Pre-Silicon USB Driver Development on SystemC Based Virtual Platforms

Frank Schirrmieister
Director, Product Marketing
Synopsys, Solutions Group
Version 1.1

February 24th 2009
North America SystemC User Group Meeting
WHY VIRTUAL PLATFORMS?
“Phone differentiation used to be about radios and antennas and things like that. We think, going forward, the phone of the future will be differentiated by software.”

Steve Jobs
CEO, Apple
August 11, 2008
http://online.wsj.com/article/SB121842341491928977.html
There Really Are Only Two Issues …

Software starts late

Bugs are expensive

Software has to wait for prototype => Semi cannot sell silicon …
Start software as early as possible …
… and make its development as productive as possible

Traditional Design Flow

End to End Prototyping Design Flow
What is a Virtual Platform?

- Fully functional software model of complete systems
  - SoC, board, I/O, user interface
- Executes unmodified production code
  - Drivers, OS, and applications
- Runs close to real time
  - Boots OS in seconds
- Highest debugging efficiency through full system visibility and control
  - Supports multi-core SoCs debug
Virtual Platforms
It’s Like Hardware – Before RTL!

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Early Availability</td>
<td>Available before chips come back from the fab and before boards have been built and debugged</td>
</tr>
<tr>
<td>Enhanced Debugging</td>
<td>Full visibility and control of multi-core platform with non intrusive access to all components</td>
</tr>
<tr>
<td>Easy to Deploy</td>
<td>No physical boards - minimal user ramp up time and logistical efforts to distribute and maintain</td>
</tr>
</tbody>
</table>
THE IMPACT OF SYSTEMC
The Impact of SystemC TLM-2.0
A Historic Moment – like Verilog in the 90’s!

<table>
<thead>
<tr>
<th>Hardware Description Languages</th>
<th>Virtual Platforms</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1980’s</strong></td>
<td><strong>1990’s</strong></td>
</tr>
<tr>
<td><strong>Age of Proprietary HDLs</strong></td>
<td><strong>HDL Standardization</strong></td>
</tr>
<tr>
<td>Verilog</td>
<td>Verilog</td>
</tr>
<tr>
<td>VHDL</td>
<td>VHDL</td>
</tr>
<tr>
<td>HiLo</td>
<td>HiLo</td>
</tr>
<tr>
<td>DABL</td>
<td>DABL</td>
</tr>
<tr>
<td>LASAR</td>
<td>LASAR</td>
</tr>
<tr>
<td>Aida</td>
<td>Aida</td>
</tr>
<tr>
<td>M (Lsim)</td>
<td>M (Lsim)</td>
</tr>
<tr>
<td>QuickSim</td>
<td>QuickSim</td>
</tr>
<tr>
<td>UDL / I</td>
<td>UDL / I</td>
</tr>
<tr>
<td>N dot</td>
<td>N dot</td>
</tr>
<tr>
<td>ISP</td>
<td>ISP</td>
</tr>
</tbody>
</table>

Babel

What it means for you:
1. Model interoperability
2. Simulation commoditization
3. It’s all about the models!
The Impact of SystemC TLM-2.0
Enabling Interoperability and Scalability

Previously proprietary (backdoor) APIs & new additions have now been standardized:

- **(DMI) Direct Memory Interface**
  - Direct backdoor access into memory
  - Allows un-inhibited ISS execution

- **LT (Loosely Timed) modeling**
  - Timing declaration
  - Allows speed/accuracy trade-offs

- **Temporal Decoupling**
  - Only synchronize when necessary
  - Allows multicore speedup

**Diagram Notes:**
- AV: Application View
- LT: SystemC Loosely Timed
- AT: SystemC Approximately Timed
- CA: Cycle Accurate

**Axes:**
- Performance (IPS)
- % Accuracy
USB DRIVER DEVELOPMENT
CASE STUDY
The Project

- Developing & validating software at pre-silicon stage is crucial to reduce time-to-market
  1. Develop software concurrently with (IP) hardware
  2. Reduce hardware/software bring-up

- Case study: prove Synopsys Virtual Platform technology on DW HS USB OTG controller

- Objectives:
  - Phase 1:
    • Develop & test Linux device driver for descriptor-based DMA HW enhancement in the OTG IP core
    • Prove developed device driver can be run “as is” on FPGA system
  - Phase 2:
    • Co-verification setup, verify RTL within virtual platform
HS USB On-The-Go Controller & PHY
*DesignWare IIP / Cores*

- **OTG Controller**
  - Dual-Role Device (DRD)
  - Flexible parameters for low & high-latency sys. integration
  - Buffer & descriptor pre-fetching maximizes host throughput
  - Firmware-selectable endpoint configurations
  - AHB integration interface
  - UTMI+ Level 3 PHY I/F
  - Verilog Source RTL
  - Linux ref. device drivers

