Software Development Infrastructure for Small Spacecraft

Flight Software Workshop 2007

Howard Cannon, Craig Pires

11/5/2007

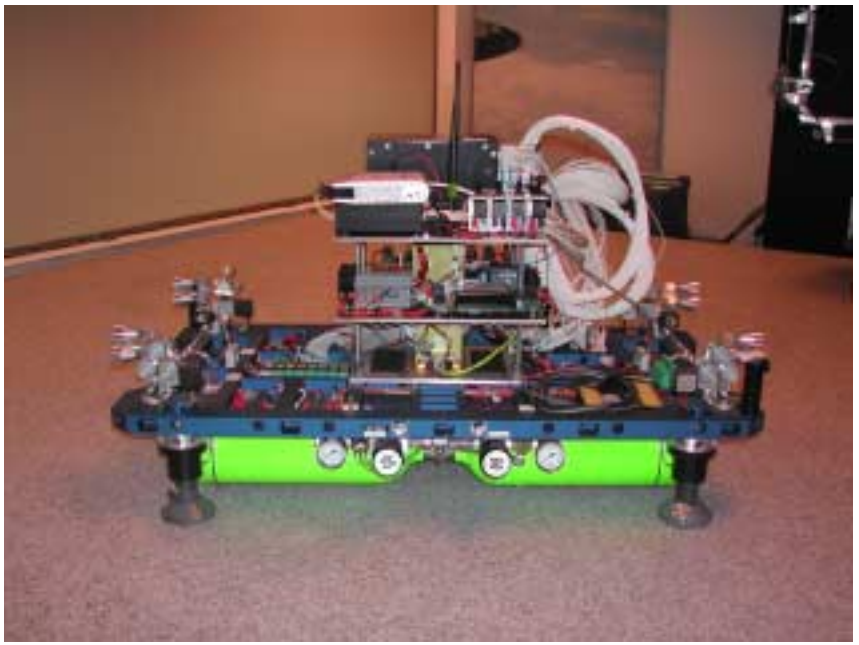
Overview

GOAL:

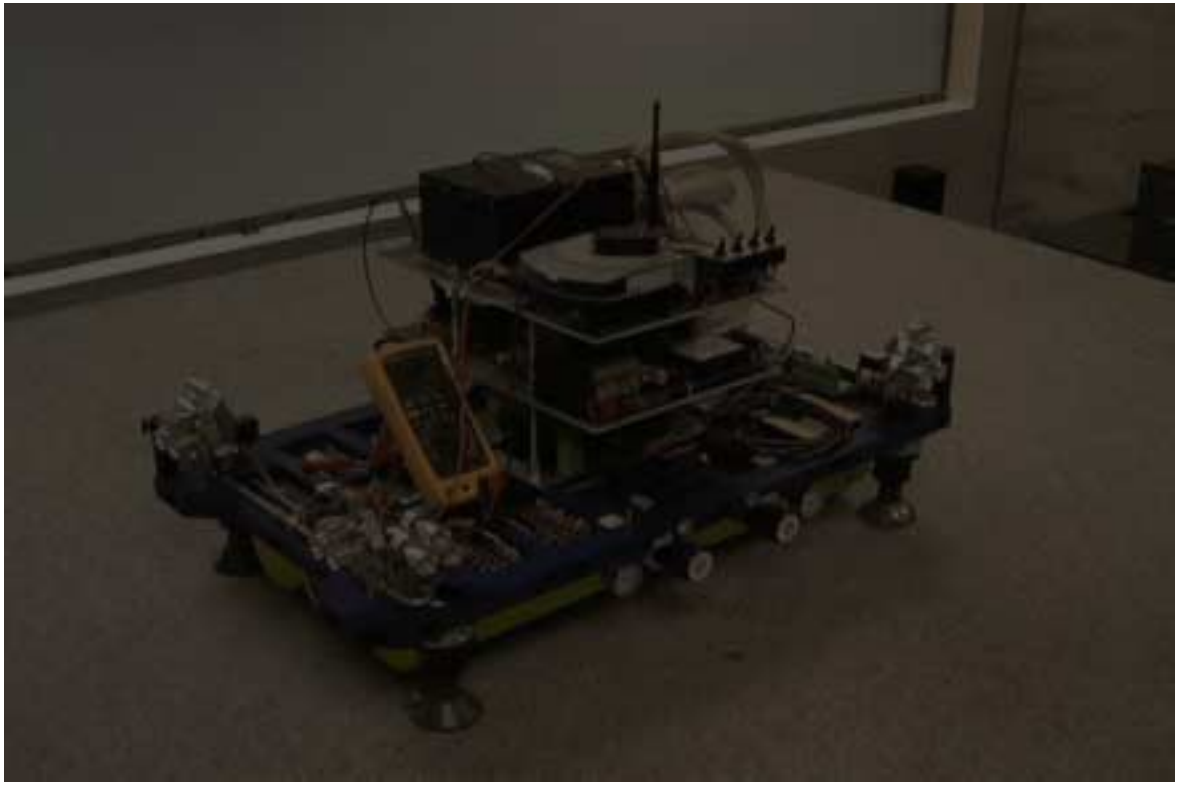
Develop infrastructure and processes for small spacecraft software development using automatic code generation techniques.

