ECE 298: System-on-Chip Description and Modeling Lecture 2

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 2: Overview

- Introduction to System-on-Chip
  - SoC design flow
  - Computation vs. communication
  - Intellectual Property (IP)
- System-on-Chip Design Methodology
  - Specification model
  - Architecture model
  - Communication model
  - Implementation model

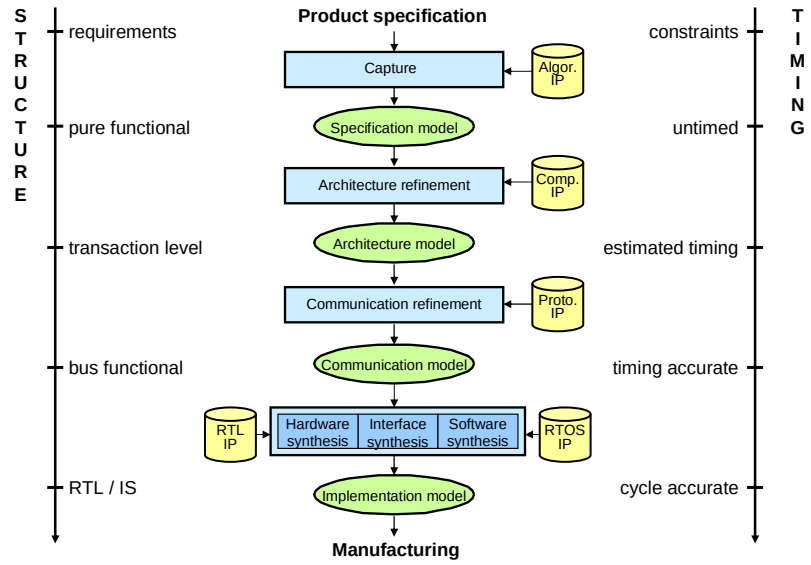
Introduction to System-on-Chip

- SoC design flow
- Computation vs. communication
- Intellectual Property (IP)

Introduction to System-on-Chip

- ➢ SoC design flow
- Computation vs. communication
- Intellectual Property (IP)

SoC Design Flow

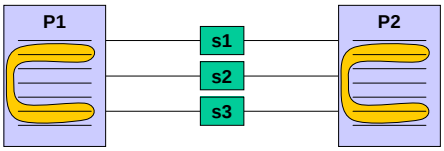


Introduction to System-on-Chip

- SoC design flow
- ➢ Computation vs. communication
- Intellectual Property (IP)

Computation vs. Communication

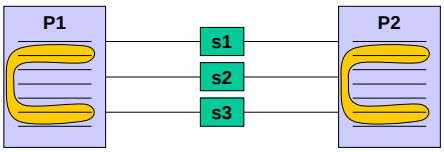
- Traditional model



  - Processes and signals
  - Mixture of computation and communication
  - Automatic replacement impossible

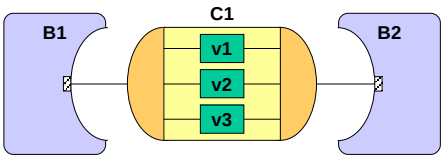
Computation vs. Communication

- Traditional model



  - Processes and signals
  - Mixture of computation and communication
  - Automatic replacement impossible

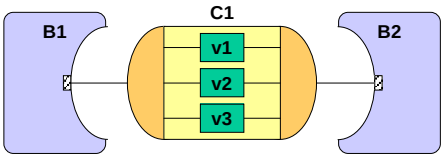
- SpecC model



  - Behaviors and channels
  - Separation of computation and communication
  - Plug-and-play

Computation vs. Communication

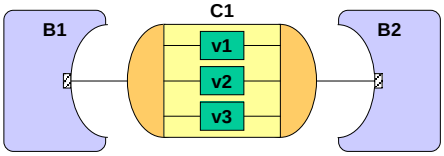
- Protocol Inlining
  - Specification model
  - Exploration model



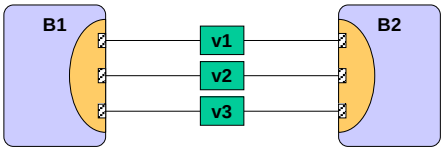
  - Computation in behaviors
  - Communication in channels

Computation vs. Communication

- Protocol Inlining
  - Specification model
  - Exploration model



  - Computation in behaviors
  - Communication in channels
  - Implementation model



