Basecamp Introduction Tutorial

Objectives

- Introduce Basecamp features so users can quickly be productive
- Provide guidance on what to do next based on your goals

Lesson 1 Objectives

- Describe Basecamp’s objectives, components, and terminology

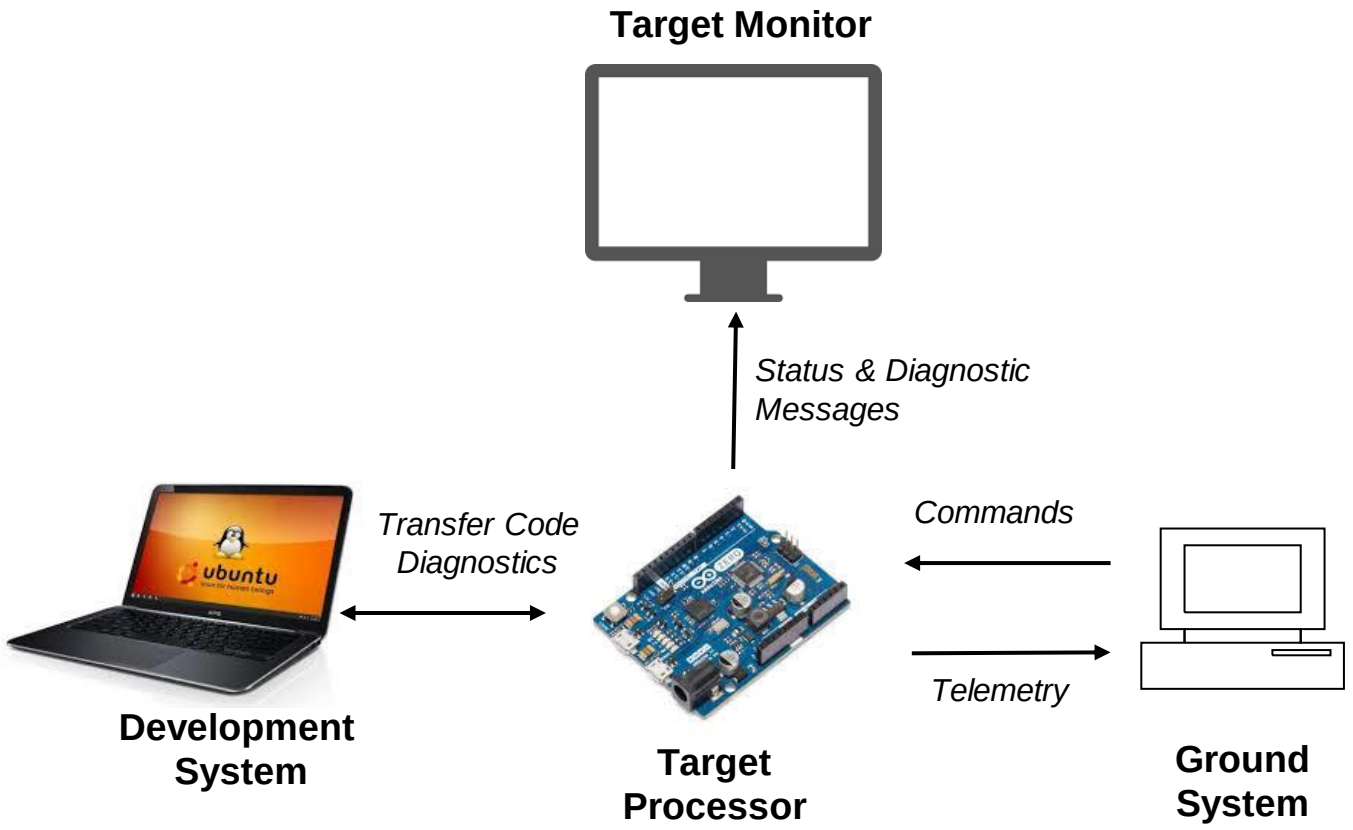
Notes

- This tutorial describes what is available. Other tutorials and documents provide details on workflows and how to acoomplish goals.

