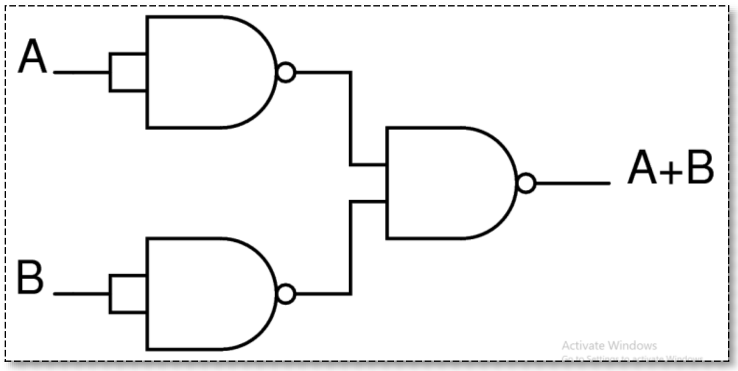
QUARTUS DESIGN FLOW

28th July 2022, Digital Systems Lab

Think it. Make it. Prove it.

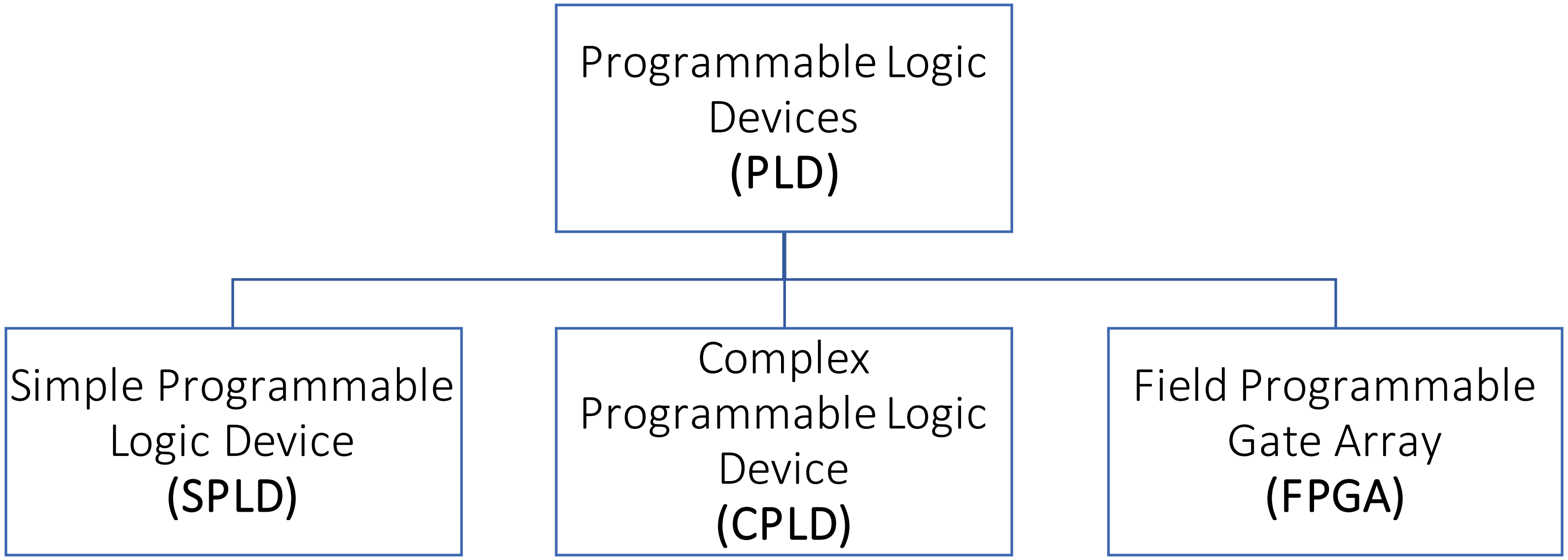
DIGITAL SYSTEMS LAB



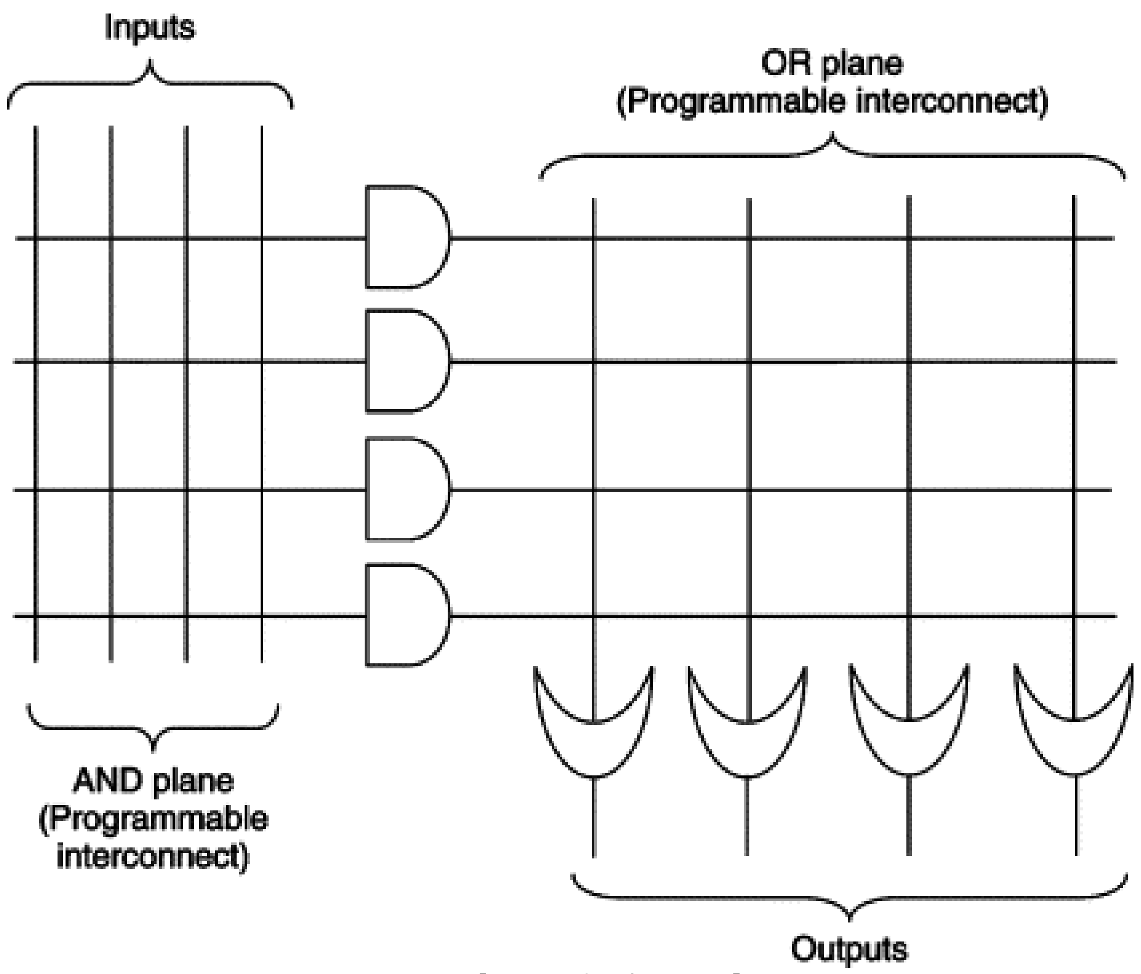
Digital Circuit

Hardware Implementation on FPGA

