

HLDVT Conference - November 5, 2009

Transaction-Level Modeling in an Hardware Emulation Environment

Lauro Rizzatti

*THE **FASTEST** VERIFICATION*

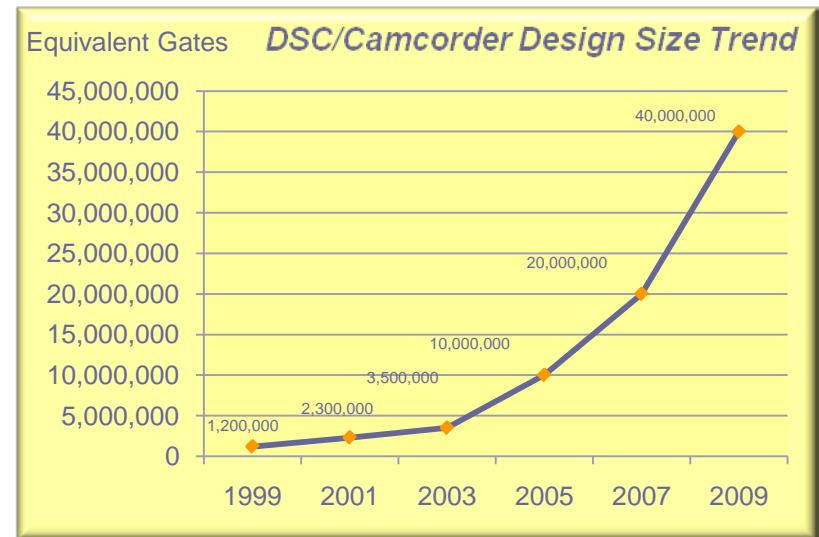
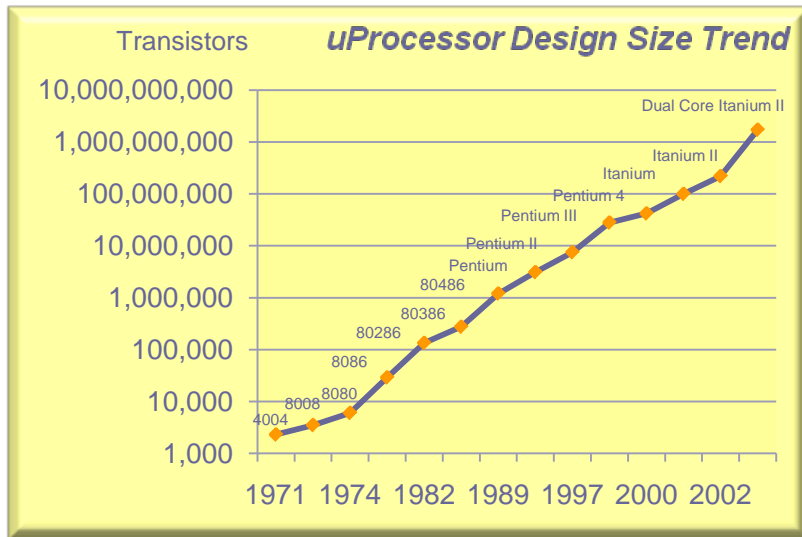


Transaction-Based Co-Emulation

- Trends in SoC Designs
- Lost Revenues by Being Late to Market
- The SOC Verification Challenge
- Deployment of Emulator in Design Flow
- Cycle-Based vs Transaction-based Co-Emulation
- In-Circuit-Emulation vs Transaction-based Co-Emulation
- Examples of Transaction-based Co-Emulation Applications

