core Flight System (cFS)

A Low Cost Solution for SmallSats

David McComas
Susanne Strege
Jonathan Wilmot

NASA Goddard Space Flight Center
Software Engineering Division Flight
Software Systems Branch

August 2015
What is the core Flight System?

• The cFS is a re-usable spacecraft flight software architecture and software suite that is both platform and project independent

• Layered architecture and compile-time configuration parameters make it scalable and portable to a wide range of platforms

- Application Layer
- FSW Service Layer
- Platform Abstraction Layer

• Original product created by NASA’s Goddard Space Flight Center

• The FSW Service and Platform Abstraction layers are now controlled by a NASA multi-center configuration control board

August 2015
Recent cFS Success Stories

- Johnson’s Morpheus: 14 months from concept to flight test in 2010

- Goddard’s Class B missions: Global Precipitation Measurement (GPM) launched February 2014 and Magnetospheric Multiscale (MMS) launched March 2015

- Goddard’s 2014 Class D balloon mission: Observatory for Planetary Investigations from the Stratosphere (OPIS)
  - Baseline command and data handling software was up and running on the target platform (Intel Core Duo/Xenomai) within a month and launched 6 months later

- DARPA’s F6 program: Emergent funded (2013-2014) to develop Flight Software to Provide Autonomous Satellite Cluster Services
  - Cluster Flight System applications ported to cFS in less 6 months and formally demonstrated in simulation test bed
In Development - NICER

- Objectives
  - Reveal the nature of matter in the interiors of neutron stars
  - Uncover the physics of dynamic phenomena associated with neutron stars
  - Determine how energy is extracted from neutron stars

- On-board Processor
  - Broad Reach Engineering Radiation Hardened BRE440 PowerPC
    - 32 Bit RISC embedded processor
    - 83 MHz OSC (2 MIPS / MHz = ~166 MIPS)
  - VxWorks 6.7

Nicerg Dragon Deployment

August 2015
In Development - Dellingr

- **Objectives**
  - Low cost science and technology demonstration

- **On-board Processor**
  - ARM7
    - 40 Mhz, 2Mb RAM
  - FreeRTOS

![Diagram of Dellingr spacecraft with labeled components: UHF Antenna, Wheels and Radio, Electronics Cards, INMS, Boom Magnetometer, Thermal Louvers Experiment, In-Spacecraft Magnetometers.]
In Development - PiSat

- Objectives
  - Low cost test bed

- On-board Processor
  - Raspberry Pi
  - Raspberry Pi OS (DEBIAN/Linux)

August 2015
cFS Architecture Highlights
cFS Key Features

- **Layered architecture**
  - Reusable components
  - Platform Independent
  - Supports advances in technology without changes to the framework
cFS Core Services

Executive Services
- Manages the software system

Software Bus Services
- Provides publish/subscribe software bus messaging interface

Time Services
- Provides spacecraft time

Event Services
- Provides interface for sending, filtering, and logging event messages

Table Services
- Provides interface to manage table images

The cFS core layer is the system glue. It provides the common software functions that are needed by all missions.
## cFS Applications