PROGRAMMABLE LOGIC DEVICE

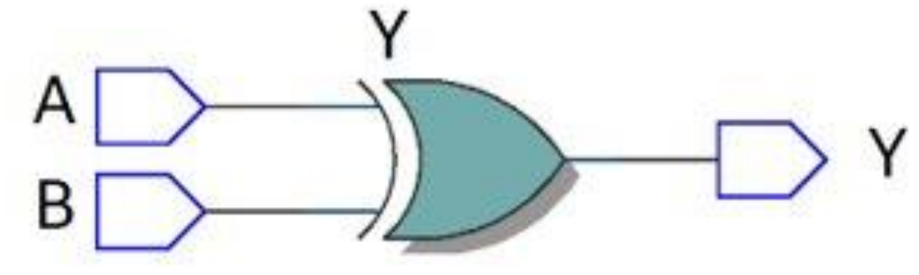

Programmable Logic Devices (PLD)
Simple Programmable Logic Device (SPLD)
Complex Programmable Logic Device (CPLD)
Field Programmable Gate Array (FPGA)

PROGRAMMABLE LOGIC ARRAY

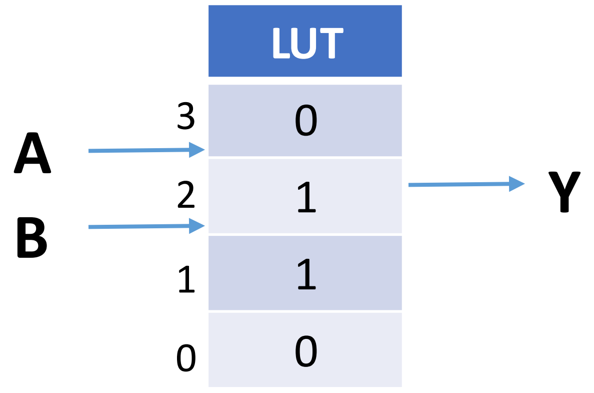


[Autodesk.com]