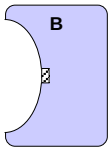
    - Channel disappears
    - Communication inlined into behaviors
    - Wires exposed

Introduction to System-on-Chip

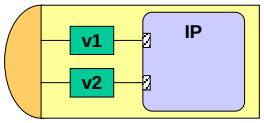
- SoC design flow
- Computation vs. communication
- ➢ Intellectual Property (IP)

Intellectual Property (IP)

- Computation IP: Wrapper model



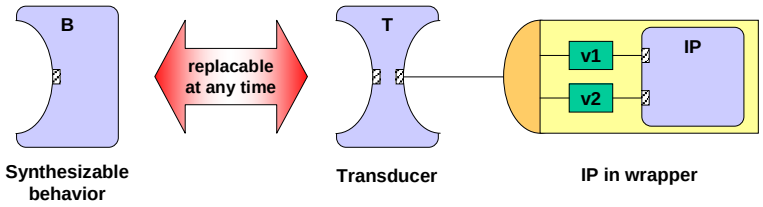
Synthesizable behavior



IP in wrapper

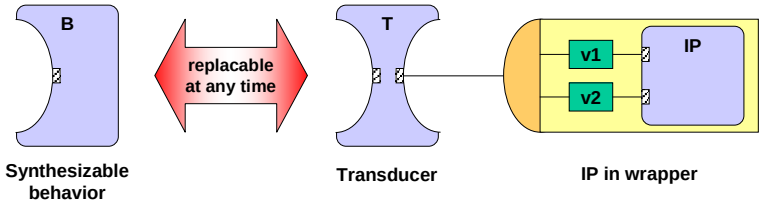
Intellectual Property (IP)

- Computation IP: Wrapper model

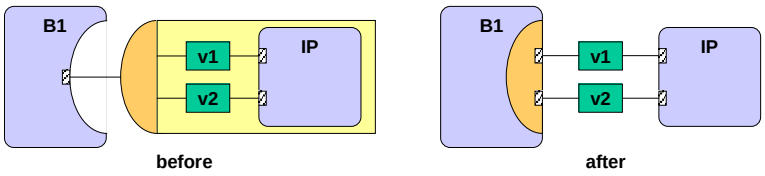


Intellectual Property (IP)

- Computation IP: Wrapper model

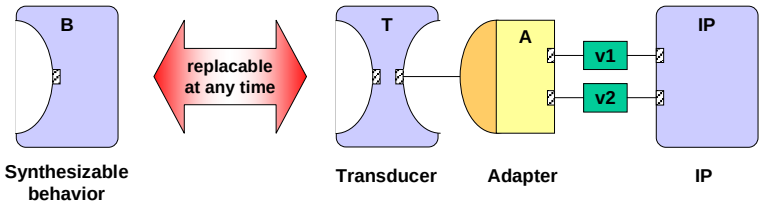


- Protocol inlining with wrapper



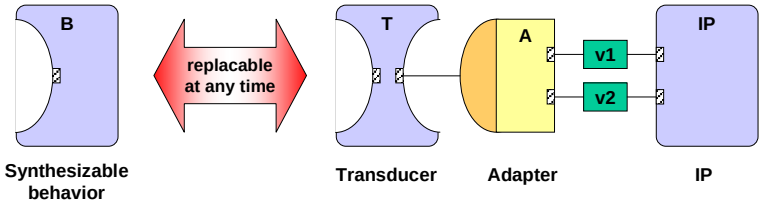
Intellectual Property (IP)

- Computation IP: Adapter model

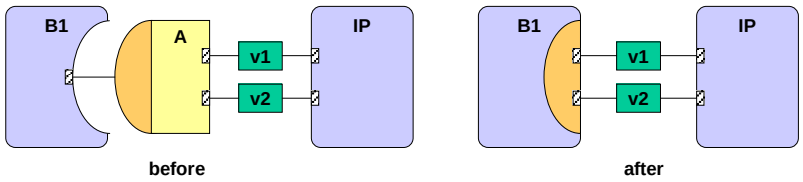


Intellectual Property (IP)

