ECE 551
System on Chip Design

Automated Design Fundamentals

Garrett S. Rose
Fall 2018

THE UNIVERSITY OF TENNESSEE KNOXVILLE

Outline

- Review of IC design basics
- Design partitioning and design approaches
- Introduction to semi-custom design approach
- Introduction to automated approach
- A snapshot of the tools you will use

Some General Terms

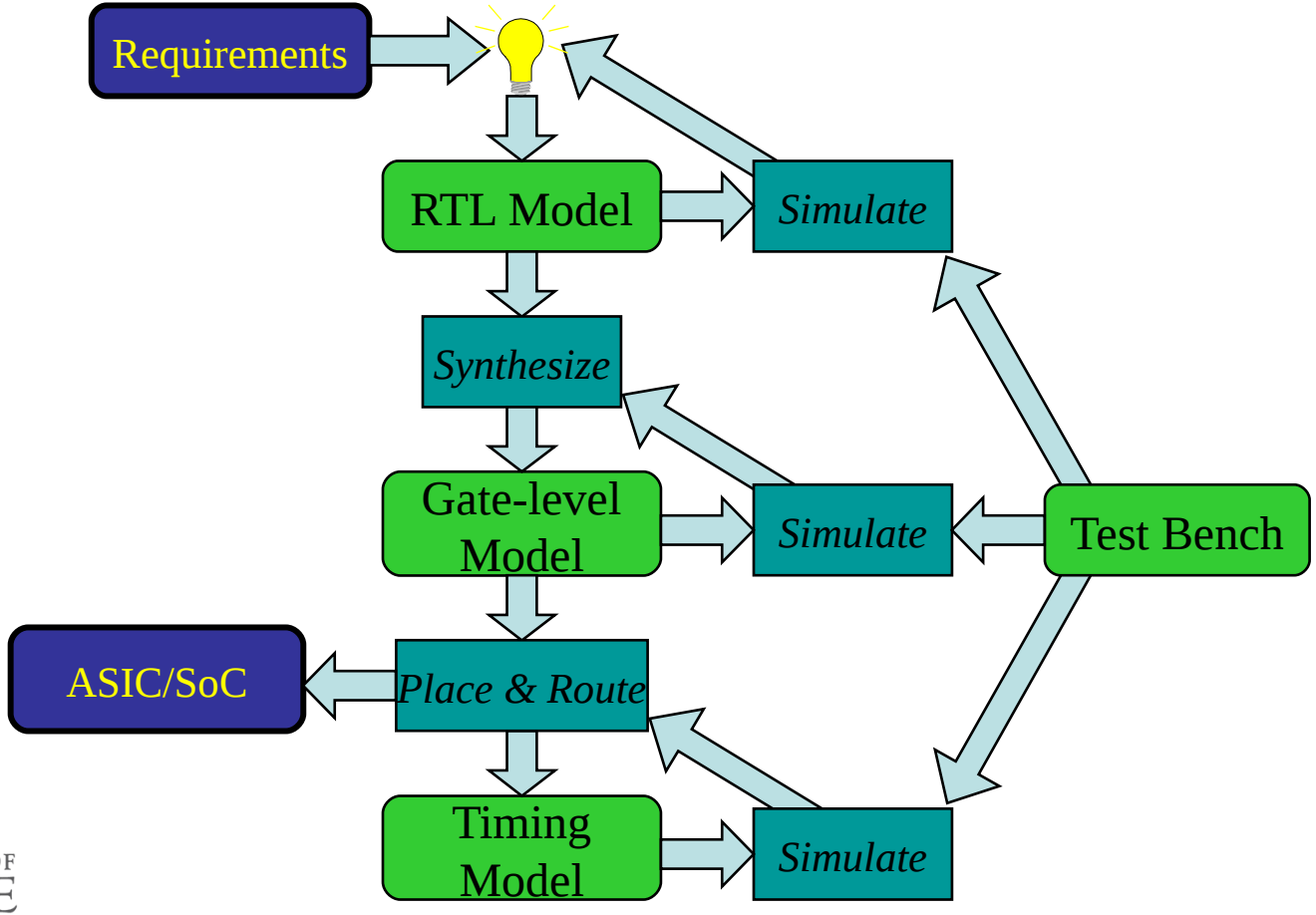
- VLSI – Very Large Scale Integration
- ASIC – Application Specific Integrated Circuit
- FPGA – Field Programmable Gate Array
- SoC – System on Chip
- NoC – Network on Chip

