen Experimental Physics Laboratory
and
Lockheed-Martin Solar and Astrophysics Laboratory
Software Management Plan
2H00005

Draft

This document describes the software management plan and activities for the Helioseismic and Magnetic Imager instrument for the Solar Dynamics Observatory.

Approved by:

Jerry Drake
Software Lead

Larry Springer
HMI Program Manager

K. Kao
Software Quality Assurance

Edward McFeaters
Mission Assurance Manager

Jeff Cullen
Manager, MSO Program
Software Process
Engineering

Dwight Wrench
Deputy Director, Computer Systems and Software Engineering

Prepared for:
Stanford University
Hansen Experimental Physics Laboratory
Stanford, CA

Under Contract:
PY-2223

Prepared by:
Lockheed Martin Space Systems Company
Space & Strategic Missiles
Advanced Technology Center – CAGE 65113
Lockheed Martin Solar & Astrophysics Laboratory (LMSAL)
3251 Hanover Street, Palo Alto, CA 94304-1191
## RECORD OF REVISIONS

<table>
<thead>
<tr>
<th>Revision</th>
<th>Description</th>
<th>Approved by</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>Initial Release</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LISTING OF FIGURES</th>
<th>VII</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISTING OF TABLES</td>
<td>VII</td>
</tr>
</tbody>
</table>

## 1. INTRODUCTION

1.1 PURPOSE AND OBJECTIVE ........................................................................................................ 1
1.2 SCOPE ...................................................................................................................................... 1
1.3 ORGANIZATION OF DOCUMENT ............................................................................................... 1

## 2. RELATED DOCUMENTATION ................................................................................................... 2

## 3. PURPOSE AND DESCRIPTION OF SYSTEM .............................................................................. 3

3.1 HIGH LEVEL SYSTEM DESCRIPTION: ......................................................................................... 5
3.2 BOOT SOFTWARE ....................................................................................................................... 6
  3.2.1 FIRST BOOT STAGE (SUROM) ............................................................................................ 6
  3.2.2 SECOND BOOT STAGE (EEPROM) ...................................................................................... 6
3.3 APPLICATION SOFTWARE ........................................................................................................... 6
  3.3.1 EXECUTIVE .......................................................................................................................... 7
  3.3.2 COMMAND HANDLER .......................................................................................................... 7
  3.3.3 TELEMETRY PROCESSOR ..................................................................................................... 7
  3.3.4 MECHANISM CONTROL ....................................................................................................... 7
  3.3.5 THERMAL CONTROL ........................................................................................................... 7
  3.3.6 CAMERA AND HIGH RATE DATA CONTROL ...................................................................... 7
  3.3.7 OBSERVING SEQUENCE CONTROL .................................................................................... 8
3.4 ELECTRICAL GROUND SUPPORT EQUIPMENT (EGSE) SOFTWARE ..................................... 8
  3.4.1 COMMAND LANGUAGE PROCESSOR .................................................................................. 8
  3.4.2 TELEMETRY DISPLAY MANAGER ....................................................................................... 8
  3.4.3 SPACECRAFT INTERFACE .................................................................................................. 8
  3.4.4 GPIB (LABORATORY INSTRUMENTATION INTERFACE) ...................................................... 9
  3.4.5 LOGGING AND ARCHIVING SOFTWARE ............................................................................ 9

## 4. SOFTWARE PROCESS MANAGEMENT DOMAIN ..................................................................... 10

4.1 SOFTWARE DEVELOPMENT PLANNING ................................................................................... 10
### 4. SOFTWARE DEVELOPMENT PROCESS
- 4.1.1 SOFTWARE DEVELOPMENT PROCESS................................................................. 10
- 4.1.2 MANAGEMENT ORGANIZATION ..................................................................... 11
- 4.1.3 SOFTWARE DEVELOPMENT LANGUAGES....................................................... 11
- 4.2 PROJECT TRACKING AND OVERSIGHT............................................................. 11
- 4.2.1 REPORTING.................................................................................................... 11
- 4.2.2 RESOURCES .................................................................................................. 11
  - 4.2.2.1 Cost estimates ........................................................................................... 11
  - 4.2.2.2 Work Breakdown Structure....................................................................... 12
  - 4.2.2.3 Schedule .................................................................................................. 12
  - 4.2.2.4 Training Plan............................................................................................. 12
  - 4.2.2.5 Equipment and facilities........................................................................... 12
- 4.3 SOFTWARE MANAGEMENT INDICATORS .......................................................... 13
- 4.4 SOFTWARE PROCESS IMPROVEMENT............................................................... 14
- 4.5 SOFTWARE ENGINEERING ENVIRONMENT (SEE)........................................... 14
  - 4.5.1 BOOT SOFTWARE ....................................................................................... 14
  - 4.5.2 APPLICATION SOFTWARE.......................................................................... 14
  - 4.5.3 EGSE SOFTWARE......................................................................................... 14
- 4.6 ACQUISITION ACTIVITIES.................................................................................... 15
  - 4.6.1 SUBCONTRACT MANAGEMENT................................................................... 15
- 4.7 RISK MANAGEMENT........................................................................................... 15
- 4.8 SECURITY, PRIVACY, AND EXPORT CONTROL................................................ 16
- 4.9 SOFTWARE RELIABILITY.................................................................................... 16
- 4.10 SOFTWARE SAFETY.......................................................................................... 16
- 4.11 INDEPENDENT VERIFICATION AND VALIDATION (IV&V)............................ 17

#### 5. SOFTWARE PRODUCT MANAGEMENT DOMAIN
- 5.1 SOFTWARE CONFIGURATION MANAGEMENT.................................................... 19
  - 5.1.1 CONFIGURATION MANAGEMENT PROCESS OVERVIEW.......................... 19
  - 5.1.2 CONFIGURATION CONTROL ACTIVITIES..................................................... 20
  - 5.1.3 CONFIGURATION IDENTIFICATION............................................................. 20
  - 5.1.4 CONFIGURATION CHANGE CONTROL...................................................... 20
  - 5.1.5 CONTROLLED STORAGE AND RELEASE MANAGEMENT......................... 21
  - 5.1.6 CHANGE CONTROL FLOW........................................................................... 21
  - 5.1.7 CONFIGURATION DOCUMENTATION.......................................................... 21
  - 5.1.8 CHANGE REVIEW PROCESS: CONFIGURATION STATUS ACCOUNTING........ 21
  - 5.1.9 PRODUCT CONTROL.................................................................................... 21
    - 5.1.9.1 Code Control......................................................................................... 21
    - 5.1.9.2 Documentation Control......................................................................... 22
  - 5.1.10 PROBLEM REPORTING.............................................................................. 22
  - 5.1.11 PRODUCT RELEASE .................................................................................. 23
- 5.2 SOFTWARE QUALITY ASSURANCE................................................................. 23
  - 5.2.1 SOFTWARE QUALITY EVALUATION PROCEDURES.................................... 23
  - 5.2.2 SOFTWARE QUALITY PROGRAM................................................................. 23
  - 5.2.2.1 Software Development Planning............................................................... 24
  - 5.2.2.2 Project Tracking and Oversight................................................................. 24
  - 5.2.2.3 Software Engineering Environment......................................................... 24
  - 5.2.2.4 Software Management Indicators............................................................. 24
  - 5.2.2.5 Software Analysis and Design................................................................... 25
### 5.3 Corrective Action Process

5.3.1 Preparation for Use

5.3.2 Software Maintenance

5.3.3 Corrective Action

5.3.4 Software Configuration Management

5.3.5 Software Peer Review

5.3.6 Software Item Qualification Test

6. Software-Systems Interface Domain

6.1 System Requirements Definition

6.2 System Design

6.3 Hardware-Software Integration

6.4 System Qualification Test

6.5 Preparation for Use

6.6 Preparation for Transition

6.6.1 Delivery and Operational Transition

6.7 Software Maintenance

7. Software Engineering Domain

7.1 Software Requirements Definition

7.2 Software Architecture Design

7.3 Software Detail Design

7.4 Code and Unit Test

7.5 Unit Integration and Test

7.6 Software Item Qualification Test

### Appendix A. Abbreviations and Acronyms
APPENDIX B. HMI SOFTWARE ORGANIZATION ...................................................... 40
LISTING OF FIGURES

Figure 5.4-1  HMI SPR Form .................................................................................................................. 28
Figure 7.1-1  HMI IGR Minutes Template.......................................................................................... 33

LISTING OF TABLES

Table 4.2.1.5-1  Software-Related Equipment for HMI ................................................................. 13
Table 4.3-1  HMI Software Measurements......................................................................................... 14
Table 4.7-1  HMI Risk Management Approach .............................................................................. 16
1. INTRODUCTION

1.1 Purpose and Objective
The purpose of this document is to describe the Software Management Plan (SMP) for the Helioseismic and Magnetic Imager (HMI) instrument for the Solar Dynamics Observatory (SDO) mission. Software for HMI is being developed by the Solar and Astrophysics Laboratory of the LMMS Advanced Technology Center (ATC).

