Modular Infrastructure for Rapid Flight Software Development

Craig Pires
NASA Ames Research Center
Overview

• Background
• Flight Software Development Process
• Simulink Model Overview
• Integration with cFE
Background

• Small Spacecraft Investigation
  – Modular Common Bus Spacecraft

• Hover Test Vehicle (HTV) Development

• Current - Lunar Atmosphere and Dust Environment Experiment (LADEE)
  – Joint ARC/GSFC Mission
  – Lunar Orbiter, Launch 2012
• Model Based Development Approach
  – Develop Models of FSW, Vehicle, and Environment in Simulink
  – Automatically generate Software using RTW/EC.
  – Integrate with hand-written and heritage software.
  – Iterate while increasing fidelity of tests – Workstation Sim (WSIM), Processor-In-The-Loop (PIL), Hardware-in-the-Loop (HIL)
Simulink HTV Architecture

FSW Integrated with CFE

Vehicle & Environmental Simulation

ITOS (GDS)
Flight Software Model

Command Processing:
- Receives commands via CDH (TCP/IP or RS422).
- Compiled in script allows flexible sequencing.
- Processes and Sets Control Modes.

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

State Estimation:
- Receives sensor data.
- Low Pass Filters
- Auto generated Kalman Filter.

Telemetry:
Passes data to the CDH so that it can be transmitted via TCP/IP or RS422.

Prop Management:
- Fires thrusters based on commands and control mode.

Simulink Bus Creator

GN&C:
- Guidance System sets desired angles based on position error.
- Guidance System maintains desired vertical velocity.
- Control System uses Bang-Bang approach to maintain desired angle.
Flight Hardware Model

Sensor Models
- Analogs (Temperature, Pressure)
- LN200 IMU
- VIZ Camera System

Thruster dynamic forces and torques.

Mass and Inertia Characteristics of Vehicle
Environment Link Model

Command and Downlink Delays

6DOF Position and Rotational Propagation

External Forces on Vehicle (Tether, platform)

Gravitational Forces

Vehicle Initial Conditions
cFE Simulink Integration
cFE – Core Flight Executive

• Goddard Space Flight Center Developed
• Derived from Legacy Missions
• Flexible infrastructure for Space Flight Software

• Components:
  – Executive Services
  – Event Services
  – Time Services
  – Table Services
  – Software Bus Services
Simulink Bus becomes cFE Message

```plaintext
Simulink Bus becomes cFE Message

'Simulink Bus becomes cFE Message'

'sprintf(''), { ...
  {'Ins_msg', 3, 'int16', -1, 'real', 'Sample'}; ...
  {'Ins_delta_velocity_counts', 3, 'int16', -1, 'real', 'Sample'}; ...
  {'Ins_delta_angle_counts', 3, 'int16', -1, 'real', 'Sample'}; ...
  {'Ins_status', 1, 'int16', -1, 'real', 'Sample'}; ...
  {'Ins_mode', 1, 'int16', -1, 'real', 'Sample'}; ...
  {'Ins_data', 1, 'int16', -1, 'real', 'Sample'}; ...
  {'Ins_counts', 3, 'int16', -1, 'real', 'Sample'}; ...
  {'Ins_checksum', 1, 'int16', -1, 'real', 'Sample'}; ...
} ...
```
cFE Interface App Loop

Struct App_Inputs In
Struct App_Outputs Out
App_Init() {
    Initialize_App_Inputs()
    Subscribe_SB_Msgs(Tick, AppMsgs, ...)
    Simulink_Init(In, Out)
}
App_Main() {
    App_Init()
    while(1) {
        sb_receive_msg(msg, timeout)
        if (msg == tick) {
            Simulink_Step(dt, In, Out)
            sb_send_msg(Out) /* app update */
        } else {
            If (msg == app_update) /* Process other App Msgs */
                App_Update_Inputs(msg, Out)
            else Process_Msg(msg) /* HK, Cmds, etc… */
        }
    }
}
New Efforts

• 3DOF Simulator
• Command & Telemetry Dictionary – XTCE
• Performance / Latency Reduction
• cFE Interface Enhancements
Summary

• NASA Ames developing infrastructure for rapid flight software development
• Model based process leverages Mathworks Simulink, RTW-EC
• Developed modular approach to integrate auto-generated code with GSFC’s cFE.
• Successfully demonstrated on HTV
• Being Utilized on NASA’s LADEE mission
Backup
IMU_Main(){
    while(1) {
        struct imu_input_str imu_in
        read_msg_que(imu_in, timeout) /* VxWorks Msg Que */
        sb_send_msg(imu_msg)
        Send_tick()
    }
}

Cnt = 0;
Send_tick() {
    sb_send_msg(400HZ_Tick)    /* Do we need 400HZ Tick or key off of IMU Data? */
    if ((Cnt % 2) == 0)     sb_send_msg(200HZ_Tick)
    if ((Cnt % 4) == 0)     sb_send_msg(100HZ_Tick)
    if ((Cnt % 40) == 0)   sb_send_msg(10HZ_Tick)
    if ((Cnt % 400) == 0) sb_send_msg(1HZ_Tick)
}

    Cnt++;
}

/* Note: Other Apps same as IMU without the Send_tick() */
• Simulink/SystemBuild Only (No Autocode)
• Early in development process
• Algorithm Development
• Requirements Analysis
• Models autocoded and running on RT processors
• Inexpensive “flight-like” processor
• Tests autocoding process & integration with C&DH software
• Integration with Telemetry Software allows early development/testing of downlink
• Can be used for initial code size and resource utilization analysis
Flight code runs on Flight Avionics EDU
• Provides testing of FSW with Avionics I/O
• Definitive answers on resource utilization
• Highest fidelity simulations for verification/validation
Automatic Code Generation

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