2007 Objectives:

- Flatsat – Small cart on flat granite table. Develop approach/processes and conduct tool trades using simple example.
- 6 DOF Integration Test – Demonstrate 6DOF control on small spacecraft bus test platform with cold gas thrusters.
- Cruise Phase Simulation – Demonstrate “lost-in-space” algorithm and trajectory correction maneuvers in simulation.

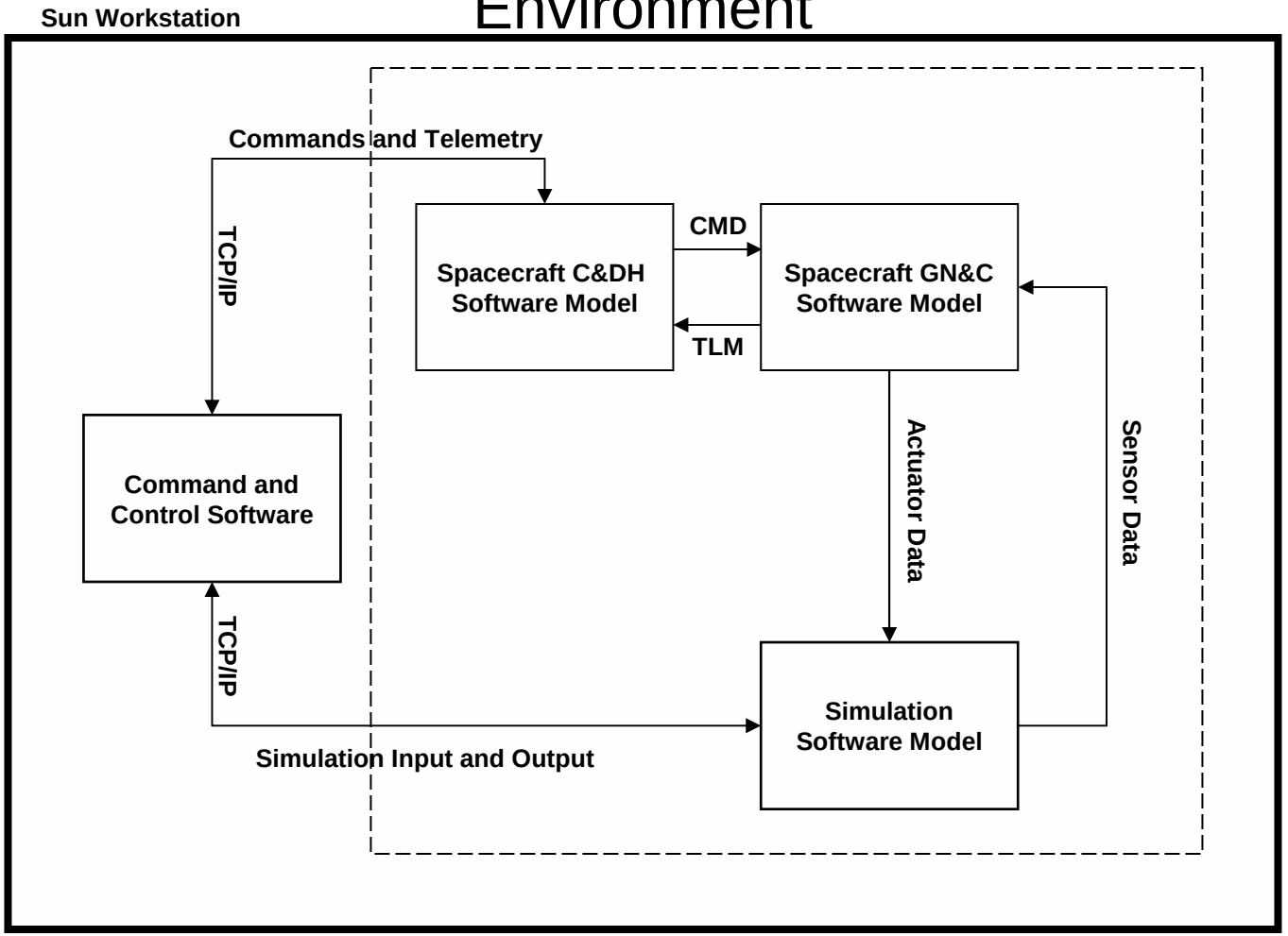