The objective of this plan is to establish the policies, procedures and guidelines to be used during the software development life cycle. This document addresses the following subjects for the HMI software:
- Program Description
- Software Process Management
- Software Product Management
- Software-System Interface
- Software Engineering

1.2 Scope
This document is applicable to the specification, development, and production of the software delivered with the HMI instrument for the control and operation of the HMI instrument. This includes the Boot Software, Application Software and the Electrical Ground Support (EGSE) Software. Definitions of these deliverable Computer Software Configuration Items (CSCIs) are contained in sections 3.2, 3.3 and 3.4 respectively. Subsystem Ground Support software, test software, simulation software, data collection/reduction software and data analysis software are not covered by this document. There are no subcontractors involved with the development of the deliverable software. This program is compliant with the LMMS Standard Software Process. The contents of this plan are consistent with the LMMS Standard Software Process as tailored for the ATC programs.

1.3 Organization of Document
The organization of this document reflects the organization of the LMMS Standard Software Process (SSP) with expansion in appropriate areas to address all items identified in the Data Item Description (DID) 5.1 of the SDO HMI Performance Assurance Implementation Plan (PAIP).
- Section 1 Introduction
- Section 2 Related Documentation
- Section 3 Purpose and Description of System
- Section 4 Process Management Domain
- Section 5 Product Management Domain
- Section 6 Software-System Interface Domain
- Section 7 Software Engineering Domain
- Appendix A Abbreviations and Acronyms
- Appendix B HMI Software Organization
2. RELATED DOCUMENTATION

The following documents are referenced in this document.

<table>
<thead>
<tr>
<th>Document Number</th>
<th>Title</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>LMMS-2H00021</td>
<td>Solar Dynamics Observatory Helioseismic and Magnetic Imager Performance Assurance Implementation Plan (PAIP)</td>
<td></td>
</tr>
<tr>
<td>SU-XXXXXXXX</td>
<td>HMI Instrument Performance Document</td>
<td></td>
</tr>
<tr>
<td>LMMS-2H00004</td>
<td>HMI Software Requirements Document</td>
<td></td>
</tr>
<tr>
<td>LMMS - 2H0008</td>
<td>HMI Boot Sequence Design Document</td>
<td></td>
</tr>
<tr>
<td>LMSS - 2HXYZW</td>
<td>HMI Application Code Design Document</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HMI Software Requirements and Traceability Matrix</td>
<td></td>
</tr>
<tr>
<td>LMMS-2B02279</td>
<td>Unit Development Folders for Flight Software Modules</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HMI WBS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HMI Overall Schedule</td>
<td></td>
</tr>
<tr>
<td>LMMS - 2H00006</td>
<td>HMI Software Coding Standards</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HMI Configuration Management Plan</td>
<td></td>
</tr>
<tr>
<td>LMMS 204A496</td>
<td>Software User’s Guide for the RAD6000 Processor</td>
<td></td>
</tr>
</tbody>
</table>
3. PURPOSE AND DESCRIPTION OF SYSTEM

The HMI instrument is designed to produce filtergrams of the full solar disk taken with a narrow band (~0.1 Å bandpass) tunable filter in multiple polarizations and with high spatial resolution (1"). Two Charge-Coupled Device (CCD) cameras are operated to maintain a single observing sequence such that one obtains line-of-sight magnetic flux images of the Sun every 45 seconds and the other obtains vector-magnetic field maps every 90 seconds. These images are obtained at the specified cadences with high time precision.

The HMI integrates 9 instrument subsystems:

- The **Optics Package** includes the optics package structure, the mounts for the various optical components and the legs that mount the optics package to the spacecraft.

- The **Optics Subsystem** includes all the optical elements except the filters.

- The **Filter Subsystem** includes the following filters:
  - Front window
  - Fixed blocker filter
  - Lyot filter with a single tunable element
  - Two tunable Michelson interferometers

- A **Thermal Subsystem** which controls the temperature of the optics package, the filter oven, the CCDs (including decontamination heating) and the front window.

- The **Image Stabilization Subsystem** consists of the following three components which act together to actively stabilize the image by reducing the effects of jitter:
  - A mirror which is actively controlled using piezo-electric transducers.
  - A limb sensor.
  - Digital and analog electronics which transform error signals to control signals to the piezo-electric transducers.

- A **Mechanisms Subsystem** consisting of motors and mechanical elements to control the following:
  - Shutter (2)
  - Calibration wheels/focus wheels (2)
  - Polarization selectors (3)
  - Filter tuning waveplates (4)
• Aperture door (2)
• Alignment legs (2)

• A **Camera Subsystem** containing two identical cameras each consisting of the following components:
  • A 4K x4K CCD
  • Focal Plane Assembly (FPA) to which the CCD is mechanically mounted
  • Camera electronics

• An **Electronics Subsystem** which contains the following:
  • Conditioned power for all HMI subsystems
  • RAD6000 processor card with EEPROM memory
  • Electronics boards to interface with the spacecraft and all HMI subsystems containing electronics

• **Flight Software** which performs the following functions:
  • Spacecraft 1553 command and housekeeping telemetry interface
  • Control the flow of high-rate data
  • Acquire and format housekeeping telemetry
  • Store and execute operational sequences
  • Control subsystem hardware as necessary to perform operational sequences
  • Ability to accept modifications to the code in orbit

Images are taken every 4 seconds with each camera. The wavelength or polarization is changed between each image. These images are transferred from the camera subsystem to the buffer/compressor card in the electronics subsystem. Software initiates the image transfers which are then performed under hardware control. The images are buffered and compressed (hardware implemented compression algorithm) in the buffer/compressor card. Parameters for the compression algorithm can be commanded through the flight software but the algorithm is implemented in hardware (FPGA) in the buffer/compressor card. Transmission of the images to the spacecraft over the high-rate bus is initiated by the software but the interface and transmission protocol are performed in hardware on the buffer/compressor card.

The temperature of the filters must be controlled very closely. The filter oven temperature control is performed under a hardware closed-loop system. Parameters to the hardware algorithm may be commanded through the flight software. There is also a back-up software control loop for the oven temperature control. In addition, several
optical package zones (which do not require as tight a control) are controlled through the flight software.

Image stabilization is performed by the image stabilization subsystem. This system is implemented in hardware. A sensor images the Sun, extracts jitter information from the images, converts this jitter information into control signals in electronics within the image stabilization system and sends those signals to piezoelectric transducers on the mirror in the optical path of HMI to remove the jitter. The image stabilization system can be turned on and off through software.

The following are under FSW control:

- Subsystem power on/off
- Set subsystem control/telemetry parameters
- Obtain high rate diagnostic data and telemetry via 10 kbps 1553 interface
- Select redundant capabilities

**Subcontractors and Institutions:**
The organizational relationships with other institutions are listed below;

1. NASA, Goddard Space Flight Center (GSFC) – US customer; Oversees development and accepts delivery of the instrument.
2. Stanford University - Prime contractor to GSFC and Principal Investigator institution.

Subcontractors:

1. British Aerospace (BAE) Federal Systems – RAD6000 CPU Board

### 3.1 High Level System Description:
The HMI instrument accepts commands from the SDO spacecraft and returns status information and images to the SDO spacecraft. It manages, monitors, and reports the health and status of the HMI.

More specifically, the HMI flight application software performs the following functions:

- Receive and process commands from the spacecraft over the 1553 interface
- Provide housekeeping telemetry to the spacecraft over the 1553 interface
- Perform the observing sequence which includes:
  - Maintaining a precise cadence (one image every 4 seconds nominally)
  - Controlling the wavelength and polarization mechanisms (between images)
  - Controlling the camera exposures
Initiating image transfers
Maintaining instrument thermal control (except for oven control)
Provide status information to be included with the image data
Initiate image telemetry transmission over the high-rate spacecraft interface

The HMI software is divided into the following Computer Software Configuration Items (CSCIs):
- Boot Software
- Application Software
- Electrical Ground Support Equipment Software (EGSE)

### 3.2 Boot Software

This Computer Software Configuration Item (CSCI) will bring the instrument from the application of power into application mode software operation. There are two components to this CSCI: Start-Up ROM (SUROM) code in a PROM on the processor card and code in EEPROM.

#### 3.2.1 First Boot Stage (SUROM)

The SUROM code will be activated on application of power to the processor card. This code is programmed into a PROM and unchangeable after physical installation onto the processor card. This code performs initialization of the processor card and 1553 interface to the spacecraft, downloads the VxWorks operating system from EEPROM into RAM, initiates the operating system and downloads the second stage boot code from EEPROM to RAM and boots it. Limited commands over the 1553 can be accepted and processed in this stage and telemetry is provided over the 1553 interface to the spacecraft.

#### 3.2.2 Second Boot Stage (EEPROM)

The second stage boot software runs under the VxWorks operating system and has access to the operating system features. This code will interface with the spacecraft over the 1553 for both commands (a more extensive set than the SUROM stage) and telemetry. Under normal operational conditions this code will download the application code from EEPROM to RAM and boot the application code. The capability to pause in the boot state (by issuance of a command during a countdown period) and upload new code or patches is also included in this code.

### 3.3 Application Software

This Computer Software Configuration Item (CSCI) performs all instrument control and communications functions. It will be designed to run on a RAD6000 radiation hard RISC processor. The software will be written in the C++ and C languages to operate under the VxWorks operating system.
3.3.1 Executive
The executive controls all the tasks which make up the flight software. For each system tick the major software components are allowed time (if needed) in a specific order to complete all the possible actions that the flight software may need to perform.

3.3.2 Command Handler
The HMI command handler software provides the capability to receive commands over the 1553 spacecraft interface, to process and verify those commands and to act appropriately on those commands. These commands allow full HMI control directly or can update tables used for the observing sequence. Results of error checking on the received commands are included in telemetry for ground monitoring and management of the HMI instrument.