- Computation IP: Adapter model

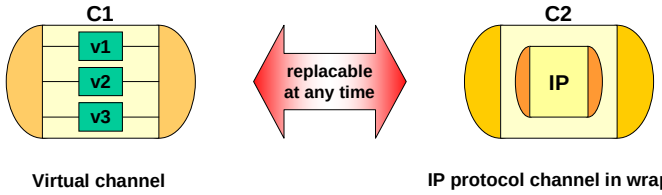


- Protocol inlining with adapter



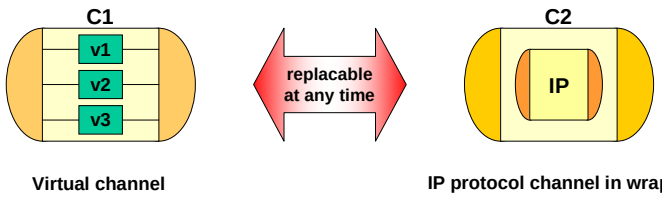
Intellectual Property (IP)

- Communication IP: Channel with wrapper

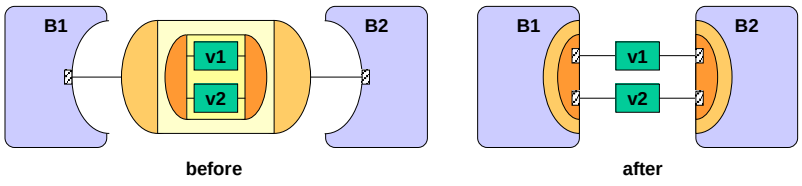


Intellectual Property (IP)

- Communication IP: Channel with wrapper

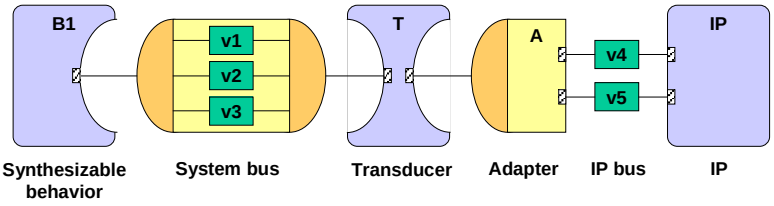


- Protocol inlining with hierarchical channel



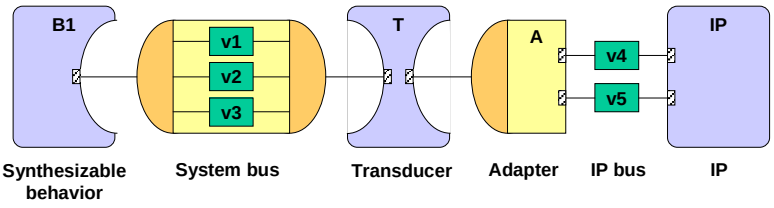
Intellectual Property (IP)

- Incompatible busses: Transducer insertion

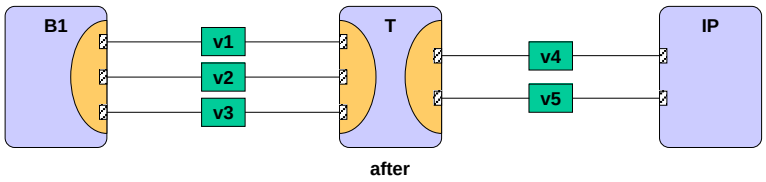


Intellectual Property (IP)

- Incompatible busses: Transducer insertion



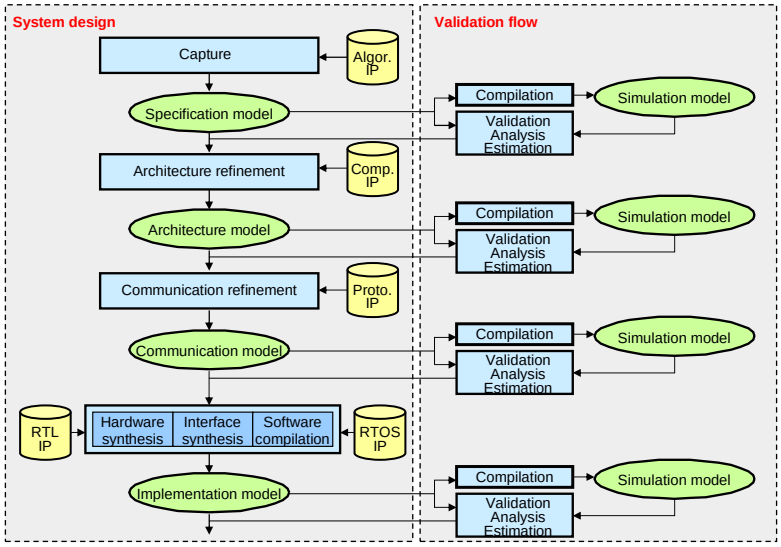
- Protocol inlining with transducer



System-on-Chip Design Methodology

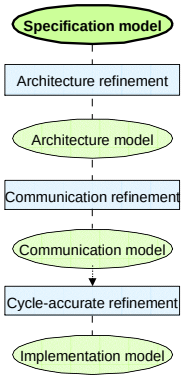
- Specification model
- Architecture model
- Communication model
- Implementation model