FLATSAT TESTBED

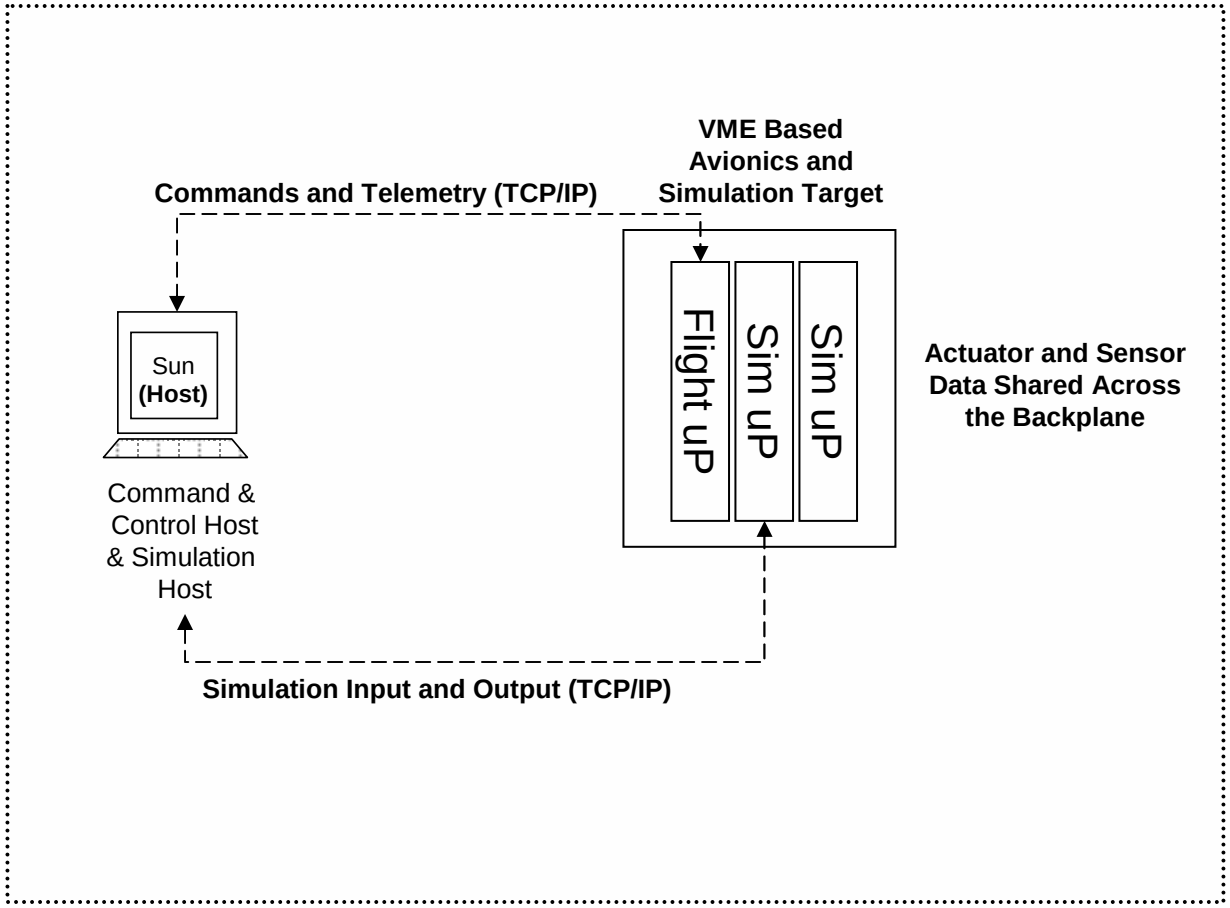


Common Spacecraft Bus Pressure Test

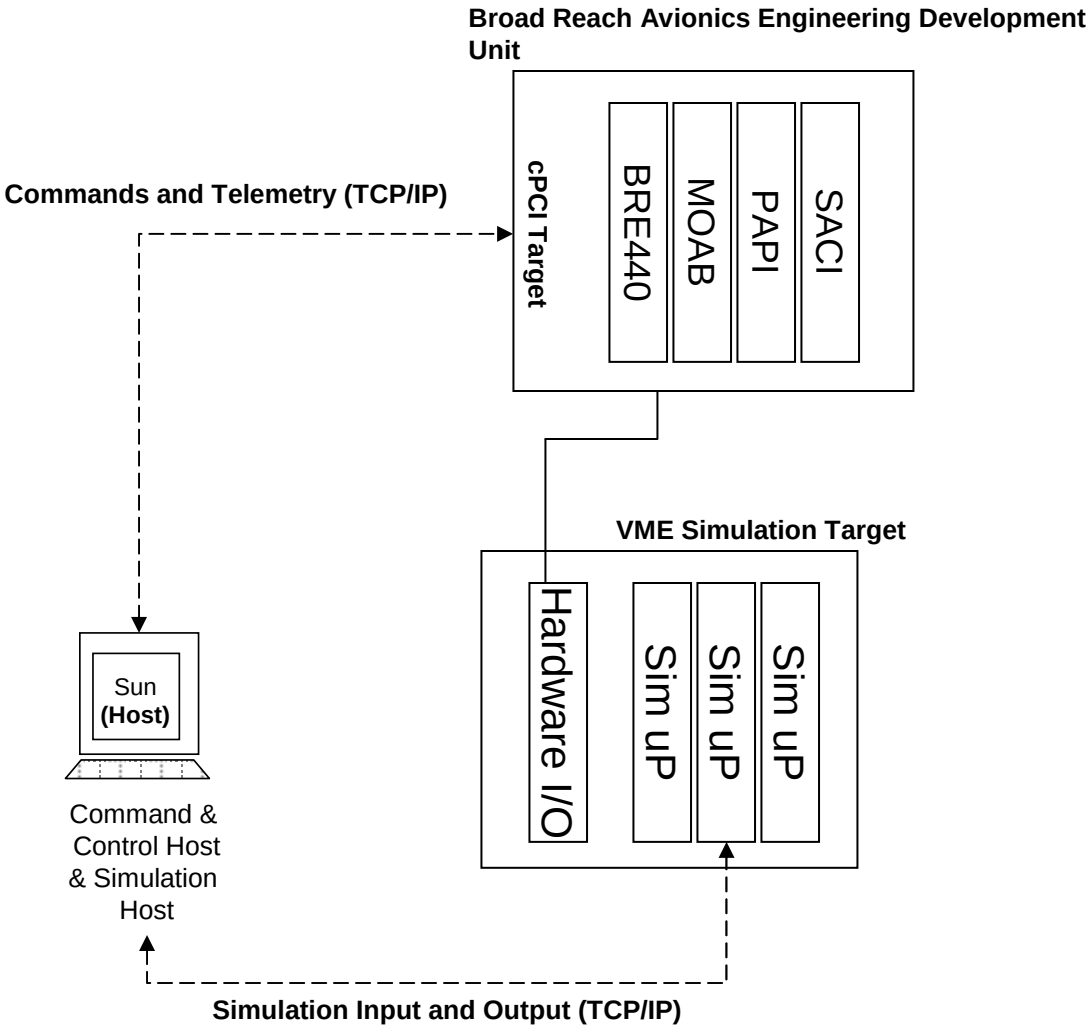
FSW Development: Workstation Simulation (WSIM) Environment



