CENG3430 Rapid Prototyping of Digital Systems

Lecture 02:
Introduction to ZedBoard

Ming-Chang YANG

mcyang@cse.cuhk.edu.hk

Outline

- Digital System Design Basics
  - Integrated Circuit Technology
  - Design Flow of Digital Systems
  - Spectrum of Design Technologies

- Zynq: All-Programmable SoC (APSoC)
  - Our Board: ZedBoard
    - ZedBoard Layout and Interfaces
    - Specifying ZedBoard in Vivado
    - Hardware Setup for ZedBoard
    - Programming the ZedBoard
    - Xilinx Design Constraints (XDC) File
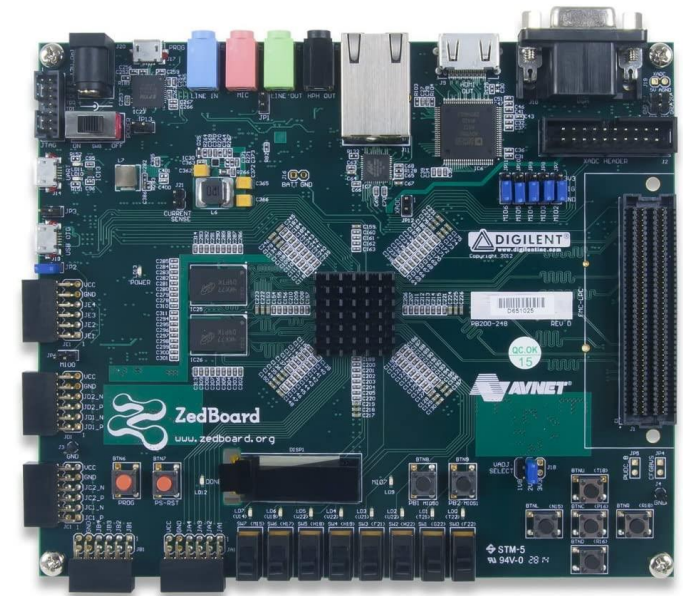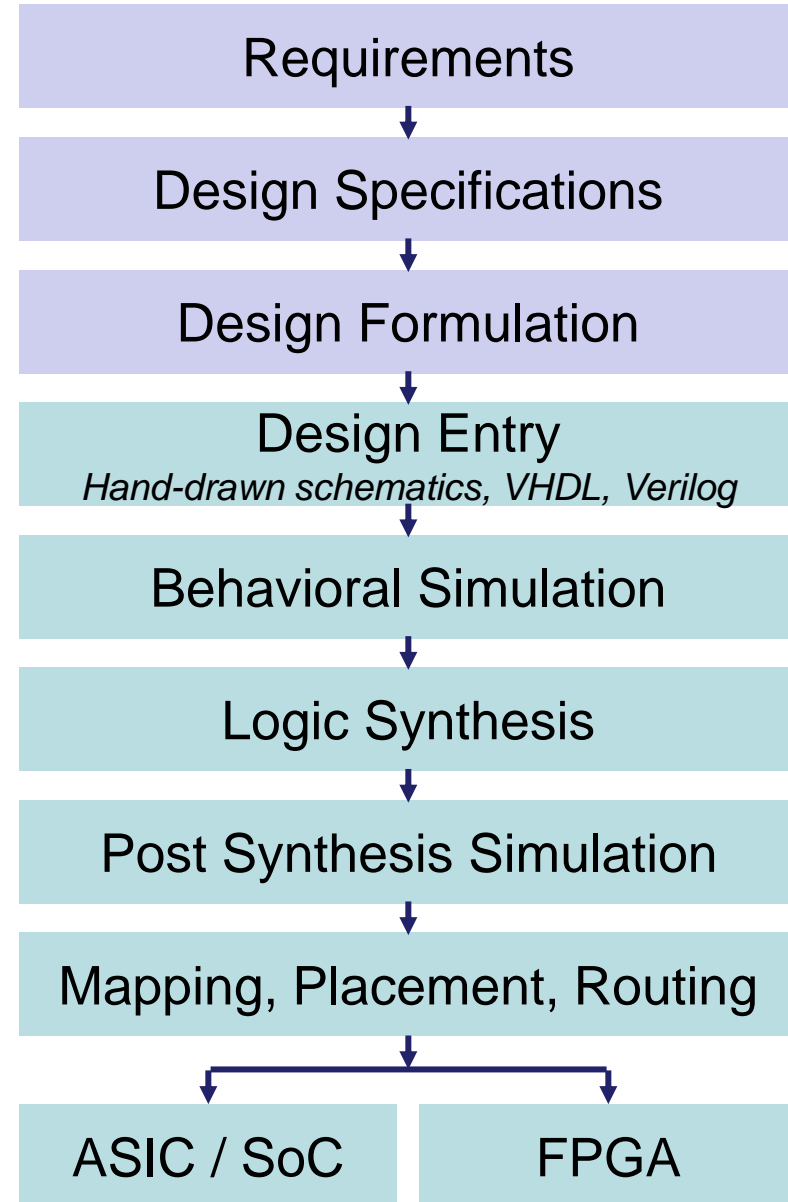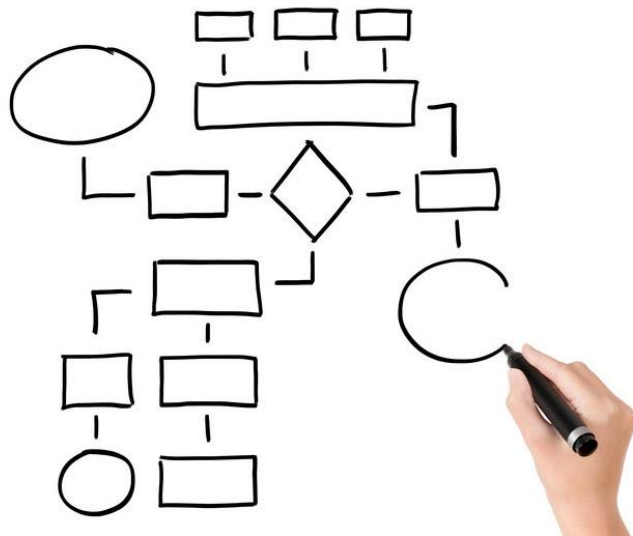
Integrated Circuit Technology

- Integrated circuit (IC) technology has improved to allow more and more components on a chip.
  - Small Scale Integration (SSI): 1 to 20 gates
  - Medium Scale Integration (MSI): 20 to 200 gates
  - Large Scale Integration (LSI): 200 to few thousands gates
  - Very Large Scale Integration (VLSI): More than 10,000 gates
  - Ultra Large Scale Integration (ULSI): 100 million transistors