HALF ADDER SUM USING LUT

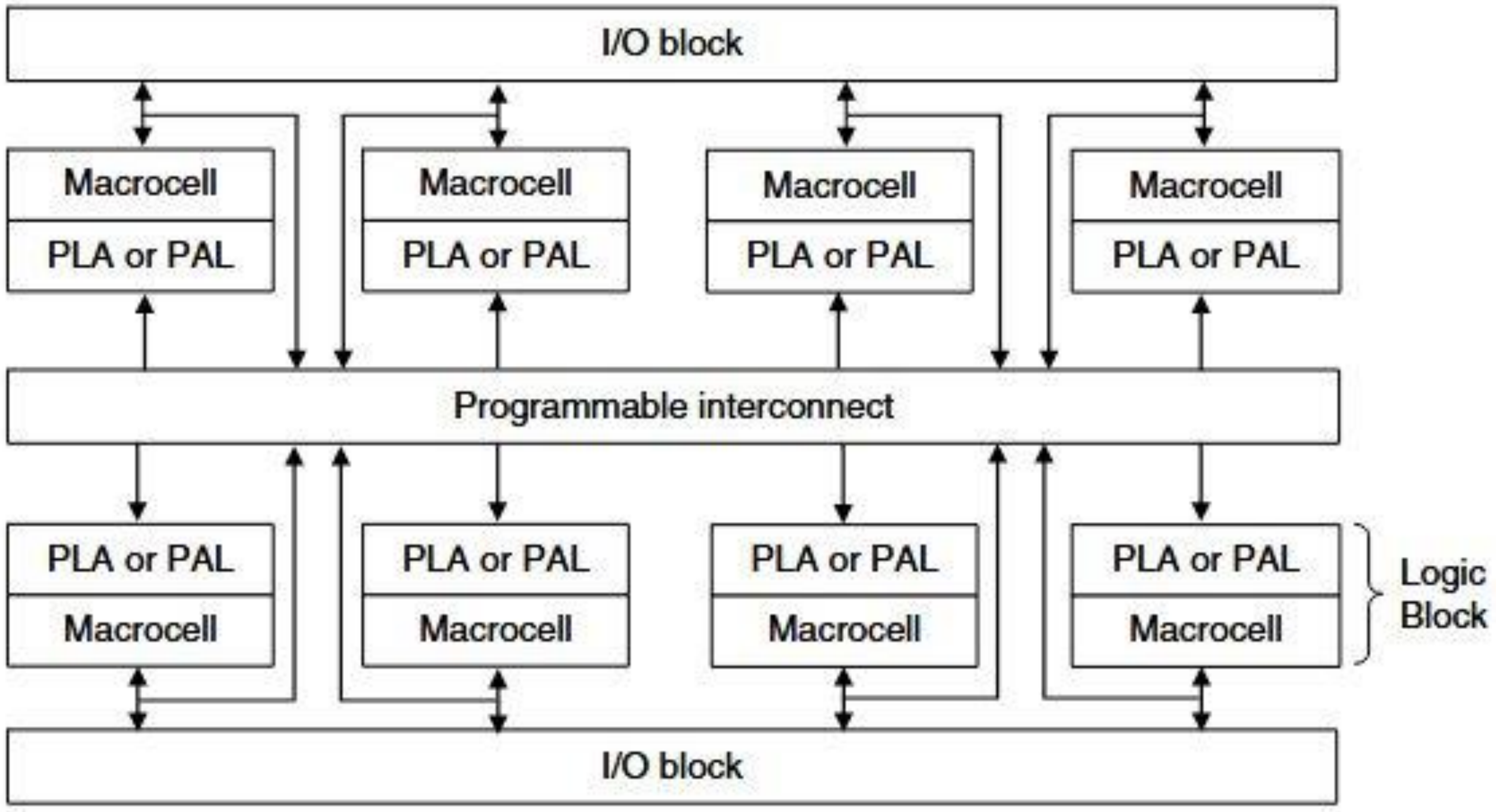


XOR circuit



