NOS3 + cFS Application-Layer Architecture

Apps, Timing, Software-Bus Multiplexing, and Integration with Vivado and PetaLinux

Joshua Wille

October 18, 2025

1 Overview

This document presents a systems-level description of the NASA core Flight System (cFS) running within the NOS3 simulation framework, emphasizing how applications communicate and stay synchronized. It describes the application suite, the role of the Software Bus (SB) as the application-layer multiplexer, and how timing behavior parallels the hardware clocking structure in Vivado/Vitis and PetaLinux runtime scheduling on Zynq-based platforms.

Reference: NASA core Flight System (cFS) GitHub: github.com/nasa/cFS.

2 Application Set (Roles and Nominal Cadence)

Table 1 summarizes typical flight applications, their rates, and their functional categories.

3 Timing Model

The timing structure is deterministic and hierarchical:

- **TIME** provides a master time reference, often disciplined by GPS or a ground sync command.
- **SCH** executes a 1 Hz schedule table, triggering apps at defined sub-frame offsets (e.g., 20 ms for ADCS, 1 s for HK).
- Each app publishes telemetry and events at its cadence, forming a software-level clock domain.
- Latency and jitter are limited by task priority and the underlying OS scheduler; both can be verified through housekeeping telemetry.

4 Software Bus as the Application-Layer Multiplexer

The cFE Software Bus (SB) acts as a message fabric connecting all applications. Each telemetry packet is a CCSDS Space Packet (SPP) with APID, sequence flags, and length. Producers publish on the SB at their scheduled rate; consumers such as TO, DS, or CF subscribe to selected MsgIDs.

- The **Telemetry Output (TO)** app is the active multiplexer: it subscribes to desired MsgIDs and forwards them via UDP or hardware channels.
- The **Command Ingest (CI)** app performs the inverse demultiplexing on uplink, injecting validated command packets onto the SB.

- Message order and timing reflect producer cadences defined in SCH, ensuring predictable interleaving.

This architecture mirrors real flight operations: when an RF or CCSDS link is added beneath, the same Space Packets are placed into fixed-size Transfer Frames (e.g., 1115-byte user field) with First Header Pointer (FHP) semantics preserved.

5 Application-Layer Data Flow

A nominal data flow sequence is:

1. **SCH** activates periodic tasks such as ADCS, EPS, Thermal, and Mode at their designated intervals.
2. Each task publishes telemetry (CCSDS packets) to the **SB**.
3. **TO** collects and multiplexes those packets for transmission via UDP or SDR-based RF downlink.
4. **CI** receives CCSDS commands from the ground and republishes them to relevant apps on the SB.
5. **BSD/CFDP** handle bulk data segmentation and delivery reliability.

6 Minimal Viable Stacks

Manual Operations (8 apps): CI, TO, HK, TIME, SCH, HS, FM/DS, plus ADCS/BSD.
Autonomous Operations (12 apps): Add Mode, EPS, Thermal, and Radio/Link for environment-aware scheduling.
Resilient Platform (16+ apps): Include CF (CFDP), TBL, EVS, Deployment, and persistent boot/restart logic.

7 Integration Notes

- Define **SCH tables** to enforce deterministic interleaving between high-rate and low-rate telemetry.
- Create multiple **TO routes** to isolate critical TT&C from bulk data.
- Assign higher priorities to HK, CI, TO threads, while throttling BSD or CFDP transfers during low-power modes.
- Bridge CI/TO to a GNU Radio chain over UDP sockets to test packet-to-frame behavior (1115 B payload per TM frame).

8 Analogy to Vivado/Vitis Clocking Wizard

The cFS timing hierarchy parallels the hardware clocking framework:

- **TIME** → Primary oscillator or reference clock.
- **SCH** → Derived clocks or dividers producing multiple frequency domains.

- **SB** → Clock distribution/AXI interconnect delivering synchronized data between logic domains.
- **TO/CI** → Serializer and deserializer at clock-domain boundaries.

Each application behaves like an FPGA peripheral operating under a defined clock rate. Deterministic messaging ensures system-wide coherence similar to phase-aligned derived clocks generated by the Clocking Wizard.

9 Relation to Clocking Wizard and PetaLinux Runtime

9.1 Software Timing as a Clocking Fabric

In the NOS3 + cFS model, the Software Bus (SB) and Scheduler form a virtual clock tree.

- The **TIME app** acts as the master reference clock, analogous to the Clocking Wizard's input PLL or base oscillator.
- The **SCH app** generates derived subrates for each application, equivalent to divided or phase-shifted clock outputs.
- Each **application** (ADCS, EPS, HK, etc.) represents a synchronous logic domain that publishes/consumes SB messages at its clock frequency.
- The **TO app** performs serialization of these timed data streams, while **CI** deserializes uplink traffic back into the SB fabric.

