Reading Assignment

- This lecture: 3.3, Project I
- Next lecture: 3.4
Outline

System Design Methodology

Processor and Communication Modelings
System Design Challenges

- Input: a high-level system specification
  - Could be functionalities only described in MoCs
- Output: a low-level system implementation
  - Software: programs for the targeted instruction sets
  - Hardware: what level?
- Methodologies and tools are mature for abstraction levels at and below RTL for hardware.
  - For system design, hardware implementations stop at RTL.
- No single step solution for system design
  - Semantic gap: there are multiple ways to implement a single MoC
  - Huge semantic gap exists between specification and implementation.
System Design Challenges

- Input: a high-level system specification
  - Could be functionalities only described in MoCs
- Output: a low-level system implementation
  - Software: programs for the targeted instruction sets
  - Hardware: what level?
- Methodologies and tools are mature for abstraction levels at and below RTL for hardware.
  - For system design, hardware implementations stop at RTL.
- No single step solution for system design
  - Semantic gap: there are multiple ways to implement a single MoC
  - Huge semantic gap exists between specification and implementation.
System Design Challenges

- Input: a high-level system specification
  - Could be functionalities only described in MoCs
- Output: a low-level system implementation
  - Software: programs for the targeted instruction sets
  - Hardware: what level?
- Methodologies and tools are mature for abstraction levels at and below RTL for hardware.
  - For system design, hardware implementations stop at RTL.
- No single step solution for system design
  - Semantic gap: there are multiple ways to implement a single MoC
  - Huge semantic gap exists between specification and implementation.
System Design Challenges

▶ Input: a high-level system specification
  ▶ Could be functionalities only described in MoCs
▶ Output: a low-level system implementation
  ▶ Software: programs for the targeted instruction sets
  ▶ Hardware: what level?
▶ Methodologies and tools are mature for abstraction levels at and below RTL for hardware.
  ▶ For system design, hardware implementations stop at RTL.
▶ No single step solution for system design
  ▶ Semantic gap: there are multiple ways to implement a single MoC
  ▶ Huge semantic gap exists between specification and implementation.
System Design Methodology

- Decompose the whole system design process into a series of smaller steps.
  - Ensure the semantic gap is small enough for a single step
- Each step is defined by a pair of system models.
  - The one at higher abstraction level serves as specification.
  - The one at lower abstraction level serves as implementation.
- Refinement: generate the implementation from the specification for each step
  - Introduce additional details limited to certain scope of the specification
  - Incorporate design decisions to choose one implementation from multiple possible ones
- While more tools are available for refinement, it is critical for designers to provide proper design decisions.
  - Especially when an initial system implementation fails to meet design constraints and multiple design iterations are necessary.
System Design Methodology

- Decompose the whole system design process into a series of smaller steps.
  - Ensure the semantic gap is small enough for a single step
- Each step is defined by a pair of system models.
  - The one at higher abstraction level serves as specification.
  - The one at lower abstraction level serves as implementation.
- Refinement: generate the implementation from the specification for each step
  - Introduce additional details limited to certain scope of the specification
  - Incorporate design decisions to choose one implementation from multiple possible ones
- While more tools are available for refinement, it is critical for designers to provide proper design decisions.
  - Especially when an initial system implementation fails to meet design constraints and multiple design iterations are necessary.
System Design Methodology

- Decompose the whole system design process into a series of smaller steps.
  - Ensure the semantic gap is small enough for a single step
- Each step is defined by a pair of system models.
  - The one at higher abstraction level serves as specification.
  - The one at lower abstraction level serves as implementation.
- Refinement: generate the implementation from the specification for each step
  - Introduce additional details limited to certain scope of the specification
  - Incorporate design decisions to choose one implementation from multiple possible ones
- While more tools are available for refinement, it is critical for designers to provide proper design decisions.
  - Especially when an initial system implementation fails to meet design constraints and multiple design iterations are necessary.
System Design Methodology

