Automotive Embedded Systems-Model Based Approach Review

Adnan Shaout
The department of Electrical and Computer Engineering
The University of Michigan - Dearborn
Dearborn, Michigan
shaout@umich.edu

Shanmukha Pattela
The department of Electrical and Computer Engineering
The University of Michigan - Dearborn
Dearborn, Michigan
Spattela@umich.edu

Abstract: The evolution of transforming from an electrical mechanical engineering discipline to a combination of software and electrical/mechanical engineering establishes software as a crucial technology. The current complex automotive system is the product of growth of embedded software. As a result, automotive industry focuses on a new trend Model based development rather than traditional method where software is handwritten in Assembly code or C language. This paper presents a review of the use of Model based Development to accelerate development process of embedded control systems and technologies. The paper also presents a review of the tools used to support Model-Based Development (MBD) from functional requirements to automated testing and Model based testing process.

Keywords: Model-based development, automotive embedded systems, embedded software, automotive industry.

Received March 24, 2022; accepted April 28, 2022
https://doi.org/10.34028/iajit/19/3A/5

1. Introduction

Automotive embedded systems become more and more complex. Traditional designing methodologies involve writing text specification and coding. However, handwritten code cannot be tested without hardware. This way leaves engineer to wait until the hardware is available to test their system. In order to overcome such roadblocks engineers should be able to separate both development and validation from the hardware availability. Model based design is one such approach which automotive industry is currently focusing.

For the past few decades, traditional development process has reviews, analysis and test activity which are handwritten and manual. Increased number of complex control systems and software-controlled innovations drive the engineers to implement model-based design approach. This is a design process based on a system model [1]. Table 1 shows the market challenges and model based development benefits [8].

More than 80% of the automotive software can be generated by models automatically using various tools reducing complexity comes with hand-coding. Advanced tools like MATLAB/SIMULINK, StateFlow (Mathworks), dSPACE TargetLink [22] provide developer to create functional models and generate AUTOSAR standard production code which can be deployed on to the target Electronic Control Units (ECUs) directly [15].

Thus, Model based design is considered as efficient approach with lot of advantages with respect to design and implement functionality up to verification and validation of models and auto generated C-language code [2]. However, quality assurance is important factor throughout the development process and testing is a key element for quality assurance [20].

This paper is an extension our paper [17] which was presented at the International Arab Conference on Information Technology (ACIT) at Department of Information Study, College of Arts and Social Sciences, Sultan Qaboos University, Muscat, Al-khoud, Oman, December 21-23, 2021.

This paper is organized as follows: State of the art research is presented in section 2. Section 3 presents the search process which is carried out by using three scientific databases as shown in Figure 2. Defining Modeling techniques and proposed Model based development technology are presented in section 4. Model based testing, some selected approaches and comparison of tools available and proposed tool are presented in section 5. Finally, conclusion on research done on various MBD approaches in context of automotive control systems is presented in section 6.

Table 1. Market challenges and model based development [8].

<table>
<thead>
<tr>
<th>Market Challenges</th>
<th>Model based Design benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increased complexity of control systems</td>
<td>Each development phase is linked by executable model</td>
</tr>
<tr>
<td>More and More product and Customer needs</td>
<td>Distributed software development with extensibility, testability and easy maintenance</td>
</tr>
<tr>
<td>Shorter time-to-prototype and time-to-market</td>
<td>Increase of reuse, productivity and reliability</td>
</tr>
<tr>
<td>Increased in cost in quality assurance</td>
<td>Auto generated production code</td>
</tr>
</tbody>
</table>

This paper is an extension our paper [17] which was presented at the International Arab Conference on Information Technology (ACIT) at Department of Information Study, College of Arts and Social Sciences, Sultan Qaboos University, Muscat, Al-khoud, Oman, December 21-23, 2021.

This paper is organized as follows: State of the art research is presented in section 2. Section 3 presents the search process which is carried out by using three scientific databases as shown in Figure 2. Defining Modeling techniques and proposed Model based development technology are presented in section 4. Model based testing, some selected approaches and comparison of tools available and proposed tool are presented in section 5. Finally, conclusion on research done on various MBD approaches in context of automotive control systems is presented in section 6.
2. State of The Art

Recent years, automotive domain is switching to implement model-based development for complex systems functionalities. The recommended solution is off the shelf state-of-the-art tools needs adaptation and configuration. MATLAB/Simulink can be chosen for developing and validating. The simulation of concepts composed with functional testing and verification can be performed using model-in-the-loop (MIL). Using model-based functionality design will reduce the amount of time-to-prototype. Workshops and dSPACE Autobox/MicroAutobox can be used for validation of functions. Therefore, implementation of code in the V-cycle would change from manual/handwritten C coding to automatic code generation for target ECUs using dSPACE Target Link.

