Open Virtual Platforms (OVP)
An Introduction and Overview

info@ovpworld.org

Dec 2011
Agenda

- Embedded software is The Big Problem with SoC products
- Existing embedded software solutions
- OVP
The growing challenge

- SW content of electronic products grows dramatically
  - Millions and millions lines code
  - In 2007 SW dev costs exceeds HW design costs for SoC ICs

- and the software needs to run faster and faster to provide more and more functionality

---

Source: Xilinx
The real solution is Multi-Core

“Von Neumann is a poor use of scaling – all the energy is going on the communication between the processor and the memory. Its much better to use 20 microprocessors running at 100MHz than one at 2GHz”

Hugo de Man, IMEC

- Early movers have been building multi-core standard processors
- And more and more System on Chips (SoCs) and Platform chips are becoming multicore
Processor count predicted to increase dramatically

Source: ITRS 2006 Update

DPE: Data Processing Engine

# of Components

Max Processing Performance [TFLOPS]

Year of Production

0 0.1 1 10 100

0 5 10 15 20 25 30


# of Main CPUs # of DPEs Max Processing Performance
Embedded Software for MPSoCs: An extreme challenge!

“30 to 50 per cent of R&D budgets are spent on software, and the cost is rising 20 per cent a year. The software effort overtakes the hardware effort at 130nm.”

Jack Browne, MIPS Technologies

“Some say we are at a crisis stage with the software side overwhelming the hardware side. Driving some of this is the proliferation of cores in system-on-chip (SoC) devices.”

Steve Roddy, Tensilica

SW problems delay SoC revenues, impacting IP developer royalties
Productizing MPSoCs

• So what does the future hold for software that is to run on these multi-core chips?
• Understanding what the future could look like, how can we make that better?
Current SoC (Software on Chip!) development methodology

- Start developing application software in parallel with hardware
  - But cannot test software
- Get chip or FPGA on prototype board
  - The hardware is always far too late in the product development cycle
  - Also it is a limited, unreliable and not up to date prototype / platform
- Connect up debugger to JTAG port
- Then there is a real challenge in verifying and debugging MP software
  - Lack of controllability, visibility, precision
  - And poor MP support
- More and more teams scrabbling about in looking for a better solution

Moving to Virtual Platforms for earlier testing of software

If initial testing of software is done on Virtual Platform, could reduce SoC schedule by months, and reduce initial development and maintenance costs significantly for SoC embedded software
Observation: New Embedded Software Development Tools needed (especially for multicore)

The current market is here

Abstraction and Efficiency

Adoption over time

The Four Steps to the MP Epiphany™

- Phase 1: simulate platforms at speed for OS porting, driver and application development
  - Virtual platform: multicore simulation, user modeling, model library

- Phase 2: next generation functionality built on simulation base – verification solutions for software development
  - new products from new companies

- Phase 3: manage/deliver application software for heterogeneous platforms
  - Workbench

- Phase 4: better programming paradigms and tools to address parallel programming and multicore platform issues
  - Programming Model, Tools, Automation

Virtual Platforms Market Grows Fast

- “Virtual Platform and simulation tools are the fastest growing segment of the electronic system level tools market …”
- “28 percent of respondents are using virtual prototyping today, and 44 percent expect to use it within the next two years.”
- “The challenge of designing and testing software earlier in the design process is becoming an increasingly significant factor, especially in cases where the hardware environment may be extremely complex or not yet available.”

  - Matt Volckmann, Senior Analyst with VDC's embedded software practice,

⇒ Big demand currently and near future
⇒ But for what virtual platform technology?
Virtual Platforms Types

- Hardware Virtual Platforms
  - Timing / Cycle accurate
  - Used for architecture performance analysis, drivers, firmware
  - Models are very complex, slow, time consuming to build
  - SystemC etc
  - Main value is same performance as RTL, no Verilog license cost…

- Software Virtual Platforms
  - Instruction accurate
  - Used for OS, applications
  - Can be very fast, programmers views
  - Model only what is needed in peripherals
  - Complete system environment
Virtual Platforms market today is fragmented, and diverging

- Approaches
  - Roll your own – from scratch in C/C++/SystemC
  - Contract product provider as service to build models of IP and platform
  - Use commercial solution to write your own models
  - Existing open source

- Today’s alternatives: pro
  - Get a virtual platform

- Today’s alternatives: con
  - Takes too long to develop
  - High risk
  - Lack of control for service model
  - Proprietary solutions
  - No interoperability
  - Cost of deploying simulation environment
  - Different solutions for different models/IP architectures
  - Designed for single cpu model not embedded MP platform

⇒ This fragmentation is not a good situation if virtual platforms are the enabler for MPSoCs
What is needed?

