University Of Diyala
College Of Engineering
Computer Engineering Department

# INTRODUCTION OF DIGITAL SYSTEM DESIGN

Asst. Prof. Dr.YASIR AMER ABBAS

THIRD STAGE

2020

# COURSE BASICS

- **Classes Morning : Monday 11:30AM- 01:30PM**
- **Classes Evening : Sunday 04:30 PM- 06:30PM**
- **Location : level 1, CSE103**
  - **Email:** dr.yasiral-zubaidi@uodiyala.edu.iq
- **Lecture : Google Classroom**

# COURSE REQUIREMENTS

❑References

- Digital Logic Circuit Analysis and Design, by Nelson, 2010.
- Digital Systems Design Using VHDL 2nd Edition, by Charles H. Roth, Jr. and Lizy Hurian John, Thomson.
- The Student's Guide to VHDL by Peter J. Ashenden, Morgan Kaufmann.

❑Computer Software:

❑Altera Quartus 13 or 9.1

https://www.intel.com/content/www/us/en/programmable/downloads/software/quartus-ii-we/120.html

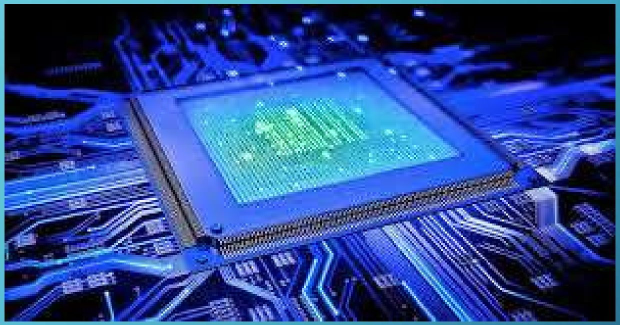
❑Xilinx ISE WebPACK & ModelSim PE Student Edition

http://www.xilinx.com/tools/webpack.htm

http://www.model.com

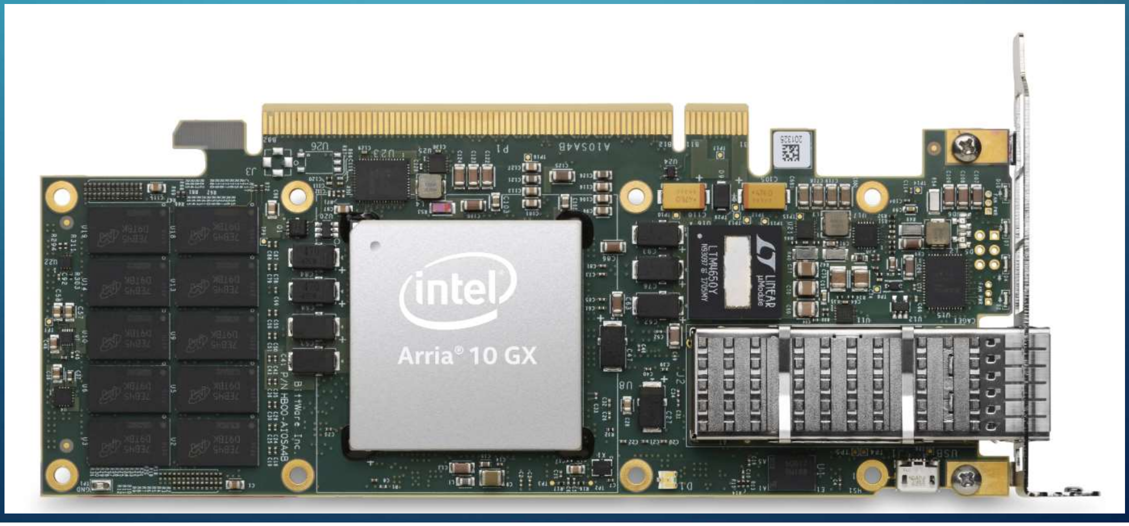
# GRADING

- Homework : 10 %
- Monthly Exam : 30 %
- Laboratory : 10 %
- Final Exam  50 %



# COURSE OUTLINE

- **Week 1: Introduction to Digital Systems Design**
- **Week 2: Complex Programmable Logic Devices (CPLDs)**
- **Week 3: Field Programmable Gate Array (FPGA)**
- **Week 4-5: Introduction to Sequential Logic Circuits**
- **Week 6-10: Analysis and Synthesis of Synchronous Sequential Circuits (Moore Model and Mealy Model)**
- **Week 11-12: Simplification of Sequential Circuits**
- **Week 13-15: Sequential circuits with programmable logic devices**



# SYSTEM

- System

  System: a set of related parts that actuate as a whole to achieve a given goal.

- System has:

  - Inputs.

  - Outputs.

  - Behaviour.

- Behaviour: a function that translates inputs to outputs.

# SYSTEM

- An entity consisting of Hardware and Software
  - ❑ Hardware:
    - High speed
    - Low power consumption
    - Less price (probably)
  - ❑ Software:
    - Flexible
    - Easy to modify and upgrade