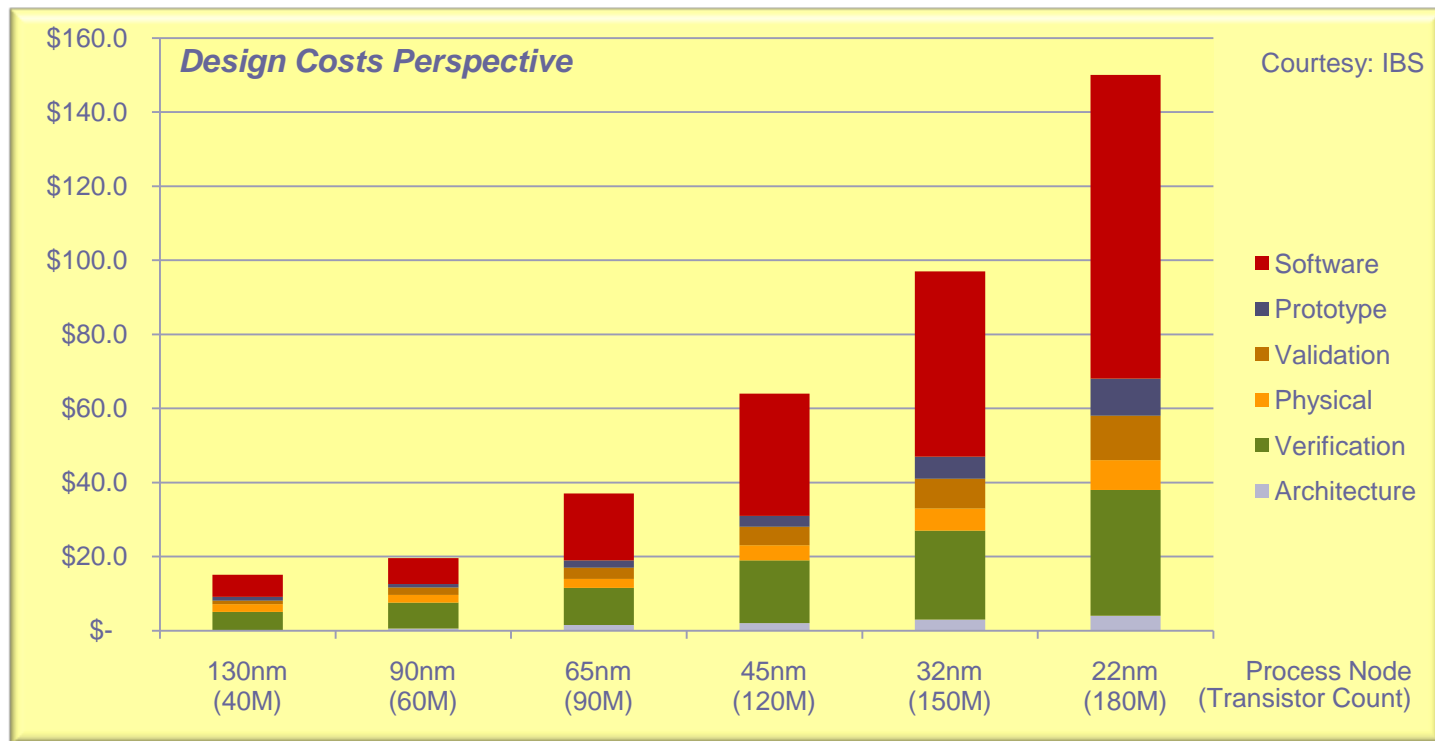
Trends in SoC Designs

- Growth of design sizes drives need for very long verification sequences



Trends in SoC Designs

- Growth of software contents makes HW/SW co-verification a critical mission



The Billion Cycles Challenge

Graphics & Video

Design	H.264 encoder/decoder
Mode	Hardware Simulation
Goal	Verify all conformance streams against golden reference



Billions
of
Verification
Cycles

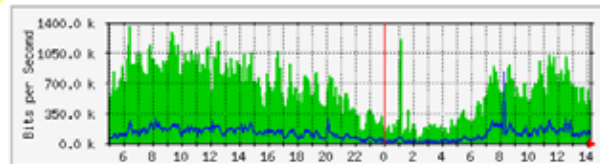
Wireless & Processors

Design	Cell phone platform (processor + DSP)
Mode	HW/SW Co-Verification
Goal	Boot operating system, device drivers and run JAVA applications on phone LCD



Networking

Design	Gigabit Ethernet router
Mode	Performance Analysis
Goal	Benchmark chip performance when subjected to long, intensive traffic

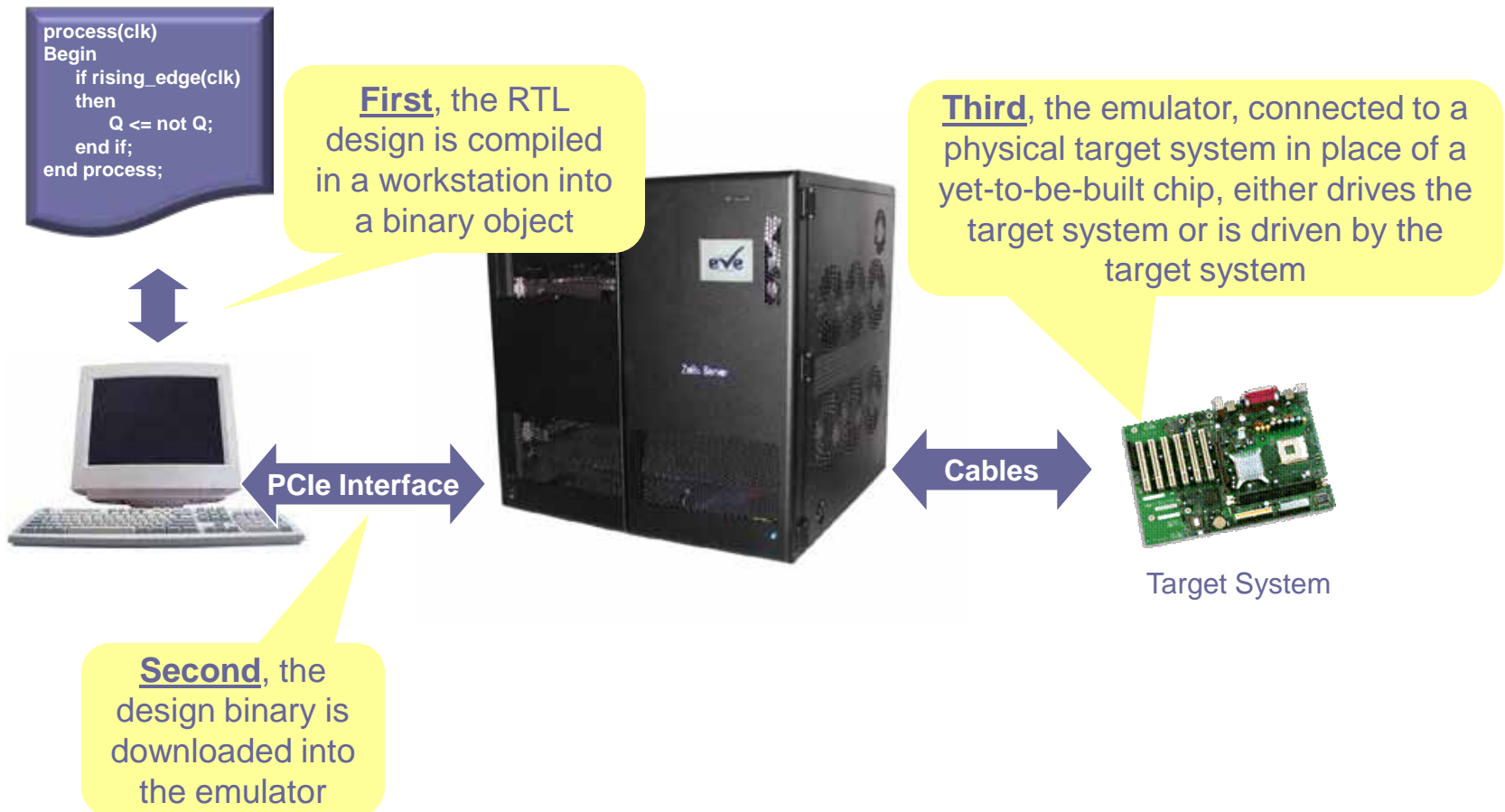


Max In: 1364.7 kb/s (13.6%) Average In: 599.8 kb/s (6.0%) Current In: 506.4 kb/s (5.1%)
Max Out: 827.2 kb/s (8.3%) Average Out: 116.7 kb/s (1.2%) Current Out: 85.9 kb/s (0.9%)