FSW Development: Processor-in-the-Loop

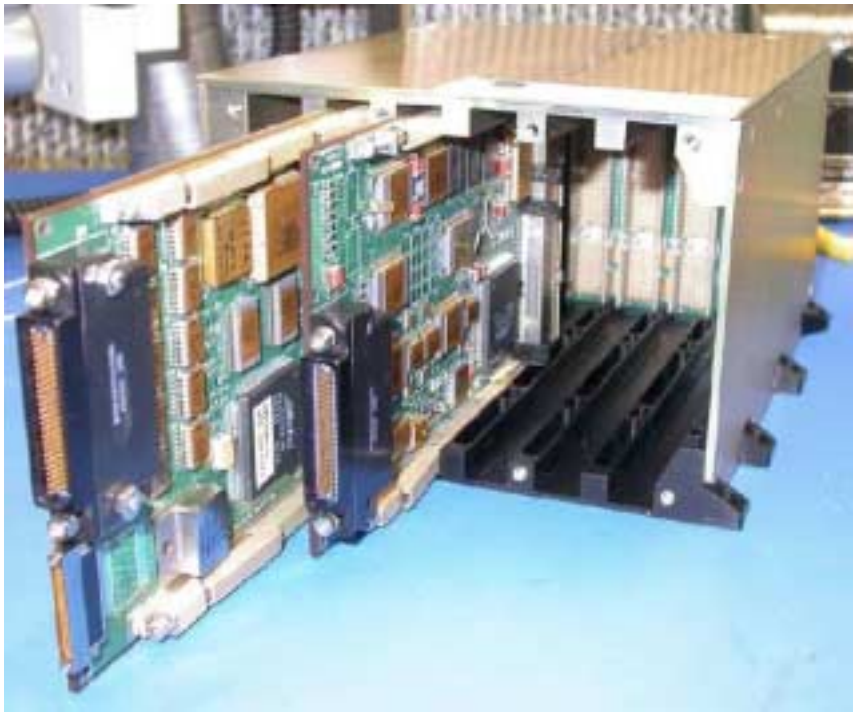


FSW Development: Hardware-in-the-Loop

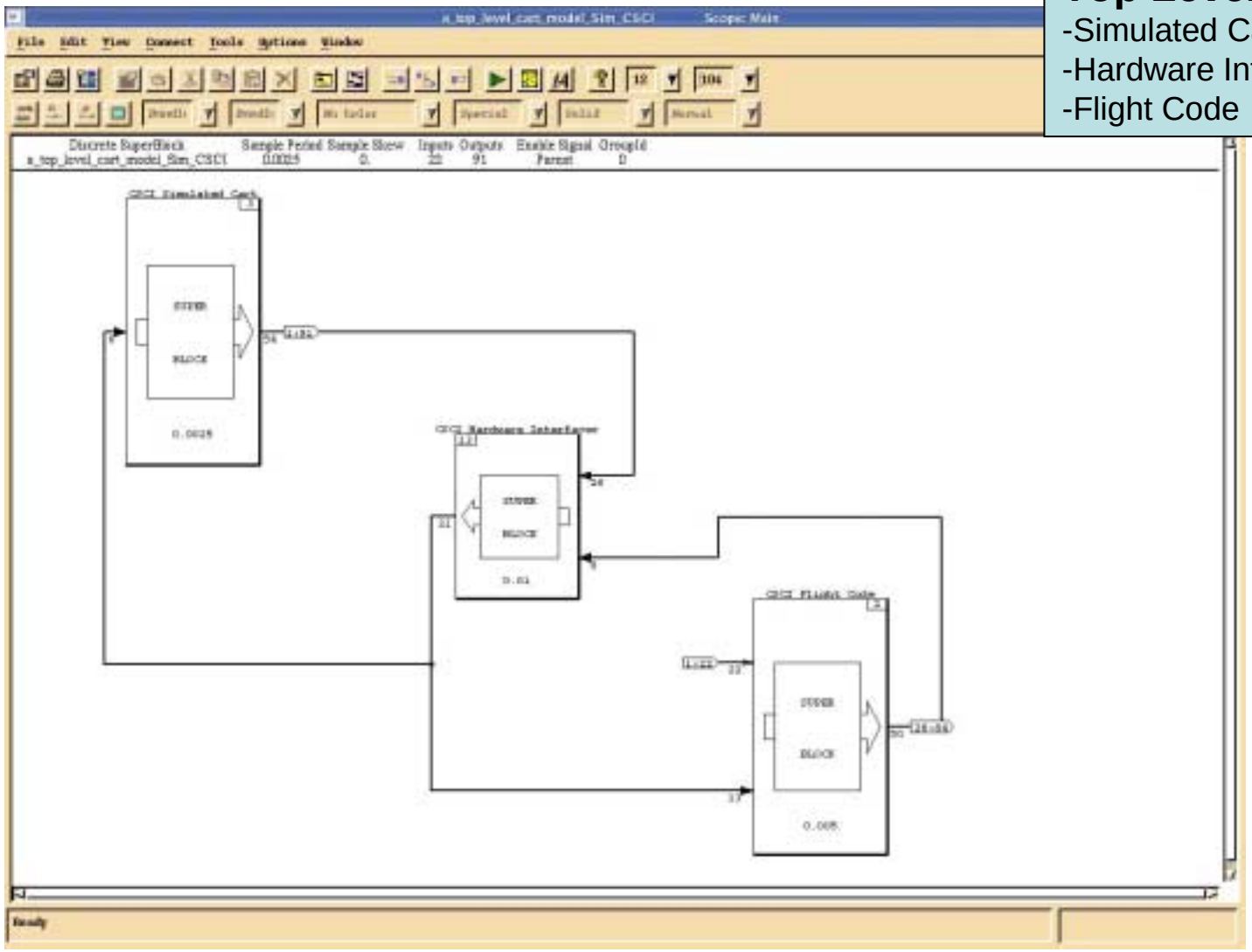


Broad Reach Engineering Development Unit

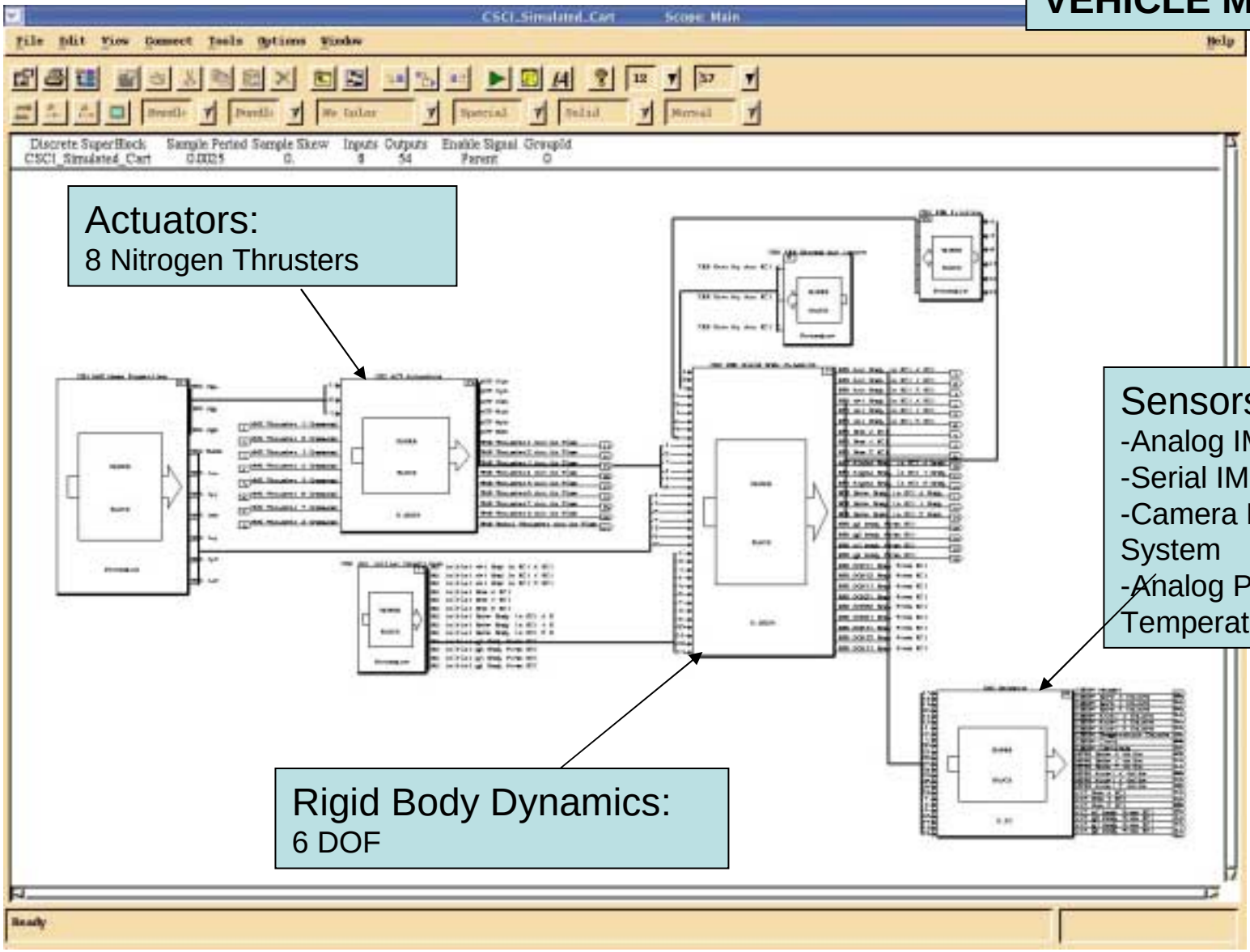
- 8 Slot (5 C&DH, 3 Power) 3U cPCI Chassis
- BRE Starter BRE440
  - 128 Mbyte SDRAM, 8 Mbytes Boot-ROM
  - 200 MHz/400 Mips
- MOAB IO Board
  - 47 AD590 Temp channels
  - 12 Sun sensor channels
  - 24 Analog channels
  - 40 RS422 /LVDS transmitters and receivers
  - 48 Discrete Inputs and Outputs
  - MIL-STD-1553
- Solar Array Control Integration (SACI) board.
- Power Switching and Pyro Integration (PAPI) board.