 # Gates

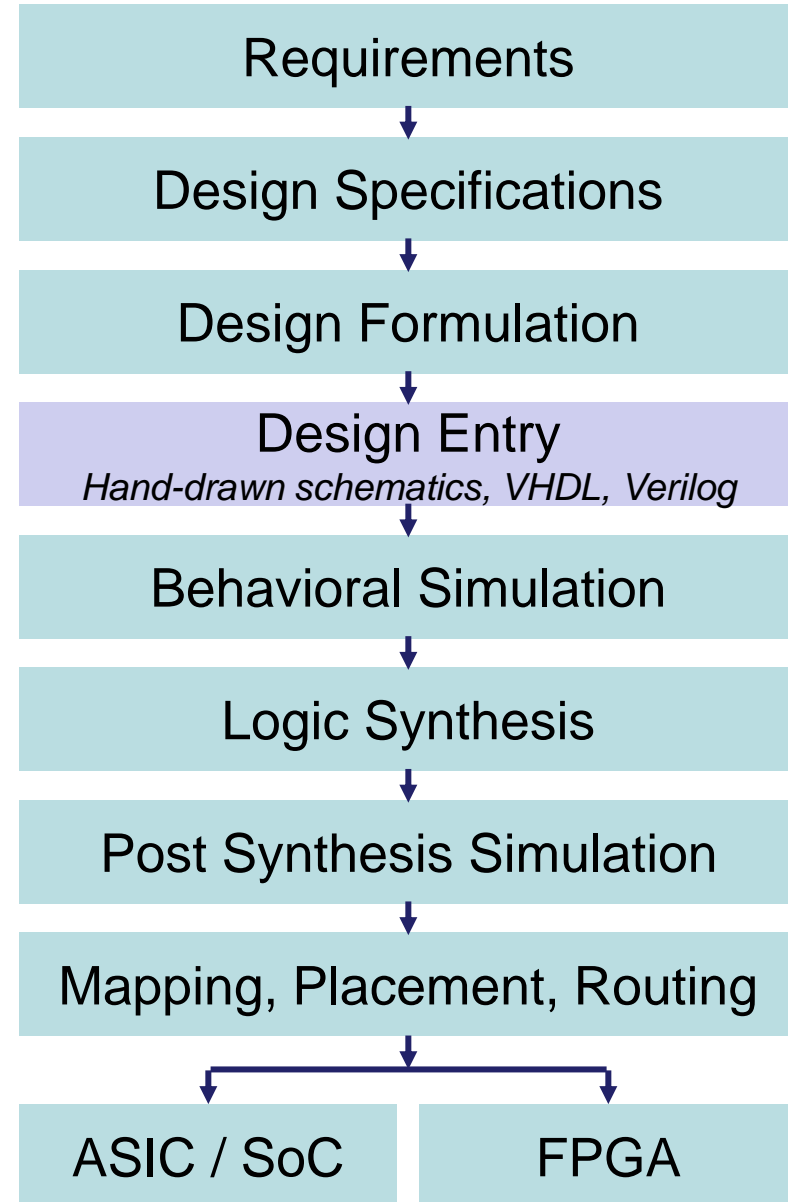- Digital system design have become more complex.

Design Flow of Digital Systems (1/6)

- All the designs start with design requirements and design specifications.

- The next step is to formulate the design conceptually.
  - Either at a block diagram level or at an algorithmic level.

Requirements

Design Specifications

Design Formulation

Design Entry
Hand-drawn schematics, VHDL, Verilog

Behavioral Simulation

Logic Synthesis

Post Synthesis Simulation

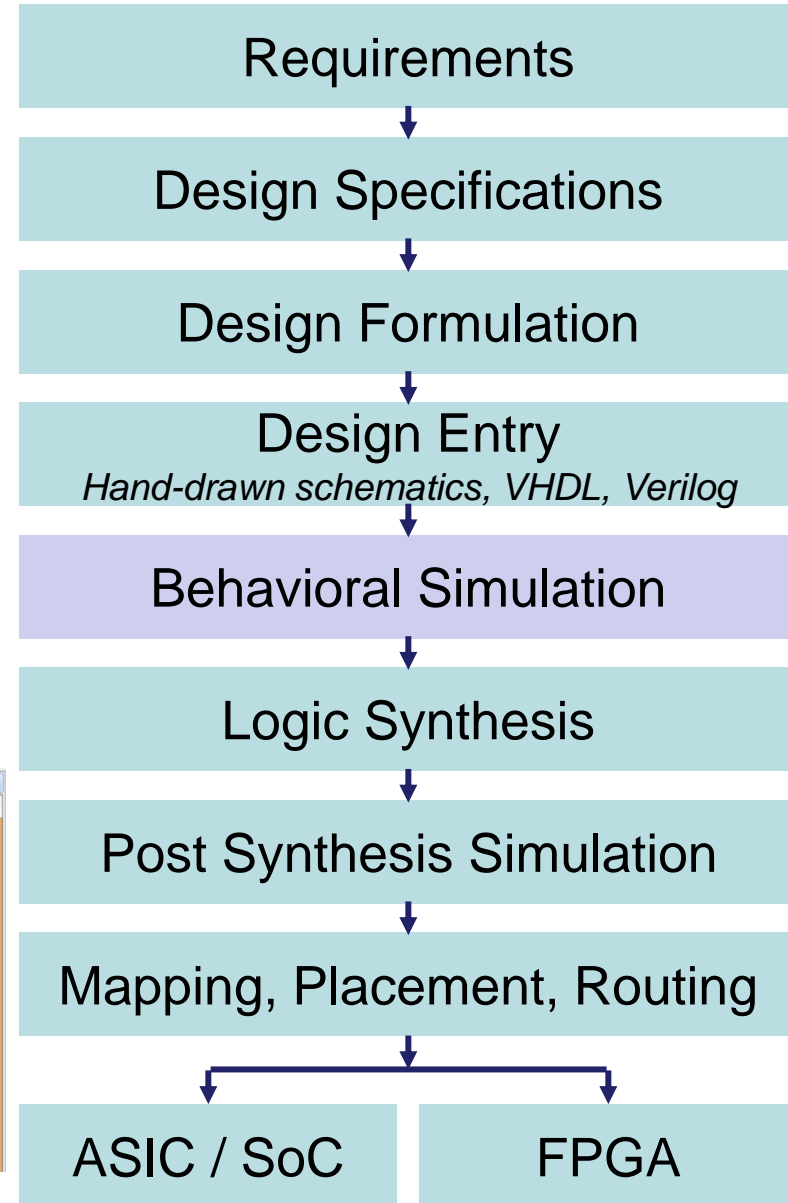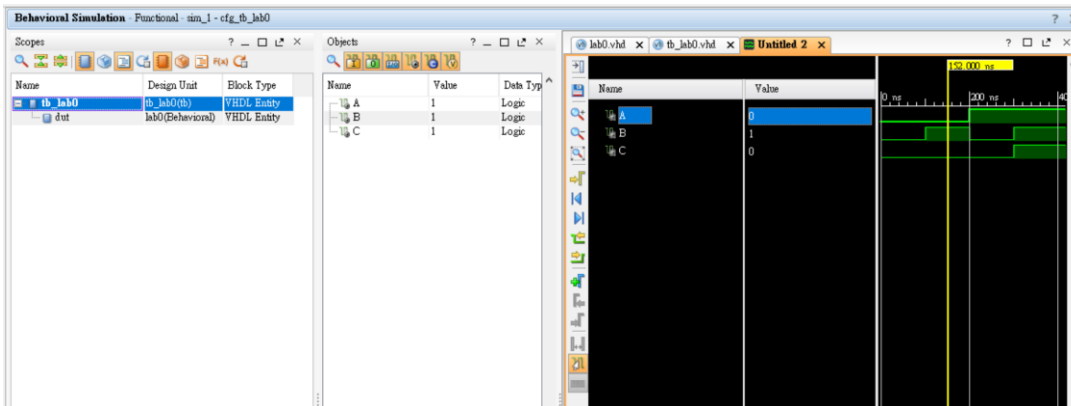Mapping, Placement, Routing

ASIC / SoC   FPGA

- ## Design Entry:

  - *Olden days*: Hand-drawn schematic or blueprint

  - *Now*: Computer-aided design (CAD) tools

    - **Schematic Capture**: Design with gates, flip-flops, and standard building blocks.
      - E.g., ORCAD
    - **Hardware Descriptions Languages (HDLs)**: Design and debug at higher level
      - E.g., VHDL and Verilog

Requirements

↓

Design Specifications

↓

Design Formulation

↓

Design Entry
Hand-drawn schematics, VHDL, Verilog

↓

Behavioral Simulation

↓

Logic Synthesis

↓

Post Synthesis Simulation

↓

Mapping, Placement, Routing

↓

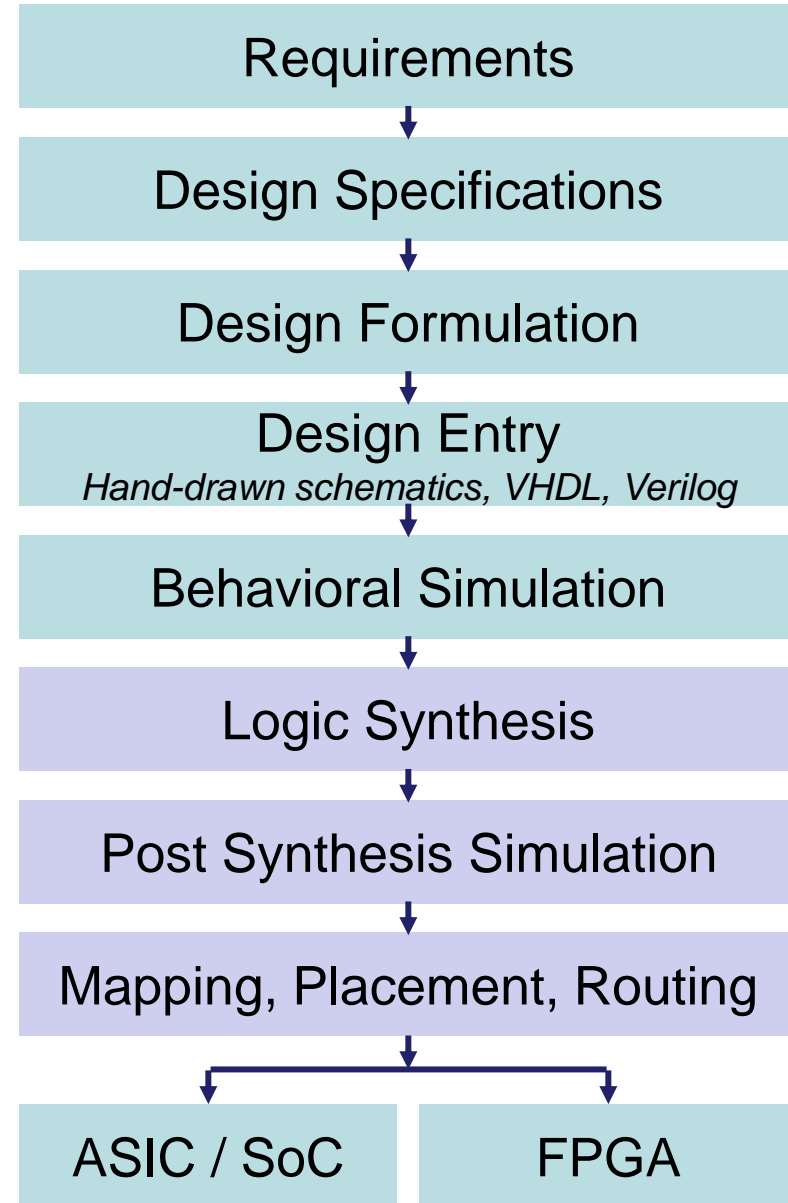ASIC / SoC   FPGA

Design Flow of Digital Systems (3/6)

- The entered design then should be simulated.
  - To ensure it function correctly at high-level behavioral model
  - To unveil problems in the design

- Recall: In Lab01, we run
  - Click "Flow" → "Run Simulation" → "Run **Behavioral Simulation**"

Requirements

Design Specifications

Design Formulation

Design Entry
Hand-drawn schematics, VHDL, Verilog

Behavioral Simulation

Logic Synthesis

Post Synthesis Simulation

Mapping, Placement, Routing

ASIC / SoC   FPGA

- ## Logic Synthesis
  - Convert <u>the high-level abstract descriptions</u> to <u>components at the gate and flip-flop levels</u>
    - Netlist: Outputs of synthesis tool
      - Specify internal connections

- ## Post-Synthesis Simulation
  - Test specific implementations of the hardware components

- ## Mapping, Placement, Routing
  - **Mapped** into the specific **target**
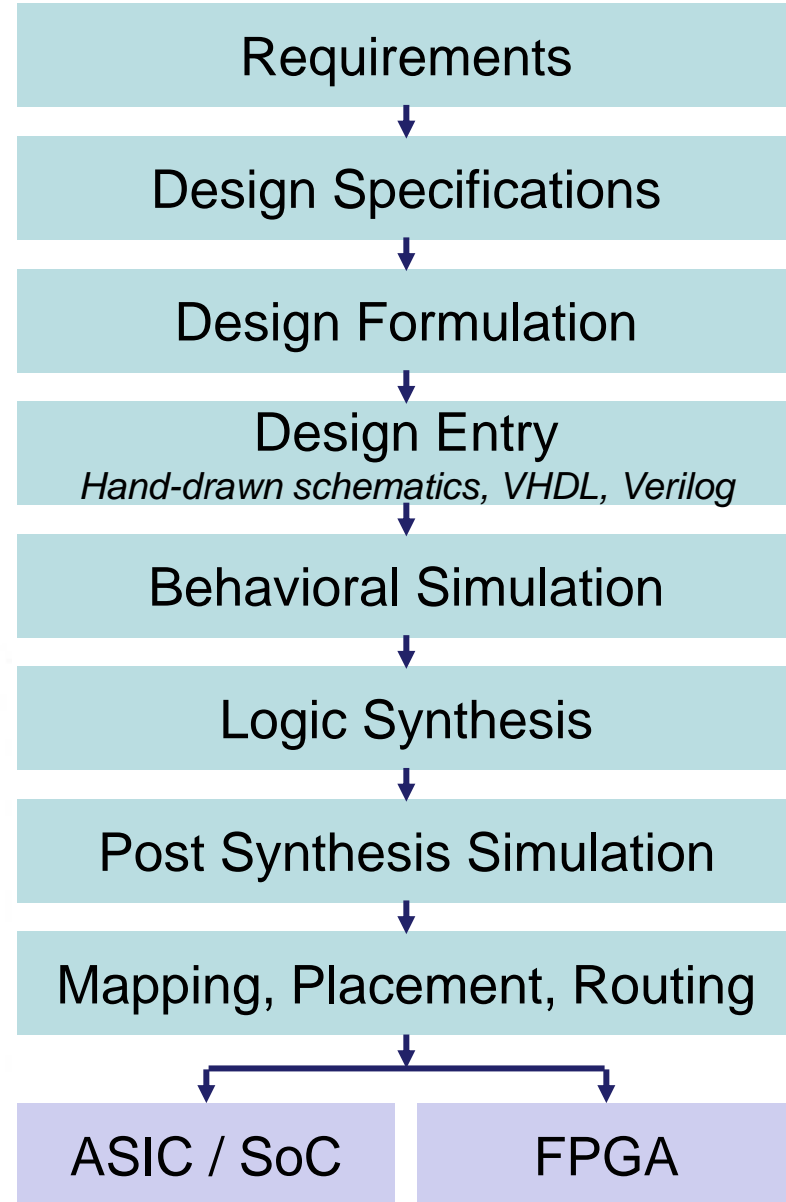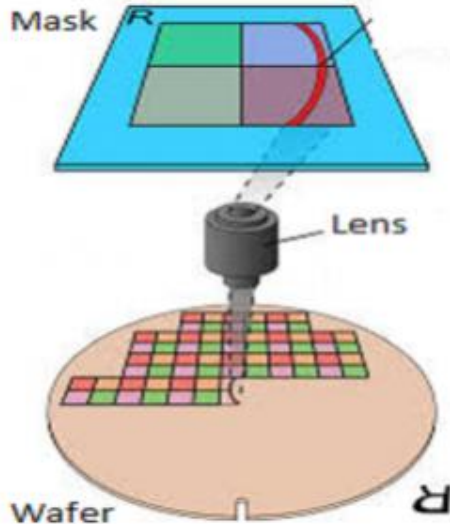 - **Placed** into specific parts
  - **Route** the paths of connections

