STEREO

SEP LET and Central MISC Processors

Flight Software Development Plan

Version G – 12/6/2001

STEREO
SEP LET and Central MISC Processors
Flight Software Development Plan

LET-SEPCentral-SoftwareDevelopmentPlanF.doc
Version G – 12/6/2001

Approved by:

_____
Rick Cook, Caltech Space Radiation Laboratory

_____
Alan Cummings, Caltech Space Radiation Laboratory

_____
Andrew Davis, Caltech Space Radiation Laboratory

_____
Richard Mewaldt, Caltech Space Radiation Laboratory

_____
Edward Stone, Caltech Space Radiation Laboratory

_____
Tycho von Rosenvinge, Goddard Space Flight Center

_____
David Curtis, Berkeley Space Sciences Laboratory

Document Revision Record

Rev.	Date	Description of Change	Approved By
A	2001-July-24	Preliminary Draft	-
B	2001-Aug-21	Incorporate Software Requirements Doc and reorganize per Project template	-
C	2001-Sept-04	Updated figures, added SEP Common Software Material	-
D	2001-Nov-05	Deleted HET and SIT material – now in separate document. Fleshed out sections 3.3, 4.2, and 4.3	-
E	2001-Nov-27	Moved Software Requirements to Separate Document. Added resource estimates for CPU, EEPROM, RAM. Added details of build-plan, manpower, schedule. Other misc. updates.	-
F	2001-Dec-03	Rewrite Test Plan - section 4.3.3. Update schedule. Update resource requirements tables. Reword to reflect that SEP is an instrument with four sensors.	-
G	2001-Dec-05	Update block diagram, update section 5.5, incorporate markups from ace and wrc.	

Table of Contents

1. Overview

1.1. *Introduction*

The IMPACT SEP instrument consists of four sensors – LET, HET, SEPT and SIT. Four microprocessors are dedicated to controlling, interfacing, and acquiring data from these sensors. This document defines the development plan for the flight software that will reside in the LET and SEP Central microprocessors. A separate document defines the development plan for the flight software that will reside in the HET and SIT microprocessors.

Flight software for the LET and SEP Central microprocessors will be developed at the Caltech Space Radiation Laboratory (SRL). Previously, the Caltech group developed the flight software for the SIS and CRIS instruments on ACE, and for several balloon instruments.

1.2. *Document Conventions*

In this document, TBD (To Be Determined) means that no information currently exists. TBR (To Be Resolved) means that a statement is preliminary. In either case, the acronym is typically followed by the initials of those responsible for providing the information.

1.3. *Applicable Documents*

Some of these documents and drawings can be found on the Berkeley STEREO/IMPACT website: http://sprg.ssl.berkeley.edu/impact/dwc/
Others are available or will be available from Caltech SRL.

1. Phase A Report/PAIP (Performance Assurance Implementation Plan)
2. LET Science Requirements Document
3. SEPT Science Requirements Document (TBD-SEPT Team)
4. LET and SEP Central Software Requirements Document
5. P24 MISC processor manual
6. IMPACT Intra-Instrument Serial Interface ICD
7. SEP Sensor Suite ICD (TBD-Kecman/WRC)
8. P24 MISC G-buss I/O Interface Document (TBD-WRC)
9. IMPACT/Spacecraft ICD
10. LET Science Data Frame Format Specification
11. SEP Central MISC Flight Software User Manual
12. LET MISC Flight Software User Manual
13. SEPT FPGA Data Sheet
14. LET and SEP Central Acceptance Test Plan and Report

1.4. *Acronyms*

ACE Advanced Composition Explorer
DPU Data Processing Unit
EEPROM Electronically Erasable Programmable Access Memory
ETU Engineering Test Unit
GSE Ground Support Equipment
HET High Energy Telescope
ICD Interface Control Document
IMPACT In situ Measurements of Particles and CME Transients
LET Low Energy Telescope
MISC Minimal Instruction Set Computer
RAM Random Access Memory
SEP Solar Energetic Particles
SIT Suprathermal Ion Telescope
SEPT Solar Electron Proton Telescope
SRL Space Radiation Laboratory

