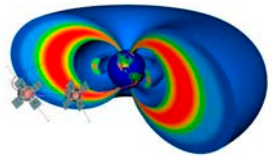


RBSP
Radiation Belt Storm Probes

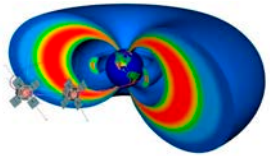
Use of a Simple EEPROM File System for the RBSP Spacecraft

Mark Reid

Johns Hopkins University, Applied Physics Lab

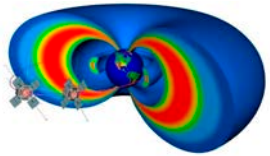
Mark.Reid@jhuapl.edu

Agenda



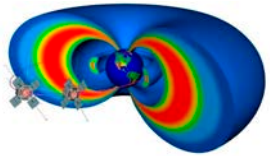
- Introduction
- RBSP Flight Software Overview
- Why use a File System for Application Code?
- Details of the RBSP File System
- Building the Initial File System Image
- Use and Benefits During Development
- Implications on Maintenance
- Plans for Future Missions
- Conclusion / Questions

Introduction



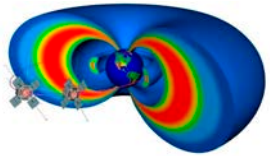
- **RBSP is a NASA mission to study near-Earth space radiation.**
- **The mission consists of two nearly identical spacecraft.**
- **Each spacecraft hosts 5 instrument suites, with a total of 10 unique science instruments per spacecraft.**
- **The Command and Data Handling (C&DH) Flight Software (FSW) for each spacecraft consists of 17 unique applications and 8 libraries built upon NASA's core Flight Executive (cFE).**
- **This presentation will discuss the use of a simple EEPROM file system for the storage and initialization of these libraries and applications.**

RBSP Flight Software Overview



- **Command and Data Handling (C&DH) software is composed of 17 applications**
- **Applications are implemented with NASA's core Flight Executive (cFE) application programmer's interface (API)**
- **Applications provide the following functionality**
  - Commanding
  - Autonomous Operations
  - Spacecraft Interface Management
  - Instrument Interface Management
  - Data Recording and Playback
  - Memory Management
  - Software Scheduling and Monitoring

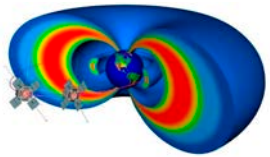
Why use a File System for Application Code?



- **Supports individual reload component applications**
  - Each application is a separate file in EEPROM
  - Code loads can be performed with protocols such as CFDP
- **Allows selective use of compression of code**
  - OS Uncompressed
  - Applications Compressed
- **Eases transition from development to flight**
  - Use of Network File System (NFS) during development

The RBSP team chose to use a simple slot-based file system, developed by NASA GSFC, to store applications as separate files in EEPROM.

Details of the RBSP File System



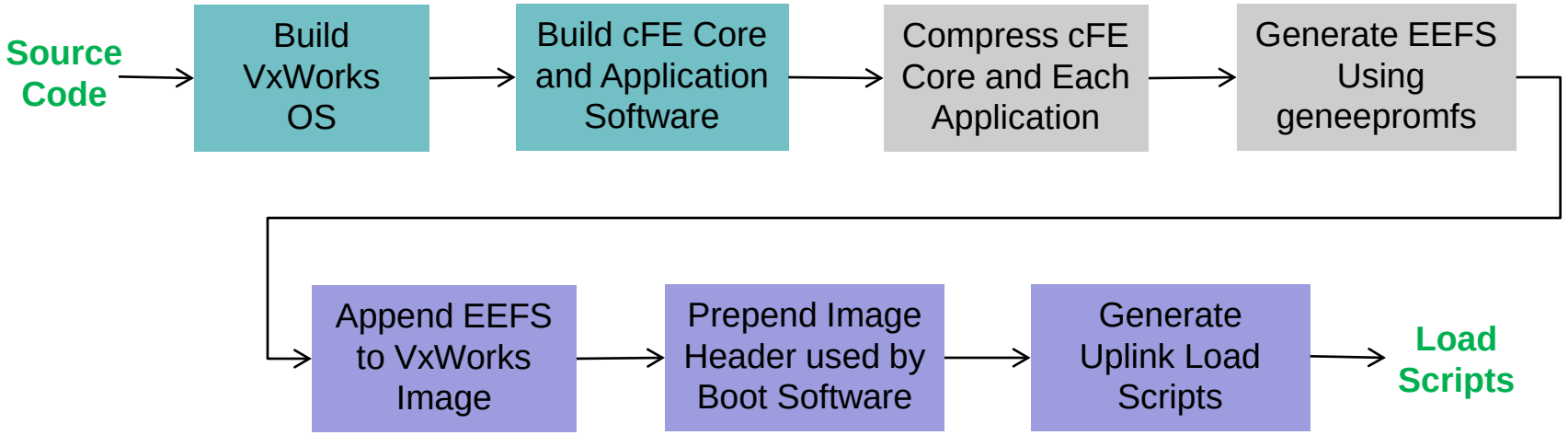
- **Developed by GSFC**
  - Thanks to Steve Slegel, Alan Cudmore
- **Configurable Slot-Based File System**
  - RBSP configuration:
    - 64 slots, 100 spare bytes per slot
    - Created “Read Only”
- **File Allocation Table (FAT)**
  - Fixed Size (based on max slots)
  - Contains location and max file size for each slot
- **File Headers for each slot contain information about each file**
  - File Name, File Size, CRC
  - Creation & Modification Dates
- **API integrates with Operating System**

