Embedded System Design

Modeling, Synthesis, Verification

Daniel D. Gajski, Samar Abdi, Andreas Gerstlauer, Gunar Schirner

Chapter 2: System Design Methodologies

7/8/2009

Overview

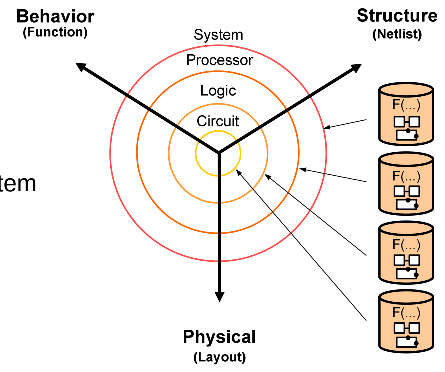
- Design methodologies
- Bottom-up
- Top-down
- Meet-in-the-middle
- Platform
- System
- FPGA
- System synthesis flow
- Processor synthesis flow
- Conclusion

Design Methodologies

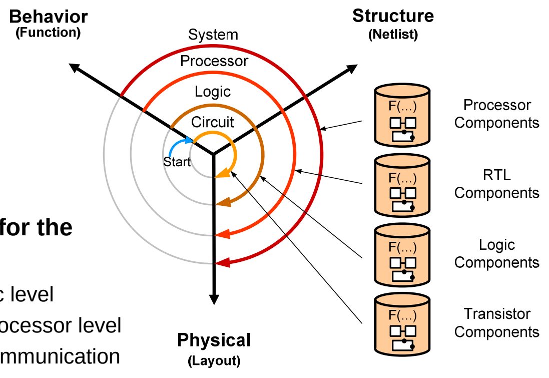
- Design methodology is a sequence of design models, components and tools used to design the product
- Methodologies evolve with technology, complexity and automation
- A methodology depends on application, company and design group focus
- Standardization arrives when the cost of being special is too high

Y Chart

- 3 design views
  - Behavior, Structure, Physical
- 4 abstraction levels
  - Circuit, Logic, Processor, System
- 5 component libraries
  - Transistors
  - Logic (standard cells)
  - RTL (ALUs, RFs, …)
  - Processor (standard, custom)
  - System (multi-core with NoC)



Bottom-up Methodology

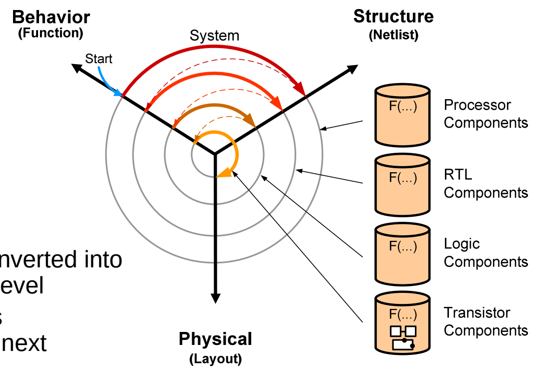


- **Starts from the bottom level**
- **Each level generates library for the next higher level**
  - Circuit: Standard cells for logic level
  - Logic: RTL components for processor level
  - Processor: Processing and communication components for system level
  - System: Embedded systems platforms for different applications
- **Floorplaning and layout on each level**

Bottom-up Methodology

- **Pros**
  - Abstraction levels clearly separated with its own library
  - Accurate metric estimation with layout on each level
  - Globally distributed development possible
  - Easy management
- **Cons**
  - An optimal library for each design is difficult to predict
    - All possible components with all possible parameters
    - All possible optimizations for all possible metrics
  - Library customization is outside the design group
  - Layout is performed on every level