<table>
<thead>
<tr>
<th>Application</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFDP</td>
<td>Transfers/receives file data to/from the ground</td>
</tr>
<tr>
<td>Checksum</td>
<td>Performs data integrity checking of memory, tables and files</td>
</tr>
<tr>
<td>Command Ingest Lab</td>
<td>Accepts CCSDS telecommand packets over a UDP/IP port</td>
</tr>
<tr>
<td>Data Storage</td>
<td>Records housekeeping, engineering and science data onboard for downlink</td>
</tr>
<tr>
<td>File Manager</td>
<td>Interfaces to the ground for managing files</td>
</tr>
<tr>
<td>Housekeeping</td>
<td>Collects and re-packages telemetry from other applications.</td>
</tr>
<tr>
<td>Health and Safety</td>
<td>Ensures that critical tasks check-in, services watchdog, detects CPU hogging, and calculates CPU utilization</td>
</tr>
<tr>
<td>Limit Checker</td>
<td>Provides the capability to monitor values and take action when exceed threshold</td>
</tr>
<tr>
<td>Memory Dwell</td>
<td>Allows ground to telemeter the contents of memory locations. Useful for debugging</td>
</tr>
<tr>
<td>Memory Manager</td>
<td>Provides the ability to load and dump memory.</td>
</tr>
<tr>
<td>Software Bus Network</td>
<td>Passes Software Bus messages over Ethernet</td>
</tr>
<tr>
<td>Scheduler</td>
<td>Schedules onboard activities via (e.g. HK requests)</td>
</tr>
<tr>
<td>Scheduler Lab</td>
<td>Simple activity scheduler with a one second resolution</td>
</tr>
<tr>
<td>Stored Command</td>
<td>Onboard Commands Sequencer (absolute and relative).</td>
</tr>
<tr>
<td>Telemetry Output Lab</td>
<td>Sends CCSDS telemetry packets over a UDP/IP port</td>
</tr>
</tbody>
</table>
A Complete Engineering Solution

Includes reusable:
- Requirements
- Source Code
- Design Documentation
- Development Standards
- Test Artifacts
- Tools
  - Unit Test Framework
  - Software Timing Analyzer
- User’s Guides
  - Application Developers Guide
  - API Reference Guides
  - Deployment Guides
  - Flight Operations Guides
- Command & Telemetry GUI

The CFS architecture reduces Non-Recurring Engineering (NRE) up to 90%
Component Based Architecture

The cFS architecture creates a Flight Software “App Store”.

Inter-task Message Router (Software Bus)

- Scheduler
- Limit Checker
- Memory Manager
- Space Wire
- Instrument Manager
- CFDP File Transfer
- Data Storage
- File Manager
- GN&C
- Mass Storage File System

- cFS Applications
- Mission Applications
- Core Services/Applications

1553 Bus Support
Telemetry Output
Command Ingest
Software Bus
Time Services
Executive Services
Event Services
Table Services

- 1553 Hardware
- Commands
- Communication Interfaces
- Real-time Telemetry File downlink

August 2015
## CFS Component Metrics

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>Logical Lines of Code</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Flight Executive</td>
<td>6.3.2</td>
<td>12930</td>
<td>General: 17, Executive Service: 46</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Event Service: 5, Software Bus: 29</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Table Service: 10, Time Service: 32</td>
</tr>
<tr>
<td>CFDP</td>
<td>2.2.1</td>
<td>8559</td>
<td>33</td>
</tr>
<tr>
<td>Checksum</td>
<td>2.2.0</td>
<td>2873</td>
<td>15</td>
</tr>
<tr>
<td>Data Storage</td>
<td>2.3.0</td>
<td>2429</td>
<td>27</td>
</tr>
<tr>
<td>File Manager</td>
<td>2.3.1</td>
<td>1853</td>
<td>22</td>
</tr>
<tr>
<td>Health &amp; safety</td>
<td>2.2.0</td>
<td>1531</td>
<td>45</td>
</tr>
<tr>
<td>Housekeeping</td>
<td>2.4.0</td>
<td>575</td>
<td>8</td>
</tr>
<tr>
<td>Limit Checker</td>
<td>2.2.1</td>
<td>2074</td>
<td>13</td>
</tr>
<tr>
<td>Memory Dwell</td>
<td>2.3.0</td>
<td>1035</td>
<td>8</td>
</tr>
<tr>
<td>Memory Manager</td>
<td>2.3.0</td>
<td>1958</td>
<td>25</td>
</tr>
<tr>
<td>Stored Commanding</td>
<td>2.3.0</td>
<td>2314</td>
<td>26</td>
</tr>
<tr>
<td>Scheduler</td>
<td>2.2.0</td>
<td>1164</td>
<td>19</td>
</tr>
</tbody>
</table>

- Two scopes of configuration parameters: mission or processor
- Configuration parameters span a large functional range from a simple default file name to a system behavioral definition like the time client/server configuration
Example Mission Code Metrics
Global Precipitation Measurement (GPM)