Requirements
Design Specifications
Design Formulation
Design Entry *Hand-drawn schematics, VHDL, Verilog*
Behavioral Simulation
Logic Synthesis
Post Synthesis Simulation
Mapping, Placement, Routing

ASIC / SoC	FPGA

- Two most common **targets**:

  - Field Programmable Gate Array (FPGA)

    - **Programming** simply involves writing a sequence of 0's and 1's into the <u>programmable cells</u> of FPGA.

  - AISC or SoC

    - The routed design is used to generate a <u>photomask</u> for producing integrated circuits (ICs).
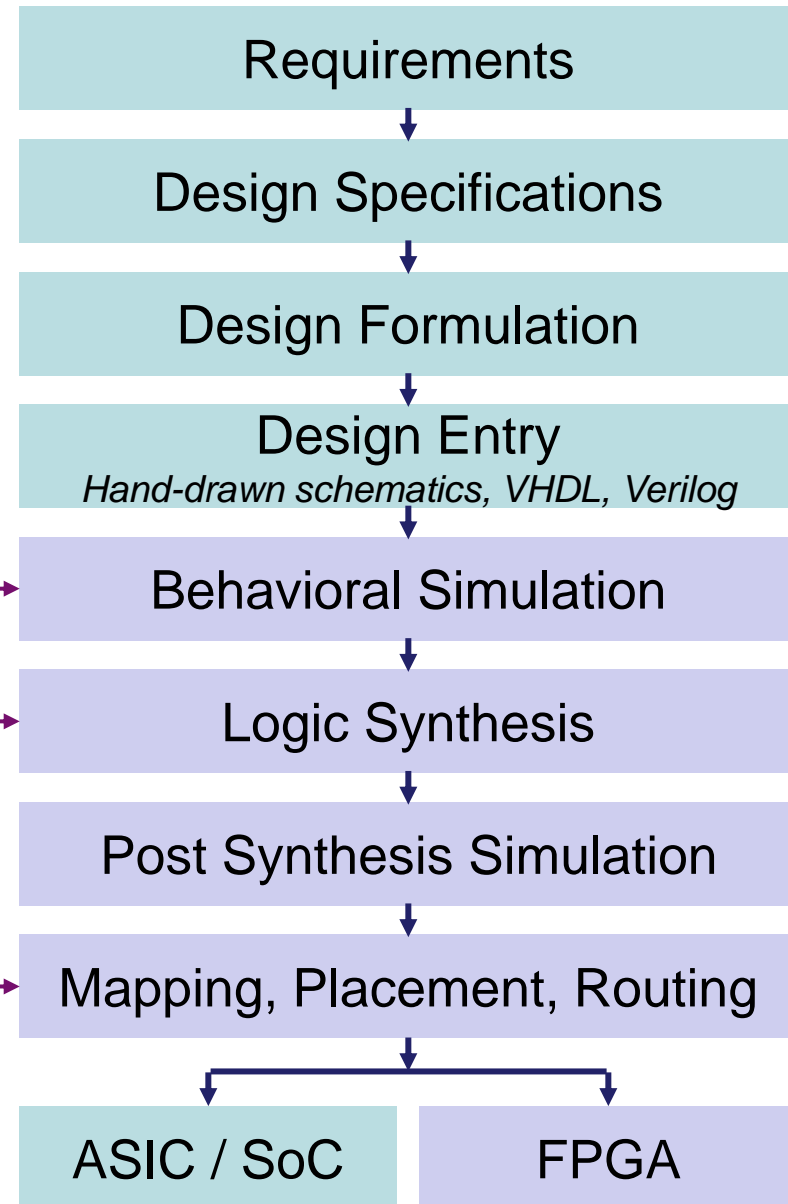

Requirements
Design Specifications
Design Formulation
Design Entry *Hand-drawn schematics, VHDL, Verilog*
Behavioral Simulation
Logic Synthesis
Post Synthesis Simulation
Mapping, Placement, Routing

ASIC / SoC	FPGA

Design Flow of Digital Systems (6/6)

Requirements

↓

Design Specifications

↓

Design Formulation

↓

Design Entry
Hand-drawn schematics, VHDL, Verilog

↓

Behavioral Simulation

↓

Logic Synthesis

↓

Post Synthesis Simulation

↓

Mapping, Placement, Routing

↓

ASIC / SoC   FPGA

Spectrum of Design Technologies

(Vertical axis) Cost, Design Time, and Speed

(Horizontal axis) Density and Degree of Customization

- Gates, flip-flops, and standard logic elements
- Programmable Logic Arrays, Devices
- Field programmable gate arrays (FPGAs)
- Custom Application-Specific ICs (ASICs)
- System on Chip (SoC)

Outline

- Digital System Design Basics
  - Integrated Circuit Technology
  - Design Flow of Digital Systems
  - Spectrum of Design Technologies

- Zynq: All-Programmable SoC (APSoC)
  - Our Board: ZedBoard
    - ZedBoard Layout and Interfaces
    - Specifying ZedBoard in Vivado
    - Hardware Setup for ZedBoard
    - Programming the ZedBoard
    - Xilinx Design Constraints (XDC) File

# Zynq: All-Programmable SoC (1/2)

- All-Programmable SoC (APSoC): Zynq provides an more ideal platform for implementing flexible SoCs.

- Zynq comprises two main parts:

  – Programmable Logic (PL): equivalent to that of an FPGA

  – Processing System (PS): formed around a dual-core ARM Cortex-A9 processor
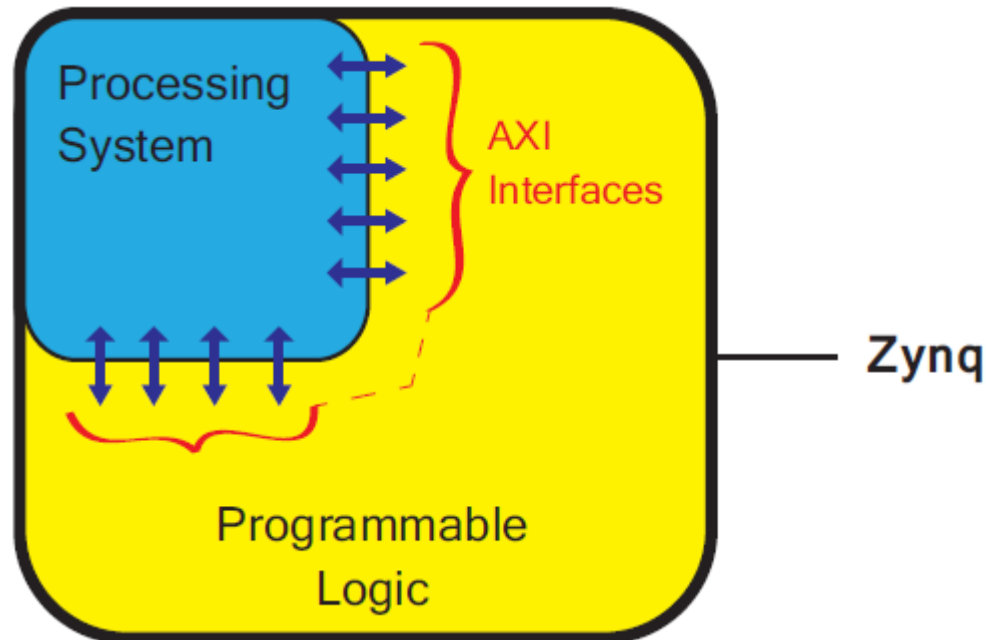
Zynq: All-Programmable SoC (2/2)

- **Programmable Logic (PL)**: Implements high-speed logic, arithmetic and data flow subsystems.