Top-down Methodology

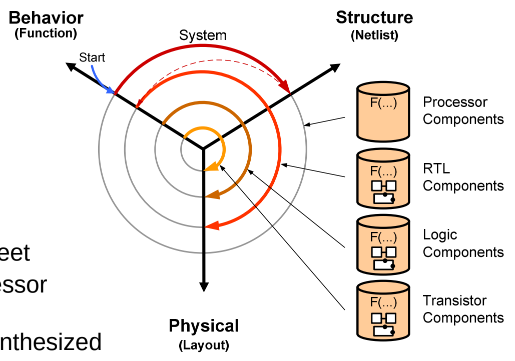


- Starts with the top level
- Functional description is converted into component netlist on each level
- Each component function is decomposed further on the next abstraction level
- Layout is given only for transistor components

Top-down Methodology

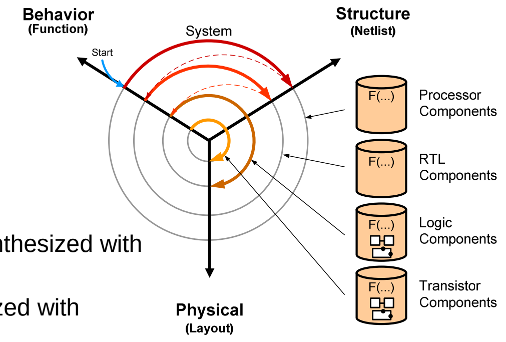
- **Pros**
  - Highest level of customization possible on each abstraction level
  - Only one small transistor library needed
  - Only one layout design at the end
- **Cons**
  - Difficult metric estimation on upper levels since layout is not known until the end
  - Design decision impact on higher level not clear
  - Hot spot removal is difficult
  - Metric annotation (closure) from lower to higher levels needed during design iterations

Meet-in-the-Middle Methodology (Option 1)



- Combines top-down and bottom-up
  - Synthesis vs. layout compromise
- Processor level where they meet
- MoC is synthesized into processor components
- Processor components are synthesized with RTL library
- System layout is generated with RTL components

Meet-in-the-Middle Methodology (Option 2)



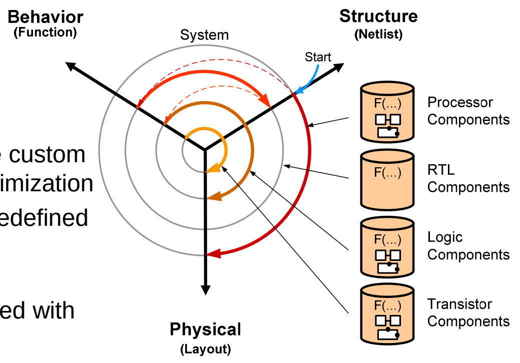
- RTL level where they meet
- MoC is synthesized with processor components
- Processor components are synthesized with RTL library
- RTL components are synthesized with standard cells
- System layout is performed with standard cells
- Two levels of layout

Meet-in-the-Middle Methodology

- **Pros**
  - Shorter synthesis
  - Less layout
  - Less libraries
  - Better metric closure
- **Cons**
  - Still needs libraries
  - More then one layout
  - Metric closure still needed
  - Library components may not be optimal

Platform Methodology

- System platform with standard components and synthesizable custom components for application optimization
- Layout is on system level or predefined with special area for custom components layout
- Custom components synthesized with RTL and logic and laid out with standard cells
- Custom components must fit into platform structure

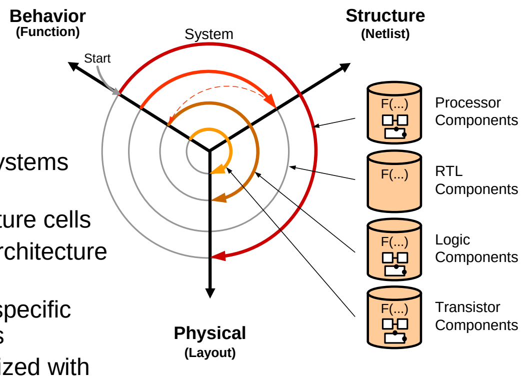


Platform Methodology

- **Pros**
  - Two types of layout: system layout for platform (could be predefined) and standard cell layout for custom components
  - Standard processors are available
  - Custom and interface components are added for optimization
- **Cons**
  - Platform customization is still needed
  - SW and IF components synthesis required

