Hardware Emulation: ICE vs Virtual
A Mano-a-Mano Duel for Supremacy

Lauro Rizzatti
Hardware Emulation Consultant
Agenda

• Hardware Emulation Timeline
• Hardware Emulation Evolution
  – The Objectives, The Modes, The Targets
  – The Usage
  – The Technology
• ICE versus Virtual
  – In-Circuit-Emulation (ICE) Mode
    • What is it?
    • Why Customers use ICE?
  – Virtual (or Acceleration) Mode
    • What is it?
    • Why Customers use Virtual Mode?
  – Advantages and Disadvantages
# Hardware Emulation Timeline

<table>
<thead>
<tr>
<th>First Decade</th>
<th>Second Decade</th>
<th>Third Decade</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FPGA-Based</strong></td>
<td><strong>FPGA-Based &amp; Custom</strong></td>
<td><strong>FPGA-Based &amp; Custom</strong></td>
</tr>
<tr>
<td>In-Circuit-Emulation (ICE)</td>
<td>ICE &amp; HDL Acceleration</td>
<td>ICE, All Types of Acceleration</td>
</tr>
<tr>
<td>• Zycad (1981)</td>
<td>• Meta-Systems (Mentor)</td>
<td>• Mentor</td>
</tr>
<tr>
<td>• Ikos (1984)</td>
<td>• Cadence (Quickturn)</td>
<td>• Cadence</td>
</tr>
<tr>
<td>• Quickturn (1987)</td>
<td>• Mentor (Ikos)</td>
<td>• Synopsys (EVE)</td>
</tr>
<tr>
<td>• PiE (1990)</td>
<td>• Akis Systems (Verisity/Cadence)</td>
<td>• EVE</td>
</tr>
</tbody>
</table>

---

2/19/2016
Lauro Rizzatti – RIZZATTI LLC
Hardware Emulation: Objectives, Modes, Targets

Early Years

• **One** Verification Objective
  • HW Functional Verification

• **One** Deployment Mode
  • In-Circuit-Emulation (ICE)

• **One** Target Industry
  • CPU/Graphics (largest designs in 1985-1995)

Today

• **Multiple** Verification Objectives
  • HW Functional Verification, HW/SW Integration, SW Validation, Power Domain Verification, Power Estimation, Performance Characterization,…

• **Multiple** Deployment Modes
  • ICE, Acceleration, Virtualization,…

• **Multiple** Target Industries
  • CPU/Graphics, Mobile, Video, Networking, Storage,…
Hardware Emulation: Usage

Early Years

• One User
• Local Resource
• No Remote Access
• Unfriendly to use

Today

• Multiple Users (> 100)
• Central Resource
• Remote Access
• Easy to use
Hardware Emulation: Technology

Early Years

- **Commercial FPGA** based
- Design Capacity: 330K gates/Box\(^{(1)}\)
- Setup Time: Several Months
- Compilation Time: Days
- Design Debug: Partial Visibility
- Cost: ~ 5$\$/gate\(^{(1,2)}\)
- Volume: 6,000 mm\(^3\)/gate
- Weight: 2 gr\$/gate
- Power: 50 mWatt\$/gate
- Reliability: < One Day

Today

- **Custom silicon** or **Commercial FPGA** based
- Design Capacity: > 250M gates/Box\(^{(3)}\)
- Setup Time: < One Week\(^{(3)}\)
- Compilation Time: Hours\(^{(3)}\)
- Design Debug: 100% Visibility
- Cost: < 1 cents\$/gate
- Volume: < 5 mm\(^3\)/gate
- Weight: < 2 mgr\$/gate
- Power: < 5 uWatt\$/gate
- Reliability: > One Months

(1): Quickturn Enterprise System (1992)
(2): Adjusted for inflation
(3): In the FPGA-based emulator the compilation time is adversely affected by the FPGA PAR time and the 100% visibility compromises significantly the speed of execution
Hardware Emulation: The Future

**Hardware**

- Two basic architectures:
  - Commercial FPGA based
  - Custom Silicon based
- Capacity: several billion gates
- Smaller
- Faster
- Cheaper
- Highly reliable

**Supporting Software**

- Setup Time: ~ one day
- Compilation Time: less than one hour
- Integrated in a continuum of verification engines, sharing GUI and design database
- Access offered via datacenters
- ICE, acceleration, system validation, functional coverage, SVA, low-power verification, power estimation, DFT and test vector verification, performance characterization
ICE Mode: What Is It?

