Advanced Digital Circuit Design

Prof. Dr. Berna Örs Yalçın

Istanbul Technical University
Faculty of Electrical and Electronics Engineering
Department of Electronics and Communication Engineering
siddika.ors@itu.edu.tr

Outline

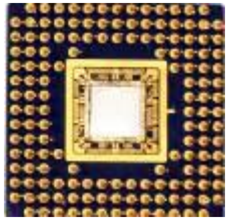
- Introduction to Digital Systems
- Hardware Description Languages (HDLs)
- Programmable Logic Devices
- Introduction to VHDL
  - HDL Models of Combinational Circuits
  - Test Benches for Combinational Circuits
  - HDL Models of Synchronous Sequential Logic
- State Reduction and Assignment
- Register Transfer Level & Design with ASM
- Simple Processor Design
  - RF & Datapath & Single Cycle Control
  - Basic Pipelining
- Asyncronous Sequential Digital Circuit Design

Grading

- Projects will be given every two weeks about the topics of related weeks.
  - Projects will be desribed by VHDL and implemented by using Xilinx Vivado
  - 7 projects in total
- A written final exam will take place during final period in academic semester.

Term grade=0,7*Average Project Grades+0,3*Final Exam Grade

Digital Systems / Motivation



- Digital systems surround us
  - Electronic system operating on 0s and 1s
  - Typically implemented on an Integrated Circuit (IC) – "chip"

- Desktop/laptop computers ("PCs") are the most popular examples
- Other increasingly common examples
  - Consumer electronics: Cell phones, portable music players, cameras, video game consoles, electronic music instruments, ...

  - Medical equipment: Hearing aids, pacemakers, life support systems, ...
  - Automotive electronics: Engine control, brakes, ...

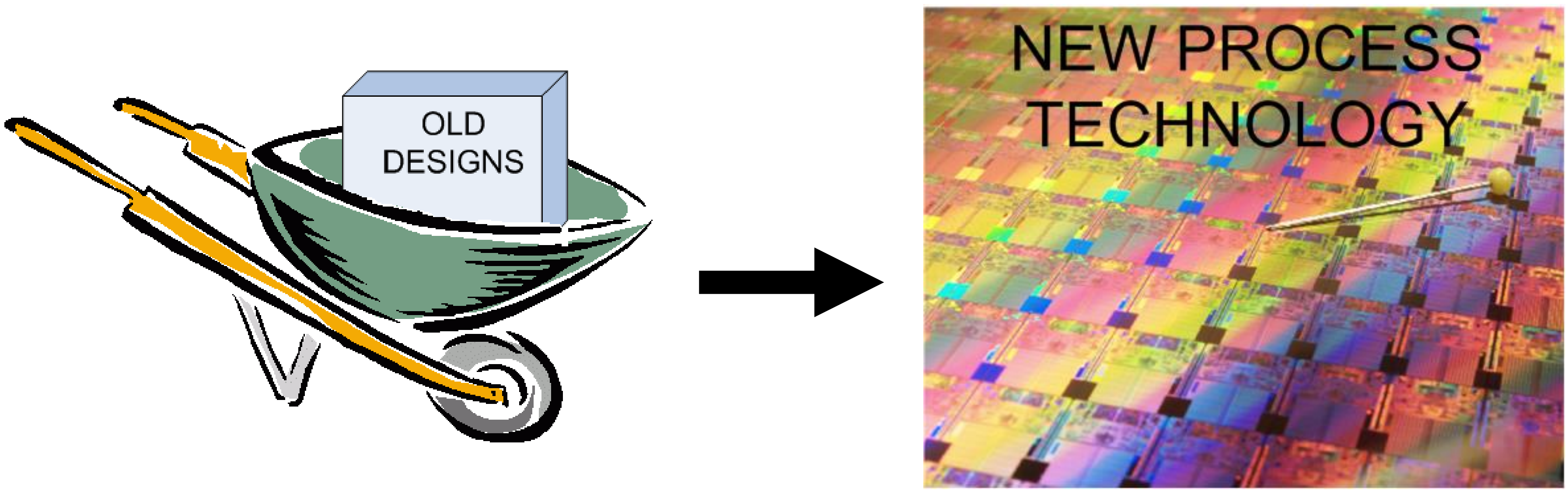
  - Military equipment
  - Networking components: Routers, switches, ...
  - Many, many more...

Why Digital?

- Ease of Design
  - Low Idea-to-Product Time

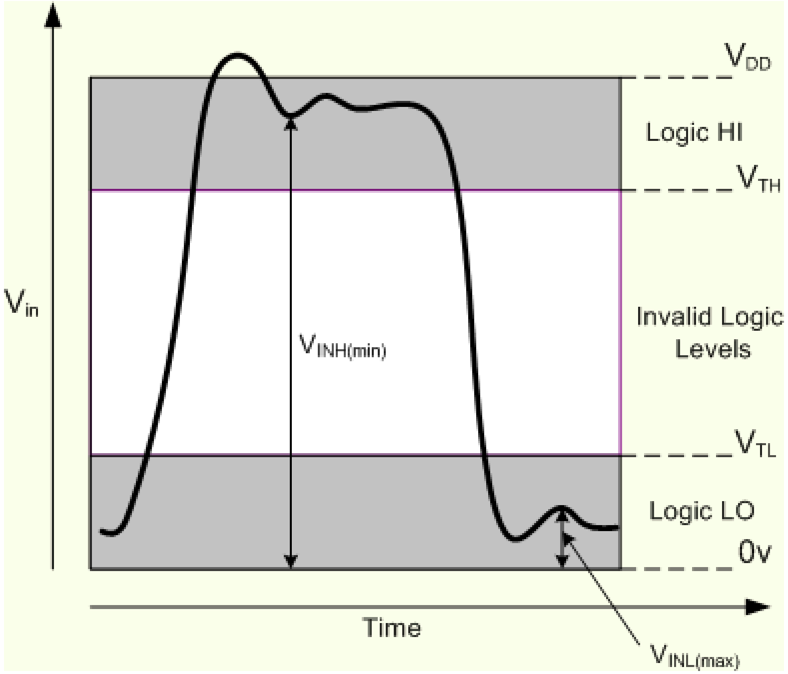
Why Digital?

