Software Performance Estimation in MPSoC Design

Marcio Seiji Oyamada\textsuperscript{1,2}, Flávio Rech Wagner\textsuperscript{1}, Wander Cesario\textsuperscript{2}, Marius Bonaciu\textsuperscript{2}, Ahmed Jerraya\textsuperscript{2}

\textsuperscript{1}UFRGS, Instituto de Informática, Porto Alegre, Brazil
http://www.inf.ufrgs.br/~lse

\textsuperscript{2}TIMA Laboratory, SLS Group, Grenoble, France
http://tima.imag.fr/sls
Motivation

- Very large design space of embedded MPSoCs
  - High-level performance estimation tools are required
  - Must be combined with fast design exploration strategies
- Embedded systems are software-dominated
  - Evaluation of processor performance under various workloads
    - e.g. exploring the allocation of tasks to various processors
Motivation

- Support for SW performance estimation and analysis at different abstraction levels
  - At specification level
    - Estimation must be fast, for fast design space exploration
    - Some inaccuracy is accepted
  - At RT level
    - Accurate performance analysis after the architecture definition
    - Evaluation of OS and communication overhead in an MPSoC environment
Motivation

• For accurate performance analysis and identification of bottlenecks, simulation models must be instrumented with appropriate profiling resources
• These models must be generated according to virtual prototypes coming from the synthesis flow
• However, there is a poor integration between the synthesis flow and the performance evaluation flow
  – Manual configurations of performance models are often required
Goals of this work

- Integrated methodology for software performance analysis at different abstraction levels
  - Different trade-offs between speed and accuracy of the performance analysis
  - High-level performance evaluation for fast design space exploration
- Performance evaluation flow tightly coupled with an MPSoC synthesis flow
Outline

1. MPSoC design flow
2. Software performance estimation methodology
3. Neural network-based performance estimation
4. Virtual prototype-based performance estimation
5. Case study: MPEG4 encoder
Outline

1. MPSoC design flow
2. Software performance estimation methodology
3. Neural network-based performance estimation
4. Virtual prototype-based performance estimation
5. Case study: MPEG4 encoder
MPSoC Design Flow

- System specification: functional components
MPSoC Design Flow

- System specification: functional components
- Architecture exploration maps the functionalities in HW and SW components
- Virtual architecture: hardware and software components with abstract communication channels
MPSoC Design Flow

- System specification: functional components
- Architecture exploration maps the functionalities in HW and SW components
- Virtual architecture: hardware and software components with abstract communication channels
MPSoC Design Flow

- System specification: functional components
- Architecture exploration maps the functionalities in HW and SW components
- Virtual architecture: hardware and software components with abstract communication channels
- Abstract interfaces are refined in hardware and software interfaces
- BFM Level:
  - CPU
  - Interconnection network and adapters
MPSoC Design Flow

- System specification: functional components
- Architecture exploration maps the functionalities in HW and SW components
- Virtual architecture: hardware and software components with abstract communication channels
- Abstract interfaces are refined in hardware and software interfaces
- BFM level:
  - CPU
  - Interconnection network and adapters
Outline

1. MPSoc design flow

2. Software performance estimation methodology

3. Neural network-based performance estimation

4. Virtual prototype-based performance estimation

5. Case study: MPEG4 encoder
Software Performance Estimation
– Processor Evaluation

- SW performance estimation
  – Goal: fast processor evaluation under a given workload
  – Analytical-based, using neural networks
  – High-level, thus some inaccuracies are allowed
Software Performance Estimation
– Virtual Prototype

- SW performance estimation using a virtual prototype
  - Simulation-based
  - Detailed analysis of interaction between hardware and software components
Outline

1. MPSoC design flow
2. Performance estimation methodology
3. Neural network-based performance estimation
4. Virtual prototype-based performance estimation
5. Case study: MPEG4 encoder
Neural Network Performance Estimation

- Why neural networks?
  - Non-linear prediction – state-of-the-art processors
  - Very fast estimation

- Training phase
  - Set of benchmarks
  - Cycle-accurate simulation: MaxSim ARM9
  - Neural network training and simulation: Matlab

- Utilization phase
  - Dynamic instruction count: instruction-accurate simulator
Neural Network Performance Estimation

• Neural network configuration
  – Input: instruction count
  – Output: # of cycles
  – Input layer and output layer: linear transfer function
  – Hidden layer: tansig transfer function

• Back-propagation training algorithm
NN Estimation Results for ARM9

- Benchmark set composed by 32 applications and algorithms
  - Total of 41 samples (some benchmarks were executed with different inputs)
  - Different domains
    - Numerical
    - Sort and search algorithms
    - Data processing
    - Synthetic algorithms
  - 20 benchmarks used as training set

<table>
<thead>
<tr>
<th></th>
<th>Max underestim</th>
<th>Max overestim</th>
<th>Mean error</th>
<th>Std deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>All benchmarks</td>
<td>-35.59%</td>
<td>29.75%</td>
<td>9.05%</td>
<td>8.90%</td>
</tr>
<tr>
<td>Training set</td>
<td>-18.30%</td>
<td>29.75%</td>
<td>7.62%</td>
<td>7.97%</td>
</tr>
<tr>
<td>Test set</td>
<td>-35.59%</td>
<td>11.58%</td>
<td>10.08%</td>
<td>9.54%</td>
</tr>
</tbody>
</table>
NN Estimation Results for ARM9

