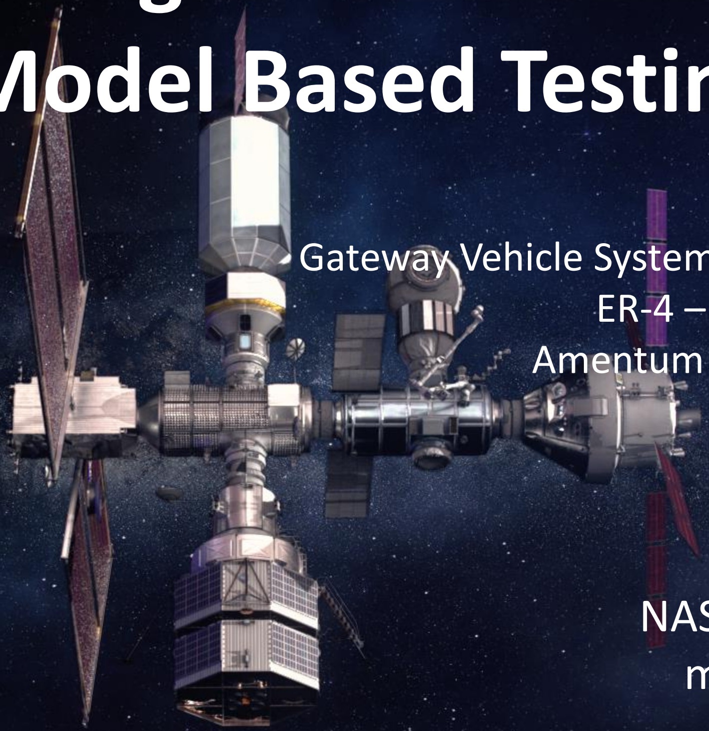
Simulink-Flight Software Bridge for Model Based Testing



Sherif Matta
Gateway Vehicle System Manager L2 VnV Tools Lead
ER-4 – NASA Johnson Space Center
Amentum – Autonomous Systems RDC
sherif.m.matta@nasa.gov

Marc Carbone
Electrical Engineer
NASA Glenn Research Center
marc.a.carbone@nasa.gov

IEEE Aerospace Conference 2026

March 11, 2026

Overview

Topics:

- Introduction – Challenges in Flight Software Verification & Validation
- Simulink–FSW Bridge Blockset Overview
- Automatic Blockset Generation from XTCE / Data Dictionary
- Electrical Power System (EPS) Use Case – NASA Gateway
- Summary & Conclusions

Introduction

Challenges in Flight Software V&V

- Complex requirements and distributed development across multiple interacting subsystems
- Limited access to integrated hardware environments adds significant V&V challenges
- Traditional HIL setups are costly, time-consuming, and difficult to scale for comprehensive testing
- Directly integrating flight software with model-based frameworks is challenging — especially with legacy C-code or CCSDS Space Packet Protocol systems

Our Solution: Simulink–Flight Software Bridge

- Direct real-time interaction between Simulink and flight software binaries via a custom Bridge
- Supports read/write access to telemetry and command data using Space Packet Protocol (SPP)
- Bridge Interface auto-generated from standardized XTCE or XML data dictionary files
- Currently applied to testing the Electrical Power System (EPS) of NASA's Gateway project