- Ease of Design
  - Portability



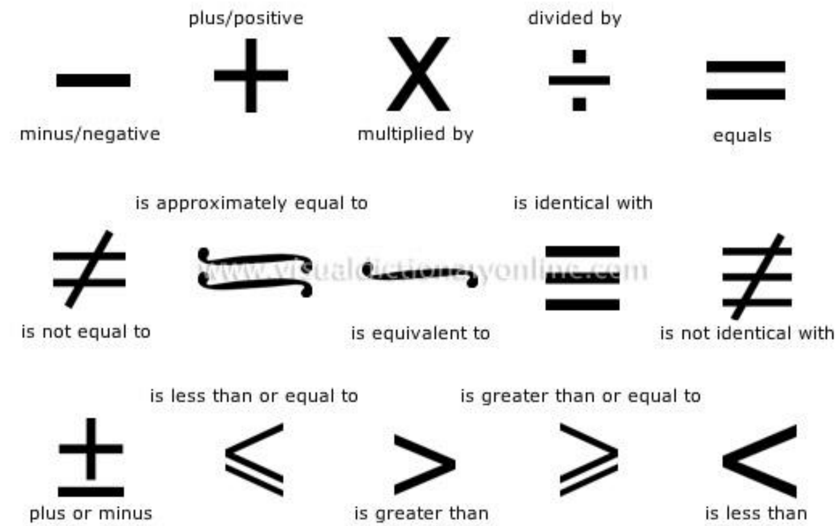
Why Digital?

- Ideal World
  - Immunity to Noise



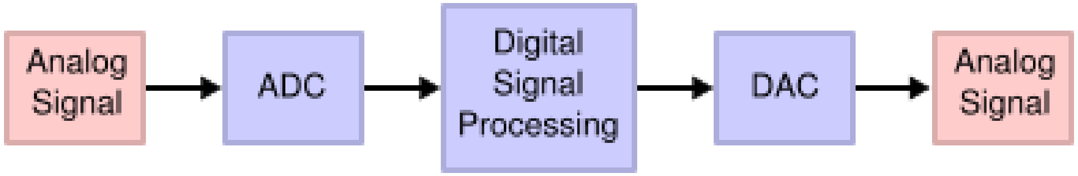
Why Digital?

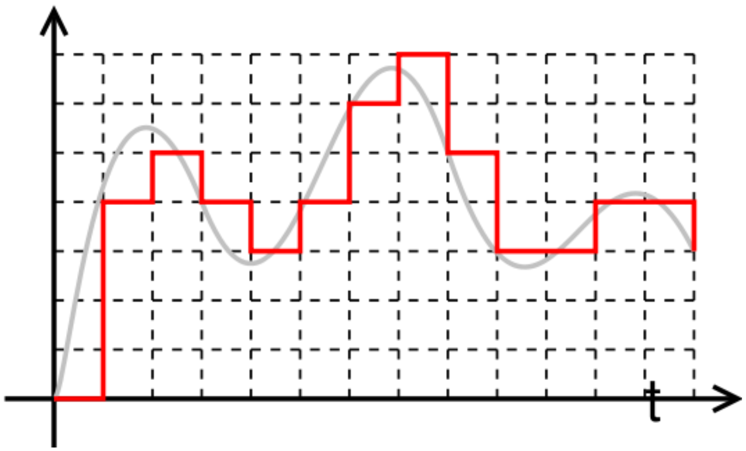
- Ideal World
  - Only mathematical and logical operations
    - Zero error for some operations
    - Customizable precision



Why Digital?

- Ideal World
  - Easy Signal Processing



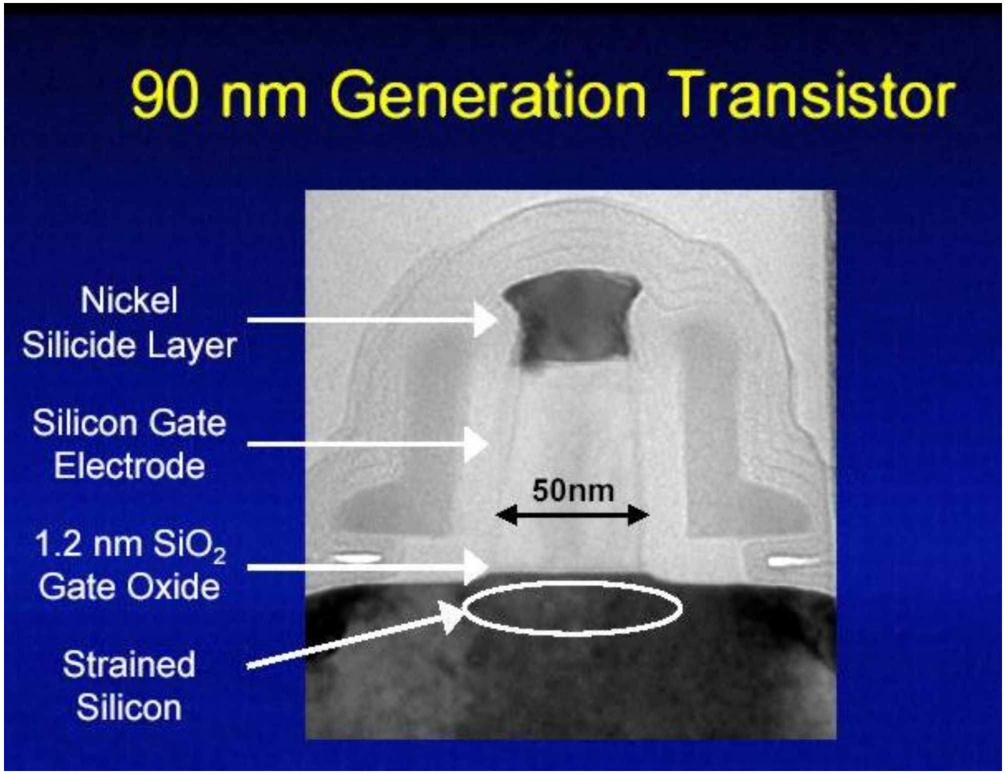


Why Digital?

- Programmability
  - Can be programmable on the fly
  - Can change the behavior completely

Why Digital?