- Benchmark set composed by 32 applications and algorithms
  - Total of 41 samples (some benchmarks were executed with different inputs)
  - Different domains
    - Numerical
    - Sort and search algorithms
    - Data processing
    - Synthetic algorithms
  - 20 benchmarks used as training set

<table>
<thead>
<tr>
<th></th>
<th>Max underestim</th>
<th>Max overestim</th>
<th>Mean error</th>
<th>Std deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>All benchmarks</td>
<td>-35.59%</td>
<td>29.75%</td>
<td>9.05%</td>
<td>8.90%</td>
</tr>
<tr>
<td>Training set</td>
<td>-18.30%</td>
<td>29.75%</td>
<td>7.62%</td>
<td>7.97%</td>
</tr>
<tr>
<td>Test set</td>
<td>-35.59%</td>
<td>11.58%</td>
<td>10.08%</td>
<td>9.54%</td>
</tr>
</tbody>
</table>
Outline

1. MPSoC design flow
2. Performance estimation methodology
3. Neural network-based performance estimation
4. Virtual prototype-based performance estimation
5. Case study: MPEG4 encoder
Virtual Prototype-based Performance Estimation

- Virtual prototype for the MaxSim environment, for performance evaluation, is generated from the architecture model.
- Processor model: cycle-accurate model provided in the MaxSim library.
- Other HW components are provided in SystemC.
- Hardware and software simulators run in synchronized way:
  - Detection of problems arising from communication between HW and SW components.
Virtual Prototype-based Performance Estimation

- Software analysis support by MaxSim
  - Timeline charts for evaluating application functions
  - Cache performance

- Analysis of hardware components
  - Custom profiling of user-defined components
Virtual Prototype – Custom Profiling

- Using the profiling interface, custom analysis is implemented in user-defined components
- Example: Analysis of transfers managed by the DMA component in the MPEG4 case study
Outline

1. MPSoC design flow
2. Performance estimation methodology
3. Neural network-based performance estimation
4. Virtual prototype-based performance estimation
5. Case study: MPEG4 encoder
MPEG4 Encoder

- Two software tasks
  - Encoder Task: core algorithms
  - VLC Task: compression algorithm
- Hardware components
  - DMA, INPUT, COMBINER
MPEG4 Encoder
Software Performance Estimation

• At specification level
  – NN estimator used to estimate the software performance
  – Choice of several ARM processor models
  – ARM9 has been selected by using estimation results

• At BFM level
  – Virtual prototype used for detailed SW performance estimation
  – Simulation model uses ARM MaxSim tool
    • CPU: cycle-accurate model
    • Hardware components: RTL models described in SystemC and instrumented with MaxSim profiling interface
Virtual Prototype: MaxSim model
MPEG4 Encoder

NN Estimation Errors

<table>
<thead>
<tr>
<th></th>
<th>NN estimation</th>
<th>VP: cycle-accurate</th>
<th>Estimation Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoder Task</td>
<td>122,910 cycles</td>
<td>137,000 cycles</td>
<td>10%</td>
</tr>
<tr>
<td>VLC Task</td>
<td>21,613 cycles</td>
<td>26,179 cycles</td>
<td>17%</td>
</tr>
</tbody>
</table>

• Estimation errors have two sources
  – Intrinsic error of the neural network method
  – Communication and OS overheads are neglected

• Communication overhead
  – The NN estimator was trained for a monoprocessor architecture
  – DMA provides point-to-point communication without contention
  – In architectures with shared resources (memories and buses), the contentions could result in a larger error of the NN estimator
NN Estimation Speed-up

- **NN network costs**
  - NN trained just once, in about 1.5 hours
  - NN utilization
    - Dynamic instruction count using instruction-accurate simulators (much faster than cycle-accurate simulators)
    - NN execution: very fast, just a matrix multiplication

- **Virtual prototype: simulation of cycle-accurate CPU + RTL hardware components**

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Cycle-accurate execution time</th>
<th>Estimation time</th>
<th>Speed-up</th>
<th>Estimation error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix sum</td>
<td>9 sec</td>
<td>0.39 sec</td>
<td>23</td>
<td>3%</td>
</tr>
<tr>
<td>LMS filter</td>
<td>12 sec</td>
<td>0.52 sec</td>
<td>23</td>
<td>1%</td>
</tr>
<tr>
<td>MPEG encoder</td>
<td>600 sec</td>
<td>17 sec</td>
<td>35</td>
<td>17%</td>
</tr>
</tbody>
</table>
Conclusions

- Integrated MPSoC design and estimation methodology
  - Performance data support the design decisions through the design flow
- Software performance estimation
  - At specification level: processor evaluation using a neural network estimator
    - High-level
    - Fast
  - Virtual prototype
    - After the HW and SW interface refinement
    - Cycle-accurate processor model with instrumented RTL hardware modules
    - Detailed performance analysis
- Offers an interesting trade-off between estimation speed and accuracy
- Case study: MPEG4 encoder
  - Neural network estimation errors up to 17%
Software Performance Estimation
in MPSoC Design

Thanks
Questions?

Contact
{marcio, flavio}@inf.ufrgs.br
{ahmed.jerraya, marius.bonaciu}@imag.fr