Virtual Integration and Test of AUTOSAR Systems

The approach of Vector: vVIRTUALtarget
## Agenda

1. Motivation
2. Use Cases of Virtualization
3. Concept of the vVIRTUALtarget
4. Summary
Motivation

AUTOSAR Complexity

- AUTOSAR is influenced by many parties
  - Complexity is increased due to different needs
  - Acceptance is also increased
- AUTOSAR adds complexity to the development process but offers a lot of advantages:
  - Clearly defined and standardized interfaces within ECUs simplify interworking of OEMs and suppliers
  - Support of reuse and repartitioning of distributed systems
  - AUTOSAR modeling and test tool based
  - Interfaces offer new mechanisms for structured access of internal ECU data
Virtualization supports verification and validation along with the evolution of the SW

- Support of interactive and automated tests
- Reuse a given set of tests across the phases
- Allow an early start of test and integration
Motivation

Virtual Target

- Support the development and testing of automotive SW using a virtual environment
- “Virtual” means: use a generic platform (e.g. PC) rather than real ECUs

- Two main use cases
  - Execution of ECU code for functional and software testing purposes
  - Integration environment for AUTOSAR ECUs for checking the correct configuration of the ASR stack
## Agenda

1. Motivation
2. Use Cases of Virtualization
3. Concept of the vVIRTUALtarget
4. Summary
Use Cases of Virtualization

Software Development

- Development of functional code in a virtual environment
- Integration in realistic AUTOSAR environment
- Easy, mainly automated configuration
- Parallel Development of test libraries for later reuse
- Stimulation via busses, IO and internal ports
Use Cases of Virtualization

ECU Integration

- Minimum effort for parallel configuration of real and virtual target
- Validation of configuration for the real target
- Verification of the configuration in the virtual environment without the need for hardware
- Debugging of behavior is possible, e.g. start up or sleep mode
Use Cases of Virtualization

System Integration

- Test of distributed functions in virtual or partially virtual environment
- Reuse of already configured virtual SUTs also in combination with real ECUs
- Usage of test libraries developed in earlier phases
- Stimulation via busses, IO and even RTE ports
Use Cases of Virtualization

Error reproduction

- Reproduce functional misbehavior with measurements of field test in the virtual world
- Difficult achievable situations can be generated virtually
- Simulation time is not linked to real time, so pausing and resuming when debugging is possible
Scope of the Tool

- **vVIRTUALtarget**
  - Configures the setup
  - Executes the virtual target

- An external test environment executes the test
  - Stimulation and observation of the SUT
  - Environment simulation (plant model)
  - Visualization and Logging
  - Test reporting generation

Test Tool (e.g. CANoe)

Diagram of vVIRTUALtarget
Concept of the vVIRTUALtarget

Use Case and Usage

Software Development

ECU Integration

SWC and their interaction

Single SWC

SWC and MICROSAR BSW

OS

BSW

MCAL

SWC and OEM-specific MICROSAR BSW

Virtual hardware (PC)

Real target hardware
Concept of the vVIRTUALtarget

ECU Integration

A: DualTarget Configuration

- SWC A
- SWC B
- SWC C
- RTE
- OS
- BSW
- MCAL
- VTT MCAL

vVIRTUALtarget

SWC and OEM-specific MICRO SAR BSW

B: ECU Integration

- SWC A
- SWC B
- SWC C
- RTE
- OS
- BSW
- MCAL

C: ECU Integration on VTT

- SWC A
- SWC B
- SWC C
- RTE
- OS
- BSW
- VTT MCAL

One source, two targets
Easy Switching to and from Virtual

Concept of the vVIRTUA因而get

Running and Testing on your PC with **vVIRTUA因而get**
- Hardware-independent features

Running and Testing on **real target hardware**
- Hardware-dependent features
ECU Integration Workflow with vVIRTUAItarget basic

Concept of the vVIRTUAItarget

Vector delivery

Vector SIP

DaVinci Configurator Pro
Configuration of RTE + BSW

One source, two targets

MS™ Visual Studio Project
BSW module configuration header and code files
RTE header and code files
SWC header files

Application source code
e.g. runnable code
Skeletons generated, code added manually

Output Real Hardware
BSW module configuration header and code files
RTE header and code files
SWC header files

.runnable code
Skeletons generated, code added manually

.DLL
Concept of the vVIRTUALtarget

Software Development

- System under Test: Single SWC, Several SWCs, Single ECU, ECU Network
- Intra-ECU Communication available as Interface
- Test Interface: automatically generated Test API
- Debug Environment: automatically generated Visual Studio 2010 Solution
Overview of Interfaces

- Two possibilities to interact with vVIRTUALtarget
  - Intrinsic interface to test tools like CANoe
  - Generic interface for easy adaptation to other test tools with an automatically generated API

- Types of interfaces
  - Interface to bus systems and I/Os
  - Interfaces to ECU internal ports defined by AUTOSAR
  - API to special tasks like stress tests, state changes, internal interfaces to provoke e.g. race conditions
  - Control of simulation time
Timing Concept

- The CPU model supports **preemption** of the simulated code
- **Computation time** for sections of the code can be configured
- This offers tests for
  - Behavior of interrupts in general
  - Detection and reaction in situations of high system load
Agenda

1. Motivation

2. Use Cases of Virtualization

3. Concept of the vVIRTUALtarget

4. Summary
Summary

Core Benefits

- PC environment has advantages compared to a real HW
  - Better availability
    - Early test/integration results since the PC is always available – no need to wait for the HW
    - Easy replication of test-/integration environments on several work places
  - Better control
    - Stress scenarios can be easily applied to provoke “race conditions” and to achieve good code coverage
    - Faster execution of tests possible
    - More comprehensive test automation possible since SUT can be controlled completely
    - Access to internal variables at defined times possible
Do you have any questions?

Author:
Marktl, Roman
Vector Informatik GmbH