- HDL – Hardware Description Language (VHDL, Verilog, or SystemVerilog)
- RTL – Register Transfer Language

The ASIC

- The term “ASIC” has been applied to many design styles
- Technically, refers only to application specific circuits
  (i.e., any microchip you design yourself)
- Often, ASIC is used to refer to automated designs developed using some hardware description language
- Usually want an ASIC fast – clear design flows applied
- ASICs are low volume integrated circuits
- In recent years, ASICs are less common since an FPGA can be used to implement desired function
  -- Some might say... “FPGA is the new ASIC”

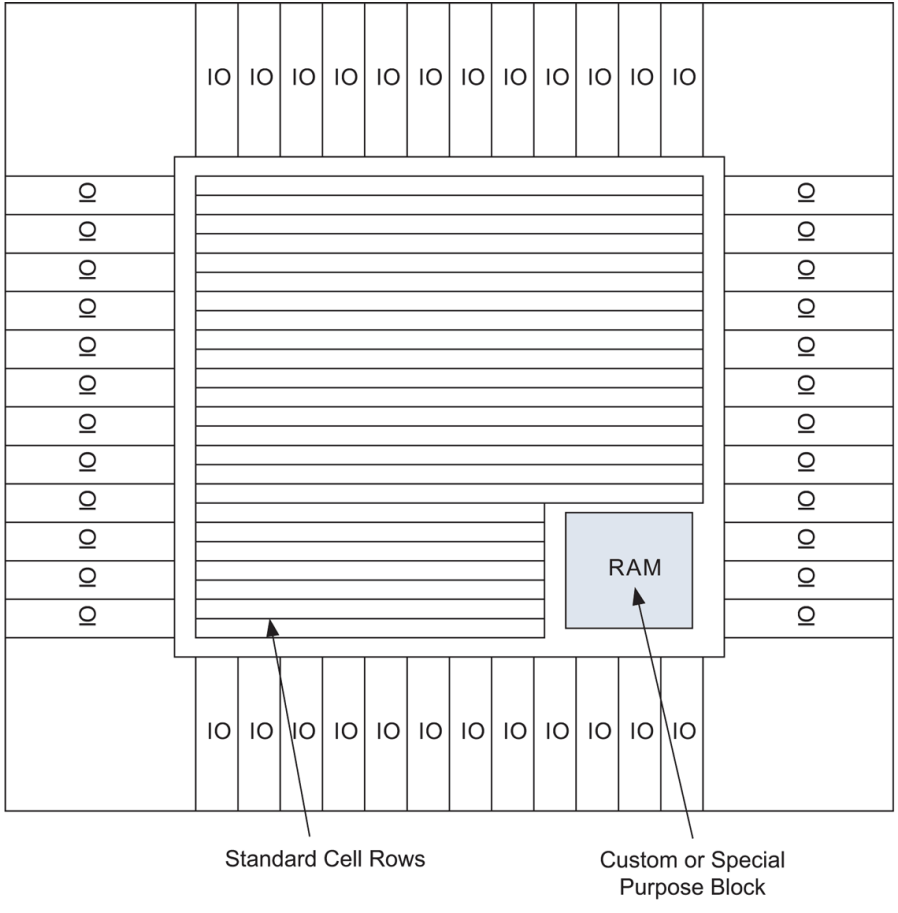
Typical ASIC Design Flow



ASIC Design Steps

- RTL design and verification – must write the HDL code
  - Can use VHDL, Verilog, or SystemVerilog
  - In this class, we will focuse on SystemVerilog
- Synthesis – compiles HDL design description into a *gate level netlist*
- Floorplanning – before place & route, must decide where functional modules will be placed on the die or FPGA
- Place & Route
  - Placement – determines where standard cells are placed
  - Routing – adds wires (configures switch blocks) connecting gates to implement final design
- ***Every step must include simulation & verification***