- Noteworthy items
  + cFE was very reliable and stable
  + Easy rapid prototyping with heritage code that was cFE compliant
  + Layered architecture has allowed COTS lab to be maintained through all builds
  - Addition of PSP changed build infrastructure midstream

- Lines of Code Percentages:

<table>
<thead>
<tr>
<th>Source</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAE</td>
<td>0.3</td>
</tr>
<tr>
<td>EEFS</td>
<td>1.7</td>
</tr>
<tr>
<td>OSAL</td>
<td>2.1</td>
</tr>
<tr>
<td>PSP</td>
<td>1.0</td>
</tr>
<tr>
<td>cFE</td>
<td>12.4</td>
</tr>
<tr>
<td>GNC Library</td>
<td>1.6</td>
</tr>
<tr>
<td>CFS Applications</td>
<td>23.5</td>
</tr>
<tr>
<td>Heritage Clone &amp; Own</td>
<td>38.9</td>
</tr>
<tr>
<td>New Source</td>
<td>18.5</td>
</tr>
</tbody>
</table>

August 2015
Building a cFS Community
<table>
<thead>
<tr>
<th>Organization</th>
<th>Contribution</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Johnson Space Center</td>
<td>Trick Simulator integration, Enhanced Build environment, Training materials, ITOS integration, multiple new platforms</td>
<td></td>
</tr>
<tr>
<td>Johnson Space Center</td>
<td>Class A certification of OSAL, cFE and selected cFS applications</td>
<td>Use in Orion Backup flight computer, video processing unit, and Advanced Space Suit</td>
</tr>
<tr>
<td>Johnson Space Center</td>
<td>Enhanced Unit tests and increased code coverage, new performance analysis tool</td>
<td></td>
</tr>
<tr>
<td>Glenn Research Center</td>
<td>Code Improvements, modern build environment (cmake), Electronic Data Sheet integration</td>
<td></td>
</tr>
<tr>
<td>Ames Research Center</td>
<td>cFS community configuration management services, continuous integration build services</td>
<td></td>
</tr>
<tr>
<td>Ames Research Center</td>
<td>Simulink Interface Layer for auto-coding cFS applications</td>
<td></td>
</tr>
<tr>
<td>JHU/APL</td>
<td>Multi-Core cFE/OSAL port</td>
<td>Joint IRAD with GSFC, will be used for GSFC MUSTANG flight processor card</td>
</tr>
<tr>
<td>DARPA/Emergent</td>
<td>Fractionated Spacecraft / Distributed Mission cFS applications</td>
<td>Part of DARPA F6 project, they hope to make the apps available as open source</td>
</tr>
<tr>
<td>Interns and misc contributors</td>
<td>cFS development tools are being created and shared by many organizations</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Miscellaneous bug fixes reported via open source sites.</td>
<td></td>
</tr>
</tbody>
</table>
Ongoings

Technical Enhancements

- Integrated Development Environment (IDE)
- Automated tests (unit, functional, build…)
- CCSDS EDS specifications for cFS components
- Integrate Multi-core support into OSAL and cFE
- Integrate/Merge ARINC653 port into OSAL and cFE
- Integrate Dellingr Cubesat FreeRTOS OSAL Port
- Improve scheduler time synchronization
- Expand SB namespace beyond $2^{11}$
- Lab upgrades
  - RTEMS 4.11 updates
  - VxWorks 6.9 updates
  - RAD750 simulator
  - MPC8377E: PowerQUICC II Pro Processor test beds
  - LEON3 test bed
  - MCP750 test bed