- **Processing System (PS)**: Supports software routines and/or operating systems.
  - The overall functionality of any designed system can be appropriately partitioned between hardware and software.

- **Advanced eXtensible Interface (AXI)**:
  - Links between the PL and PS are made using industry standard Advanced eXtensible Interface (AXI) connections.
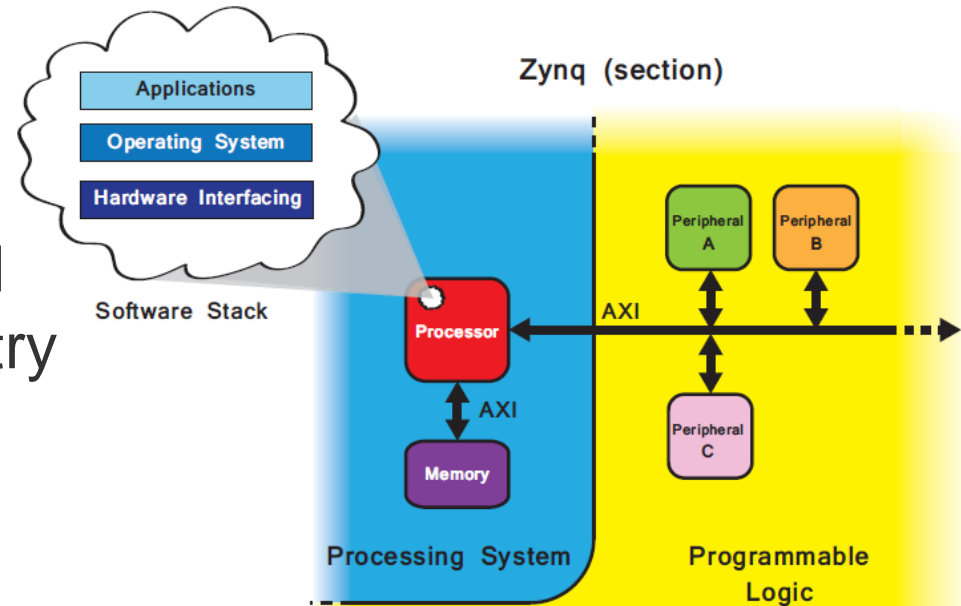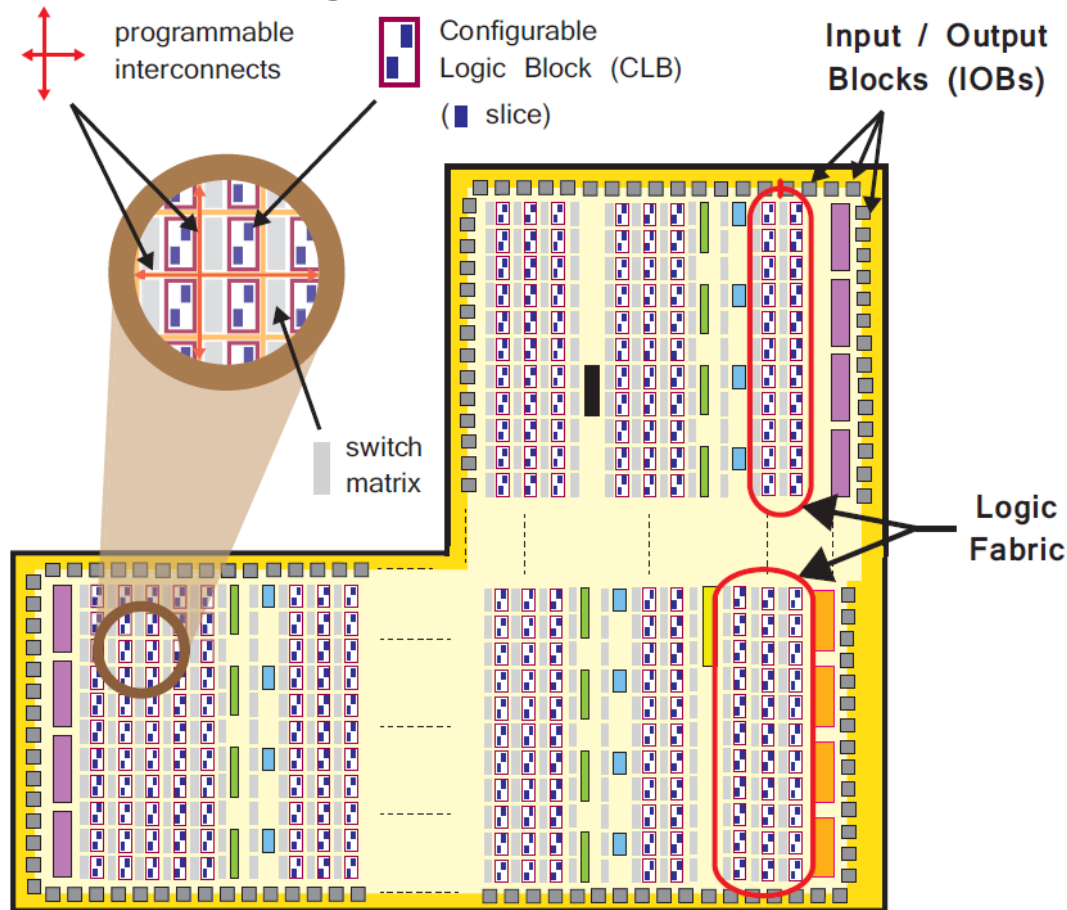
Processing System (PS)

- PS supports software routines and operating systems.
  - The overall functionality of any system can be partitioned.
- PS has a "hard" dual-core ARM Cortex-A9 processor.
  - Hard processors can achieve higher performance.
- By contrast, "soft" processor (e.g., Xilinx MicroBlaze) can be made by the programmable logic elements.
  - The number and precise implementation of soft processor instances is flexible.