2. Host System and Interfaces

2.1. *System Overview*

The LET, HET and SIT sensors each require a dedicated microprocessor for onboard data processing. The microprocessor used for LET will be the P24 MISC (Minimal Instruction Set Computer), described below and in Reference 5. Processed data from the microprocessors associated with the three sensors will be gathered by the SEP Central MISC (also a P24 MISC processor), and formatted for transmission to the IMPACT DPU (per Reference 6 ICD). The SEPT sensor does not have a dedicated microprocessor, and data from SEPT will flow directly to the SEP Central MISC. Some processing of SEPT data will occur in the SEP Central MISC before the data are formatted and transmitted to the IMPACT DPU. Figure 1 shows a block diagram of the SEP Sensor Suite.

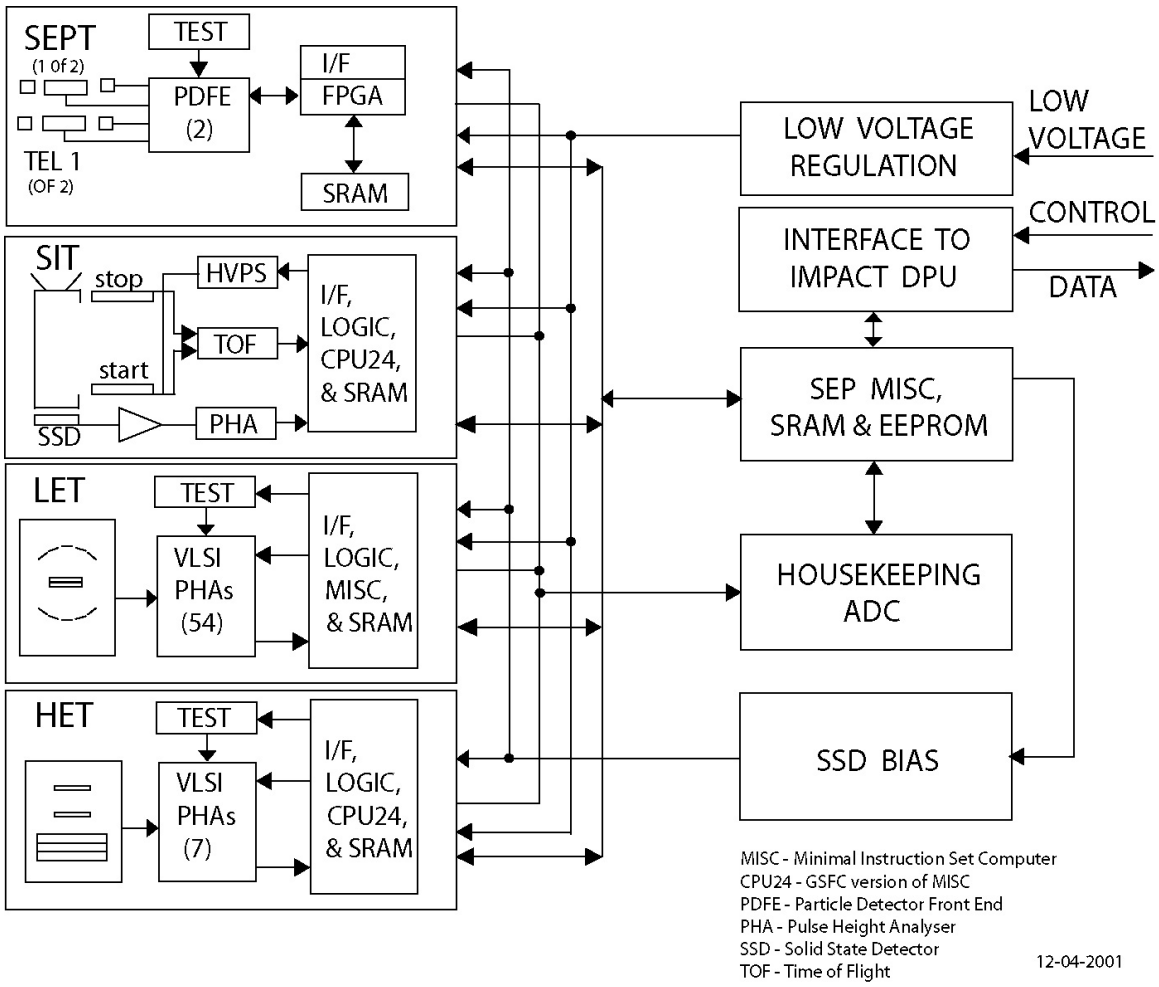

Figure 2.1: SEP Sensor Suite Block Diagram

2.2. *MISC Microprocessor*

The P24 MISC has a 24-bit CPU core with dual stack architecture intended to efficiently execute Forth-like instructions. The processor design is simple to allow implementation within field programmable gate arrays. For the LET and SEP Central applications, the MISC is implemented in the ACTEL 54SX72A FPGA. The MISC can be clocked at speeds ranging from 4 to 10 MHz (tested at 16MHz also).

2.2.1. **Code Memory**

MISC development boards are provided with 128K×24 SRAM and either 128K×24 EEPROM. The MISC can boot either from EEPROM or over a serial link (configurable via jumper). During boot, system software is copied from EEPROM (or serial link) to SRAM and the software runs from SRAM.

2.2.2. Memory Map

The MISC is capable of addressing a memory page of 256Kwords (each word is 24 bits). Other pages can be reached by pushing a 24-bit address on the return stack and executing the RET instruction, as described in Reference 5, but this capability is not expected to be required for STEREO (128K words of SRAM per MISC will be adequate).

2.2.3. I/O Bus (G-Buss) Peripherals

G-buss peripherals will be described in detail in Reference 8.
Currently, G-buss peripherals include an interrupt control and status register at address 0, supporting seven prioritized interrupts. Two of these are currently in use to support RS232 serial I/O. G-buss functions that will be added include:

1) A timer to produce periodic interrupts.
2) I/O ports.
3) Additional serial I/O UARTs.

2.2.4. Watchdog Timer

The processor includes a watchdog timer system that will be used to detect software crashes and reset the processor.

2.2.5. Operating System

A Forth operating system with an embedded optimizing forth compiler is implemented. Multi-tasking is implemented via a round-robin system similar to the system implemented in the Harris RTX2010 microprocessor.

2.2.6. Boot PROM

Included in the FPGA implementation of the MISC are 16 words of prom that currently hold a small program to boot over the serial link. Alternatively, booting can occur directly from external EEPROM. The boot method is selected with a jumper on the MISC development board. In serial boot mode, after initial power up, the MISC expects to receive a certain number of bytes over the serial link. Every three bytes received are packed into a 24-bit word, with the first byte going into the most significant slot and the third byte going into the least significant slot. Words are stored beginning at address 1 in SRAM. Execution begins at address 1 following the serial transmission. The boot from EEPROM is similar, however the boot code itself is stored beginning at address $20001 near the start of the EEPROM. This boot program copies a certain number of words from EEPROM starting at $20010 to SRAM starting at address 1. After the copy it jumps to address 1. The serial versus EEPROM boot behavior is achieved by altering the memory map depending on the jumper. The MISC starts at address 0 after power on reset, so address 0 is always mapped to the first location available in the internal ACTEL prom. This prom location currently contains a jump to location $20001. Addresses $20001 through $2000F are mapped either to the internal prom, for serial boot, or to the external EEPROM, for EEPROM boot.

2.3. *External Interfaces*

2.3.1. Interface between the SEP Central MISC and the IMPACT DPU

The SEP Central MISC will interface with the IMPACT DPU via a serial interface as defined in Reference 6 (prepared by Dave Curtis of UCB). Specific commands, telemetry and data formats transferred over this interface will be defined in References 6 and 7.

