Model-based Testing of Automotive Systems

Eckard Bringmann, Andreas Krämer

Presented By: Sharon Choy


QNX
QNX SOFTWARE SYSTEMS
BKFV 716

Background

- Software is a key technology
- Use of software has increased from 20% to 80%
- Development process is founded on model-based technologies
- Model-based technologies have many advantages for the automotive domain

Previous Work

- Examples of model-based technologies used in the automotive domain:
  - MATLAB/Simulink
  - Statemate
  - MatrixX
  - LabView

Motivation for Model-Based Technologies

- Handle and process continuous signals
- Allow development of high-level models that can be used for simulation
- Improve communication within development teams

Motivation of this Model-Based Testing

- Quality assurance (testing) is poorly supported
- Creating testing solutions is expensive

Main Contribution

- Discusses the characteristics of automotive model-based development processes
- Discusses the consequences for test development
- Motivates the need to reconsider testing for automotive systems
- Introduces the TPT Test Tool

Model-based Testing in Practice

- Need a virtual environment before implementation and integration to the ECU (Electronic Control Unit)
- Testing embedded devices involves considering the following
  - Electromagnetic compatibility
  - Electrical tests
  - Environmental tests
  - Field tests

Requirements for Automotive MBT

- Test Automation
  - Iterative process for interim releases
- Portability between integration levels
  - Model-in-the-loop
  - Software-in-the-loop
  - Processor-in-the-loop
  - Hardware-in-the-loop
  - Test Rig
  - Car

Requirements for Automotive MBT

- Systematic Test Case Design
- Readability
- Reactive testing/Closed loop testing
- Real-time issues and continuous signals
- Testing with continuous signals

TPT For Automotive Model-Based Testing

- Current technologies lack support for desired features
- Current solutions focus on individual model-based development projects
- New approach (Time Partition Testing) addresses the challenge of model-based automotive testing

Time Partition Testing Goals

- To support a test modeling technique that allows systematic selection of test cases
- To facilitate a precise, formal, portable and simple representation of test cases
- To provide an infrastructure for automated test execution and assessment

Time Partition Testing - Design

- TPT test cases are independent of the underlying software architecture and test platform
- TPT test method is strongly associated with the corresponding a test language