Copyright © 2009 EVE

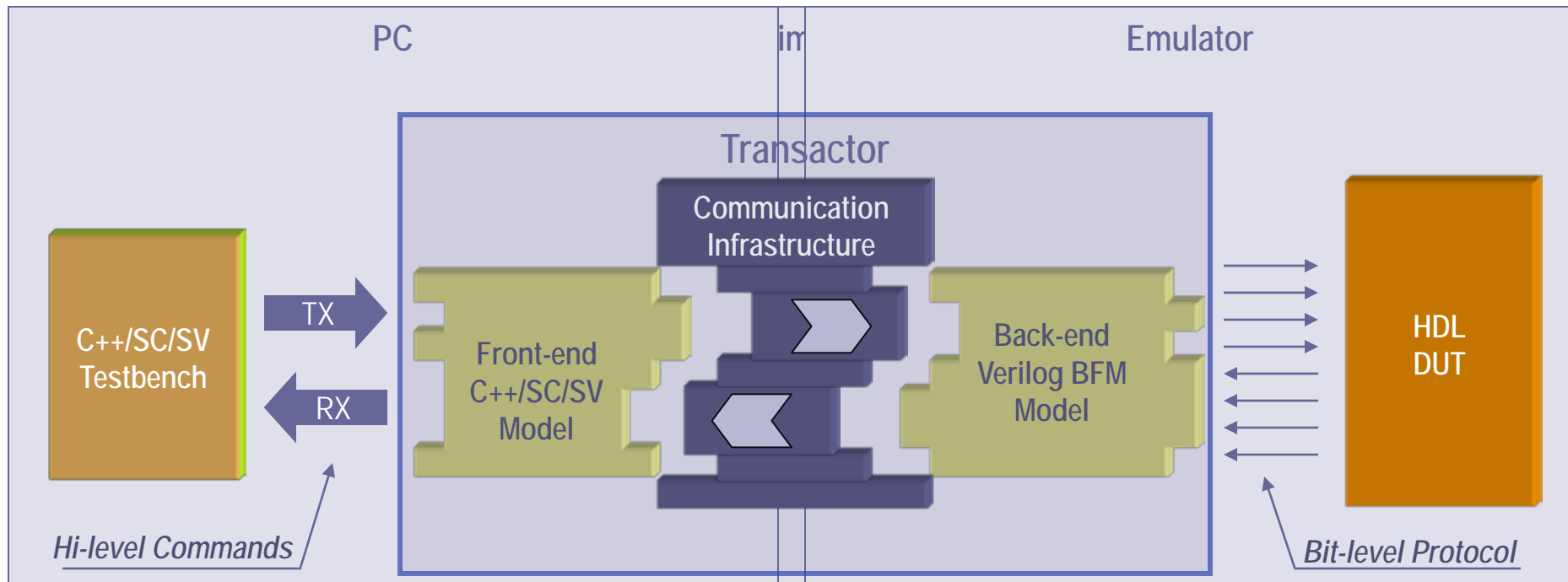
Traditional Deployment of Emulator

- In-Circuit-Emulation (ICE)



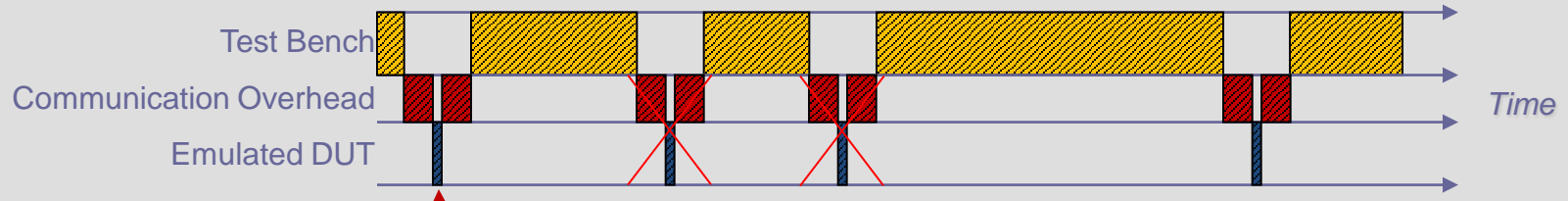
What is a Transactor: Conceptual Description

- A transactor is an interface that implements a given protocol and consists of two parts:
 - Front-end:
 - C++ (SC/SV) model to send/receive high-level commands (transactions) to/from Testbench
 - Not compute-intensive
 - Back-end:
 - Verilog BFM model to convert high-level commands into bit-level protocol
 - Compute-intensive
 - The data exchange between the hardware and the software part of the transactor is done through a communication infrastructure typically described via the SCE-MI Standard