- Decompose the whole system design process into a series of smaller steps.
  - Ensure the semantic gap is small enough for a single step.
- Each step is defined by a pair of system models.
  - The one at higher abstraction level serves as specification.
  - The one at lower abstraction level serves as implementation.
- Refinement: generate the implementation from the specification for each step.
  - Introduce additional details limited to certain scope of the specification.
  - Incorporate design decisions to choose one implementation from multiple possible ones.
- While more tools are available for refinement, it is critical for designers to provide proper design decisions.
  - Especially when an initial system implementation fails to meet design constraints and multiple design iterations are necessary.
Typical System Design Tool and Design Process

**FIGURE 3.6** System design and modeling flow

(Gajski et al.)
Roles of Models

- The implementation of the previous design step will serve as the specification of the next one.
- For implementation, models allow designers to reason about design decisions by simulating and analyzing certain aspects of the system.
- For specification, models document system features that need to be implemented and decided.
- As design progresses,
  - More details are included into the models so simulation and analysis takes more time to finish.
  - Simulation and analysis will become more accurate due to the available details.
  - Designers will be able to afford the increased simulation and analysis time by focusing on the most important parts of the system.
Roles of Models

- The implementation of the previous design step will serve as the specification of the next one.
- For implementation, models allow designers to reason about design decisions by simulating and analyzing certain aspects of the system.
- For specification, models document system features that need to be implemented and decided.
- As design progresses,
  - More details are included into the models so simulation and analysis takes more time to finish.
  - Simulation and analysis will become more accurate due to the available details.
  - Designers will be able to afford the increased simulation and analysis time by focusing on the most important parts of the system.
Roles of Models

- The implementation of the previous design step will serve as the specification of the next one.
- For implementation, models allow designers to reason about design decisions by simulating and analyzing certain aspects of the system.
- For specification, models document system features that need to be implemented and decided.
- As design progresses,
  - More details are included into the models so simulation and analysis takes more time to finish.
  - Simulation and analysis will become more accurate due to the available details.
  - Designers will be able to afford the increased simulation and analysis time by focusing on the most important parts of the system.
Roles of Models

- The implementation of the previous design step will serve as the specification of the next one.
- For implementation, models allow designers to reason about design decisions by simulating and analyzing certain aspects of the system.
- For specification, models document system features that need to be implemented and decided.
- As design progresses,
  - More details are included into the models so simulation and analysis takes more time to finish.
  - Simulation and analysis will become more accurate due to the available details.
  - Designers will be able to afford the increased simulation and analysis time by focusing on the most important parts of the system.
Abstraction Levels for System Design

- At the highest abstraction level, we would assume the system is specified using process-based models while each process is specified using state-based models.

- What intermediate abstraction levels should we introduce for HW/SW implementations?

- Separate communication from computation
  - Compilers will help to implement a single process as HW or SW.
  - Communications become limiting factors for system performance.

- Accurate system analysis demands accurate communication modeling.
  - The ratio of communication latency to computation latency generally increases as more transistors are packed into a chip.
  - Complex system requires more data to be transferred among subsystems, resulting in latency with limited bandwidth and excessive power consumption.
Abstraction Levels for System Design

- At the highest abstraction level, we would assume the system is specified using process-based models while each process is specified using state-based models.

- What intermediate abstraction levels should we introduce for HW/SW implementations?
  - Separate communication from computation
    - Compilers will help to implement a single process as HW or SW.
    - Communications become limiting factors for system performance.

- Accurate system analysis demands accurate communication modeling.
  - The ratio of communication latency to computation latency generally increases as more transistors are packed into a chip.
  - Complex system requires more data to be transferred among subsystems, resulting in latency with limited bandwidth and excessive power consumption.
Abstraction Levels for System Design

- At the highest abstraction level, we would assume the system is specified using process-based models while each process is specified using state-based models.
- What intermediate abstraction levels should we introduce for HW/SW implementations?
- Separate communication from computation
  - Compilers will help to implement a single process as HW or SW.
  - Communications become limiting factors for system performance.
- Accurate system analysis demands accurate communication modeling.
  - The ratio of communication latency to computation latency generally increases as more transistors are packed into a chip.
  - Complex system requires more data to be transferred among subsystems, resulting in latency with limited bandwidth and excessive power consumption.
At the highest abstraction level, we would assume the system is specified using process-based models while each process is specified using state-based models.