- Technology (scaling) driven

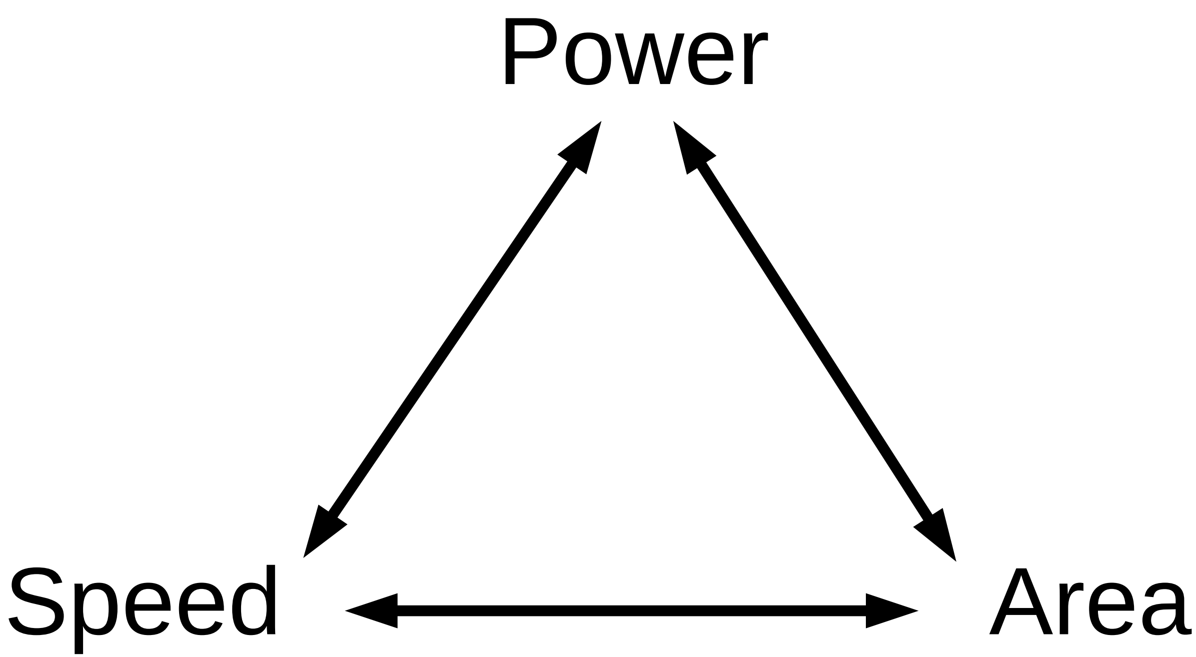


Why NOT Digital?

- Sound
- Electromagnetic Waves
  - Light etc.
- All other sensory data
  - Pressure
  - Temperature
  - Humidity etc.
- Short, The World is **ANALOG**

