CS221: Digital Design

http://jatinga.iitg.ernet.in/~asahu/cs221

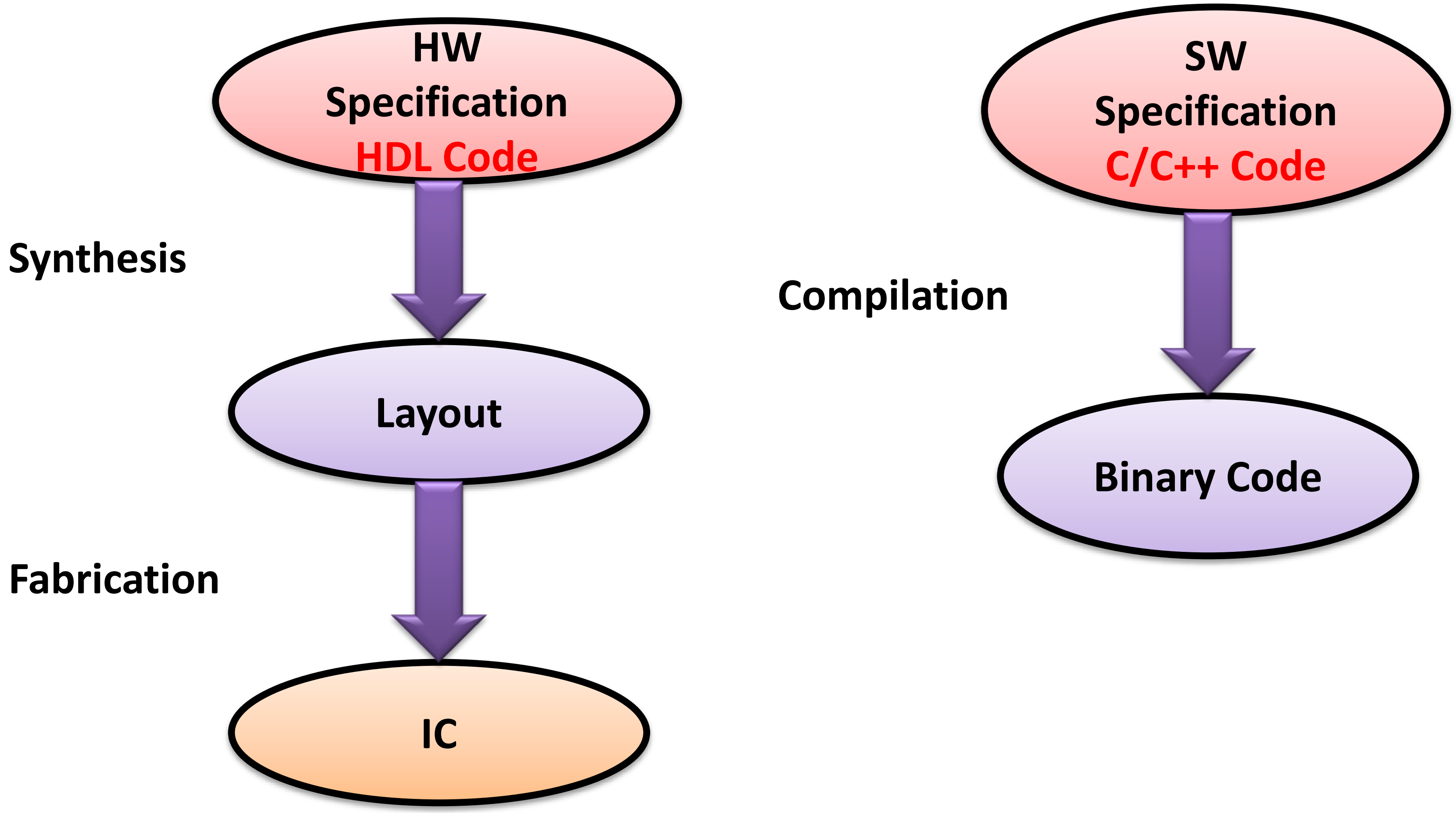
FPGA and HDL

A. Sahu
Dept of Comp. Sc. & Engg.
Indian Institute of Technology Guwahati

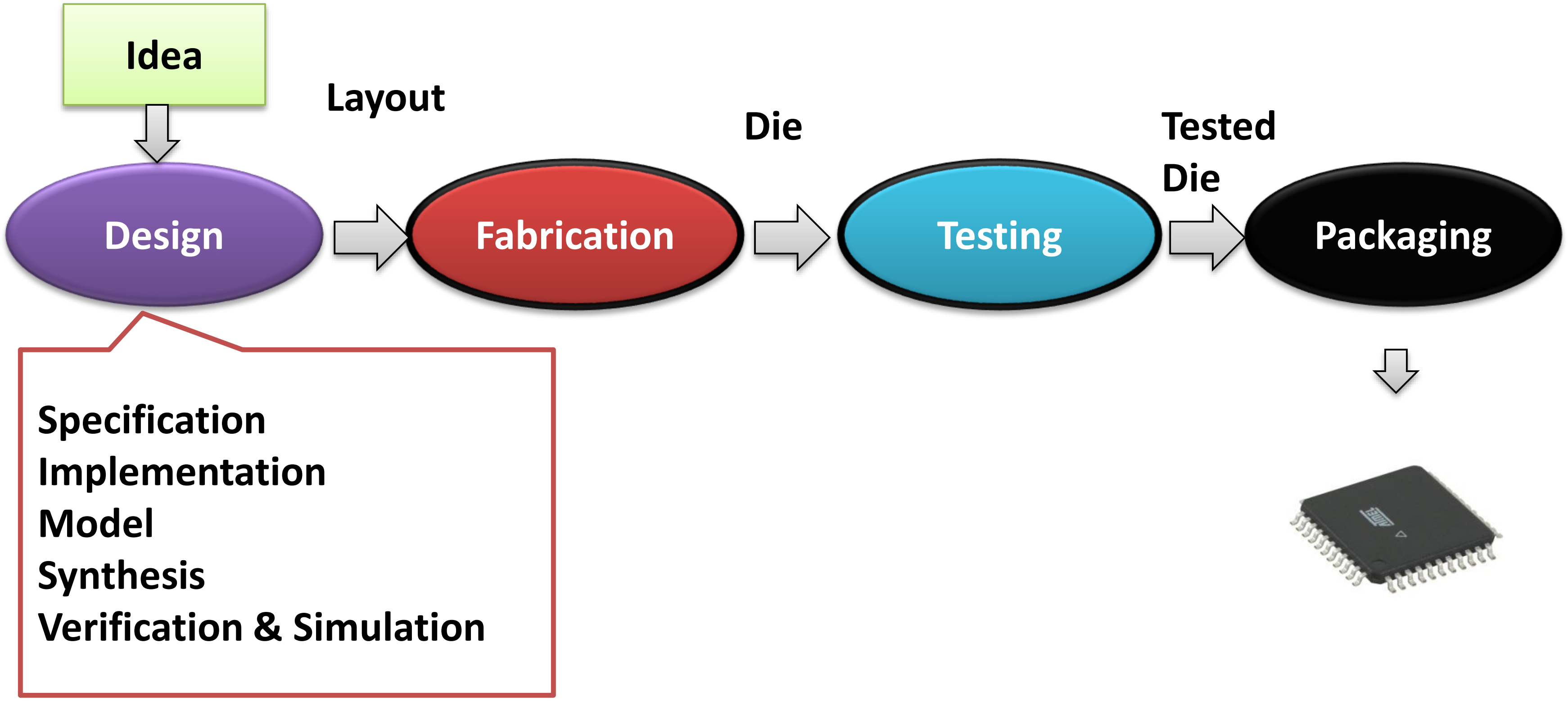
Outline

- IC Design process
- FPGA
- HDL
  - Requirement of VHDL, Model
  - Basic language concepts
  - Basic design methodology
  - Examples

Design Flow: Hardware Vs Software



IC Design Process