- **Open way of modeling needed for Virtual Platforms**
  - Targeted at Instruction Accurate Software Virtual Platform need
  - Easy to use, high level, 100s of MIPS
  - Covers complexity of current designs easily
  - Built-in interoperability for models from different developers
  - Proven and in use technology

- **Methodology that leads to an ecosystem**
  - Ability to enable model builders to protect IP
  - Interfaces that enable the growth of tool chains
  - Tools that provide verification, debug, analysis of embedded sw

- **Backwards compatible with legacy solutions**
Wind River View

- Tomas Evensen, Wind River Systems CTO, said he's noticed a shift during the past couple of years both within his company and among Wind River customers towards an increasing use of various types of simulation, including virtual platform systems. Previously, he noted, software developers wanted real hardware, but now "they have to start using simulation because there's no chip available."

- Simulators are especially valuable for multicore platforms, Evensen noted, because real multicore hardware tends to be non-deterministic and exhibit race conditions.
  - SCDSource.com
Introducing OPEN VIRTUAL PLATFORMS

- “Imperas believes that software virtual platform infrastructure should be open and be freely available.
- To that end, we are sharing, making public, and making open our simulation infrastructure technologies with the intention of establishing a common, open standard platform for software virtual platforms for software developers.
- We are also placing many complex processor and peripheral models in open source.
- Imperas will support and manage the OVP site, and will contribute much of our innovation to keep this infrastructure evolving.
- However, it is not solely through our efforts that these technologies become successful. Participation of organizations and individuals around the world is critical to the success of OVP.

- We offer our thanks to all those that are participating in this community.”

Simon Davidmann, CEO Imperas and OVP Founder
Virtual Platforms (OVP) are the foundation for the next generation of [embedded] software development environments – especially for multicore.

The Four Steps to the MP Epiphany™

**Phase 1:** simulate platforms at speed for OS porting, driver and application development
- Open Virtual Platforms

**Phase 2:** next generation functionality built on simulation base – verification solutions for software development
- new products from new companies

**Phase 3:** manage/deliver application software for heterogeneous platforms

**Phase 4:** better programming paradigms and tools to address parallel programming and multicore platform issues

OVP fulfils this need
Requirements on OVP Software Virtual Platforms

- Easy to create virtual platforms of MANY peripherals and MANY processors
- Easy to create your own processors, peripherals, platforms
- Library of processor and peripheral models
- Full programmers views, registers, addressing, interrupts
- Model only what is needed in peripherals
- Simulations are Instruction Accurate, very fast
- Used for application, OS, embedded software development
- Connect to 3\textsuperscript{rd} party debuggers, e.g. GDB
- Efficient, Complete system environment for developing embedded software
What is in OVP?

- **Modeling APIs**
  - Publishing of C OVP APIs for Processor, Peripheral, and Platform modeling
  - Documentation & header files

- **Open Source library of models**
  - C source of models written to C OVP APIs
  - Processor models of ARM, ARC, MIPS, PowerPC, Renesas V850, M16c, Xilinx MicroBlaze, OpenRisc, SPARC, x86, …
  - Peripheral models of standard embedded devices
  - Example embedded platforms in C, C++, SystemC, TLM2.0
    - Including full platforms that boot operating systems like uClinux, Linux, Nucleus, Micrium uC/OS-II, FreeRTOS

- **OVP reference simulator, free for non-commercial use**
  - Runs processor models fast, 500 MIPS typical
  - Interfaces to GDB via RSP/socket
  - MP Capable, scalable and very efficient
  - Can encapsulate existing processor models (ISS)
  - Callable with C/C++/SystemC/TLM2.0
OVP Community

- Founded March 2008 by Imperas
- Website community/portal
- Nearly 5,500 people registered on the website
- OVP ecosystem
  - Users
  - Tool developers needing integration/interoperability, e.g.
    - SystemC simulators
    - Cadence, Carbon, CoWare, Mentor, OSCI, Synopsys
    - Emulation/FPGA prototyping
    - Aldec, EVE, Hitachi, Mentor
  - Engineering services companies available to provide model building services to users
  - Universities for both research and courses
Components of OVP Processor Models