Top Level Model:
-Simulated Cart
-Hardware Interfaces
-Flight Code

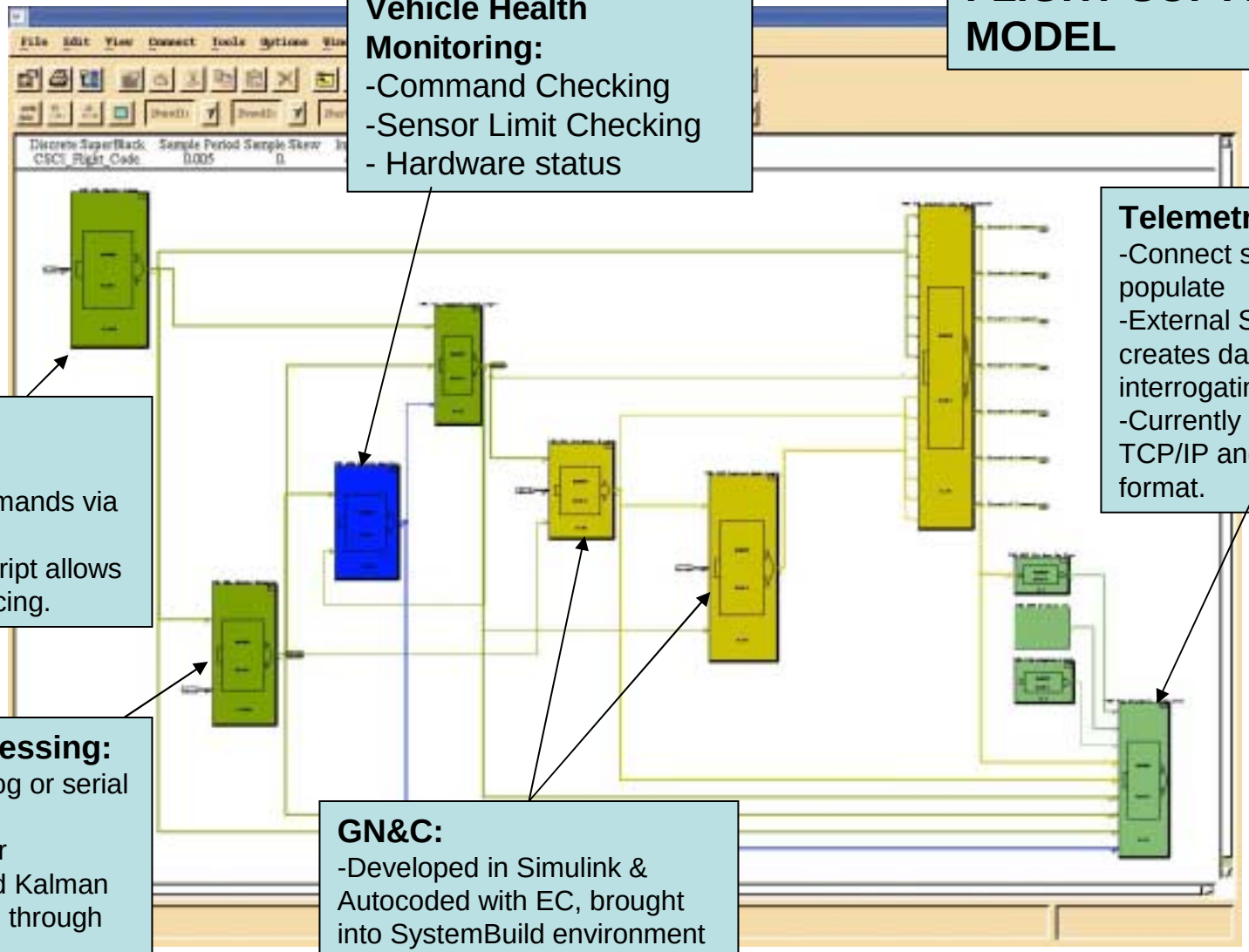


VEHICLE MODEL



FLIGHT SOFTWARE MODEL

Vehicle Health Monitoring:
-Command Checking
-Sensor Limit Checking
- Hardware status



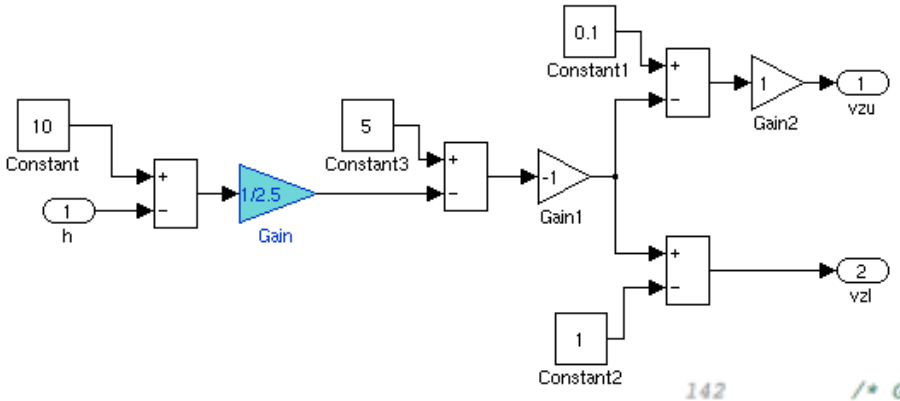
Telemetry:
-Connect signals to populate
-External Script file creates database by interrogating model
-Currently implemented TCP/IP and 422 in TCM format.

Command Processing:
-Receives commands via TCP/IP or 422.
-Compiled in script allows flexible sequencing.

Sensor Processing:
-Receives analog or serial data.
-Low Pass Filter
-Auto generated Kalman Filter integrated through UCB.