The following key points are identified for generating production ready software using model based approach as shown in Figure 1 [8]:

- Model-based function design.
- Model-based software design.
- Model-based documentation.
- Model-based testing.

Figure 1 shows the roadmap for overview of automotive software model based approach in this paper.

Steps followed during search process to select relevant research papers are as follows:
1. Using the keywords searched four scientific databases and analyzes around 6700 search results.
2. We discarded 3500 researches based on Title of research.
3. We discarded 2300 researches by reading Abstract.
4. We read various relevant sections of each paper and discarded 950 researches those are not relevant for detailed study.
5. We performed detailed study of 100 research papers and discarded 60.
6. We selected 23 papers which are relevant to the topic we choose.

3. Model Based Approach

Many auto companies switching their approach to Model-Based Design process in their Electronic and Electrical Systems (E and ES) that focused on [16]:

- Executable system specifications and algorithm development using Simulink models.
- Rapid Prototyping for quick turnaround in
- Automatic Code generation from the Simulink models
- Continuous Test and Verification

As a result of their implementation of Model-Based Design, they were able to reduce the time to develop and implement a standard project.

Model based development process follows typical V-Model Process. Development starts with requirement capturing, analysis, functional model creation according to the clear, complete and unambiguous requirement document. At early stage we can able to test the functional model using virtual prototype.
Once system and ECU design are validated on virtual environment, design have to move to implementation phase. To generate code which satisfies all the requirements, model have to be imposed with software design, implementation details and constraints. Finally, integration of generated source code, I/O drivers and OS code is performed. Figure 3 shows multiple V-model which is different physical representations of same system on different abstraction levels, aiming for same final output [23].

**4.1. Benefits of Model Based Design**

Many automotive companies are switching to model based approach, including Caterpillar, General Motors, Toyota, Continental Teves, Jaguar and others [7]. Table 2 shows the analysis for Caterpillar implementation for a Model Based Design in their project as an example.

Table 2. Benefits of MBD using real time example.

<table>
<thead>
<tr>
<th>Benefit Factor</th>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost For Project</td>
<td>Reduced By Factor 2</td>
</tr>
<tr>
<td>Total Project Time</td>
<td>Reduced By Factor 2</td>
</tr>
<tr>
<td>Reduced Man Hours For Project</td>
<td>Reduced By Factor 2-4</td>
</tr>
<tr>
<td>Time For Completion Of Project</td>
<td>Twice As Quickly</td>
</tr>
</tbody>
</table>

**4.2. Other Model Based Techniques and Technologies**

Table 3 shows the different techniques of MDB.

Table 3. Techniques of MBD.

<table>
<thead>
<tr>
<th>Modeling Techniques</th>
<th>Modeling Technologies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical Modeling</td>
<td>MATLAB/SIMULINK/Stateflow [3, 7], LabView, Unified Modeling Language</td>
</tr>
<tr>
<td>Graphical Modeling</td>
<td>SCADE Suite [7], ASCET, Charon, Dymola, HYSDEL, Hy-Visual, Modelica, hySC</td>
</tr>
<tr>
<td>Integrative Modeling</td>
<td>MATLAB/SIMULINK/Stateflow, UML/SYSML/MARTE [4, 22], TargetLink [3]</td>
</tr>
<tr>
<td>Correct-by-Construction</td>
<td>Enable auto generated code ensuring that what is verified at embedded code level</td>
</tr>
</tbody>
</table>

Selection of modeling technology depends on the type of system being modeled and the task for which model is being developed [23]. The modeling can as follow:

- **Continuous Systems**: are best modelled by differential equations supplemented by algebraic constraints.
- **Discrete Systems**: demands Petri nets, finite state automata, timed communicating sequential process.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Matlab/Simulink/Stateflow</th>
<th>Lab View</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Dependencies</td>
<td>Supports hardware from multiple vendors</td>
<td>Only NI Hardware</td>
</tr>
<tr>
<td>Focus</td>
<td>Dynamic Simulations</td>
<td>Measuremnt Systems</td>
</tr>
<tr>
<td>Code generation</td>
<td>Generates production level code</td>
<td>No efficient code generators for dynamic simulations</td>
</tr>
</tbody>
</table>

1) MATLAB/SIMULINK/STATEFLOW:

Most advanced solutions for modeling automotive embedded systems is ML/SL/SF as shown in Figure 4. This tool is applicable to design 50% of behavior models within automotive control systems.