3.3.3 Telemetry Processor
The telemetry processor software gathers HMI status information, formats this information and makes this housekeeping information available to the spacecraft 1553 bus for downlink. Additionally diagnostic data at a higher rate can be made available by command request on the 1553 bus. The telemetry processor also makes available to the high-rate data stream selected status information for inclusion with high-rate image data.

3.3.4 Mechanism Control
Several different kinds of mechanisms (motors) are used in the operation of HMI. Tuning waveplates are controlled by tuning motors (4). Polarization can be selected for each exposure through the polarization selectors (3). A calibration can be performed through the calibration/focus wheel units which are controlled by motors (2). Focus is set by the calibration/focus wheels (2). Images are obtained by controlling the exposures of the CCDs using shutter motors (2). The alignment is adjusted by alignment motors (2). The door can be opened or closed by exercising the door motors (2). All these mechanisms are controlled in the mechanism control module.

3.3.5 Thermal Control
The HMI thermal control software will maintain the temperatures in each of several zones upon command. A higher precision thermal control is provided by an oven control system in which the control loop is closed entirely in hardware. For the hardware oven control system, the flight software provides certain control parameters and only turns on or off the oven control. There is also a back-up software control loop for the oven temperature control which allows a set-point and deadband for oven temperature software control.

3.3.6 Camera and High Rate Data Control
Control of the readout of the cameras, the data compression performed on the images obtained from the cameras and the transmission of the high rate image data are performed in this module. Data compression is performed in hardware with different options selectable by command from the flight software. No image processing is performed in the flight software. The high rate data transmission protocol is in the hardware; the flight software only initiates and monitors the transmission.
3.3.7 Observing Sequence Control
To perform an observing sequence, the duration of the exposure, the cadence (time between exposures), the wavelength and the polarization of each exposure must be specified. The HMI flight software must control the shutters in accordance with the observing sequence, must control the readout of the camera, initiate the data transfer from the camera to the data compression/buffer card, must initiate the high-rate data transfer and must keep the cadence constant to a high timing accuracy. The observing sequence will be provided in a tabular form and must be maintainable and changeable as well as interpreted and performed by the flight software.

3.4 Electrical Ground Support Equipment (EGSE) Software
The Electrical Ground Support Equipment (EGSE) is used during the integration and test activities to issue commands to the instrument and to receive and archive telemetry from the instrument. It consists of the following components;

- Command Language Processor
  - Command Database
- Telemetry Display Manager
  - Telemetry Database
- Interface to the SDO simulator or Spacecraft
- Logging and Archiving Software
- Evaluation Software

EGSE software will be essentially the same as that developed for the SXI and Solar-B instruments. The main exceptions are the databases and spacecraft interface modules.

3.4.1 Command Language Processor
The EGSE will include a STOL like language to implement test scripts. The language provides for conditional and looping facilities in addition to all arithmetic operations on different types of variables. The variables can be integers, floating point or string types. A compiler converts ASCII scripts to an internal format. It generates the appropriate binary instrument commands.

3.4.2 Telemetry Display Manager
The display manager displays telemetry data in real time or from a playback file. The housekeeping parameters to display and the display format are controlled by a user defined formatting file. It uses a database to translate telemetry mnemonics to an offset within the housekeeping stream and converts raw values as specified in the database. Conversions include extraction of bit fields and applying a polynomial fit to convert raw values to engineering units.

3.4.3 Spacecraft Interface
A spacecraft interface is required to function between the HMI instrument and the EGSE in simulation of the SDO spacecraft for the 1553 command and telemetry function, the high-rate data interface and to provide power to the HMI instrument. A spacecraft simulator will be provided by GSFC to perform these functions. The EGSE software then interfaces to either the SDO spacecraft simulator or to the spacecraft to send commands and to receive housekeeping and diagnostic telemetry over the 1553 bus and
to receive science telemetry over the high-rate interface. This module will perform bit-syncing if necessary, de-packetize the telemetry stream and put the data into buffers which can be used for displaying and archiving.

3.4.4 GPIB (Laboratory instrumentation interface)
During testing laboratory instrumentation, such as power supplies, temperature monitors, relays, function generators, etc. are controllable through this module of the EGSE and are controllable using the STOL language. Packets can be generated with measurements from these instruments that can be displayed along with housekeeping telemetry.

3.4.5 Logging and Archiving Software
The telemetry packet data is sorted by Application Identifiers and logged to individual packet files. Event messages are logged to the ASCII history file which also contains a record of all telecommands sent to the instrument. Raw telemetry data can also be saved. File names are time tagged for ease of retrieval. The telemetry archive files can be played back to debug software or to check out new equipment.
4. Software Process Management Domain

4.1 Software Development Planning

The HMI flight software development process will follow the incremental model, which allows an operational, albeit limited functionality, version of the software to be released relatively early in the development process. Further builds providing greater functionality are released regularly throughout the development period. This provides two primary benefits: As hardware development proceeds, the hardware can be tested using flight-like software and the software is tested for a longer period under more realistic operational conditions.

The major milestones for HMI flight software development are based upon the hardware development schedule. This includes the build and test of brassboards for a subset of the system. The schedule for the necessary flight software builds is dependent on the order of the hardware development and will be identified when the hardware development schedule is more definitized. These are listed below:

- **Build 1** – Implements fully functional spacecraft 1553 interface, command handler, telemetry handler and executive.
- **Build 2** - Mechanisms, heaters, housekeeping data acquisition, image stabilization system and power subsystem interfaces and control
- **Build 3** - Camera and high rate data interfaces and control
- **Build 4** - Observing sequence control

The parameters related to a formal Build and used to track the software configuration are identified in Section 5.1.3.

4.1.1 Software Development Process

The incremental model develops the full functionality in a series of iterations or defined builds of the software. Each build is developed following the top-down model. Through derivation of detailed requirements, the model is refined at each iteration to take into account interface, timing or other detailed design changes that might result from work on the previous build. New or changed software requirements generated due to hardware features or re-scoping of science needs are also reviewed with each build. Once the requirements are defined for a build, they are assigned to individual tasks within a CSCI. Each CSCI consists of a number of Computer Software Components (CSCs). For HMI flight software the CSC’s will be implemented as classes or functions. Software requirements are established during the software requirements phase and each build implements a subset of those requirements.

Following the decomposition of the requirements into CSCs, the detailed design will be undertaken for each CSC, using object oriented and/or structured programming techniques, as appropriate. The use of these techniques improves the modularity of the design, readability of the documentation, effectiveness of testing, and the tracking of the software development process.
4.1.2 Management Organization

The HMI organization chart is shown in Appendix B. The identification of the individual with a task may change during the course of this program. Changes in personnel will not cause an update of this document. NASA will be notified of significant changes to these personnel. In general, two people will be knowledgeable in each subject area such that the loss of any one individual due to whatever cause will not have a significant effect on either the HMI software development process or on the HMI program. Additional personnel will be added as necessary to maintain schedule within the budgetary allocation. This chart does not address the manpower resources required for the formal test phase of the software which will additionally include hardware personnel.

4.1.3 Software Development Languages:
The HMI flight software will be written using a combination of both C++ and C languages. While the high level functionality lends itself naturally to object orientation, some time critical functions require predictable and consistent execution times. These will be written in C. Assembly code will only be used if required for achieving time critical performance or to facilitate memory management if suitable facilities are not available in the Green Hills compiler.

4.2 Project Tracking and Oversight

4.2.1 Reporting
Schedule reporting will be contained in the Program Monthly Schedule Report. Resource and budget information will be reported in the Monthly and Quarterly Financial Management report. Software status reports will be included in the Monthly Progress reports. Changes to the software deliverables agreed upon with the customer will be reflected in modifications in the Work Breakdown Structure and the Statement of Work. Lessons learned are reviewed with program management as they are identified.

4.2.2 Resources

4.2.2.1 Cost estimates
The costs have been estimated from software development experience on the following programs: Solar X-Ray Imager (SXI), Solar-B Focal Plane Package (FPP) and the High Resolution Dynamics Limb Sounder (HIRDLS) instruments. The same processor (RAD6000) was used on all three programs. All three programs were NASA managed programs. All HMI flight software developers have experience on one or more of these three programs. Similar instrument hardware was used on SXI and Solar-B as will be used on HMI. The development costs take into account the development experience obtained from these prior instruments as well as on the TRACE and MDI missions.
4.2.2.2 Work Breakdown Structure
The detailed work breakdown structure is described in the HMI WBS. The software development tasks are grouped under WBS 2.6 (with the exception of Software Quality Engineering which is in WBS 2.3.4) and are broken down into the following categories:

<table>
<thead>
<tr>
<th>WBS #</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.4</td>
<td>ATC Software Quality Engineering</td>
</tr>
<tr>
<td>2.6.1</td>
<td>Software Management (Software Requirements and Oversight)</td>
</tr>
<tr>
<td>2.6.2</td>
<td>Boot Code Flight Software</td>
</tr>
<tr>
<td>2.6.3</td>
<td>Application Code Flight Software</td>
</tr>
<tr>
<td>2.6.4</td>
<td>EGSE Software</td>
</tr>
<tr>
<td>2.6.5</td>
<td>Subsystem GSE Software</td>
</tr>
</tbody>
</table>

4.2.2.3 Schedule
The overall program schedule is available in the HMI Overall Schedule. The software development schedule will be governed by the overall program schedule. In particular, it will be driven by availability of brassboard and/or flight hardware.

