Growing Flight Software Hands-On Experience: cFS and Educational Outreach

Allen Brown, Mathew Benson
Odyssey Space Research, LLC

2016 Workshop on Spacecraft Flight Software
California Institute of Technology – Pasadena, CA
Company Background

- Odyssey Space Research LLC
  - Established 2003

- Core Areas
  - GN&C design, analysis, integration, evaluation, test
  - Flight software development, test
  - Simulation development, integration

- Current Principal Projects
  - SpaceX Dragon ISS Resupply/Commercial Crew
  - Orbital ATK Cygnus ISS Resupply
  - Orion Multi-Purpose Crew Vehicle
  - Visiting Vehicle Integration with ISS
  - Exploration Mission analysis and design
  - Flight Dynamics for Mission Operations
  - NASA Training Systems for Operations
Company Background

- LM Orion (MPCV) Flight Software
  - Development, Integration, Verification, Test
  - GNC, ECLSS, EPS, BFS...
- FSW Safety IV&V
  - Dragon, Cygnus
- Core Flight System (cFS)
  - Internal Research & Development
    - Integrated FSW Simulations
    - Unmanned Aircraft Systems (UAS) Platforms
    - Integrated Development Environments
    - Command and Telemetry UI, data management
    - Human-in-the-Loop Flight Mockup
  - NASA Support
    - Integrated FSW Simulations
    - Class A Certification (Human Rating)
    - New cFS platforms (Xenomai, ARINC 653)
    - cFS Voting Architecture
    - Distributed cFS Integration
    - New cFS Apps/Libs for Commands/Telemetry
    - Command and telemetry UI architecture & development

- Other S/W Projects with Outreach
  - SpaceLab for iOS
cFS Education Outreach

• “Jump in and learn by doing”
  - A structured, hands-on, set of demos and tutorials
  - Focus on basic cFS application development
  - Include the basics of cFS mission integration

• “Test like you fly, fly like you test”
  - Engage with cFS on flying hardware
  - Illustrate the elements of a minimal FSW development process
    • Test with cFS in an integrated simulation environment
  - Use the same command/telemetry tools between test and flight

• Course material aim
  - University-level
    • Quite useful for industry engineers as well
  - Assume some software development exposure
  - Include cFS docs & add reference material

✔ Interesting
✔ Usable
✔ Representative
✔ Available
Components & Packaging

• A Linux VM Image
  – Everything included for the outreach demos & tutorials
  – VM setup instructions

• User:
  – Laptop with WiFi
  – Parrot SA AR.Drone 2.0 quadcopter
  – *(Drone is somewhat optional, could be shared)*
Components & Packaging

- AR.Drone 2.0 (COTS)
  - 1 GHz ARM Cortex A8, Linux
  - WiFi a/b/n
  - 2 cameras, gyro, accelerometer
  - magnetometer
  - barometric pressure sensor
  - ultrasound altitude sensor

- cFS interacts via UDP over WiFi
  - Existing Interface, no drone firmware or GNC mods
    - Simpler, doesn't risk drone, easier to simulate
  - Run cFS in Linux VM or on the AR.Drone
Components & Packaging

- CentOS x86_64 Linux Distro
- cFS from SourceForge
  - PSP, OSAL, cFE, apps/libs
  - Source code, documentation
  - Script tools
- Trick Simulation (GitHub)
  - Drone simulation provided
- x86_64, x86_32 Linux toolchain
- ARM cross-compiling toolchain
  - On-target debug support
- Eclipse/CDT
- Demos, tutorials, references
  - Editable wiki inside the VM
- cFS Command & Display Tool
  - Yamcs Studio (cFS plugin)
  - Custom AR.Drone pages
- cFS Performance Monitor
- Minor mods for streamlining:
  - SCH table generation
  - ARM Linux PSP
  - SC Table generation
  - Builds on drone kernel
Screenshot: VM Desktop

2016 Workshop on Spacecraft Flight Software (FSW-16)
Screenshot: Provided Tools & Shortcuts

2016 Workshop on Spacecraft Flight Software (FSW-16)
Screenshot: Example Course Material
Screenshot: cFS Running With Simulated Drone

Not shown: Trick graphs and plots

2016 Workshop on Spacecraft Flight Software (FSW-16)
Screenshot: Example cFS Drone App Display Page
(Yamcs Studio w/ cFS plugin)
Content Overview

- Install & Configure VM
  - Connecting to the Drone
- Getting Started
- Overview of cFE and cFS Applications
- Demo: Hover
  - Basic build, deploy to drone
  - Use cmd/tlm interface
  - Fly the drone with cFS!
- Demo: Hover (Linux)
  - Build, run cFS with drone sim, same cmd/tlm tools
- Tutorial: Hello World
  - Create, integrate, build, execute
  - Test with cmd/tlm tools
  - Introduce tools/scripts
- Demo: Tables with Stored Command
  - Introduce cFE tables, via SC
- Tutorial: SC for Canned Maneuvers
  - Learn how to integrate an existing app (SC)
  - Create and run your own table
  - Test/debug in drone simulation

**Bold** = requires drone
Content Overview (cont'd)

- Demo: Loss of Comm Scenario
  - Showcase – in the sim!

- Tutorial: Add Loss of Comm Watchdog cFS App
  - A small custom cFS watchdog app to auto-land the drone
  - Create, integrate, build, debug, and test (in the sim!)

- Tutorial: Performance Monitoring
  - cFE Performance Monitor log entries, ID coordination
  - Using the cmd/tlm & GUI tools

- References:
  - Released cFS documentation
  - Build instructions, debugging, remote debugging on the target
  - cFE and cFS Yamcs cmd/tlm
  - Overview of cFS and the (supplied) ARDrone apps
  - cFS App Design Comments
  - cFS Operational Concepts
  - cFS App Integration Checklist
  - cFE Startup Script Info
  - Trick Sim References
Lessons Learned

- No such thing as “THE way” to make a cFS App
  - App structure, message response behavior
  - Relationship with: SCH, telem/HK, upstream & downstream apps

- cFS App development requires cFS integration minimum knowledge
  - Coordinating Message IDs, Perf IDs, etc.
  - Constructing/updating SCH tables, TO tables, etc.
  - Building, deploying, executing, commanding, debugging

- Command & telemetry tools are required from the start for development
  - Need to be easily tailored by the engineer

- “Polish” takes time & effort
  - “Local” audience vs. unknown audience
  - Wiki format looks like a great choice (in VM for now, room for growth)
  - The NASA release process does affect the design/build approach of an outreach package
  - A repeatable VM build system is critical

- Yes, expansion/refinement is possible on every single front...
Current Status

• Going through the process for NASA Public Release
  – Goal: Release under NASA Open Source Agreement (NOSA)

In progress...
...gears turning!
Thanks

• NASA
  – Dr. Lorraine Prokop, JSC
  – David McComas, GSFC

• Developers & Reviewers

• The Open-Source Communities
  – cFS, Trick software supporters
Thank you.

Q & A