Master Class – Verification, Validation and Testing Techniques with Model-Based Design

MATLAB EXPO
April 21, 2016
Software Complexity Increases Rapidly

- **1970**: Microcode/Assembly
  - Motorola 6800
    - 8 bit / 1 MHz / 4 kBytes

- **1990**: Modeling
  - C

- **2000**: Research
  - Freescale MPC5674F
    - 32 bit / 200+ MHz / 4000 kBytes

- **2010**: Production

- **2020**: Abstraction
  - Processor

- **1980**

- **1990**

- **2000**

- **2010**

- **2020**
Finding Errors Late in Project is Costly

“each delay in the detection and correction of a design problem makes it an order of magnitude more expensive to fix…”

Clive Maxfield and Kuhoo Goyal
“EDA: Where Electronics Begins”
Verification & Validation of Models and C/C++ Code
Typical Simulation-Based Testing Scenarios

**Concept Phase**

- How can I interactively test and debug my system?
- How can I show what I have done to my manager?
- How can I perform reactive testing of my system?

**Production Phase**

- Does my system meet the design requirements?
- How complete is my testing?
- The algorithm worked last week… does it still?
- Another engineer modified the algorithm… still working ok?
- Does the generated code performance match the model?
Case Study: Fault-Tolerant Fuel-Rate Controller
Case Study: Fault-Tolerant Fuel-Rate Controller

- Component 1: Safety Logic
Case Study: Fault-Tolerant Fuel-Rate Controller

- Component 2: Fuel-Rate Calculation
DEMO: Conceptual Test with Dashboard Blocks
DEMO: Assessing Simulation Results with SDI*

*Simulation Data Inspector
DEMO: Reactive Test using Test Sequence Block

Test Harness: Fuel Rate Controller

- **Test_Overspeed**: throttle = 30; speed = 300+ramp(10°/et);
  - 1. speed > 700
  - Reset_to_Normal_Speed

- **Reset_to_Normal_Speed**: throttle = 10; speed = 700-ramp(10°/et);
  - 1. speed < 400
  - Waypoint

- **Waypoint**: 1. after(2, sec)
  - Test_EGO_Fault

- **Test_EGO_Fault**: ego = 12;
  - 1. after(3, sec)
  - EGO_Reset

- **EGO_Reset**: 1. after(10, sec)
  - Test_Throttle_Fault

- **Test_Throttle_Fault**: throttle = 0;
  - 1. after(3, sec)
  - Throttle_Reset

- **Throttle_Reset**: 1. after(3, sec)
  - Test_Speed_Fault

- **Test_Speed_Fault**: speed = 0;
  - 1. after(3, sec)
  - Speed_Reset

- **Speed_Reset**: 1. after(3, sec)
  - Test_MAP_Fault

- **Test_MAP_Fault**: map = 0;
  - 1. after(3, sec)
  - MAP_Reset

Sequence Sensor Failures

Assessments
## DEMO: Reactive Test using Test Sequence Block

### Test Harness: Fuel Rate Controller

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Overspeed</td>
<td>speed &gt; 700</td>
<td>Reset to Normal Speed</td>
</tr>
<tr>
<td>Reset to Normal Speed</td>
<td>speed &lt; 400</td>
<td>Waypoint</td>
</tr>
<tr>
<td>Waypoint</td>
<td>after(2, sec)</td>
<td>Test EGO Fault</td>
</tr>
<tr>
<td>EGO_RESET</td>
<td>after(3, sec)</td>
<td>EGO_Reset</td>
</tr>
<tr>
<td>Test Throttle Fault</td>
<td>throttle = 0</td>
<td>Test Throttle Fault</td>
</tr>
<tr>
<td>Throttle_RESET</td>
<td>after(3, sec)</td>
<td>Throttle_RESET</td>
</tr>
<tr>
<td>Test Speed Fault</td>
<td>speed = 0</td>
<td>Speed_RESET</td>
</tr>
<tr>
<td>Speed_RESET</td>
<td>after(3, sec)</td>
<td>Test_MAP_FAULT</td>
</tr>
<tr>
<td>Test_MAP_FAULT</td>
<td>after(3, sec)</td>
<td>MAP_RESET</td>
</tr>
</tbody>
</table>
Typical Simulation-Based Testing Scenarios

Concept Phase

- How can I interactively test and debug my system?
- How can I show what I have done to my manager?
- How can I perform reactive testing of my system?