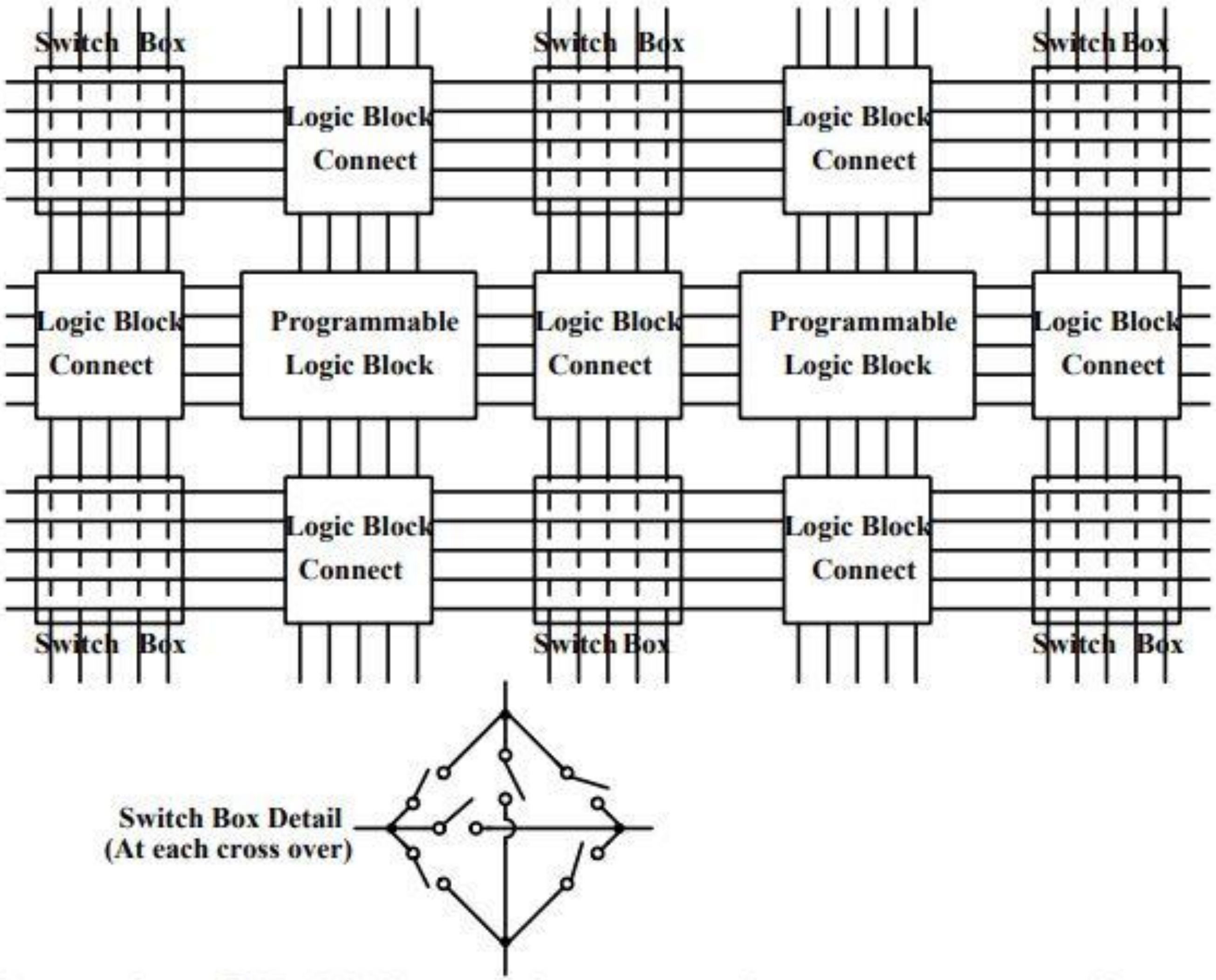
Look Up Table (LUT)

COMPLEX PROGRAMMABLE LOGIC DEVICES (CPLD)