Layout of File System

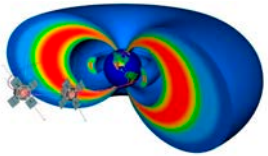
File Allocation Table (FAT)
File Header – Slot 0
File Data – Slot 0
File Header – Slot 1
File Data – Slot 1
…
…
File Header – Slot N
File Data – Slot N

Building the Initial File System Image

- **EEPROM File System images are created using the geneepromfs tool**
- **This command line tool:**
  - Reads an input file listing the files to be included in the File System
  - Outputs an EEPROM File System image ready to be written to EEPROM.
- **Make Process**

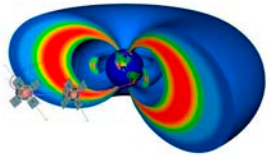


File System Use and Benefits During Development



- **NFS Used During Development**
  - Used for Developer Unit Testing
  - Mount Point is Dynamically set to Developer's Sandbox
  - Enables Rapid Build & Test Cycle
  - Enables Use of Network Development Tools (Workbench)
- **Local Linux File System Used for Faster Than Real-time Simulator**
  - Allows Flight Software to Run a Workstation
  - Mission Ops Tests Command Sequences Faster Than Real-Time
- **Statically Linked EEFS Used for Build**
  - Used for System Testing
  - Used for Independent Acceptance Testing
  - Used for Deployed Systems
    - Mission Operations Hardware Simulator
    - Autonomy Testing

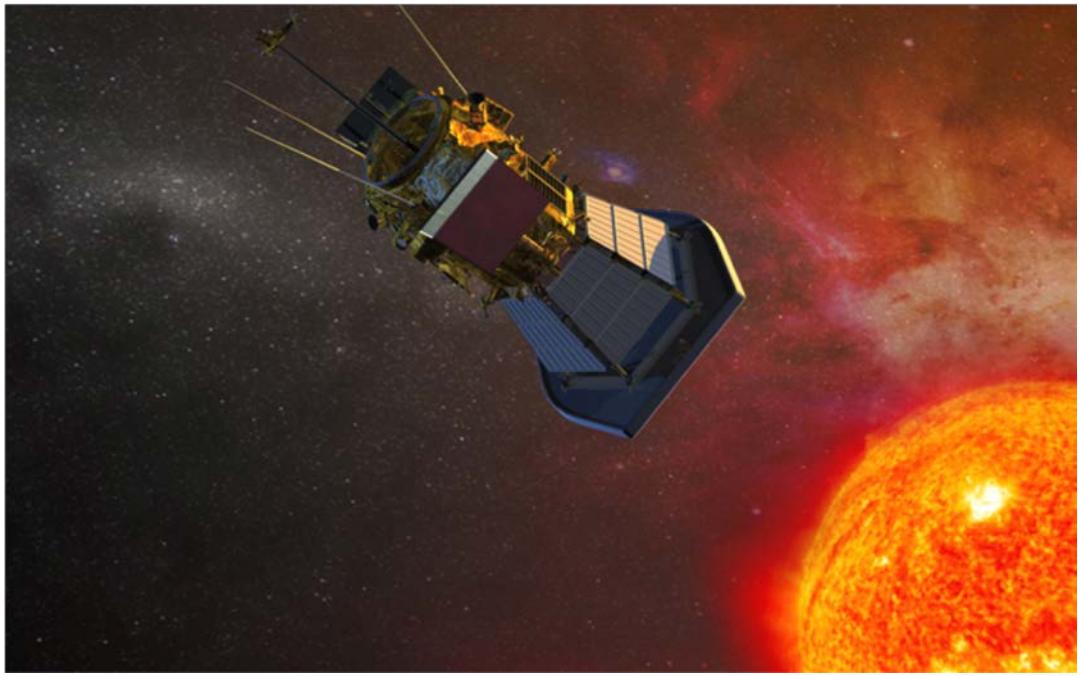
Implications on Maintenance



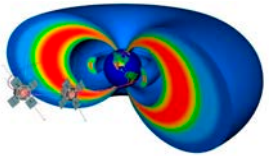
- **The use of a File System to Store Application Software**
  - Enables the use of Protocols Like CFDP for Uploads
  - Removes Need for Traditional Load/Dump/Compare Ground Software
  - Allows for Easier Modification of Smaller Components
  - Minimizes Uplink Time
- **However**
  - Modification of Components Requires Better CM and Testing
  - Components Can be Mixed-And-Matched
    - Requires CM at the Component (Application) Level
    - Requires Discipline in Keeping Components Decoupled
    - Required Discipline in Testing “Sets” of Components

Plans for Future Missions

- **Solar Probe Plus**
  - Using a File System for Application Code
  - Also Using the File System for Memory Objects (Tables)
  - Baselining CFDP Uplink



Conclusion / Questions



- **RBSP Took Steps to Lay the Building Blocks for Integrating cFE and the EEFS for APL Missions**
- **Again I would like to thank NASA GSFC for the efforts on both cFE and the EEFS used on RBSP.**
- **Questions?**