System-on-Chip Design Methodology



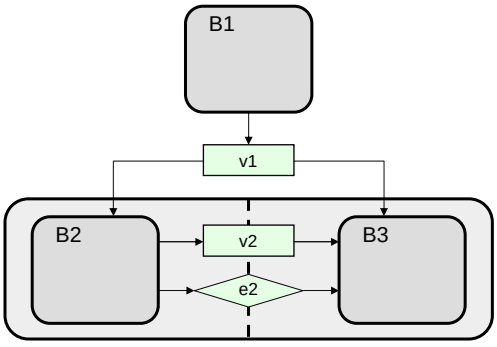
Specification Model

- High-level, abstract model
  - Pure system functionality
  - Algorithmic behavior
  - No implementation details
- No implicit structure / architecture
  - Behavioral hierarchy
- Untimed
  - Executes in zero (logical) time
  - Causal ordering
  - Events only for synchronization



(Source: A. Gerstlauer)

Specification Model Example

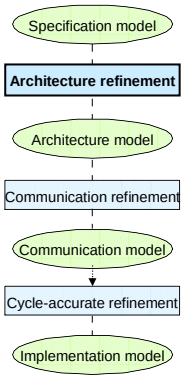


- Simple, typical specification model
  - Hierarchical parallel-serial composition
  - Communication through ports and variables, events

(Source: A. Gerstlauer)

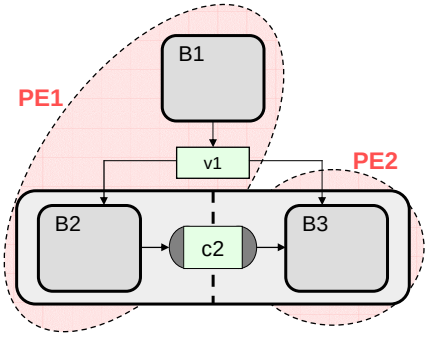
Architecture Refinement

- Component allocation / selection
- Behavior partitioning
- Variable partitioning
- Scheduling



(Source: A. Gerstlauer)

Allocation, Behavior Partitioning

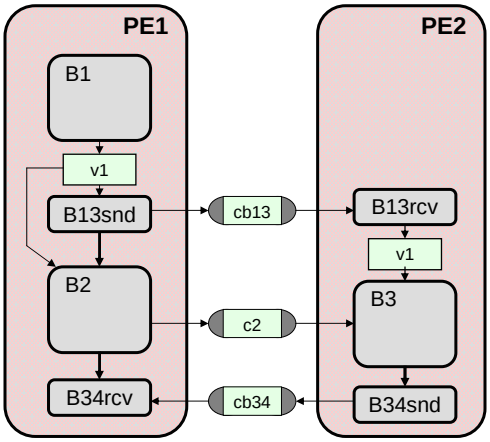


- Allocate PEs
- Partition behaviors
- Globalize communication

➢ Additional level of hierarchy to model PE structure

(Source: A. Gerstlauer)

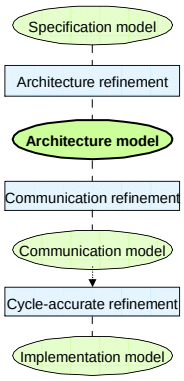
Architecture Model Example



(Source: A. Gerstlauer)

Architecture Model

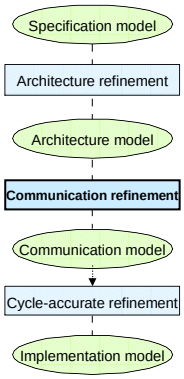
- Component structure/architecture
  - Top level of behavior hierarchy
- Behavioral/functional component view
  - Behaviors grouped under top-level component behaviors
  - Sequential behavior execution
- Timed
  - Estimated execution delays



(Source: A. Gerstlauer)