TPT Test Process

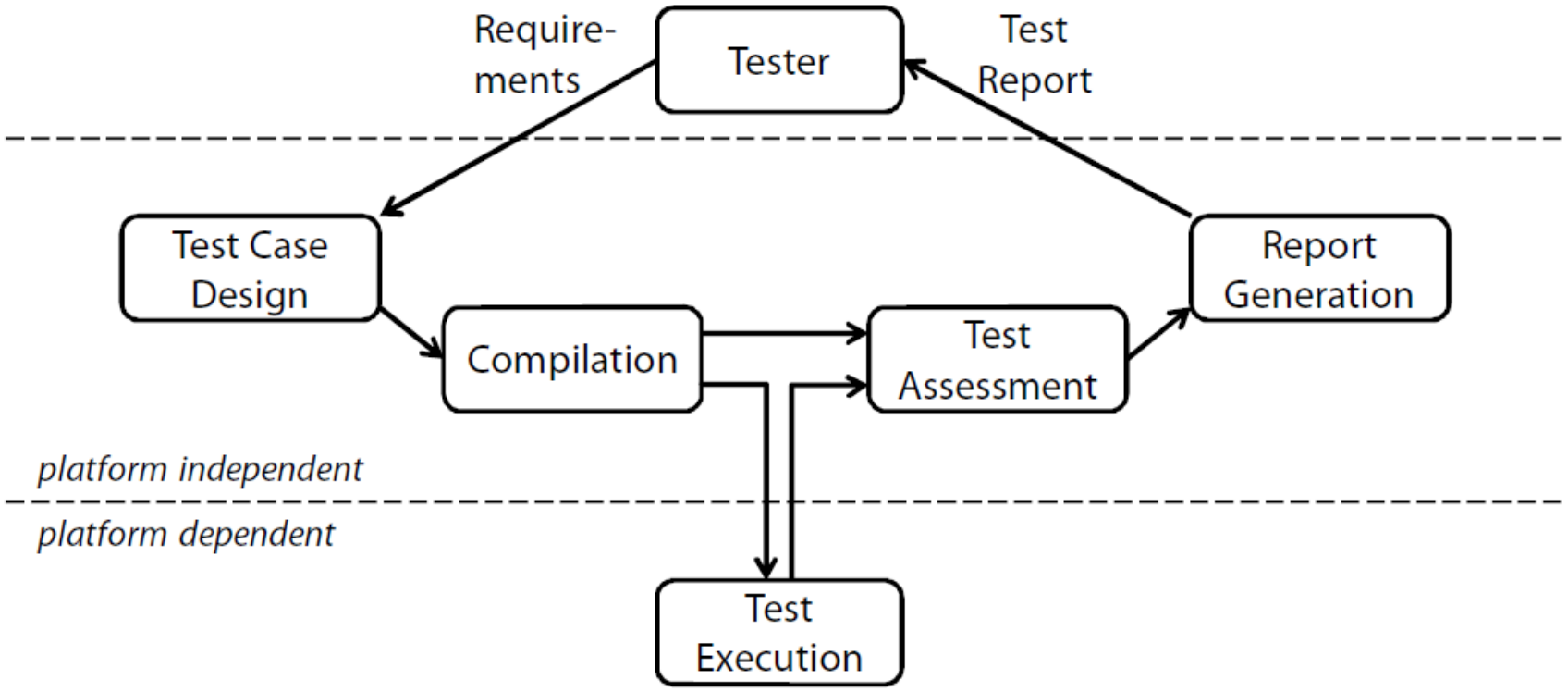

Figure 1: TPT test process

Case Study

- Case study of a simplified version of an exterior headlight controller (EHLC)
  - Three states: ON, OFF, AUTO
  - Lights turn on depending on the state and the ambient light
- Modeled using MATLAB/Simulink

