Introduction to Embedded Systems

Outline

- Embedded Systems
- High Performance Embedded Systems
- Verification and Validation
- Conventional Verification of Embedded Systems
- Verification of Complex Systems
- Conclusion
- Questions and Answers

Introduction to Embedded Systems (1/4)

- An application specific electronic sub-system which is completely encapsulated by the main system it belongs to.

- The main systems can range from household appliances, home automation, consumer electronics, ATMs, network routers, automobiles, aircrafts, etc.

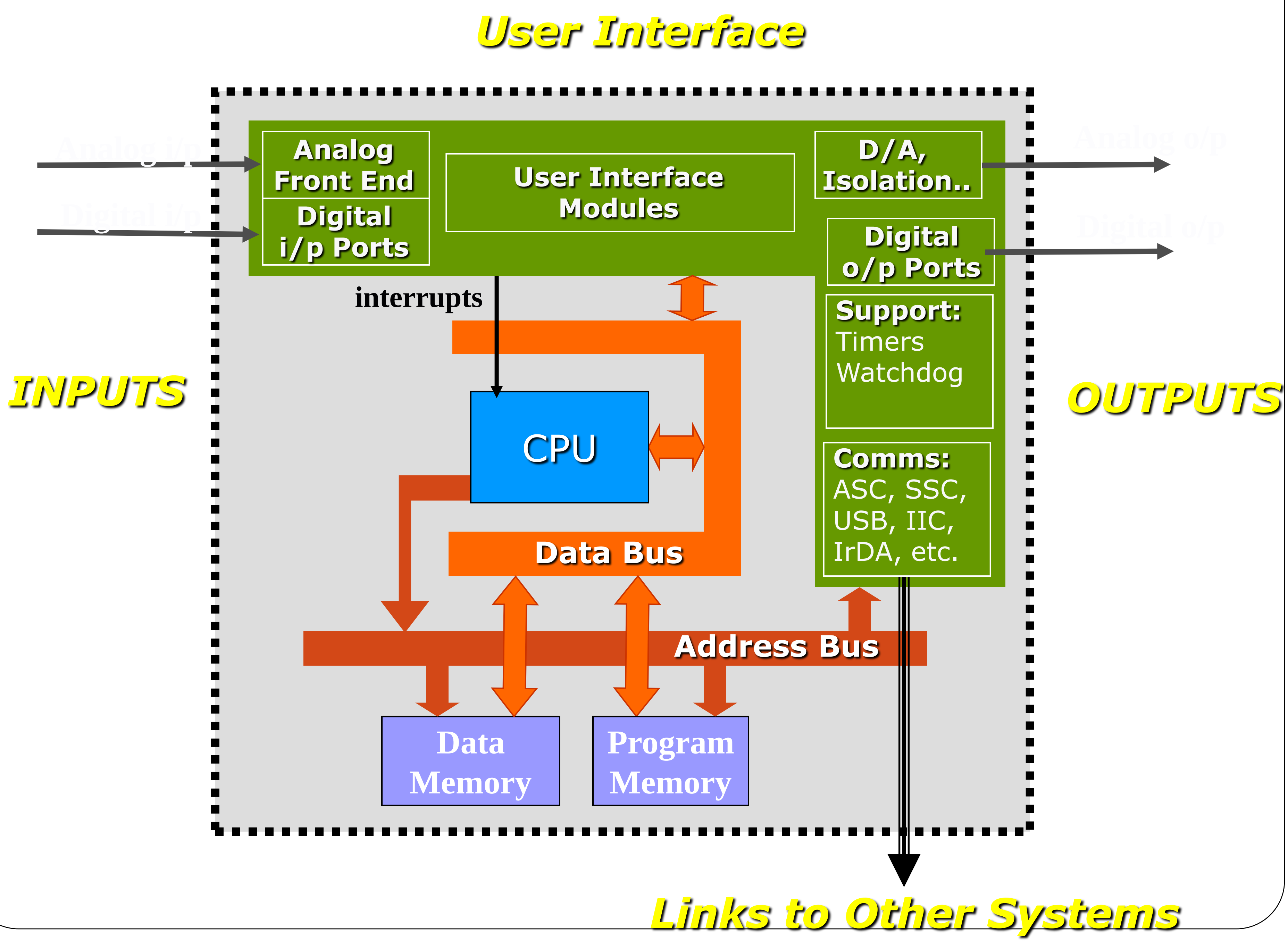
Introduction to Embedded Systems (2/4)

- Designed for some specific tasks
- Subjected to real time performance constraints that must be met
- Feature tightly integrated combinations of hardware and software

Introduction to Embedded Systems (3/4)

- Typical embedded software components:

Embedded Application Code
Device Drivers
A Real-Time Operating System (RTOS)
Hardware abstraction layer(s)
System initialization routines


User Interface
INPUTS
Analog Front End
Digital i/p Ports
User Interface Modules
D/A, Isolation..
Digital o/p Ports
Support:
Timers
Watchdog
Comms:
ASC, SSC,
USB, IIC,
IrDA, etc.
interrupts
CPU
Data Bus
Address Bus
Data Memory
Program Memory
OUTPUTS
Links to Other Systems

Outline

- √ Embedded Systems
- High Performance Embedded Systems
- Verification and Validation
- Conventional Verification of Embedded Systems
- Verification of Complex Systems
- Conclusion
- Questions and Answers

High Performance Embedded Systems (1/10)

- Massive computational resources with requirements of
  - Small size
  - Low Weight
  - Very low power consumption.
- Need to employ innovative, advanced system architectures

High Performance Embedded Systems (2/10)

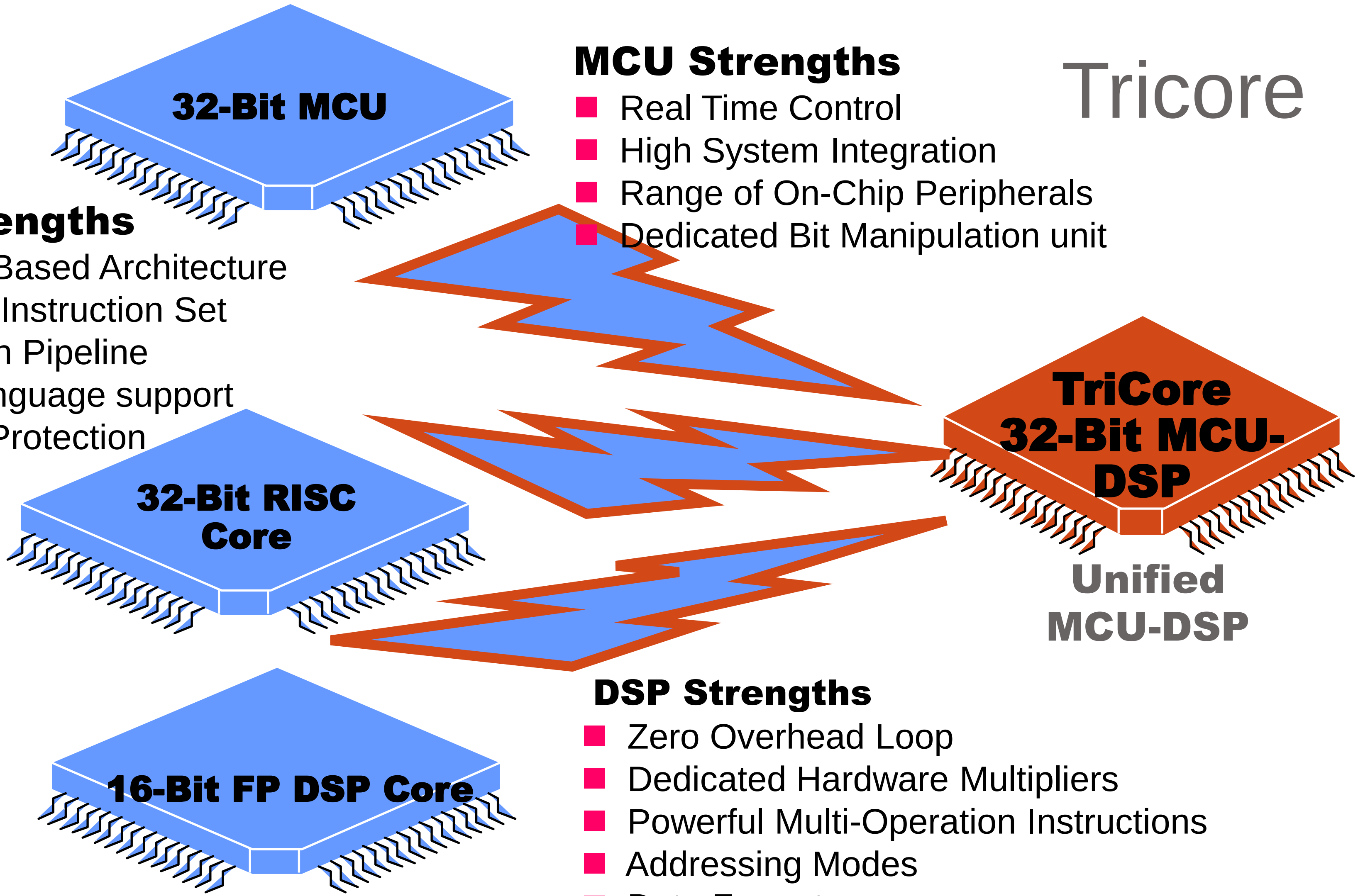
- Architectures typically feature
  - Multiple processor cores
  - Tiered memory structures with multi-level memory caching
  - Multi-layer bus structures.
  - Super-pipelining and/or super-scaling