It is planned to develop a prototype of the observable construction modules relatively early, using C++. This will allow the scientific staff to use the prototype to develop and evaluate observing sequences early in the program, thus refining the concept of operations as well as testing the observing sequence modules thoroughly. The prototype code will provide an early implementation of all external interfaces required for the flight code. However, the prototype code is not delivered as flight code. The actual flight code is based on the prototype code but is developed and tested separately.

4.2.2.4 Training Plan.
The flight software development team members have the experience of using the hardware and software platforms needed for the HMI project. No additional training needs are foreseen at this time, however, training needs will be reviewed as the need arises, either due to new members joining the team or team members being re-assigned to a different type of work (i.e. from EGSE software to flight software or from coding to testing).

4.2.2.5 Equipment and facilities
Table 4.2.1.5-1 provides a listing of the software-related equipment needed for the HMI software development effort.

<table>
<thead>
<tr>
<th>Equipment Item</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sun Workstation</td>
<td>SW Development</td>
</tr>
<tr>
<td>Tornado Development Kit</td>
<td>SW Development</td>
</tr>
<tr>
<td>VxWorks Target License</td>
<td>SW Development</td>
</tr>
<tr>
<td>Greenhills Multi Development System</td>
<td>SW Development</td>
</tr>
<tr>
<td>Revision Control System tool</td>
<td>SW Development</td>
</tr>
<tr>
<td>Brassboard motherboard and 1553 interface card</td>
<td>Initial SW Development</td>
</tr>
<tr>
<td>Engineering model processor card</td>
<td>Initial SW Development</td>
</tr>
</tbody>
</table>
The HMI program will construct brassboards of selected systems needing early prototyping and the flight model of the instrument. The brassboards will be used for early interface testing and for software testing and debugging while the flight model is being constructed. Two development stations are needed to support development and test activities for software and hardware in parallel.

### 4.3 Software Management Indicators

Management of the software effort is aided by the regular collection, reporting, and analysis of software management indicators. These measurements follow the guidelines in the LMMS Software Measurement Guidebook provided by the Central SEPG. Table 4.3-1 describes the software management indicators that will be captured by this program.

The HMI program will also provide periodic software metrics reporting to the central software/process organization in accordance with Engineering Instruction 3.3.0.8.

In addition to these measurements, a monthly schedule of completion of each CSC of the flight software will be drawn up. This schedule will be the primary operational tool for tracking progress. The schedule will be maintained by the software lead and status will be reported in the monthly report.

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Reporting Cycle</th>
<th>Responsible For Reporting</th>
<th>Responsible for Analysis and Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schedule Performance</td>
<td>Weekly</td>
<td>All</td>
<td>Program Manager</td>
</tr>
<tr>
<td>Labor Performance</td>
<td>Weekly</td>
<td>All</td>
<td>Program Manager</td>
</tr>
<tr>
<td>Problem Tracking</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Software DAIs/IARs Opened</td>
<td>Monthly</td>
<td>All</td>
<td>Program Manager Software Lead</td>
</tr>
<tr>
<td>Number Software DAIs/IARs Closed</td>
<td>Monthly</td>
<td>Software Lead</td>
<td>Program Manager Software Lead</td>
</tr>
<tr>
<td>Milestones</td>
<td>Monthly</td>
<td>Software Lead</td>
<td>Program Manager</td>
</tr>
</tbody>
</table>

The table above lists the software-related equipment for HMI.
Process Maturity

| Completion/update of PMI questionnaire | Monthly | Software Lead | Central SEPG |

Definitions:
Software Milestone – A major event during the software product lifecycle. Typically, there will be 5 to 10 such events during the software development and test phases, such as SRR, PDR, CDR, Software Acceptance Tests, and completion of the planned builds.

**4.4 Software Process Improvement**
The HMI program manager (or his designee) participates on the ATC SEPG (Software Engineering Process Group). The program manager (or his designee) monitors HMI process status. Process improvements will be made as needed with guidance from the ATC SEPG.

**4.5 Software Engineering Environment (SEE)**
This section describes the Software Engineering Environments under which the different components of HMI software will be developed. The development platforms are all Commercial Off the Shelf (COTS) products and are described below. The deliverable HMI software products are: boot software, application software and EGSE software.

**4.5.1 Boot Software**
The boot software will be developed on an RS6000 workstation using the PowerPC C compiler. The version control tool will be the public domain Revision Control System (RCS).

**4.5.2 Application Software**
The application software will be developed on a Sun Microsystems Sunblade platform running Solaris Version 9. The Green Hills compiler v1.8.9 will be used for code compilation. The Tornado 2.0 integrated development environment (IDE) will be used for debugging and special test code module loads (to edit, compile and archive source code). The initial testing and debugging of CSC’s will be carried out by running the software on a brassboard/EM platform. All source code will be version controlled using the RCS public domain tool (or the Green Hills “Multi Version Control” tool). All files associated with each software build will be grouped into separate directories. The released versions will be write-locked.

The resources required for HMI flight software development are listed in Table 4.2.1.5-1, Software-Related Equipment for HMI.

**4.5.3 EGSE Software**
The EGSE software will be developed on an Instrument Workstation (IWS), which will be a Sun Microsystems Sunblade workstation running the Solaris OS, Version 9. The
software development environment will be provided by the COTS product, Sun Microsystems Visual Workshop for C++ Development Software Suite. This includes a C/C++ compiler, debugger, optimizer, and a Motif GUI builder. Each IWS will be purchased with a C/C++ language development environment and a single user license. A maintenance contract will be procured for each development environment package according to standard procurement practices.

A log of any changes to the base SEE shall be maintained. This log will cover any patches/updates to the COTS software and changes, upgrades or modifications to the hardware platforms. Changes to network configurations shall also be logged.

4.6 Acquisition Activities

Items to be purchased for HMI software development include COTS software (Wind River operating system, Greenhills Multi development environment), computer hardware (Sun computers, monitors, disk drives, etc.) for software development and EGSE, hardware for subsystem GSE (prototyping boards, cables, connectors, etc.) and the processor board (RAD6000) from BAE. The COTS software, computer hardware and hardware for subsystem GSE will be procured through the standard LMSSC Purchase Order process. The processor board(s) from BAE will be procured through a subcontract to BAE which is administered by LMSSC Subcontracts (Org. EG-11, Bldg. 157) in the main plant in Sunnyvale.

4.6.1 Subcontract Management

The BAE subcontract is administered through the Subcontract Administrator (Org. EG-11, Bldg. 157) and through the Subcontract Technical Manager (Org. L9-41, Bldg. 252, HMI Program Office). The Subcontract Administrator is the sole authority to make changes, revisions or amendments on behalf of LMSSC and to effect deviation (by way of additions or deletions) to the subcontract. The Technical Manager provides technical direction as necessary to direct the subcontract within the written technical parameters, specifications and schedule of the subcontract as necessary to meet program objectives.

4.7 Risk Management

This section contains the HMI Flight Software approach to risk management. Risk management will be reviewed at each monthly status meeting and updated as new risks are identified or the plan is changed. Each risk is assigned a priority and will be addressed in priority order. The program manager is responsible for managing program risk. This responsibility includes monthly review including software risk management. The current Software Risk Management status will be reported to the customer as part of the monthly program status report.

<table>
<thead>
<tr>
<th>Priority</th>
<th>Risk Item</th>
<th>Mitigation Approach</th>
<th>Backup Plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Requirements volatility</td>
<td>Define requirements early in program</td>
<td>Escalate to program manager if requirements start to &quot;creep&quot;</td>
</tr>
<tr>
<td>2</td>
<td>Adequate testing</td>
<td>Test at board-level, box-level and instrument level</td>
<td>Participate in planning of testing for all elements which</td>
</tr>
</tbody>
</table>

15
Table 4.7-1  HMI Risk Management Approach

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Out of Spec. Hardware</td>
<td>Thorough Unit testing to understand hardware behavior. Develop workarounds.</td>
</tr>
<tr>
<td>4</td>
<td>Management of the corrective action process</td>
<td>Perform problem report tracking</td>
</tr>
<tr>
<td>5</td>
<td>Staff size and volatility</td>
<td>Define and document all software development and test procedures so manpower resources are realistically estimated.</td>
</tr>
<tr>
<td>6</td>
<td>Schedule Changes</td>
<td>Software will be divided into a number of independent modules, which can be rescheduled more easily.</td>
</tr>
</tbody>
</table>

4.8 Security, Privacy, and Export Control

The flight software created in this program is subject to ITAR Export control.

4.9 Software Reliability

Software reliability will be achieved by following the sound software development practices described in this SMP (as stated in the PAIP, Section 4.3.5). It will be verified by using the flight software as early in the program as possible (starting at board level testing), by the Software Acceptance Test that will be performed including stress testing and off-nominal conditions testing and by use in all instrument testing. DAI s or IARs will be written against the flight software if any reliability issues are found during formal instrument testing. A DAI or IAR will cause the software problem to be resolved at the Program level through the CCB.