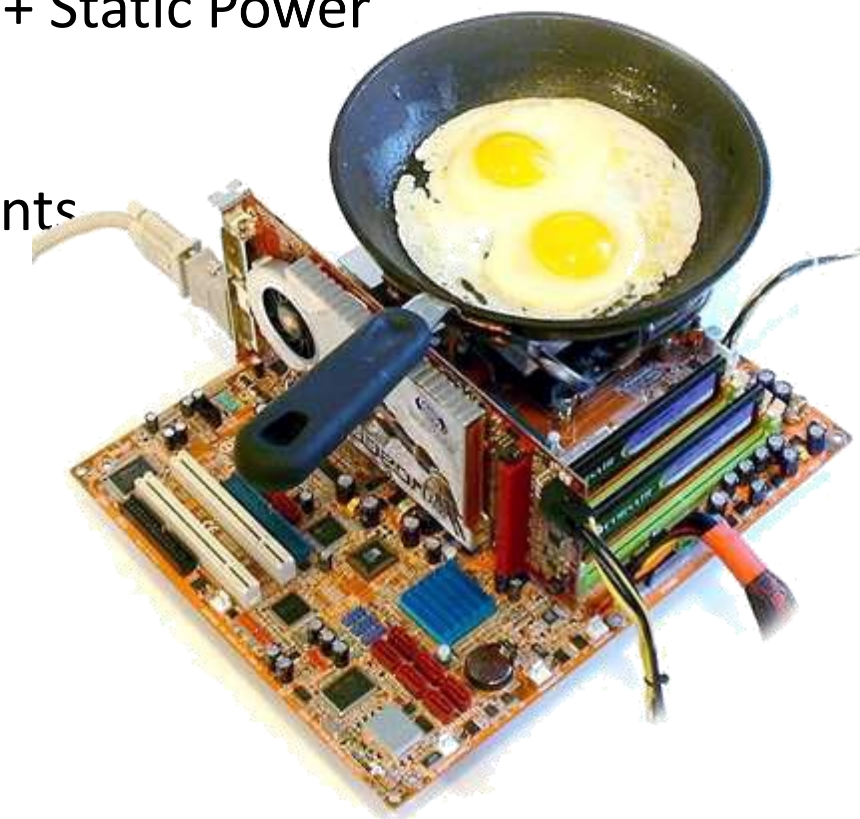
Considerations of Digital Technology

Trade-off Triangle



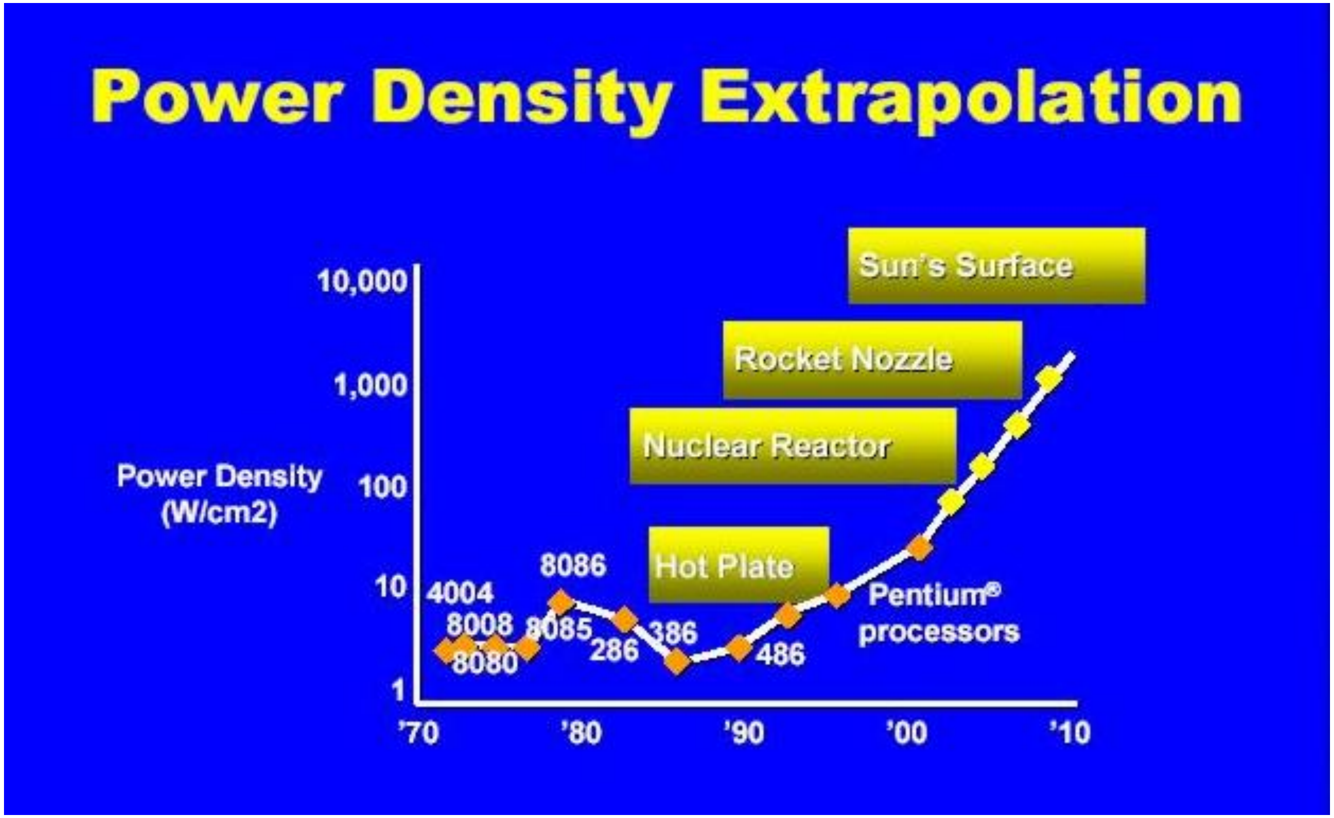
Limitations of Digital Technology

- Total Power = Dynamic Power + Static Power
- Dynamic Power → $C.f.V_{DD}^2$
- Static Power → Leakage Currents