High Performance Embedded Systems (3/10)

- The current state-of-the-art:
  Multiple computational and data-processing engines, memory, and peripherals, all constructed on a single silicon chip called a System-on-Chip (SoC).
- Designs to feature multiple general-purpose central processing unit (CPU) cores as well as special-purpose digital signal processor (DSP) cores

High Performance Embedded Systems (4/10)

Tricore



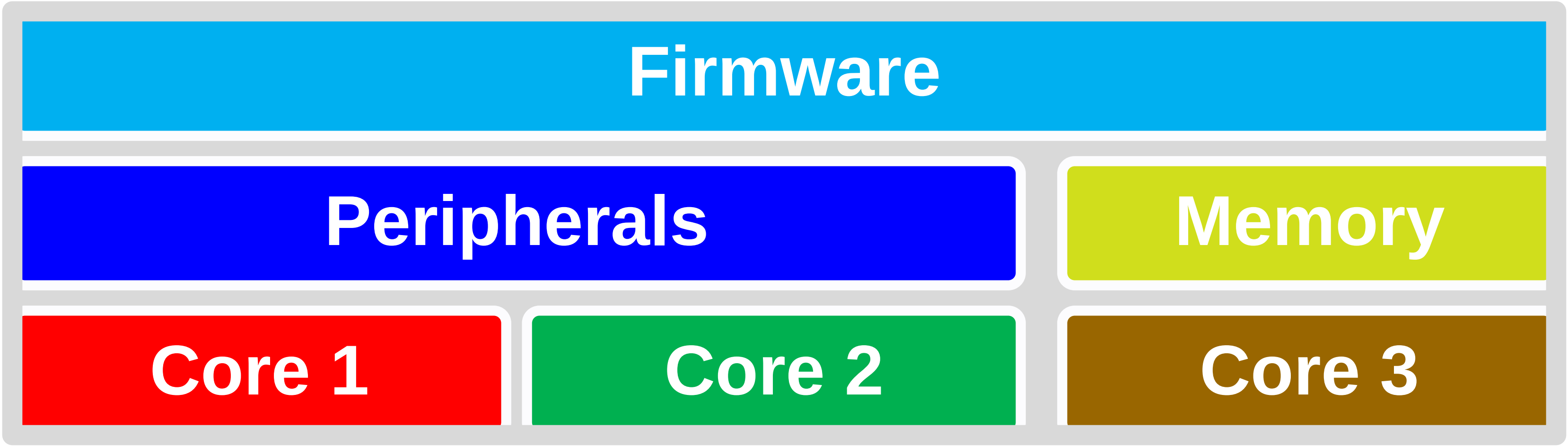
MCU Strengths
- Real Time Control
- High System Integration
- Range of On-Chip Peripherals
- Dedicated Bit Manipulation unit

RISC Strengths
- Register Based Architecture
- Reduced Instruction Set
- Instruction Pipeline
- C/C++ language support
- Memory Protection

DSP Strengths
- Zero Overhead Loop
- Dedicated Hardware Multipliers
- Powerful Multi-Operation Instructions
- Addressing Modes
- Data Formats

High Performance Embedded Systems (5/10)

- Embedded designs to include multiple general-purpose central processing unit (CPU) cores as well as special-purpose digital signal processor (DSP) cores



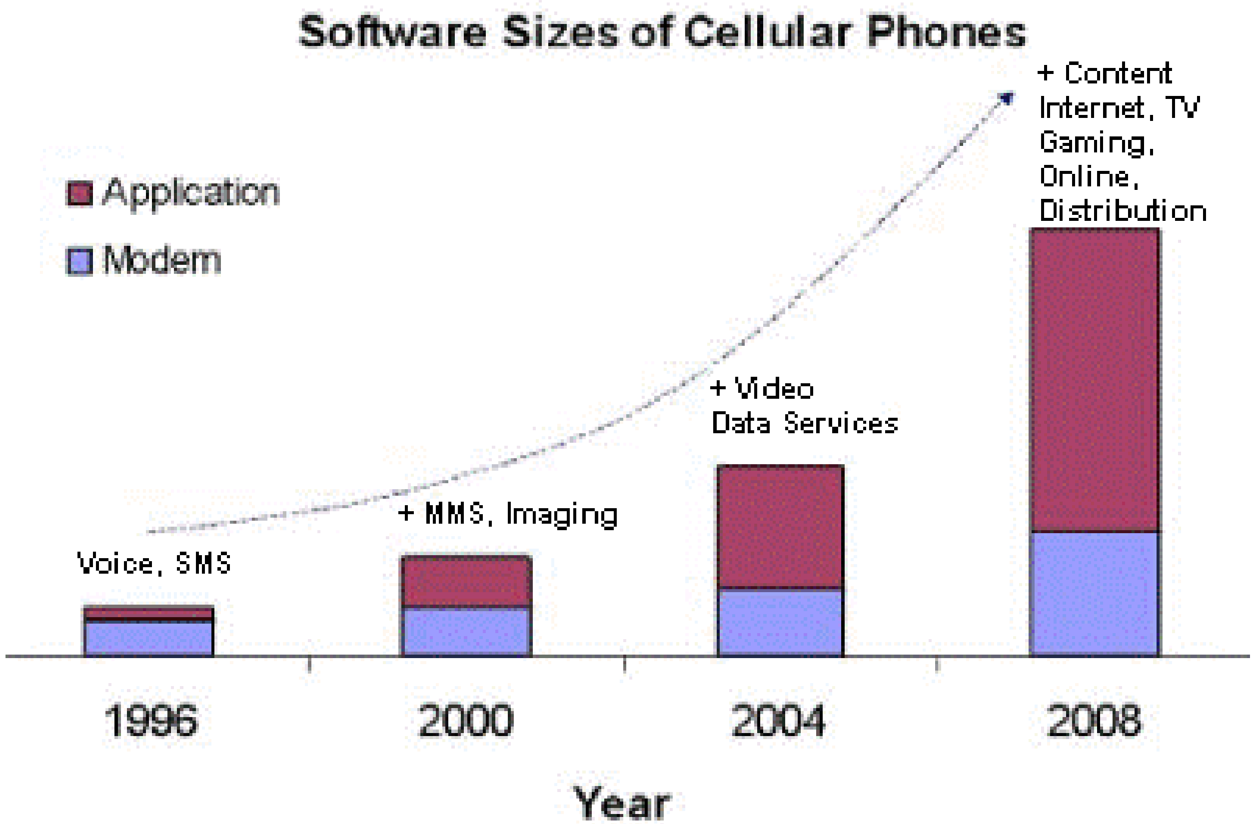
High Performance Embedded Systems (6/10)

- **Multilayer Bus Structures**
  - CPU and DSP cores can have separate buses for control, instructions, and data,
  - DMA buses along with one or more dedicated peripheral buses.
  - Both the CPUs and DSPs can have tightly-coupled memory buses, external memory buses, and shared memory buses.

High Performance Embedded Systems (7/10)

- **Increasing software content**
  - The software content of embedded systems is increasing at a phenomenal rate
  - **software development** and **test** often dominate the costs, timelines, and risks associated with today's embedded system designs.