4.10 Software Safety

Software safety (as indicated in the PAIP, Section 3.5) is addressed in this section of the SMP and can be considered using the categories used to evaluate the need for Independent Verification and Validation (IV&V) or Independent Assessment (IA) (see section 4.11). The HMI software does not have the potential to cause loss of life, serious injury, catastrophic mission failure, loss of equipment, waste of software resource investment of more than 20 man-years and no effect on routine operations above inconvenience. Safety will be reviewed at all Program reviews. If any safety issues are identified, they will be dealt with by providing software Fault Management to remove the
possibility of occurrence of that safety issue or by adding operational constraints. Software safety items will be brought to the attention of Systems Engineering and subsystem Responsible Engineers as the design and implementation of HMI progresses. Any items identified will be placed in the Software Requirements Document and dealt with in the software development process as all requirements must be satisfied and their implementation or solution properly verified (Software Acceptance Test which may include inspection and/or analysis).

4.11 Independent Verification and Validation (IV&V)

The criteria for determining the applicable Independent Verification and Validation (IV&V) or Independent Assessment (IA) for a software development effort are based on a combination of the probability that a program or project will experience an undesired event such as cost overrun, schedule slippage, safety mishap or failure to achieve a needed technological breakthrough and the consequences, impact or severity of the undesired event were it to occur. Consequences of software failure may result in any one or more of the following (in decreasing order of significance):

- Loss of life
- Potential for serious injury
- Potential for catastrophic mission failure
- Potential for loss of equipment
- Potential for waste of software resource investment
- Potential effect on routine operations

Consequences of software failure may be considered according to the following:

**Grave**
- Potential for loss of life
- Potential for loss of equipment (greater than $100,000,000)
- Potential for wast of resource investment (greater than 200 man-years software effort)
- Potential for adverse international publicity

**Substantial**
- Potential for serious injury
- Potential for catastrophic mission failure
- Potential for loss of equipment (greater than $20,000,000)
- Potential for waste of resource investment (greater than 100 man-years software effort)
- Potential for adverse publicity (national)

**Significant**
- Potential for partial mission failure
- Potential for loss of equipment (greater than $1,000,000)
- Potential for waste of resource investment (greater than 20 man-years software effort)
- Potential for adverse publicity (NASA)
- Potential effect on routine operations (NASA Center work stoppage or Agency inconvenience)

**Insignificant**
- No potential for loss of life
No potential for serious injury
No potential for catastrophic mission failure
No potential for loss of equipment (less than $1,000,000)
Potential for waste of resource investment (less than 20 man-years software effort)
Potential for adverse publicity (no more than local visibility)
Potential effect on routine operations (no more than NASA Center inconvenience)

A matrix is used to perform a risk assessment to determine the level of IV&V or IA needed for a software development effort. This is a matrix of the consequences of software failure versus the total likelihood of failure based on the software environment. HMI has been evaluated against the criteria specified and falls in the matrix box for "Insignificant" consequence of software failure and in the box for the lowest possible total likelihood of failure based on the software environment. The recommendation for the box in which HMI falls is that neither IV&V nor IA is needed. Consequently the HMI program does not plan to perform either IV&V or IA.
5. **Software Product Management Domain**

5.1 **Software Configuration Management**

5.1.1 **Configuration Management Process Overview**

This section describes the plan for performing software configuration management on the HMI program.

The purpose of software configuration management is to first establish a software baseline and then control changes to the software against the baseline. The HMI program will follow the software configuration management approach described below. Software configuration management plans are reviewed by the HMI program manager with input from the HMI ATC SQE.

Software will be put under formal Program level configuration control at the Software Acceptance Test. The primary Software Acceptance Test will occur just prior to the start of instrument level environmental testing. An acceptance test for certain functionality (such as the 1553 spacecraft interface or other items which may not be fully testable at the instrument level) may occur prior to the primary Software Acceptance Test. Any acceptance testing occurring prior to the primary Software Acceptance Test will be repeated (to the extent possible) at the Software Acceptance Test.

Prior to the time the software comes under formal Program level control, it will be under informal configuration control by the software development Responsible Engineer (RE) for that particular software CSCI. Informal configuration control is performed using RCS similarly to that used when the code is under formal configuration control. Code may be changed by the RE while code is under informal configuration control with no CCB action required. When code is under formal configuration control, Program CCB approval is required for code changes. Code may be changed in the development directory while under formal configuration control in order to continue development and/or to debug and fix problems that have been identified. The functioning of the HMI Program CCB is described in the HMI Configuration Management Plan.

Software configuration management will be implemented by the Software Lead. Changes to the software will be put into configuration management following test of the changed software item. System regression tests will be performed before releasing software for use by non-software team members and as well as prior to all other software releases to the customer.

A web-based problem reporting system, used on previous programs (SXI and Solar-B FPP) will be put in place to log problems or requests for enhancements. The problem report numbers will be referenced in the source code modules.
The flight software configuration will be controlled using the Green Hills Multi Version Control System (MVCS) software package installed on the development machine network that will be used by all software team members (or RCS Revision Control System as used on HIRDLS). The MVCS check-in and check-out utilities will be used. The software module author will provide comments to the check-in module describing the nature of the software change. Additionally, the development environment configuration will be backed up periodically.

5.1.2 Configuration Control Activities

Configuration control activities include:
- Documentation of current code (Version Description Documents and Design Documents)
- Control of current code (frozen release directories and code archived on CD or DVD)
- Using Developer Anomaly and Improvement (DAI) forms for problem/change identification (for Level 2 or below changes)
- Using Instrument Anomaly (IAR) forms for problem/change identification (for Level 1 changes)
- Problem tracking
- Problem understanding and solution determination
- Formal HMI Program Change Control Board action to approve the solution or to use as is if that is the better resolution
- Regression testing of any updated flight code version
- Documentation of the change (Version Description Document, CCB minutes and any affected design or requirements documents)
- Archival of the newly released code version and all supporting software (makefiles, etc.).

5.1.3 Configuration Identification

The software will be identified by unique version numbers embedded in the code and included in telemetry. The version number format related to a formal Build and used to track the software configuration is:

- Alpha bit 1 bit Under development (1) or released (0)
- Build number 4 bits Build number identified in schedule based on functionality and need
- Revision 6 bits Incremental
- CPU type 2 bits Brassboard (0), Protoflight Model (PFM, 1), development target (2) or spare (not used, 3)

5.1.4 Configuration Change Control

Configuration changes are controlled by the HMI Program through the CCB process.
5.1.5 Controlled Storage and Release Management
Code is released by setting the alpha bit to zero, checking into RCS all code modules, running the makefile, verifying no compilation errors occurred and write-locking the directory containing the released code. The code is then archived off-line (CD or DVD) with the Data Management group.

5.1.6 Change Control Flow
Following is the change control flow:
- Identify and document the problem (DAI or IAR)
  - Problems are identified but whether they are software or hardware is not always known initially
- Troubleshoot the problem until its cause and solution are understood
- CCB the problem and solution
- Add test case(s) or update Test Plan as necessary
- Prepare Test Procedure
- Prepare code alpha release and Test Procedure for dry-run testing (if CCB decision was approval of change)
- Dry-run code and regression test
- Release flight code and regression test procedure
- Install flight code and perform regression test
- Archive flight code
- Document regression test in Test Report

5.1.7 Configuration Documentation
Documentation related to configuration changes may include:
- DAI or IAR
- CCB minutes
- Software requirements document(s)
- Design document(s)
- User's Guide
- Version Description Document(s)
- VCRS or RCS statements regarding code changes

5.1.8 Change Review Process: Configuration Status Accounting
Configuration status will be tracked in in an electronic file (Word or Excel) and updated as new items are added and/or existing items are brought to completion.

5.1.9 Product Control

5.1.9.1 Code Control
Code is under the software development team control until the first formal release for the Software Acceptance Test (SWAT). Code changes until the SWAT will be based on the addition of functionality as hardware becomes available, on errors which must be fixed due to problems found or to requirements changes. Code during this time is physically controlled by the software developer using RCS.
After the first formal release, code changes are implemented by the software RE but only upon approval of the CCB.

5.1.9.2 Documentation Control
Documentation related to software products will be processed through the standard HMI Program and LMSSC document release system. Documents will be reviewed, approved and signed by the appropriate HMI Program personnel as identified in the HMI Program Directive regarding required document signators. Upon final approval, they will be processed through the release cycle, receive a "diamond stamp" and be placed in the document vault. Any changes to the documentation will be processed (through the HMI CCB) as an update to the released document.

5.1.10 Problem Reporting
Problem reporting will use a web-based tool in which an anomaly can be reported in the DAI system or in the IAR system. Use of these systems will be initiated upon the completion of the Software Acceptance Test. Problem reporting and tracking prior to the Software Acceptance Test will be performed informally by the software development team.

DAI items may be entered by anyone working on the HMI program and are generally items which have impacted an assembly, fabrication or test. The DAI system is used to track all elements, including software, which need attention by the program. An entry in the DAI system contains the following information:
- ID number
- Title
- Status
  - Open
  - Assigned
  - RE Closed
  - Formal Closed
- Primary area (software for this management plan)
- HW System
- Anomaly Date
- RE (Responsible Engineer)

The Software Lead assigns priority and the Responsible Engineer (RE) will proceed to evaluate the problem and solution. This should be done within a few days of receiving the DAI. When the RE thinks that the problem is fixed, he sends the report to the review board for formal closure. A review process will be conducted to formally close the DAI.