- L1/L2 Cache
- Decode instruction from memory
- Instruction Behavior and Disassembly
- Standard interface to Debugger (RSP)
- Exception Modeling
- High Performance Model TLB / MMU
- Asynchronous External Events

- Model cache (MMC)
- Decode Instruction from Memory
- Instruction Behavior and Disassembly
- Standard Interface to Debugger (RSP)
- Exception Modeling
- High Performance Model TLB / MMU
- Asynchronous External Events

- Create and manage resources (e.g., Registers) shared between multiple processor instances
- Add processor independent I/O support
- Add instrumentation into application (no overhead)

=> complete, proven technology and methodology – use our models, or write your own…
OVP includes Native SystemC TLM 2.0 interface

Example showing TLM2.0 platform encapsulating OVP fast CPU models with shared memory
- Local memory for heap and stack
- Shared memory for program and data
- Very simple to use and runs very fast
Library of OVP Fast Processor Models

- Verification, Licensing, and Distribution Partnership with MIPS
  - MIPS Processor cores verified by MIPS Technologies under the **MIPS-Verified™** program
  - Current verified models:
    - Single core MIPS32 4KEm/Ec/Ep
    - Single core MIPS32 24KEc/Ef/Kc/KF
    - Single core MIPS32 74Kc/Kf
    - microMIPS single core M14KcTLB/KcFMM
    - Dual core MIPS32 34Kc/Kf
    - Quad core MIPS32 1004Kc/Kf, 1074Kc/Kf

- Verification performed by ARC
  - Imperas developed models of ARC processors **verified by ARC**
    - e.g. single core ARC605
  - ARC6xx, ARC7xx families available

- Models of ARM cores
  - ARMv4, ARMv5, ARMv6, ARMv7 architectures including MMU, MPU, TCM, VFP, Neon
  - Processor core models include Cortex-A9, Cortex-A9MP, Cortex-A8, Cortex-A5, Cortex-M3, Cortex-M4
  - Models of Classic cores - ARM1136, 1156, 920, 922, 926, 940, 946, 966, 968, 7EJ-S, 720T, 7TDMI

- Renesas (NEC) Electronics models of v850, v850ES, v850E1, v850E2, M16C

- Models of Xilinx MicroBlaze, verified with Xilinx verification suites

- Licensing, Distribution Partnership with Tensilica
  - Models of Tensilica Xtensa® and Diamond Cores run in OVP platforms
  - OVP Tensilica model is instanced and used in the same way as native OVP processor models

- Other CPU families (not certified) available: PowerPC, openCores, x86, proprietary
Modeling Processors using OVP - some performance numbers for OVP CPUs

Required Speed

<table>
<thead>
<tr>
<th>OVP CPU Models</th>
<th>1 GIPS</th>
<th>100 MIPS</th>
<th>10 MIPS</th>
<th>1 MIPS</th>
<th>100 KHz</th>
<th>10 KHz</th>
<th>1 KHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Software Development</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Firmware Development</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Driver Development</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Transaction Level Verification (SystemC)</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Hardware / Software Co-Verification</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>RTL Hardware Verification</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

OVPsim solution: Ready for next generation application software development!