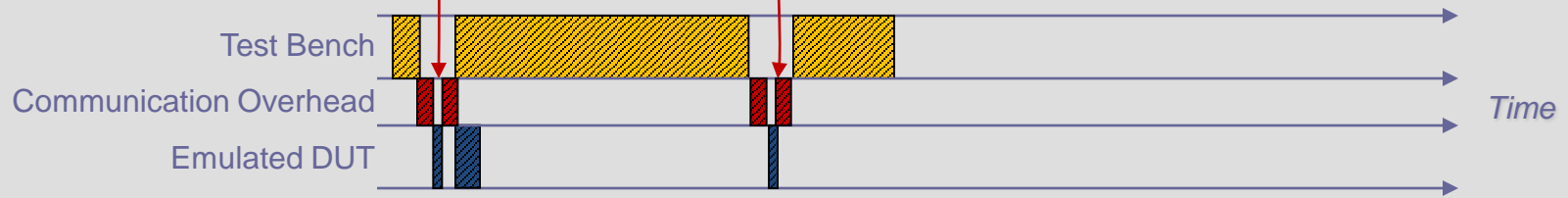
Cycle-Based vs Transaction-based Co-Emulation

Cycle-Based



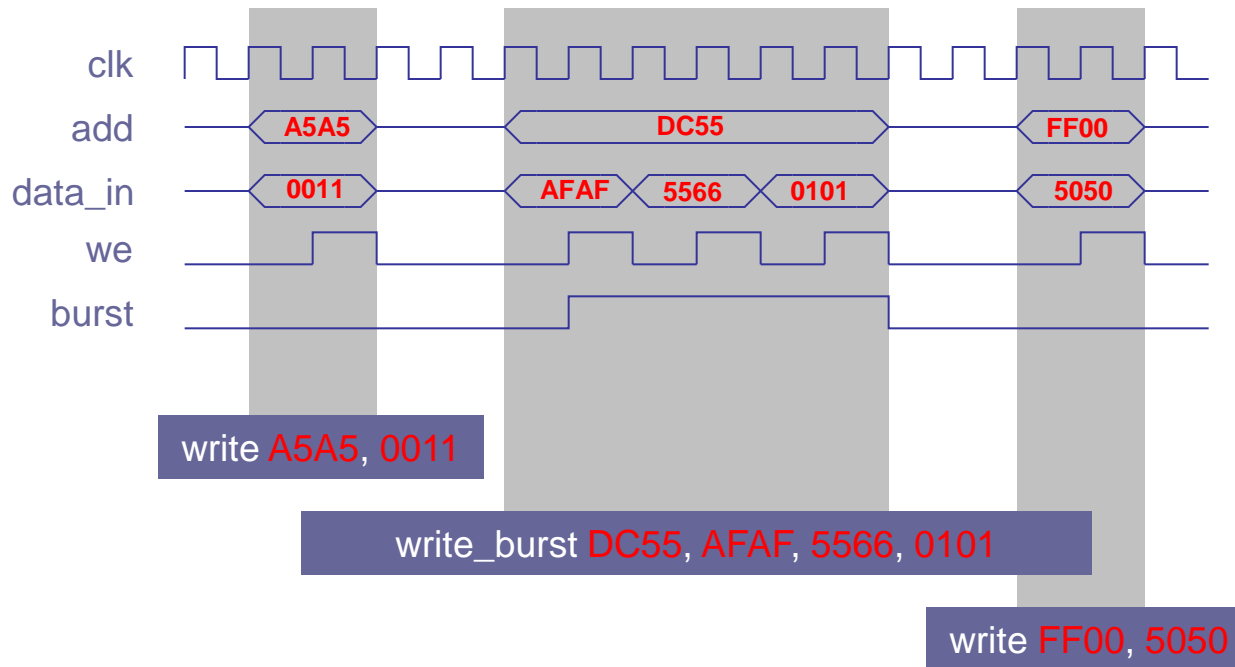
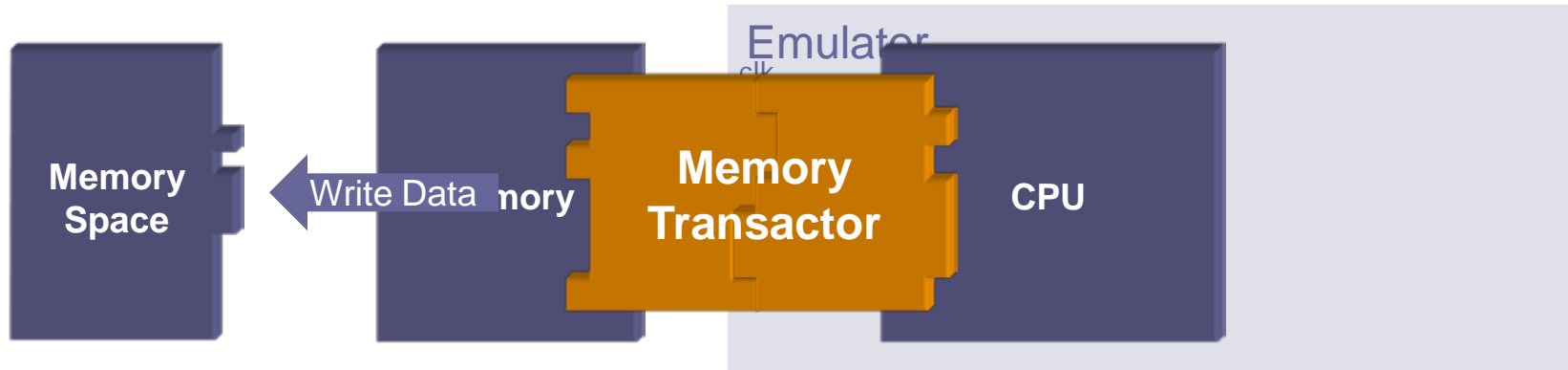
- In cycle-based co-emulation, the typical sequence of operations is:
 - Force DUT inputs – run one cycle – read DUT outputs
 - Testbench and DUT are synchronized on a cycle-by-cycle basis, and communication overhead occurs at each and every cycle

