The Design and Implementation Process

Bruce Shriver
University of Tromsø, Norway

What Must Be Done?

- we need to take the high-level representation of the instruction set architecture and transform it into a correctly working, high-performance, reliable silicon chip
- doing this is not so simple

(Some) Difficulties Involved

- microprocessors are complex
- microprocessors require and increasingly large number of devices to realize their designs
- highly dependent trade-offs are required to achieve specific size, voltage, power, temperature, yield, performance and price points

Workloads & Benchmarks
Assist in Providing Data to Make the Trade-Offs

- the workloads & benchmarks and subsequent data analysis may be targeted at:
  - a specific component
    - e.g., the floating-point unit
  - a group of components
    - e.g., the logic involved in the fetching and decoding of instructions
  - the entire microprocessor itself
  - the microprocessor connected to one or more external components
    - e.g., an off-chip cache

Workloads & Benchmarks

- the design of specific workloads and benchmarks for any of the targets is an art in itself
  - typically, millions of instruction executions are required to gather even a small understanding of the specific behavior under study
- bottlenecks and their causes must be identified in order to be removed

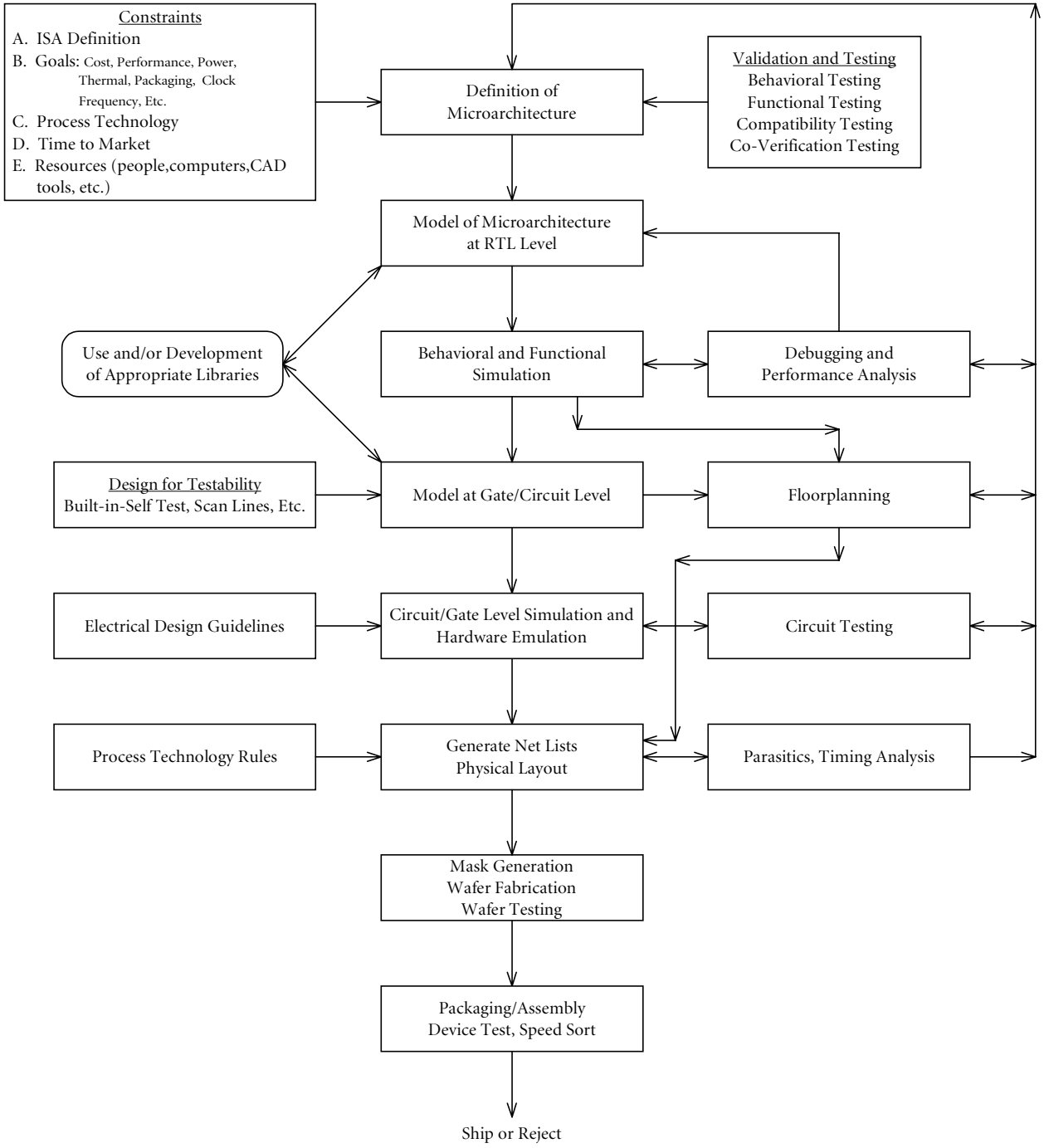
Workloads & Benchmarks

- knowledge of
  - typical application and system software
  - typical coding practice
  - nature of code generated by compilers
- analysis of instruction and data references traces of actual software implies
  - the ability to capture traces of any and all code
    - e.g., privileged OS code or device driver code
    - e.g., applications code