This system transforms timing signals into scheduled message flows—software-level equivalents of electrical clock nets in FPGA logic.

9.2 Integration with PetaLinux Runtime on Zynq Platforms

When running cFS on PetaLinux (e.g., Zynq-7020, Kria KV260), hardware and software timing become unified:

- The **Linux kernel clock source** provides the base tick aligned with the Processing System (PS) clock domain—comparable to the Clocking Wizard's master clock.
- Each cFS **application** executes as a POSIX thread under PetaLinux, with priority scheduling defined via `pthread_setschedparam()`.
- **SB message queues** correspond to AXI interconnects; latency and determinism can be tuned using CPU affinities and kernel real-time extensions.
- **CI/TO apps** connect to hardware interfaces such as Ethernet (UDP), UART, or SPI defined in the PetaLinux device tree, linking SB messaging directly to hardware PHY layers.
- Subsystems such as **BSD/CFDP** and **ADCS/EPS/HK** can be pinned to separate CPU cores, analogous to distributing clock outputs to independent FPGA logic domains.

Together, these layers produce a synchronized two-level timing system:

1. **Hardware/OS Level:** PetaLinux clock sources, interrupt timing, and thread scheduling.
2. **Application Level:** TIME and SCH define message cadence and inter-app synchronization.

This alignment ensures that even under heavy data loads, telemetry and command flows remain cadence-locked across the entire software-hardware stack.

10 References

References

[1] NASA Goddard Space Flight Center, *Core Flight System (cFS) Training and Architecture Overview*, NASA Technical Report, 2020. https://ntrs.nasa.gov/api/citations/20205011588/downloads/TM%2020205000691%20REV%201.pdf

[2] NASA IV&V, *NOS3: NASA Operational Simulator for Small Satellites – cFS Scenario Documentation*. https://nos3.readthedocs.io/en/latest/Scenario_cFS.html

[3] NASA Core Flight System (cFS) GitHub Repository. https://github.com/nasa/cFS

[4] Xilinx Inc., *Vivado Design Suite User Guide: Clocking Resources (UG472)*. https://docs.xilinx.com/r/en-US/ug472-clocking-resources

[5] Xilinx Inc., *PetaLinux Tools Documentation: Workflow and Runtime Environment*. https://docs.xilinx.com/r/en-US/ug1144-petalinux-tools-documentation

[6] Consultative Committee for Space Data Systems (CCSDS), *Space Packet Protocol*, CCSDS 133.0-B-2, Blue Book, September 2012. https://ccsds.org/Pubs/133x0b2e2.pdf

[7] Consultative Committee for Space Data Systems (CCSDS), *TM Space Data Link Protocol*, CCSDS 132.0-B-3, Blue Book, September 2015. https://ccsds.org/Pubs/132x0b3.pdf

App	Type	Typical Period	Primary Function
CI (Command Ingest)	Core	Event-driven	Receives CCSDS commands from ground, validates, republishes on SB.
TO (Telemetry Output)	Core	Event-driven	Subscribes to telemetry MsgIDs/APIDs; forwards to ground or RF chain.
HK (Housekeeping)	Core	1 Hz	Aggregates voltages, currents, temps, CPU metrics.
TIME	Core	1 Hz tick + ISRs	Maintains spacecraft time; syncs to GPS/ground.
SCH (Scheduler)	Core	1 Hz base tick	Dispatches periodic events; orchestrates app timing.
TBL (Table Services)	Core	N/A	Loads and verifies configuration tables.
EVS (Event Services)	Core	Event-driven	Manages system logging and event distribution.
HS (Health & Safety)	Core	1–2 Hz	Monitors app heartbeats and triggers safing.
DS (Data Storage)	Core	1–10 Hz	Routes selected telemetry streams to files.
FM (File Manager)	Core	Event-driven	Handles file I/O, CRC, and data movement.
CF (CFDP Agent)	Service	Background	Reliable file transfer for bulk science data.
ADCS	Mission	10–50 Hz	Fuses IMU/GPS data; controls attitude.
BSD (Bulk Science Data)	Mission	1–5 Hz + bursts	Segments science data into CCSDS packets.
Mode Manager	Mission	1 Hz	Controls system modes: SAFE, NOMINAL, DOWNLINK.
EPS	Mission	0.5–1 Hz	Power-state management, load shedding.
Thermal	Mission	0.5–1 Hz	Heater control, thermal protection.
Radio/Link	Mission	1 Hz + events	Manages RF link parameters, UDP bridges.
Deployment	Mission	Event-driven	Controls one-shot actuators and inhibits.

Table 1: Core and mission-specific application set with example cadences.