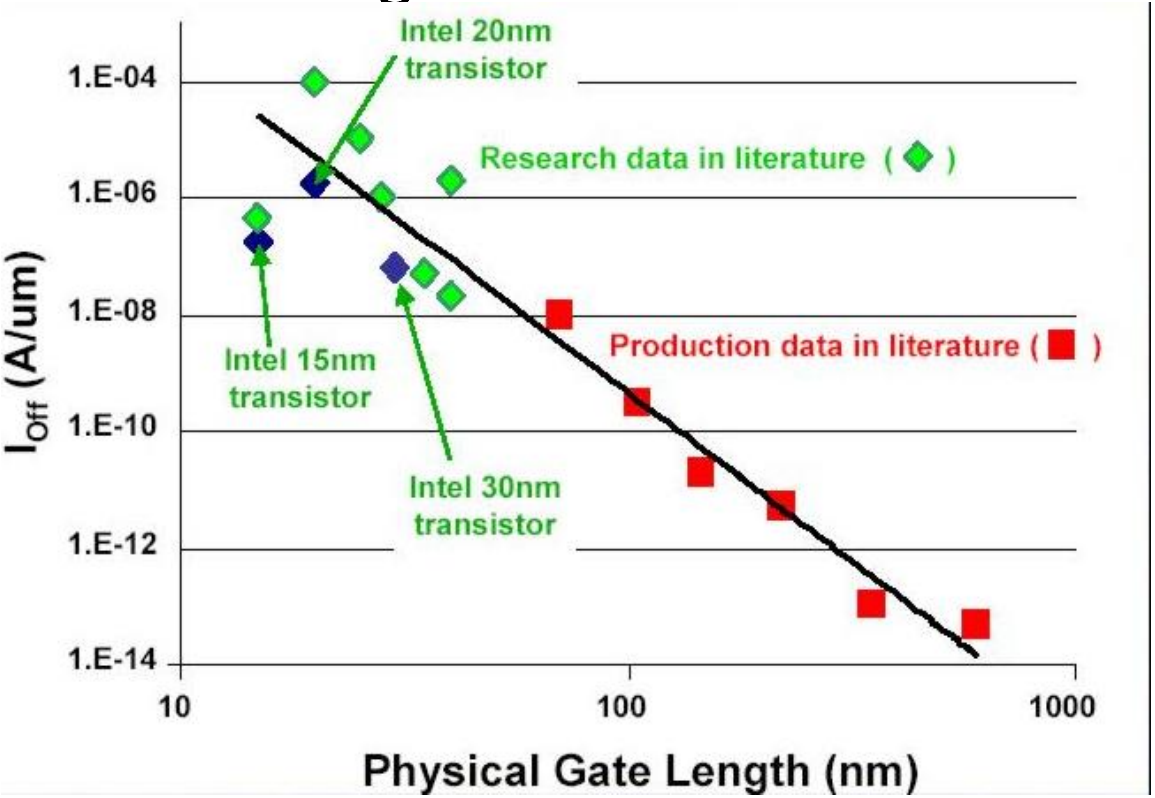
Limitations of Digital Technology

- Power Density = Power Consumption per Unit Area



Limitations of Digital Technology

- Subthreshold Leakage Currents



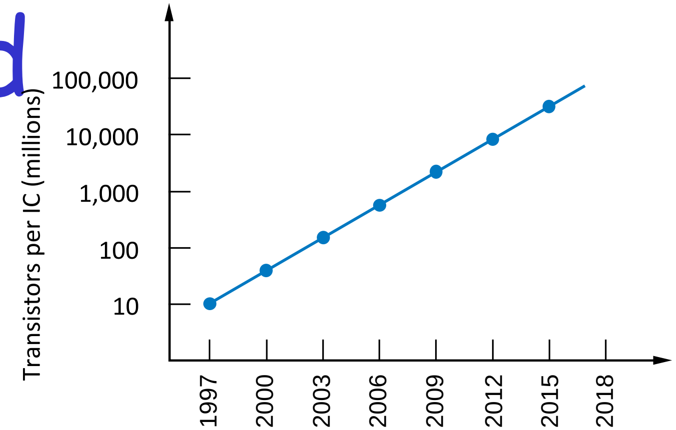
References

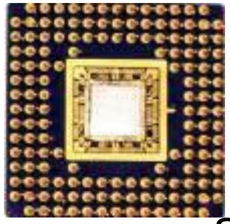
- http://download.intel.com/pressroom/kits/45nm/pin.jpg
- http://www.clipartheaven.com/show/clipart/tools_&_hardware/wheelbarrow-gif.html
- http://www.topnews.in/health/files/Aircraft-noise.jpg
- http://www.ami.ac.uk/courses/ami4822_dsi/u02/index.asp
- http://en.wikipedia.org/wiki/Digital_signal_processor
- http://upload.wikimedia.org/wikipedia/commons/thumb/9/9a/Digital.signal.svg/567px-Digital.signal.svg.png
- http://upload.wikimedia.org/wikipedia/en/2/24/Lenna.png
- http://ixbtlabs.com/articles2/intel-65nm/
- http://www.tomshardware.com/reviews/cheap-thrills,1335.html

Hardware Description Languages (HDLs)

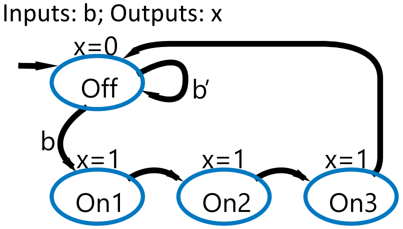
Digital Systems and HDLs

- Typical digital components per IC
  - 1960s/1970s: 10-1,000
  - 1980s: 1,000-100,000
  - 1990s: Millions
  - 2000s: Billions
- 1970s
  - IC behavior documented using combination of schematics, diagrams, and natural language (e.g., English)
- 1980s
  - Simulating circuits becoming more important
    - Schematics commonplace
    - Simulating schematic helped ensure circuit was correct before costly implementation





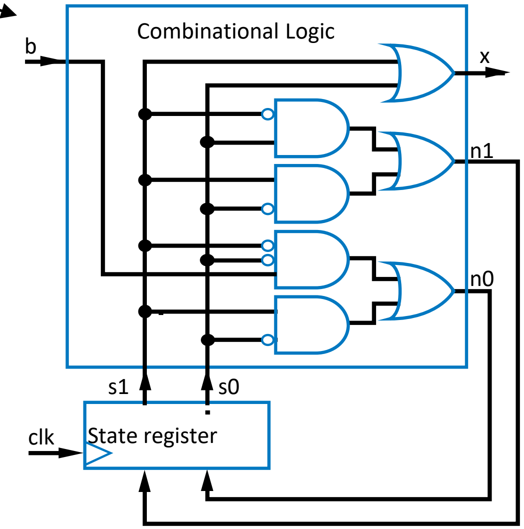
diagrams



natural language

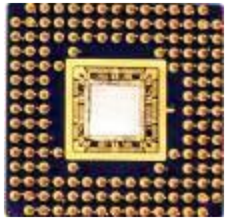
"The system has four states. When in state Off, the system outputs 0 and stays in state Off until the input becomes 1. In that case, the system enters state On1, followed by On2, and then On3, in which the system outputs 1. The system then returns to state Off."

schematics



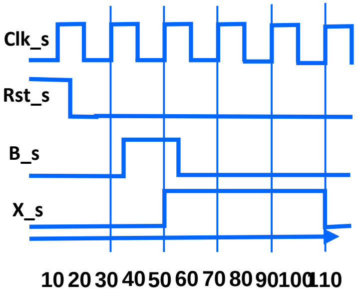
HDLs for Simulation

- Hardware description languages (HDLs) – Machine-readable textual languages for describing hardware
  - Text language could be more efficient means of circuit entry than graphical language



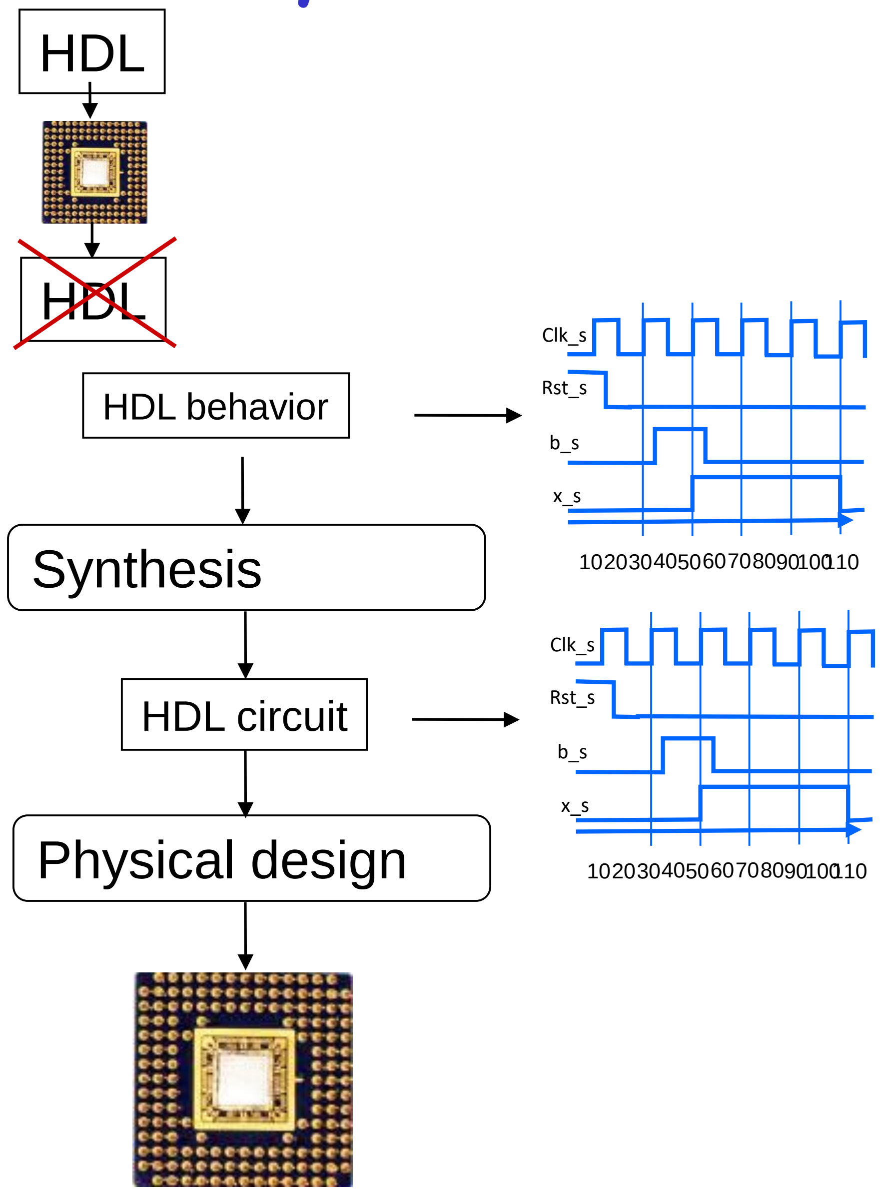
```
// CombLogic
process(State, B) begin
   case State is
      when S_Off =>
         X <= '0';
         if B = '0' then
            StateNext <= S_Off;
         else
            StateNext <= S_On1;
         end if;
      when S_On1 =>
         X <= 1;
         StateNext <= S_On2;
      when S_On2 =>
         X <= 1;
         StateNext <= S_On3;
      when S_On3 =>
         X <= 1;
         StateNext <= S_Off;
   end case;
end process;
```