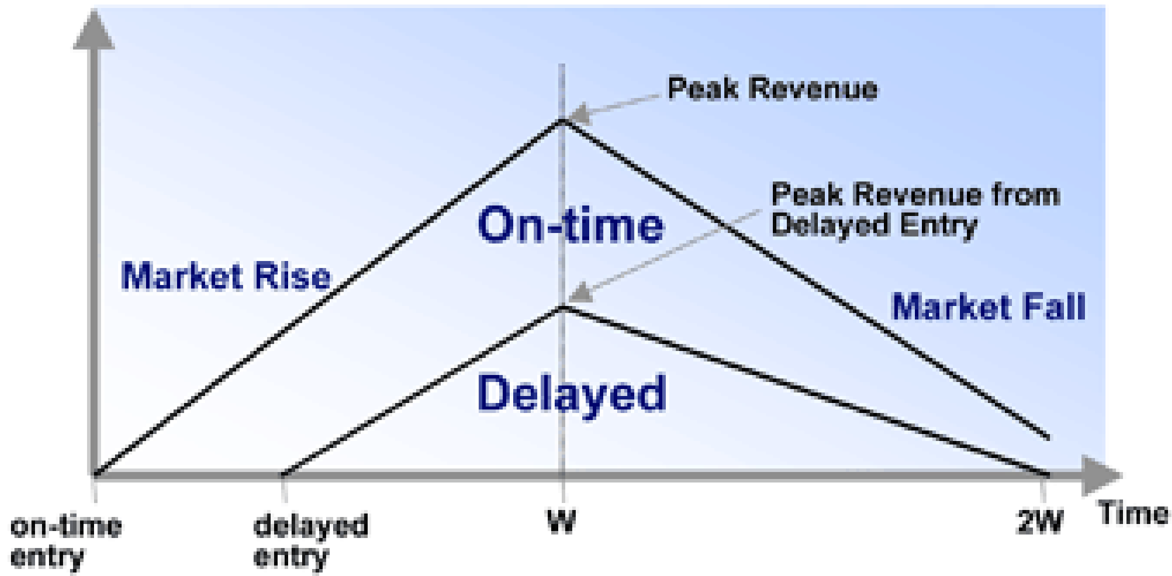
High Performance Embedded Systems (8/10)



High Performance Embedded Systems (9/10)

- **Decreasing design cycles**
  - Market windows are continually narrowing
  - Competition gets more and more aggressive
  - Consumer electronics markets are extremely sensitive to time-to-market pressures

High Performance Embedded Systems (10/10)



Source: VCD, Leading Semi-conductor and Systems companies, Gartner Dataquest

Outline

- √ Embedded Systems
- √ High Performance Embedded Systems
- Verification and Validation
- Conventional Verification of Embedded Systems
- Verification of Complex Systems
- Conclusion
- Questions and Answers

Verification and Validation (1/7)

- **What is Verification and Validation?**
  - **Verification**

    Verification confirms that work products properly reflect the requirements specified for them. In other words, verification ensures that ‘the product has been built right’.

Verification and Validation (2/7)

- **Validation**

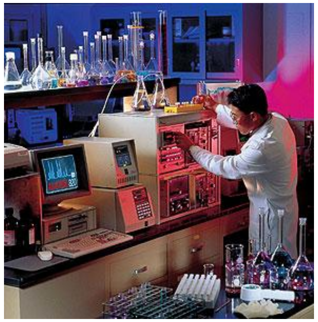
  Validation confirms that the product, as provided, will fulfill its intended use. In other words, validation ensures that 'you built the right thing'".

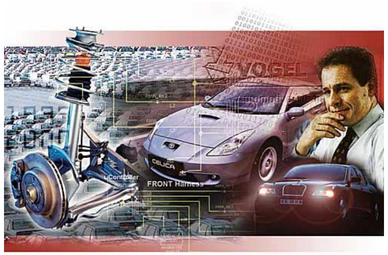
Verification and Validation (3/7)

- **Why Verification and Validation?**
  - Business considerations
    - Legal
    - Refutation
    - Warranty / Recall
  - Regulatory issues
    - FDA
    - FAA
    - DoD

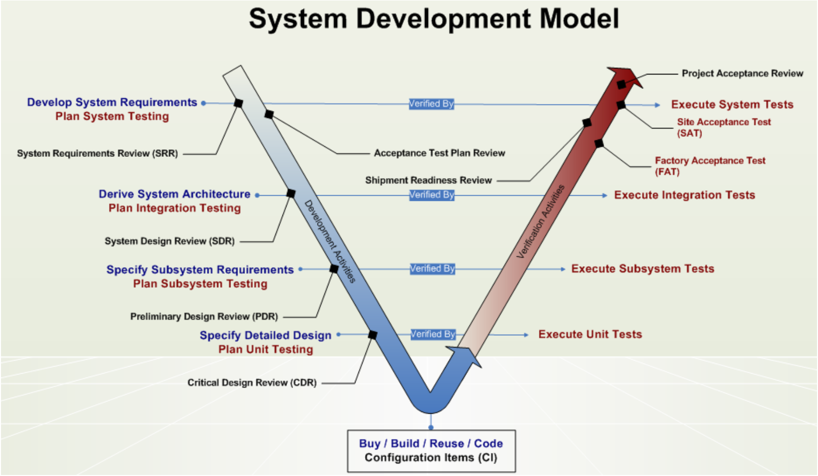
Verification and Validation (4/7)

- Safety considerations
  - Life sciences
  - Mission critical
  - Automotive examples:
    - Drive by wire
      - Electronic throttle control
      - Electronic steering
    - ABS
    - Airbag Systems





Verification and Validation (5/7)

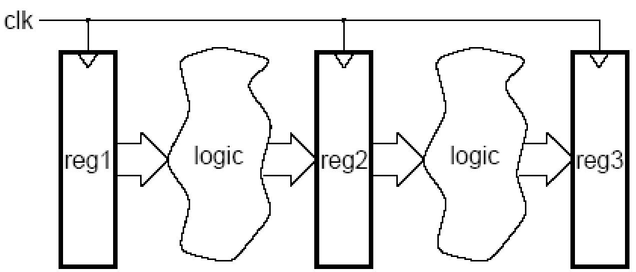


Verification and Validation (6/7)

Abstraction Levels of Design Under Verification

- Behavioral Model
  - Example: `c <= a * b`
  - May not include timing information
  - Verification examines the basic operation and interactions among the systems' components

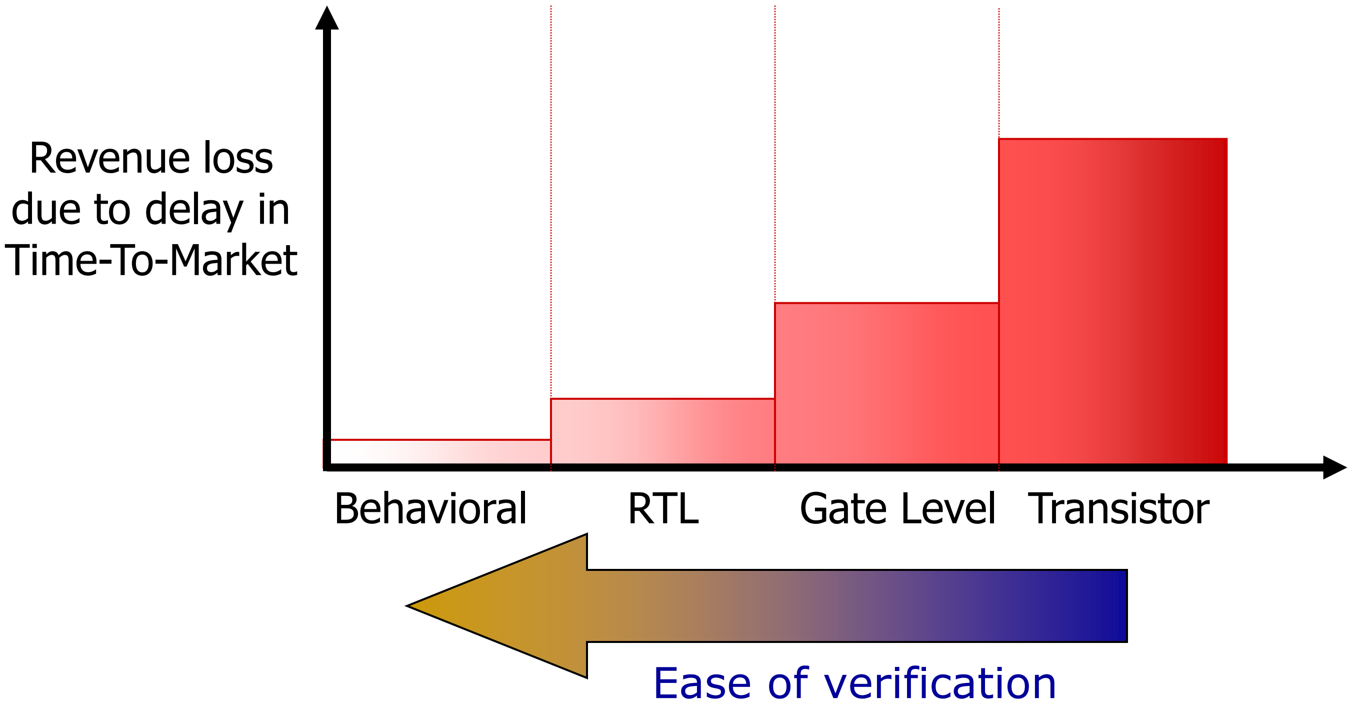
- RTL (Register-Transfer-Level) Model
  - VHDL/Verilog commonly used to model RTL
  - Accurate cycle-level information (no propagation delays)
  - Verification of exact cycle behavior