Placement of Cells
ASIC Standard Cell View



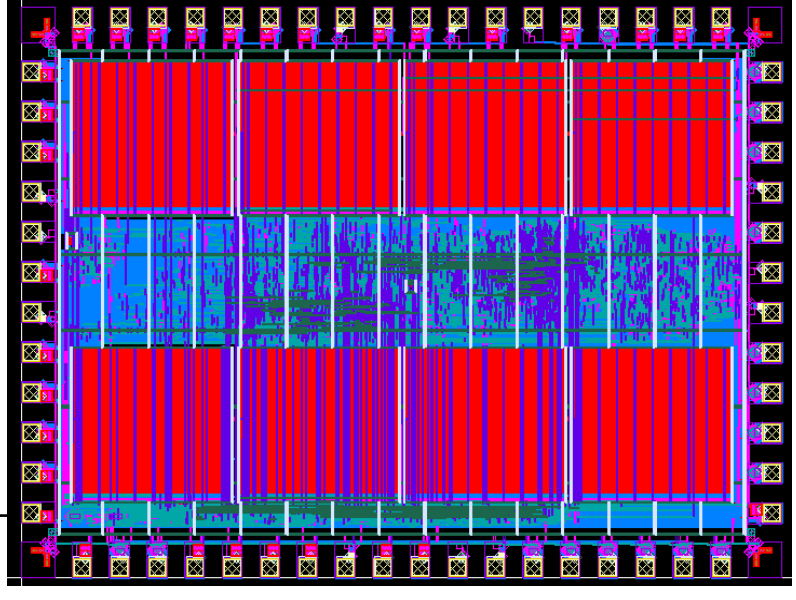
System on Chip (SoC)

- SoC implies a system of fairly high level blocks (e.g., memory, processors, DSP, etc.) integrated into one design
- SoC often refers to heterogeneous systems encompassing a great deal of functionality, often mixed signal
- Complex blocks are designed individually and not modified at the highest level – each block essentially a “black box”
- *Designers often use intellectual property (IP) cores for the building blocks of higher level designs*

- Repository of useful, yet free IP: www.OpenCores.org

System on Chip Design

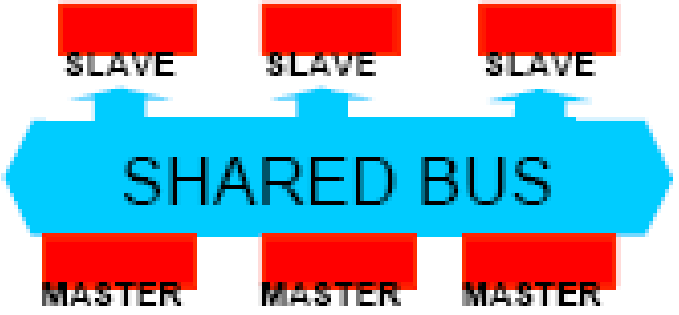
```
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
--
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity SRAD_Top is
```



- Start with high level HDL description
- Some blocks synthesized from HDL, some custom
- Research opportunities in power/temp. management, interconnection issues, etc.
- Example: an ultrasound image processing system