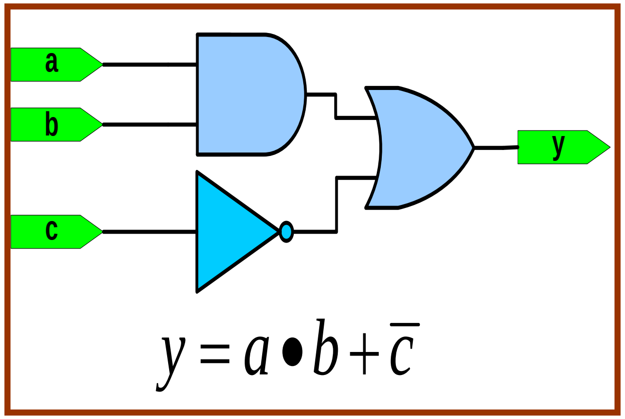
Specification
Implementation
Model
Synthesis
Verification & Simulation

FPGA:-
Field Programmable Gate Array

LUT

- LUT is a RAM with data width of 1bit.
- The contents are programmed at power up

Required Function

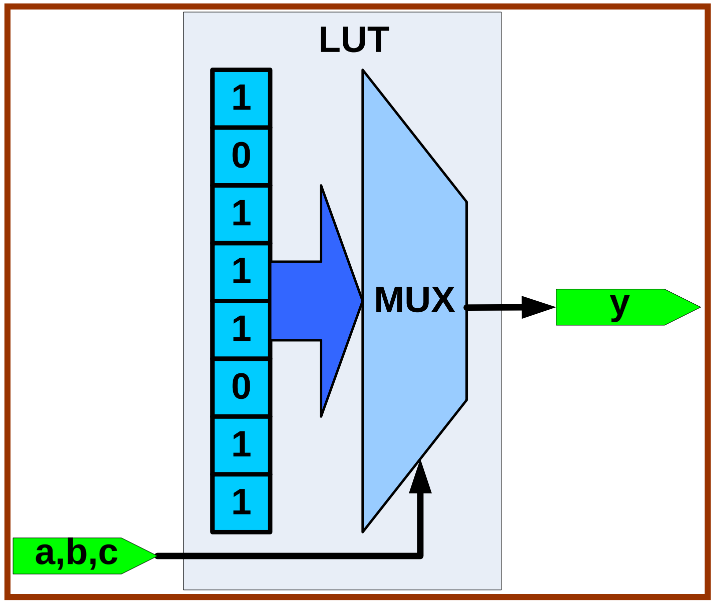


$$y = a \bullet b + \overline{c}$$

Truth Table

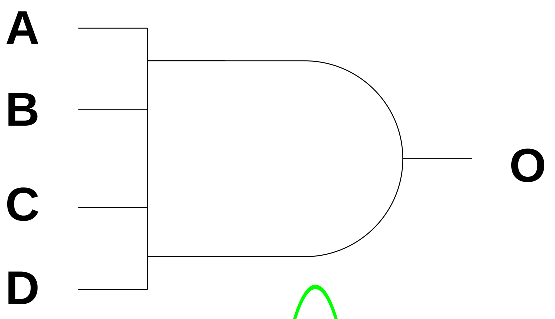
a	b	c	y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