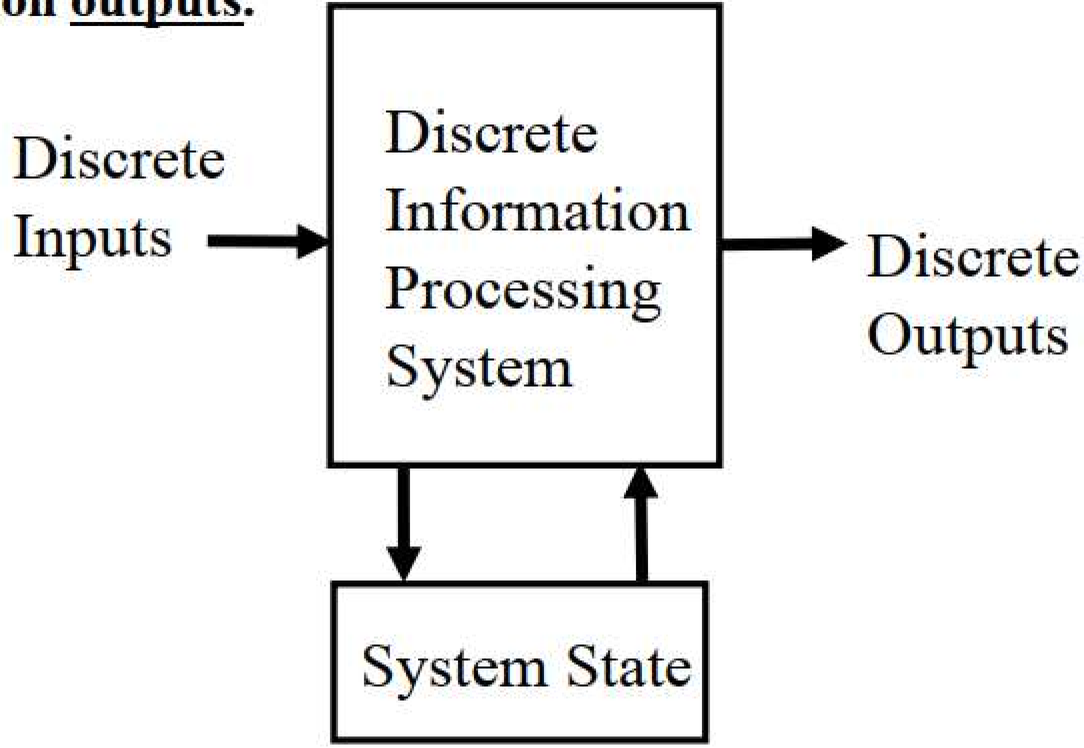
# DIGITAL CIRCUITS AND SYSTEMS?

- What are digital circuits and systems?
  - Any system that can be implemented with digital circuits
- What kinds of systems can be implemented with digital circuits?
  - This depends on the complexity of the problem to be solved
  - For those who has complexity proportional to polynomial functions, they are generally solvable with digital systems
    - E.g. The inversion of a N × N matrix

# What is a Digital System

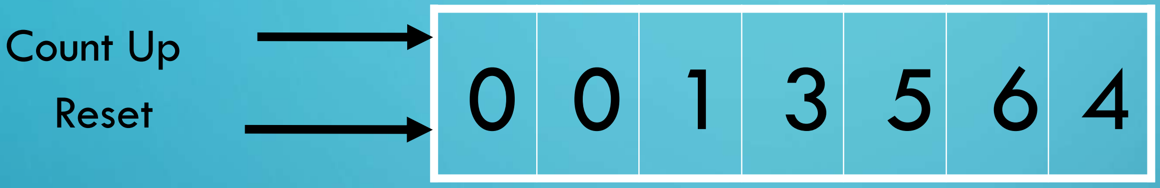
- **Structure:** **a collection of interconnected *digital* modules designed to perform a particular service or function**

- **Function:** **takes a set of *discrete* information inputs and discrete internal information (system state) and generates a set of discrete information outputs.**



# DIGITAL SYSTEM EXAMPLE:

**A Digital Counter (e. g., odometer):**



Count Up

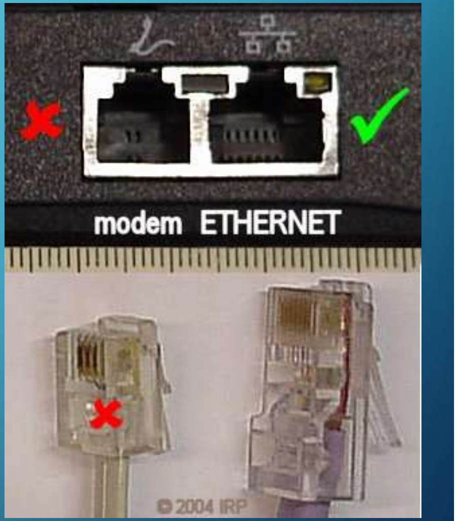
Reset

0 0 1 3 5 6 4

| | |
|---|---|
| **Inputs:** | **Count Up, Reset** |
| **Outputs:** | **Visual Display** |
| **State:** | **"Value" of stored digits** |

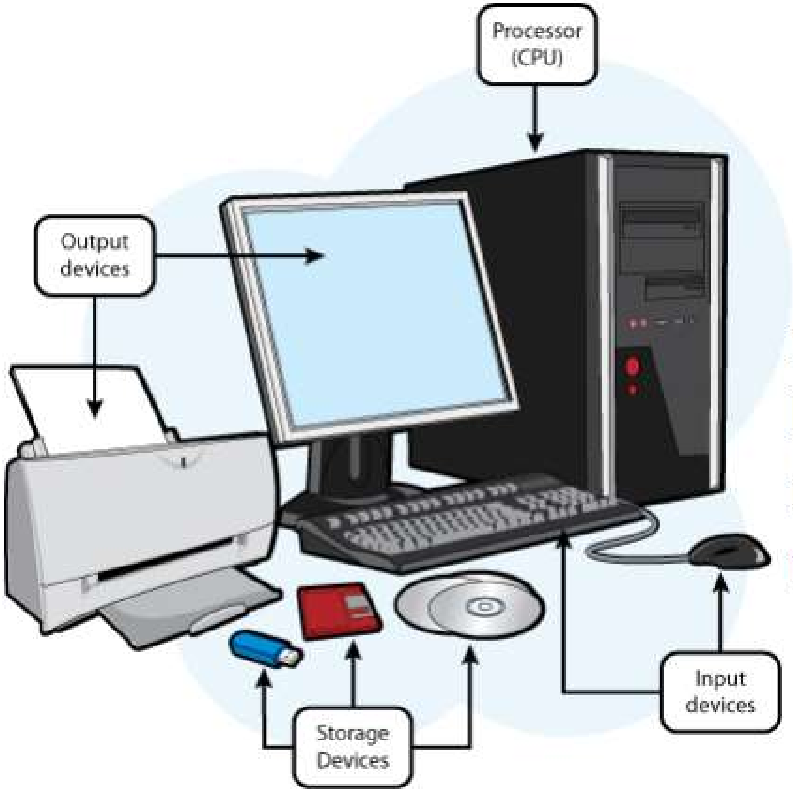
- What is a digital system?
  - A system that implements functions using digital logics

- What are the constitutional blocks of a digital system?
  - It has input interfaces
    - Key boards, antennas, wire jackets, sensors and microphones
  - It also has output interferences
    - Monitors, speakers, motors, printers, antennas, wire jackets and actuators
  - It definitely has signal processing units, or, in other words, arithmetic and logic units.