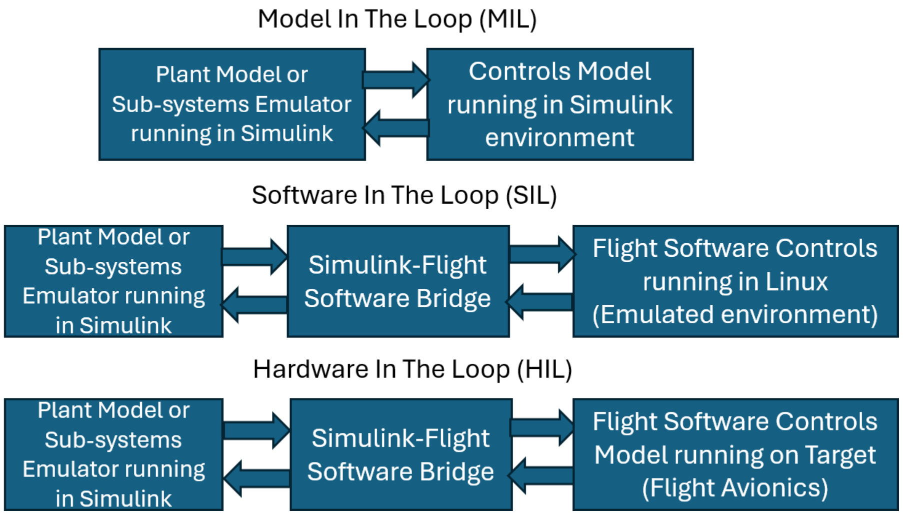
Testing Methodology: MIL → SIL → HIL

- Model-in-the-Loop (MIL): Both control logic and subsystem models run entirely within Simulink. Enables early functional validation and rapid algorithm iteration.

- Software-in-the-Loop (SIL): Actual compiled FSW runs in Linux emulated environment while plant models remain in Simulink. Communication via Simulink–FSW Bridge over SPP.

- Hardware-in-the-Loop (HIL): Simulink subsystem models interact with FSW on actual flight hardware or avionics targets. Validates HW–SW interfaces and real-time execution.

- Progressing MIL → SIL → HIL identifies integration issues early and ensures robust, validated control before flight deployment.



Model In The Loop (MIL)

Plant Model or Sub-systems Emulator running in Simulink

Controls Model running in Simulink environment

Software In The Loop (SIL)

Plant Model or Sub-systems Emulator running in Simulink

Simulink-Flight Software Bridge

Flight Software Controls running in Linux (Emulated environment)

Hardware In The Loop (HIL)

Plant Model or Sub-systems Emulator running in Simulink

Simulink-Flight Software Bridge

Flight Software Controls Model running on Target (Flight Avionics)

Figure 1: Simulink-FSW Testing Methods Block Diagram

Simulink Bridge Blockset Overview

Custom Simulink Bridge Blockset – 5 Primary Components:

A. Telemetry/Command Sender Block

Transmits TLM & CMD packets from Simulink to FSW. Emulates MCC or external subsystems (HALO, PPE). Auto-configurable via XTCE.

B. Telemetry/Command Receiver Block

Captures incoming TLM/CMD streams from FSW in real-time. Configurable by packet type. Outputs Trigger_out, Raw_msg, Msg_Header for downstream chaining.

C. Telemetry/Command Parser Block

Parses wrapped packets and handles multiple messages on the same port. Reduces port open/close overhead.

D. Port Test Block

Tests point-to-point connectivity. Sender transmits known patterns (constant, counter, sine). Receiver verifies correctness and SBNG network layer integrity.

E. Event & Sequence Viewer

Timeline view of all received FSW events. Critical for sequence analysis and verifying expected system response.

Sender & Receiver Blocks

A. Telemetry/Command Sender Block

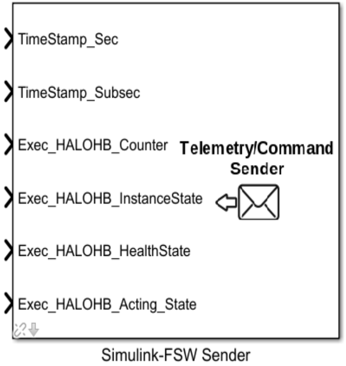
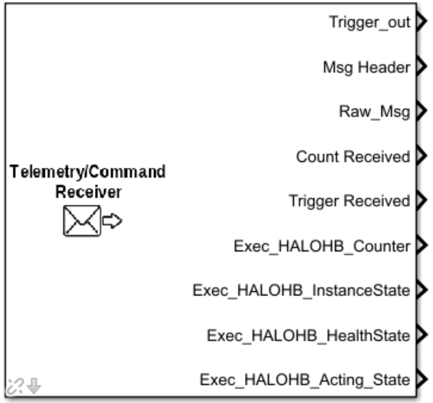