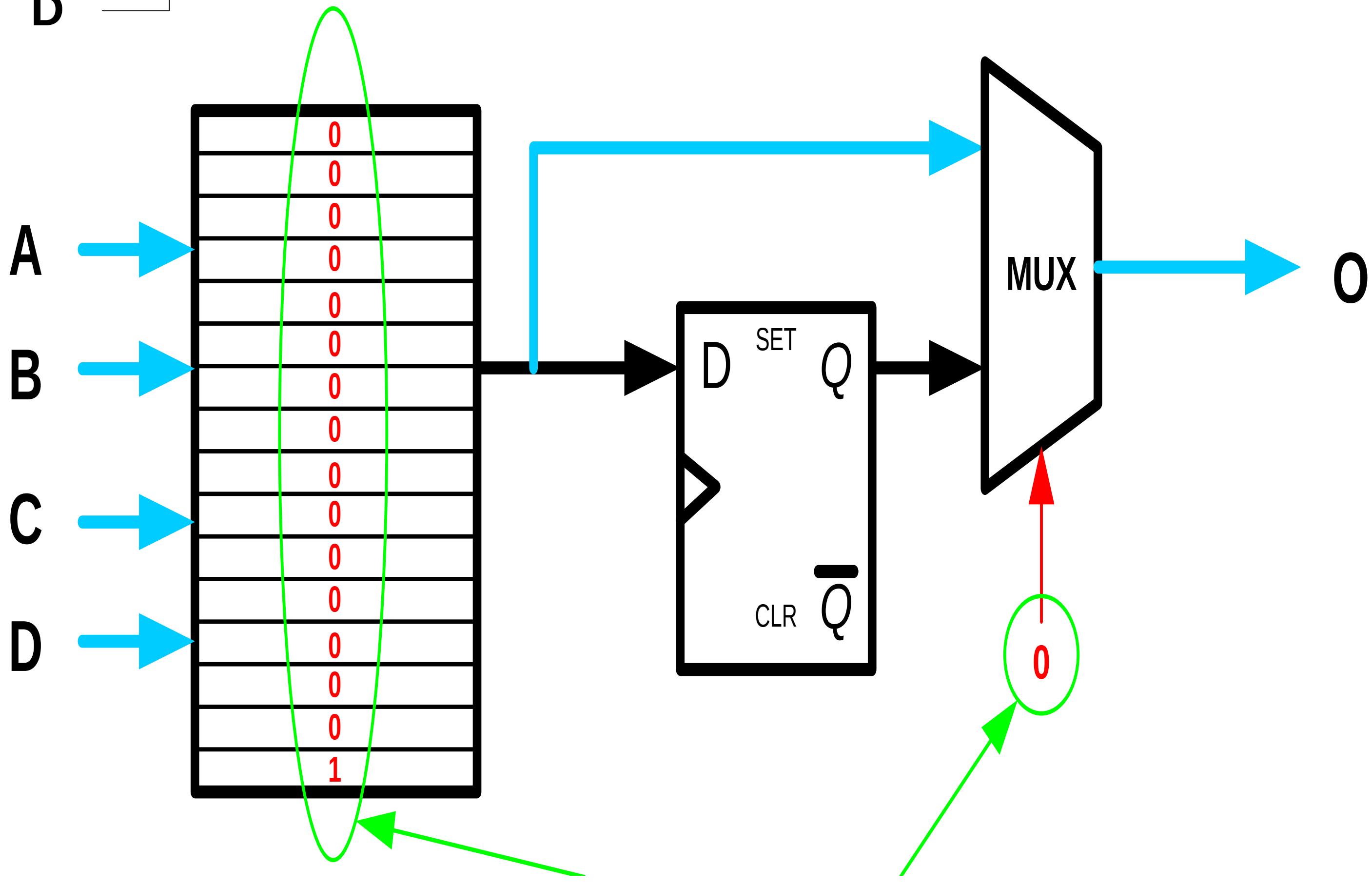
Programmed LUT



Example: 4-input AND gate

A	B	C	D	O
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1





Configuration bits

Shannon's expansion theorem

Used to implement many variable logic functions using MUX and LUTs

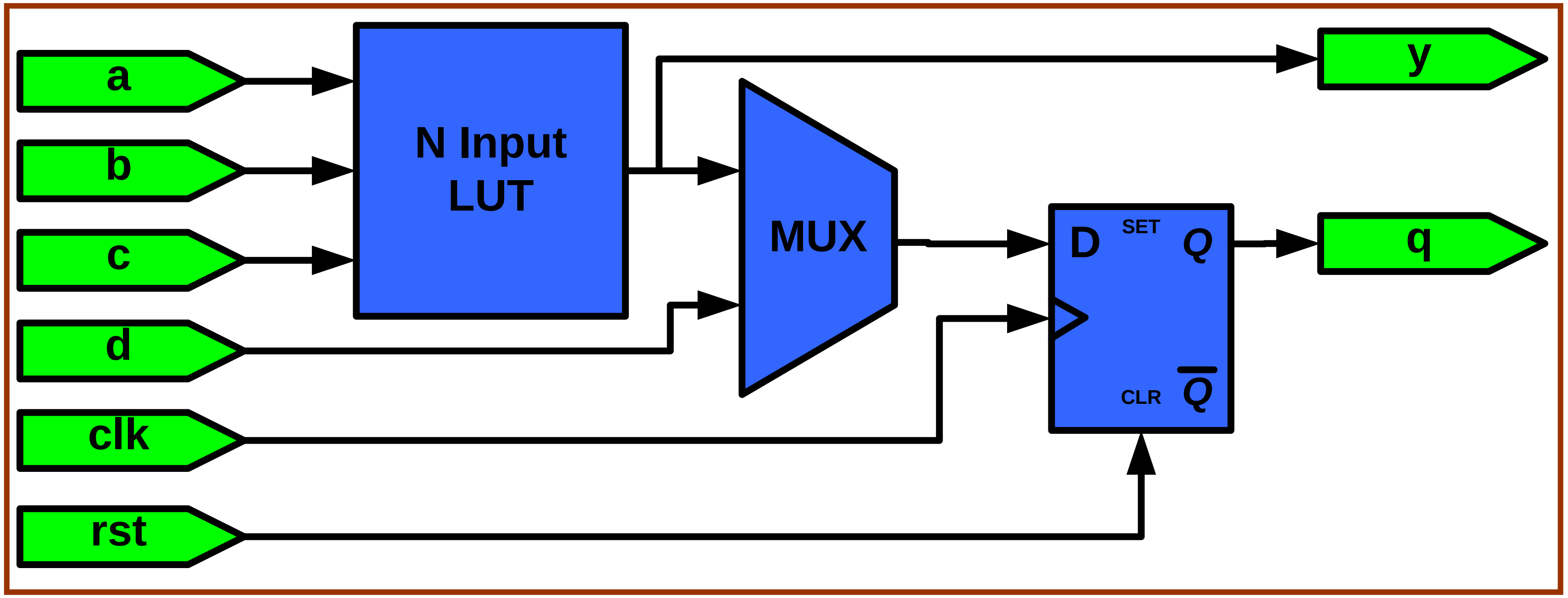
f(x1, x2, …, xn) = x1' f(0, x2, .., xn) + x1 f(1, x2, …, xn)

Since x1 is a boolean variable, we need to look at only two cases: x1 = 0 and x1 = 1.

- **Setting x1 = 0 in the above expression, we have:**
  - **f(0, x2, …, xn) = 1 f(0, x2, …, xn) + 0 f(1, x2, …, xn)**
    = f(0, x2, …, xn)
- **Setting x1 = 1, we have:**
  - **f(1, x2, …, xn) =0 f(0, x2, …, xn) + 1 f(1, x2, …, xn)**
    = f(1; x2; …; xn)

FPGA - Field Programmable Gate Array

- Programmable logic blocks or CLB
  - (Logic Element "LE")
  - Implement combinatorial and sequential logic. Based on LUT and DFF.