# A Digital Computer Example

Outputs: CRT, LCD, modem, speakers

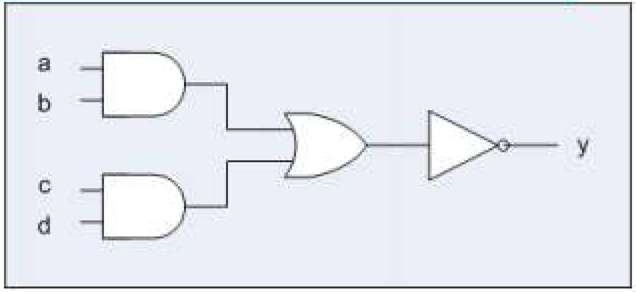


Inputs: Keyboard, mouse, modem, microphone

Synchronous or Asynchronous?

# Types of Digital Systems

- No state present
  - Combinational Logic System
  - Output = Function (Input)

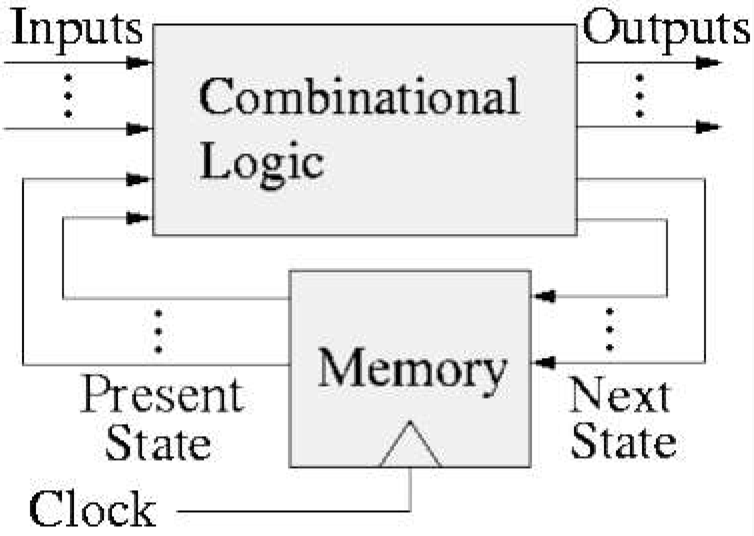


- State present
  - Sequential Logic System
    - State updated at *discrete* times

      => Synchronous Sequential System
    - State updated at *any* time

      =>Asynchronous Sequential System
  - Next State = Function (State, Input)
  - Output = Function (State, Input) – Mealy machine

    or Function (State) –Moore machine



# Digital System Modules

- **Low level digital modules**
  - Gates - AND, OR, NOR, etc.
  - Blocks - Adder, subtractor, shifter, etc.
- **High level digital modules**
  - PLDs (Programmable Logic Device)
  - ASICs (Application Specific Integrated Circuits)
  - Microprocessors/Microcontrollers

# Digital System Implementations

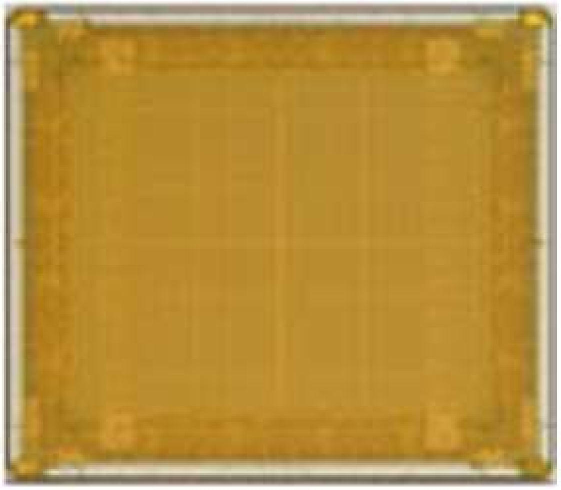
- Vary in granularity, flexibility, capability, etc.
  - **PCB** - printed circuit board
  - **FPGA** - field programmable gate array
  - **VLSI** - very large scale integration
  - **SoC** - system on a chip