Production Phase

- Does my system meet the design requirements?
- How complete is my testing?
- The algorithm worked last week… does it still?
- Another engineer modified the algorithm… still working ok?
- Does the generated code performance match the model?
Linking Requirements and Test Specifications to Simulink

Test Specification

1. All sensors functioning (LOW mode)
   Total simulation time = 3 sec
   Inputs:
   - Set throttle sensor to normal operating condition for time = 0 to 3 sec
   - Set engine speed to normal operating condition of 300 for time = 0 to 3 sec
   - Set EGO sensor to normal operating condition for time = 0 to 3 sec
   - Set MAP sensor to normal operating condition for time = 0 to 3 sec
   Expected output (measurement to be taken after time >= 2.5 sec):
   - All other output state shall be set to 0
   - Fuel_mode shall be set to LOW

2. Throttle Sensor short to ground, open circuit (RICH mode)
   Total simulation time = 3 sec
   Inputs:
   - Set throttle sensor to normal operating condition for time = 0 to 1 sec,
     then set throttle sensor error at time = 1.01 sec to 3 sec
   - Set speed sensor to normal operating condition for time = 0 to 3 sec
   - Set EGO sensor to normal operating condition for time = 0 to 3 sec
   - Set MAP sensor to normal operating condition for time = 0 to 3 sec
   Expected output (measurement to be taken after time >= 2.5 sec):
   - Fail_safe output for throttle sensor shall be set to 1
     - All other output state shall be set to 0
   - Fuel_mode shall be set to RICH

3. Manifold pressure sensor short to ground, open circuit (RICH mode)
   Total simulation time = 3 sec
   Inputs:
   - Set throttle sensor to normal operating condition for time = 0 to 3 sec
   - Set speed sensor to normal operating condition for time = 0 to 3 sec
   - Set EGO sensor to normal operating condition for time = 0 to 3 sec
   - Set MAP sensor to normal operating condition for time = 0 to 1 sec, then
     set MAP sensor error at time = 1.01 sec to 3 sec

Test Implementation
DEMO: Assessing Functional Requirements
DEMO: Assessing Verification Results using SDI*

*Simulation Data Inspector
DEMO: Automatic Assessment of Verification Results using Simulink Test Manager

Throttle Sensor Failure

Test Result Information
- Result Type: Test Case Result
- Parent: Requirements Tests (Individual)
- Start Time: 2016-Mar-30 13:32:03
- Outcome: Passed

Test Case Information
- Name: Throttle Sensor Failure
- Type: Simulation Test

Verify Result
- Test Assessment/Warm_Up:verify(fail_state(O2) = ...)
- Test Assessment/Warm_Up:verify(fuel_mode == ...
- Test Assessment/.../One_Sensor_Failure:verify(fu...
- Test Assessment/.../Multiple_Sensor_Failures:veri...
- Test Assessment/.../No_Sensor_Failure:verify(fuel...
- Sim Output (Step_00_logic : normal)
How much have we tested?

