Virtual Platform Environment for the Bring Up and Test of a Secure Many-Core RTOS for Automotive Use

Atsushi Shinbo and Shuzo Tanaka, eSOL TRINITY Co., Ltd.

Masaki Gondo, eSOL Co., Ltd.

Duncan Graham and Larry Lapides, Imperas Software Ltd.

Embedded World conference
28 February 2018
Agenda

- The RTOS challenge for automotive systems
- Virtual platforms for software development
- Building the virtual platform
- eMCOS RTOS
- Debug and test of the RTOS
Agenda

- The RTOS challenge for automotive systems
- Virtual platforms for software development
- Building the virtual platform
- eMCOS RTOS
- Debug and test of the RTOS
Automotive Electronics Is Not Just ADAS and Autonomous Vehicles

- Classic automotive electronics – power train, braking systems, body control – have become more complex
- SoCs for classic automotive applications now have multiple processors
- ECUs for classic automotive applications now have multiple SoCs
- Automotive systems now include multiple ECUs communicating with each other
- Security requirements are now layered on top of the quality, reliability and safety requirements
Today’s Automotive Challenge

- How to provide a software environment that enables easy communication and control of the complex automotive systems?

- How to test such an environment?
One Answer

- How to provide a software environment that enables easy communication and control of the complex automotive systems?
- How to test such an environment?
- Develop a many-core RTOS that can support Autosar, including the security requirements, and test that RTOS/Autosar software stack using both virtual platforms (software simulation) and real hardware.
This Paper

- eMCOS RTOS / Autosar / RTE software stack
- ECU composed of 1 x RH850F1H
- 2 x ECU in virtual platform
- Testing of the software running on the “pseudo-ECU”
Agenda

- The RTOS challenge for automotive systems
- Virtual platforms for software development
- Building the virtual platform
- eMCOS RTOS
- Debug and test of the RTOS
Current Techniques for Embedded Software Testing

- Hardware based testing
  - Actual production hardware
  - Development boards, FPGA prototypes
  - Hardware emulators
- Cycle accurate simulation
- Instruction accurate simulation

- Hardware based testing is the norm
- Cycle accurate simulation is too slow, too expensive
- Instruction accurate simulation has advantages of controllability, observability, determinism, ease of automation
  - Now starting to move into mainstream as a complementary tool to hardware based testing
Hardware-Based Software Testing

- Has timing/cycle accuracy
- JTAG-based debug, trace
- Traditional development board / emulation based testing
  - Limited external test access (controllability)
  - Limited internal visibility
  - Limited physical system availability

- To get around these limitations, software is modified
  - printf
  - Debug versions of OS kernels
  - Instrumentation for specific analytical tools, e.g. code coverage, profiling

- Modified software may not have the same behavior as clean source code
Advantages of Virtual Platform Based Software Development

- Earlier system availability
- Full controllability of platform both from external ports and internal nodes
  - Corner cases can be tested
  - Errors can be made to happen
- Full visibility into platform: if an error occurs, it will be observed by the test environment
- Fully deterministic testing
- Easy to replicate platform and test environment to support automated CI and regression testing on compute farms
Virtual Platforms Complement Hardware-Based Software Development

- Current methodology employs testing on hardware
  - Proven methodology
  - Has significant limitations
- Virtual platform based methodology delivers controllability, visibility, repeatability, automation

Application Layer: Customer Differentiation
Middleware: TCP/IP, DHCP, LCD, …
OS: Linux, FreeRTOS, µC/OS-III, ThreadX, …
Drivers: USB, SPI, ethernet, …

Virtual platforms – software simulation – provide a complementary technology to the current methodology

Actual Hardware or Virtual Platform
Agenda

- The RTOS challenge for automotive systems
- Virtual platforms for software development
- **Building the virtual platform**
- eMCOS RTOS
- Debug and test of the RTOS
Virtual Platforms Provide a Simulation Environment Such That the Software Does Not Know That It Is Not Running On Hardware