- In ICE mode, the emulator is plugged into a socket on the physical target system in place of a yet-to-be-built chip to support exercising and debugging the design-under-test (DUT) mapped inside the emulator with live data.
Why Customers Use ICE

- Run Real Applications
- Connect Custom/Proprietary Platforms, OS etc.
- Reproduce Prototype Setup for Debug
Challenges using ICE

- Iterative and long debug cycles
- Randomness from one run to another
- Restricts other Virtual Use Models
Virtual Mode: What Is It?

- In Virtual mode, the test environment is modeled in software at high-level of abstraction and connected to the design-under-test (DUT) mapped inside the emulator via a set of transaction-based interfaces.
Why Customers Use Virtual Mode

- Remote Access
- What-if analysis
- Deterministic behavior
Challenges using Virtual Mode

- May need to create transactors
- May need to write testbenches
- No access to proprietary IP interfaces
## ICE vs Virtual

<table>
<thead>
<tr>
<th>Criteria</th>
<th>ICE</th>
<th>Virtual</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Testbench Type</strong></td>
<td>• Physical Target System</td>
<td>• Virtual Target System</td>
</tr>
<tr>
<td></td>
<td>o Run real applications or real traffic</td>
<td></td>
</tr>
<tr>
<td><strong>Testbench Creation</strong></td>
<td>• No need to create testbenches; they are the target system</td>
<td>• Necessary to create testbenches, a time-consuming, error-prone process</td>
</tr>
<tr>
<td><strong>Interface between Target System and DUT</strong></td>
<td>• Requires speed adapters to target system for each interface protocol</td>
<td>• Requires transactors for each interface protocol</td>
</tr>
<tr>
<td></td>
<td>o Significant H/W dependencies (reset circuitry, noise)</td>
<td>o No H/W dependencies</td>
</tr>
<tr>
<td></td>
<td>o Board completion is in critical path; emulation cannot begin until board is ready</td>
<td>o Completion of S/W development is not in critical path; can begin with existing testbench and emulate parts of design incrementally</td>
</tr>
<tr>
<td></td>
<td>o Lab setup becomes a bottleneck</td>
<td>o No power consumption</td>
</tr>
<tr>
<td></td>
<td>o Power hungry</td>
<td>o Less expensive than speed adapters</td>
</tr>
<tr>
<td></td>
<td>o Additional cost</td>
<td></td>
</tr>
<tr>
<td><strong>Customization</strong></td>
<td>• ICE is critical for customized interfaces</td>
<td>• Custom virtual interface protocols can be built but require unusual expertise not readily available</td>
</tr>
<tr>
<td></td>
<td>o Fidelity of the model</td>
<td></td>
</tr>
<tr>
<td></td>
<td>o Non-standard interface</td>
<td></td>
</tr>
<tr>
<td></td>
<td>o Knowledge (IP) can’t be shared outside the company</td>
<td></td>
</tr>
<tr>
<td></td>
<td>o Development platforms can be supplied by OEM vendors to early adopters</td>
<td></td>
</tr>
<tr>
<td></td>
<td>o Reproduce FPGA Prototype Setup for Complete Debug</td>
<td></td>
</tr>
</tbody>
</table>
## ICE vs Virtual

<table>
<thead>
<tr>
<th>Criteria</th>
<th>ICE</th>
<th>Virtual</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Flexibility</strong></td>
<td>• Inflexible, difficult to modify</td>
<td>• Flexible, easier to modify S/W than H/W</td>
</tr>
<tr>
<td></td>
<td>• Restrict emulation access</td>
<td>• Ideal from remote access</td>
</tr>
<tr>
<td><strong>Reusability</strong></td>
<td>• Limited reuse; if next design has different interfaces, new board design required</td>
<td>• Highly reusable; transactors are written once and used many times</td>
</tr>
<tr>
<td><strong>Productivity</strong></td>
<td>• Hinders productivity</td>
<td>• Increases productivity</td>
</tr>
<tr>
<td><strong>Reliability</strong></td>
<td>• Complex, potentially not reliable</td>
<td>• No HW means highly reliable</td>
</tr>
<tr>
<td><strong>Design Debug</strong></td>
<td>• Not deterministic</td>
<td>• Deterministic</td>
</tr>
<tr>
<td></td>
<td>• Randomness from one run to another</td>
<td>• Repeatable behavior</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td>• Run at maximum emulation speed</td>
<td>• May run at maximum emulation speed</td>
</tr>
</tbody>
</table>
Conclusion