- Gate-Level Model
  - Specifies each individual logic element and their interconnections
  - Verification at this level is time-consuming but necessary for clock boundaries and reset conditions

Verification and Validation (7/7)

Importance of Verification in Early Design Stages



Remove as many bugs as possible in early designs stages

Source: Verification Methodology Manual, 2000 - TransEDA

Outline

- √ Embedded Systems
- √ High Performance Embedded Systems
- √ Verification and Validation
- Conventional Verification of Embedded Systems
- Verification of Complex Systems
- Conclusion
- Questions and Answers

Conventional Verification of Embedded Systems (1/13)

- Determine the overall system architecture
- Design System Hardware
- Construct hardware prototype
- Install and test OS and/or middleware
- Develop, port, integrate, and debug embedded software

Conventional Verification of Embedded Systems (2/13)

- Conventional verification drawback (mainly due to shorter design cycles)
  - SoC to be fabricated before developing the software
  - Having to wait for the implementation-level representation of the design (specified RTL) to become available before developing the embedded software.

Conventional Verification of Embedded Systems (3/13)

- **Physical Prototypes as primary verification mechanism**
  - Typically involves a circuit board and the SoC in the form of working silicon.
    - The hardware portion of the design is now almost 100 percent tied down
    - Not much useful in the context of exploring and evaluating alternative architectures

Conventional Verification of Embedded Systems (4/13)

- Important hardware/software tradeoffs can’t be made before the design partitioning is locked down and the chips are manufactured
  - System design must be largely based on experience and intuition, as opposed to hard data.
  - Unacceptable in today's complex algorithms, multi-core systems, tiered memory systems, and multi-layered bus structures.

Conventional Verification of Embedded Systems (5/13)

- **Hardware acceleration and emulation as verification mechanism**
  - These typically involve arrays of field-programmable gate arrays (FPGAs) or processors.
  - These solutions accept RTL representations of the design and translate them into an equivalent suitable for hardware acceleration.
  - The verification can get very costly

Conventional Verification of Embedded Systems (6/13)

- Issues in multi-processor designs
- Emulators also have problems with limited visibility into the design
- Software development cannot commence until a long way into the design cycle. (The hardware design is largely established → limitations with regard to exploring and evaluating alternative architectures)

Conventional Verification of Embedded Systems (7/13)

- **RTL-based simulation as verification mechanism**
  - An RTL simulation solution requires RTL representations of the hardware → Delays in meaningful software development until the RTL becomes available
  - It simply isn't possible to use software simulation to determine how well the architecture performs on real software workloads

Conventional Verification of Embedded Systems (8/13)

- A software simulation running on a on very high-end (and correspondingly expensive) machine would hardly achieve equivalent simulation speeds of more than a few Hz
  - That is, a few cycles of embedded system clock for each second in real time
  - Detailed simulations can be performed on only small portions of the software.

Conventional Verification of Embedded Systems (9/13)

- **ISS-based simulation as verification mechanism**
  - Verify and debug chips using software models that can execute the same binary code as the actual processors
  - Limitations:
    - Only processor cores can be modeled
    - Accuracy is compromised for high performance verification (typically not cycle accurate)
    - Lack of synchronization support for multi-processor based systems

Outline

- √ Embedded Systems
- √ High Performance Embedded Systems
- √ Verification and Validation
- √ Conventional Verification of Embedded Systems
- Verification of Complex Systems
- Conclusion
- Questions and Answers

Verification of Complex Systems (1/15)

- System-On-Chip (SoC) designs increasingly become the driving force of a number of modern electronics systems
- A number of key technologies integrate together in forming the highly complex embedded platform
- Verification need to account for integration of a number of different IPs into new designs, the coupling of embedded software into designs, and the verification flow from core to the design, etc.

Verification of Complex Systems (2/15)

- IP Core Verification and System Level Verification both need to be addressed adequately
- On top of structural complexities, further bottlenecks are introduced by:
  - Time to market pressures
  - Increasing software content
  - Other stringent design constraints such as size, weight, low power levels, etc.

Verification of Complex Systems (3/15)

- The system level design strategies should be considered together with the complex task of verification
- Hardware , first, then software, is no longer a viable theme
- Appropriate verification strategies need to be employed from the outset to minimize downstream defects including SoC re-spins
- Concurrent hardware and software development would mandatory.

Verification of Complex Systems (4/15)

- SoC architects to employ a broad system level design strategy that will allow:
  - Explore and evaluate system level architectural choices
  - Concurrent hardware-software design
  - Easily evaluate and integrate a number of different technologies
  - Adequate verification at every level of the design cycle

Verification of Complex Systems (5/15)

- Carry out an architecture level power analysis
- Drive requirements for executable specifications
- Provide visibility into designs
- Easily handle regression testing

How do we achieve all this with such complexity?

Verification of Complex Systems (6/15)

- The answer to all above is to employ a Unified System Model without committing to any pre-conceived hardware / software partitioning.
- This will be a type of electronic systems level (ESL) prototype
- The form of these models can be anywhere from a cycle-accurate RTL model and to a time-efficient ISS model, or a hybrid

