Top Semiconductor Design Flow Challenges
System Level Integration

Frank Schirrmeister
EDPS
Monterey
April 17th 2014
What’s in a System?

Software

- Applications
- Middleware
- Operating Systems (OS)
- Drivers
- Firmware / HAL
- Communications L3
- Communications L2
- Communications L1
- RTOS
- Drivers
- Firmware / HAL

System on Chip (SOC)

ARM CPU Subsystem
- Cache Coherent Fabric
- L2 cache

SoC Interconnect Fabric
- 3D GFX
- DSP A/V
- Apps Accel
- Bare Metal Software
- Bare Metal Software
- Bare Metal Software

Modem

System on Printed Circuit Board (PCB)

Memory Card

Bare Metal

Modem Comms

Applications

- Processor

Application Processor

Hardware
- NAND
- LPDDR
- Wi-Fi
- Cellular Modem
- Bluetooth
- SD 3.0
- SD 4.0
- UFS
- LPDDR 2
- eMMC 4.5
- UFS
- LPDDR 3
- SD 3.0
- SD 4.0

Other hardware
- SLIMbus
- DSI
- CSI2
- CSI3
- SLIMbus
- Motion Sensors
- cJTAG
- GBT
- SPMI
- Power Control
- Multimedia Processor
- Camera Interface
- Graphics Interface
- Audio Interface
- Display Driver
- Touch Screen Controller
- Memory Card
- FPGAs
- Application Specific Components

Software

- Bare Metal Software
- DSP Software
- Apps Accel
- Bare Metal Software

RTOS

Communications L2

Drivers

Firmware / HAL

Operating Systems (OS)

Drivers

Middleware

Applications

Bare Metal

Modem Comms

Application Processor
Just how complex and expensive will it get?
Cost of Developing New Products

Source: IBS, February 2014
Just how complex and expensive will it get?
Growth of # of SIP Blocks per Design

Source: Semico, October 2013
Integrating at the System & Software Level

**Multi-core early SW bring-up and integration on 64-bit**

**Developing environments for HW/SW integration & use case verification on sim/emulation platforms**

**Bare-metal SW use case testing to verify multi-core cache & IO coherency, concurrency, Power Shut Off, etc…**

**Verification of IPs on AMBA interconnect w/adherence to ACE protocol**

**Debugging of complex multi-core SoC SW scenarios on RTL sim/emulation platforms**

**Characterizing & analyzing SoC performance & efficiently debugging issues**

**How do I represent the system on chip environment?**
Software Based Testing and Benchmarking

Validate HW/SW with OS-based Tests
Bring-Up and Validate OS Stack with SoC
Bring-Up OSs (stock)
Verify HW/FW
Bring-Up FW on SoC
Develop SW Modules

HW/Firmware Bare-Metal Tests
Registers, memory, device drivers, sub-system

Firmware Bring-up
Boot code, Firmware

SW Development
Boot code, device drivers, Kernel, SoC Drivers

Quality Assurance Tests
SoC Stress Tests
Graphics Tests
SoC Basic Tests

Graphics
Framework
Libraries
HAL
Drivers
Test Linux

Android
Linux

Graphics
Kernel Mode
Resources
Memory
Power
Win RT

Color Key
Production SW
Test / Verification SW
Example System Integration Scenarios
Hardware Interfaces & Virtualization are complementary

<table>
<thead>
<tr>
<th>SWEET SPOTS</th>
<th>ICE</th>
<th>Embedded</th>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW</td>
<td>GFX drivers</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GPU compute</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>System power management</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Diagnostics</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Operating systems</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BIOS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW</td>
<td>Firmware</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I/O controllers</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DFT/DFD logic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Full reset</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

With Hardware Interfaces

With Virtualization

- Highest performance
- Save/Restore with real CPU
- Early bring-up
- Highest model accuracy
Example System Integration Scenarios
Hardware Interfaces & Virtualization are complementary

**Performance**

**Virtualization**
1. Migrate VIP-based verification to SoC & system-level on an acceleration platform
2. Validate HW/SW behavior early in development cycle by linking RTL with virtual system
3. Provide an acceleratable soft-model of a peripheral on the bus – such as a disk-drive

**Hardware Interfaces**
1. Validate HW/SW system interactively with live traffic including long sequences with multiple ports and increased bandwidth
2. Interface to testers & target systems for system level validation
3. Stress test with real applications & full protocol stacks

**SpeedBridges**

**Verification IP**
1. Exhaustive verification of standard interfaces
2. Facilitate constrained-random simulation with VIP-supplied stimulus
3. At SoC level, check flow of commands and data across the chip

**Simulation VIP**

**Block / IP Verification**

**Sub-system / SoC Verification**

**SoC level HW/FW Integration**

**Full System Validation w. App SW**

**Field Prototype**
• More complex designs
  – more at <90nm
  – overall less starts
• An average of 180 IP Blocks
• More than 80% re-use
• More than 60% of effort in software
• Multi-core
• Complex interconnect with cache coherency
• Software distributed across cores
• Low power issues
• Application specific issues
• High analog mixed signal content
• Hardware and virtualized system representations