# Digital Systems

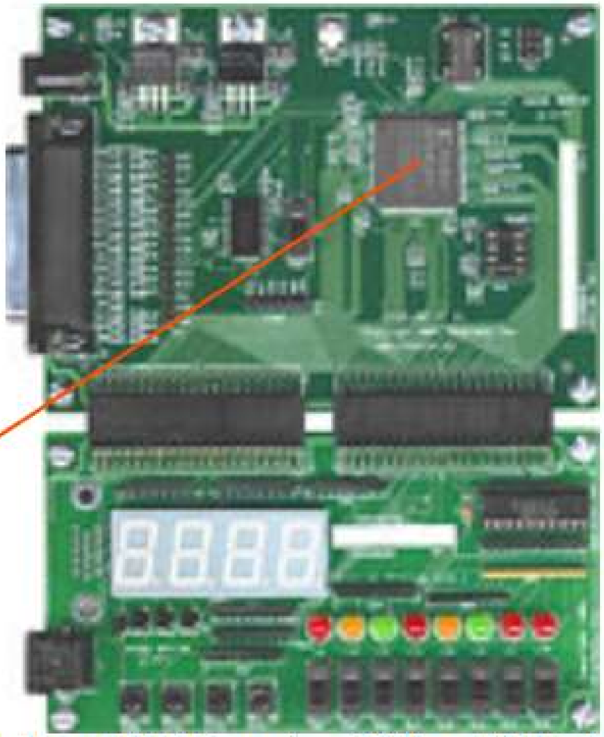
- **Printed Circuit Board**

# Digital Systems

- **FPGA**



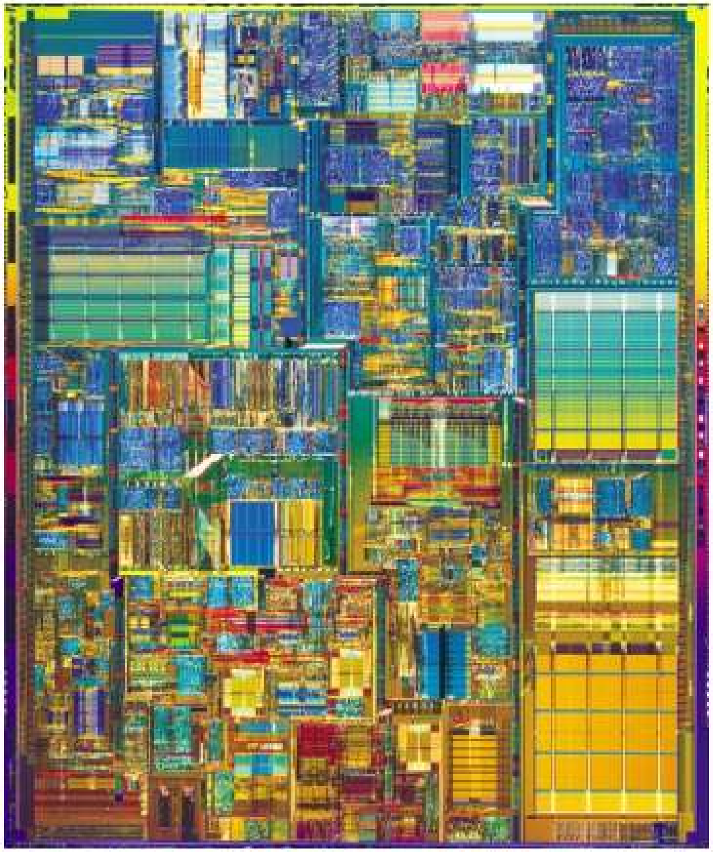
Xilinx Spartan3 FPGA



Digilent PCB using Xilinx FPGA

# Digital Systems

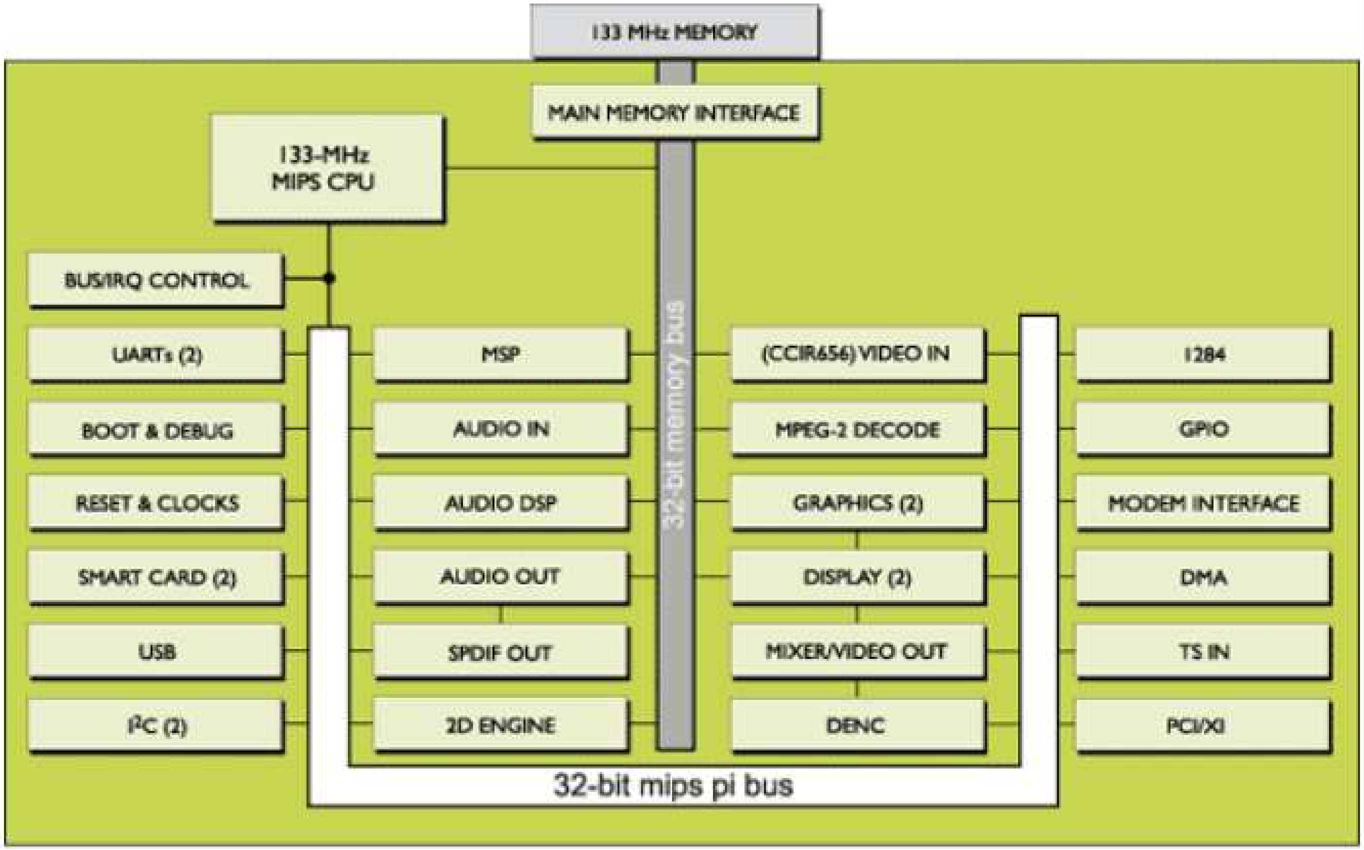
- VLSI



Intel Pentium IV

# Digital Systems

- **SoC**
  - Reusable IP
  - Embedded processor cores



Philips Nexperia PNX831 Set Top Digital Video Chip

- How to develop a digital system?
  - First, it is not only designing a digital circuit
  - A digital circuit is designed to realize functions that serve systems' requirements
  - A system may involves knowledge of control theories, communications, biology, mechanics, chemistries and etc.
- Digital circuits can be categorized into
  - General purpose circuits
    - Computers, Digital signal processors
  - Application specific circuits
    - Modems, GPS, cellphones