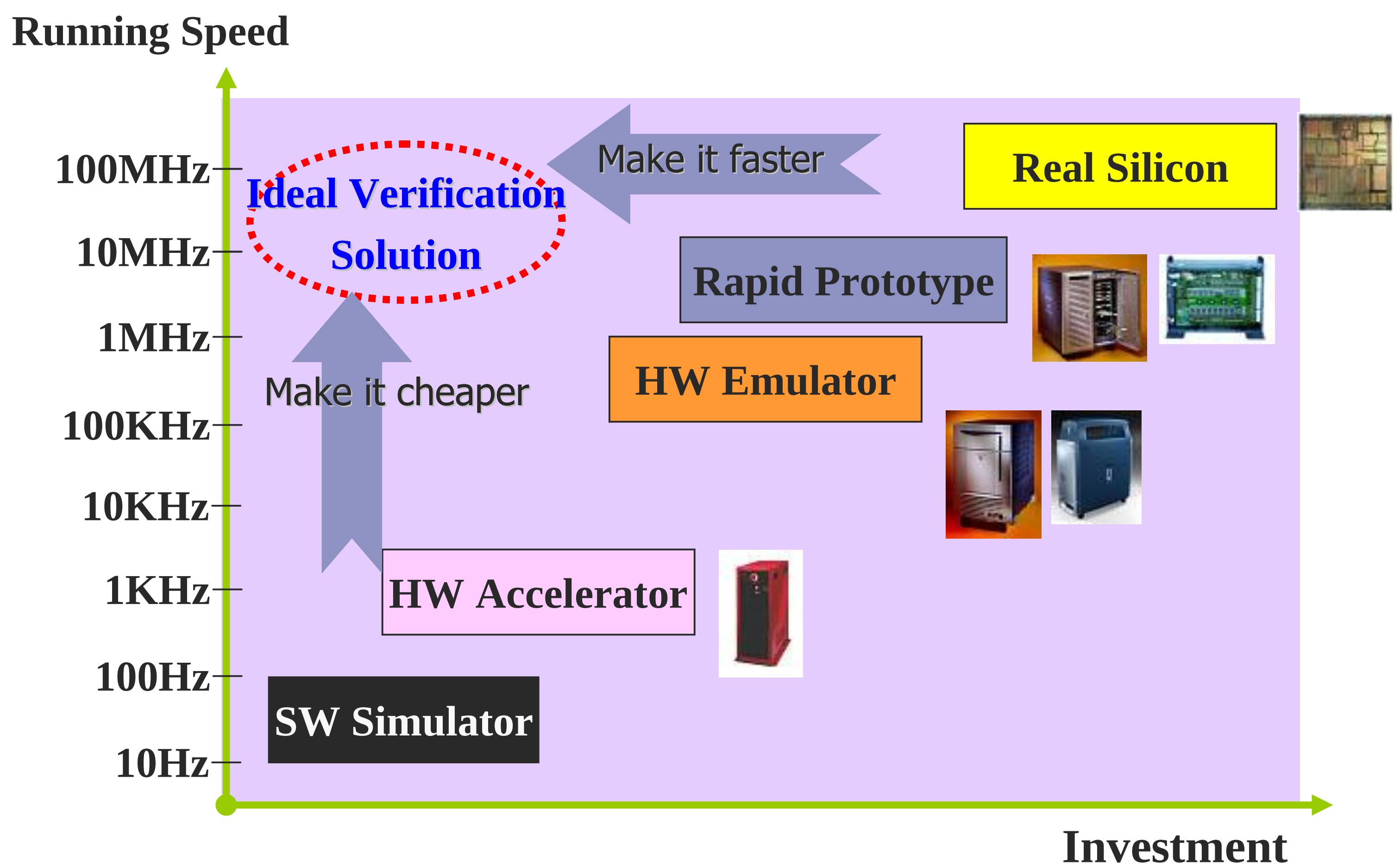
Verification of Complex Systems (7/15)

Model style	Cycle based	Internally timed	Bus-interface timed	Performance range	Uses
Instruction set accurate (ISA)	Y, Instruction cycle	N	N	10 - 100 MCPS	Software development
Cycle approximate (CX)	Y, Clock cycle	Cycle count	Y	~ 2 MCPS	Software execution, System development, Performance profiling
Cycle accurate (CA)	Y, Clock cycle	Pipeline accurate	Y	100s kCPS	Software execution, System development, Performance profiling

Table 1. Model styles.

Source: "**Mixed-Abstraction Virtual System Prototypes Close SOC Design Gaps",** Carbon Design Systems, Inc.

Verification of Complex Systems (8/15)



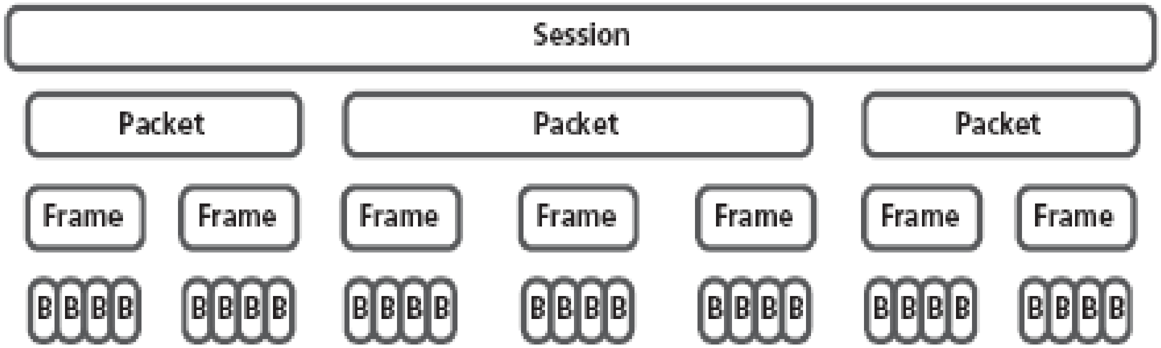
Verification of Complex Systems (9/15)

- The basis for a unified system model is Transaction Level Modeling (TLM).

- Transactions
  - Basic representation for exchange of information between two blocks
  - Improve efficiency and performance of verification by raising the level of abstractions from the signal level
  - Can be as simple as a single data write operation or linked together to form a complex IP packet transfer

Verification of Complex Systems (10/15)

Transactions:

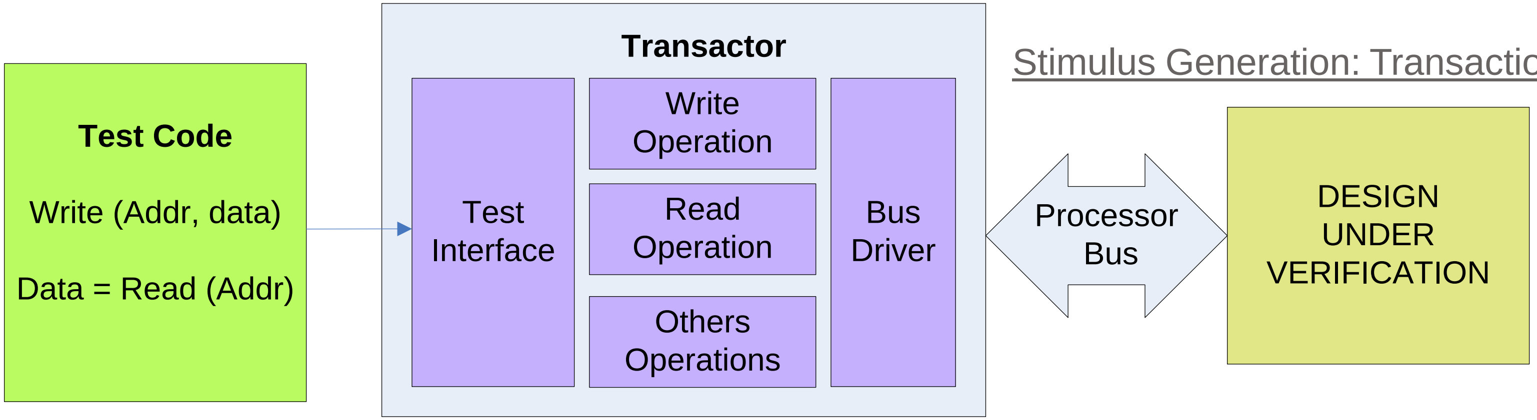


Level	Data unit	Operations	Fields
Interface	Byte	Send, receive, gap	Bits
Unit	Frame	Assemble, segment, address, switch	Preamble, data, FCS
Feature	Packet	Encapsule, retry, ack, route	Header, address, data
Application	Session	Initiate, transmit, complete	Stream

Source: Cadence white paper, “The Unified Verification Methodology”

Verification of Complex Systems (11/15)

