POWERTRAIN CONTROL FEATURE DEVELOPMENT THROUGH MODEL BASED DESIGN
• Introduction to Powertrain
• Powertrain feature – Clutch Control
• Requirements
• Function Development
• Model verification
• Design verification & Coverage analysis
• Software in loop checking
• Auto code generation
• Concluding points
Outline

• This presentation describes how Model Based Design (MBD) concept is successfully deployed to address problem and challenges associated in designing complex powertrain control system.
Engine Electronics Interfaces

- **ECU**
- **Aux Systems**
- **Actuators**
- **Sensors**
- **Vehicle Electrical Systems**
- **Vehicle ECUs**
- **Diagnostics Programming**
- **Communication**
- **EOL**
Integrated Powertrain Control

Engine Management system

Transmission control system

Integration

Fuel Control

Gearshift

ECU

TCU

Coolant thermo sensor
Throttle position sensor
Knock sensor
Airflow sensor
Lambda sensor
Engine RPM sensor

Selector lever switch
Mode selection switch
Brake switch
Trans oil temp sensor
Trans oil pr sensor
Vehicle speed

GPS
Control Design Challenges

• Powertrain requires to deliver best - performance, emission, Fuel economy, drivability, safety etc.

• Capable of handling Multi-domain tasking

• Respond to tight coupling of powertrain components

• Manage lot of interdependencies and exchange of huge no. of parameters

• Ensure reliable working under all operating scenarios
Approach

• Conventional design approach has limitations in terms of analysis, testing, risk mitigation & confidence building
• MBD provides platform for quick building of control design and verification
• Support step by step design integrated with testing throughout the development cycle
• Provide ease of modification to refine algorithm to build optimum system
• Tools used
  • Matlab, Simulink, Stateflow
  • Embedded Coder
  • Simulink Design Verifier
  • Model Advisor
MBDS & Test technologies

Model advisor checks
SLDV

Function realization

Function/HW/SW Verification

Field trials
Powertrain Control feature - Clutch Control

- Engine Mgmt. System
- Sensor input
- Clutch control Module
- Actuator

[Diagram showing Clutch Control system including Engine Mgmt. System, Sensor input, Clutch control Module, and Actuator components like release fork, clutch master cylinder, fluid reservoir, release bearing, slave cylinder]
Key Challenges

• Auto Clutch control is combination of Manual and Automated actions

• Critical success factors
  • To be in sync & respond near real time basis
  • Clutch actuation trigger with driver input
  • Require continuous motoring of clutch movement & act accordingly
  • Fault management & safety
  • Any mismatch in timing may cause component damage and affect drivability
  • Need to incorporate self learning & neural logic to build right control mechanism
Requirements

• Stateflow and Simulink are used for gap analysis
• Multiple iterations of review and discussions were performed
• Requirements were in the form of text / diagrams
• Referencing of interfacing inputs and feedbacks
• Some High level requirement examples –
  • While changing gear clutch should be disengaged
  • While Brake pedal pressed for time-T, then Clutch Should be Disengaged
  • User can select reverse gear ONLY when vehicle speed is zero
  • Current gear & next gear should be identified and clutch shift timings to be varied accordingly
Requirement Reviews & Discussions

- While change in gear Clutch should be disengaged
- While Brake pedal press for T time then Clutch Should be Disengaged
- User can select reverse gear ONLY when vehicle speed is zero

- Added tuning parameters (timing, calibration parameter)
- While modeling missing parameters, relationships, interfaces were identified and corrected.
- Stateflow enabled to define transition, conditions and actions in the control logic
Derived Control Logic

Sample time = T m sec
Inputs simulation
Function development

<table>
<thead>
<tr>
<th>Local parameter</th>
<th>Range</th>
<th>Res</th>
<th>Unit</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>0-10</td>
<td>Sec</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>0-10</td>
<td>Sec</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>T3</td>
<td>0-10</td>
<td>Sec</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>T4</td>
<td>0-10</td>
<td>Sec</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Design Verification

• Requires to check following
  • Individual function
  • Possible real time failure
  • Impact of calibration limits
  • Diagnostics
  • Safety functions

• Following tools were used as part of design verification:
  • Model Advisor
  • Simulink Design Verification
  • Software in Loop
Model Analysis

- Following standard guideline checks were performed:
  - MATHWORKS Automotive Advisor Board
  - MISRA 2004
  - ISO 26262
  - IEC 61582
Model Advisor-Report

Simulink version: 8.5
System: Clutch_Eng_DEngage_1_dec_2015/AUTO_CLUTCH MODEL

Run Summary
Pass  Fail  Warning  Not Run  Total
125   2     42       47       216

Check for blocks not recommended for MISRA-C:2004 compliance

Identify blocks that are not supported or recommended for MISRA-C:2004 compliant code generation.

Passed
Blocks that are not supported or recommended for MISRA-C:2004 compliant code generation were not found in subsystem.

Check for Lookup Table blocks using cubic spline interpolation or extrapolation methods.

Passed
No Lookup Table blocks using cubic spline interpolation or extrapolation methods found.