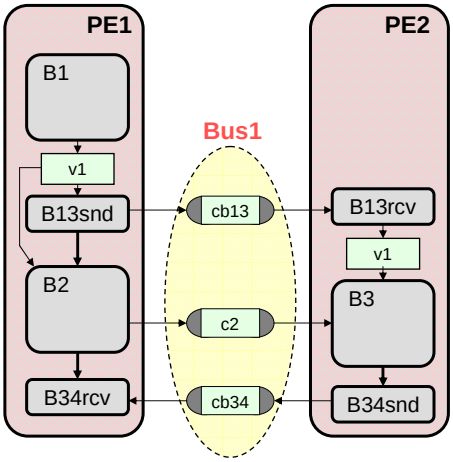
Communication Refinement

- Bus allocation / protocol selection
- Channel partitioning
- Protocol, transducer insertion
- Inlining



(Source: A. Gerstlauer)

Bus Allocation / Channel Partitioning

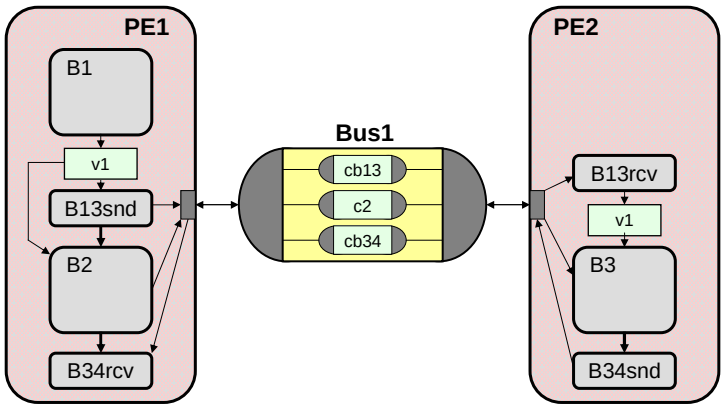


- Allocate busses
- Partition channels
- Update communication

➢ Additional level of hierarchy to model bus structure

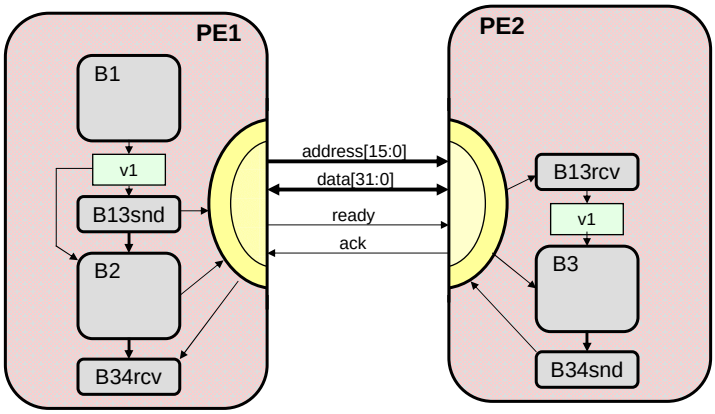
(Source: A. Gerstlauer)

Model after Channel Partitioning



(Source: A. Gerstlauer)

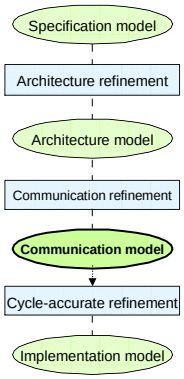
Communication Model Example



(Source: A. Gerstlauer)

Communication Model

- Component & bus structure/architecture
  - Top level of hierarchy
- Bus-functional component models
  - Timing-accurate bus protocols
  - Behavioral component description
- Timed
  - Estimated component delays

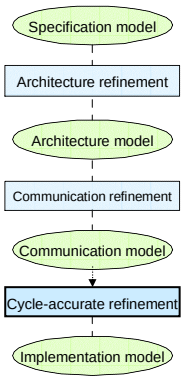


(Source: A. Gerstlauer)

ECE298: SoC Description and Modeling, Lecture 2 (c) 2004 R. Doemer 33

Cycle-accurate Refinement

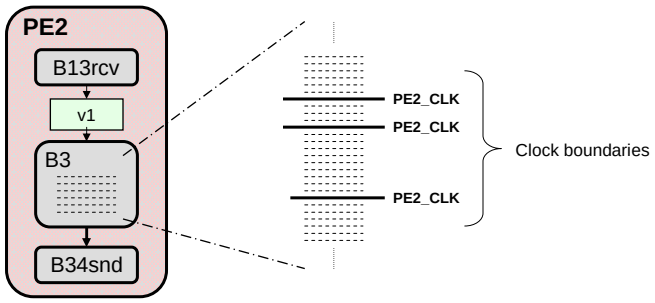
- Clock-accurate implementation of PEs
  - Hardware synthesis
  - Software synthesis
  - Interface synthesis



