Automated model-based testing of control software with TPT

PikeTec GmbH
Dr. Eckard Bringmann
Motivation

Testing control systems requires **special features**.
What makes automotive testing special?

**Feature 1:**
Tests must be **portable** and **reusable** between the integration levels and technologies.

**Objective:** Test refinement, instead of test rewriting!
What makes automotive testing special?

**Feature 2:**
Tests must handle **signals**, **parameters**, and **diagnosis** in a simple way.

**System under Test**

**Controller**

**Stimulation**

**Test Case Behavior**

Data tables, recorded data, parameter sets
What makes automotive testing special?

Feature 3:
Tests must be able to react to system behavior online (closed-loop tests).

Expressive “closed-loop” test models

Stimulation

Observations

System under Test

Controller

Test Case Behavior

2008, PikeTec
What makes automotive testing special?

**Feature 4:**
Tests must be **real-time** enabled.

- **Stimulation**
- **System under Test**
  - ECU
- **Observations**
  - Test Case Behavior
  - Real time!
What makes automotive testing special?

Feature 5:
Test results should be evaluated automatically based on the requirements.
TPT is a test tool for testing control and feedback control systems
How to test with TPT

- Test modeling
- Test assessment
- Test documentation
- Test execution
- Platform specific

Test modeling

- assessment description
- scenario description
- fully automated

Test assessment

- test results
- data logs

Test execution

- TPT Virtual Machine
- Platform adapter
Test modeling

Modeling characteristics

- Graphical test modeling
- Based on automata (hybrid, hierarchical, parallel)
- Supports modeling of signal data
- Unique approach to integrate all test cases into a single test model
- Clear structured, compact test models
- Independent of the test platform
Automated test execution

TPT VM is embedded into a dedicated module in the environment

Communication with ECU code and environment (car model) is managed by the backbone

Easy to integrate (less than one week)
Example 2: Matlab/Simulink environment

- Automated test execution
- TPT VM is embedded into a Simulink S-function
- Communication with ECU code and environment (car model) is managed by Simulink
- Interface is specified by a generated m-script
- Test cases are hand over by means of workspace variables
- RTW and Targetlink enabled
Example 3: Module testing (‘C’ Code)

- test frame =
  - custom c code (SUT)
  - tpt-vm.dll
  - glue code

- test frame runs the test cases using the TPT VM and records result signals to a file
TPT Test Process

- Test modeling
- scenario description
- assessment description
- Test results
- data logs
- Test assessment
- fully automated
- Test documentation
- TPT Virtual Machine
- Platform adapter
- Platform specific
- SUT
- Test execution
Test assessment

- Logical assessment criteria + signal comparison

- **Online assessments** integrated in automata
- **Offline assessments** using Extended Python
  - Minimize online efforts
  - Huge library, reference comparison, access to file system, external tools etc.
Test documentation

- Test documentation is generated automatically based on assessment results
- Reports may contain signals and variables
- Customizable based on report templates
Features:

Platform independent test models
Consistency from model to assessment and report
Automated tests (from test execution to test report)
Closed loop tests supported
Abstract test language
Systematic test case definition
Intuitive graphical models
Continuous behavior testing
Visit our booth 10.0 - 516