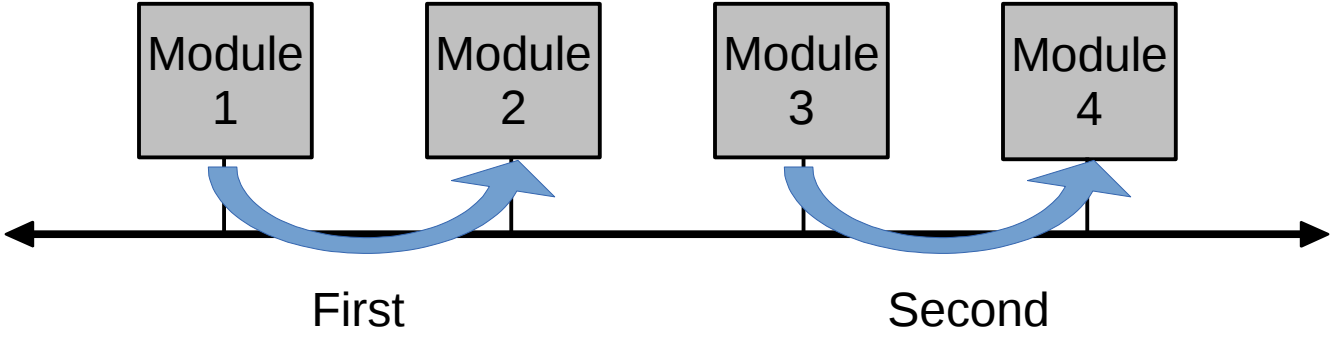
SoC Communication

- On-chip communication major design consideration for IP blocks
- Shared Bus (broadcast)
  - Low area
  - Poor scalability
  - High energy consumption
- Network on Chip (point-to-point)
  - Scalability
  - Low energy consumption
  - High area