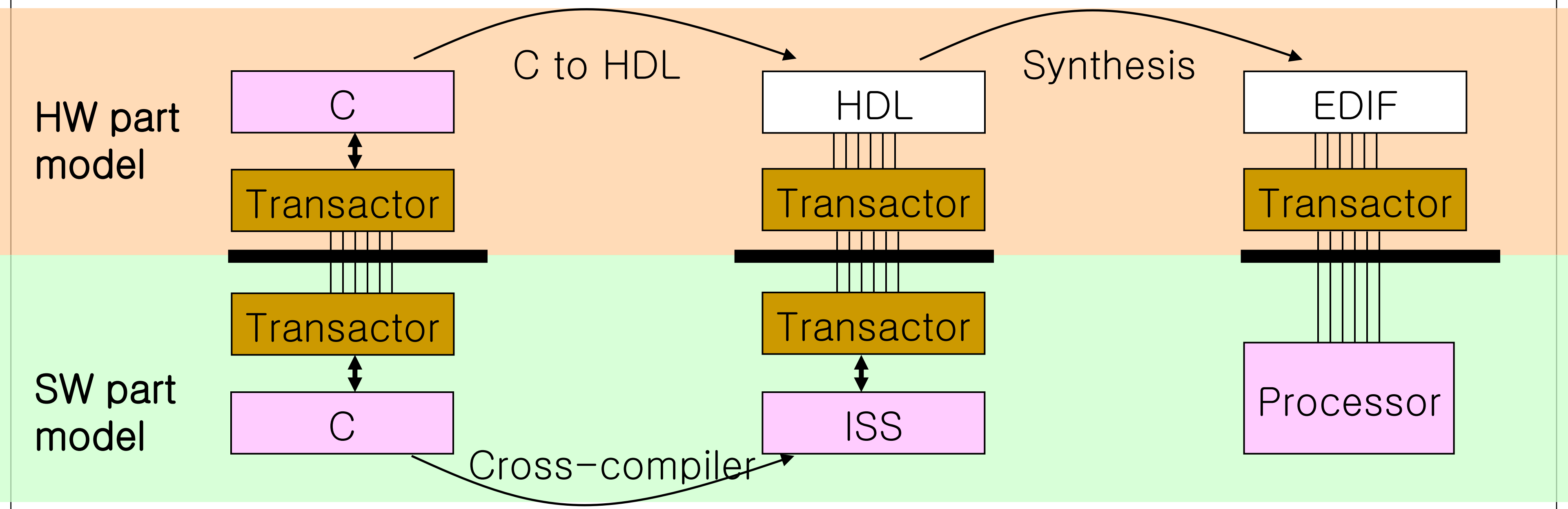
- Transactor provides a level of abstraction between the pins of the model and the test code
  - Encapsulation: Test code does not need knowledge about the bus protocols
  - Abstraction: Allows test to be written in an abstract fashion that specifies the required transactions, instead of the operation execution details
  - Re-use: Transactor provides a standard set of routines that the test can call
  - Modularity: Verification environment can be built from a set of parts



Stimulus Generation: Transactions

Verification of Complex Systems (12/15)

- Support functional design and verification at various abstraction levels
- Advantages
  - Enhance reusability in the test-benches
  - Improve debugging and coverage analysis



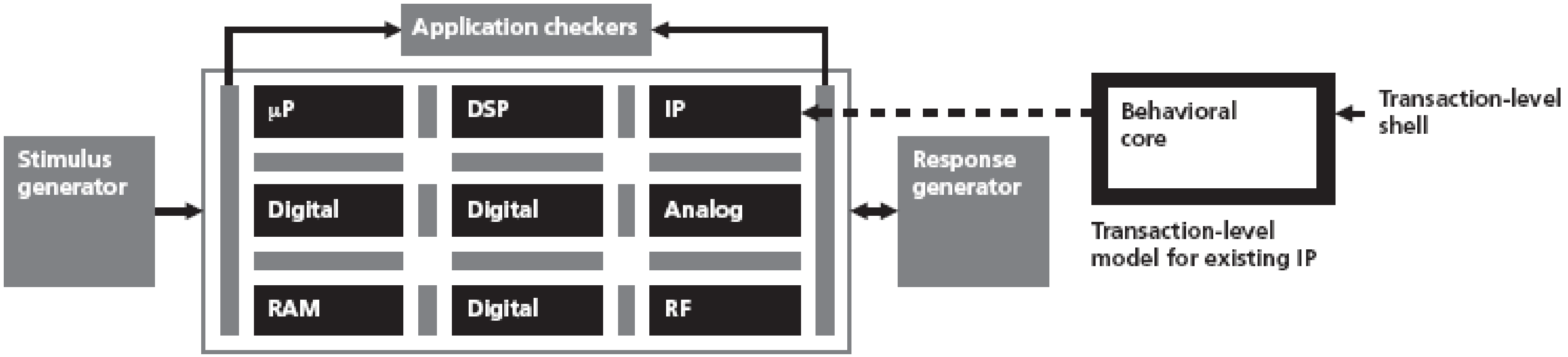
Transaction Level Models (TLM)

Source: Chong-Min Kyung, “Current Status and Challenges of SoC Verification for Embedded Systems Market”, IEEE International SOC Conference, 2003

Verification of Complex Systems (13/15)

Unified System Model: Functional Prototype

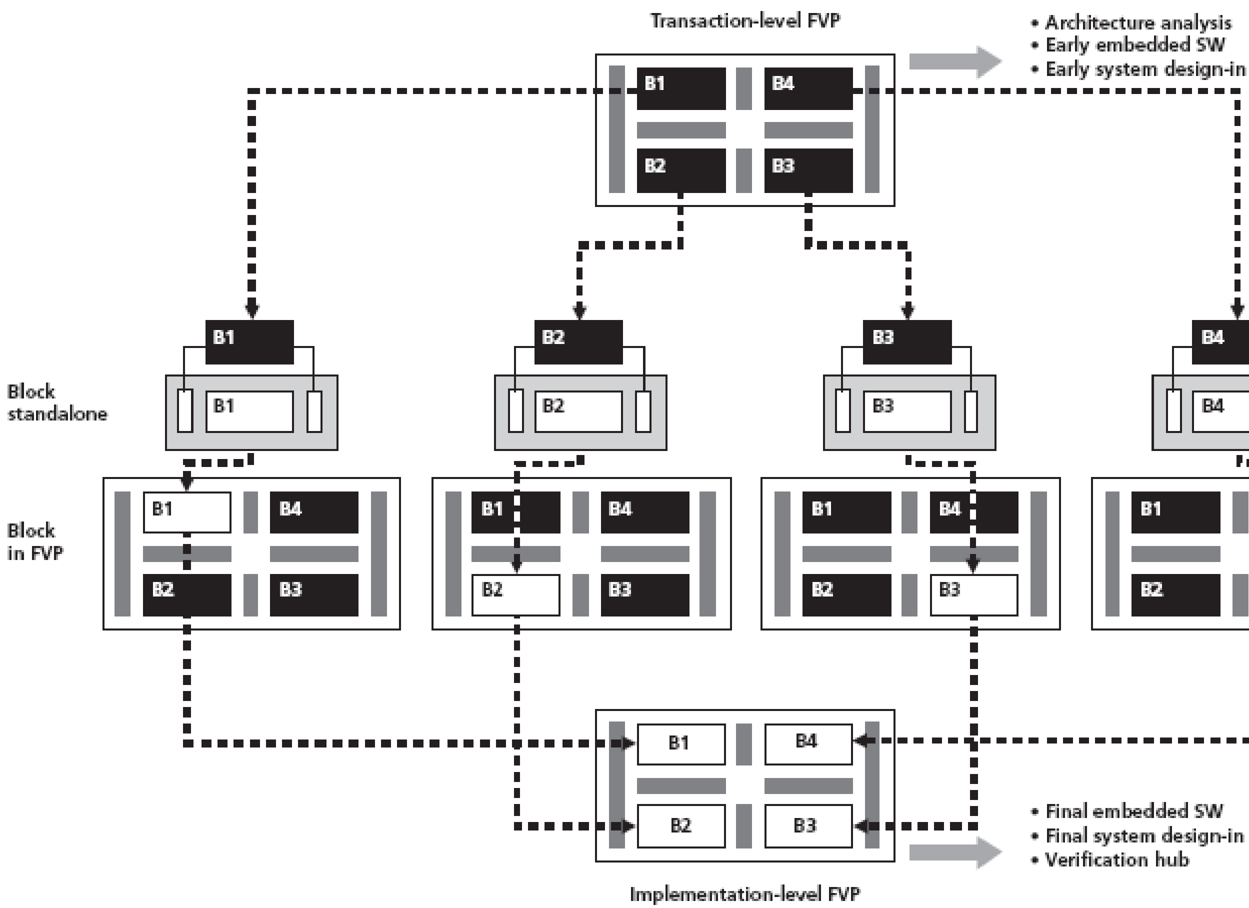
- Unambiguous executable specification
- Golden top-level verification environment and integration vehicle
- Reference for defining transaction coverage requirements
- Model for performing architectural trade-offs
- Early handoff vehicle to system development teams
- Fast executable model for early embedded software development



Source: Cadence white paper, "The Unified Verification Methodology"

Verification of Complex Systems (14/15)

Functional Level to Implementation Level Prototype



Source: Cadence white paper, "The Unified Verification Methodology"

Verification of Complex Systems (15/15)

- Unified System Model with the highest desirable abstraction is created early in the design process by the SoC verification team working closely with the architects
- A test suite is included with the Functional Prototype
- Each subsystem has its own TLM (Transaction Level Model) defined at the SoC partition
- Individual subsystem teams proceed to develop the implementation level of the subsystem
- The test suite is run on the FVP as each subsystem implementation is integrated into the FVP
- The process of integration is facilitated by transactors, which translate information between the transaction and signal level
- Once all the transaction-level models are replaced, the implementation level prototype is complete

