Maximize Virtual Platform ROI by Efficient Model Development

Presented by:

Sam Tennent
Manager, R&D
Synopsys NE,
Livingston, Scotland.
sam.tennent@synopsys.com
Legal Reminder

This presentation contains roadmap information

“Information contained in this presentation reflects Synopsys plans as of the date of this presentation. Such plans are subject to completion and are subject to change. Products may be offered and purchased only pursuant to an authorized quote and purchase order. Synopsys is not obligated to develop the software with the features and functionality discussed in the materials.”
## Agenda

<table>
<thead>
<tr>
<th>Time</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>5min</td>
<td>HW / SW integration – the problem &amp; how Virtual Platforms can help.</td>
</tr>
<tr>
<td>10min</td>
<td>Tools for efficient VP development</td>
</tr>
<tr>
<td>10min</td>
<td>Models for efficient VP development</td>
</tr>
<tr>
<td>10min</td>
<td>Services for efficient VP development</td>
</tr>
<tr>
<td>5min</td>
<td>Summary</td>
</tr>
<tr>
<td>5min</td>
<td>Q&amp;A</td>
</tr>
</tbody>
</table>
Challenge - Making Excellent Products

Excellent phone

...but a bit buggy
… Not Surprising, Given HW/SW Content

**HW Platform**
- Multi-Core (DSP & RISC)
- Security Accelerator
- Multimedia Accelerators
- Video & Audio Sub-System

**SW Stack**
- Embedded Wireless Communications Software
  EDGE, CDMA, UMTS, WCDMA, GPS, Bluetooth, WiFi
- Embedded Camera and Video Codecs
- Embedded Voice and Audio Codecs
- 3D Gaming Embedded SW and Middleware
- ISV Applications
- Multimedia Messaging Service (MMS)
- Java Run-Time
- Inter-Processor Communication Middleware
- Hardware Peripheral Drivers
- High-Level OS

Multi-Band Cell Phone
Full PDA
5MP Camera
MP3 Player
Camcorder
Video Player
Game Consol
Web Browser
Email Terminal
GPS & Navigation Connectivity
(USB, WiFi, BT)
HW/SW Integration & Whole-Product Validation is a Growing Challenge

Multi-Band Cell Phone
Full PDA
5MP Camera
MP3 Player
Camcorder
Video Player
Game Consol
Web Browser
Email Terminal
GPS & Navigation
Connectivity
(USB, WiFi, BT)
Solution - Virtual Platforms

- A Virtual Platform is a fully functional software model of complete systems
  - SoC, board, I/O, user interface
- Executes unmodified production code
  - Drivers, OS, and applications
- Runs at up to 50 MIPS
  - Boots OS in seconds
- Provides high system visibility and control
  - Supports multi-core SoCs debug
Elements of A Virtual Platform

- Virtual I/O
- User Interface Emulation
- High-speed C/C++/SystemC Models
- Fast Instruction-Accurate Simulator
- Transaction-level Interfaces
- Graphical Peripheral Models
How Can Virtual Platforms Help?
Improved SW Development Flow w/ Virtual Platforms

Virtual Platform Development

- Architecture Design
- Chip Design
- Prototype Silicon Manufacturing
- Architecture & Low-Power Optimization
- System Validation
- Development Board Design
- Board Mounting

Software Development & HW/SW integration (Using Virtual Platforms)

First-day Integration

9 to 12 Months Earlier
BUT – Early Availability is Key

• Need Virtual Platform available as early as possible in design cycle

• Requires efficient & repeatable platform creation process

• Process must deliver
  • on time
  • high performance simulation
    ▪ high quality
    ▪ reasonable cost
Synopsys System-Level Solutions
All Of The Ingredients for Virtual Platforms

DesignWare System Level Library

- DesignWare® System-Level Library
  - SystemC™ Transaction Level Models
  - Processors
  - DesignWare Cores
  - DesignWare AMBA Components
  - Pre-Assembled Platforms