GN&C:
-Developed in Simulink & Autocoded with EC, brought into SystemBuild environment using UCB.
-Multiple approaches investigated: Bang-Bang, PWPF

Automatic Code Generation

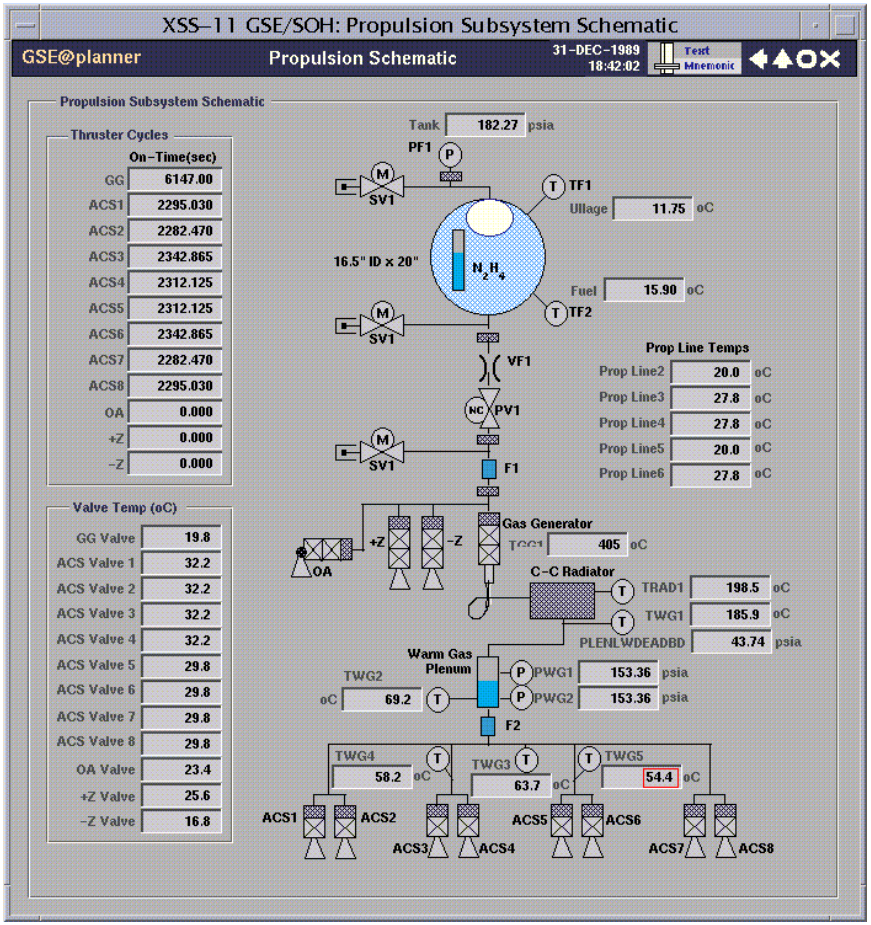



```
142         /* Gain: '<S16>/Gain1' incorporates:
143          *  Constant: '<S16>/Constant'
144          *  Constant: '<S16>/Constant3'
145          *  Gain: '<S16>/Gain'
146          *  Sum: '<S16>/Sum'
147          *  Sum: '<S16>/Sum3'
148          */
149         rtb_Gain1_o = (5.0 - (10.0 - rtb_h_cg) * 0.4) * -1.0;
150         rtb_Switch_g_idx = 0.1 - rtb_Gain1_o;
151         rtb_Switch_g_idx_0 = rtb_Gain1_o - 1.0;
```

- Simulink supports two way trace-ability between models and generated code
- Code Easy to read, well commented

Mission Ops Software

- Integrated with commercially available (Octant Technologies) mission operations software
  - CmdBuilder – GUI for spacecraft telecommands and scripting.
  - TelemScope – telemetry monitoring, archiving, and trend analysis.
- Investigations ongoing into ASIST and SCL



V&V

- Utilizing html based documents for tracking requirements, procedures, specifications, and verification results.
- Unit test scripts exercise low level blocks within the model in the WSIM.
- Perl scripts for automatic execution of integrated tests in WSIM, PIL, and HWIL. Test results and links to data automatically populated in html docs.

Lessons Learned

- **Spiral Development Approach** – Prototype, code, test, and debug early and often.
- **Elimination of Errors** – Eliminated need to manually translate GN&C algorithms to flight code.
- **Reduced Training** – No need to teach control systems experts how to write “good code”.
- **Compatibility** – Using Simulink, SystemBuild and hand written code within the same development framework allows compatibility with various vendors, tools, and legacy code.
- **Enhanced Debugging** – Model based tools provide graphical debugging facilities in addition to standard embedded systems debugging tools.
- **Reuse** – The model based technique lends itself to reusing components. Successfully reused a majority of the software components for the 6 DOF integration tests that were originally built for the granite table tests.
- **Wide Applicability** – Approximately 85% of the software we have developed is automatically generated. Low level software more suited to hand coding. Didn’t try to force it.

Summary

- NASA Ames has been implementing an infrastructure for small spacecraft software development based on Automatic Code Generation techniques.
- Demonstrating this approach on two testbeds.
- Automatic Code Generation seems both feasible and desirable.
- Continuing to refine approach and look at tools/process trades.