- The virtual platform is a set of instruction accurate models that reflect the hardware on which the software will execute
  - Could be 1 SoC, multiple SoCs, board, system; no physical limitations
- Run the executables compiled for the target hardware
- Models are typically written in C or SystemC
- Models for individual components – interrupt controller, UART, ethernet, … – are connected just like in the hardware
- Peripheral components can be connected to the real world by using the host workstation resources: keyboard, mouse, screen, ethernet, USB, …
- High performance: 200 – 500 million instructions per second typical, or boots Linux in <5 sec
Renesas RH850F1H

- 2 x RH850G3M processors
- Lots of peripherals
- Have a test plan: what will be tested using the virtual platform, what with hardware based testing
- Only build the peripheral models that are needed for the virtual platform testing tasks
RH850F1H Virtual Platform: A Virtual, or Pseudo, ECU

- 2 x RH850G3M processor models
- UART
- INTC
- Timer
- Memory

- Processor models are from the Open Virtual Platforms (OVP) Library ([www.OVPworld.org](http://www.OVPworld.org))
- Peripherals models and platforms built using OVP APIs

✓ Building peripheral models and RH850F1H virtual platform, and initial bring up of eMCOS RTOS, took about 1 week
- True CAN model is not needed
- Test objective is to have communication between ECUs, not to test specific protocol
Agenda

- The RTOS challenge for automotive systems
- Virtual platforms for software development
- Building the virtual platform
- eMCOS RTOS
- Debug and test of the RTOS
eMCOS RTOS

- Distributed microkernel architecture
- Optimized for many-core hardware – does not depend on cache coherency
- Uses MPUs in target hardware to enable users to designate secure regions
- **eMCOS AUTOSAR**: eMCOS supporting AUTOSAR Classic Platform (CP) AUTOSAR OS specification
- **RTE**: The Run Time Environment module compliant to AUTOSAR Classic Platform RTE specification
  - RTE provides API to AUTOSAR CP application called SW-C (Software Components)
  - RTE provides communication between SW-Cs on the same ECU, and also between SW-Cs that resides in different ECUs via CAN, for example
Agenda

- The RTOS challenge for automotive systems
- Virtual platforms for software development
- Building the virtual platform
- eMCOS RTOS
- **Debug and test of the RTOS**
Imperas Environment for Embedded Software Development, Debug & Test

Application Software & Operating System

Virtual Platform
- Peripheral
- Memory
- CPU

Software Verification, Analysis & Profiling (VAP) tools
- Trace
- Profile
- Coverage
- Schedule
- Memory monitor
- Protocol checker
- Assertion checkers
- ...

JIT simulator engine

SlipStreamer API

Multiprocessor / Multicore Debugger

Eclipse IDE
Same cross-compiler with same compiler options as for target hardware is used to build software

The software should not know that it is not running on hardware
Test Objectives for the Virtual Platform Environment

- Verification of eMCOS/ Autosar/ RTE
- Enable Continuous Integration (CI) flow
- Enable multiple teams to use the same test environment
Test Results

- Virtual platform performance was > 200 MIPS
  - Performance of > 200 MIPS critical because of large test cases
- Virtual platform environment easy to replicate and deliver to additional engineering teams
- Visibility of virtual platform enabled debug of secure elements of software stack
- Easy to set up simulation in CI flow
Conclusions

- Using the virtual platform accelerated software testing
- Using the virtual platform caught bugs that would have been found much later in the test cycle, if at all
- Virtual platforms are a complementary technology to hardware based testing
  - Use the virtual platform where significant ROI can be achieved
- Further work: start using the virtual platform environment for code coverage, fault injection
- See eSOL at Hall 4, booth 4-634
- See Imperas at the RISC-V Foundation booth, Hall 3A, booth 3A-419

- Any questions?

- Thank you!