• ICE and Virtual are not necessarily replacement for one another
• ICE is best to test the DUT with real traffic, and when proprietary interfaces are required
• Virtual is best for remote access, deterministic behavior, and what-if-analysis
Hardware Emulation: ICE vs Virtual User Experiences

Alex Starr, AMD
Traditional GPU Emulation Solution

- ICE: In-circuit emulation
- PCIe rate adaptor connects ‘Test PC’ to emulator
- Hard disk provides software
- Windows software debug done via additional PC running WinDBG

- Three main classes of testing done
  - Traditional Verification work
    - Linux Based
    - Bring up and initialization
    - Low level feature testing
  - OS Driver & User level code
    - Windows Based (Software Team)
    - Real Kernel Mode Driver
  - Feature/System diagnostics
    - Memory coherence
    - Power Gating
    - Early silicon bring up work
In-Circuit Challenges

- Efficiency, productivity and reliability challenges
  - Physical disk swapping
  - Ethernet controlled disk switchers
  - Complex target cabling a reliability problem
  - Custom hardware for “Test PC”
  - No save and restore

Emulator

AMD GPU
Multi-setup Complexity

- Efficiency, productivity and reliability challenges
  - Multiple setups increase complexity
  - Specific target hardware for each setup limits flexibility
- On a single emulator and across sites
  - Humans cause problems!
Virtual Target Solution

- Enable connection to virtualized host machines
- PCIe transactor connects from the host workstation to the emulation hardware
  - No in-circuit PCIe connection
- Removal of all in-circuit target devices
- Virtual disks allow images to be used by the VM: no physical disk swapping
- Disk images can be prepared offline in standalone VM
- WinDBG host connection maintained for software debug
## Virtual Targets: Reduced Complexity

<table>
<thead>
<tr>
<th>Challenge Area</th>
<th>Traditional ICE Solution</th>
<th>Virtual Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘Test PC’ hard disk changes</td>
<td>Physical Disk Swapping Ethernet Controlled Disk Switchers</td>
<td>Eliminated: VM disk image selection</td>
</tr>
<tr>
<td>Save/Restore</td>
<td>Not possible</td>
<td>Possible (untested)</td>
</tr>
<tr>
<td>Cabling Complexity</td>
<td>Complex Target/Test PC cabling</td>
<td>Just single cable to workstation</td>
</tr>
<tr>
<td>“Test PC” custom hardware requirements</td>
<td>Custom hardware for “Test PC” required</td>
<td>Generic workstation, unless specific new silicon required</td>
</tr>
<tr>
<td>Data Center Compatibility</td>
<td>“High touch”</td>
<td>“No touch”</td>
</tr>
<tr>
<td>Flexibility</td>
<td>Physical target/emulator board constraints</td>
<td>Dynamically reconfigurable targets via VM</td>
</tr>
</tbody>
</table>

**Notes:**
- Traditional ICE Solution includes physical hardware and cabling complexity.
- Virtual Solution reduces complexity with virtualization techniques.
- Eliminating VM disk image selection provides efficiency.
- Just single cable to workstation simplifies cabling.
- Generic workstation reduces hardware requirements.
- Dynamically reconfigurable targets via VM offer flexibility.

---

3/22/2016  Alex Starr - AMD
End User Experience

Virtual Machine Manager
Emulator Console
Virtual Target PC VM
WinDBG VM

End User Experience

Virtual Machine Manager
Emulator Console
Virtual Target PC VM
WinDBG VM
**Advanced Virtual Options**

**Coupled**
- Deterministic
- Typically reduced throughput
- Emulator clocks stop to maintain determinism
- Great for debugging
- Keeps Verification engineers warm and cozy!

**Decoupled**
- Non-deterministic
- Full speed
- Same performance as ICE
- Streaming unidirectional monitors
- Can scare Verification engineers!

![Diagram showing coupled and decoupled options](image)
ICE + Virtual Combination?

- Get the benefits of ICE and transactors
- Decouple transactors from ICE clock domains
- Concept: Controlled vs Uncontrolled clock domains
- “ICA”: In-Circuit Acceleration
- Don’t stop those clocks with dynamic targets attached
Disclaimer & Attribution

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions and typographical errors.

The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION.

AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

ATTRIBUTION
© 2016 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. Other names are for informational purposes only and may be trademarks of their respective owners.