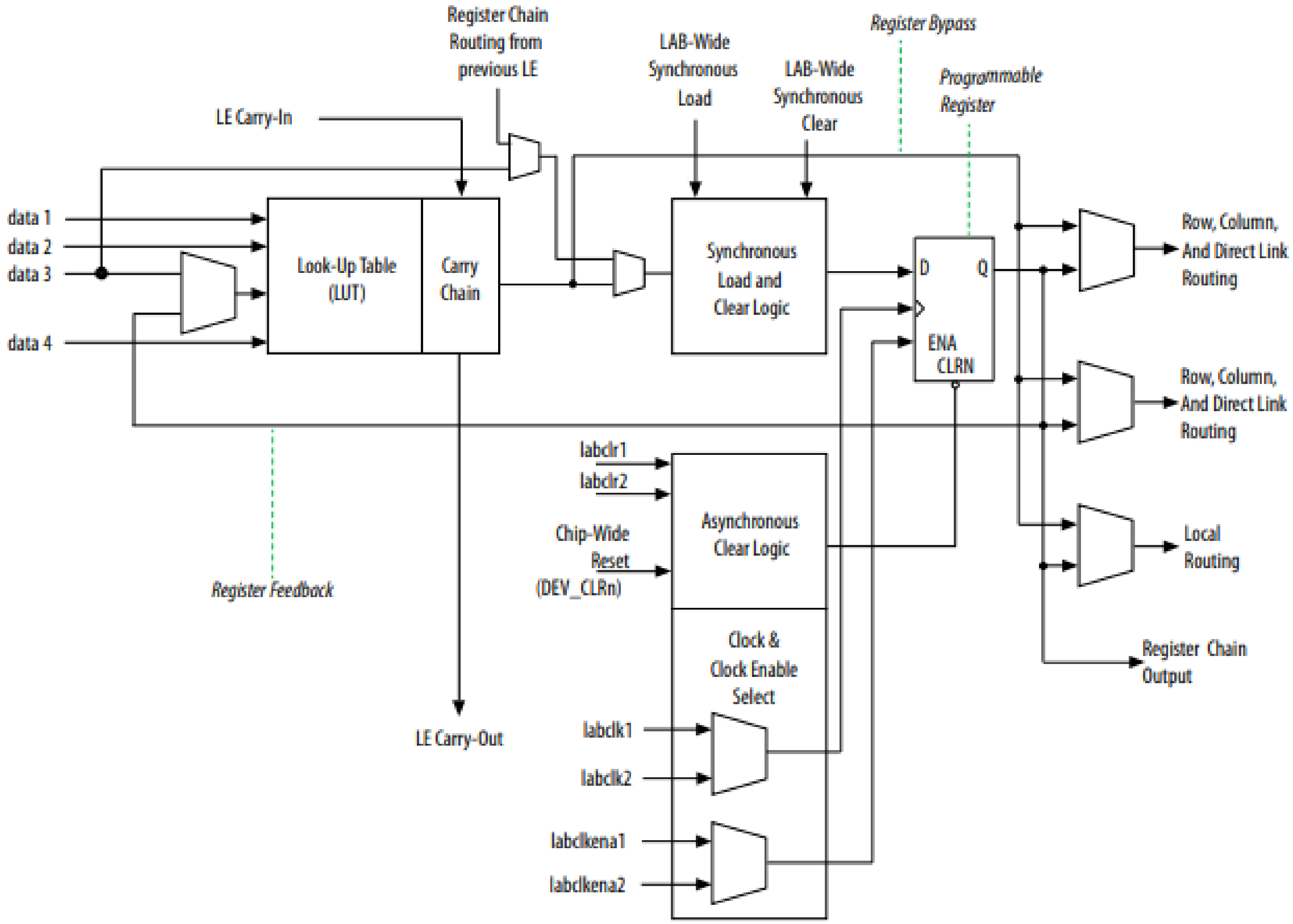


[Autodesk.com]