Operational Enhancements

- Formalize cFS user community
- Web based app store
Back Up
Acronyms

- API: Application Programmer Interface
- ARC: Ames Research Center
- BAT: Burst Alert Telescope
- CCSDS: Consultative Committee for Space Data Systems
- CDH: Command Data Handling
- CFDP: CCSDS File Delivery Protocol
- cFE: Core Flight Executive
- CFS: Core Flight System
- CMMI: Capability Maturity Model Integrated
- FSW: Flight Software
- GLAS: Geoscience Laser Altimeter System
- GN&C: Guidance, Navigation, and Control
- GPM: Global Precipitation Measurement
- GSFC: Goddard Space Flight Center
- JSC: Johnson Space Center
- LADEE: Lunar Atmosphere and Dust Environment Explorer
- LOC: Lines of Code
- LRD: Launch Readiness Date
- LRO: Lunar Robotic Orbiter
- MAP: Microwave Anisotropy Probe
- MMS: Magnetic Multiscale Mission
- NRE: Non-Recurring Engineering
- OSAL: Operating System Abstraction Layer
- RBSP: Radiation Belt Storm Probe
- RTEMS: Real-Time Executive for Multiprocessor Systems
- SAMPEX: Solar Anomalous and Magnetospheric Particle Explorer
- SARB: Software Architecture Review Board
- SDO: Solar Dynamics Observatory
- SMEX: Small Explorer
- ST-5: Space Technology 5
- SWAS: Submillimeter Wave Astronomy Satellite
- TRACE: Transition Region and Coronal Explorer
- TRL: Technology Readiness Level
- TRMM: Tropical Rainfall Measuring Mission
- WIRE: Widearea Infrared Explorer
- XTE: X-Ray Timing Explorer
Where is the cFS?

- cFE open Internet access at http://sourceforge.net/projects/coreflightexec/
  - Source code
  - Requirements and user guides
  - Tools

- OSAL open Internet access at http://sourceforge.net/projects/osal/
  - Source code
  - Requirements and user guides
  - Tools

- cFS application suite is also available on sourceforge
Questions? Contact:

- CFS 582 Branch Management
  - David McComas - NASA GSFC/Code 582 Flight Software Branch
    - david.c.mccomas@nasa.gov
    - 301.286.9038

- CFS Lead Architect
  - Jonathan Wilmot - NASA GSFC/Code 582 Flight Software Branch
    - jonathan.j.wilmot@nasa.gov
    - 301.286.2623

- CFS Product Development Lead
  - Susie Strege - NASA GSFC/Code 582 Flight Software Branch
    - susie.strege@nasa.gov
    - 301.286.3829

- OSAL Product Development Lead
  - Alan Cudmore - NASA GSFC/Code 582 Flight Software Branch
    - alan.p.cudmore@nasa.gov
    - 301.286.5809
Software Facts

<table>
<thead>
<tr>
<th>Class</th>
<th>Class A, B and lower instantiations</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRL</td>
<td>OSAL &amp; cFE TRL 9, selected cFS Apps TRL 9</td>
</tr>
<tr>
<td>CMMI</td>
<td>Certified level 2 for Class B (GSFC)  Certified level 3 for Class A (JSC)</td>
</tr>
<tr>
<td>Operating Systems</td>
<td>VxWorks, RTEMS, Linux, ARINC 653</td>
</tr>
<tr>
<td>Hardware Supported</td>
<td>MCP750, BAE RAD750, Coldfire, LEON3, MCP405, BRE440, and many more at JSC, GRC, ARC, MSFC, and APL</td>
</tr>
<tr>
<td>Lines of Code</td>
<td>45K (LOC)</td>
</tr>
<tr>
<td>Components available</td>
<td>13</td>
</tr>
<tr>
<td>Documentation Available</td>
<td>Requirements, User’s Guides, Deployment Guides, Design Documents, Test Plans, Test Reports</td>
</tr>
</tbody>
</table>

cFS is a software system designed to address software quality and usability issues of performance, reliability, reuse, maintainability, and lifecycle cost.

August 2015
What’s in the cFE open source tarball

- There are many other PSPs at each center that are not open source
  - We are compiling a list to be hosted on the NSCKN site
- Included are simple UDP/IP command and telemetry GUIs
What’s in the OSAL open source tarball

- There are other BSPs at each center that are not open source
  - We are compiling a list to be hosted on the NSCKN site