IAR items require disposition with the NASA GSFC Program whereas DAI items can be dispositioned within the HMI program at Stanford University and LMSSC. The format and information content are the same for both systems (DAI and IAR). The process for tracking and closure for the IAR is the same as that for the DAI except that the IAR Manager (Program Manager) is responsible for managing and tracking the IARs.
5.1.11 Product Release

Release of a code version will involve the following steps:

- The final build for this code version will be run.
- The directory containing the source code and build will be write-locked (no further development can occur in this directory).
- The directory containing the source code and build will be archived to CD.
- The Version Description Document (VDD) will be updated to document the changes contained in the released code version.
- Any documentation requiring update will be identified for update.
- Documentation updates will then occur (this may be after the code is released depending on schedule pressure).

All code releases (after the first formal code release) will undergo regression testing which will consist of either a repeat of any test cases affected by the changes in the released code or new test cases (if needed because the problem corrected was not captured by any previous test case).

5.2 Software Quality Assurance

The description of the HMI Software Quality Engineering approach provided in this section constitutes the HMI Software Quality Program Plan. The planned activities to achieve high quality in all HMI software products involve the entire program software staff and require that they understand and adhere to the standards and practices described in this Software Management Plan. The Software Quality Engineer (SQE) for the Lockheed-Martin Advanced Technology Center (ATC) provides independent inspections and audits to ensure that the HMI software development efforts adhere to their planned software development and management methods. The paragraphs below describe the activities of the ATC SQE in maintaining continuous life cycle surveillance over the HMI software development processes and products.

5.2.1 Software Quality Evaluation Procedures

The Lockheed Martin Missiles and Space (LMMS) central SQE Organization maintains a suite of Software Quality Assurance Practices defining the process used in performing the Quality Program. These practices are adapted to each program to best serve that program’s needs in the most cost-effective manner. The most current version of the practices is located at URL: http://policy.global.lmco.com/lmssc/2_0.html.

5.2.2 Software Quality Program

The HMI Software Quality Program addresses activities that occur throughout the lifecycle of a software development effort. It specifically focuses on processes and products that are described in the LMMS Standard Software Process (SSP) as adapted for the HMI program. Process and product evaluations are scheduled and the SQE reports status of its activities and evaluation results to program management.
The following paragraphs provide a description of ATC SQE activities in support of software quality on the HMI program.

5.2.2.1 Software Development Planning

The ATC SQE supports the development of the HMI Software Management Plan by reviewing draft sections and providing inputs for the Software Quality Assurance section. The ATC SQE audits the final draft for completeness, correlation to the SSP and to determine those processes and products that will be subject to audit and inspection throughout the HMI software development life cycle.

5.2.2.2 Project Tracking and Oversight

The ATC SQE periodically attends software staff and other meetings where progress on software-related tasks is discussed and assessed. The SQE is kept informed of:
- Software-related replanning discussions and decisions (scope, schedule, staff, resources)
- Problems related to software progress or quality and plans to address them

5.2.2.3 Software Engineering Environment

The ATC SQE audits the final Software Engineering Environment (SEE) to ensure its adequacy for supporting HMI software development activities. The focus of the audit includes:
- Does the SEE provide adequate support for all software life cycle phases?
- Are software personnel adequately trained in the use of SEE tools?

5.2.2.4 Software Management Indicators

The ATC SQE maintains surveillance of the HMI program’s definition and use of software progress and quality indicators. The focus of this surveillance includes:
- Do the software management indicators provide HMI program leadership with an accurate understanding of the software effort’s schedule and budget status?
- Do the software management indicators provide HMI program leadership with sufficient advance warning of software budget or schedule problems to allow for effective pro-active corrective actions?
- Do the software requirements indicators provide HMI program leadership with sufficient insight into the degree of program requirements stability and the risk that late life cycle instability will pose to the software development effort?
- Do the software peer review and defect indicators provide HMI program leadership with sufficient insight into the volume, types, and causes of software errors to allow for effective process improvements designed to eliminate the error causes?
5.2.2.5 Software Analysis and Design

The ATC SQE maintains surveillance of the HMI software analysis and design efforts by auditing the related processes and products as they are described in this Software Management Plan. The focus of such audits is to ensure that:

- The systems and software analysis efforts have resulted in documented software requirements that completely specify the mandatory capabilities that the HMI software must provide.
- The HMI software engineering staff understands the software requirements and how they flow to the individual software units that are to be developed.
- The software design effort has resulted in documented software designs that fully reflect allocated requirements and that reflect a level of software design detail that allows coding to begin.
- The HMI software engineering staff understands the designs of the software that they will code, as well as any coding standards applicable to their assigned development units.
- The HMI software engineering staff understands the various verification activities applicable to their software (i.e. peer reviews, unit testing, higher levels of integration testing) and is participating in those verification activities.

5.2.2.6 Software Testing

The ATC SQE reviews and signs off on the HMI Software Test Plan, witnesses the formal testing of the final integrated software configuration item and signs off on the final as-run Test Procedure. The focus of the Test Plan and Test Procedure review and the test witnessing is to ensure that:

- The Test Plan addresses all software requirements
- The Test Procedure accurately reflects the intentions of the Test Plan and shows which test cases and detailed test steps will verify which requirements
- The documented test cases and detailed steps provide the correct demonstrations and analyses to sufficiently verify their assigned requirements
- The formal testing accurately follows the detailed test steps of the Test Procedures and any anomalies observed during execution of those detailed test steps are recorded for subsequent analysis and corrective action.

5.2.2.7 Corrective Action

The ATC SQE evaluates the software corrective action process for adequacy, and maintains surveillance over the operation of that process throughout the HMI software life cycle. The focus of such surveillance includes:

- Are observed software anomalies being formally captured for analysis and correction by cross-disciplinary teams?
- Is each anomaly that is accepted for analysis assigned to a responsible software engineer?
• Is the progress of analysis and corrective action tracked?
• Are observed anomalies closed in a timely manner by appropriate corrective actions?
• Are corrected code units re-introduced into software builds in a controlled manner?

5.2.2.8 Software Configuration Management
The ATC SQE will review the HMI software Configuration Management Plan and procedures, change control process, and the build and delivery process. The focus of these reviews is to ensure that:

• Integrated software and associated documentation is placed under formal CM control prior to formal software testing and that such control is effective in precluding uncontrolled changes to the software and documentation.
• The CM process uniquely identifies, through a planned and enforced alphanumeric naming scheme, software entities down to the unit level.
• The software that undergoes formal testing is under CM control and is identified down to each individual unit of code in the build.
• The CM process provides a means for corrected, verified code units to be placed in updated software builds in a controlled manner.
• The CM process provides for code unit status accounting that allows the status and most current version of any code unit to be determined.

5.2.2.9 Software Peer Review
The ATC SQE supports the HMI Software Peer Review process by ensuring that HMI software personnel have been trained in the LMMS software peer review process, by randomly participating in HMI software peer reviews, by auditing to ensure that the peer review process is performed properly and by ensuring that appropriate peer review records are kept.

5.3 Corrective Action Process
The HMI software corrective action process is designed to identify and correct anomalies in the HMI software in a manner that maintains configuration control of the code and its associated documentation. The process is supported by the use of an automated web-based discrepancy reporting system that archives software discrepancy or change request forms, broadcasts them to a planned list of email recipients and tracks their status to closure. An HMI Change Control Board (CCB) will be established before the software enters Software Acceptance Testing. Identified software anomalies or requests for change are discussed, and appropriate actions assigned, in HMI CCB meetings. See paragraph 5.1 for the software configuration management process.

5.4 Software Peer Reviews
It is planned to subject all critical modules (as determined by the software lead) to peer review. The reviewers may include scientific staff, where appropriate, who can assess if the requirements have been interpreted correctly for the software modules being reviewed. The software peer review plan has been developed based upon guidelines
provided in the Software Peer Review Guidebook which is available on the LMMS intranet.

5.4.1 Overview of SPR Approach
This project will use colleague or inspection level peer reviews on different software entities as described in the Software Peer Review Guidebook. The main criteria in deciding to perform colleague or inspection reviews will be determined by the complexity and criticality of the product. Products with high and medium levels of complexity will be reviewed by inspection. All others will go through a colleague review. Software Peer Reviews are included in the detailed software development schedule.

5.4.2 SPR Roles
The primary lead for a product is responsible for distributing review materials to team members. Team members are responsible for reviewing and red-lining material and trying to pinpoint any existing or possible defects. The primary lead is responsible for gathering a list of findings. Findings will be reported to the appropriate persons for a fix. Once the fix has been completed, the SQE will be notified and invited to review related items as deemed necessary.

5.4.3 Identification of Software Products
The following is a list of software entities that will be peer reviewed:
- HMI software requirements
- HMI software architecture
- Detailed design of each major module
- Code of critical functions – These will be identified during the design review.
- Test plans and procedures

5.4.4 Software Peer Review (SPR) Forms
Figure 5.4-1 shows the template of the form used for capturing the Saves generated at an SPR.

The recorder for each meeting is responsible for compiling the list of findings on a record form as depicted in Figure 5.4-1. Note that the possible findings for each reviewed entity will be documented in a separate table. Deviations or non-conformance to the Software Requirements Document (SRD) or the HMI coding standards shall also generate findings (in addition to programming errors).