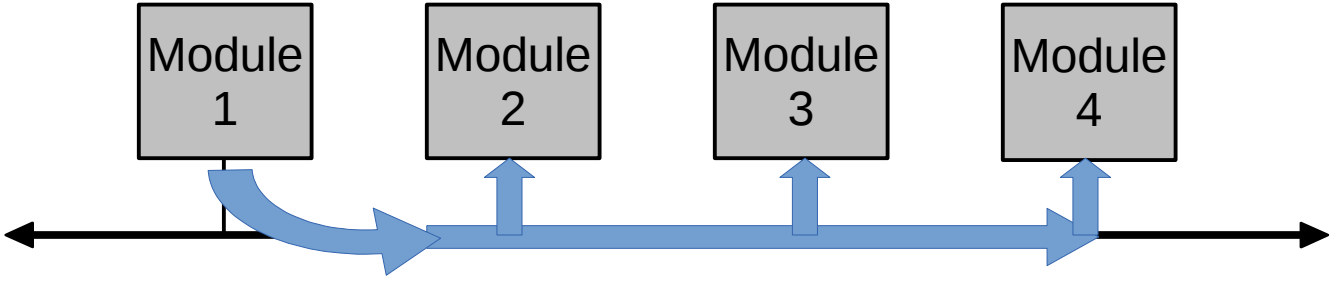


Bus Basics

- Bus communications follows strict order – serial nature

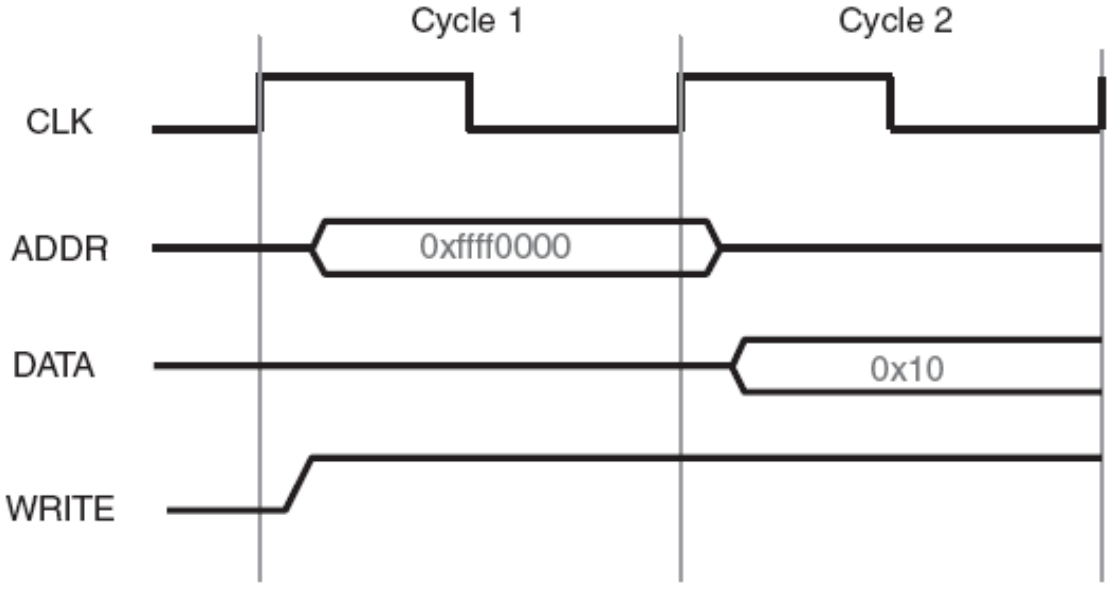


- Can broadcast – multiple destinations at the same time



Bus Basics

- Bus communication operates in units of cycles, messages and transactions

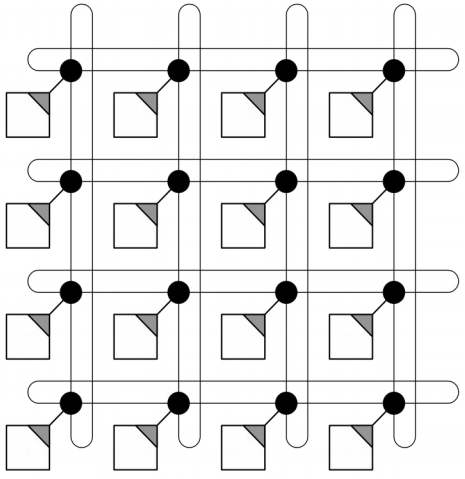


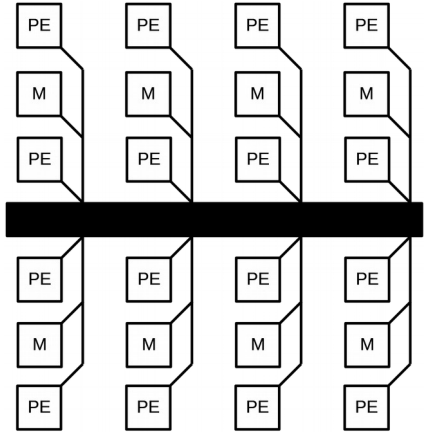
Network on Chip (NoC)

- As more and more complex systems are integrated, interconnection becomes a critical issue
- An NoC is literally a network (usually passing packets) on the chip not unlike the networks of macroscopic systems such as supercomputers, LANs, or the internet
- NoC has become more attractive since bus architectures only allow two devices to communicate at a time
- The on-chip network can be implemented in a variety of ways such as a simple crossbar, Clos, mesh, and so on

Network on Chip (NoC)

- Networks can be implemented on chip to circumvent issues:
  - Synchronization – NoC may be globally asynchronous
  - Multiple paths to avoid faults and allow many connections
  - Cool, low-power operation





Communications Standards

- Standards must be adopted to allow reliable communication between different IP blocks on the chip
- Bus architecture defines what IP blocks gain access to bus and handshaking / flow control mechanisms
- Some example bus standards:
  - AMBA Bus Architecture – ARM Microcontroller Bus Arch.
  - IBM CoreConnect
  - OpenCores Wishbone – used for most IP on www.OpenCores.org
- Socket-based standards are popular way to allow any generic IP block to be connected to virtually any bus (or network) via a socket adapter
  -- example: Open Core Protocol (OCP)