Trace-Drive Performance Model

- such a model is required and must
  - output a wide variety of performance-related statistics
  - be able to be readily changed to explore alternative design options
- length of required traces may be ten million to one billion instructions long

- Let’s look at a diagram of a process that can be followed to design and implement a microprocessor


Constraints
A. ISA Definition
B. Goals: Cost, Performance, Power, Thermal, Packaging, Clock Frequency, Etc.
C. Process Technology
D. Time to Market
E. Resources (people,computers,CAD tools, etc.)
Definition of Microarchitecture
Validation and Testing
Behavioral Testing
Functional Testing
Compatibility Testing
Co-Verification Testing
Model of Microarchitecture at RTL Level
Use and/or Development of Appropriate Libraries
Behavioral and Functional Simulation
Debugging and Performance Analysis
Design for Testability
Built-in-Self Test, Scan Lines, Etc.
Model at Gate/Circuit Level
Floorplanning
Electrical Design Guidelines
Circuit/Gate Level Simulation and Hardware Emulation
Circuit Testing
Process Technology Rules
Generate Net Lists Physical Layout
Parasitics, Timing Analysis
Mask Generation
Wafer Fabrication
Wafer Testing
Packaging/Assembly
Device Test, Speed Sort
Ship or Reject

Multiple Levels

- this process deals with many levels
  - the RTL level
  - the gate/circuit level
  - netlists and physical layout level
  - wafer fabrication level (transistor /chip level)
  - packaged device level

Levels of Abstraction

Digital Design Engineer's Level	**Examples of Modeled Entity**	**Approximate Equivalence to Computer Architect's Level**
system level	pipelines, instruction decoders, and TLBs	microarchitecture
register transfer level (RTL)	registers, buses, multiplexers, and combinational logic	microarchitecture and logical implementation
gate level or logic level	library cells of AND, OR, etc. gates	logical implementation
transistor level	transistors in various process tech nologies	chip

Defining the Microarchitecture
(one of many potential starting points)

- this staring point assumes the instruction set architecture has been defined
- constraints
- validation and testing
- developing a “golden representation” at the RTL level

“Golden Representation”

- in order to appreciate this concept, we need some background related to hardware description languages (HDLs)

Conventional Programming Languages

- extended to include:
  - notions of time, parallelism and synchronization
  - architectural and hardware data structure extensions
- the extended languages are called hardware description languages or HDLs
- a model, which is the definition of the microprocessor implemented as a program in the HDL, can be simulated by compiling and executing it
- since the program can be written to describe the microprocessor at any level of abstraction, the simulation will be of the microprocessor at that particular level

Hardware Description Languages
(hardware modeling languages)

- used to describe the complex behavior and structure of digital systems
  - accuracy
    - the level of representation
    - the details described
    - the precision with which they are described
  - speed
- example: an accurate description of a microprocessor at the physical level would have to model physical device characteristics. The resulting simulation would be so slow that it would be completely useless for functional compatibility tests
- descriptions (modeling) are a compromise between accuracy and speed

HDLs

- what needs to be done using the language
  - functional (behavioral) description
  - simulation
  - verification
  - logic synthesis
- candidates
  - APL, VHDL, Verilog, C, C++
- advantages / disadvantages
  - complexity of the entity being described
    - from components to systems
  - machine requirements and cost per seat
  - nature of resulting simulation
    - event driven, cycle-based, cycle accurate, trace-driven
  - the ECAD community

“Golden Representation”

- the HDL description of the microprocessor
- when changes are made to the “golden representation,” changes are made to all other representations to match it
- the flow is always one way, from the single “golden representation,” to the representations used by the various design teams

Design & Implementation Process

- Let’s return to the design & implementation process

Constraints

- instruction set architecture definition
- goals
  - cost, performance, power, thermal packaging, clock frequency, etc.
- process technology
- time-to-market
- resources
  - people, computers, CAD tools, etc

Validation and Testing

- the size of the problem
  - devices
  - components
  - collections of components
  - the microprocessor
  - platform
  - system
- design for testability
- self-test

Testing

- the design
  - verifying the design realizes the instruction set architecture without any error
- the logical implementation
  - verifying the logical implementation realized the design without any errors
- the chip
  - verifying the physical device realizes the logical implementation without any errors

Testing

- functional (behavioral) testing
  - test vectors
- compatibility testing
  - compatibility with a previous generation processor

“The Process” (continued)

- behavioral simulation
  - cycle-based, event-driven, cycle-accurate
- model at gate and circuit levels
- gate and circuit level simulation and hardware emulation
- generate netlists and physical layout