Outline

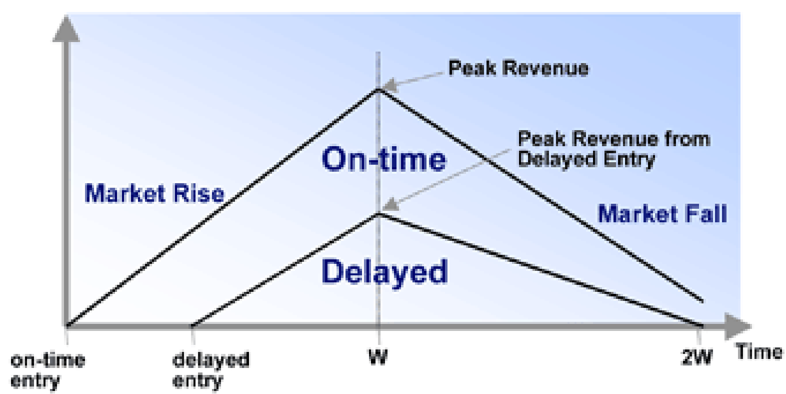
- √ Embedded Systems
- √ High Performance Embedded Systems
- √ Verification and Validation
- √ Conventional Verification of Embedded Systems
- √ Verification of Complex Systems
- Conclusion
- Questions and Answers

Conclusion

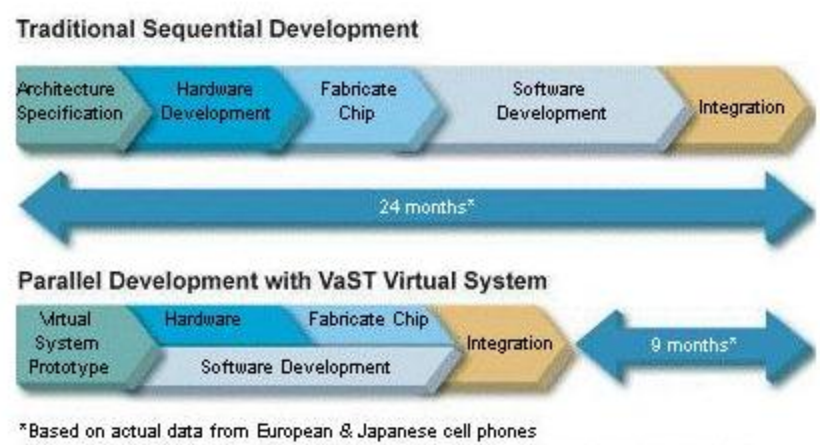
- Embedded systems tend to contain tens of processor cores with multi-layered busses and bus-bridges.
- Hardware and software development a mandatory design methodology.
- Existing embedded system verification strategies do not offer enough sophistication for today's complex systems.

Conclusion

- TLM based Unified System Models provide a means to carry out design and verification hand in hand while promoting hardware / software co-development.



Source: VCD, Leading Semi-conductor and Systems companies, Gartner Dataquest



Source: DSP Design Line

End of Presentation

Thank you!

Any Questions ?

Introducti… …edded Systems (1/4)

- …usehold …mer electronics, ATMs, network routers, automobiles, aircrafts, etc.

Introduction to Embedded Systems (1/4)

-
- applia… electronics, ATMs, network routers, automobiles, aircrafts, etc.

Introduction to Embedded Systems (1/4)

-

-

el , e omobiles,
aircrafts, etc.

Introduction to Embedded Systems (1/4)

-

- mer electronics, ATMs, net utomobiles, aircrafts, etc.

Introduction to Embedded Systems (1/4)

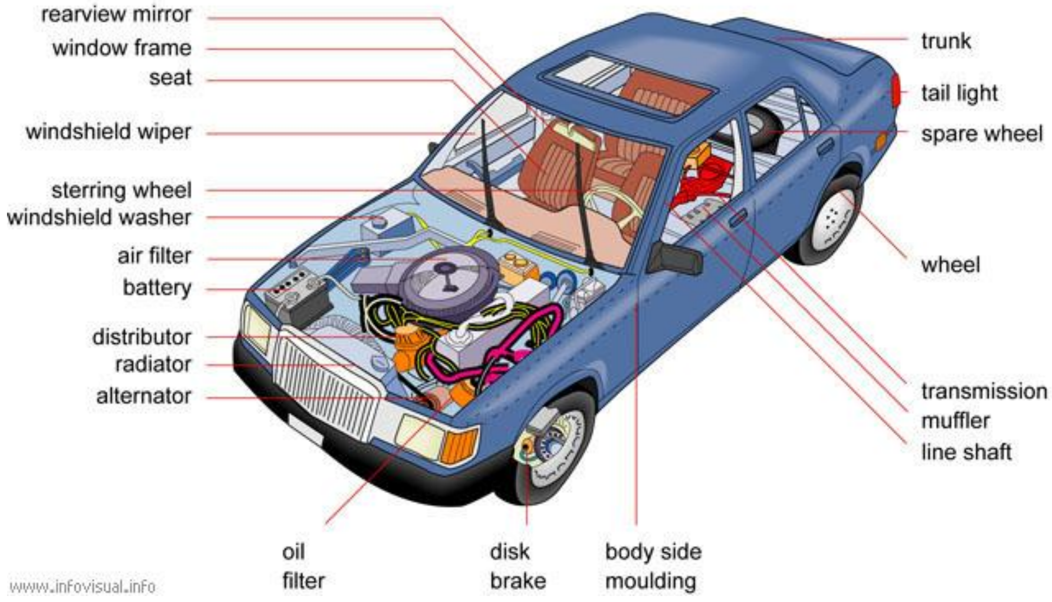
-

-

electronics, ATMs, netwo es, aircrafts, etc.

Introduction to Embedded Systems (1/4)

-



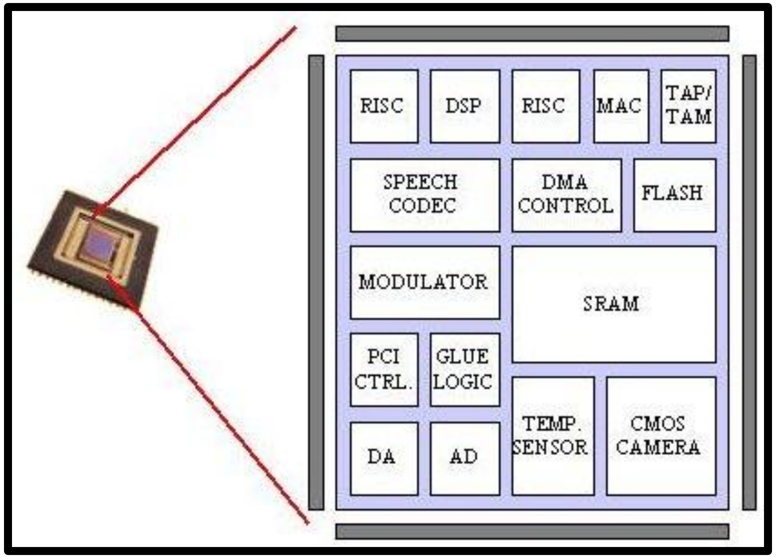
- automobiles,

Introduction to Embedded Systems (1/4)

- routers, automobiles, aircrafts, etc.

