MIL/SIL/PIL Approach
A new paradigm in Model Based Development

Narayananurthy Srinivas, Narendra Kumar Panditi
Stefan Schmidt, Ralf Garrelfs
Agenda

1. Motivation - Model Based Development (MBD)
2. Model verified by Simulation (MvS)
3. Case study on MIL/SIL/PIL
4. MIL/SIL/PIL Simulation results in SDA
5. Comparison of MIL/SIL/PIL results
6. Conclusion
## Motivation - Model Based Development (MBD)

<table>
<thead>
<tr>
<th></th>
<th>Manual</th>
<th>Model In the Loop (MIL)</th>
<th>Model In the Loop (MIL) Software In the Loop (SIL) Processor In the Loop (PIL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specification Design</td>
<td>Manual in the form of document</td>
<td>Model design using MBD MIL: Model verification</td>
<td>Model design using MBD MIL: Model verification</td>
</tr>
<tr>
<td>Coding</td>
<td>Manual coding</td>
<td>Auto code generation (ACG)</td>
<td>Auto code generation (ACG)</td>
</tr>
<tr>
<td>Code Verification</td>
<td>Manual prepared test cases to perform Unit Testing</td>
<td>Tool generated test cases to perform unit testing</td>
<td>Reuse MIL test cases SIL : Software verification PIL : Software verification on Target processor or equivalent instruction set simulator</td>
</tr>
</tbody>
</table>
Model Based Development: V-Cycle

- **FLP**: Floating point model
- **FXP**: Fixed point model
- **MvS**: Model verified by Simulation
- **SIL**: Software In the Loop
- **PIL**: Processor In the Loop
- **HIL**: Hardware In the Loop
- **TVG**: Test Vector Generation
Model Verified by Simulation (MvS)

MIL
Model In the Loop
- Test cases
- Functional Requirements
- Modeling
- Physical model (FLP)
- Scaling
- Implementation Model (FXP)
- Result

SIL
Software In the Loop
- ACG
- C-Code (s-function in model)
- Result

PIL
Processor In the Loop
- C-Code (compiled for µC-Target)
- Result
## Definition – MIL/SIL/PIL

<table>
<thead>
<tr>
<th>MIL</th>
<th>SIL</th>
<th>PIL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model In the Loop</strong></td>
<td><strong>Software In the Loop</strong></td>
<td><strong>Processor In the Loop</strong></td>
</tr>
<tr>
<td>Refers to the kind of testing done to verify the accuracy / acceptability of a plant model or a control system.</td>
<td>Refers to the kind of testing done to validate the behavior of the auto generated code used in the controller.</td>
<td>Refers to the kind of testing done to validate the referenced model by generating production code using the model reference target.</td>
</tr>
<tr>
<td>MIL testing means that the model and its environment are simulated in the modeling framework without any physical hardware components.</td>
<td>The embedded software is tested within a simulated environment model but without any hardware.</td>
<td>The code is cross-compiled for and executed on a target processor or an equivalent instruction set simulator.</td>
</tr>
<tr>
<td>MIL allows testing at early stages of the development cycle.</td>
<td>SIL also allows to verify the code coverage.</td>
<td>PIL level of testing can reveal faults that are caused by the target compiler or by the processor architecture.</td>
</tr>
</tbody>
</table>
Case study on Engine Temperature function

1. Test suite for calculation of load information for coolant temperature model
Model In the Loop (MIL): Floating point model

Stimuli/Inputs

Controller/Module

Display/Verify

Simulation project

ENTE_SIGCVTCOAI

LOAD_TCO_MDL

SLIMULI

MAF_KGH
LV_ES

<MAF_KGH>
<LV_ES>

FLP

LOAD_TCO_MDL

Narayanamurthy S, Narendrakumar P, Stefan Schmidt & Ralf Garrelfs © Continental AG
Model In the Loop (MIL): Fixed point model
MvS: SDA Simulation Manager

Deviations can be detected and can be solved at early stages.
Comparison results MIL - FLP/FXP (Error)

Deviations due to wrong scaling
Comparison results - FLP/FPX (Corrected Case)

Deviations are within the resolution.
Present Situation after MIL

1. Random test cases are generated to test production code.

2. Execute generated test cases in the project environment.

3. More effort is required to prepare test cases to verify production code.

4. Completely different test cases are used to verify model and generated code.
Wouldn’t it be nice to reuse the MIL test cases for test of the Automatically Generated Code?
Software in the Loop: SIL

Stimuli/Inputs

Sfunction

Simulation project

Display/Verify

STIMULI

MAF_KGH
V.5.8
LV_ES
V.5.8

SIL

LOAD_TCO_MDL
V.5.8

ENTE_SIGCVTCOAI
SimMode:SIL

LOAD_TCO_MDL

Display/Verify
Comparison results - MIL/SIL (wrong case)
Comparison results - MIL/SIL (correct case)

No Deviation
Processor In the Loop: PIL

Stimuli/Inputs

Sfunction block

Display/Verify
Comparison results - SIL/PIL
PIL results for different target processors - Reusability

No Deviations
Conclusion

1. Necessary test effort can be essentially minimized across simulations.

2. Tests suites are portable and reusable.

3. Cost-efficient consistent testing for all phases of the development: One test suite for all development phases (MIL, SIL, PIL).

4. Early malfunction detection.

5. Eases the updating of test suites for changed requirements.

6. Shorter development process resulting in significant time-to-market advantage.
Thank you for your attention!