(Source: A. Gerstlauer)

ECE298: SoC Description and Modeling, Lecture 2 (c) 2004 R. Doemer 34

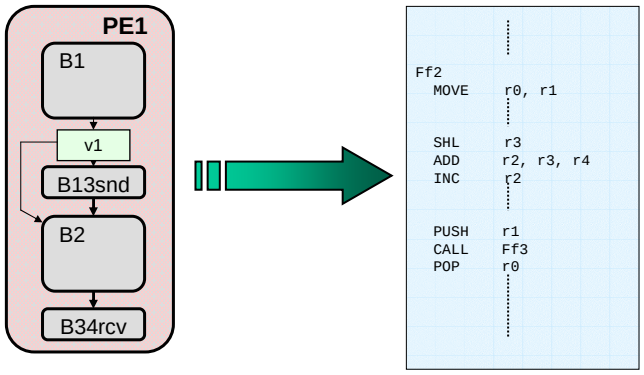
Hardware Synthesis



- Schedule operations into clock cycles
  - Define clock boundaries in leaf behavior C code
  - Create FSMD model from scheduled C code

(Source: A. Gerstlauer)

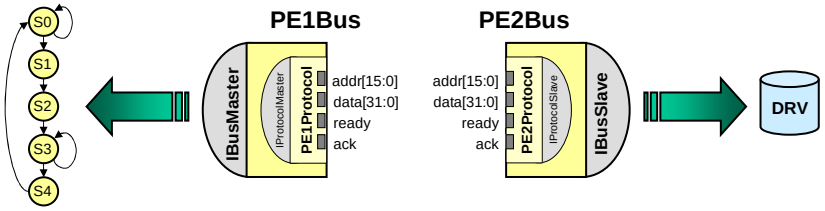
Software Synthesis



- Implement behavior on processor instruction-set
  - Code generation
  - Compilation

(Source: A. Gerstlauer)

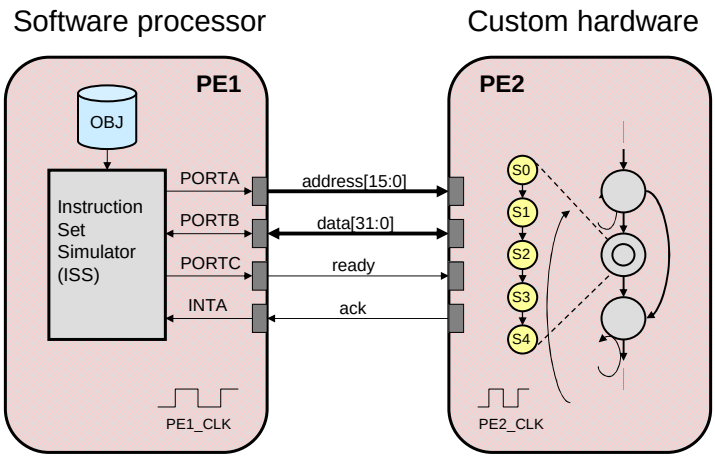
Interface Synthesis



- Implement communication on components
  - Hardware bus interface logic
  - Software bus drivers

(Source: A. Gerstlauer)

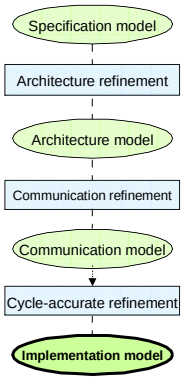
Implementation Model Example



(Source: A. Gerstlauer)

Implementation Model

- Cycle-accurate system description
  - RTL description of hardware
    - Behavioral/structural FSMD view
  - Object code for processors
    - Instruction-set co-simulation
  - Clocked bus communication
    - Bus interface timing based on PE clock



(Source: A. Gerstlauer)

System-on-Chip Design Methodology

- Four levels of abstraction
  - Specification model: untimed, functional
  - Architecture model: estimated, structural
  - Communication model: timed, bus-functional
  - Implementation model: cycle-accurate, RTL/IS
- Three refinement steps
  - Architecture refinement
  - Communication refinement
  - Cycle-accurate refinement
    - HW / SW / interface synthesis
- Well-defined, formal models & transformations
  - Automatic, gradual refinement
  - Executable models, test bench re-use
  - Simple verification

(Source: A. Gerstlauer)