- How to develop a digital circuit?
  - We need knowledge about digital logic
  - We need tools to explore the ideas, to simulations the operational conditions, to realize circuits and to validate functionalities
  - We need platforms to field test, emulate, the designs
- Do we have a language that can specify these purposes?
  - This language is called hardware description language (HDL), e.g. VHDL and Verilog HDL
- In addition to language, we need translators, simulators, synthesizers and hardware evaluation and development systems

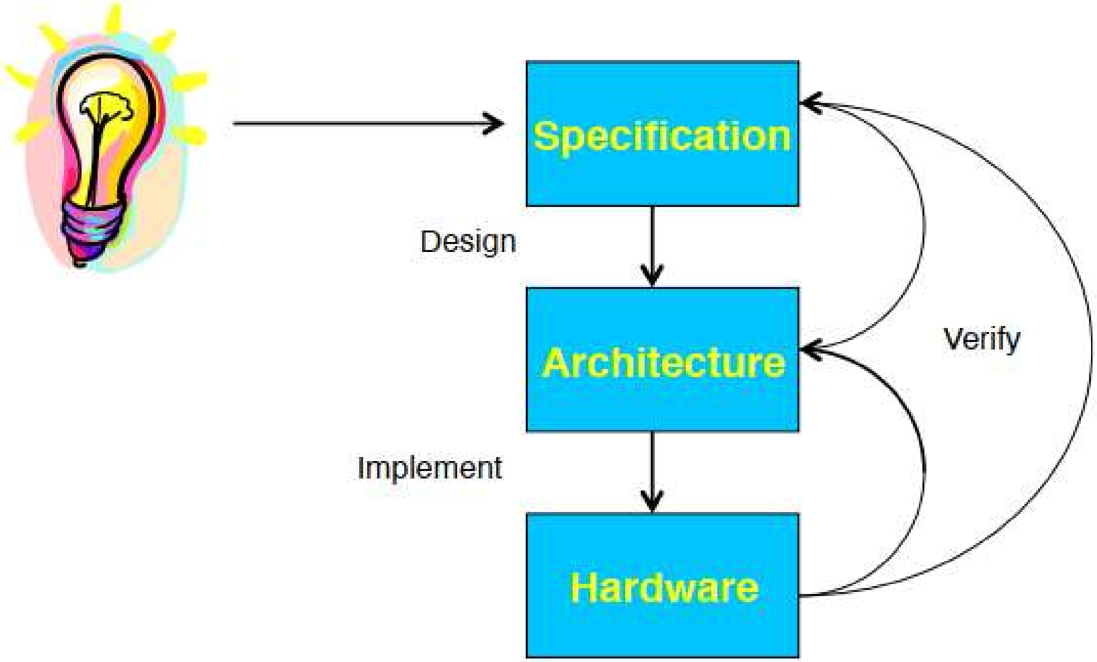
- Translators :
  - To compile VHDL or Verilog HDL languages into a language that can be understand by a computer
- Simulators:
  - To simulate the system response according to input signals
- Synthesizers:
  - To synthesize VHDL or Verilog language into digital circuits
- Validations:
  - Digital circuit development systems

- Digital circuits developments
  - There are many of them

Altera DE1-S0C

# What is Digital Systems Design ?

- Digital Systems Design is a process that entails a systematic development of an idea into an **architecture** that can be implemented digitally.



