Best practices and Lessons Learnt during test case generation using Simulink Design Verifier for higher complexity models for testing C-code

- Guhan G
- Rajesh Kannan R
- Thavamani T

Embedded Software Department
Agenda

1. Introduction
2. Challenges in Test case generation
3. Alternative Method of Validation
4. Generation of Test cases using SLDV
5. Lessons Learnt in SLDV
6. Customization in SLDV
7. Results – SLDV Test Case generation
8. Benefits – SLDV Test Case generation
Introduction

- Automotive software is safety critical, hence ensuring the quality of the software plays a vital role.

- The test case are written against an algorithm, which is present in the form of MATLAB/SIMULINK model.

- User friendly environment is required to test the software (Code) and the SIMULINK model.
Challenges in Unit Testing Using Manual Test Cases & Other Tools

➢ **Manual Test Cases:**
  - Effort is High
  - Time to Market
  - Ensuring the quality of the test cases
  - Very tedious to achieve structural coverage in manual test cases
  - Extensive test cases to be written to detect design issues
  - Complex to test bigger modules

➢ **Other Tool Generated Test Cases:**
  - Possibility of generation of Redundant test cases
  - Reusability of the existing test cases are difficult
  - Complex to test bigger modules
  - Hard to find design issues
SLDV Test case Generation

- Easy to find design issues
- Increase in productivity
- Time to market
- Reusability of the generated test cases
- Very easy to get maximum structural coverage
- Simple to test bigger modules

Simulink Design Verifier
Generation of Test cases using SLDV

- **Input Model:**
  - The input SIMULINK model has to be provided with the necessary slope and bias information.

- **SLDV Settings:**
  - Appropriate objective and condition blocks has to be added with the intended values.
  - Based on the requirements, model coverage objectives has to be set.

- **SLDV Results:**
  - The intended input will be generated in harness model as signal builder.
  - The Test cases with the expected output can be obtained from the model_sldvdata.mat
## Challenges:

- Generation of Structural test cases
  - The test cases required to be based on structural as well as functional requirements
  - Based on the slope and bias information, the test has to be generated for the fixed point model.
  - Wide range of variation of values, to be generated for all the variables.
  - Interpolation tables to be tested with the appropriate values.
  - Some blocks are not supported in SLDV
  - Validating the calibrations present in the model.

## Work Arrounds:

- Appropriate Objective and condition blocks has to be added in the model with the intended values.
  - fixdt(1,16,2^0,0) values has to be entered in the signal attributes
  - Split up of ranges in the objective blocks can generate the various set of values
  - An objective block has to be added with the Map point table range for all the Interpolation outputs.
  - Replacement blocks with the same functionality can be replaced for the generation of the test cases.
  - The calibrations can be fed into “.m” file with the calibration values in the array format.
# Lessons learnt on Pilot Project

## CHALLENGES:

### Model Coverage

- The model should be always 100% coverage, other than the unreachable path in the model.

- Maximum analysis time for the test case generation has to be minimal.

- The majority of the generated test cases shall be in higher range values.

- Multiport switch coverage is not always 100%.

- All the objective blocks present in the model should be covered.

## WORK AROUNDS:

- For better coverage, Large Model (Non linear Extended) option has to be selected in test suite optimization.

- The sample time in the options “maximum analysis time” and “maximum test case steps” has to be inversely proportional.

- To attain the maximum variance of the output values, the input range can be set from [Mid… Max].

- Iterative methods with the different values, has to be used for the multiport switch coverage.

- The condition blocks has to be followed by objective blocks.
## Lessons learnt on Pilot Project

### CHALLENGES:

<table>
<thead>
<tr>
<th>Reusability</th>
<th>Work Arounonds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reusing the existing test cases for the coverage.</td>
<td>The old .mat files can be used in the “Extend existing test cases” option.</td>
</tr>
<tr>
<td>Test cases has to be generated apart from the existing test vectors/ model coverage.</td>
<td>The option “Ignore objectives satisfied by existing test cases” has to be selected to generate test cases other than the existing values.</td>
</tr>
</tbody>
</table>

### Results

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>The design errors present in the model has to be detected at the initial phase of the test case generation.</td>
<td>Option “Detect Design Errors” can be clicked, so that the design errors present in the model can be detected before the generation of the test cases.</td>
</tr>
<tr>
<td>Exporting the results with the expected output.</td>
<td>The Test cases with the expected output can be obtained in the .csv format from the model_sldvdata.mat</td>
</tr>
<tr>
<td>The model coverage has to be in the report format.</td>
<td>After generation of the test cases, Model coverage option can be clicked and the model coverage report can be extracted.</td>
</tr>
</tbody>
</table>
Customization in SLDV for RNTBCI Requirements

Automation tool is developed in M-scripts:

- To extract the Range, slope and Bias information from the requirements document.

- Addition of the Objectives and the condition blocks at the intended place with the required values.

- Options for the test case generation, either with both the fixed point or floating point values.

- Different iterations with range of values from [Minimum…Maximum], [Mid….Maximum].

- Addition of the scheduler blocks for testing the multiple events and schedulers.
The average productivity gain in Test case generation using SLDV, is ~67% for the 10 Sample Live modules.
Benefits of SLDV realized by the piloting team

✓ **Very low effort** for test case generation.

✓ **Early detection of Design issues** of the SIMULINK model.

✓ **User friendly** report with the coverage and test case details.

✓ Customization of test vectors as per the **functional requirement**.

✓ **Expected output** is obtained by setting the exact objective in the output port.

✓ **Reuse of old test cases** for the next evolution.
Thank you for your attention!

Any Questions?