Simulation



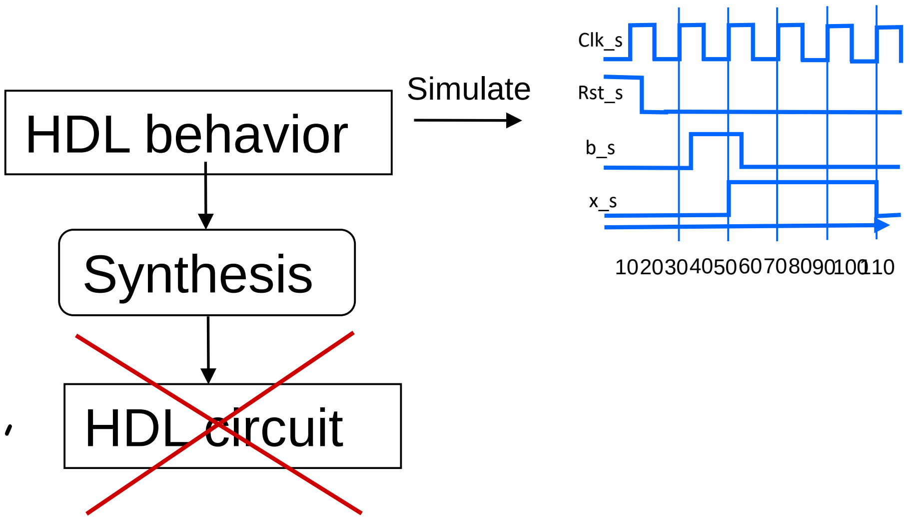
HDLs for Design and Synthesis

- HDLs became increasingly used for *designing* ICs using top-down design process
  - Design: Converting a higher-level description into a lower-level one
  - Describe circuit in HDL, simulate
    - Physical design tools automatically convert to low-level IC design
  - Describe behavior in HDL, simulate
    - e.g., Describe addition as $A = B + C$, rather than as circuit of hundreds of logic gates
      - Compact description, designers get function right first
    - Design circuit
      - Manually, or
      - Using synthesis tools, which automatically convert HDL behavior to HDL circuit
      - Simulate circuit, should match



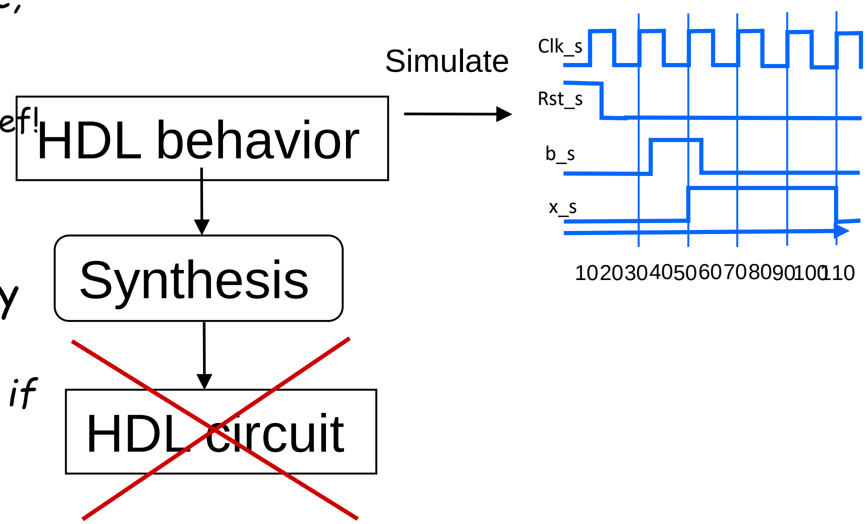
HDLs for Synthesis

- Use of HDLs for synthesis is growing
  - Circuits are more complex
  - Synthesis tools are maturing
- But HDLs originally defined for simulation
  - General language
  - Many constructs not suitable for synthesis
    - e.g., delays
  - Behavior description may simulate, but not synthesize, or may synthesize to incorrect or inefficient circuit
- Not necessarily synthesis tool's fault!



