A SysML-based Framework with QEMU-SystemC Code Generation

Da He, Fabian Mischkalla, Wolfgang Mueller
University of Paderborn / C-Lab
da.he@c-lab.de
Motivation

UML as high level modeling language for electronic system design

⇒ Closing existing design gap from model specification to verification and synthesis

Executable specification for verification and synthesis

Model Specification

Design Gap

Simulation & Verification

Synthesis

© 2011 Siemens AG und Universität Paderborn
Agenda

- Motivation
- SysML-based codesign methodology
- System architecture model
- Automatic code generation
- SystemC-QEMU cosimulation framework
- Evaluation
- Summary
SysML-based Codesign Methodology

- System Architecture Model captured in Artisan Studio
- Systems Modeling Language (SysML) is a UML2 profile for system engineering
- Automatic code generation from model specification to HW/SW executable specification
- HW/SW coverification by means of SystemC-QEMU cosimulation
- Getting hardware prototype through HL synthesis
Cooperative Computing & Communication Laboratory

System Architecture Model

- **<<Processor>>**
  - cpu_0: CPU
  - ISA: PowerPC405
  - OS: Linux

- **<<Executable>>**
  - Application

- **<<SC Module>>**
  - mt_0: Master_Transactor
  - plb_0: BUS

- **<<SC Module>>**
  - st_0: Slave_Transactor
    - base_addr: 0xFF000000
    - high_addr: 0xFF001FFF
  - st_1: Memory_Transactor
    - base_addr: 0xFF001000
    - high_addr: 0xFF001FFF

- **<<SC Module>>**
  - slave_0: Slave
  - memory_0: Memory

Predefined Library Modules

- *h, *.c

Mischkalla, F., He, D., Mueller, W.: Closing the gap between UML-based modeling, simulation and synthesis of combined HW/SW systems. DATE 2010, Dresden

© 2011 Siemens AG und Universität Paderborn
Code Generation Scheme

UML/SysML model

<<Executable>> SW components

<<Processor>> Processors

<<SC Module>> HW components

Code Generation

External C code

Makefiles, Scripts

OS images for specific ISA

SystemC TLM IPs (Bus, Transactors)

SystemC modules

Compilation

Compilation

Compilation

Synchronization

SW Emulator (QEMU)

SW/OS Image

Simulation

SystemC model

Synthesis

VHDL files
Code Generation

<<Processor>>
cpu_0:CPU

<<Executable>>
Application

<<SC Module>>
mt_0:Master_Transactor

<<SC Module>>
plb_0:BUS

<<SC Module>>
st_0:Slave_Transactor

<<SC Module>>
slave_0:Slave

<<SC Module>>
memory_0:Memory

QEMU

CPU (e.g. PowerPC405, ARM926)

SW Application & OS

QEMU Plugin
Cosimulation API

Shared Memory

SystemC

<<SC Module>>
st_1:Memory_Transactor

<<SC Module>>
slave_0:Slave

<<SC Module>>
memory_0:Memory

TLM Bus

Master Transactor

Slave Transactor

Slave Module

Memory Transactor

Memory Module

Code Generation for "Executable"

**Driver function files** (Generated as .h and .c)
- Linux user-space
- Generic write/read
- Optimized for burst transfer
- Use `mmap` function

**I/O address map file** (Generated as .h)
- Base address
- High address
- Offset of each register
- Defined as macros

**Makefiles (Generated)**
- Generate application binaries

**Bash Script file (Generated)**
- Generate the file system
- Integration of application binaries into the file system
Cooperative Computing & Communication Laboratory

Code Generation for <<Processor>>

QEMU Plugin

- Initialization of Inter-Process Communication (IPC) channels via shared memory
- **Device registration via** `cpu_register_io_memory` and `cpu_register_physical_memory_offset`
- Implementation of read/write call-back functions (of type `CPUReadMemoryFunc` and `CPUWriteMemoryFunc`) by means of sending/receiving data to/from the IPC channels
- Optimized for burst transfer
QEMU-SystemC Cosimulation Framework

- QEMU and SystemC are running concurrently
- Data transfer via shared memory
- Support for burst transfer
- Support for *Programmers View* (PV), *Bus Accurate* (BA) and *Cycle Count Accurate* (CC) in TLM Bus
Evaluation

- Master-Slave example
- P2P data transfer
- IBM CoreConnect based architecture

- Complete system model in UML/SysML
- Executable model automatically generated
- Speedup achieved via PV and burst transfer
Summary

• SysML-based HW/SW combined codesign methodology

• Automatic code generation and configuration for executable model
  • Hardware component
  • Software component
  • Processor component

• QEMU-SystemC based cosimulation framework
Thank you for your attention!

Question?