2.3.2. Interface between the SEP Central MISC and the SEP Sensors

There will be two serial interfaces between the SEP sensors and the SEP Central MISC. The first interface will be bi-directional, for transferring boot-code, commands, and command responses. The second interface will be uni-directional, for transferring data from the instruments to the SEP Central MISC. These interfaces will be defined in Reference 7.

2.4. *Hardware/Software Interfaces*

The hardware/software interfaces for the SEP Central and LET MISCs (and software routines that will be used to interface to the MISC hardware) will be defined by Rick Cook of Caltech and documented in References 8, 11 and 12.

3. Software Requirements

3.1. *Top Level Requirements*

Caltech will develop two software packages for STEREO IMPACT:

1. SEP Central Software
2. LET Software

The SEP Central software consists of the software routines unique to the SEP Central MISC, including the software dedicated to onboard processing of SEPT data. The LET software package consists of the software routines running on the LET MISC.

In addition to the two software packages above, Caltech will also develop and maintain a version of the Forth operating system for the LET and SEP Central MISCs that implements all standard Forth words necessary to implement flight software for LET and SEP Central. This Forth system will allow for the use of assembly-language subroutines where necessary for performance considerations. Associated with this Forth system, Caltech will develop an API that will enable access to the MISC hardware I/O interfaces. A round-robin software multi-tasking environment for the MISC will also be provided. All these items (the Forth operating system for the MISC, the I/O API, the multitasking software) are being developed at Caltech for other missions and projects as well as

STEREO. Therefore the software development plan for these items is not described in this document.

Detailed software requirements for LET and SEP Central are defined in the LET and SEP Central Software Requirements document (Reference 4).

3.2. *System Resource Requirements*

Tables 1 and 2 list estimated microprocessor and RAM resource requirements for LET and SEP Central, extracted from draft software requirements and based on previous experience and analysis. The estimates assume a MISC running at 8 MHz in both cases.

Table 1: LET System Resource Requirements

Task	Processor Cycles, %	Code Size, kwords*	Buffer Memory, kwords*
Operating System	4%	2	2
Data Acquisition	5%	2	3
Data Processing	35%	8	42
Data Formatting	5%	3	1
Command Processing	1%	2	1
Total	**50%**	**17**	**49**
Available	100%	128kwords	
% Usage	**50%**	**52%**	

*Note: for the MISC, one word = 3 bytes

Table 2: SEP Central System Resource Requirements

Task	Processor Cycles, %	Code Size, kwords*	Buffer Memory, kwords*
Operating System	4%	2	2
Data Acquisition	10%	2	10
SEPT Data Processing	20%	4	4
Data Formatting	5%	3	1
Beacon Data Processing	5%	2	1
Command Processing	1%	2	1
Total	**45%**	**15**	**19**
Available	100%	128kwords	
% Usage	**45%**	**27%**	

*Note: for the MISC, one word = 3 bytes

Each MISC can accommodate up to 256kwords of EEPROM. In addition, the large matrix lookup tables can be compressed by a factor of three before they are written to EEPROM, so EEPROM requirements are not as large as RAM requirements. With 256 kwords of EEPROM in SEP Central, and given that LET, HET, and SIT will all boot via

the serial link, we estimate a 50% EEPROM margin for the SEP instrument suite. If GSFC decides to implement EEPROMS for HET and SIT, the margin will be higher.

4. Software Development

"Top-down" and "Bottom-up" approaches to software development will run in parallel for the LET and SEP Central MISCs. The Top-down approach can be divided into four phases:

1. Requirements definition and analysis
2. Design
3. Implementation
4. System testing and acceptance testing

The activities occurring during each of these phases are detailed in below. During the requirements definition and design phases, the following Bottom-up activities will occur:

1. Gain familiarity with MISC processor, using small test routines
2. Gather together a suitable set of tools for MISC software development, including MISC simulator, serial communication software, version control software, etc.
3. Verify Forth system on MISC with standard software test suite
4. Prototype onboard processing algorithms to verify feasibility of MISC hardware approach

By the time the implementation phase of the Top-down approach begins, the Bottom-up approach will have resulted in a stable hardware platform and operating system, and software development tools adequate for implementing the software design.

