Domain-specific front-end for virtual system modeling

Workshop on Graphical Modeling Language Development at ECMFA 2012

Janne Vatjus-Anttila, Jari Kreku, Kari Tiensyrjä
VTT Technical Research Centre of Finland
Outline

- Motivation
- System-level performance evaluation techniques
- ABSOLUT performance modeling and evaluation
  - Workload and platform modeling
  - Performance simulation
- Domain-specific front-end for ABSOLUT
  - Workload, platform and allocation modeling DSLs
- Case Example
- Conclusions
Motivation

- Increasing complexity of embedded system design
  - New methods and tools needed for increasing productivity
- ABSOLUT
  - Early-phase performance modeling and evaluation
  - Set of prototype tools
  - Command-line based tools: high learning curve
- Domain specific modeling (DSM)
  - Raise the level of abstraction above programming
  - Used on many different domains
- DSM was used in this work to shorten learning curve, improve usability and raise modeling abstraction of ABSOLUT
Techniques for system-level performance evaluation

- Virtual system
  - Abstract application and platform models (no ISS)
  - Instruction and cycle approximate
  - System level exploration
- Virtual platform
  - Real application software and a virtual platform model (ISS)
  - Instruction accurate
  - Software development
- Virtual prototype
  - Real application software and a virtual prototype (HDL)
  - Instruction accurate, clock accurate
  - Co-verification
ABSOLUT virtual system modeling

Application modelling
- Simulation control
- Utilisation of resources

Platform modelling
- Interfaces
- Resources
- Platform characteristics

Performance simulation

Simulation results
- OSCI SystemC
- Utilisation
- Latencies
- Etc
Workload modeling

Application control behavior

- Enables early simulation
- Improves simulation speed
- Reduces modeling effort

= Workload model

Data processing and memory access load

- Presents resource requirements of applications
- Platform independent
- Support tools (ABSINTH)
Platform capacity modeling

- Model of both hardware and platform software components and interconnections
  - Abstract, transaction-level SystemC models
  - Platform, subsystem and component layers
  - Computing, communication and storage resources
- Role in ABSOLUT
  - Processes load primitives
  - Provides higher-level services
  - Consumes time (cycle-approximate)
- Support tools and libraries (COGNAC, ALE)
Performance simulation

System is simulated using OSCI SystemC kernel 2.2

1. Workload models utilise resources with read, write, execute and/or service requests

2. OS models propagate load from WL models to the CPUs

3. CPU models model execution time for data processing instructions and propagate reads and writes to other components
Simulation results

- Instrumentation of workload and/or platform models performance and power probes
  - Status probes, e.g. resource utilisation
  - Timers, e.g. service processing time
  - Counters, e.g. number of reads and writes
- Visualisation tool (VODKA)
Domain-specific modeling (DSM)

- DSM commonly used as productivity tool for
  - Developing a DSL to model a specific application/product
  - Generating documentation, validity checking, code, etc.
- Domain-specific front-end for ABSOLUT
  - Develop DSLs for early phase embedded system design exploration
  - DSLs for the three modeling phases in ABSOLUT
    - Workload modeling
    - Platform modeling
    - Allocation
Workload modeling DSL

Define and generate workload models

Control (gzipped trace)

Workload primitives of basic blocks (XML)

Output: "pointers" to generated workload models
Platform modeling DSL

Design a platform model from existing components; generate (XML)

Components instantiated from model library

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Object is used to model processor and accelerator components.</td>
</tr>
<tr>
<td>Memory</td>
<td>Object is used to model memory components.</td>
</tr>
<tr>
<td>Bus</td>
<td>Object is used to model bus components.</td>
</tr>
<tr>
<td>Interface</td>
<td>Object is used to model interface of subsystem.</td>
</tr>
<tr>
<td>Subsystem</td>
<td>Object is used to model subsystems of platform.</td>
</tr>
<tr>
<td>Router</td>
<td>Object is used to model connections between subsystems.</td>
</tr>
<tr>
<td>Connection</td>
<td>Relationship used to model all connections in the platform diagram.</td>
</tr>
<tr>
<td>Master</td>
<td>Role that connects master component to connection.</td>
</tr>
<tr>
<td>Slave</td>
<td>Role that connects slave component to connection.</td>
</tr>
</tbody>
</table>
Allocation modeling DSL

Allocate workload models on the processing elements of the platform model

Output: allocation specification (INI-file)
ABSOLUT Y-chart approach with DSLs

- **Application modelling**
  - Simulation control
  - Utilisation of resources

- **Platform modelling**
  - Interfaces
  - Allocation
  - Resources
  - Platform characteristics
    - OSCS SystemC
    - Utilisation
    - Latencies
    - Etc

- **Simulation results**
  - Back-annotation

- **Back-annotation**
Case example: H.264 video player / recorder

- Graphical video player / recorder application
  - x264 codec
  - Single- and multi-threaded
- OMAP4-like execution platform
  - Two ARM Cortex A9 cores, crossbar, mobile DDR2 memory
  - Software decoding (no accelerators used)
- Modeling and simulation using DSLs with ABSOLUT
  - Back-annotation of filtered simulation results
- Verification of simulation results by comparing them with those measured on PandaBoard (work in progress)
Environment

- MetaEdit+ 4.5 Workbench
  - Definition of DSLs
  - Easy to use tool
  - Great support
- ABSOLUT
  - ABSINTH workload model generator
  - COGNAC platform model generator, ALE model library
  - BEER simulator
- Red Hat Enterprise Linux 6 operating system, Intel Core i7-based workstation
Results [1/2]

- Case study successfully modelled and simulated
  - Platform model with MetaEdit+, COGNAC and ALE
  - Workload models with MetaEdit+, ABSINTH
  - Simulation with BEER
- Experiences
  - Modelling/simulation UI improves usability and provides lower learning curve
    - Raises modelling abstraction
  - Platform and allocation DSLs work well
  - Workload DSL could be improved
  - Result backannation could be done in a limited way
Results [2/2]

- Example: CPU utilisation caused by the multi-threaded x264 workload
Conclusions

- New methods and tools needed for embedded system design due to the increasing complexity of both applications and platforms
- ABSOLUT is early-phase performance modeling and evaluation technique with a set of supporting tools
- DSLs for the workload, platform and allocation modeling phases of ABSOLUT implemented with MetaEdit+ and serve as front-ends
- MetaEdit+ and ABSOLUT combination experimented with a video player / recorder case example on OMAP4 platform model
- DSLs provided improved usability to the modelling/simulation flow
Acknowledgements

This work is supported by the European Commission and Tekes – the Finnish Funding Agency for Technology and Innovation - under the grant agreement ARTEMIS-2010-1-269362 PRESTO.

The adoption of DSM principles and MetaEdit+ environment was guided by experts of Meta-Case. Especially Mr. Janne Luoma contributed in solving issues faced during the work. Dr. Juha-Pekka Tolvanen receives also our gratitude for professional comments.

Thank you!