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Simulated Instructions</th>
<th>Run time</th>
<th>Simulated MIPS</th>
<th>Simulated Instructions</th>
<th>Run time</th>
<th>Simulated MIPS</th>
<th>Simulated Instructions</th>
<th>Run time</th>
<th>Simulated MIPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>linpack</td>
<td>5,028,517,285</td>
<td>5.59s</td>
<td>900</td>
<td>1,119,997,559</td>
<td>1.21s</td>
<td>926</td>
<td>1,450,948,024</td>
<td>1.03s</td>
<td>1402</td>
</tr>
<tr>
<td>Dhrystone</td>
<td>2,118,116,353</td>
<td>1.86s</td>
<td>1139</td>
<td>1,082,061,368</td>
<td>1.42s</td>
<td>763</td>
<td>974,078,238</td>
<td>1.12s</td>
<td>873</td>
</tr>
<tr>
<td>Whetstone</td>
<td>10,924,250,068</td>
<td>9.96s</td>
<td>1097</td>
<td>1,324,605,091</td>
<td>1.59s</td>
<td>833</td>
<td>4,265,685,378</td>
<td>2.03s</td>
<td>1510</td>
</tr>
<tr>
<td>peakSpeed1</td>
<td>1,700,004,130</td>
<td>0.81s</td>
<td>2109</td>
<td>1,400,001,624</td>
<td>0.83s</td>
<td>1697</td>
<td>1,400,001,382</td>
<td>0.59s</td>
<td>2379</td>
</tr>
<tr>
<td>peakSpeed2</td>
<td>7,600,010,999</td>
<td>3.5s</td>
<td>2174</td>
<td>7,600,001,850</td>
<td>4.48s</td>
<td>1695</td>
<td>6,400,001,651</td>
<td>2.94s</td>
<td>2257</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Simulated Instructions</th>
<th>Run time</th>
<th>Simulated MIPS</th>
<th>Simulated Instructions</th>
<th>Run time</th>
<th>Simulated MIPS</th>
<th>Simulated Instructions</th>
<th>Run time</th>
<th>Simulated MIPS</th>
<th>Simulated Instructions</th>
<th>Run time</th>
<th>Simulated MIPS</th>
<th>Simulated Instructions</th>
<th>Run time</th>
<th>Simulated MIPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>linpack</td>
<td>4,104,162,664</td>
<td>5.53s</td>
<td>757</td>
<td>4,991,344,159</td>
<td>7.06s</td>
<td>707</td>
<td>3,163,966,113</td>
<td>3.95s</td>
<td>802</td>
<td>3,163,966,113</td>
<td>3.95s</td>
<td>802</td>
<td>3,163,966,113</td>
<td>3.95s</td>
<td>802</td>
</tr>
<tr>
<td>Dhrystone</td>
<td>1,262,082,476</td>
<td>1.56s</td>
<td>809</td>
<td>2,564,132,573</td>
<td>2.66s</td>
<td>965</td>
<td>766,066,243</td>
<td>1.03s</td>
<td>763</td>
<td>766,066,243</td>
<td>1.03s</td>
<td>763</td>
<td>766,066,243</td>
<td>1.03s</td>
<td>763</td>
</tr>
<tr>
<td>Whetstone</td>
<td>7,830,770,799</td>
<td>7.61s</td>
<td>1030</td>
<td>6,202,408,024</td>
<td>7.21s</td>
<td>860</td>
<td>3,515,867,502</td>
<td>3.65s</td>
<td>963</td>
<td>3,515,867,502</td>
<td>3.65s</td>
<td>963</td>
<td>3,515,867,502</td>
<td>3.65s</td>
<td>963</td>
</tr>
<tr>
<td>peakSpeed1</td>
<td>1,500,001,786</td>
<td>1.02s</td>
<td>1471</td>
<td>1,200,003,128</td>
<td>0.87s</td>
<td>1374</td>
<td>1,000,002,001</td>
<td>0.65s</td>
<td>1549</td>
<td>1,000,002,001</td>
<td>0.65s</td>
<td>1549</td>
<td>1,000,002,001</td>
<td>0.65s</td>
<td>1549</td>
</tr>
<tr>
<td>peakSpeed2</td>
<td>6,800,002,055</td>
<td>3.91s</td>
<td>1740</td>
<td>6,000,007,540</td>
<td>3.26s</td>
<td>1842</td>
<td>5,200,002,884</td>
<td>2.5s</td>
<td>2078</td>
<td>5,200,002,884</td>
<td>2.5s</td>
<td>2078</td>
<td>5,200,002,884</td>
<td>2.5s</td>
<td>2078</td>
</tr>
</tbody>
</table>

All measurements on 2.83GHz Intel Core2, OVPsim 20100528.0
And OVP TLM2.0 is fast too

- Using DMI ensures fastest possible speeds
- OVP CPU models in TLM2.0 platforms provide 200-500 MIPS performance
- Use un-modified cross compiled binaries
- Memory can be in OVP model or SystemC platform