We can develop an embedded system using Model-Based approach with combined technology MATLAB/SIMULINK/STATEFLOW in more efficient manner, and reduce development time and cost than traditional manner of developing the control strategy. Figure 5 shows an example of developing Automotive Control System using ML/SL/SF [22].
4. Model-Based Test Process

The manual testing is simple and usually used for verification and validation. Tester should create test cases, execute and analyze the results manually. Depending on the tester analyzing skills and experience result and test coverage can be different. This method is costly, more error prone and time consuming. To improve this problem, test automation using test scripts was introduced. This method supports automatic test case execution and reports generation and saves time.

Therefore, automated testing for embedded software is a popular and widely used technique as it simplifies the testing effort and tests can run repeatedly at any point of time in a day [8]. Main important parts of test automation are: Test case generation, execution and reports analysis.

In automotive domain before the growth of software in control systems testing was performed in the areas like Electromagnetic Compatibility (EMC) testing, electrical testing (short- circuits, stress, current peaks), environmental testing( test under extreme climate conditions) and field testing(on road testing) [9]. Need for functional specific function testing methods came with the increased complexity of the system. Thus model based testing gained attention in recent years [10, 11, 12, 13].

Like Model based development process, model based testing follows the typical V-Process where development and testing activities start more or less right after the project starts [9]. Commonly defined test methodologies at each phase of automotive software development process are MIL, SIL, and HIL [9] etc., and describe in detail in Figure 6.

In general, automotive control systems need test scenarios with very precise sequence with respective to time. Since there are huge numbers of such scenarios and test cases to execute, it is obvious that automation is the only way.

2) Reuse of Test Assets between Integration levels

Functionality test of the model prior the software integrated into target ECUs includes intermediate levels as described below. It is recommended to re-use test assets throughout all the integration levels and implementation to eliminate ambiguity [21].

This results in reduction of efforts in test case design and allows for evaluation of test results between each integration levels. Test cases can be modified or adjusted in central model without updating a lot of different integration and implementation levels [9, 19].

Integration levels are as follows [6]:
- Model-in-the-loop (MIL)
- Software-in-the-loop (MIL)
- Process-in-the-loop (MIL)
- Hardware-in-the-loop (MIL)
- Test Rig
- Vehicle

3) Systematic Test case design:

Control systems interact with physical components have complex functionality with number of variables. Testing such complex systems requires smart way of test case selection to ensure all test relevant aspects are covered and redundancies avoided.

Categories of Model Based Testing:

Figure 7 shows the taxonomy overview of model based testing.

Figure 6. Automotive control system development process with relevant testing methods [5].

5.1. Requirements for MBT

1) Test Automation

In general, automotive control systems need test scenarios with very precise sequence with respective to time. Since there are huge numbers of such scenarios and test cases to execute, it is obvious that automation is the only way.

2) Reuse of Test Assets between Integration levels

Functionality test of the model prior the software integrated into target ECUs includes intermediate levels as described below. It is recommended to re-use test assets throughout all the integration levels and implementation to eliminate ambiguity [21].

This results in reduction of efforts in test case design and allows for evaluation of test results between each integration levels. Test cases can be modified or adjusted in central model without updating a lot of different integration and implementation levels [9, 19].

Integration levels are as follows [6]:
- Model-in-the-loop (MIL)
- Software-in-the-loop (MIL)
- Process-in-the-loop (MIL)
- Hardware-in-the-loop (MIL)
- Test Rig
- Vehicle

3) Systematic Test case design:

Control systems interact with physical components have complex functionality with number of variables. Testing such complex systems requires smart way of test case selection to ensure all test relevant aspects are covered and redundancies avoided.

Categories of Model Based Testing:

Figure 7 shows the taxonomy overview of model based testing.

Figure 7. Taxonomy overview of model based testing [23].

5.2. Tools for Test Suite Generation

The following are several available automatic test suite generators in the market: Table 5 shows classification of
selected approaches based on taxonomy [23] and Table 6 classification of selected test approaches based on test specifications [23].