Why Basecamp?

- Basecamp provides a cFS architectural framework, build/runtime tools, and a lightweight GUI that simplify creating, integrating, testing, and deploying cFS applications
- Provides a foundation for users and educators to create cFS-based projects
- Command and telemetry routing design supports interfacing to external systems
- Supports the following application activities
  - Learn the cFS application architectural model
  - Learn Basecamp's application framework (heritage from OpenSatKit)
  - Develop new applications
  - Download apps from the github cfs-apps repositories
  - Integrate apps into Basecamp's cFS target
  - [future] Learn Basecamp's application packaging specification
  - [future] Certify new apps comply with Basecamp's packaging specification
- Not intended to be a fully functional ground system
- Basic command and telemetry GUI/script interfaces provide app development and runtime support
- [future]cFS build tools can be customized to generate command and telemetry definitions for different ground systems

Embedded Flight Systems Context



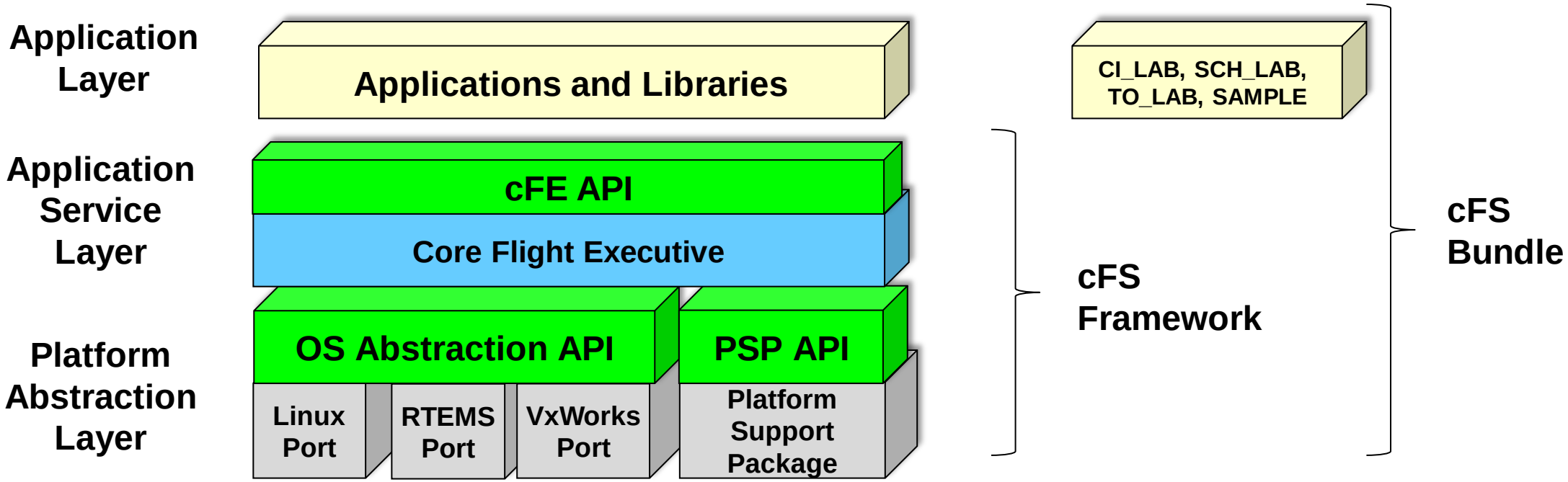
Target Processor
- A processor that runs the cFS target image

Development System
- Used to build and transfer the cFS target image to the target processor
- Requires a 'cross compiler' if the target process is different than the development system
- May include runtime diagnostic tools

Target Monitor
- A common diagnostic tool used to help verify the embedded system is operating correctly
- Often a monitor connected over a serial port