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Single: ARM Core</th>
<th>MultiCore: 2 MIPS Cores</th>
<th>ManyCore: 24 ARC Cores</th>
<th>Single: NEC v850</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simulated</td>
<td>Simulated</td>
<td>Simulated</td>
<td>Simulated</td>
</tr>
<tr>
<td></td>
<td>Instructions</td>
<td>Instructions</td>
<td>Instructions</td>
<td>Instructions</td>
</tr>
<tr>
<td></td>
<td>Run time</td>
<td>Run time</td>
<td>Run time</td>
<td>Run time</td>
</tr>
<tr>
<td></td>
<td>Simulated MIPS</td>
<td>Simulated MIPS</td>
<td>Simulated MIPS</td>
<td>Simulated MIPS</td>
</tr>
<tr>
<td>linpack</td>
<td>578,061,601</td>
<td>6,505,622,575</td>
<td>100,419,903,936</td>
<td>20,244,902,429</td>
</tr>
<tr>
<td>fibonacci</td>
<td>531,517,832</td>
<td>1,739,440,093</td>
<td>23,772,708,000</td>
<td>536,096,399</td>
</tr>
<tr>
<td>Dhrystone</td>
<td>4.544,064,499</td>
<td>1,010,156,498</td>
<td>14,977,987,872</td>
<td>1,168,132,213</td>
</tr>
<tr>
<td>peakSpeed1</td>
<td>5.600,061,524</td>
<td>11,200,002,891</td>
<td>144,000,042,864</td>
<td>4,800,003,128</td>
</tr>
</tbody>
</table>

All measurements on 3 GHz Pentium
Agenda

- Embedded software is The Big Problem with SoC products
- Existing embedded software solutions
- OVP
  - Demonstrations available for easy download from www.OVPworld.org/download.php
OVPsim Example Single Processor Platform

- Provide socket to attach debugger
- Attach GDB to single instance in platform

Several pre-compiled examples
Fibonacci
Dhrystone
Linpack
Peakspeed1

Easy to run
- C:\> platform.exe application.elf
- Loads application into memory and runs it
- On 3GHz PC runs up to 2000 MIPS
OVPsim Demo Single Processor – Fibonacci series

Platform Source (C compiled on X86 Windows)

Application Source (C cross-compiled for target processor)

OVPsim run log (runs easily and very fast on normal Windows PC)
OVPsim Demo Single Processor - Peakspeed

Example of peakspeed
- Simple contrived example application
- Demonstrates maximum speed

Shows maximum speed – depends on application and platform – 1,000 MIPS possible on desktop...
OVPsim multicore2
2 Processor Platform

- Application is Fibonacci series generated on one and read by second processor from shared memory
- Local memory
  - heap and stack
- Shared memory
  - program and data

- Easy to create platform and use
  - C:\> platform .exe application .elf
  - Loads application into shared memory and runs it
  - On 3GHz PC runs up to 500 MIPS
2 processors in platform

Add buses & connect

Add memory

Connect memories to buses

Easy to see instructions per processor and cumulative
- No slow down with sharing resources!
OVPsim manycore24 demo
24 Processor Platform

- Any number of processors can be instantiated in platform
- No penalty on system simulation performance

- Easy to create platform and use
  - C:\> platform .exe application .elf
  - Loads application into local memory and runs it
  - On 3GHz PC runs up to 500 MIPS
Easy to create platforms of many processors

Easy to load software onto many processors

Easy to see instructions per processor and cumulative - no slow down with more processors! - simulate easily 2048 processors, and more...
Hetero multicore4
4 Processor Platform
(3x ARM7, 1x MIPS32)

Easy to run: C:\> platform .exe application .elf
Tensilica Integration

- Tensilica ISS Encapsulated in OVPsim
- OVP instances just like native processors
- In this example FIFOs used for communication
- Easy to run: C:\> platform .exe application .elf
OVPsim Demo Tensilica Integration

2X Diamond Cores

Encapsulated Models are instanced easily like native OVP models

Software on processor 1 is reader

OVP includes processor to processor communication using FIFOs

Simulation runs easily

Software on processor 2 is writer
OVP enables debugging with GDB attached to one processor

OVPsim platform waits for GDB debugger connection on standard network socket

GDB target remote command connects GDB to OVPsim processor

Once connected GDB enables full source level debugging of simulated application

Port for GDB connection is specified in icmInit() - zero means use any free port

Processor to debug in OVPsim platform is nominated as part of icmNewProcessor()
Debugging with Eclipse is easy using CDT encapsulation

Eclipse enables graphical source-level debug of OVPsim simulated application – Platform and debugger launched with two mouse clicks.

Eclipse connects to OVPsim through GDB using GDB’s remote serial protocol.