Figure 2: Simulink-FSW Sender Block

- MCC Emulation: Sends command packets to the FSW under test
- Subsystem Emulation: Simulates TLM from HALO, PPE, and other external subsystems
- Auto-configurable via XTCE metadata — integrates into any Simulink test harness
- Timestamp inputs: TimeStamp_Sec and TimeStamp_Subsec
- Additional command arguments appear as extra block inputs

Sender & Receiver Blocks

B. Telemetry/Command Receiver Block



Simulink-FSW Receiver

- Captures TLM/CMD streams from FSW or external sources in real-time
- Configurable for specific packet types or response codes
- Trigger_out → triggers downstream blocks on receive events
- Raw_msg → passes payload to Parser blocks for further processing
- Msg_Header, Count Received, Trigger Received → execution control

Parser Block, Port Test & Event Viewer

C. Telemetry/Command Parser Block

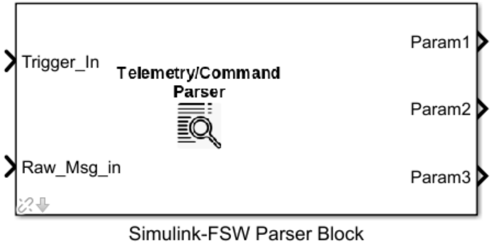
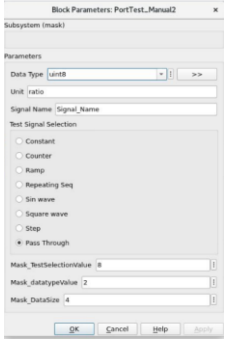


Figure 4: Simulink-FSW Parser Block

- Parses wrapped packet-within-packet structures (Trigger_In + Raw_Msg_in inputs)
- Handles multiple TLM/CMD messages on the same network port
- Reduces overhead by eliminating repeated port open/close cycles
- Extra parsed parameters appear as additional outputs

D. Port Test Block



Block Parameters: PortTest_Manual2

Subsystem (mask)

Parameters

Data Type: uint8

Unit: ratio

Signal Name: Signal_Name

Test Signal Selection

○ Constant
○ Counter
○ Ramp
○ Repeating Seq
○ Sin wave
○ Square wave
○ Step
● Pass Through

Mask_TestSelectionValue: 8

Mask_datatypeValue: 2

Mask_DataSize: 4

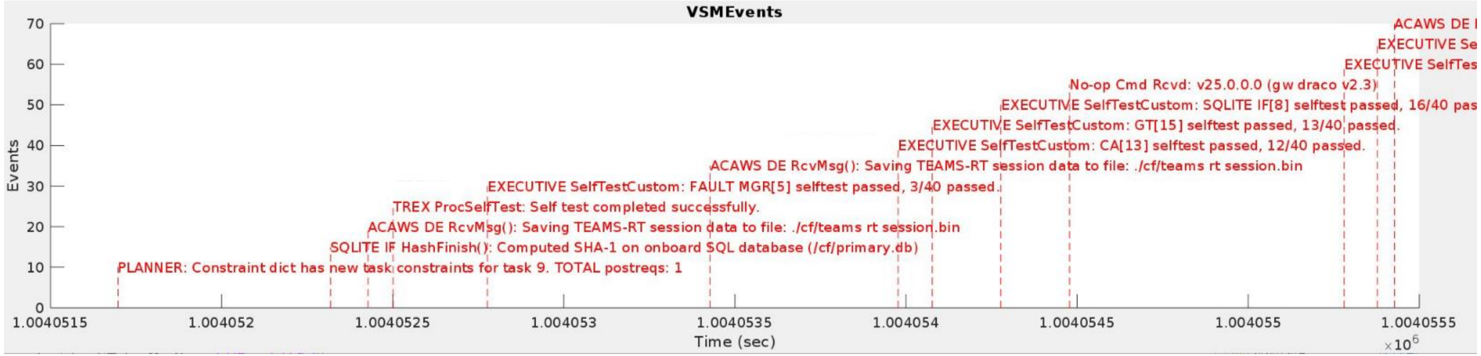
Figure 5: Port Test Block Layout

- Validates point-to-point connectivity and SBNG network layers
- Test signals: Constant, Counter, Sine wave, Ramp, Square wave
- Embedded in interface generation — transparent to test cases