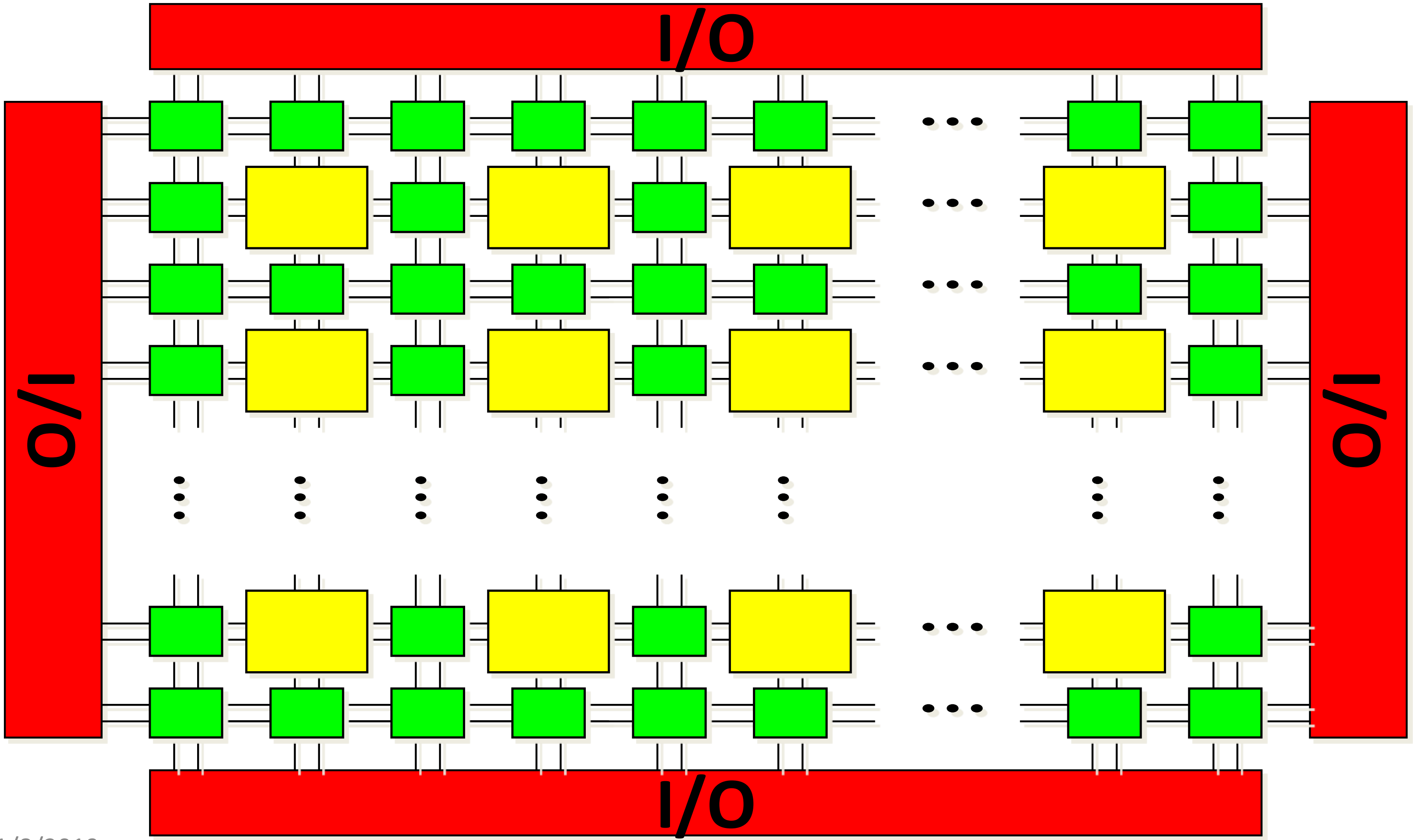


FPGA - Field Programmable Gate Array

- Programmable logic blocks (Logic Element “LE”) or CLB
  - Implement combinatorial and sequential logic. Based on LUT and DFF.
- Programmable I/O blocks
  - Configurable I/Os for external connections supports various voltages and tri-states.
- Programmable interconnect
  - Wires to connect inputs , outputs and logic blocks.
  - Clocks
  - short distance local connections
  - long distance connections across chip