FIELD PROGRAMMABLE GATE ARRAY

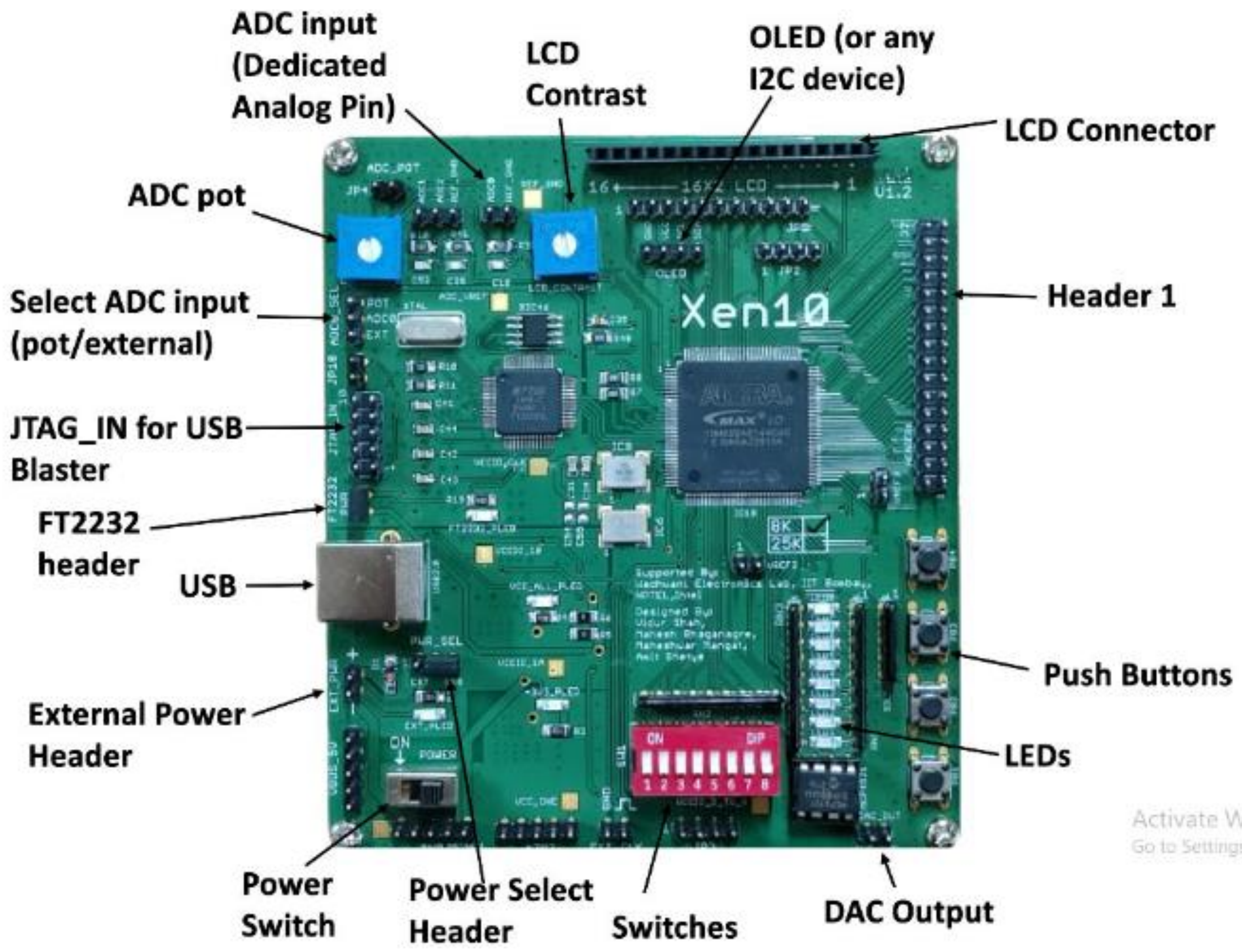


LOGIC ELEMENT OF ALTERA MAX-10 FPGA

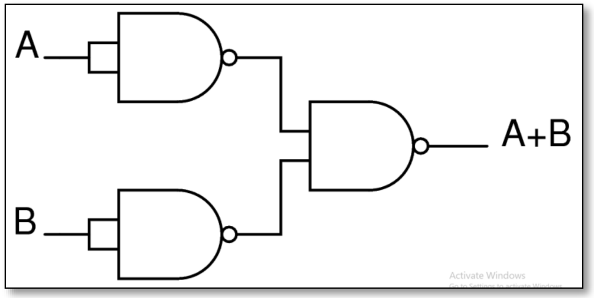

Register Chain Routing from previous LE
LAB-Wide Synchronous Load
LAB-Wide Synchronous Clear
Register Bypass
Programmable Register
LE Carry-In
data 1
data 2
data 3
data 4
Look-Up Table (LUT)
Carry Chain
Synchronous Load and Clear Logic
D
Q
ENA
CLRN
Row, Column, And Direct Link Routing
Row, Column, And Direct Link Routing
Local Routing
Register Chain Output
labclr1
labclr2
Chip-Wide Reset (DEV_CLRn)
Asynchronous Clear Logic
Register Feedback
Clock & Clock Enable Select
LE Carry-Out
labclk1
labclk2
labclkena1
labclkena2

High-Level Block Diagram of Logic Element of MAX-10 [www.intel.com]