4.1. *Top-down Software Development Phases*

Although the development phases listed below can be thought of as dividing the software development period into consecutive non-overlapping time periods, activities associated with one phase may be performed in other phases, e.g. most design activity will occur during the design phase, but some preliminary design work may be performed during the requirements definition phase.

4.1.1. Requirements Definition and Analysis Phase

During this phase, Caltech will develop a set of science requirements for LET. These requirements will be recorded in the LET Science Requirements Document (Reference 2). At the same time, Caltech will also develop preliminary designs of the sensor and its front-end electronics.

Using the science requirements, the preliminary instrument and electronics designs, the IMPACT ICD (Reference 6) and consultations with other SEP and IMPACT team

members, Caltech engineers and developers will derive a set of software requirements and interface specifications for the LET MISC and the SEP Central MISC. These requirements and specifications will define what data flows into and out of each MISC, the operations each MISC will perform on the data, and the interactions that will occur between the SEP Central MISC and the microprocessors in the LET, HET and SIT sensors. The specifications will also define the relevant properties of the host system, such as memory requirements, I/O peripherals, safing and reliability requirements, etc. All these requirements will be defined in the LET and SEP Central Software Requirements document (Reference 4).

4.1.2. **Design Phase**

During this phase, software developers will define the software architecture that will meet the software requirements. The requirements will be sorted into subsystems and all internal and external interfaces will be defined. The design phase will result in a set of design specifications that will include the following:

- Functional design diagrams
- Detailed descriptions of all inputs, outputs and data formats
- Data processing algorithms
- SEP sensor suite ICD
- P24 MISC processor manual
- P24 MISC G-buss I/O Interface Document
- Command lists
- Other TBD design specs

4.1.3. **Implementation Phase**

In this phase, developers will build software from the design specifications. The software will be written in Forth and assembly language. Assembly language will be used when optimization is required for performance reasons. Coding guidelines and standards used on the ACE mission will be carried over to this project.

4.1.4. **System Testing and Acceptance Phase**

End-to-end sensor, electronics, and software functional tests will be done using accelerator tests, radiation sources, and built-in self-test routines and test procedures. Software walkthroughs will occur at software peer reviews, and reports will be presented at PDR and CDR.

4.2. ***Development Environment and Equipment Needed***

The software development environment for the LET and SEP Central MISCs will consist of PCs running Windows 2000 and Win32Forth. A complete MISC simulator running under Win32Forth currently exists and is used to test code independent of MISC hardware.

The Forth system running on the MISC can be controlled via an RS232 serial link to a PC. Code can be uploaded via this serial link and so development/test of software can easily take place directly on the MISC.

4.3. *Product Assurance*

The following is based on the approach taken on ACE for software configuration control and flight software integrity assurance. This same approach will be taken for the STEREO SEP software development at Caltech.

4.3.1. Software Configuration Management/Backup Plan

- Only one person will be authorized to load flight software into flight CPUs – the lead electrical engineer.
- The lead engineer will maintain separate directories for the LET and SEP Central flight software. The directory name will include the date the code was last modified.
- At the beginning of any day on which the lead engineer will be working on the flight software, the flight software directories will be duplicated, and the new directories renamed with the current date. Any changes will be made within the new directories.
- Beginning in the implementation phase, a Software Development Log will be maintained by the lead engineer containing the date of any change, a description of the change, and the reason for the change.
- At the end of each day of software work, the lead engineer will backup the current flight software directories to a different computer and/or to removable media.
- One or two software developers will be working under the lead engineer. Each developer will be required to implement version control for his/her portion of the code.
- Periodically, a software developer will deliver his/her portion of the code to the lead engineer, who will incorporate it into the main software package.
- There will never be more than one person working on the same piece of code. If a change is required, the responsible developer will make the change and re-deliver new code to the lead engineer.
- During the EEPROM burn-in process, checksums will be calculated and recorded. During boot of a flight CPU, the EEPROMS will be read and same checksum will be calculated and typed out. Formal checkout procedures will include recording these checksums and verifying the proper values.