Top-level model of EHLC

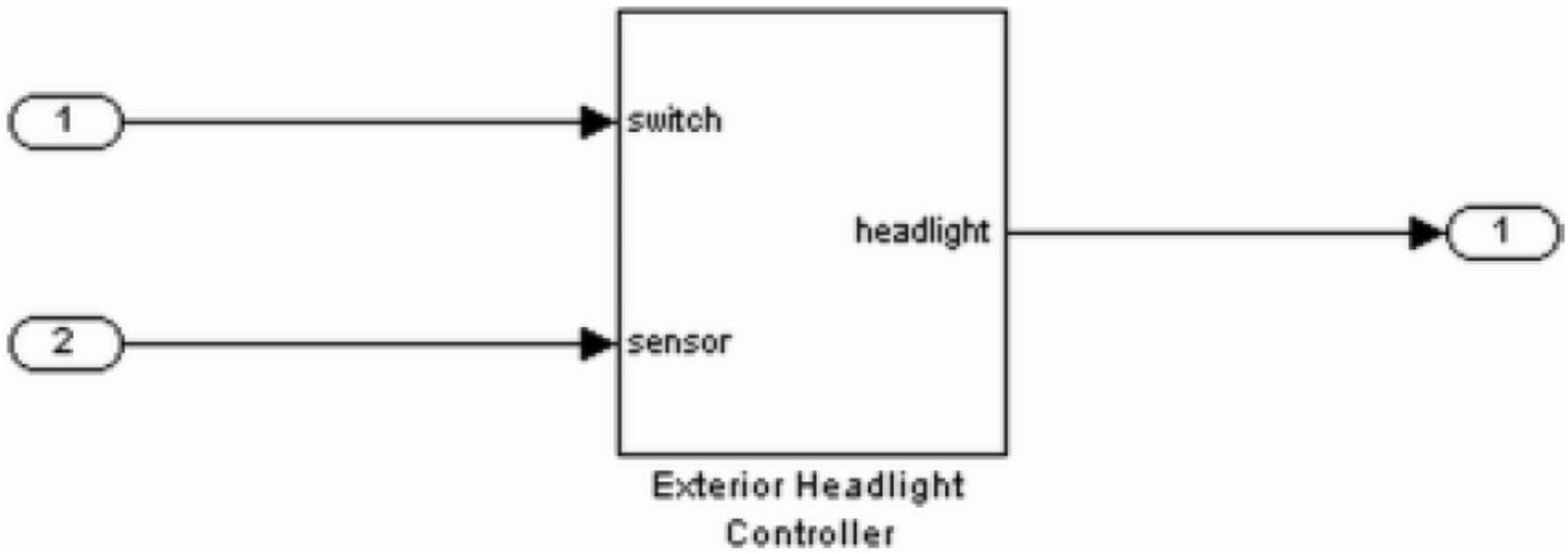

Figure 2: Top-level model of EHLC

Test Case with TPT

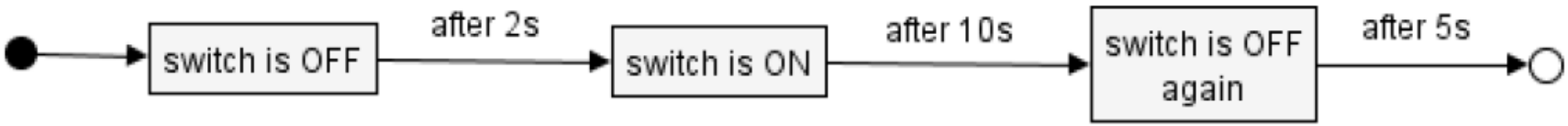

Figure 3: First test case OFF-ON-OFF

Test Case With Formal Statements

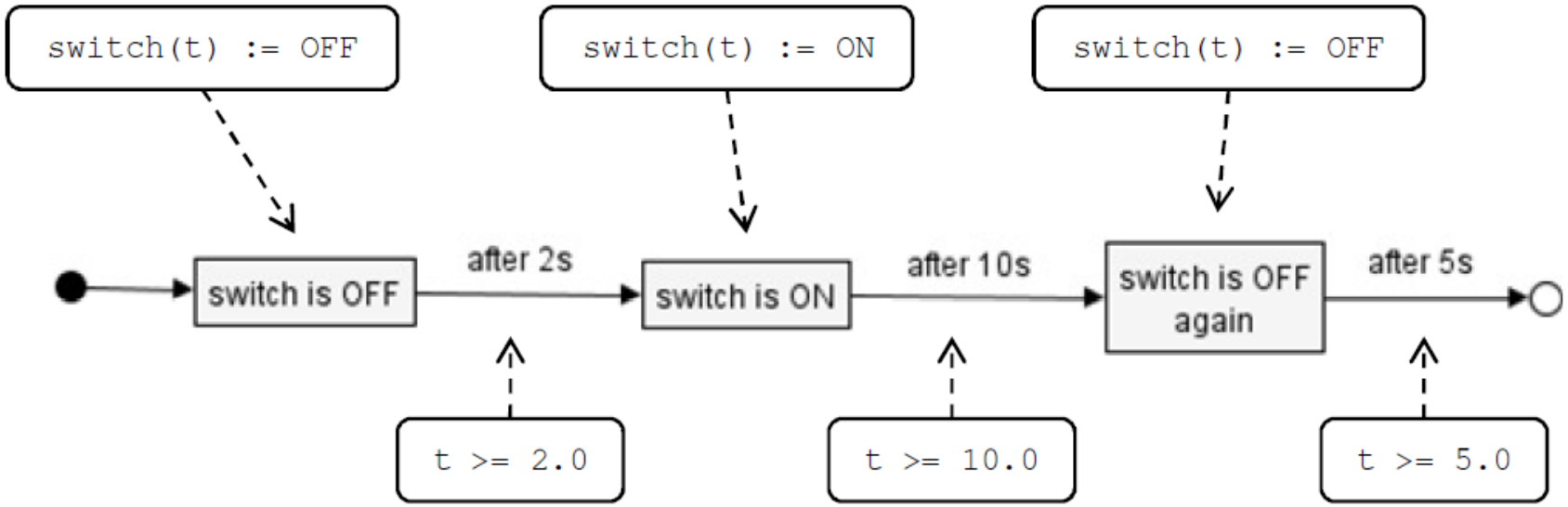

Figure 4: First test case with formal statements

The Complete Test Case

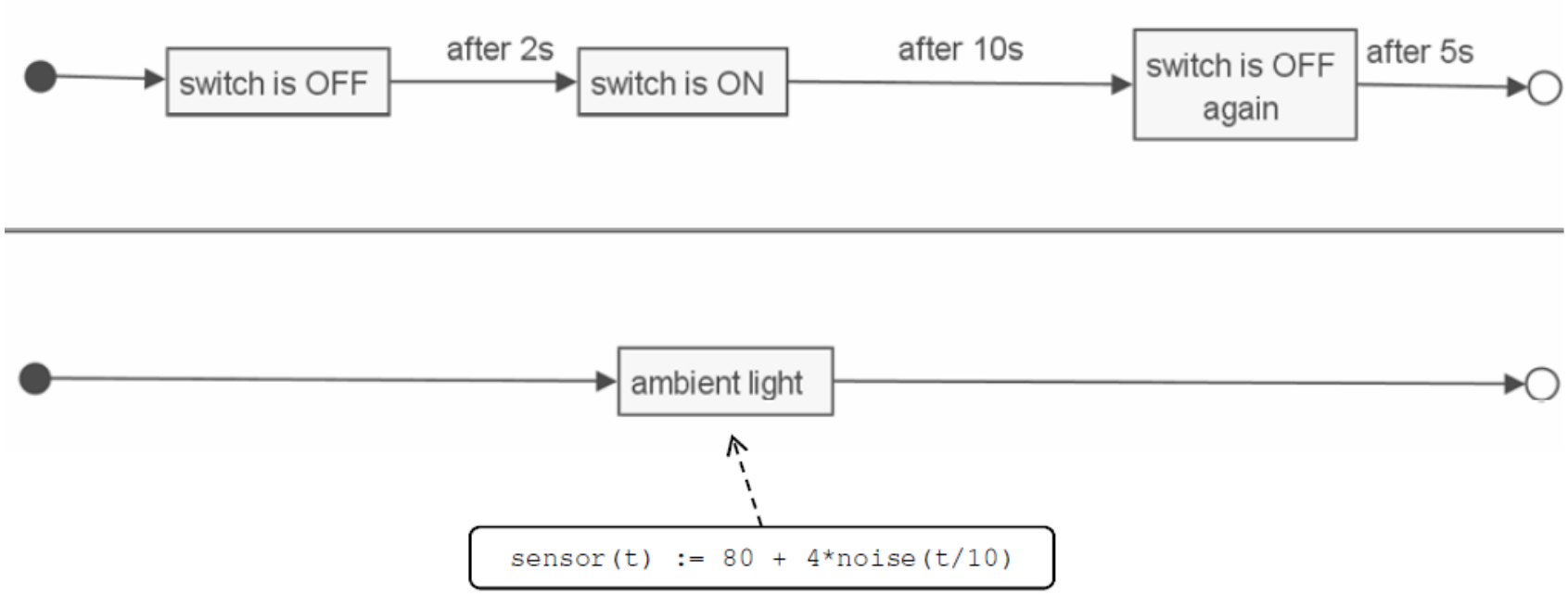