Low-level processor state can also be inspected and controlled.
OVPsim Booting uClinux (OR1K)

- Easy to run
  - C:\> platform.exe uClinux.elf
  - Loads application into memory and runs it
  - In telnet console can interact with uClinux
OVPsim Demo running simple OS - uClinux

OVPsim console starts up - and then awaits a telnet session - When session finished it exits with information

Use hyperTerminal to connect to UART in platform - When type in terminal - interact with OS running on simulated processor
OVPsim booting uClinux on ARM Atmel AT91sam7

Boot uClinux
OVPsim booting Nucleus on ARM Integrator / ARM920

Easy to run

telnet localhost 9999
OVPsim booting Linux on ARM Integrator / ARM926

OVPsim booting Linux on ARM Integrator / ARM926

Keyboard / Mouse

Boot Linux

OVPsim MIPS Linux platform

- Boot Linux on Windows PC…
- Easy to run: platform .exe vmlinux
OVPsim Demo running Linux 2.6 on MIPS32

- OVPsim console starts up and then runs Linux booting
- User types into VGA screen
- When session finished it exits with information

- VGA peripheral in platform displays information
- User types into display window and interacts with Linux running on simulated processor

- Booting linux takes only a few seconds
- Platform runs at several times real time
- In this example platform is 100MHz chip, and simulation runs approx. 4X faster...
- Your mileage may differ...
OVPsim MIPS platform
MIPS Malta / MIPS 24K / Android
And OVP works for MP users

- VSMP Linux in MIPS
  - Malta 34K platform
    - 2 processors
    - Easy to run
OVPsim Homogeneous Platform, Dual ARM Cortex-A9
Heterogeneous OS: ARM Linux / ARM Nucleus

- Two cores
  - One running Linux Kernel (Output using LCD peripheral)
  - One running Nucleus Demo (Output using UART)
OVPsim Heterogeneous Platform
ARM Cortex-A9MPx2 running Linux and Xilinx MicroBlaze

Keyboard / Mouse

Xilinx MicroBlaze

- UART
- Memory
- Peripheral

Bridge

ARM Cortex-A9

- SSRAM
- SDRAM
- PIC
- config regs

ARM Cortex-A9

- UART
- LED
- RTC
- MMC Interface
- UART

Xilinx MicroBlaze

- PIC
- AHB Decoder
- GPIC

Keyboard/Mouse

LCD Controller

Flash
ARM Versatile Express
Cortex-A9MPx2 / SMP Linux

This root FS contains most basic Linux utilities (implemented with busybox) and the Lynx web browser.

Kernel config is available through /proc/config.gz

Welcome to OVP simulation from Imperas

Log in as root with no password.

Imperas login:
ARM Versatile Express
Cortex-A9MPx1 / Android
What products does Imperas sell: Verification is key in SW Development

- “Debugging is the most time-consuming aspect of the embedded software development process, and multicore debugging challenges are especially acute, according to 354 developers who responded to a survey conducted at the April Embedded Systems Conference”
- “56 percent of survey respondents said that debugging is the most time-consuming aspect of their work. In contrast, 24 percent pointed to writing original code.”
- “35 percent of respondents said they spend over half of their total development time debugging.”
- “59 percent said the debugging tool they use does not support multicore or multiprocessor development. It’s an important question, because 68 percent said they spend at least some of their time working on multicore or multiprocessor systems. 20 percent, in fact, spend over half their time on such systems.”

- EETimes, Richard Goering

⇒ Imperas products are focused on verification, debug and analysis of embedded software for all SoCs– please visit www.imperas.com
OVP Summary

- The key to efficiency for future software development environments is the use of simulation and Virtual Platforms
- The availability of high performance, high quality models is critical
- The Open Virtual Platforms solution is increasingly important – especially with the move to MP
- The Imperas technology donation and formation of OVP kick starts a new phase in embedded software development
- OVP provides a fantastic OpenSource modeling solution to be the foundation of the next generation of software development environments
- Committed collection of partners developing the ecosystem
  - The most important opening & donation since SystemVerilog
Open Source models in Wikki based library, User Forums, Downloads, Presentations, Documentation...
Thank you for watching.

Please have a look at what the ecosystem is saying at:
FAST SIMULATION, FREE MODELS, EASY TO USE

http://www.ovpworld.org

Enabling the next generation of embedded [MultiCore] Software Development
Thanks
Thanks
Thanks
Thanks