Specification

Design

Architecture

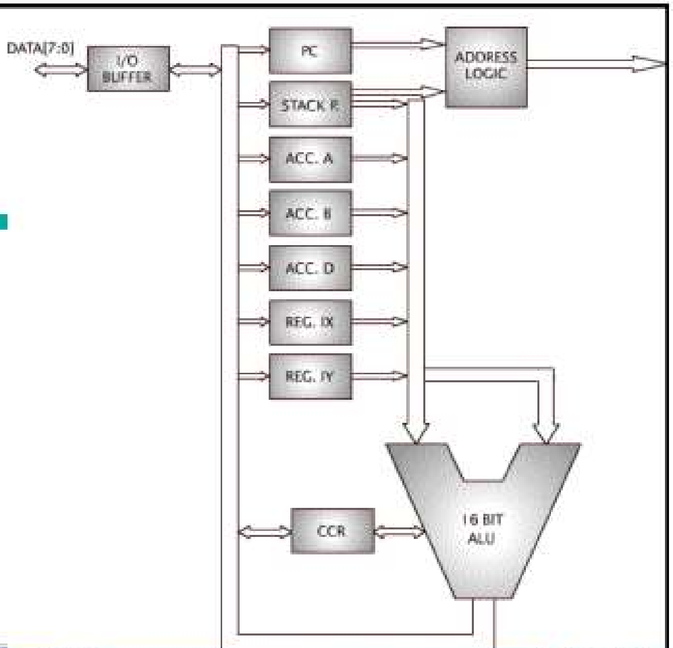
Implement

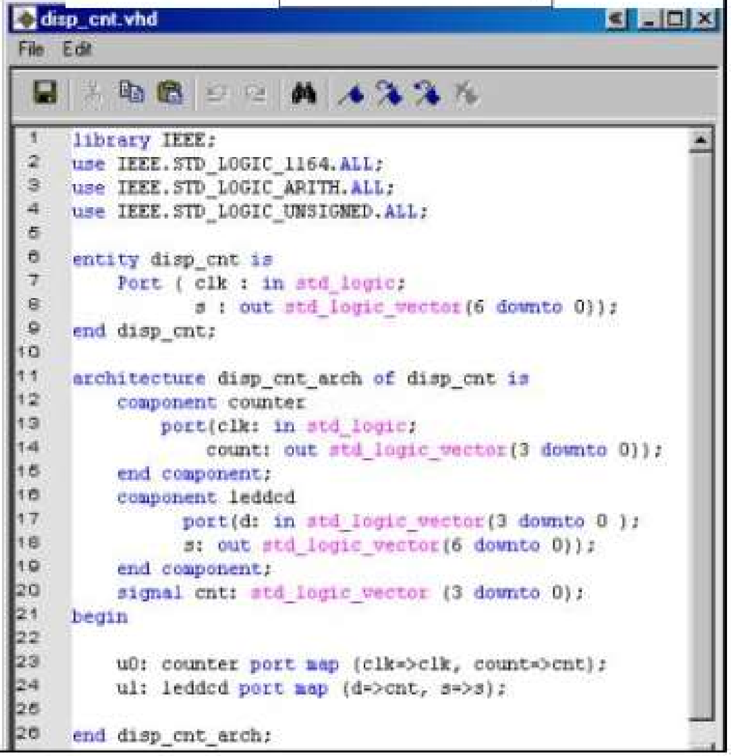
Hardware

Verify

# Specification

- Translation from idea into a **formal** description of behavior
- The highest level of abstraction is a declarative statement or written expression that specifies the design idea.
- Forms: Text description; Diagrams; Specialized languages (VHDL, Verilog, etc.)

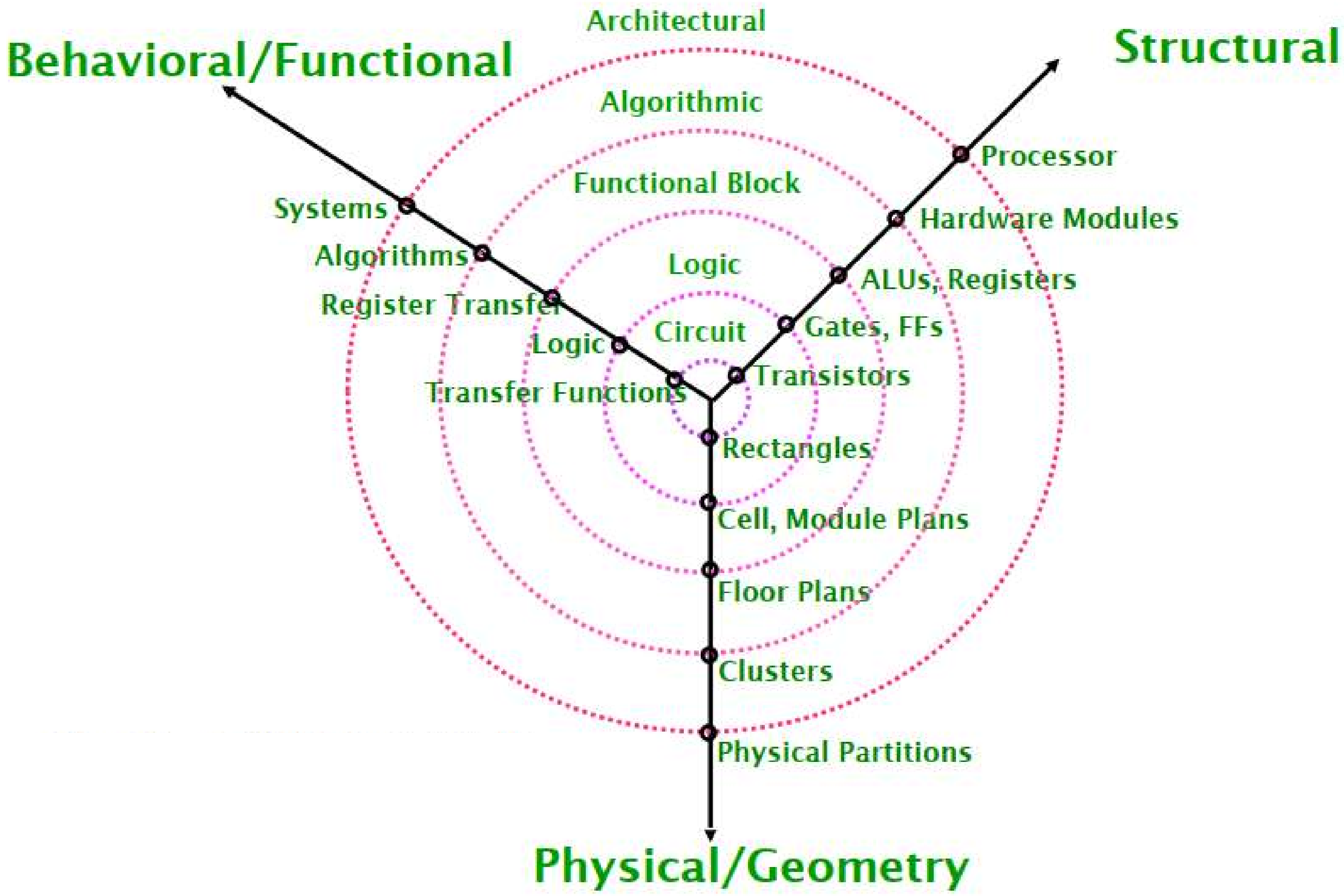




```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity disp_cnt is
    Port ( clk : in std_logic;
           s : out std_logic_vector(6 downto 0));
end disp_cnt;

architecture disp_cnt_arch of disp_cnt is
    component counter
        port(clk: in std_logic;
            count: out std_logic_vector(3 downto 0));
    end component;
    component leddcd
         port(d: in std_logic_vector(3 downto 0 );
         s: out std_logic_vector(6 downto 0));
    end component;
    signal cnt: std_logic_vector (3 downto 0);
begin

    u0: counter port map (clk=>clk, count=>cnt);
    u1: leddcd port map (d=>cnt, s=>s);

end disp_cnt_arch;
```