Parser Block, Port Test & Event Viewer

E. Event & Sequence Viewer

- Timeline view of all received events from the FSW
- Enables sequence analysis — verify expected system response
- Critical for event-driven and fault-handling test scenarios



```
ACAWS DE I
EXECUTIVE Se
EXECUTIVE SelfTes
No-op Cmd Rcvd: v25.0.0.0 (gw draco v2.3)
EXECUTIVE SelfTestCustom: SQLITE IF[8] selftest passed, 16/40 pas
EXECUTIVE SelfTestCustom: GT[15] selftest passed, 13/40 passed.
EXECUTIVE SelfTestCustom: CA[13] selftest passed, 12/40 passed.
ACAWS DE RcvMsg(): Saving TEAMS-RT session data to file: ./cf/teams rt session.bin
EXECUTIVE SelfTestCustom: FAULT MGR[5] selftest passed, 3/40 passed.
TREX ProcSelfTest: Self test completed successfully.
ACAWS DE RcvMsg(): Saving TEAMS-RT session data to file: ./cf/teams rt session.bin
SQLITE IF HashFinish(): Computed SHA-1 on onboard SQL database (/cf/primary.db)
PLANNER: Constraint dict has new task constraints for task 9. TOTAL postreqs: 1
```

Figure 6: Simulink-FSW Event Viewer

Automatic Blockset Generation from XTCE

The process of manually adding the Simulink–FSW Bridge is time-consuming. A dedicated tool was developed to extract the TLM/CMD structure from XTCE or Data Dictionary files and automatically generate and configure the bridge blockset.

Auto-Generation Process

- Parse XTCE or Data Dictionary file to extract TLM & CMD definitions
- Define producer-consumer relationships with controller network configuration
- Auto-generate Sender and Receiver blocks simultaneously
- Each generated interface block includes a Port Test block for end-to-end connectivity verification
- Enhance signal names/descriptions using ICD and FSW Header files

Key Benefits

- Reduces setup time — eliminates manual configuration bottleneck
- Minimizes human error in interface configuration
- Ensures integration-free interfaces between supplier FSW components
- Accelerates test case development
- Auto-coded executables run faster than Simulink model simulation
- Compatible with NASA's broader testing toolchain

Workflow:

XTCE/Data Dict → Parse TLM & CMD → Configure Network → Auto-generate Sender + Receiver → Run Port Test → Switch to Functional Test Mode

Electrical Power System Use Case – NASA Gateway

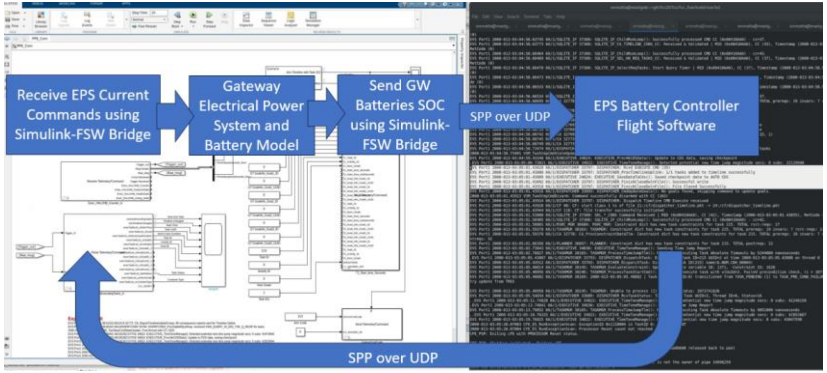
Application Context

- Gateway is NASA's planned lunar orbit space station — a key component of the Artemis program
- Power manager controls distributed batteries during insolation, eclipse, and fault/disturbance events
- Without access to physical Gateway DC power hardware, implementation and testing presents a unique challenge
- Simulink-based EPS model developed using PC Krause & Associates DC space power library