Transaction-Based

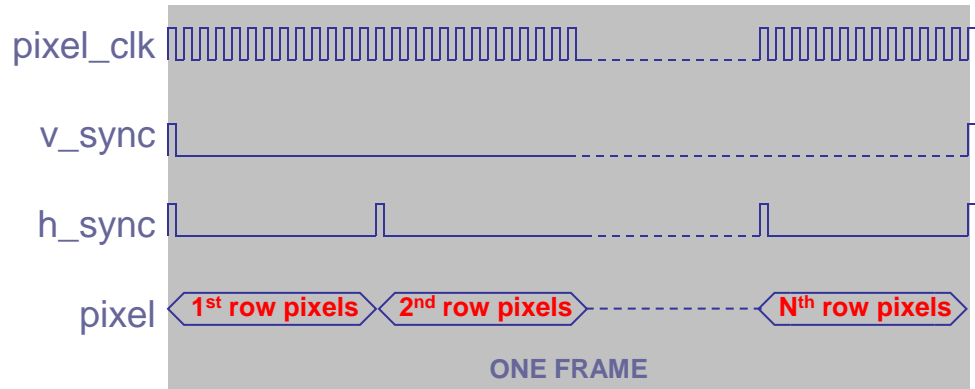
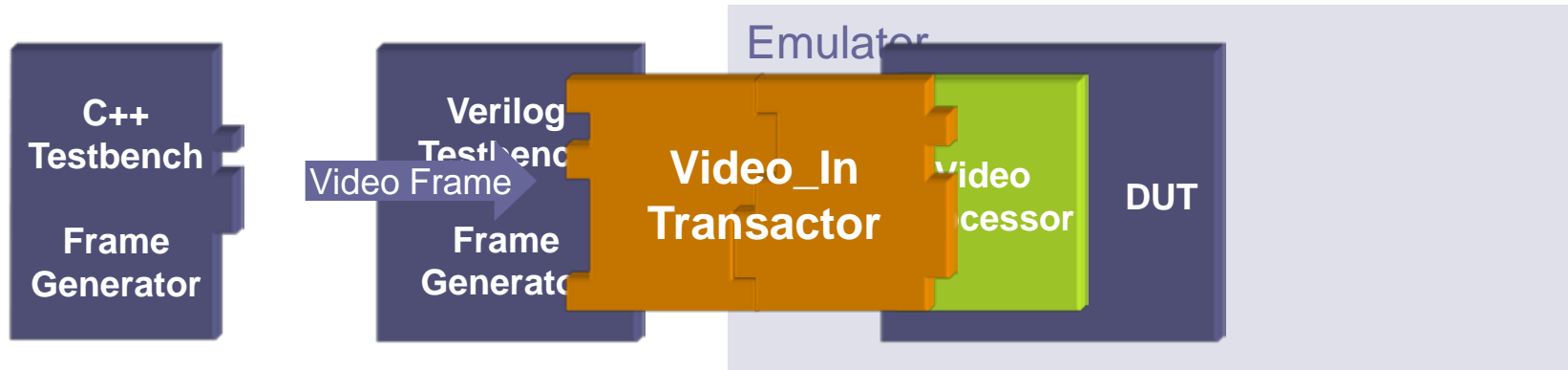


- In transaction-based co-emulation, testbench and DUT are synchronized only when required
 - Test bench and DUT can run in parallel and transactions can be queued
 - The speed improvement over cycle-based can be orders of magnitude faster reaching tens of MHz

Example : Memory Transactor



Example: Video_In Transactor



Cycle-based versus Transaction-based Co-Emulation

Benefits	Cycle-Based Co-Emulation	Transaction-Based Co-Emulation
Initial Investment	<ul style="list-style-type: none"> No transaction based knowledge required 	<ul style="list-style-type: none"> Requires developing hardware transactors
Easier, Faster Test Bench Generation	<ul style="list-style-type: none"> Knowledge of low level "protocol" required for test bench writing 	<ul style="list-style-type: none"> High level specific API knowledge sufficient
	<ul style="list-style-type: none"> Constant investment for each test bench 	<ul style="list-style-type: none"> Initial investment for developing a transactor, but lower investment for further test bench development
	<ul style="list-style-type: none"> Only low level re-usability possible 	<ul style="list-style-type: none"> High-level reusability possible: transactors are written once and used many times
Faster Execution	<ul style="list-style-type: none"> Activity scheduled on a cycle basis even without any event to process 	<ul style="list-style-type: none"> Activity scheduled on a transaction basis, i.e., only when events happen
	<ul style="list-style-type: none"> Serial run of test bench and emulator 	<ul style="list-style-type: none"> Parallel run of test bench and emulator (higher speed, especially for data streaming application and for standard-based designs)

Legend:

Pro

Con

ICE versus Transaction-based Co-Emulation

Benefits	In-Circuit Emulation	Transaction-Based Co-Emulation
Setup Costs & Efforts	<ul style="list-style-type: none"> • Requires connection to target system and speed rate adapters (FIFO) for each interface (HW costs) • Requires Reset Circuitry • Significant HW vulnerabilities (noise, RFM, etc.) 	<ul style="list-style-type: none"> • No H/W costs, only S/W development costs • No need for speed rate adapters
Flexibility	<ul style="list-style-type: none"> • New design requires new adapter board • If I/O protocol not well defined, no applicable • Only one test case per protocol • Board completion is in critical path: emulation cannot begin until board is ready • Limited reuse: new design requires new adapter board 	<ul style="list-style-type: none"> • Easier to modify S/W than H/W • Completion of S/W development is not in critical path: can begin with existing TB and emulate parts of design incrementally • Highly reusable, transactors are written once and used many times
Determinism	<ul style="list-style-type: none"> • Non-deterministic behavior compromises and prolongs design debug 	<ul style="list-style-type: none"> • Deterministic behavior makes testing predictable and repeatable and accelerates design debugging
Remote Access	<ul style="list-style-type: none"> • Cannot be accessed remotely without on-site help 	<ul style="list-style-type: none"> • Can be accessed remotely without on-site help

