Full Chip Verification of Multi-Core SoC for Mobile Application Engine

Badri Seshadri and Sandeep Agrawal
Agenda

- Introduction to Mobile Engine.
- Full Chip Verification Methodology.
- Verification Speed Methods.
- Verification Environment.
- Conclusions
Agenda

- Introduction to Mobile Engine.
- Full Chip Verification Methodology.
- Verification Speed Methods.
- Verification Environment.
- Conclusions
Mobile Application Engine

Application Processor

- Video Engine: Programmable High Performance
- Audio Engine: Audio DSP Sub-System
- CPU Sub-System
- Security Engine: security tasks
- Graphics Engine: HW assisted 2D/3D engine based
High Level Functional Needs

- **Video**
  - Video playback / streaming
  - TV on Mobile: DVB-H, DVB-T
  - Video Enhancement Algorithms
  - 3G Video Telephony

- **Audio**
  - for music, voice, FM capture and more
  - Codecs package
  - OpenMax Compatibility.

- **Base Band**
  - GSM/3G/SCDMA/GPRS/EDGE & UMTS

- **Security**
  - DRM client
  - VPN client
  - PKI processing

- **Connectivity**
  - USB (OTG/Wireless), Bluetooth, WLAN, NFC
Application Engine :- Architecture
The Verification Dilemma

- Complexity of designs increasing
- Time-to-Market remains the same

The Verification See-Saw

1. Decreased likelihood of bugs on silicon
2. Increase Verification Effort
3. Increase Time-to-Market

1. Decreased Verification Effort
2. Decreased Time-to-Market
3. Increased likelihood of bugs on silicon
Agenda

- Introduction to Mobile Engine.
- Full Chip Verification Methodology.
- Verification Speed Methods.
- Verification Environment.
- Conclusions
vSW – Concept (IP provider view)
vSW – Concept (sw arch)

Chip context

Test Application
(calls IP integration verification API methods)

UART
IP integration verification

TIMER
IP integration verification

IPx
IP Integration verification

Print and file IO
Infra abstraction

UART hardware
TIMER hardware
IPx hardware

DMA

interrupt controller
clock and reset controller

Interconnect
Generated by Tool
(E.g. Nx-Builder)

Generated by Tool
(E.g. Nx-Builder)

sw

hw

Generated by Tool
(E.g. Nx-Builder)

Interconnect
Generated by Tool
(E.g. Nx-Builder)
Agenda

- Introduction to Mobile Engine.
- Full Chip Verification Methodology.
- Verification Speed Methods.
- Verification Environment.
- Conclusions
Problems in Full System Verification

- **Simulation time** is a **bottleneck**
  - avoid **redundant** verification!
- IP-level Verification suite for a CPU can take days to run.
- Don’t need the RTL of a CPU to verify most IPs?
Trick: Move vSW Out Of The Simulator

- **Method:**
  - Replace the CPU with a Transactor
  - Decide which test cases use CPU, and which use Transactor.

- Next step – execute the software on the host CPU instead of the simulated CPU.

- Memory is “shared” between Software processes and Simulator.

- vSW can access memory directly, without using the simulator.

- Speed gain are 10X

- Can use Linux based software debuggers – gdb, ddd, eclipse, etc.
Important: Combine Code Bases

- Verification Software (vSW) is written in C.
- Can target both the simulated RTL of a CPU and the Transactor with the same code.
- Coding in Assembly on average, takes more time.
- Prefer “function-level” C instead of “register-level” C.

```c
// register focused code:
if (myIP.regA == 0x1) {
    myIP.regB = 0x2;
}

// function focused code:
if (isReadMode()) {
    startupIP();
}
```
Tricks : Continue

- Replace with Dummy Blocks when not excised.
- Use case clock setting for initialization of Chip.
- Initialization of Interrupts/Memory/Peripherals can be TCL based setting in-directly driven by Application.
Agenda

- Introduction to Mobile Engine.
- Full Chip Verification Methodology.
- Verification Speed Methods.
- Verification Environment.
- Conclusions
Verification Env Support

- Application Debug support.
- Infrastructure to host file Interface.
- Smart Application Control interface to Test-bench.
- Testing the External Interface through application.
- Co-Verification Simulation setup.
Printf Support

```c
//
int maxunitid;
int id;
comps_struct VpbBridgeStruct;
if (vhGetMaxUnitId(VH_IP_ID)==0) {
    vhPrintf("No VPB Bridge found, \n");
} else {
    maxunitid=vhGetMaxUnitId(VH_IP_ID);
    for ( id=0; id<maxunitid ; id++) {
    }
```
int maxunitid;
int id;
comps_struct VpbBridgeStruct;
if (vhGetMaxUnitId(VH_IP_ID)==0) {
    PathSet(VH_IP_ID,
        unitid,
        VH_IP_ID,
        unitid,1);
} else {
    maxunitid=vhGetMaxUnitId(VH_IP_ID);
    for ( id=0; id<maxunitid ; id++) {

}
Agenda

- Introduction to Mobile Engine.
- Full Chip Verification Methodology.
- Verification Speed Methods.
- Verification Environment.
- Conclusions
Conclusion and Inputs

- **Conclusion**
  - Ip level Interconnect/Interoperability software easy the Full chip verification.
  - Verification speed tricks.
  - Smart Verification Env.
  - TLM Based Test stimulus way forward for Complex Verification.

- **Inputs**
  - Smart ways to identify “X” Propogation in Netlist.
  - SystemC save and restore options to be supported in simulator
  - Connection to “Gdb” support for open cores
  - **PDES** kind of Simulation kernel which can improve the speed.