5.4.5 NASA Involvement in Software Peer Reviews
Internal peer reviews will be performed as described above involving LMSSC and Stanford software peers. Additional peer reviews will be scheduled for participation by NASA personnel at NASA's direction.
# HMI Software Peer Review (SPR) Record

<table>
<thead>
<tr>
<th>SPR ID</th>
<th>HMI -SW</th>
<th>Meeting Date</th>
<th>SPR Type</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product Name</th>
<th>HMI Flight Software</th>
<th>Moderator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attendees</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reader</th>
<th>List Any</th>
<th>Supporting Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Author</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reviewer</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reviewer</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To insert or delete a row, place the cursor in the Finding # column, then press Insert or Delete

<table>
<thead>
<tr>
<th>Disposition</th>
<th>Rework</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td></td>
</tr>
<tr>
<td>Cond Accept</td>
<td>Due Date:</td>
</tr>
<tr>
<td>Re-review</td>
<td>Estimated Hrs:</td>
</tr>
</tbody>
</table>

## Types of FINDINGS:

- Place [x] in the category box for a MINOR SAVE
- Place [m] in the category box for a MAJOR SAVE
- Place [d] in the category box for a DEFECT
- Place [e] in the category box for an ENHANCEMENT

## FINDINGS Categories:

- CO: Correctness
- CM: Completeness
- CL: Clarify
- CS: Consistency
- ST: Standards
- OT: Other

## FINDING Status

- O: Open
- C: Closed
- D: Closed to SDR
- E: Closed to SCR

## Software Peer Review Findings List

(Note: The coding standards applicable to HMI software are documented in LMMS- 2HWWWWWW)

<table>
<thead>
<tr>
<th>Finding Number</th>
<th>Location</th>
<th>Description</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>CO</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ST</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Status</td>
</tr>
</tbody>
</table>

---

**Figure 5.4-1** HMI SPR Form
5.5 **Intergroup Reviews**

Intergroup reviews of the software requirements and the software design will be held with Stanford University as well as with other subsystem engineers within the HMI program. Intergroup meetings and reviews will occur regularly throughout the SDO program focusing on various technical items as they are identified and/or as the need for completion of their definition occurs. This effort is expected to be ongoing with the SDO project and this activity has already started on the HMI program.

5.6 **Joint Technical & Management Reviews**

A number of formal reviews will be conducted during the software development process. These will give the customer (GSFC) and the user community an opportunity to assess how the software design is being implemented. The review process provides an exchange of technical details and program status.

5.6.1 **Software Requirements Review**

HMI instrument Software Requirements will be reviewed at Technical Interchange Meetings with the science team from Stanford University. Additional reviews will cover the spacecraft interface with the SDO spacecraft team at GSFC. Most of these reviews will be held prior to the instrument PDR.

5.6.2 **Preliminary Design Review**

A Preliminary Design Review of the software will be held to review all CSCIs. It will be part of the HMI PDR and will formally evaluate the adequacy of the specification, design, associated management plans, and identify unresolved technical issues. The review items shall be the flight software requirements document, the software top-level design documentation, and the software design approach and progress.

5.6.3 **Critical Design Review**

A Critical Design Review of the software will be held to formally establish the detailed design specifications for software end products. It will be part of the HMI CDR. Design documentation presented and delivered at the CDR will become the development baseline.

5.6.4 **Test Readiness Review**

This review will be performed immediately prior to the Software Acceptance Test and will review the status of the flight software, the availability and suitability of test facilities and any liens against hardware or software that might affect the Software Acceptance Test.

5.6.5 **Acceptance Review**

The final software review is the Software Acceptance Review. This review will cover the results of software acceptance testing and will be part of the corresponding hardware Pre-shipment Review.
6. **Software-Systems Interface Domain**

6.1 **System Requirements Definition**

The top level system requirements for the HMI instrument will be developed by the science team at Stanford University and will be documented in the HMI Instrument Performance Document. System-level software requirements will be a portion of the total system-level requirements.

6.2 **System Design**

6.3 **Hardware-Software Integration**

The integration of the software modules will follow the build structure stated in section 4.1. Each build, upon completion, will be tested. Each build after Build 1 adds functionality to that provided by the previous build. Each build integration test, after Build 1, includes regression testing to verify that the capabilities verified in previous builds are still performing correctly. This integration and test must be performed on the HMI flight unit.

Personnel independent of the flight software under test will develop the test software required to perform the integration and testing activities on the hardware bench. In addition, independent personnel will design the tests, prepare the test procedures and perform the actual testing using auxiliary instrumentation and test equipment as necessary to establish the software performance. To the extent that simulations of external subassemblies can provide a real time environment, run-time performance will be demonstrated.

6.4 **System Qualification Test**

The system qualification test will consist of a set of tests which for a NASA program include: EMC/EMI, vibration, thermal-vacuum and the long-form and short-form functional tests which are run periodically throughout the qualification testing to verify proper functionality of the instrument.

6.5 **Preparation for Use**

Preparation for use will involve creation of the Software User's Guide, preparation of the STOL procedures which will be used to test and operate the instrument, archival of the final version of the flight software and updating any documentation (design, test plan, test procedures, etc) which may not be current.
6.6 Preparation for Transition

Following the Software Acceptance Test, the flight software enters a phase during which it is used as an integral configuration item of the operational system. The sustaining engineering phase can be defined in two concurrent parts, operations and software maintenance.

6.6.1 Operations and Sustaining Engineering

The operations phase is conducted for the remaining life of the software and consists of tasks necessary to configure software for mission-unique applications through commanding. The software maintenance phase is conducted for the remaining duration of the contract. Configuration management practices are used to track software changes in the same way as was used through the development life cycle to this point.

6.6.1.1 Delivery and Operational Transition

The HMI flight software will be maintained throughout the life of the HMI instrument by the flight software development team at LMSSC. In that sense there is no delivery. The operational transition will be accomplished by several actions:

- Involvement in mission operations and planning from the start of the program
- Stanford mission operations personnel involved with flight software development and review from the start of the program
- Small program so a minimum number of people and low (or no) turnover is expected
- Use of STOL for writing all formal tests (to the maximum extent possible)
- Involvement with Instrument level and spacecraft level testing by personnel who understand the instrument hardware and software and who write the flight software acceptance test procedures
- Adequate documentation

6.7.2 Software Maintenance

The software developed for the HMI instrument will be maintained at the same site as development was done. The transition to the maintenance process will consist of maintaining space for documentation and development hardware in a location convenient and accessible to maintainers. The location of this site will be a designated area in Building 252 or the current location of the HMI development team.

6.7.2.1 Software Requirements and Design Updates

The software and system design descriptions for software will be updated to match the actual software items prior to the Pre-Ship review. The system requirements allocated to software will be updated to match the actual system prior to the Pre-Ship review. Software modifications determined to be required after delivery will be tracked by updates to the requirements and design documents.

6.7.2.2 Software Maintenance Training

Software maintenance will be performed mainly by the developers. They may also train additional personnel to perform software maintenance. The size of the effort for all aspects of the program indicates one on one training in an informal environment. The developer will compile the materials and set a presentation schedule. Training will cover
operation, design, development methods, test, validation, and configuration management for each of the delivered software systems.

6.7.2.3 Maintenance Manuals
Preliminary manual material will be prepared during software detailed design. This material will be combined with supporting information and revised to reflect actual implementations to produce the site maintenance manuals. The manuals will consist primarily of electronic documents for ease of storage, use and updating. The site for this information will be a directory on the flight software development system. Backup of this information will be made to CD or DVD.

6.7.2.4 Product Support
Product support will be performed on the delivered software items, EGSE software, instrument simulator software and flight instrument software.

The lead software engineer will monitor and assess system performance during the commissioning phase of the mission. Following instrument checkout, members of the development team will monitor system performance for an extended period while it is in normal operational use. Any anomalies will be reported via a web based Instrument Anomaly Reports to the lead software engineer.

6.7.2.5 User Support
The Instrument Anomaly Report (IAR), a web based form, will be used to log software problems. Software questions will be handled primarily through e-mail and by telephone for urgent issues.
7. **Software Engineering Domain**

7.1 **Software Requirements Definition**

The purpose of the requirements analysis process is to ensure that the requirements for each build are both complete and consistent. A complete and consistent set of requirements is necessary to prevent later software development problems, such as cost overrun, schedule slip, or deficient performance. The Software Requirements Document contains the complete baseline of program software requirements. Functional, interface, and performance requirements for each CSCI are established through interaction with various personnel and with reference to related documentation.

Intergroup reviews (IGR) will be conducted, as needed, with the Software, Hardware, Systems, and Integration & Test Engineers to resolve any issues or questions related to the correct interpretation of derived software requirements and their correct verification.

IGR Minutes will be generated using the template shown in Figure 7.1-1.

<table>
<thead>
<tr>
<th>HMI Intergroup Review Minutes Template</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Meeting date, time, and location</strong></td>
</tr>
<tr>
<td><strong>Meeting attendees</strong></td>
</tr>
<tr>
<td><strong>Major items of discussion</strong></td>
</tr>
<tr>
<td>1.</td>
</tr>
<tr>
<td>2.</td>
</tr>
<tr>
<td>3.</td>
</tr>
<tr>
<td><strong>Identified Lessons Learned</strong></td>
</tr>
<tr>
<td><strong>Identified Software Process Improvements</strong></td>
</tr>
<tr>
<td><strong>Time meeting adjourned</strong></td>
</tr>
</tbody>
</table>

**Figure 7.1-1  HMI IGR Minutes Template**

At the completion of the requirements definition and analysis phase, a flight software requirements review will be conducted and a requirements traceability matrix will be established. The requirements traceability matrix will be configured to trace the flow of system specifications and user requirements throughout the software development process.
7.2 **Software Architecture Design**

The software architecture design is driven by the timing requirements needed for proper performance of the science observations for HMI and by the choice of processor (RAD6000) made for the HMI instrument.