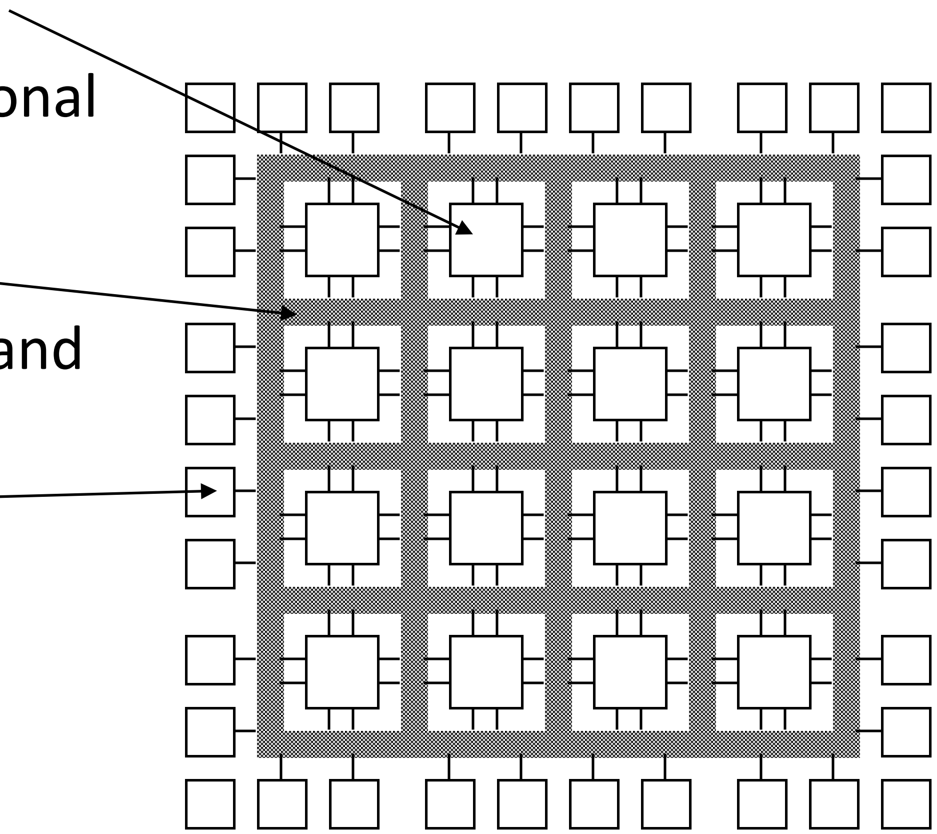
FPGA - Field Programmable Gate Array

🟩 Logic block 🟨 Interconnection switches

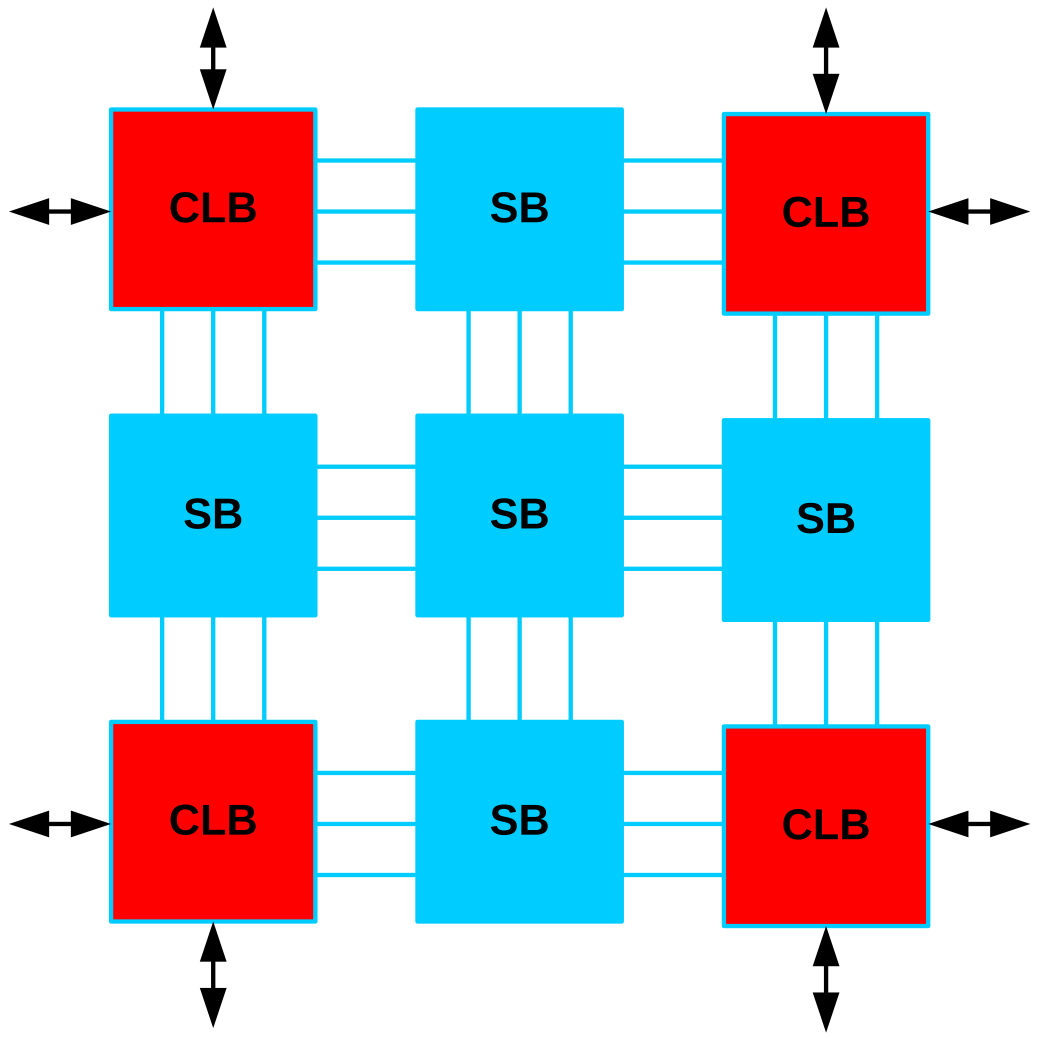


Field-Programmable Gate Arrays structure

- **Logic blocks**
  - To implement combinational and sequential logic
- **Interconnect**
  - Wires to connect inputs and outputs to logic blocks
- **I/O blocks**
  - Special logic blocks at periphery of device for external connections
- **Key questions:**
  - How to make logic blocks programmable?
  - How to connect the wires?