ARM processor
(hard processor)

PS

PL

optional
MicroBlaze
processor
(soft processor, built from logic slices)
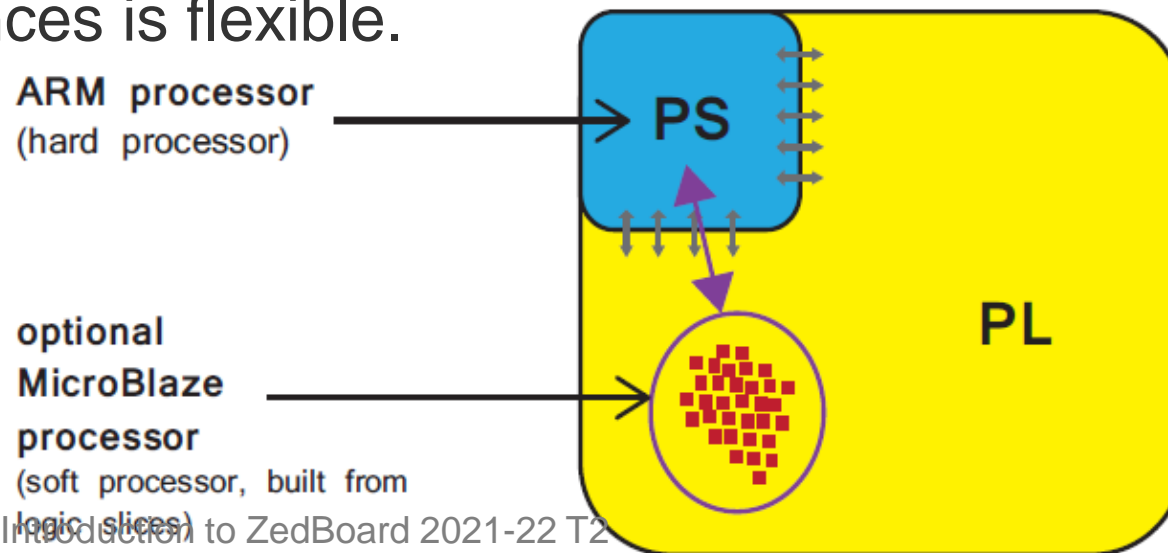
Programmable Logic (PL)

- PL section is ideal for implementing high-speed logic, arithmetic and data flow subsystems.
- PL is composed of general purpose FPGA logic fabric.

programmable interconnects

Configurable Logic Block (CLB)
(▮ slice)

Input / Output Blocks (IOBs)
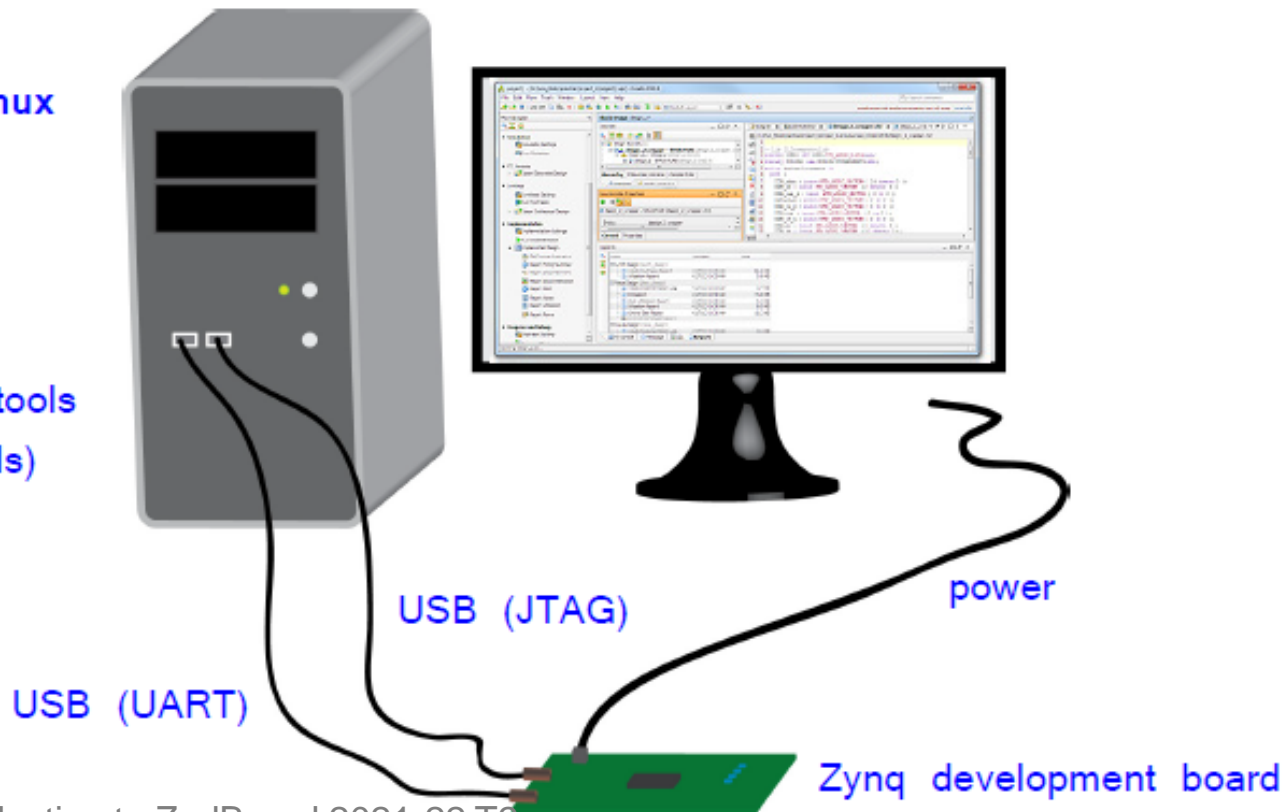
switch matrix

Logic Fabric

Zynq Development Setup

- Joint Test Action Group (JTAG): Downloading designs onto the development board over JTAG

- Universal Asynchronous Receiver/Transmitter (UART) and Terminal Applications: Interfacing and debugging
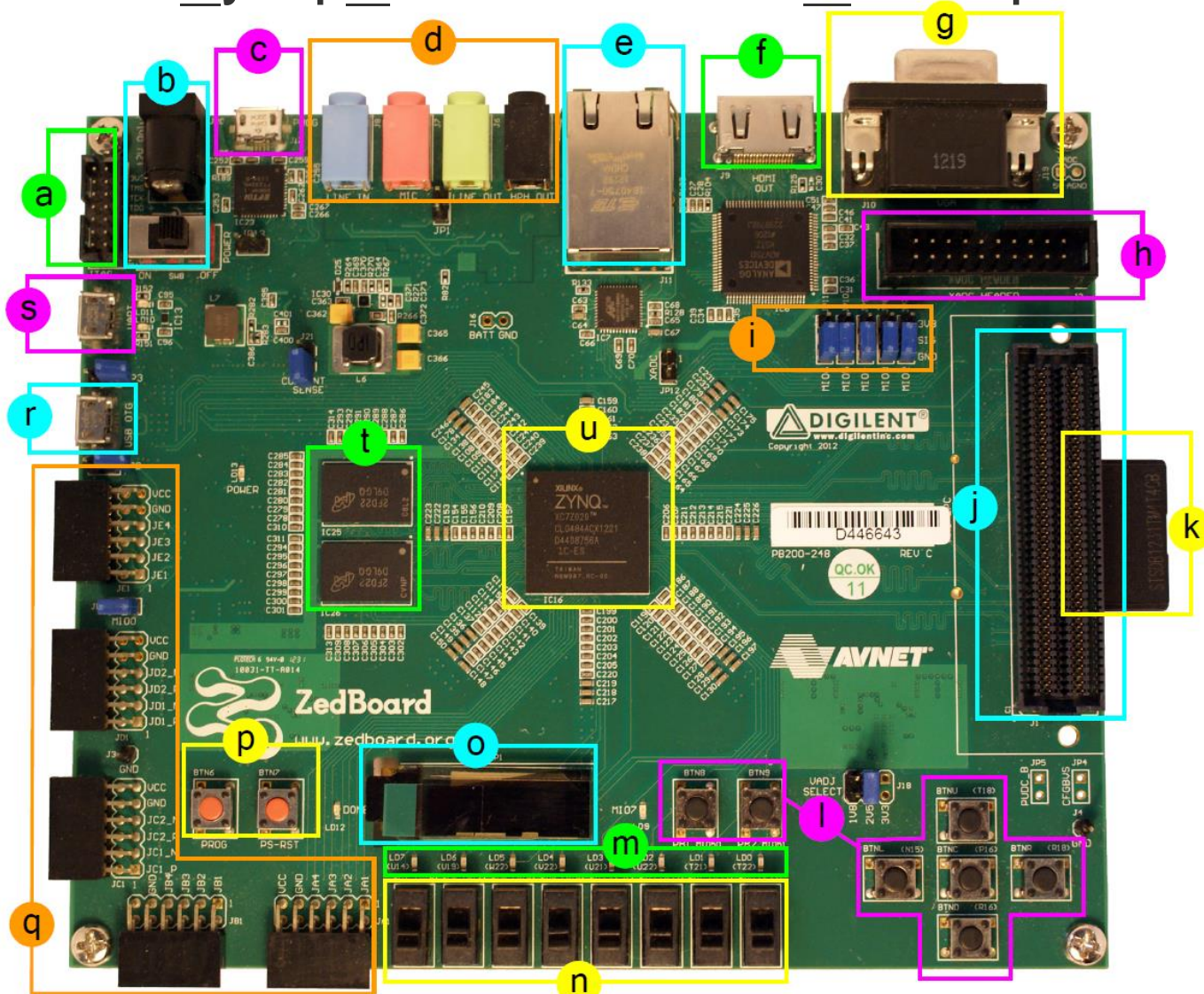
Windows / Linux computer