- Model Coverage
  - Structural metric
  - Measure of test completeness

\[ Z = 1 \text{ or } Z = -1 \]
Test completeness of the system and execution paths

- Data collected during simulation

<table>
<thead>
<tr>
<th>Test Case</th>
<th>D1</th>
<th>C1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Step_00_logic</td>
<td>51</td>
<td>83%</td>
</tr>
<tr>
<td>2. safety_logic</td>
<td>50</td>
<td>83%</td>
</tr>
<tr>
<td>3. SF: safety_logic</td>
<td>49</td>
<td>83%</td>
</tr>
<tr>
<td>4. SF: Fail</td>
<td>12</td>
<td>59%</td>
</tr>
<tr>
<td>5. SF: Multi</td>
<td>5</td>
<td>18%</td>
</tr>
<tr>
<td>6. SF: Fueling_Mode</td>
<td>18</td>
<td>85%</td>
</tr>
<tr>
<td>7. SF: Fuel_Disabled</td>
<td>4</td>
<td>83%</td>
</tr>
<tr>
<td>8. SF: Running</td>
<td>10</td>
<td>80%</td>
</tr>
<tr>
<td>9. SF: Low_Emissions</td>
<td>4</td>
<td>75%</td>
</tr>
<tr>
<td>10. SF: O2</td>
<td>5</td>
<td>100%</td>
</tr>
<tr>
<td>11. SF: Pressure</td>
<td>5</td>
<td>100%</td>
</tr>
<tr>
<td>12. SF: Speed</td>
<td>5</td>
<td>100%</td>
</tr>
<tr>
<td>13. SF: Throttle</td>
<td>5</td>
<td>100%</td>
</tr>
</tbody>
</table>
DEMO: Simulation Test Coverage of Models
Summary: Simulation-Based Testing

**Concept Phase**
- How can I interactively test and debug my system?
- How can I show what I have done to my manager?
- How can I perform reactive testing of my system?

**Production Phase**
- Does my system meet the design requirements?
- How complete is my testing?
  - The algorithm worked last week… does it still?
  - Another engineer modified the algorithm… still working ok?
  - Does the generated code performance match the model?
DEMO: Regression Test using Simulink Test Manager

Synchronization
DEMO: Regression Test using Simulink Test Manager
DEMO: Equivalence Test using Simulink Test Manager

SIL Equivalence Test

**Step 00 fuelsys Tests > Step 00 fuelsys > SIL Equivalence Test**

**Equivalence Test**

- **DESCRIPTION**
- **REQUIREMENTS**

**SIMULATION 1**

- **SYSTEM UNDER TEST**
  - Model: Step_00_fuelsys

- **TEST HARNESS**
  - Harness: Step_00_fuelsys_Reactive

**SIMULATION 2** Copy settings from Simulation 1

**SYSTEM UNDER TEST**

- Model: Step_00_fuelsys

- **TEST HARNESS**
  - Harness: Step_00_fuelsys_Reactive_SIL

**Test Result Information**

- **Result Type:** Test Case Result
- **Parent:** None
- **Start Time:** 2015-Sep-09 14:12:37
- **End Time:** 2015-Sep-09 14:12:51
- **Outcome:** Passed

**Test Case Information**

- **Name:** SIL Equivalence Test
- **Type:** Equivalence Test

**Equivalence Comparison**

<table>
<thead>
<tr>
<th>Name</th>
<th>Abs Tol</th>
<th>Rel Tol</th>
<th>Interp</th>
<th>Sync</th>
<th>Link to Plot</th>
</tr>
</thead>
<tbody>
<tr>
<td>air_fuel_ratio</td>
<td>0</td>
<td>0</td>
<td>linear</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>throttle</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>speed</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>ego</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>map</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>fail_state[1]</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>fail_state[2]</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>fail_state[3]</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>fail_state[4]</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>fail_state[5]</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>fuel_mode</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
<tr>
<td>fuel_rate</td>
<td>0</td>
<td>0</td>
<td>zoh</td>
<td>union</td>
<td>Link</td>
</tr>
</tbody>
</table>
## Summary: Simulation-Based Testing

### Concept Phase
- How can I interactively test and debug my system? ✔
- How can I show what I have done to my manager? ✔
- How can I perform reactive testing of my system? ✔

### Production Phase
- Does my system meet the design requirements? ✔
- How complete is my testing? ✔
- The algorithm worked last week… does it still? ✔
- Another engineer modified the algorithm… still working ok? ✔
- Does the generated code performance match the model? ✔
Q&A