Ground System
- An application that sends command messages to the target and receives telemetry messages from the target
- The command & telemetry communications link may vary between test configurations and operations

core Flight System Context



- Platform Abstraction Layer ports to different operating systems (OS) / processor combinations
  - Contains the Operating System Abstraction Layer (OSAL) and the Platform Support Package (PSP)
- Application and libraries that only use the cFS APIs are portable across platforms
- The cFS Framework managed by NASA at https://github.com/nasa/cFE/
- The cFS bundle provides a starter system with a minimal runtime app suite, https://github.com/nasa/cFS

Basecamp Ecosystem

Learning Resources

https://www.youtube.com/channel/UC2wfvAIkrrgyC4ITwL3zokg

https://openmissionstack.com/

cFS App Repos

https://github.com/orgs/cfs-apps/repositories

cFS target runs as a Linux Process

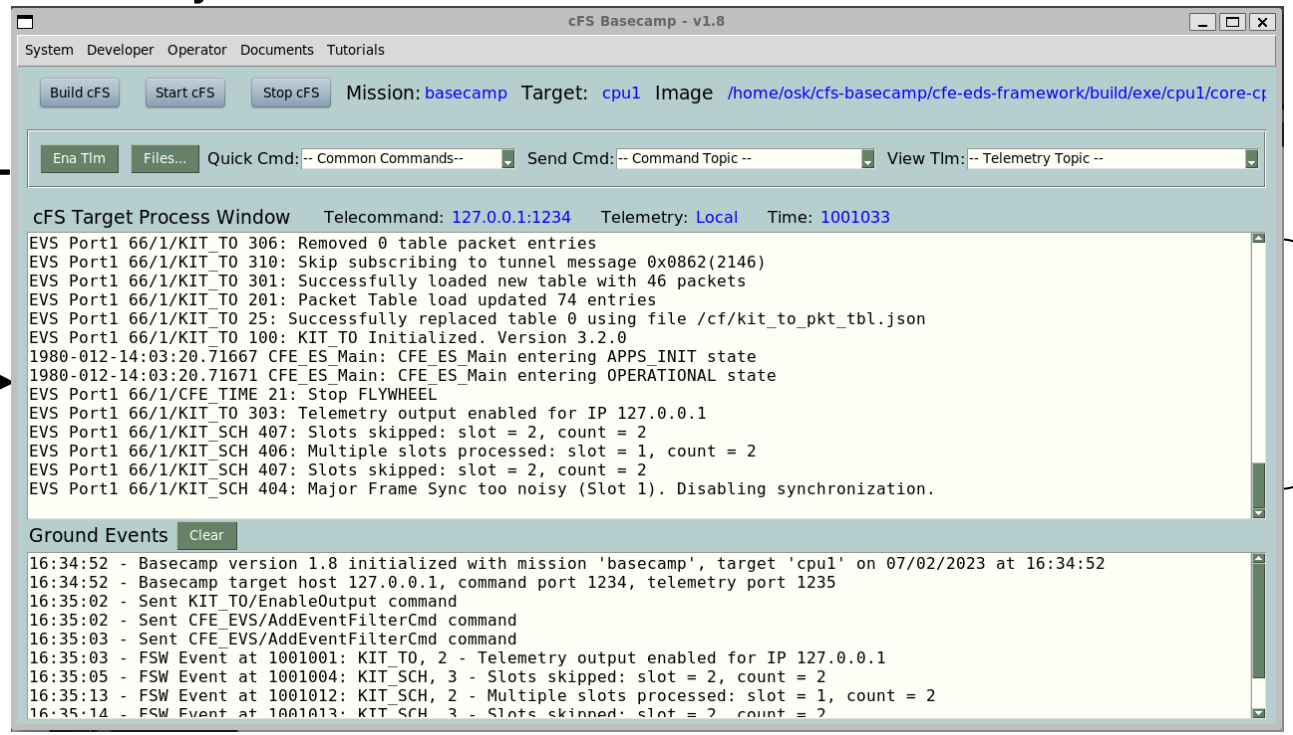
Apps

Commands

Telemetry

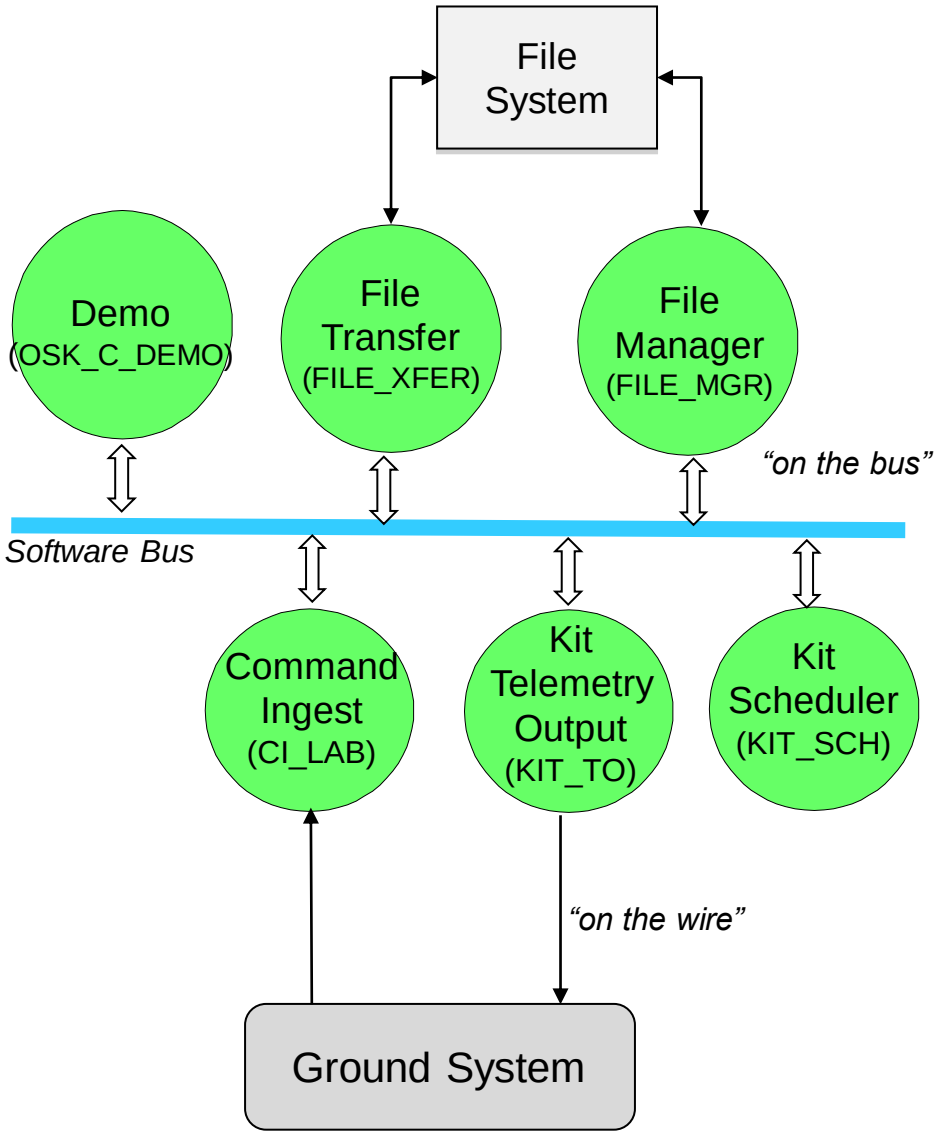
Status & Diagnostic Messages

Ground System



Target Monitor Display

Basecamp cFS Target Apps



- **Electronic Data Sheets (EDS) specs define command and telemetry messages**
  - *"on the wire"* ⟶ are off card interfaces
  - *"on the bus"* ⟺ are native host definitions

- **Basecamp comes preconfigured with 6 apps**
  - *CI_LAB* and *KIT_TO* manage external-to-internal message bus translations
  - *KIT_SCH* coordinates synchronous application functionality
  - *FILE_MGR* provides onboard directory and file management services
  - *FILE_XFER* manage file transfers between flight and ground
  - *APP_C_DEMO* is used for educational purposes

Basecamp Directory Structure