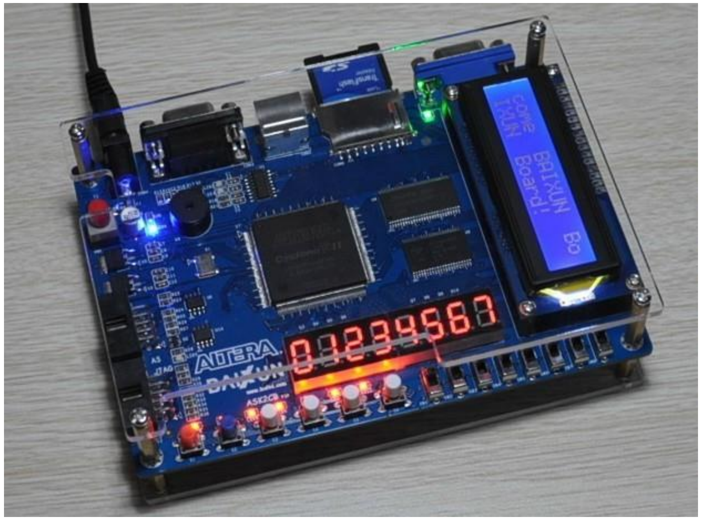
HDLs for Synthesis

- Consider the English language
  - General and complex; many uses
  - But use for *cooking recipes* is greatly restricted
    - Chef understands: *stir, blend, eggs, bowl, ...*
    - Chef may not understand: *bludgeon, harmonic, forthright, castigate, ...*, even if English grammar is correct
      - If the meal turns out bad, don't blame the chef!
- Likewise, consider HDL language
  - General and complex; many uses
  - But use for *synthesizing circuits* is greatly restricted
    - Synthesis tool understands: *sensitivity lists, if statements, ...*
    - Synthesis tool may not understand: *wait statements, while loops, ...*, even if the HDL simulates correctly
      - If the circuit is bad, don't blame the synthesis tool!
  - This course emphasizes on the use of VHDL for design and synthesis

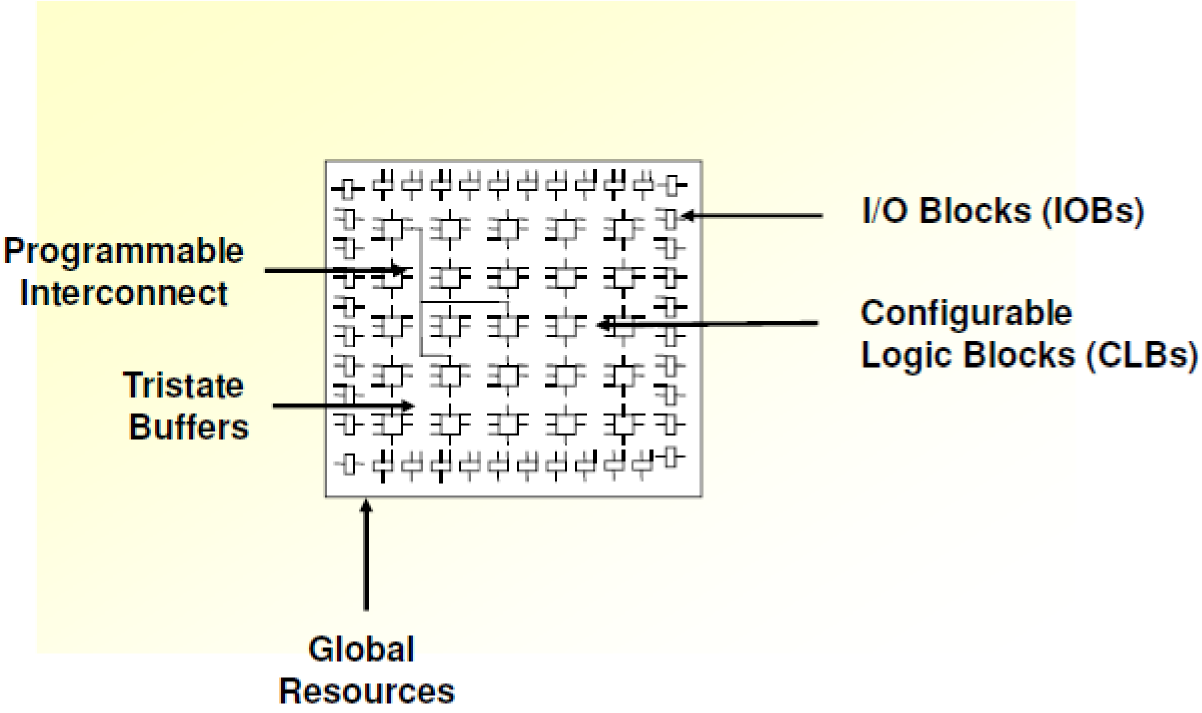


Programmable Logic Devices

- FPGAs – Field Programmable Gate Array
- Virtex 4, 5, 6, 7 !
- Spartan 3, Spartan 6
- Consist of configurable logic blocks
- Provides look-up tables to implement logic
- Storage devices to implement flip-flops and latches