Table 5. Classification of selected approaches based on taxonomy [23].

```
<table>
<thead>
<tr>
<th>Selected Tools</th>
<th>Test Generation</th>
<th>Test Execution</th>
<th>Test Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedded Validator (EVM)</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Embedded Monitor (EM)</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Mixed Methods</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Reuse Test [20]</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Simulink Design Verification (SLDV)</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Simulink Design Verification (SLDV) (v1)</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>System Test</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>TPT [18]</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Transformation Approach</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>VISTA</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>
```

Table 6. Classification of selected test approaches based on test specifications [23].

```
<table>
<thead>
<tr>
<th>Selected Tools</th>
<th>Test Specification</th>
<th>Test Case</th>
<th>Test Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedded Validator (EVM)</td>
<td>Manual Test Case Specification</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Embedded Monitor (EM)</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Mixed Methods</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Reuse Test [20]</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Simulink Design Verification (SLDV)</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Simulink Design Verification (SLDV) (v1)</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>System Test</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>TPT [18]</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Transformation Approach</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>VISTA</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>
```

To satisfy all the mentioned requirements of model based testing, TPT approach is chosen and explained below. TPT test cases are independent of architecture and ensures to run /reuse them on different test platforms. However, the chosen TPT method can be adopted to any other test suite format that has the vectors of input & output singles or formally defined statements.

5.3. TPT for Automotive Model-Based Testing

Figure 8 shows the model based testing workflow.

Simulink Design Verifier Documentation—Math Works (SLDV) is capable of both automatic generation of test cases and proving properties. The advantage of using SLDV is that both property and system modeling can occur in the same environment. Properties and models used for test case generation can be created in a wide verity of ways. It can be done in either Simulink, Stateflow or as MATLAB code.

TPT test process

1. Test case design

Throughout test case design, test cases are chosen and modeled by means of the graphical test modeling language. The base of this test case design is the functional system requirements. So tests cases modeled with Time Partition Testing (TPT) approach are black-box tests [18]. Figure 9 shows the Mat Lab automatic test case generation [18].

2. Compiling

TPT VM (Virtual Machine)-Executes Test cases which are compiled into highly compacted byte code representations. The byte code has been particularly planned for TPT and contains precisely the set of operations, information sorts, and structures that are required to computerize TPT tests. This concept guarantees that test cases as well as the TPT - VM have a little impression. This can be vital in test situations with constrained memory and CPU resources, such as PIL and HIL [18].

3. Test execution

During test execution the TPT-VM executes the byte code of the test cases and communicates constantly with the SUT via platform adapters. The platform adapter is also liable for recording all signals all through the test run. Due to the separation between test modeling and execution, tests can run on different test platforms such as MIL, SIL, PIL, and HiL environments. HiL environments (which usually run in real-time) can be automated with TPT tests because the TPT - VM is able to run in real-time too. The unique semantic model of TPT test cases allows the test execution on each test
6. Conclusions

This paper presented the significant characteristics of automotive model-based development processes and the need for the testing process in early stages. The paper also explained taxonomy of MBT that covers main aspects of MBT approached. This intended to help with understanding the characteristics, similarities and differences of the different approaches, and an approach MBT tool is provided. Further research can be extended to explore about UML SYSML/MARTE language for modeling embedded systems and investigate more on tools for development.

References


Adnan Shaout is a full professor and a Fulbright Scholar in the Computer Science Department at the Electrical and Computer Engineering Department at the University of Michigan – Dearborn. At present, he teaches courses in AI, Embedded Systems, Software Engineering, Computer Architecture, Cloud Computing, Fuzzy Logic and Engineering Applications and Computer Hardware Design. His current research is in applications of software engineering methods, cloud computing, embedded systems, fuzzy systems, real time systems and artificial intelligence. Dr. Shaout has more than 40 years of experience in teaching and conducting research in the Computer Science, Electrical and Computer Engineering fields at Syracuse University and the University of Michigan - Dearborn. Dr. Shaout has published over 280 papers in topics related to Computer Science, Electrical and Computer Engineering fields. Dr. Shaout has obtained his B.Sc, M.S. and Ph.D. in Computer Engineering from Syracuse University, Syracuse, NY, in 1982, 1983, 1987, respectively.

Shannukha Pattela Currently she is a HIL Engineer at Ford Motor Company since May 2021.She has an MS degree in Computer Engineering from the University of Michigan-Dearborn in 2020.