Innovator

- Environment for developing, running & debugging virtual platforms

Services

- Expert services for model creation, virtual platform assembly & customization

Virtual Platforms

Abstract, high-performance models of SoC blocks
Synopsys – The Virtual Platform Leader

• Over 50 engineering years of Virtual platform and TLM modeling experience
  ▪ Unmatched on-time, high-quality modeling services track record

• Over 50 platforms deployed
  ▪ Fastest and most complete virtual platforms
  ▪ Incl. TI OMAP1®, OMAP2®, OMAP3®, Freescale i.MX, MXC, Intel® XScale™
  ▪ In the hands of SW developers today
Efficient Process Based on “3 Pillars”

- **Tools** - Comprehensive tool suite
  - **Innovator** – Virtual Platform creation and execution tool.
  - **Component Creator** – Model creation tool.

- **Models** - Synopsys provides complete platforms and IP models
  - Over 30 commercial platforms.
  - Over 2,000 TLM models in library.

- **Services**
  - Expert platform creation & support team.
  - World-wide training & technology transfer.
ARM Developers’ Conference 2007

Tool Support for Efficient Virtual Platform Creation

Virtual Platform
Tools
Models
Services
VP Creation
Innovator - Rapid System Prototyping IDE

Simulation Infrastructure
- Supports multi-core designs
- Open & flexible framework

Web Updates
- Version Control

System Specification Languages
- Magic-C
- SystemC

Graphical Design Editor
- Model Creation
- Model Connectivity
- Model Parameterization

GUI Cockpit Editor
- Create User Interface.

3rd party Debugger Integration
- Optimized, lite Virtual Debug Interface (VDI) protocol
- Provides easy customization to RDI, MDI, GDI, …
- VDI Integration Kit available

Run-time Control
- Control Platform Execution
- Built-in Debug Facilities

© 2007 Synopsys, Inc. (15)
Innovator - Features

• **Single tool supports platform execution and design creation**
  - One IDE to learn. Easy to migrate from VP usage to VP design
• **Supports multiple design entry methods**
  - Can choose which approach suits best
• **Graphical design entry**
  - Faster development of models
• **Support for standard SW development interfaces & tools**
  - No changes in SW development process and no need for new debug tools
Innovator - Infrastructure

• **Supports Quick Platform creation**
  - Allow basic platform creation within days
  - DesignWare System-Level Library provides common generic infrastructure parts – MemMux, I/O Config RAM/ROM

• **Supports Multi-Core Designs.**
  - In-build scheduler algorithms

• **Supports High Performance Simulation**
  - Temporal de-coupling allows masters to “run-ahead” and minimizes context switching in simulation

• **Support for standard TLM interfaces**
  - Allow development of “stub” components
  - Supports early platform availability
MAGIC-C Design Entry

- **Graphical Design Entry**, elements of SDL and C/C++
  - Finite State Machine (FSM) concept.
  - Driven by signal exchange.

- **Benefits**: graphical features
  - Efficient model design using rapid-prototyping IDE
  - Visibility of design
  - Extra debug capabilities – breakpoints in models – easier to debug models
  - Easy for SW developers to understand model flow
MAGIC-C Constructs

- Concise, easy to learn language
- Based on C/C++ so familiar to S/W developers
- Provides encapsulation:
  - Efficient design through scoping
  - Supports design re-use at several levels
C++ Modeling Style

- C++ scheme is based on “co-operating co-routines”
- Components use RPC to access each other's methods through standard interfaces.
- Device functionality modeled in C++.
- Tightly coupled with tool infrastructure.
Languages - Magic-C vs C++

- Gives designer flexibility to choose best approach.
- Trade-off performance with design visibility & design effort.