Early Design Considerations

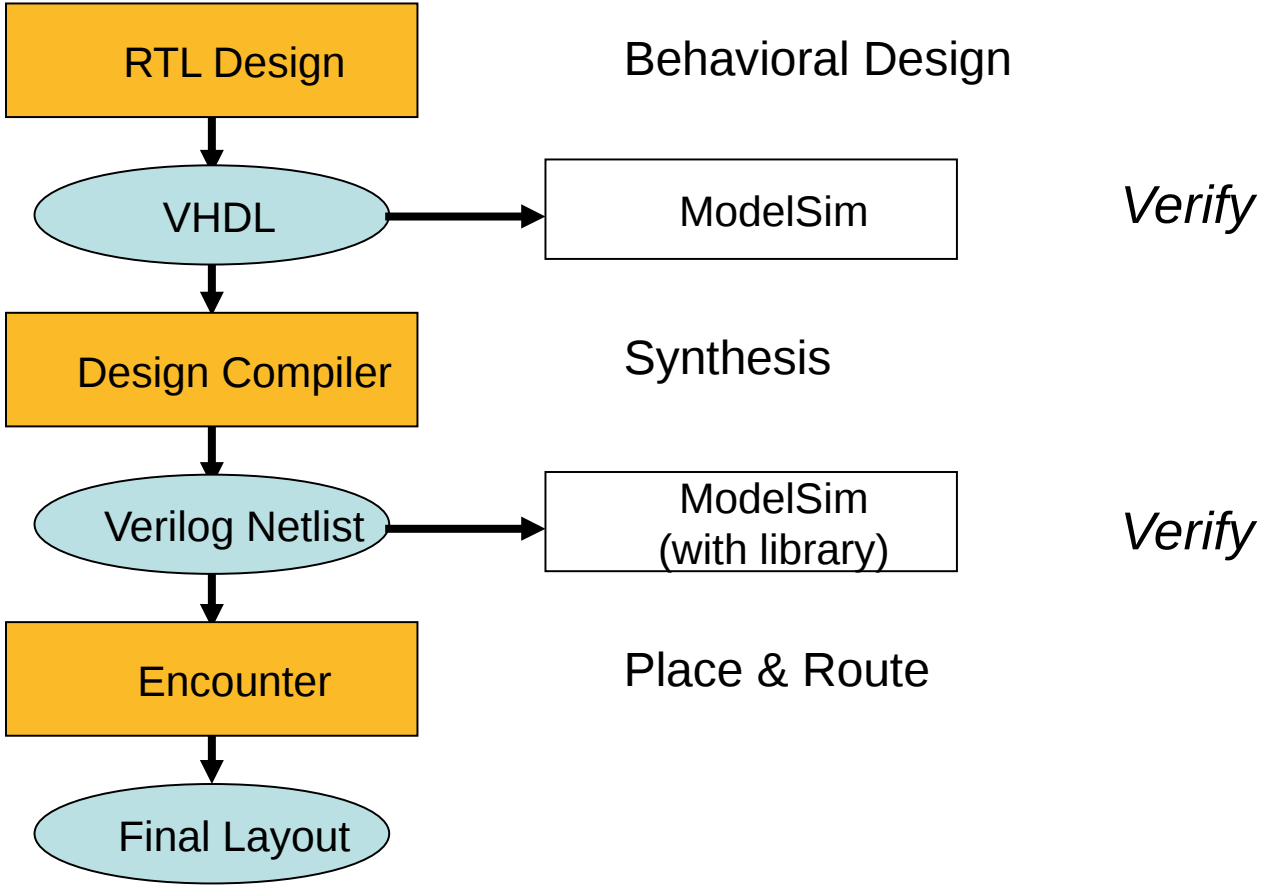
- First, need to partition the design
  - What, if anything will be custom?
  - What 3rd party IP can I use? Where will IP be used?
  - What HDL code do I need to develop?
- Determine communications standard
  - What am I familiar with? If AMBA, go with that...
  - What IP is available for a particular standard?
- Sketch high-level block diagram of the system
- Start working!