Figure 5: The complete test case

Reading Results

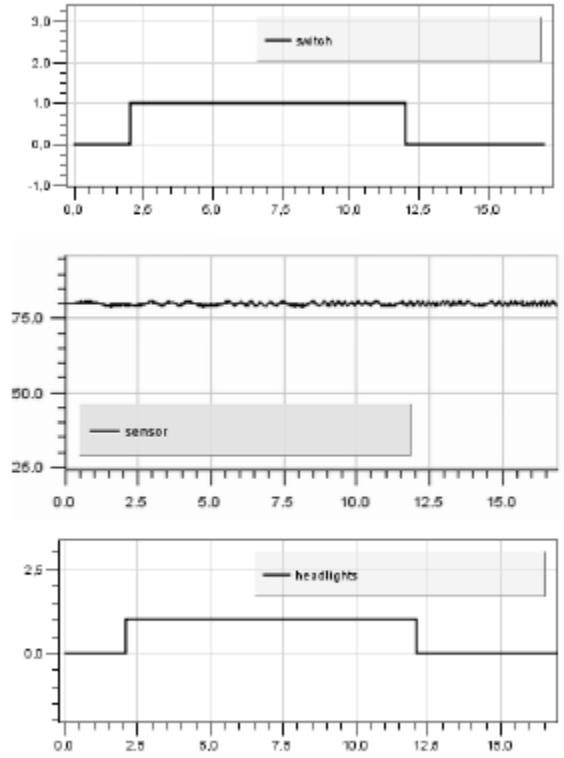

Figure 6: Test data recorded during execution

Advantages to Using TPT

- Suppose we have two similar cases
  - If test are modeled independently, comparisons are difficult
- All TPT test cases are integrated into a single test model that allows them to share common parts