4.3.2. Walkthroughs

A walkthrough of mission critical code shall be conducted some months before acceptance testing of the flight code. Project personnel will be invited to this review, in addition to IMPACT team members not involved in the writing of the code.

4.3.3. Test Plan

Software tests will start at the module level. As the code builds up, system-level testing will start, and the majority of the test time will be targeted at the system level. Formal acceptance tests will be performed on the LET sensor, including the flight software, during the SEP integration period. Similarly, SEP-Central and its associated software will undergo acceptance tests as a unit, and as part of the SEP sensor suite. During environmental tests and suite level testing, more experience and test time with the flight software and real sensor data will be gained, and changes are expected. Any subsequent changes in the flight software will result in a repeat of these acceptance tests. The tests shall be designed to verify each of the software functional requirements as called out in the requirements documents. The SEP Acceptance Test Plan shall be reviewed internally and with Project personnel prior to the start of testing. Results of the acceptance tests will be documented in a SEP Acceptance Test Report.

A preliminary requirements/test matrix will be provided in the in the SEP Acceptance Test Plan and a final requirements/test matrix will be provided in the SEP Acceptance Test Report.

4.3.4. Software Problem Reporting and Tracking

Prior to installation in flight hardware, problems shall be documented and tracked in the Software Development Log. Once the software has passed acceptance tests and is installed in flight hardware, the common problem reporting and tracking system used by the flight hardware will be used, as described in the PAIP (Reference 1).

4.3.5. Risk Assessment

Loss of the lead engineer Rick Cook would have significant schedule impacts.

4.3.6. Software Maintenance

After LET and SEP Central are delivered, the ETUs together with the supporting GSE shall be maintained at Caltech. Any problems can be recreated and diagnosed in this environment. Code changes shall be run through an acceptance test prior to loading into the flight hardware. Every effort will be used to keep the flight software programmers available (though probably on another program) to make any necessary changes through the life of the program. Software documentation will be adequate to allow another programmer to understand and make changes to the software if necessary.

5. Management Plan

5.1. *Build Plan*

Code shall be developed using many incremental builds. The Forth environment is extremely modular in nature, with complex software structures being built from many smaller, simpler structures that are independently developed and tested. However, the following software development milestones can be identified:

- Build 1 – Support Initial Logic Board. Includes Forth OS, VLSI and Actel interface diagnostics. Preliminary data processing routines
- Build 2 – Support LET EM integration. Commands and telemetry processing
- Build 3 – Support SEP integration. Final data processing routines
- Final Build – Final Flight software

5.2. *Reviews*

- A Software Requirements Review shall be held with the SEP team STEREO project personnel attending to ensure that the documented requirements are complete and clear.
- At Instrument CDR, the Software Design shall be reviewed
- An Critical Software Walkthrough shall be held with project personnel a few months before the SEP acceptance tests are scheduled.

At each review, action items will be recorded and addressed.

5.3. *Documents and Source Code*

The LET and SEP Central Flight Software shall be documented by:

1. P24 MISC processor manual
2. IMPACT Intra-Instrument Serial Interface ICD
3. SEP Sensor Suite ICD
4. P24 MISC G-buss I/O Interface Document
5. LET Science Data Frame Format Specification
6. SEP Central MISC Flight Software User Manual
7. LET MISC Flight Software User Manual
8. SEPT FPGA Data Sheet
9. Developers Software Development Log
10. Source code comments

5.4. *Heritage and Reuse*

Previously, the Caltech group developed the flight software for the SIS and CRIS instruments on ACE, and for several balloon instruments. The GSFC group developed the flight software for the EPACT instrument suite on WIND. Many algorithms and software routines developed for thee projects will be reused for STEREO SEP.