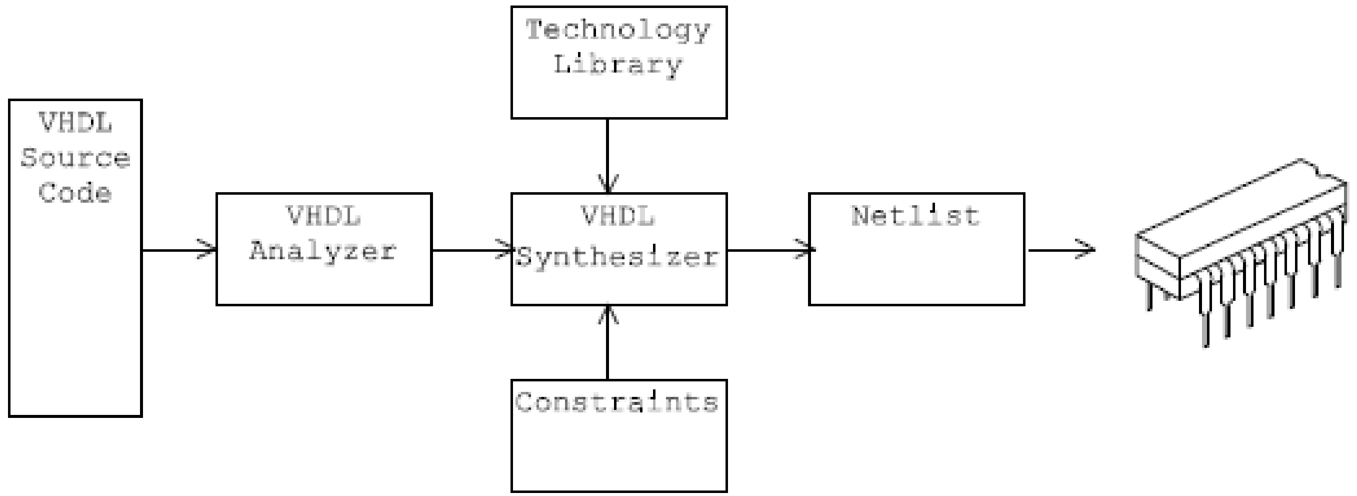


Overview of XILINX FPGA Architecture



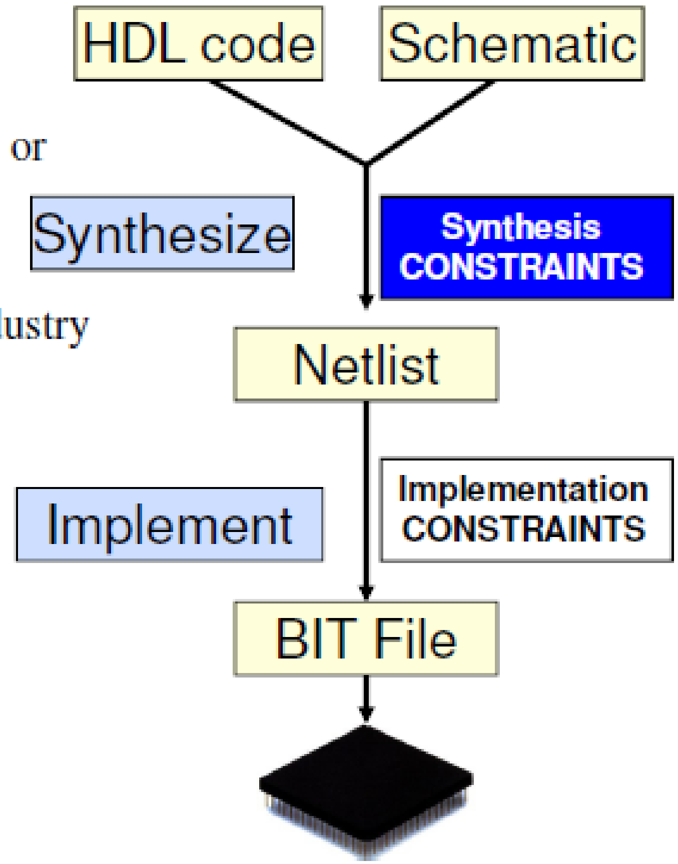
Logic Synthesis

A process which takes a digital circuit description and translates it into a gate level design, optimized for a particular implementation technology.

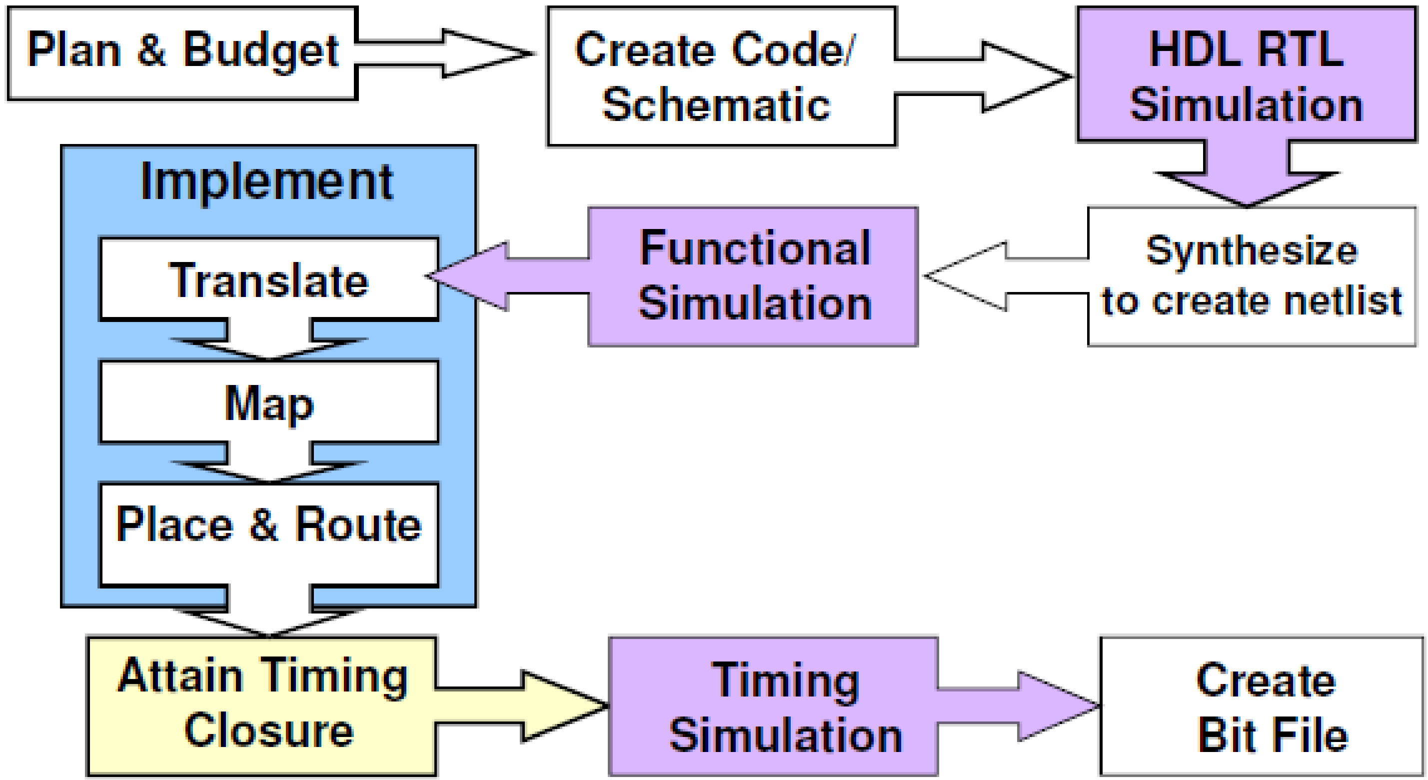


XILINX Design Process

- Step1: Design
  - Two design entry methods: HDL(Verilog or VHDL) or schematic drawings
- Step 2: Synthesize to create Netlist
  - Translates V, VHD, SCH files into an industry standard format EDIF file
- Step 3: Implement design (netlist)
  - Translate, Map, Place & Route
- Step 4: Configure FPGA
  - Download BIT file into FPGA



XILINX Design Flow (Xilinx)



Textbook & References

- Textbooks
  - M. Morris Mano and Michael D. Ciletti “Digital Design”, Fifth Edition, Prentice Hall, 2013.
  - Frank Vahid and Roman Lysecky ”VHDL for Digital Design”, Wiley 2007.

Xilinx WebPACK

[**Xilinx Vivado WebPACK**]

You must first register, and then download and install the WebPack.