**Magic-C**
- Finite State Machine
- Communication: Signals (events)
- Speed: Very fast
- Visibility: Graphical / C specification
- Typical use: Peripherals e.g. UARTs, SPI, INTC, RTC …

**C++**
- RPC-based function calls
- Communication C++ interfaces
- Speed: Extremely fast
- Visibility: Black-box
- Typical use: CPU, Memory models, DMA Controller, Security Devices
SystemC Support

• Native SystemC development in Innovator
  ▪ In-built editor

• Import of existing SystemC models

• SystemC code generation for Magic-C design entry
  ▪ Both for new Magic-C descriptions and legacy models

• SystemC code generation from VRE
  ▪ Conversion of existing VRE models

• Retain advantages of existing design methodologies while producing SystemC compliant models
Component Creator

• **Tool for quick creation of Magic-C components**
  - Grew out of “home-grow” development tool
• **Captures register information**
  - Graphical entry, Windows standard IDE
• **Produces design files**
  - Generates all design files required for Innovator
• **Produces component documentation**
  - HTML based document can be viewed within Innovator
• **Encapsulates “best-practice” for Magic-C design**
  - Most efficient code practice encapsulated in tool
• **Generates standard framework for test development**
Component Creator – Register Details

- Graphical entry of all bit-fields, their properties and default values
- Supports all bit field types – r/w, r/o, w1c, w1s etc.
Component Creator – Model Parameters

- Supports model parameterisation
Component Creator – Standard Functionality

• Support addition of functionality – Clocks, Interrupts, DMA support
Component Creator – Signal Addition

• Support addition of standard TLM signals
Component Creator – Documentation

- Supports creation of HTML documentation
Test Framework

• **Standard framework**
  - Standard GUI for test invocation
  - Generic “start-up” code – vector tables, stack, heap initialisation etc.
  - Standard handlers for interrupts, exceptions etc.

• **Common approach to test creation**
  - Encourages “good practice” in testing

• **Layered approach – driver plus tests**
  - Reference driver available to code developer
  - Test code visible

• **In place at initial platform creation**
  - Framework is there at day one with placeholders for tests
  - Real code which executes and validates initial platform

• **Requires only generic UART and VT100 models**
  - Can run with minimal peripheral support

• **Uses GNU tools**
  - Sources supplied – User can extend or re-use the test code
Test Framework

- Used for Regression
  - Script support for execution of test programs.
Model Support for Efficient Virtual Platform Creation
Modeling Choices – PV, PVt, CA

(PV) Programmers View - Functional
• Loosely timed, functional
• No clock in un-timed system; simulation kernel will execute one potential ordering of concurrent processes

(PVT) Programmer’s View With Timing – Cycle-Approximate
• TLM buses with timing
• Incorporates timing as functional labels rather than executing waveform

(CA) Cycle Accurate
• Accurate at cycle boundaries
• (Clock-driven simulation)
PV First Approach

• Can start from functional descriptions
  ▪ No need for implementation details

• Produces functionally accurate model
  ▪ Early availability of “Executable Specification”

• Earliest availability / fastest performance
  ▪ Less design effort to produce than CA or PVt approaches

• Can incrementally add timing as required
  ▪ Existing and validated models can be extended to PVt as required

• Gets critical software development task moving
  ▪ PV models have all the details required for S/W development
  ▪ No unnecessary detail to slow down execution
  ▪ When timing added real software is available to run
PVt – Extensions to TLM interfaces.

• Cycle Approximate TLM Approach
  ▪ Cycle counting approach. Avoids overhead of clocking simulation

• PV Models are extended to add timing
  ▪ TLM interfaces have placeholders for timing information
  ▪ Timing information passed in TLM interfaces and accumulated by bus masters

• TLM Transactions have contention added
  ▪ Contention for resources can now be modeled.