Design Flow Revisited

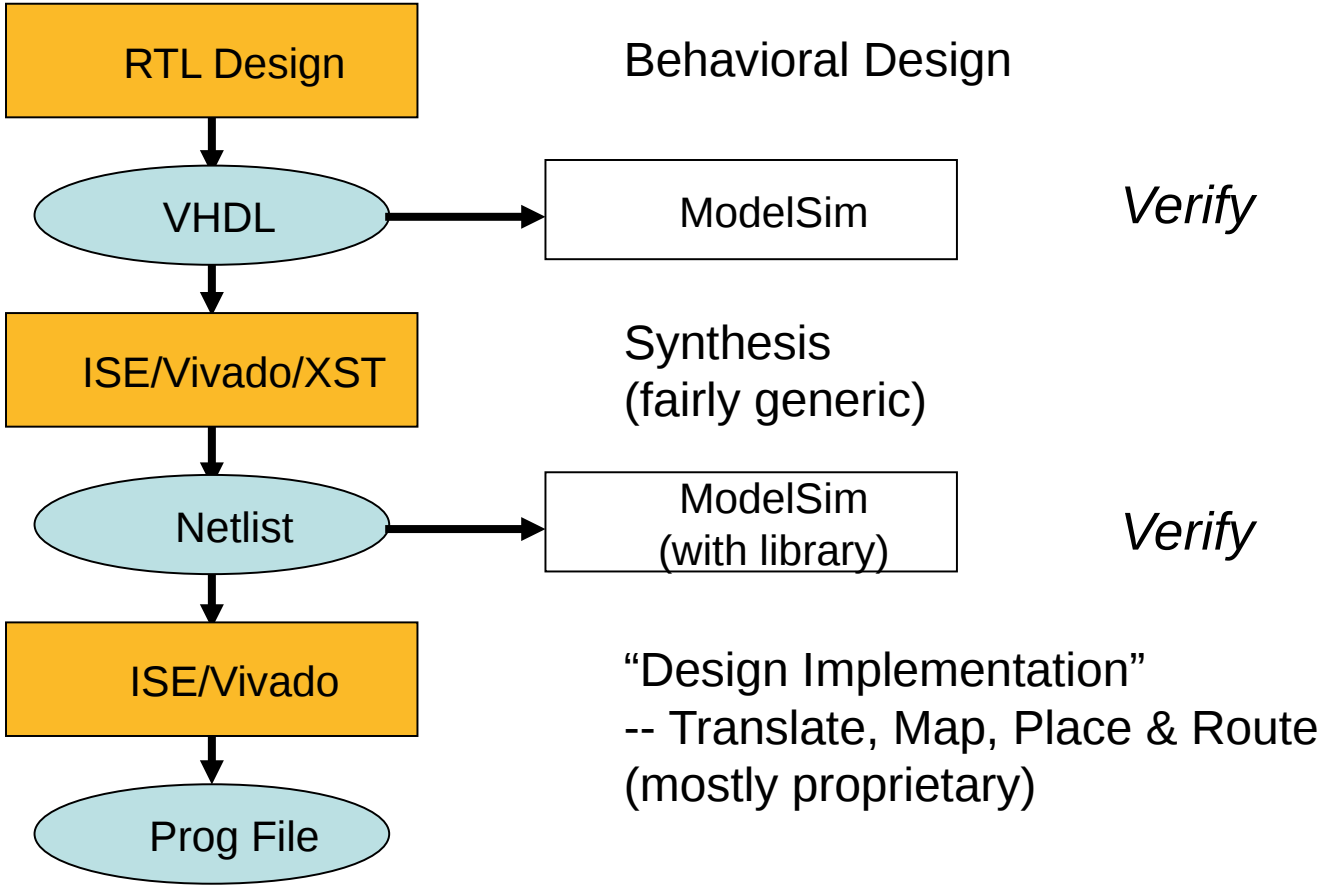
- Some tools useful in flow:
  - RTL Verification – ModelSim
  - Synthesis – Design Compiler
  - Place & Route – Cadence Encounter

- RTL (register transfer level) written in VHDL, Verilog, or SystemVerilog using any text editor (e.g., *gedit*) and verified with *NCLaunch*, *ISim* or *ModelSim*
- *Design Compiler* takes high level HDL code and synthesizes to a gate-level netlist (this is a Verilog netlist)
- *Encounter* takes the Verilog netlist from *Design Compiler* as input to place and route the final design

