A Novel Performance Evaluation Methodology using Virtual Prototyping and Emulation

Woojoo Space Kim
Samsung Electronics Co., LTD.
Index

• Virtual Prototyping
• IP Hybrid Emulation Platform
• The Proposed Design Methodology
• Results
• Conclusion
Virtual Prototyping

- **Definition**
  - Virtual platform (VDK: virtual SW development kit) using SystemC based IP models
  - Overall flow to develop and validate SW/HW using virtual platform
  - Same design flow with RTL, Different languages and methodologies
Virtual Prototyping

- ESL environment – SystemC/TLM model based virtual platform environment
  - LT (Loosely Timed) based TLM model: for high speed simulation (up to 300MHz)
  - Full SoC level virtual platform: ONE branch SW development – SW stability
  - At early HW design stage, SW development started

- Economical Efficiency
  - Replacement of silicon test board: In Automotive, a huge number of SW developers
  - Easy board modification for bug fix and enhancement: only code modification
Virtual Prototyping

• SW Architecture exploration / Early stage SW bring-up
  – Full SoC Level virtual prototyping for SW architecture exploration
  – Saving development cost by replacing silicon board
  – From specification stage, virtual prototyping started

• Simulation Performance
  – Only 10 minutes are required to boot-up Android Platform
  – Enough simulation performance for SW testing more than dozens of times a day

• Advantages for SW development
  – Enough time can be secured for SW development and optimization
  – Since a separate SW branch for virtual prototyping is not needed, efficiency of SW development can be secured
SoC Hybrid Emulation Platform

**Definition**

- Co-emulation platform using partial VP and emulator
- Two domains: VP (CPU paths) and Emulator (other IPs)
- Improvement of SW operation speed using VP (15 times faster than pure emulation)
IP Hybrid Emulation Platform

• Definition
  – Focusing: IP HW/SW development
  – Reducing platform building time including IP SystemC/Emulation modeling by omitting modeling targets
  – OS aware IP SW/HW development from IP design stage

• Required Technologies
  – Transactor: interfacing between VP domain and Emulation domain
Proposed Design Methodology (1)

• **Main Idea**
  – Early SW development for IP Hybrid Emulation Platform using VP
  – Pull-in starting time → Securing sufficient time for IP development/verification

• **Key Points**
  – Design flow enhancement
  – Incremental approach
  – Smart transactor design
  – OS/SW aware HW/SW co-verification using a real Benchmark application
Proposed Design Methodology (2)

• Design Flow
  – Incremental VP development
  – Incremental SW development
  – RTL design & Emulation modeling
  – Performance aware IP verification
  – IP enhancement via what-if analysis
Proposed Design Methodology (3)

- **Incremental Approach**
  - For parallel HW/SW development (= design pipelining)
  1. Basic CPU path platform for **OS enabling**
  2. SW (device driver, benchmark application porting) development on a target IP model
  3. Replacing target IP model to real IP design (RTL, emulation model)
Proposed Design Methodology (4)

• **Transactor**
  – Transactor enhancement: APB, AXI4, ACE-Lite support
  – Performance enhancement for protocol conversion

• **Benchmark Application**
  – GFXBenchmark 3.1.0:
    - **Manhattan 1080p & 3000ms**
  – On Android O Platform
  – **Virtual I/O**: APK installation via a virtual Ethernet interface
Experimental Results

- **Runtime Analysis**: Run time as SW stack (sec)
  - Slower than pure virtual prototyping due to co-emulation (x2.75, 14.8 min vs 40.7 min)
  - Enough speed for debugging: 11.8 times run per a 8 hours (normal working time)

<table>
<thead>
<tr>
<th></th>
<th>Tool Invoking</th>
<th>Kernel boot-up</th>
<th>Android Boot-up</th>
<th>APK installation</th>
<th>7 frame run</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pure VP</td>
<td>47</td>
<td>52</td>
<td>506</td>
<td>147</td>
<td>134</td>
<td>886</td>
</tr>
<tr>
<td>Proposed Platform</td>
<td>314</td>
<td>63</td>
<td>948</td>
<td>165</td>
<td>949</td>
<td>2439</td>
</tr>
<tr>
<td>Delta (%)</td>
<td>568.1</td>
<td>21.2</td>
<td>87.4</td>
<td>12.2</td>
<td>608.2</td>
<td>175.3</td>
</tr>
</tbody>
</table>

- **Runtime Comparison**: comparison as Model (KHz)

<table>
<thead>
<tr>
<th></th>
<th>Cycle Model [10]</th>
<th>Cycle Accurate SystemC Model</th>
<th>Emulation Model</th>
<th>loosely timed Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation Speed</td>
<td>1.5 KHz</td>
<td>895 KHz</td>
<td>2,542 KHz</td>
<td>75,020 KHz</td>
</tr>
</tbody>
</table>
Conclusion

• **Virtual Prototyping**
  – *Early SW development:* Early SW Securing
  – Virtual I/O: Providing *the same user interface* as the actual set using Virtual I/O
    *(Fast APK installation after OS boot-up using virtual Ethernet interface)*

• **IP Hybrid Emulation Platform**: accurate function/performance validation
  – *Function verification* on RTL emulation model for a target IP
  – *Accurate performance measurement* on RTL emulation model for a target IP

• **The proposed design flow**
  – *Start-time pull-in using VP*
  – *Enabling architecture exploration* via what-if analysis by securing enough dev. time
Questions