FPGA structure

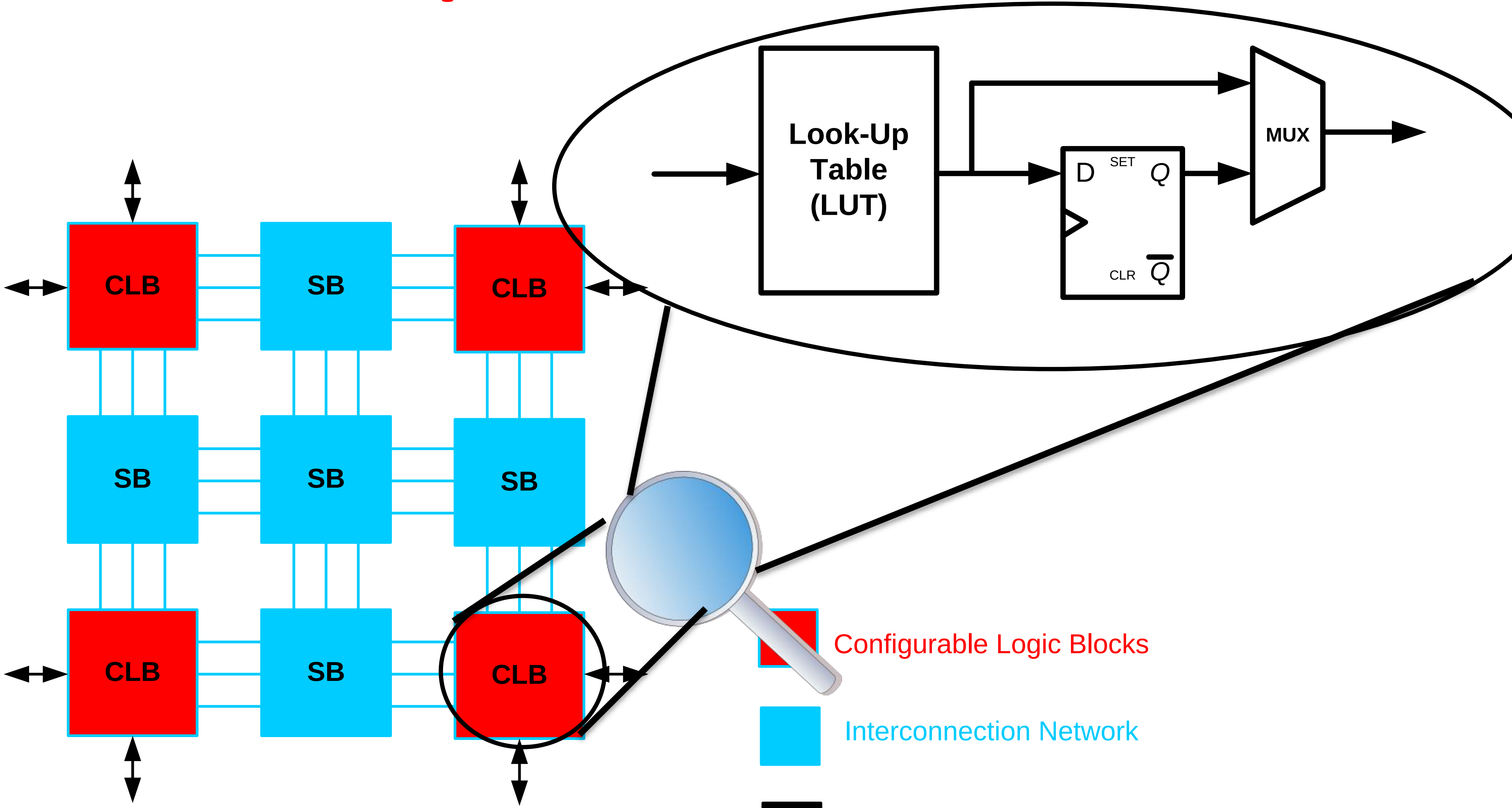


🟥 Configurable Logic Blocks

🟦 Interconnection Network

⬛ I/O Signals (Pins)

Simplified CLB Structure

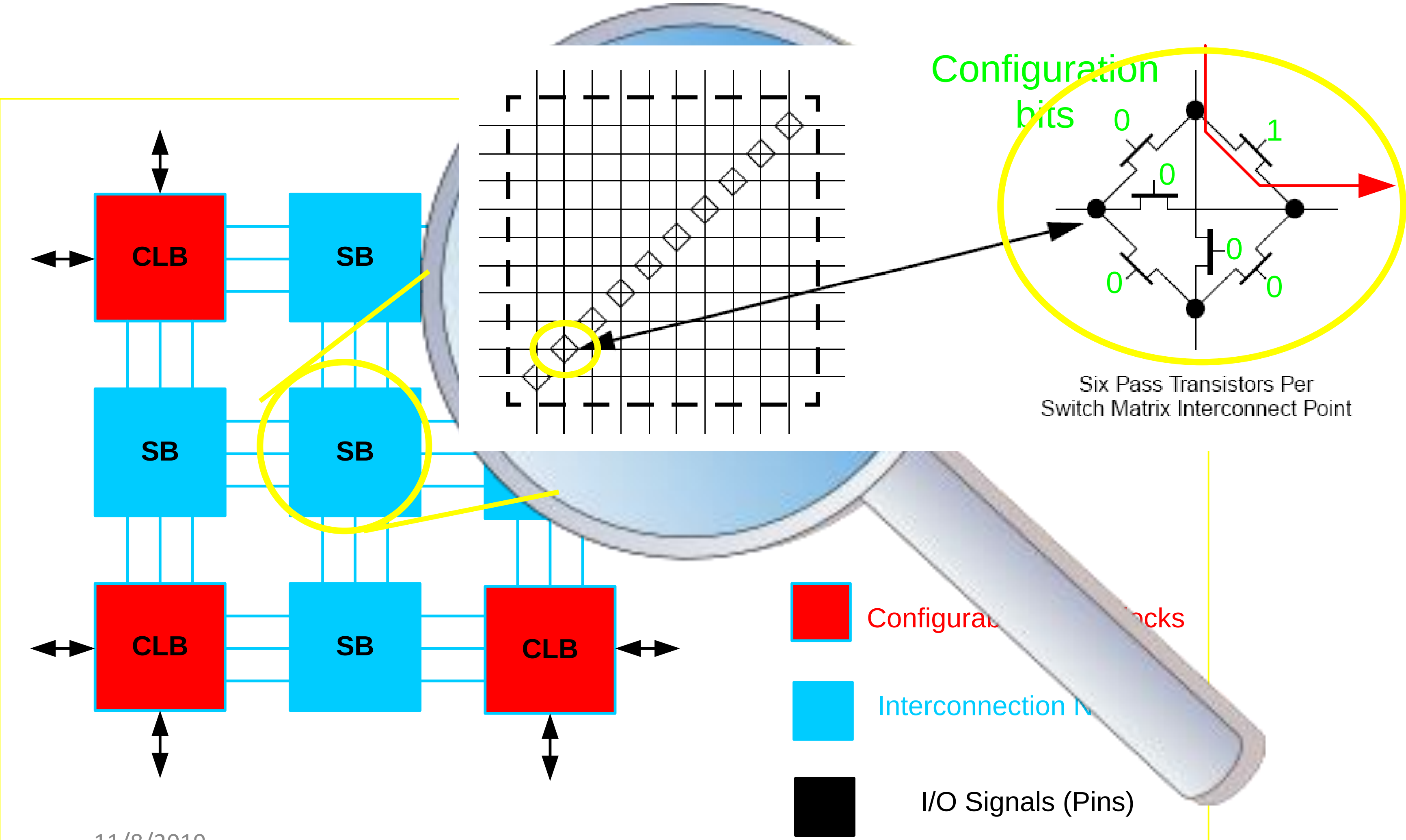


🟥 Configurable Logic Blocks

🟦 Interconnection Network

⬛ I/O Signals (Pins)