Testing Approach

- Proportional Discharge Mode Control: Weighted integral control for load sharing across distributed batteries during insolation and eclipse phases
- Pre-scripted load/solar generation schedules emulate the orbital phase of the mission
- Faster-than-real-time EPS dynamics simulation with telemetry/commands exchanged via FSW Bridge each controller update period
- EPS model supports varying loads, inserting faults, and injecting disturbances for maximum test coverage

EPS Testing Framework & Results



Receive EPS Current Commands using Simulink-FSW Bridge

Gateway Electrical Power System and Battery Model

Send GW Batteries SOC using Simulink-FSW Bridge

SPP over UDP

EPS Battery Controller Flight Software

SPP over UDP

Figure 7: EPS Testing Framework — Simulink-FSW Bridge with Gateway EPS on flight hardware

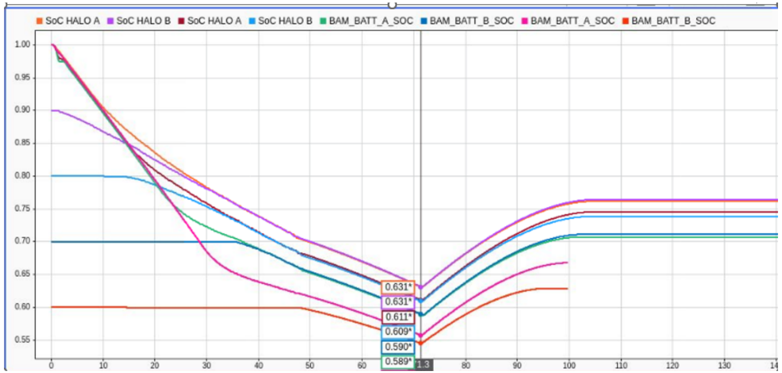


Figure 8: MBT EPS Monte Carlo results — different battery initial SOC conditions

- Closed-loop SIL demonstrated: FSW controller receives EPS telemetry, computes power electronic set points, returned to Simulink EPS model each update period
- Monte Carlo simulations: Multiple battery initial State of Charge (SOC) scenarios validated — controller performance across edge cases and disturbances
- Compares MIL, SIL, and HIL results — validates consistency across all testing stages

Future Work

- Extend the framework to additional NASA Gateway subsystems (Propulsion, ECLSS, Thermal Control)
- Enhance XTCE parser to leverage Interface Control Document (ICD) for richer signal naming and descriptions
- Deeper integration with NASA's formal Verification & Validation toolchain and regression test automation
- Expand Monte Carlo test automation for broader edge case and fault coverage
- Apply framework to other NASA programs and non-Gateway FSW using CCSDS-based architectures
- Investigate auto-code generation paths for even faster execution of auto-generated bridge models

Conclusion

- Presented a model-based testing framework integrating flight software with Simulink via a custom Simulink Bridge Blockset
- Blockset supports real-time bidirectional communication using CCSDS Space Packet Protocol — within the Simulink simulation environment
- Configurable Sender, Receiver, and Parser blocks emulate both MCC commands and subsystem telemetry
- Bridge is automatically configurable through standard XTCE interface definitions — reduces setup time and errors
- Supports MIL, SIL, and HIL testing methodologies — full V-model coverage without hardware dependency in early phases
- Successfully applied to NASA Gateway EPS battery controller testing — demonstrated scalability and effectiveness for mission-critical applications

Questions & Discussion

Sherif Matta
NASA Johnson Space Center
sherif.m.matta@nasa.gov

Marc Carbone
NASA Glenn Research Center
marc.a.carbone@nasa.gov

ACKNOWLEDGEMENTS

This work was supported by the NASA Gateway project under the ER4 division.

Grateful acknowledgement to:

- Jonathan Rogers
- Dr. Julia M. Badger
- Laura Barron
- Dr. Michael Whitzer
- Pavan Rajagopal

Simulink-FSW Bridge
for Model-Based Testing

IEEE Aerospace Conference 2026