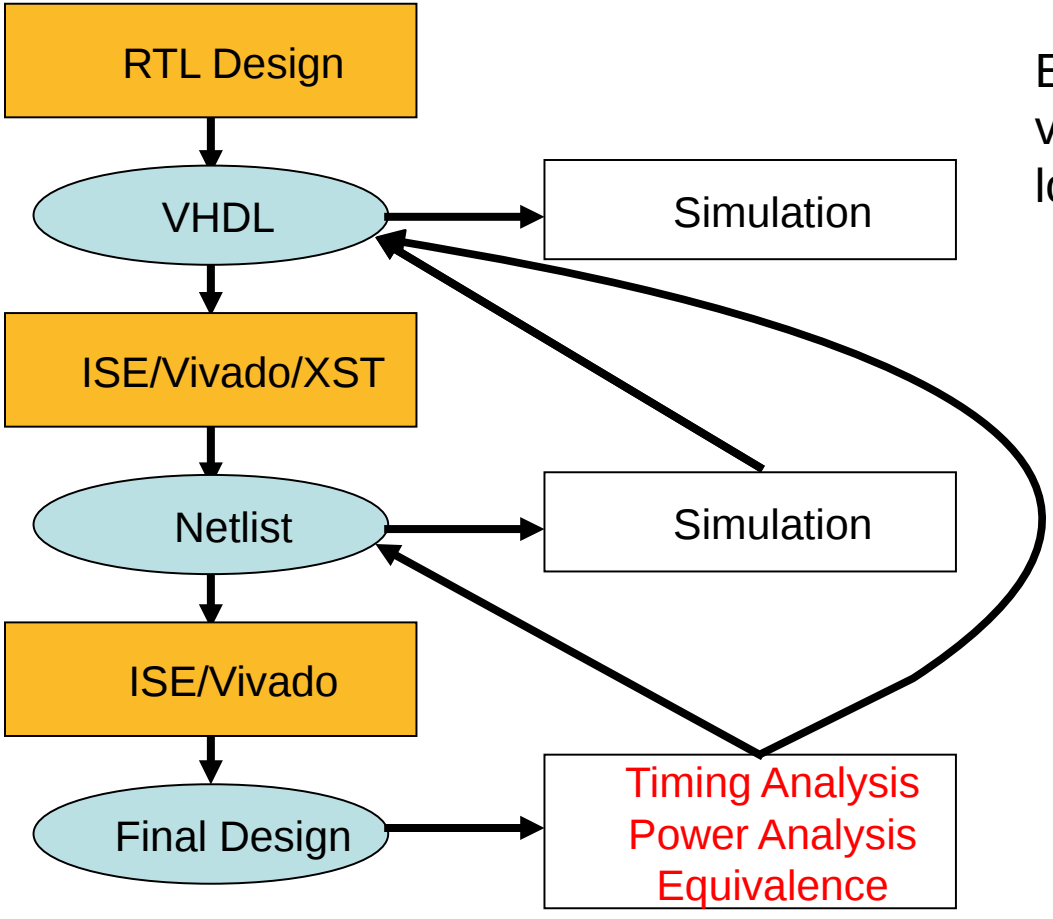
Design Flow Revisited – Silicon



Design Flow Revisited – FPGA



Still More...



Each stage has verification and loop(s) back

What the Designer Controls

- HDL description – likely starts high-level then becomes more structured with time
  - *It all starts here...*
- Constraints – extra files are included with HDL indicating performance targets to synthesis and other tools
  - Timing constraints needed to meet performance targets
  - Pin placement also falls under constraints
  - Can constrain tool to place blocks at certain locations
- CAD tool options – tools can be “tweaked” to use different algorithms, seed parameters, etc.

Coding for Circuits

- Design always begins with initial behavioral description – RTL code
- RTL description is very high level form written in some HDL,
  either VHDL, Verilog or SystemVerilog
- RTL describes the design in terms of microarchitectural components such as registers & ALUs
- Example of lower level HDL is the gate level netlist and even lower than that is transistor level
  -- netlists can be written in an HDL such as Verilog