Interconnection Network

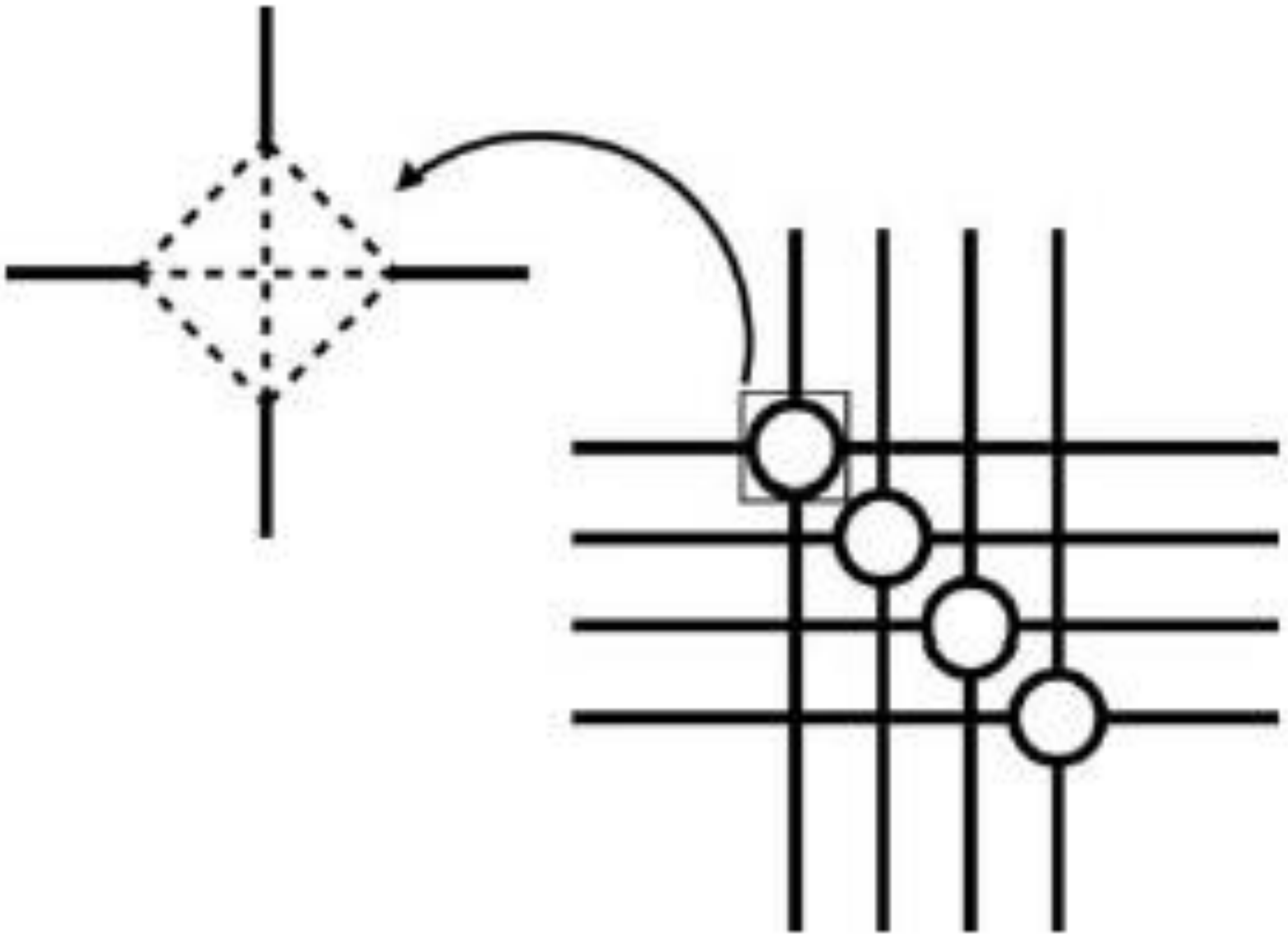


🟥 Configurab… …ocks

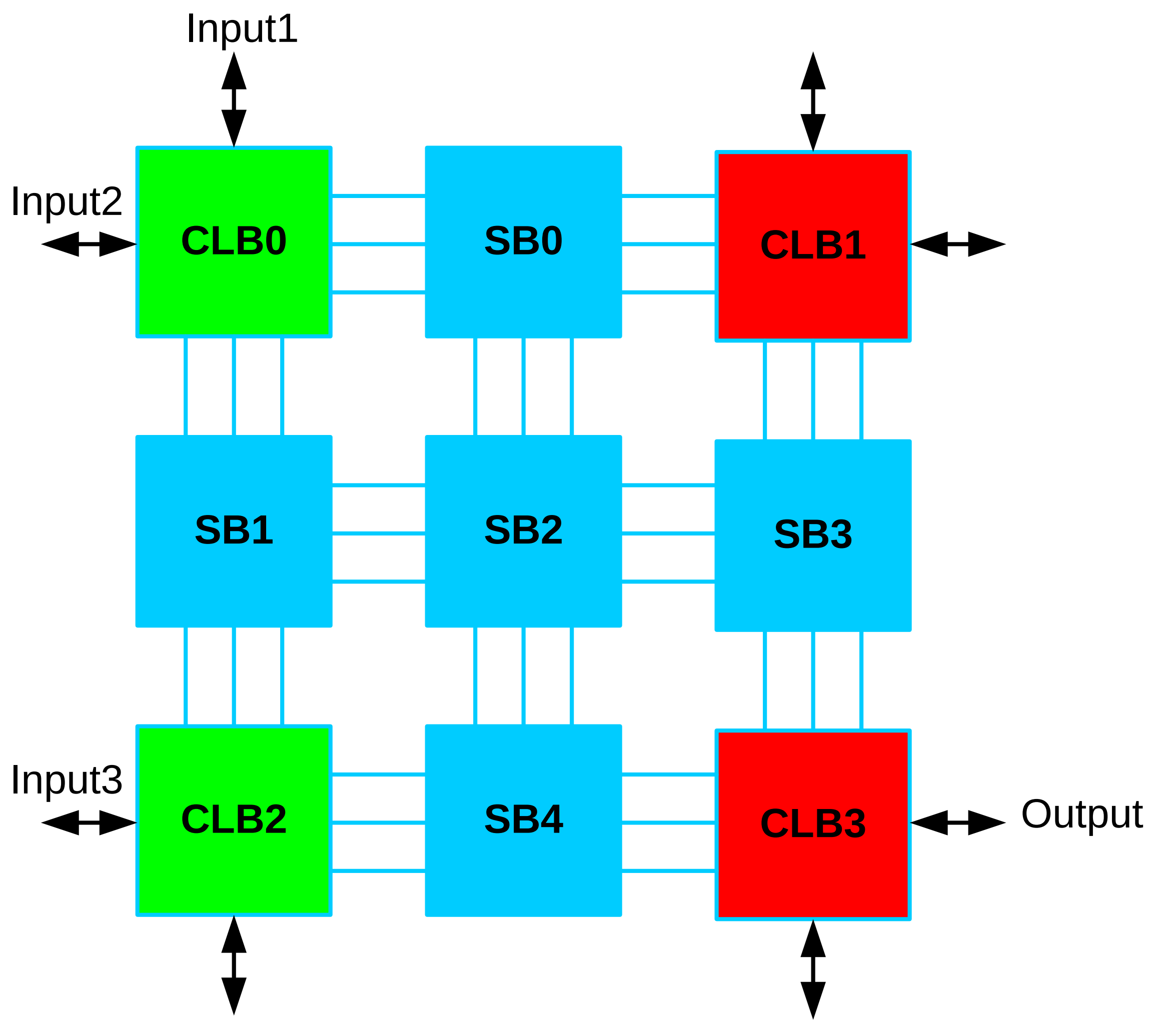
🟦 Interconnection N…

⬛ I/O Signals (Pins)