• Temporal decoupling
  ▪ Approach still allows masters to “run ahead” and exploit gains from context switching avoidance
Model Re-use – Parameterization

• Model Parameterization used to enhance re-use
  ▪ Parameters allow model functionality to be changed without re-building the model
  ▪ Parameters can be used to switch on extra debug facilities such as transaction tracing
  ▪ Parameterization supported by Innovator. Easily changed through browser
  ▪ Also supports higher level parameterization
  ▪ Can have several “configurations” which store parameter values for each model in the Virtual Platform
Model Re-use – Encapsulation

• **Low level re-use**
  - Models such as peripherals, memories, etc can be re-used
  - Also possible to create libraries of generic code which can be re-used across several similar models – e.g. UARTs, Timers, DMA Controllers

• **Intermediate re-use**
  - Complex peripherals
  - split up and designed as separate re-usable blocks, but can also be re-used at the component level

• **High level re-use**
  - Entire SoC’s can be re-used across different Virtual Platforms making it easy to create platform variants
Re-use – Virtual I/O

- Allows Virtual Platforms to use Host PC facilities – examples:
  - LAN controllers can have real-world connectivity
  - USB controller models can be configured to act as
    - host to devices plugged into PC’s USB connectors or
    - as a USB device plugged into the Host PC
Re-use – Virtual I/O

• **Complex development – costly and difficult**
  - Designed to be re-usable across different models / platforms
  - Example – VHub utility
    - Standard API
    - Hooks into TCP/IP stack via protocol driver
Re-use – Virtual I/O

• Debugger Connection

- Innovator supports connection of 3rd party debugger tools.
- Lauterbach, CodeWarrior, GDB, Arm AXD.
- Uses VDI - Supports connection to any RDI compliant debugger
Re-use – Model Library

DesignWare® System-Level Library

Processor Models
- ARM7TDMI
- ARM920T
- ARM926EJ-S
- ARM946E-S
- ARM1136JFS
- ARM1176JZF-S

DesignWare Cores
- USB 2.0 High Speed OTG
- SATA AHCI

DesignWare AMBA Components
- AMBA AHB
- AMBA APB
- Peripherals
- UART
- Interrupt Controller
- I2C

Pre-assembled Platforms
DesignWare System-Level Library

High-Performance Processors
- ARM7TDMI
- ARM920T
- ARM926EJ-S
- ARM946E-S
- ARM1176JZF-S
- ARM1136JFS

DesignWare Bus Infrastructure & Peripherals
- AMBA AHB/APB peripherals
- UART
- Interrupt Controller
- WatchDog Timer
- DMA Controller
- I2C

DesignWare Connectivity IP Cores
- USB 2.0 High Speed OTG
- SATA AHCI
- Ethernet

Pre-Assembled Platforms
- Multi-media Player
- ARM Integrator
- Multi-layer AMBA Reference Platform
- VP Test (test platform)
ARM Developers’ Conference 2007

Services Approach to Efficient Virtual Platform Creation
Dedicated Modeling Team

• **Modeling is a Skill!**
  - Experience is essential for efficient model creation
  - Ability to model at the appropriate level of abstraction
  - Need to understand Embedded Software design
  - Need to understand hardware concepts
Defined, Repeatable Process

• **Platform Creation requires more than Modeling Skills**
  - Consistent, predictable model creation
    - Component Creator
    - Design entry alternatives
  - System level awareness – not just a collection of models
    - Virtual I/O
    - 3rd party tool integration
  - Standard product packaging and delivery process.
    - Mature process – delivered multiple products to multiple customers
Customer Visibility

• **Typical Customer Engagement**
  - Dedicated Project Management on both sides
  - Control of specification availability / updates from customer
  - Weekly progress meetings – risks and issues tracking

• **Project Driven from Statement of Work**
  - Agreed by customer and supplier
  - Defines models to be developed and system level requirements
  - Typically defines Phased delivery schedule
Our Experience - Practical Model Development

“JIT” virtual platform development & releases

- Deliver platform incrementally and with growing functionality
- Periodically, every 3-4 weeks
- Plan deliveries to match staggered SW development tasks
- Developed by teams of 4-5 engineers within 3-6 months