- **PHY**
  - Complete mixed-signal physical layer for single-chip USB 2.0 OTG applications
  - 8-/16-bit ctrl. Interface (UTMI+ Level 3)
HS USB On-The-Go Controller & PHY

**TLM Model**

**Digital Controller**
- Loosely Timed (LT) model
  - Register & functionally accurate
  - AHB TLM interfaces (master & slave)
  - UTMI+ interface not modeled
- Parameters to support IP configuration options
- “USB Real-World I/O” (Host + Client)
  - Allows virtual platform to act as “physical” USB host / client to host OS
- SystemC TLM-2.0 API

**PHY**
- Only control registers modeled, no data path
HS USB On-The-Go Controller & PHY
Using TLM Model in Virtual Platform
HS USB OTG Case Study

• **IP hardware feature addition (DW USB OTG IP core)**
  - Hardware support was added for descriptor-based DMA transfers
  - Device mode only
  - (Linux ref.) software device drivers needed to be extended to support new features

• *A virtual platform was used for developing & testing these driver updates*
HS USB OTG Case Study
*Past Design Flow*

- Long, sequential development flow
- HW/SW meet late = risk!
- Silicon not a productive SW development vehicle (visibility, control)!
HS USB OTG Case Study
Improved Design Flow

Virtual Platform Approach

Time-to-Market Gain
HS USB OTG Case Study

Results

• Virtual platform approach was used for developing & testing USB OTG **Linux** device drivers
  – SW engineer extended existing drivers based on published DW IP specification
    • Added support for descriptor based DMA enhancement in the DW OTG IP core
  – At the same time, VP modeling group enhanced DW core model to support the new feature
  – SW engineer completely debugged & tested the basic driver functionality early on Virtual Platform
  – Software ‘bringup’ in a **shorter** time after the HW (FPGA) was available
HS USB OTG Case Study

Results

• Summarized results
  – Driver availability: 4 weeks before FPGA
  – Reduction in SW bring-up: 4 weeks savings

• Conclusions – Results promising!
  – More significant productivity could be demonstrated for developing multiple pieces of SW on a more complex sub-system
  – SW driver was not developed from scratch - developing a complete new piece of SW would produce more significant time savings
USB VERIFICATION
Virtual Platform for RTL Verification

Key Values & Usage

- **Software Validation and Architectural Verification**
  - Validate HW/SW integration on actual hardware (RTL)

- **System Validation**
  - Scale verification to system context
  - Software becomes part of the verification setup
  - Verification confidence increases with “real” system scenarios

- **RTL Simulation Speed-up**
  - Maintain TLM level where possible
HS USB OTG Case Study

RTL Verification Setup

**Virtual Platform Approach**

- HW Spec.
- Hardware Design (IP RTL)
- FPGA Prototype Design
- Virtual Platform Design
- OS & Device Driver
- Apps
- Final Validation

**Traditional approach**

- HW Spec.
- Hardware Design (IP RTL)
- FPGA Prototype Design
- Virtual Platform Design
- OS & Device Driver
- Apps
- Final Validation

**HW/SW co-verification**
Case Study: USB OTG, RTL Validation
Case Study: USB OTG, RTL Validation

TLM

ARM920T CPU

AHB Master

AHB Slave

USB OTG Ctrl (Host)

USB OTG Ctrl (Device)

Bridge (Loopback)

Transaction Level Interface (TLI)

DW AMBA VIP (master & slave)

RTL
Case Study: Driver Development and RTL Validation

1. Build Virtual platform for both USB OTG host and device
2. Use Virtual platform to run real driver software
3. Replace TLM model with RTL model
   - Add TLM-to-RTL transactors
4. Rerun real driver software, verify RTL in the system context
HS USB OTG Case Study

Lessons Learned

• Virtual platforms accelerate software development for hardware IP
  – Early development, prior to silicon/prototype availability
  – Reduced bring-up time on physical prototype
  – Binary compatibility allows to develop production software (drivers)
  – Enhanced visibility & controllability of VP is greatly appreciated by SW developers, even when physical prototype is available
• Solid (architecture) specification process is required
• By modeling full prototype board as a virtual platform, transition from “virtual” to “physical” is seamless
  – USB HS OTG: Samsung IPMate-MB-S3C2410X board + Virtex FPGA
• Reusing virtual platforms for hardware verification accelerates system verification
  – Provides system testbench, early
  – Enables software driven verification
  – Takes advantage of early software availability
The Complete Virtual Platform Solution, SystemC-based

Models: DesignWare System-Level Library

Building blocks for Virtual Platforms:
- Transaction-level models (TLM),
- written in SystemC™
- TLM-2.0 support
- tool independent

Virtual Platform Creation and Execution: Innovator

Unmatched Expertise Predictable Time-to-Model!

- Over 60 Virtual Platforms and 2500+ models
- Recognized experts for SystemC TLM-2.0

It’s Like Hardware – Before RTL!

Early Availability Efficient Debugging Easy to Deploy