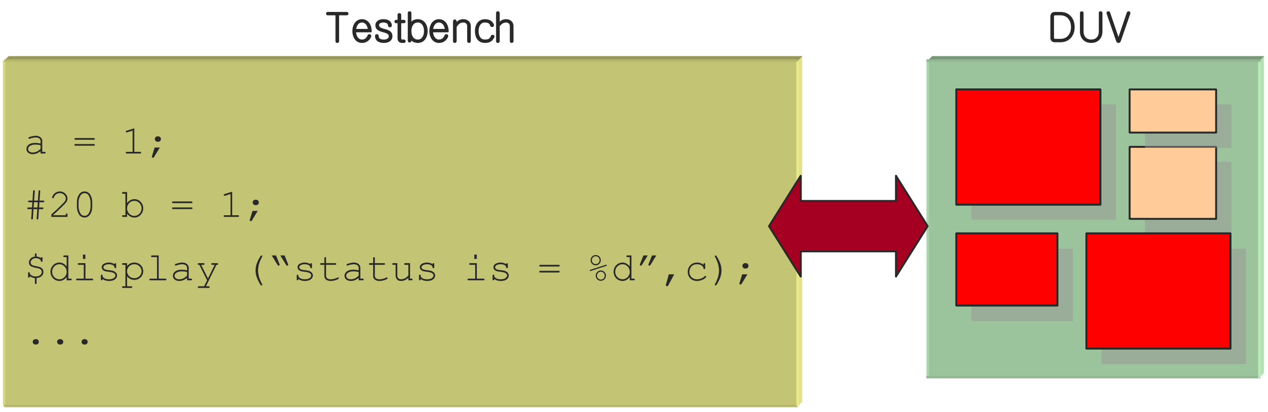
High Performance Embedded Systems (3/10)

- The current state-of-the-art:
  Multiple computational and data-processing engines, memory, and peripherals, all constructed on a single silicon chip called a System-on-Chip (SoC).
-



Conventional Verification of Embedded Systems (7/13)

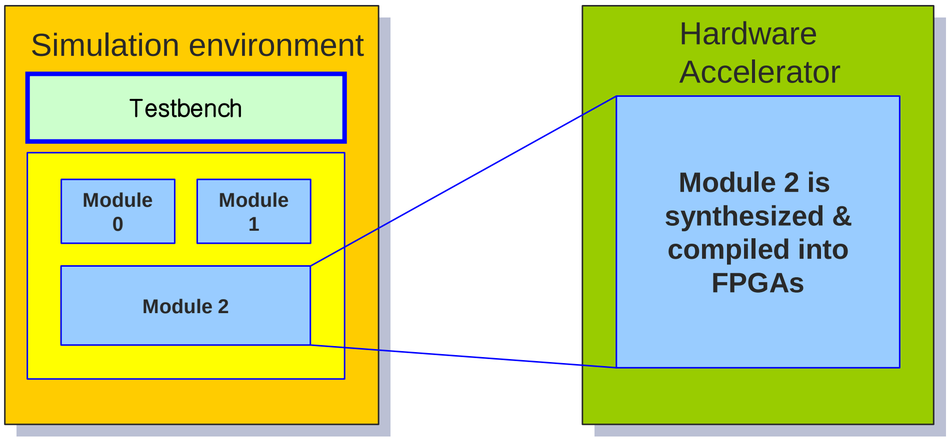
- **RTL-based simulation as verification mechanism**



Source: Chong-Min Kyung, "Current Status and Challenges of SoC Verification for Embedded Systems Market", IEEE International SOC Conference, 2003

Conventional Verification of Embedded Systems (5/13)

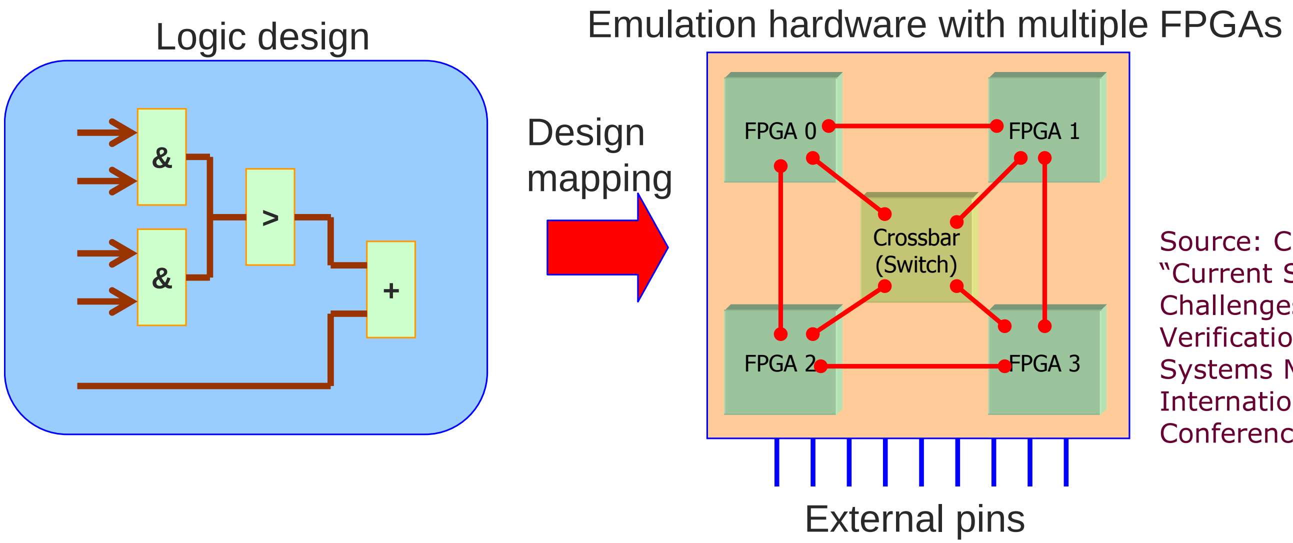
- **Hardware acceleration and emulation as verification mechanism**
  - These typically involve arrays of field-programmable gate arrays (FPGAs) or processors.



Source: Chong-Min Kyung, "Current Status and Challenges of SoC Verification for Embedded Systems Market", IEEE International SOC Conference, 2003

Conventional Verification of Embedded Systems (5/13)

- **Hardware acceleration and emulation as verification mechanism**

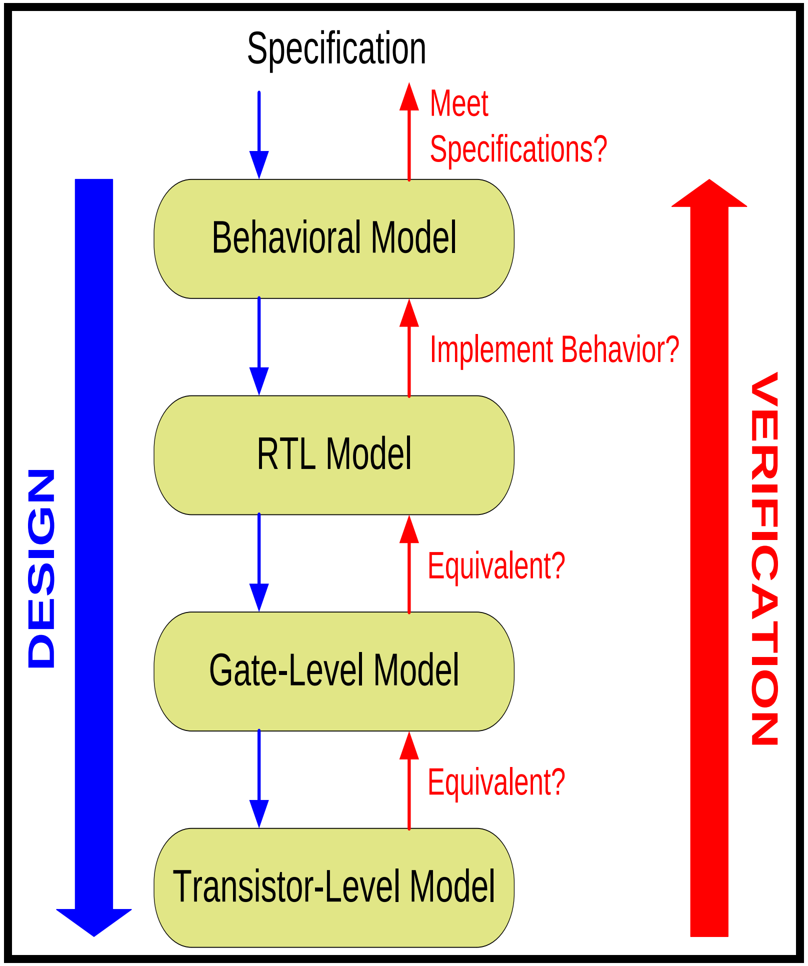


Source: Chong-Min Kyung, "Current Status and Challenges of SoC Verification for Embedded Systems Market", IEEE International SOC Conference, 2003

Verification of Complex Systems (4/15)

SoC architects to employ a broad system level design strategy that will allow:

- Explore and evaluate system level architectural choices
- Concurrent hardware-software design
- Easily evaluate and integrate a number of different technologies
- Adequate verification at every level of the design cycle



Verification of Complex Systems (1/15)

- System-On-Chip (SoC)