4GB+ RAM

Xilinx design tools (3rd party tools)

USB (JTAG)

power

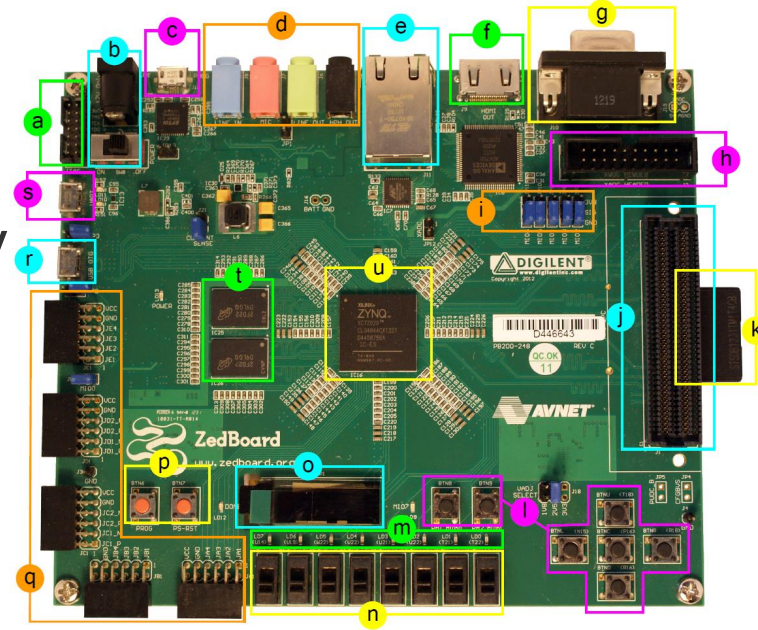USB (UART)

Zynq development board

- ZedBoard: **Z**ynq **E**valuation and **D**evelopment **Board**

ZedBoard Layout and Interfaces

- ZedBoard features a ZC7Z020 Zynq device.
  - Artix-7 logic fabric, with a capacity of 13,300 logic slices, 220 DSP48E1s, and 140 BlockRAMs
  - DDR3 Memory, and Flash
  - Several peripheral interfaces

|			
a Xilinx JTAG connector	**h** XADC header port	**o** OLED display
b Power input and switch	**i** Configuration jumpers	**p** Prog & reset push buttons
c USB-JTAG (programming)	**j** FMC connector	**q** 5 x Pmod connector ports
d Audio ports	**k** SD card (underside)	**r** USB-OTG peripheral port
e Ethernet port	**l** User push buttons	**s** USB-UART port
f HDMI port (output)	**m** LEDs	**t** DDR3 memory
g VGA port	**n** Switches	**u** Zynq device (+ heatsink)

Specifying ZedBoard in Vivado

- **ZedBoard Zynq Evaluation and Development Kit**:
  - The design tools have knowledge of the specific facilities and peripheral connections of ZedBoard.

- Target part: **xc7z020clg484-1**, Rev: **d**
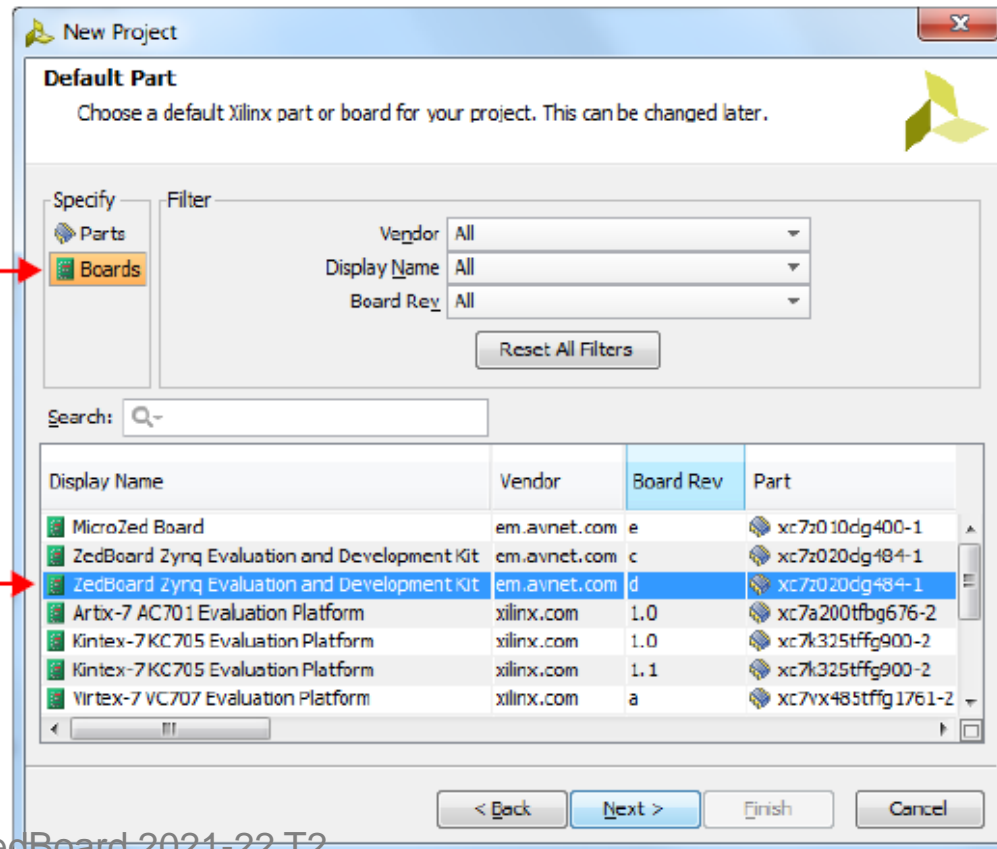
Hardware Setup for ZedBoard

- The ZedBoard must be connected to a power supply.