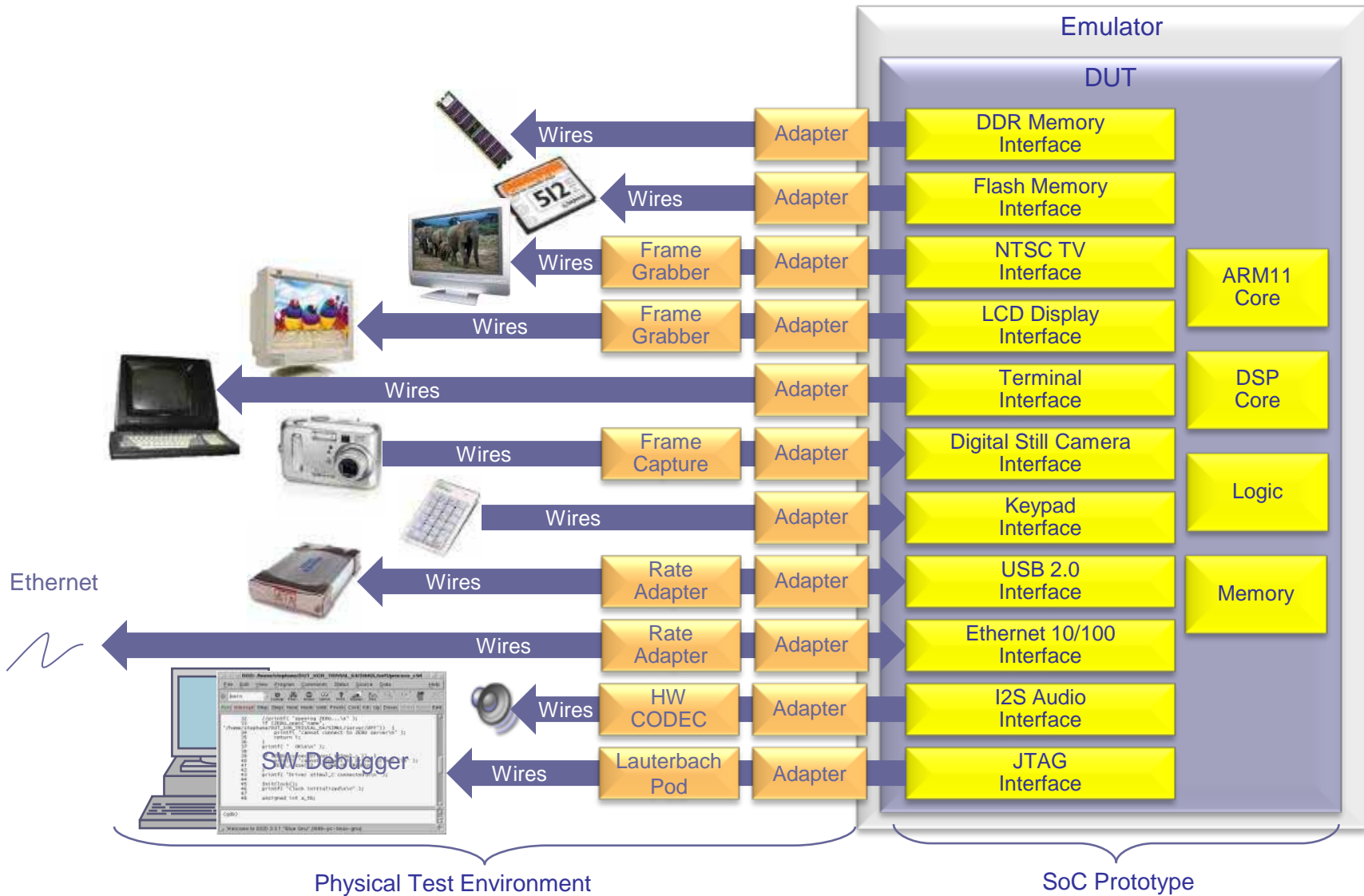
Legend:

Pro

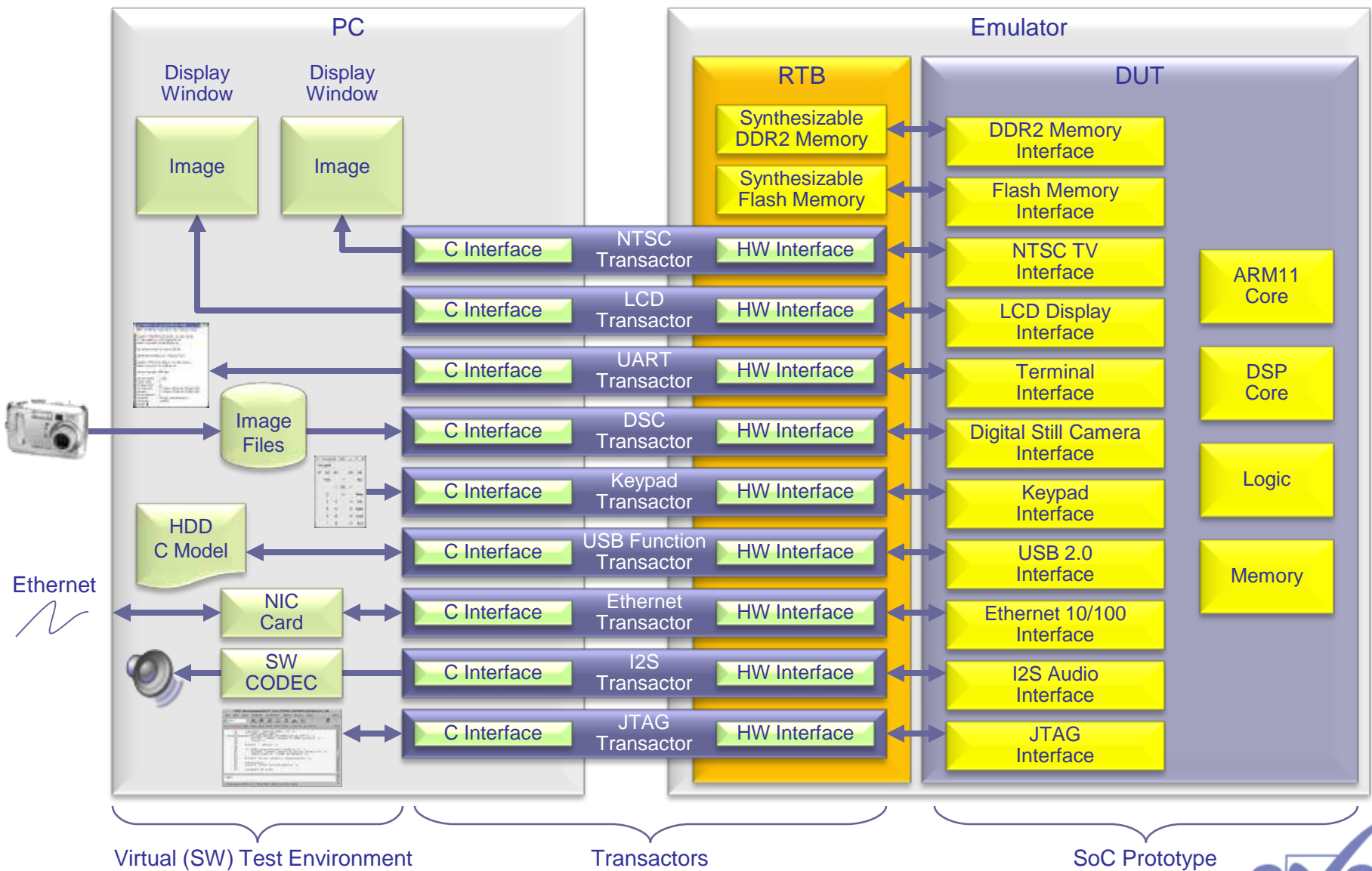
Con



Physical Wireless Platform: The ICE Way

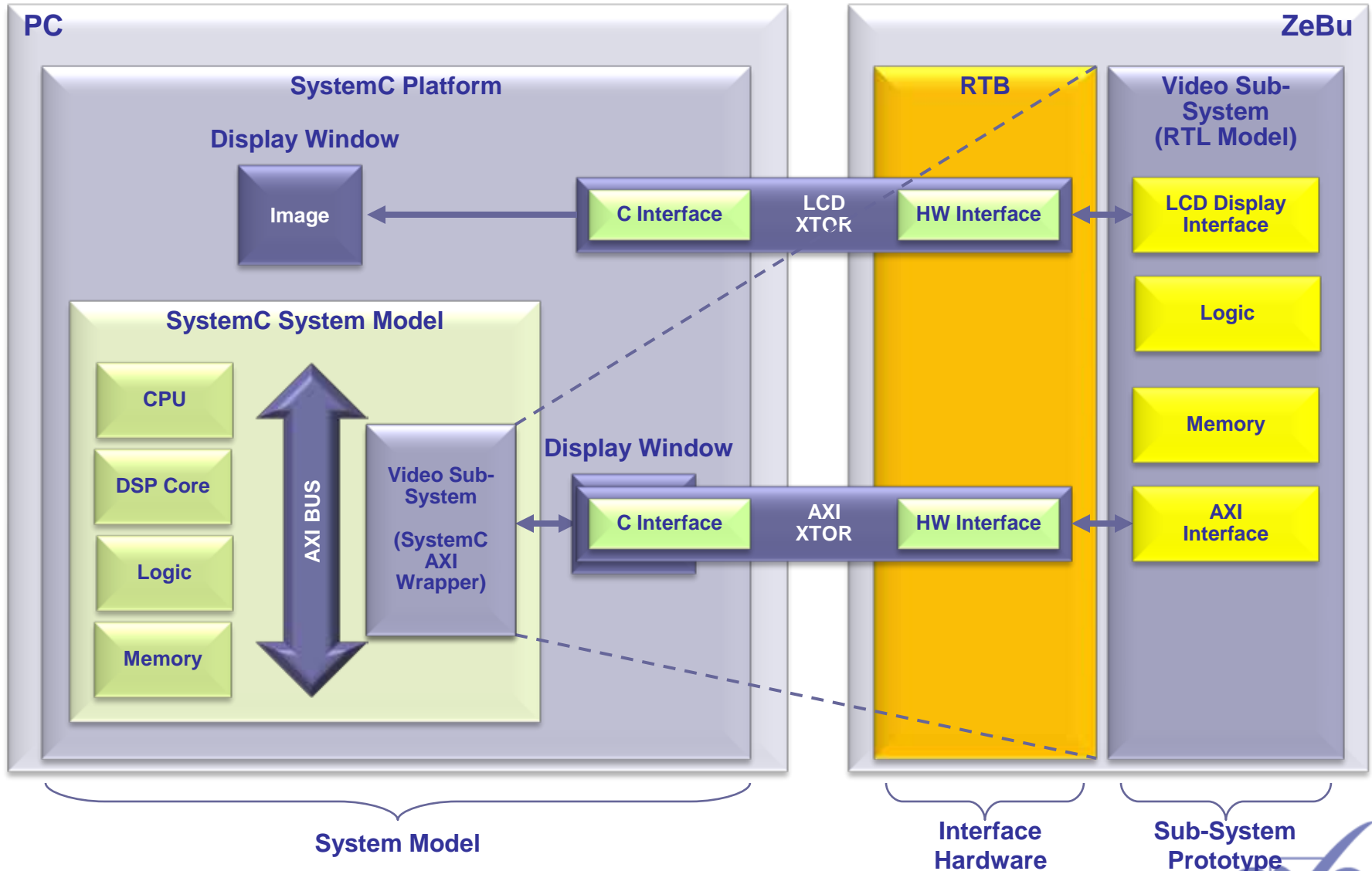


Virtual Wireless Platform: The TRANSACTOR Way



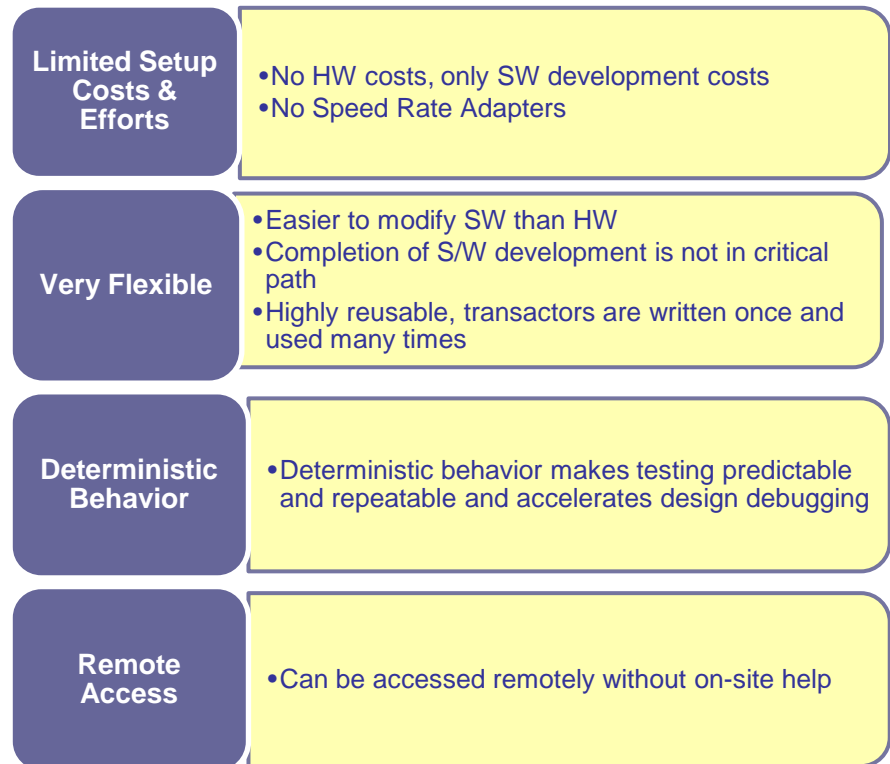
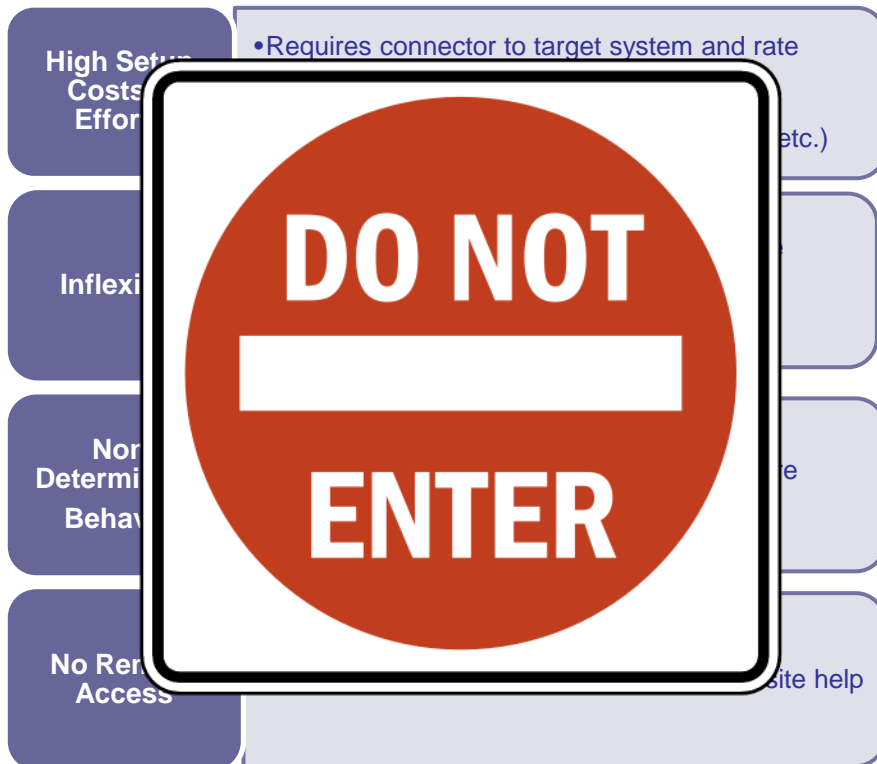
ESL Emulation via TRANSACTORS

Example of Video Sub-System Verification



Conclusion

- Transaction-based co-emulation is vastly superior to cycle-level co-emulation and a viable alternative to ICE: it is a better and faster way to test your design



unEVErsity Connections Program

- Sandra Larrabee: University Connections Program Manager
 - Email: sandra@eve-usa.com
 - Phone: (408) 691-0463
 - <http://university.eve-team.com/>