5.5. *Interaction between Caltech and GSFC*

GSFC will develop flight software for the SIT and HET microprocessors. An ICD (Reference 7) will describe the communications interface between the SEP sensors and the SEP Central MISC, with adequate information to allow GSFC to design and build hardware and software that utilizes the interface.

Prior to SEP integration, Caltech personnel will visit GSFC with a SEP Central MISC to test the interface between HET, SIT and the SEP Central MISC. This would occur after software build 2.

5.6. *Staff and Schedule*

Lead engineer and software developer: Rick Cook, Caltech SRL
Software Developer: Andrew Davis, Caltech SRL.

Software maintenance following delivery of the IMPACT suite to the spacecraft will be on an as-needed basis, not included in the table below.

LET and SEP Central Software Development Schedule **11/29/2001**

Task Description	**Hours**	**Start**	**Finish**	**Duration**
Flight software	2,366.4 hrs	08/15/01	04/20/04	140 wks
Write preliminary software dev plan	48 hrs	08/15/01	11/27/01	15 wks
Davis	48 hrs	08/15/01	11/27/01	
Write software requirements doc	28.8 hrs	11/14/01	01/15/02	9 wks
Davis	28.8 hrs	11/14/01	01/15/02	
Write data transfer protocol section of SEP ICD	20.4 hrs	01/02/02	04/30/02	17 wks
Cook	20.4 hrs	01/02/02	04/30/02	
LET software build 1	243.2 hrs	03/01/02	09/30/02	30.4 wks
Cook	121.6 hrs	03/01/02	09/30/02	
Davis	121.6 hrs	03/01/02	09/30/02	
SEP Central software build 1	243.2 hrs	03/01/02	09/30/02	30.4 wks
Cook	121.6 hrs	03/01/02	09/30/02	
Davis	121.6 hrs	03/01/02	09/30/02	
Prepare for software design review	16 hrs	04/22/02	04/26/02	1 wk
Cook	8 hrs	04/22/02	04/26/02	
Davis	8 hrs	04/22/02	04/26/02	
Write final software dev plan	16 hrs	10/21/02	10/25/02	1 wk
Cook	8 hrs	10/21/02	10/25/02	
Davis	8 hrs	10/21/02	10/25/02	
LET software build 2	312 hrs	01/01/03	04/01/03	13 wks
Cook	130 hrs	01/01/03	04/01/03	
Davis	182 hrs	01/01/03	04/01/03	
SEP Central software build 2	312 hrs	01/01/03	04/01/03	13 wks
Cook	130 hrs	01/01/03	04/01/03	

Davis	182 hrs	01/01/03	04/01/03	
LET software build 3	352 hrs	04/02/03	09/02/03	22 wks
Cook	176 hrs	04/02/03	09/02/03	
Davis	176 hrs	04/02/03	09/02/03	
SEP Central software build 3	352 hrs	04/02/03	09/02/03	22 wks
Cook	176 hrs	04/02/03	09/02/03	
Davis	176 hrs	04/02/03	09/02/03	
Software support of SEP integration	184 hrs	07/22/03	12/29/03	23 wks
Cook	92 hrs	07/22/03	12/29/03	
Davis	92 hrs	07/22/03	12/29/03	
Write flight software user manuals	12 hrs	07/22/03	07/28/03	1 wk
Davis	12 hrs	07/22/03	07/28/03	
Final flight software build	52.8 hrs	12/01/03	12/30/03	4.4 wks
Cook	17.6 hrs	12/01/03	12/30/03	
Davis	35.2 hrs	12/01/03	12/30/03	
Support acceptance tests of SEP sensor suite	78 hrs	01/21/04	02/24/04	5 wks
Cook	16 hrs	01/21/04	02/24/04	
Davis	62 hrs	01/21/04	02/24/04	
Software support of SEP accelerator tests	96 hrs	03/24/04	04/20/04	4 wks
Cook	48 hrs	03/24/04	04/20/04	
Davis	48 hrs	03/24/04	04/20/04	