=> Allows for earliest platform use

Apply to Device / Family Roadmap

- Subsystem re-use shortens platform development
- Follow IP (hardware) re-use

=> Enables getting ahead of silicon by 9-12 months!
DesignWare™ Virtual Platforms Decreases SW Development Time for Complex SoCs

Case Study: TI OMAP2420

• Problem – Shorten SW dev and HW/SW integration schedule
  - ROM Code
  - OS Base Ports
  - DSP Bridge SW
  - Customer software

• Solution – Pre-silicon DW Virtual Platform
  - Delivered first platform for ROM code in 3 weeks
  - Multiple deliveries: JIT model

Results – Average software productivity increased by 2x
  - OS base port: completed in half the time
  - DSP Bridge: made up 20 days on schedule in 30 days with virtual platform
  - Customer software: deployed to key customer 7 months ahead of silicon
DesignWare™ Virtual Platforms Decreases SW Development Time for Complex SoCs

Delivery Example: TI OMAP2420

• First delivery in 4 weeks
  ▪ Enabled OS development start
  ▪ 10 months ahead of silicon!

• 85-90% of all SW developed pre-silicon

<table>
<thead>
<tr>
<th>Deliverable</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1610 with OS Porting Peripherals</td>
<td>Start + 4 weeks</td>
</tr>
<tr>
<td>1136 ARM mode only for DSP/BIOS Bridge development</td>
<td>Start + 6 weeks</td>
</tr>
<tr>
<td>Security Driver support</td>
<td>Start + 8 weeks</td>
</tr>
<tr>
<td>Remaining peripherals integrated</td>
<td>Start + 11 weeks</td>
</tr>
<tr>
<td>Multimedia</td>
<td>Start + 15 weeks</td>
</tr>
<tr>
<td>IVA</td>
<td>Start + 20 weeks</td>
</tr>
<tr>
<td>GFX</td>
<td>Start + 24 weeks</td>
</tr>
<tr>
<td>2420 Beta platform delivery including draft documentation</td>
<td>Start + 24 weeks</td>
</tr>
<tr>
<td>Final Platform delivery including final documentation</td>
<td>Beta end + 2 weeks</td>
</tr>
<tr>
<td>Expected Project Duration</td>
<td>Start + 30 weeks</td>
</tr>
</tbody>
</table>

Table 3: Top Level Project Deliverables
Texas Instruments
Proven Results & Successes

- 9-12 month pre-silicon availability, allowing 85-90% of all software to be developed pre-silicon
- 2-5x SW dev. productivity improvement
- “Single day HW/SW integration”
- Deployed to TI customers

“TI used the VPOM-2430 Virtual Platform successfully to accelerate software development for the OMAP2430 processor device. Because of the effective simulation environment provided by Virtio, our teams were able to immediately run and test our software when the OMAP2430 processor became available. Virtio is a key element of TI's plan to reduce the time needed to provide software after silicon becomes available.”

Synopsys System-Level Solutions

All Of The Ingredients for Virtual Platforms

DesignWare System Level Library

- DesignWare System-Level Library
- SystemC™ Transaction Level Models
- Processors
- DesignWare Cores
- DesignWare AMBA Components
- Pre-Assembled Platforms

Innovator

- Abstract, high-performance models of SoC blocks
- Environment for developing, running & debugging virtual platforms

Services

- Expert services for model creation, virtual platform assembly & customization

Virtual Platforms
Synopsys – The Virtual Platform Leader

• Over 50 engineering years of Virtual platform and TLM modeling experience
  - Unmatched on-time, high-quality modeling services track record

• Over 50 platforms deployed
  - Fastest and most complete virtual platforms
  - Incl. TI OMAP1®, OMAP2®, OMAP3®, Freescale i.MX, MXC, Intel® XScale™
  - In the hands of SW developers today
# Virtual Platforms – ESL “Killer App”

*It’s Like Hardware – Only Better!*

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available Early</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>

© 2007 Synopsys, Inc.
ARM Developers’ Conference 2007

Questions & Answers