Managing Test Cases Using TPT

- Variations in test cases could result in millions of combinations
- To keep large models comprehensible, TPT utilizes a classification tree
- Each test model automatically generates classification tree and combination table
- Classification tree can express logical constraints between classes

Classification Tree

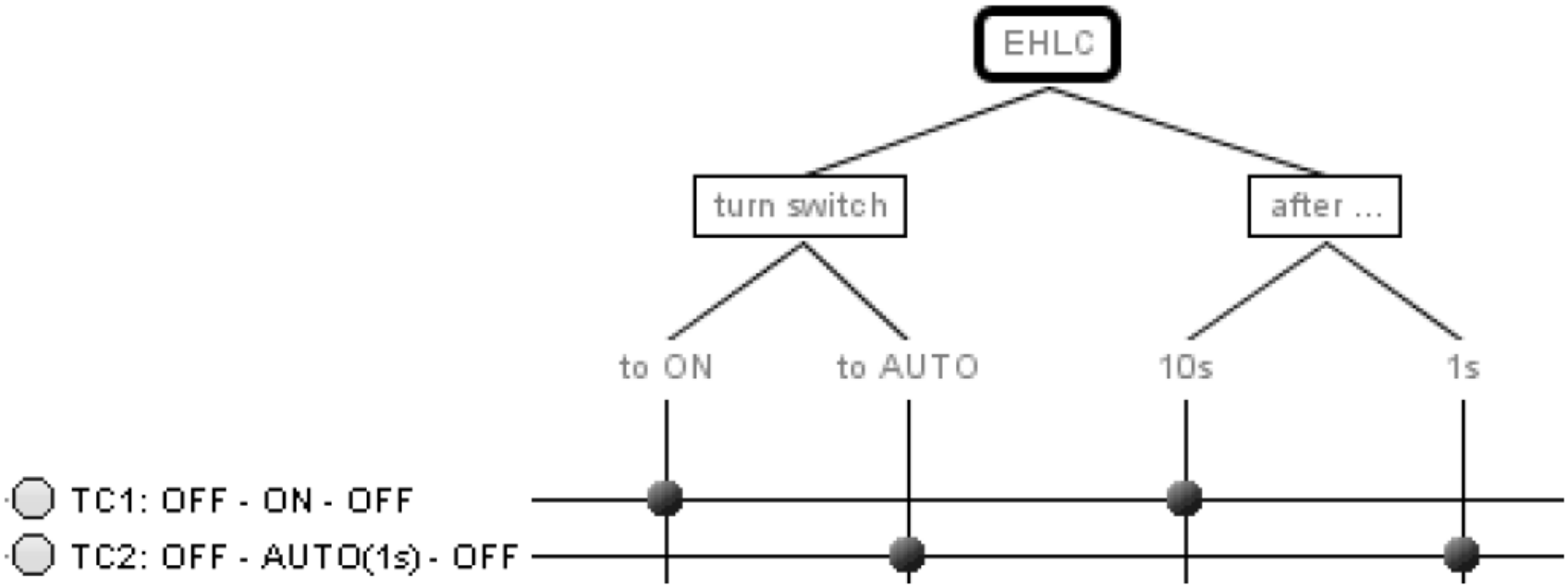

Figure 8: Classification tree as an alternative view

Practical Experience

- Initially developed at Daimler Software Technology Research
- Used in many production-vehicle development projects
- Test cases can run on different platforms without modification

Future Work

- Test management
- Test coverage measurement
- Data logging
- Diagnosis handling
- Integration of configuration management and testing

Conclusion

- Model-based development caused a radical change in the automotive industry
- Software needs to be tested as early as possible
- TPT facilitates the design, execution, assessment and report generation of test cases for automotive model-based systems

Discussion – Usage of Model-Based Testing

- Aside from simulating complex environments, what are some other advantages of model based testing?
- Are there any limitations to model-based testing?
- What are some barriers to using model-based testing?

Discussion – Extending Model-based Testing

- What would be helpful features for a model-based testing tool?
- How can we extend model-based testing beyond functional testing?
- Can model-based testing replace humans?