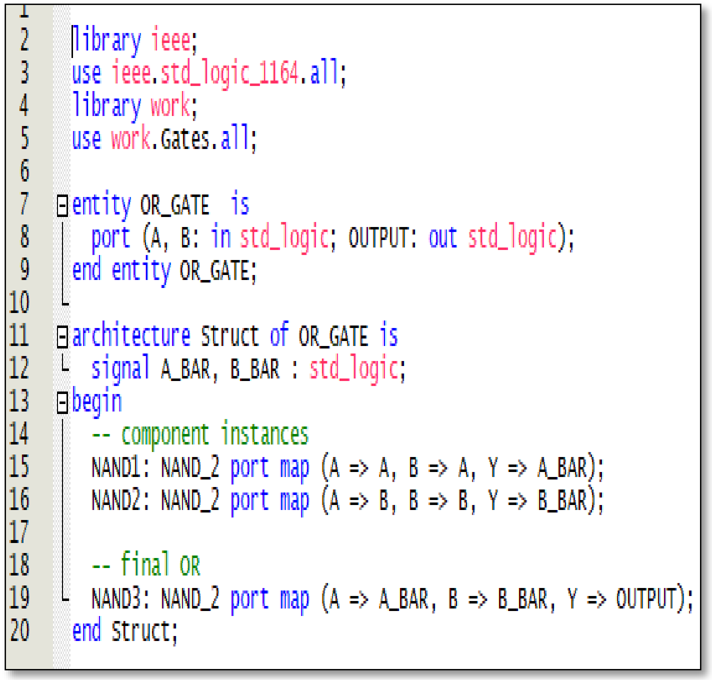
MAX 10 – 10M25SAE144C8G



HARDWARE DESCRIPTION LANGUAGE



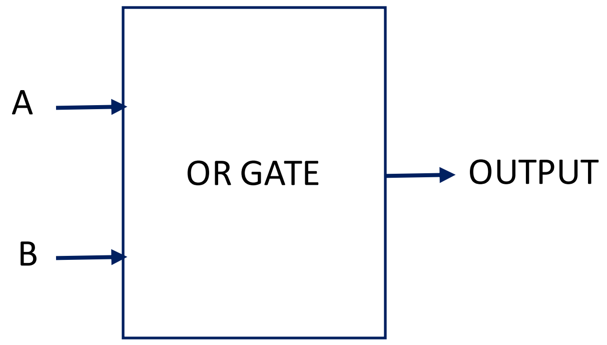
Digital Circuit



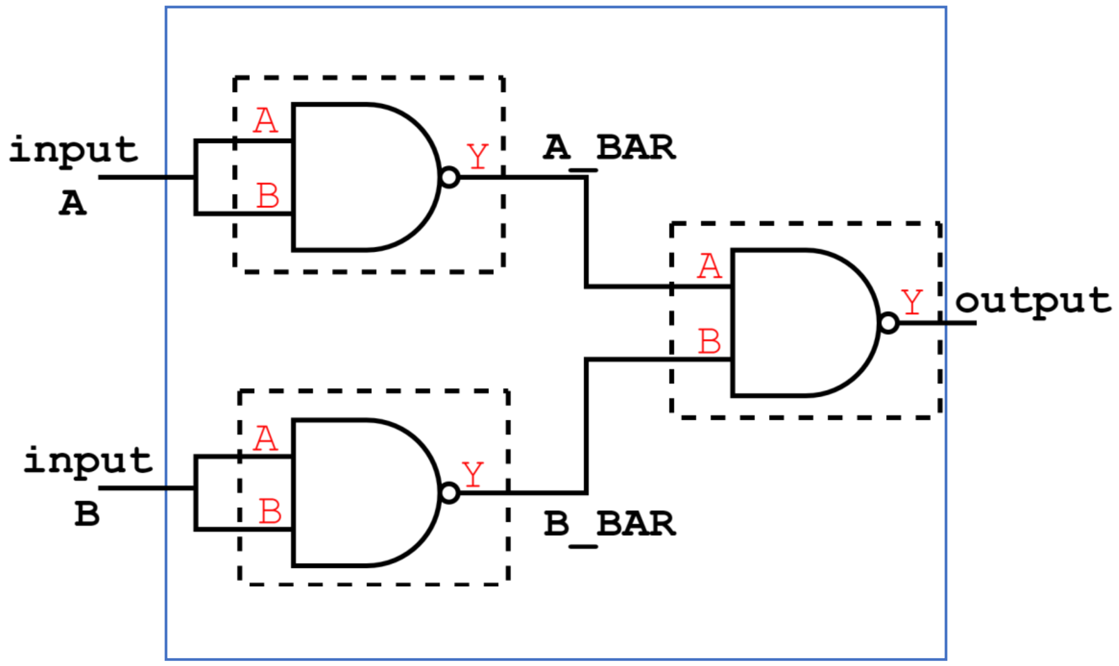
```
library ieee;
use ieee.std_logic_1164.all;
library work;
use work.Gates.all;

entity OR_GATE  is
  port (A, B: in std_logic; OUTPUT: out std_logic);
end entity OR_GATE;

architecture Struct of OR_GATE is
  signal A_BAR, B_BAR : std_logic;
begin
  -- component instances
  NAND1: NAND_2 port map (A => A, B => A, Y => A_BAR);
  NAND2: NAND_2 port map (A => B, B => B, Y => B_BAR);

  -- final OR
  NAND3: NAND_2 port map (A => A_BAR, B => B_BAR, Y => OUTPUT);
end Struct;
```