System Methodology

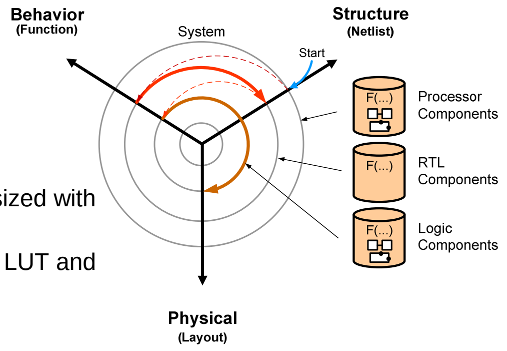
- Methodology for embedded systems developers
- System platform with architecture cells
- Layout on system level with architecture cells
- Architecture cells defined for specific application and design metrics
- Architecture cells pre-synthesized with RTL and logic and laid out with standard cells
- A retargetable compiler for architecture cells



System Methodology

- **Pros**
  - Processor-level component only
  - Single retargetable compiler for all architecture cells
  - Processor-level layout
  - Methodology for application experts
  - Minimal knowledge of system and processor levels
- **Cons**
  - Architecture cell definition and library
  - IS definition
  - Change of mind

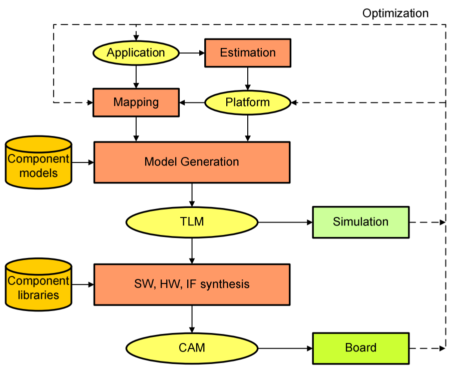
FPGA Methodology



- Starts with system structure
- Processor components synthesized with RTL and logic components
- Components implemented with LUT and BRAMs
- Layout only once
- Metric estimation very difficult
- Estimation is hidden in the FPGA supplier tools

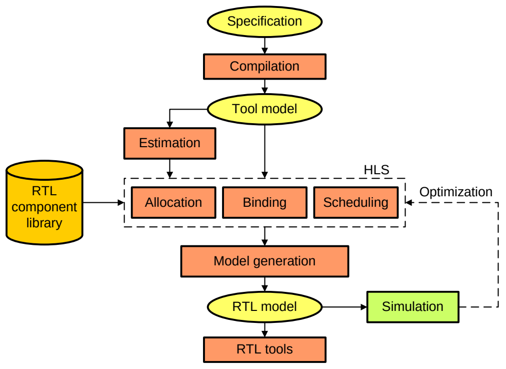
System Level Synthesis

- **Application given in a MoC**
- **TLM tools**
  - Estimation and platform definition
  - Application mapping
  - TLM generation
- **Simulation**
- **Optimization**
  - Application change
  - Platform change
  - Mapping change
- **CAM tools**
  - Platform SW, HW and IF synthesis
  - CAM generation
- **Board prototyping**
- **Satisfactory or optimization**



Processor Synthesis

- **Compilation**
- **Estimation**
  - Resources
  - Metrics
- **HLS**
  - Allocation
  - Binding
  - Scheduling
- **RTL model generation**
- **Simulation**
- **Optimization**
  - Change estimates
  - Change HLS parameters
- **RTL synthesis**



Conclusion

- **Basic concepts of system design methodologies introduced**
- **Many different methodologies in use**
  - One for every group, product, and company
- **Methodologies differ in:**
  - Input specification, MoC
  - Modeling styles and languages
  - Abstraction levels and amount of detail
  - Verification strategy and prototyping
  - CAD tools and component libraries
- **Standards emerge slowly through experience**
- **Application-oriented methodologies are necessary for embedded systems**