What intermediate abstraction levels should we introduce for HW/SW implementations?

Separate communication from computation
  - Compilers will help to implement a single process as HW or SW.
  - Communications become limiting factors for system performance.

Accurate system analysis demands accurate communication modeling.
  - The ratio of communication latency to computation latency generally increases as more transistors are packed into a chip.
  - Complex system requires more data to be transferred among subsystems, resulting in latency with limited bandwidth and excessive power consumption.
System Design Models

- Cycle-Accurate Model (CAM)
  - Both computation and communication are specified cycle-by-cycle.
  - Model functionality in implementation
- Specification Model (SM)
  - Both computation and communication are not timed.
- Timed Functional Model
- Transaction-Level Model (TLM)
  - Both computation and communication are approximated timed
- Bus Cycle-Accurate Model (BCAM)
- Computation Cycle-Accurate Model (CCAM)
System Design Models

- Cycle-Accurate Model (CAM)
  - Both computation and communication are specified cycle-by-cycle.
  - Model functionality in implementation
- Specification Model (SM)
  - Both computation and communication are not timed.
- Timed Functional Model
- Transaction-Level Model (TLM)
  - Both computation and communication are approximated timed
- Bus Cycle-Accurate Model (BCAM)
- Computation Cycle-Accurate Model (CCAM)
System Design Models

- **Cycle-Accurate Model (CAM)**
  - Both computation and communication are specified cycle-by-cycle.
  - Model functionality in implementation

- **Specification Model (SM)**
  - Both computation and communication are not timed.

- **Timed Functional Model**
  - **Transaction-Level Model (TLM)**
    - Both computation and communication are approximated timed
  - **Bus Cycle-Accurate Model (BCAM)**
  - **Computation Cycle-Accurate Model (CCAM)**
System Design Models

- **Cycle-Accurate Model (CAM)**
  - Both computation and communication are specified cycle-by-cycle.
  - Model functionality in implementation

- **Specification Model (SM)**
  - Both computation and communication are not timed.

- **Timed Functional Model**

- **Transaction-Level Model (TLM)**
  - Both computation and communication are approximated timed
    - **Bus Cycle-Accurate Model (BCAM)**
    - **Computation Cycle-Accurate Model (CCAM)**
System Design Models

▶ Cycle-Accurate Model (CAM)
   ▶ Both computation and communication are specified cycle-by-cycle.
   ▶ Model functionality in implementation
▶ Specification Model (SM)
   ▶ Both computation and communication are not timed.
▶ Timed Functional Model
▶ Transaction-Level Model (TLM)
   ▶ Both computation and communication are approximated timed
▶ Bus Cycle-Accurate Model (BCAM)
▶ Computation Cycle-Accurate Model (CCAM)
System Design Models

- **Cycle-Accurate Model (CAM)**
  - Both computation and communication are specified cycle-by-cycle.
  - Model functionality in implementation

- **Specification Model (SM)**
  - Both computation and communication are not timed.

- **Timed Functional Model**

- **Transaction-Level Model (TLM)**
  - Both computation and communication are approximated timed

- **Bus Cycle-Accurate Model (BCAM)**

- **Computation Cycle-Accurate Model (CCAM)**
System Design Methodology as Model Transformations

A design methodology is illustrated as a path from A to F. We will be interested in the one passing C.
A design methodology is illustrated as a path from A to F.

We will be interested in the one passing C.
Outline

System Design Methodology

Processor and Communication Modelings
Processor modeling provides necessary details to evaluate mappings from processes to processors.

- Processors: a.k.a. processing elements
  - General-purpose processors
  - DSPs and ASIPs
  - ASICs

- The computation part of system modeling

- The communication part of system modeling is based on communication modeling.
Processor modeling provides necessary details to evaluate mappings from processes to processors.