Equivalent VHDL code

ENTITY AND ARCHITECTURE

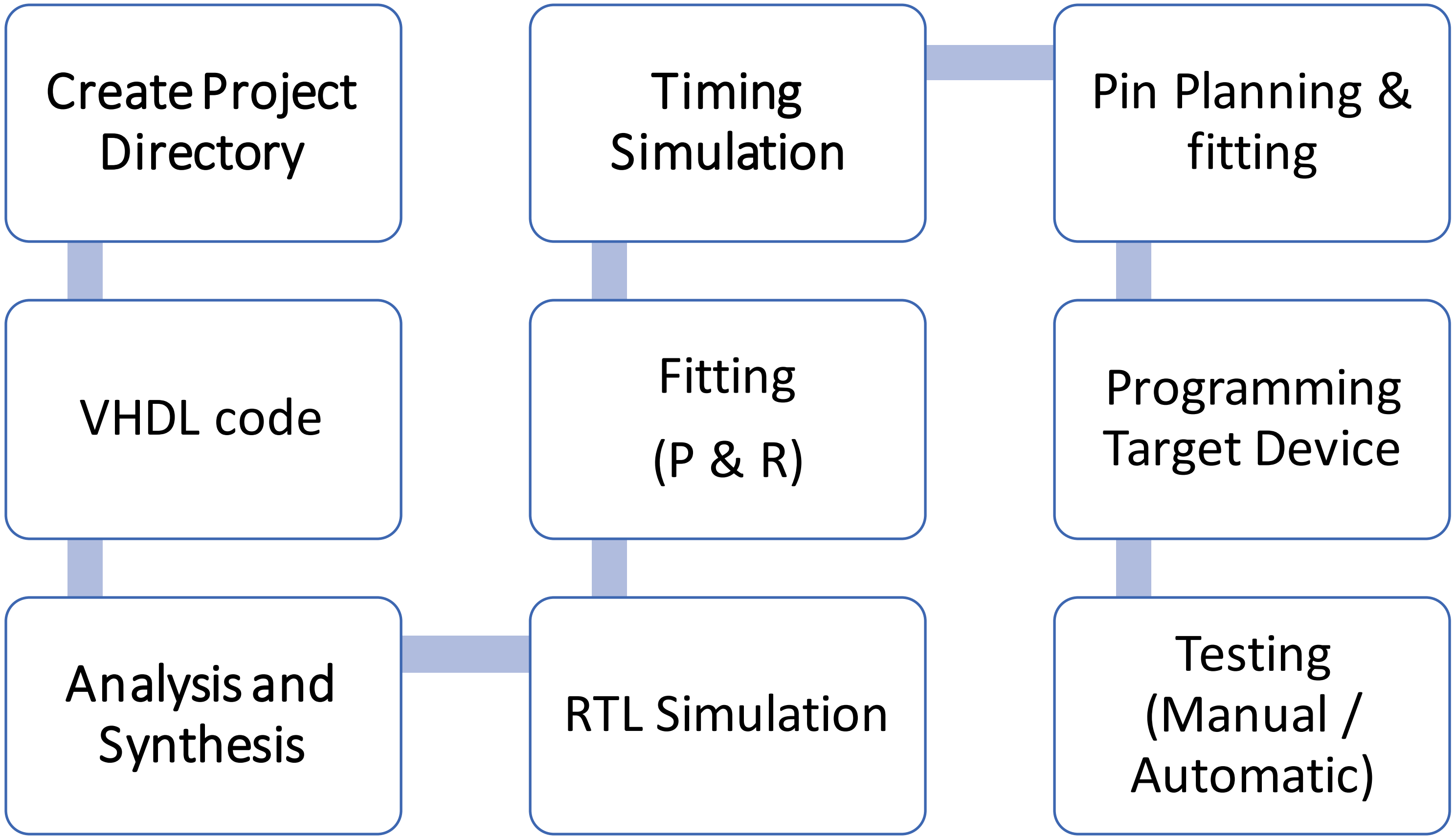


Entity



Architecture

QUARTUS DESIGN FLOW



FLOWCHART FOR GETTING YOUR VHDL CODE READY

Library

- Include all required libraries for your **design**

Entity

- Describe how your **design** looks from outside
- Declare inputs and outputs **in** port list

Architecture

- Declare intermediate **signals** and **components in** architecture **declaration zone**
- Describe your architecture **in** architecture **zone**

DUT

- Wrap your **design in** DUT
- Change number of inputs and outputs **in** DUT **entity**
- Declare your **design** entity as **component and** do port mapping by **instantiating** it

Testbench

- Modify the number of inputs **and** outputs **in testbench**

After all the above steps are performed, you can simulate your design.

SUMMARY OF QUARUS DESIGN FLOW TILL RTL SIMULATION

- <u>Creating Project Directory:</u>
  - From New Project Wizard create new project directory
  - Copy paste DUT.vhdl, Testbench.vhdl, Gates.vhdl and TRACEFILE.txt in newly created project directory
  - Add all this files in Quartus from settings -> Assignments -> Files section
- <u>Writing VHDL code:</u>
  - Write your VHDL code to describe your digital design, This code is sectioned in three parts:
    - <u>Libraries</u>: Include all the required libraries in code. This libraries have the declaration and definition of data types, general gate primitives and commonly used VHDL keywords
    - <u>Entity</u>: Here you will describe how your digital circuit looks from outside. Inputs and outputs signal will be mentioned in port list
    - <u>Architecture</u>: In this you have two zones, one is architecture declaration zone where all the intermediate signals and components will be declared; the other zone is Architecture zone where you will describe how exactly your entity/circuit looks from inside. Structural, Behavioral and Design flow are three coding styles to describe Architecture.

- Wrapping your design inside DUT:
  - This step is required for you to test your design using generic testbench.
  - In already provided template of DUT code, make the following changes:
    - Modify the number of inputs and outputs of DUT entity from port list.
    - In architecture declaration zone declare your design entity as component (replace the already written default component)
    - In architecture zone instantiate your component and do port mapping correctly.
- Adding generic testbench:
  - Modify the values of number of input and output variables value according to your design.
  - Add the testbench and TRACEFILE.txt in the project from Settings -> Assignments -> Simulation -> compile testbench.
- Analysis and Synthesis:
  - Run the analysis and synthesis task, this task will check for the syntax and semantics of your code and will synthesis the hardware.
  - In RTL viewer you can check the synthesized hardware.

- RTL simulation:
    - Run the RTL simulation from Tools -> Run Simulation Tools -> RTL simulation.
    - Modelsim window will be displayed with your simulation results.

THANK YOU