Warnings are Corrected after analysis(e.g.)

- Identify signal labels that are not correct for C variable names.
- Check Simulink block or Stateflow objects that do not link to a requirement documents
- Identify mismatches between names of Stateflow ports and the associated signals.
Design verification

Simulink Design Verifier Results

Close results
Design error detection completed normally
All 74 objectives active logic.

Results:
- generate detailed analysis report
SLDV - Results

• Dead Logic Detection

<table>
<thead>
<tr>
<th>Objectives Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Objectives: 98</td>
</tr>
<tr>
<td>Dead Logic: 11</td>
</tr>
<tr>
<td>Active Logic: 87</td>
</tr>
</tbody>
</table>

• Check Specified Intermediate Minimum and Maximum Values

<table>
<thead>
<tr>
<th>Objectives Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Objectives: 0</td>
</tr>
</tbody>
</table>

• Division by zero

<table>
<thead>
<tr>
<th>Objectives Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Objectives: 0</td>
</tr>
</tbody>
</table>

• Integer Overflow

<table>
<thead>
<tr>
<th>Objectives Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Objectives: 2</td>
</tr>
<tr>
<td>Objectives Proven Valid: 1</td>
</tr>
<tr>
<td>Objectives Undecided when the Analysis was Stopped: 1</td>
</tr>
</tbody>
</table>

• Out of bound Array

<table>
<thead>
<tr>
<th>Objectives Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Objectives: 0</td>
</tr>
</tbody>
</table>
Model Coverage Analysis

- **Generated Input data**

  Generated Input Data

  | Time | 0-1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
  |------|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
  | Step | 1/2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
  | Manual_Switch | 0  | 1  | 1  | 1  | 1  | 2  | 1  | 0  | 0  | 2  | 1  | 0  | 0  | 1  | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
  | stMode       | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
  | gear_shift_Position | 0  | 3  | 3  | 0  | 3  | 3  | 3  | 3  | 3  | 1  | 3  | 3  | 3  | 3  | 3  | 1  | 1  | 3  | 3  | 0  | 1  | 0  | 1  | 3  | 2  | 1  | 3  | 3  | 3  | 2  | 1  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  |
  | gear_box_Position | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
  | Diagnostic    | 0  | 1  | 1  | 0  | 2  | 0  | 0  | 1  | 0  | 0  | 2  | 1  | 0  | 0  | 2  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

- **Coverage Report**

  Model Hierarchy/Complexity:

<table>
<thead>
<tr>
<th>AUTO CLUTCH MODEL S</th>
<th>D1</th>
<th>Test</th>
<th>MCDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTO CLUTCH MODEL</td>
<td>51</td>
<td>69%</td>
<td>38%</td>
</tr>
<tr>
<td>SF: AUTO CLUTCH MODEL</td>
<td>50</td>
<td>69%</td>
<td>38%</td>
</tr>
<tr>
<td>SF: MAIN FUNCTION</td>
<td>49</td>
<td>69%</td>
<td>38%</td>
</tr>
<tr>
<td>SF: CLUTCH ACTUATOR FUNCTION</td>
<td>46</td>
<td>69%</td>
<td>38%</td>
</tr>
<tr>
<td>SF: CLUTCH POSITION MODE</td>
<td>32</td>
<td>67%</td>
<td>33%</td>
</tr>
<tr>
<td>SF: CLUTCH PRESSURE MODE</td>
<td>16</td>
<td>67%</td>
<td>33%</td>
</tr>
</tbody>
</table>

© TATA MOTORS LTD.
Software in Loop

- Results of SIL are compared with Model test results
- Same test cases can be used
- Test source code on development computer
Auto code Generation

- Inline parameters
- Signal storage reuse
- Enable local block outputs
- Eliminate superfluous local variables (expression folding)
- Minimize data copies between local and global variables
- Use memcpy for vector assignment
- Use memory for vector assignment
- Loop unrolling thresholds
- Memory threshold (bytes)
- Memory threshold (bytes)
- Use memory for vector assignment
- Inherit from target

- Hardware Implementation
- Freescale
- Device type: 32-bit PowerPC
- Device vendor: Freescale
- Number of bits:
  - char: 8
  - short: 16
  - int: 32
  - long: 32
  - float: 32
  - double: 64
- Signed integer division results
- Byte ordering: Big Endian
- Emulation hardware (code generation only)
Concluding points

✓ Automotive Electronics has become competitive necessity to stay ahead of competitors

✓ Increased risk of product defects due to Complex E/E architecture, large number of ECU & seamless networking

✓ Test & development strategies to ensure flawless functioning of individual and all ECU’s together

✓ Way forward is to deploy robust development process, In depth testing, V&V

✓ Use of qualified and proven tools & services
Thanks for your attention
Summary

- As on now, attempt is made to check proof of concept
- Require thorough checking on testbed & on vehicle to firm up control strategies
- Important benefit derived is identifying gaps while modeling and same were addressed and verified
- Evolved model around 10 times to reach final executable requirements
- Realized: Timings, State transition sequence is a key parameter to achieve synchronization
- Need to incorporate self learning & neural logic to build right control mechanism