- By default, ZedBoard connects to the host PC for programming over USB-JTAG.

- An additional connection can be made over USB-UART.

  – If intending to facilitate simple board-PC communication using the Terminal application.

- Note that there is also a third micro-USB port for connecting USB peripherals (USB-OTG)

power

USB-JTAG (programming, host PC)

USB-UART (host PC comms)

USB-OTG (peripherals)
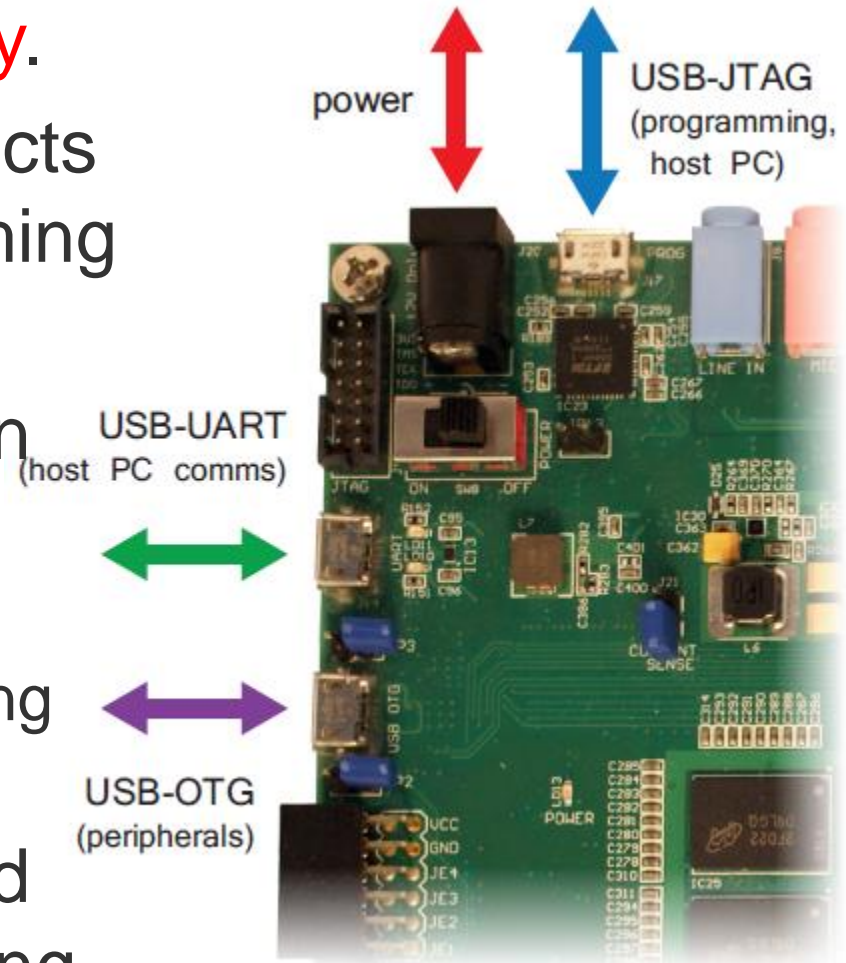
Programming the ZedBoard (1/2)

- ZedBoard can be programmed in four different ways:
    - **USB-JTAG:** This is the default and most straightforward method of programming the ZedBoard, given that it can be done directly over the USB-micro-USB cable supplied in the ZedBoard kit.
    - **Traditional JTAG:** A Xilinx JTAG connector is available on the board and may be used in place of the USB-JTAG connection, if desired. This will require a different type of cable or a *Digilent USB-JTAG programming cable*.
    - **Quad-SPI flash memory:** The non-volatile flash memory on the board can be used to store configuration data which persists when the board is powered off. Using this method removes the requirement for a wired connection to program the Zynq device.
    - **SD card:** There is an SD slot on the underside of the ZedBoard. This facility can be used to program the Zynq with files stored on the SD card, thus requiring no wired connections.

Zedboard Programming Guide in SDK: https://reference.digilentinc.com/learn/programmable-logic/tutorials/zedboard-programming-guide/start

- The ZedBoard user specifies the method of booting / programming via a set of jumper pins.
  - The middle three are for specifying programming source.

	MIO[6]	MIO[5]	MIO[4]	MIO[3]	MIO[2]
In Xilinx Technical Reference Manual...	Boot_Mode[4]	Boot_Mode[0]	Boot_Mode[2]	Boot_Mode[1]	Boot_Mode[3]
JTAG Mode					
Cascaded JTAG[a]	-	-	-	-	0
Independent JTAG	-	-	-	-	1
Boot Device					
JTAG	-	0	0	0	-
Quad-SPI (flash)	-	1	0	0	-
SD Card[a]	-	1	1	0	-
PLL Mode					
PLL Used[a]	0	-	-	-	-
PLL Bypassed	1	-	-	-	-

Cascaded: A single JTAG connection is used to interface to the debug access ports in both the PS and PL.
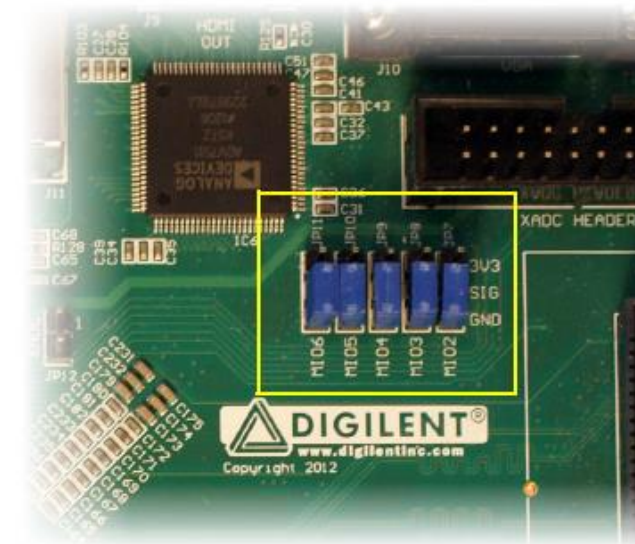
The *PLL* mode determines whether the process of configuring the device includes a phase of waiting for the PLL to lock.

Xilinx Design Constraints (XDC)

- XDC: A file that maps <u>external I/Os of the design</u> to <u>physical pins on the ZedBoard</u>.
  - External interfaces examples: switches, LEDs
  - For example: `C <= A and B;`
    - `set_property PACKAGE_PIN` **T22** `[get_ports {`**C**`}];` **# "LD0"**
    - `set_property PACKAGE_PIN` **F22** `[get_ports {`**A**`}];` **# "SW0"**
    - `set_property PACKAGE_PIN` **G22** `[get_ports {`**B**`}];` **# "SW1"**
    - Note: Voltage levels also need to be specified (see Lab02).

Class Exercise 2.1

- Complete the XDC file for the design `C <= A and B` with the following pin assignments on ZedBoard:
  - **A**: Switch #6 (SW6)
  - **B**: Switch #7 (SW7)
  - **C**: LED #7 (LD7)

Summary

- Digital System Design Basics
  - Integrated Circuit Technology
  - Design Flow of Digital Systems
  - Spectrum of Design Technologies

- Zynq: All-Programmable SoC (APSoC)
  - Our Board: ZedBoard
    - ZedBoard Layout and Interfaces
    - Specifying ZedBoard in Vivado
    - Hardware Setup for ZedBoard
    - Programming the ZedBoard
    - Xilinx Design Constraints (XDC) File