- Processors: a.k.a. processing elements
  - General-purpose processors
  - DSPs and ASIPs
  - ASICs

- The computation part of system modeling

- The communication part of system modeling is based on communication modeling.
Processor Modeling

- Processors specific to certain applications
  - Processes are usually specified with MoCs that can be directly mapped to the processors.
  - Based on hardware metrics, estimating system design metrics could be straightforward.
  - Otherwise, we can model such processors as special cases of general-purpose processors.

- General-purpose processors, or software processor
  - Processes are usually specified as sequential programs.
  - To estimate system design metrics, one has to consider not only the programs but the supporting software (e.g. OS).
  - Much more complicated than the above case. Will be our focus.

- Challenges for software processors modeling
  - Most estimations of system design metrics, e.g. latency, throughput, and power consumption, depend on simulation.
  - Models enable fast simulations while provide accurate (relatively) estimations are desired.
  - At what level should we model software processors?
Processor Modeling

- Processors specific to certain applications
  - Processes are usually specified with MoCs that can be directly mapped to the processors.
  - Based on hardware metrics, estimating system design metrics could be straight-forward.
  - Otherwise, we can model such processors as special cases of general-purpose processors.

- General-purpose processors, or software processor
  - Processes are usually specified as sequential programs.
  - To estimate system design metrics, one has to consider not only the programs but the supporting software (e.g. OS).
  - Much more complicated than the above case. Will be our focus.

- Challenges for software processors modeling
  - Most estimations of system design metrics, e.g. latency, throughput, and power consumption, depend on simulation.
  - Models enable fast simulations while provide accurate (relatively) estimations are desired.
  - At what level should we model software processors?
Processor Modeling

- Processors specific to certain applications
  - Processes are usually specified with MoCs that can be directly mapped to the processors.
  - Based on hardware metrics, estimating system design metrics could be straightforward.
  - Otherwise, we can model such processors as special cases of general-purpose processors.

- General-purpose processors, or software processor
  - Processes are usually specified as sequential programs.
  - To estimate system design metrics, one has to consider not only the programs but the supporting software (e.g. OS).
  - Much more complicated than the above case. Will be our focus.

- Challenges for software processors modeling
  - Most estimations of system design metrics, e.g. latency, throughput, and power consumption, depend on simulation.
  - Models enable fast simulations while provide accurate (relatively) estimations are desired.
  - At what level should we model software processors?
Communication Modeling

- Adopt well-established ISO/OSI 7-layer model
  - Layers are stacked on top of each other.
  - Each layer provides services to layers above by using services of the layer below.
- Layers are tailored to specific system design requirements.
  - E.g. to reflect the HW/SW partitioning of the communication functionality
- Use of layers facilities reasoning about communication stacks
  - However, it should not prevent implementations to merging functionalities across layers for various optimizations.
  - The whole communication stack should be treated as a single specification.
Communication Modeling

- Adopt well-established ISO/OSI 7-layer model
  - Layers are stacked on top of each other.
  - Each layer provides services to layers above by using services of the layer below.
- Layers are tailored to specific system design requirements.
  - E.g. to reflect the HW/SW partitioning of the communication functionality
- Use of layers facilities reasoning about communication stacks
  - However, it should not prevent implementations to merging functionalities across layers for various optimizations.
  - The whole communication stack should be treated as a single specification.
Communication Modeling

- Adopt well-established ISO/OSI 7-layer model
  - Layers are stacked on top of each other.
  - Each layer provides services to layers above by using services of the layer below.
- Layers are tailored to specific system design requirements.
  - E.g. to reflect the HW/SW partitioning of the communication functionality
- Use of layers facilities reasoning about communication stacks
  - However, it should not prevent implementations to merging functionalities across layers for various optimizations.
  - The whole communication stack should be treated as a single specification.
Summary

- System design is decomposed into steps.
- A model for implementation at one step will serve for specification at the next step.
- System design models can be classified by how they model computation and communication.
- System design methodology is represented by the transformations of the system at various models.