7.3 **Software Detail Design**

The preliminary design phase activity begins during the requirements analysis and definition phase. After the Software Requirements Review, the primary software activity is preliminary design culminating at the Preliminary Design Review. The software requirements analysis and preliminary design activities are performed once for the program software. Detailed design then occurs from the Preliminary Design Review until the Critical Design Review. The top level of integration for each software package is the Computer Software Configuration Item (CSCI). It is defined as a deliverable software product, which consists of one or more Computer Software Components (CSCs). During the design phase each flight software CSCI is decomposed into smaller logically related entities called Computer Software Components (CSCs). A CSC consists of one or more software modules, which perform the required functions. The detailed design activity is repeated during each planned build to ensure that design changes resulting from the previous build are accounted for in the design of the next build.

Data and control flow diagrams for each module may be generated for the preliminary design. In addition to refining the software architecture, prototyping will be used throughout the design phase to obtain performance information on different design alternatives. Peer reviews will be held during the software design phase to ensure that system level requirements are correctly interpreted.

More specifically, the design phase activities include: 1) allocating CSCI requirements to the Computer Software Components (CSCs), 2) establishing a Unit Development Folder (UDF) for each CSC, 3) decomposing the structure/flow charts into progressively more detailed levels, and 4) defining and describing the major software data structures and interfaces. To aid in the implementation phase, a preliminary CSCI build and integration plan is produced at the end of the detailed design phase. In parallel with the design effort, the test matrix for the CSCI is produced and the development of software test procedures for the CSCI is started. This phase is concluded with the Critical Design Review (CDR).

The Unit Development Folders are designed to capture the requirements implemented by the unit, the files comprising the unit, the unit testing methodology and test results. If a unit needs to be updated, the unit testing is repeated and appended to the UDF. The UDF template is described in FPP document, 2B02279.

Intergroup reviews will be conducted, as needed, with the Software, Systems, and Hardware Engineers to resolve any issues related to software performance or hardware/software interfaces. The template shown in Figure 7.1-1 shall be used to capture the Minutes of an IGR.
7.4 Code and Unit Test

After the requirements and design models are established and reviewed, the coding and unit-testing phase begins. During this phase, the software design is translated into the code required to implement the design. The coding will follow the HMI software coding standards as described in HMI document 2H00006. The flight code for selected modules, (identified during design peer review) will be peer reviewed. The coding and preliminary integration and testing activities occur on the software bench. For the HMI flight software, the software bench consists of a SUN workstation running the TornadoII development environment. The test code will be targeted for a RAD6000 CPU. These, preliminary integration and test activities will be performed by the software developers on the development computers.

Due to the embedded nature of the flight software, it is not practical or possible to test all modules on the software bench. Therefore, once a functional version of the software for a particular set of modules has been built, simulation of the major components will be used on the software bench. The simulation will incorporate all modules that do not require direct hardware access to be tested. When practical, certain hardware components will be integrated into the software bench to permit proper simulation / emulation activities. Once the simulation has proven (to a reasonable extent) that no major failures, which could damage hardware, are present, integration with the brassboard hardware will commence. Software integration and testing activities move from the software bench onto the hardware bench for subsystem testing.

On the hardware bench, a logic analyzer may be used to provide visibility into the flight software execution. To the extent to which it is practical during this early hardware/software integration and testing stage, software simulations and/or hardware emulation of the subassemblies, which interface to the processor, will be generated. In addition to allowing the software to be tested with a minimum set of hardware, the software simulations of interface subassemblies will allow anomalous conditions to be set up to test the response of the software.

The software is tested to verify that it meets the performance and functional requirements contained in the Software Requirements Document (SRD). Software development iterates between the coding, integration, and test phases as software errors are identified and corrected. Software errors will be reported and tracked using the web-based Developer Anomaly and Improvement (DAI) and Instrument Anomaly Report (IAR) system after the Software Acceptance Test. After Software Acceptance Testing has begun, the formal Program CCB will approve all software changes for incorporation back into the code baseline. All modules that have software errors identified will be subject to regression testing. Regression testing involves a retest of the affected module, and all other modules, that have an interface with the affected module. The regression testing is performed to ensure that errors corrected by one module repair do not propagate different errors into a related but independent module.

Errors are identified during regression testing by running benchmark tests under known and controlled conditions, then comparing the results with those previously generated and
verified. Discrepancies must then be identified, understood and corrected until eliminated and a clean run is made.

Intergroup reviews will be conducted, as needed, with the Software, Systems, Hardware, and Integration & Test Engineers to resolve any issues related to infeasible requirements, alternate design strategies, hardware/software interfaces, timing or software testing strategies. The HMI Instrument Performance Document is the primary reference for high level specifications. However, the appropriate sub-system design specification documents will be used for detailed specifications. Other reference documents such as, the RAD6000 User’s Guide (204A496) will be used as applicable.

### 7.5 Unit Integration and Test

System integration and validation is a joint effort between the project test organization and the system and software engineering team members. The system being developed, which may consist of multiple CSCIs and Hardware Configuration Items (HWCIs), is tested as an integrated entity to validate that the system meets all requirements contained in the applicable subsystem and system level specifications. Software simulations and hardware emulations are replaced with the actual hardware elements as they become available. Activities in this testing level ultimately provide "acceptance" of the CSCIs as part of the acceptance of the aggregate system as demonstrated by performance of the Software Acceptance Test.

Preliminary instrument integration and test activities will take place on the hardware bench through the integration of individual engineering validation units (brassboards). Once the brassboards have been integrated and tested, the integration and test activities will move onto the system bench, which consists of the actual flight HMI instrument. The flight software is tested on the system bench by its use in the subsystem and system functional testing. A suite of flight software specific tests will be constructed. These will be executed at the subsystem or system level to ensure that the flight software requirements have been met.

To gain visibility into the flight software during the system integration and test process, a debug monitor resident in the processors with flight software may be used. Software development iterates between the CSCI integration and test and system integration test phases as system-level performance problems are assigned to software, corrected, and retested. After successful system test, software modules are controlled as part of the system baseline. Program-level change control and problem reporting are established for the source code at this time.

Final acceptance testing of the flight software CSCIs is conducted just prior to acceptance and environmental testing of the fully integrated flight Instrument. This phase is concluded by a successful Software Acceptance Test which establishes the acceptability of the software CSCI. Upon completion of this phase, the flight software is considered operational.
7.6 **Software Item Qualification Test**

The CSCIs will undergo qualification tests at the appropriate point in their development. The qualification tests will be as follows:

- **Pre-burn test of boot code and EGSE:** With brassboard processor with SUROM in EEPROM, brassboard motherboard and brassboard 1553 card and with EGSE supporting 1553 communications only using the GSFC-supplied spacecraft simulator.
- **Post-burn test of boot code and EGSE:** With flight processor, flight motherboard and flight 1553 card and with EGSE supporting 1553 communications only using the GSFC-supplied spacecraft simulator.
- **Application code and EGSE:** On flight HMI at the end of instrument integration just prior to start of instrument-level environmental testing with complete EGSE including the GSFC-supplied spacecraft simulator.
Appendix A. Abbreviations and Acronyms

A Angstrom
ASCII American Standard Code for Information Interchange
ATC Advanced Technology Center, a part of LMMS
AVTEC
BaE British Aerospace Electronics
CCD Charge Coupled Device
CCSDS Consultative Committee for Space Data Systems
CD Compact Disc
CDR Critical Design Review
CM Configuration Management
COTS Commercial Off The Shelf
CPU Central Processing Unit
CSCI Computer Software Configuration Item
CSC Computer Software Component
DAI Developer Anomaly and Improvement
DVD Digital Video Disc
EEPROM Electrically Erasable Programmable Read Only Memory
EGSE Electrical Ground Support Equipment
EM Engineering Model
FEP Front End Processor
FPA Focal Plane Assembly (mechanical structure used to hold the CCD)
FPP Focal Plane Package (Solar-B)
FSW Flight Software
GFSC Goddard Space Flight Center
GPIB General Purpose Interface Bus
GSE Ground Support Equipment
GUI Graphic User Interface
HIRDLS HIgh Resolution Dynamics Limb Sounder
HMI Heleioseismic and Magnetic Imager
HWCI Hardware Configuration Item
IAR Instrument Anomaly Report
IDE Integrated Development Environment
IGR Inter Group Review
ITAR International Traffic in Arms Regulations
IWS Instrument Workstation
K Thousand
LMMS Lockheed Martin Missiles & Space
LMSAL Lockheed Martin Solar Astrophysics Laboratory
MDI Michelson Doppler Imager
MVCS Multi Version Control
NASA National Aeronautics and Space Administration
NT
Appendix B. HMI Software Organization

- P. Scherrer
  Principal
  Investigator

- R. Bush
  HMI Program Mgr

- L. Springer
  LMSAL HMI Program Manager

- J. Drake
  Software Lead

- E. McFeaters
  Mission Assurance Mgr

- K. Kao
  ATC SQE

Flight Software

- Application Software
  G. Heyman
  R. Rehse

- Boot Software
  R. Baraze

Ground Software

- EGSE Software
  R. Chevalier

- S/S GSE Software
  R. Baraze

- STOL Procs
  J.-P. Riley