# Digital Systems Modeling

- Gajski and Kuhn Y Chart



## Hardware Description Languages

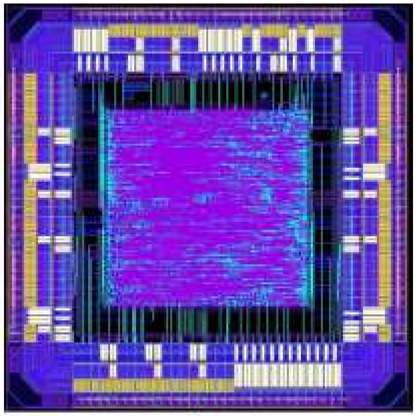
- Two primary choices
  - VHDL (VHSIC (Very high-speed IC) hardware description language)
  - Verilog HDL
- Can be used for behavioral specification, architectural definition, implementation, and verification
- Other HDLs used in specific areas include SystemC, HandelC, Rosetta

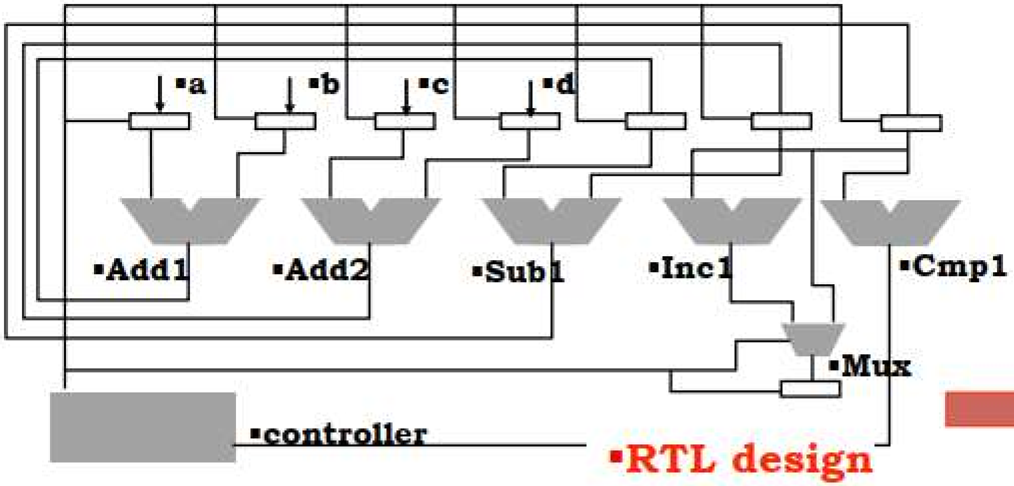
## Architecture

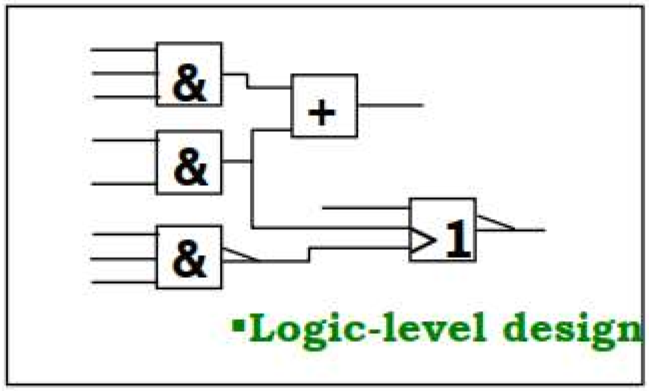
- High-level partitioning of problem into functional blocks
  - Can be expressed in a variety of forms - text, graphically, formal languages
  - Difficult procedure without experience or tools to assist you
- Must choose between different possible architectures and weigh the costs and benefits of each choice
  - Architecture design is often a matter of balancing tradeoffs

# Implementation

- Transformation of architecture into hardware
  - Register Transfer Level (RTL)
  - Schematic Entry (logic)
  - Transistor Level

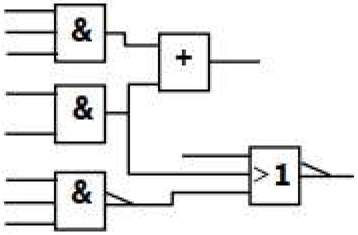




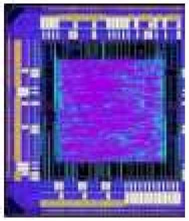


# Synthesis Tools

- Logic Synthesis tools generate a netlist of equations from design descriptions in VHDL or Verilog
  - Logic Synthesis Tools: Synopsys Design Compiler, Cadence BuildGates
  - The final phase is technology mapping
    - Optimization: optimize area/delay.
    - Technology-specific: ASIC, FPGA, CPLD



- Physical Synthesis tools: Synopsys Physical Compiler, Cadence PKS



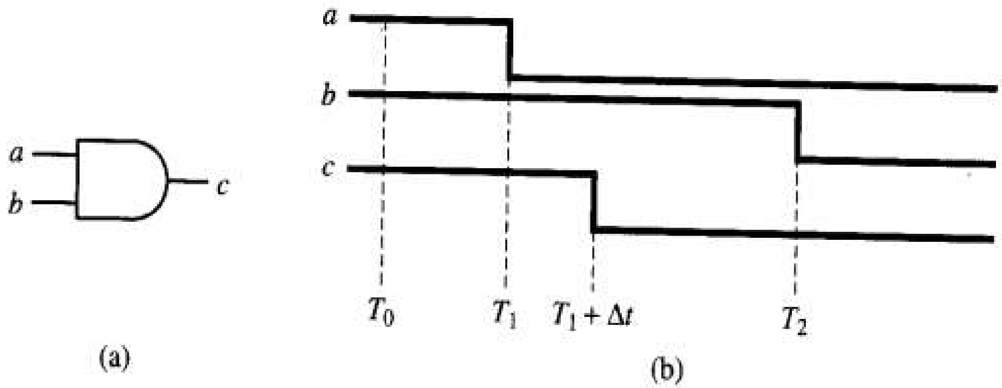
- FPGASynthesis tools: Altera Quartus II Synthesis, Xilinx ISE logic design tools, Mentor Graphics LeonardoSpectrum, Synopsys FPGA Compiler II.

# Verification

- Check whether implementation matches specification
  - Simulation: ModelSim (Mentor Graphics), VCS simulator (Synopsys), NC-Sim (Cadence).
  - Formal equivalency checking: Formality (Synopsys), Conformal (Cadence)
- Iterative process
  - Simulate
  - Refine specification or architecture if necessary
  - Repeat

# How Logic Simulation Works

- VHDL/Verilog Simulator: Event-driven simulator
  - When a circuit node changes in value, the time, the node and the new value are collectively known as an **event**.
  - When a specified time is reached, the logic value of the node is changed.
  - Changes are detected and executed in parallel using concurrent VHDL statements.
- Event-driven simulators are more suitable for digital systems because of parallelisms in digital systems operations.

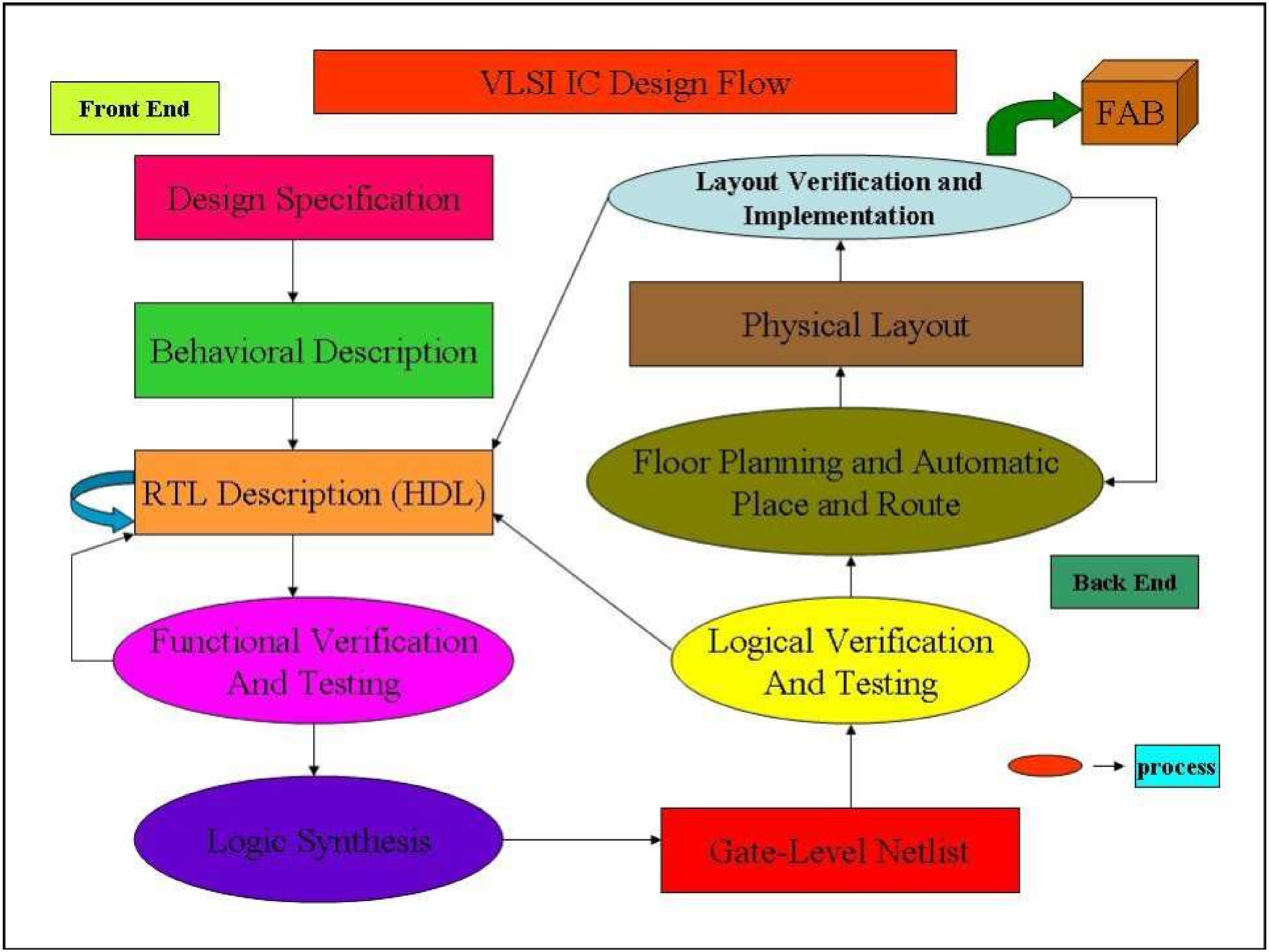


(a)

(b)

# VHDL Simulation

- VHDL simulation process can be broken into
  - Elaboration: Before simulation begins, the design hierarchy is first elaborated. This means all the pieces of the model code (entities, architecture and configurations) are put together.
  - Initialization: The nets in the model are initialized just before simulation starts.
  - Simulation cycle:
    - Simulation cycle is then continuously repeated during which processes are executed and signals are updated.
- Advantage: Top-Down design methodology, Technology independent


VLSI IC Design Flow
Front End
FAB
Design Specification
Layout Verification and Implementation
Behavioral Description
Physical Layout
RTL Description (HDL)
Floor Planning and Automatic Place and Route
Back End
Functional Verification And Testing
Logical Verification And Testing
process
Logic Synthesis
Gate-Level Netlist

- What we are going to learn?
  - Design a general purpose logic circuit
  - Design a application specific logic circuit
- Systematic ways of designing digital systems
  - More advanced topics in logic designs
  - Modeling logic problems with HDLs
  - Simulating logic functions
  - Synthesizing logic circuits with HDLs
  - Validating designs with programmable logic devices on a hardware development system
  - The concept of a digital system