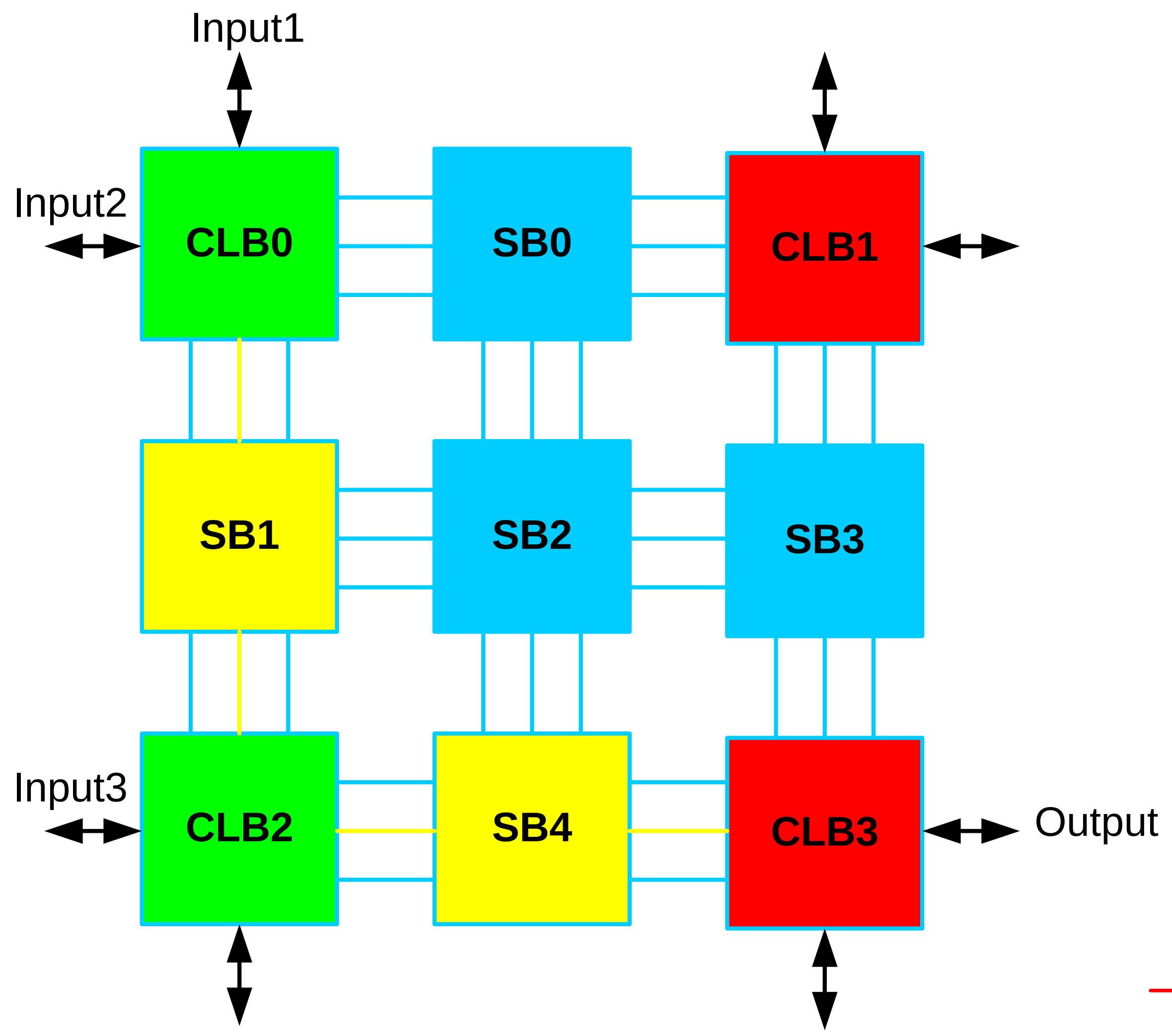
Configurable Interconnect



Placement: Select CLBs

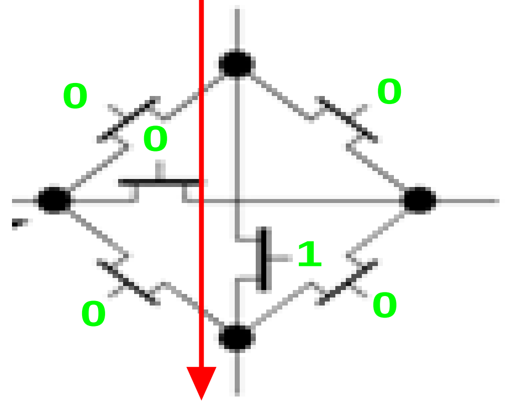


Routing: Select path



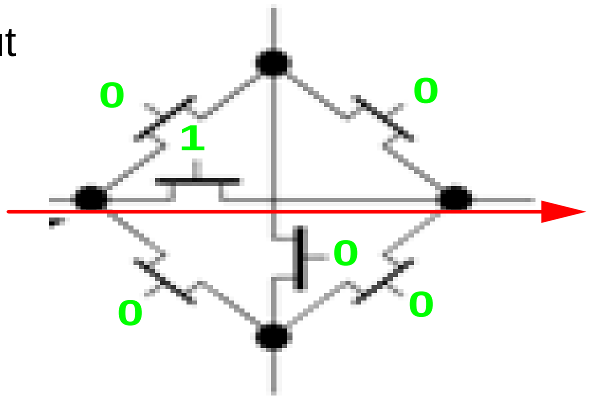
SB1

Configuration bits



SB4

Configuration bits



FPGA Advantages

- **Designing with FPGA: Faster, Cheaper**
- **Ideal for customized designs**
  - Product differentiation in a fast-changing market
- **Offer the advantages of high integration**
  - High complexity, density, reliability
  - Low cost, power consumption, small phy. size
- **Avoid the problems of ASICs**
  - high NRE cost, long delay in design and testing
  - increasingly demanding electrical issues

FPGA Advantages

- **Very fast custom logic**
  - massively parallel operation
- **Faster than micro-controllers/and processors**
  - much faster than DSP engines
- **More flexible than dedicated chipsets**
  - allows unlimited product differentiation
- **More affordable and less risky than ASICs**
  - no NRE, min order size, or inventory risk
- **Reprogrammable at any time**
  - in design, in manufacturing, after installation

User Expectations

- **Logic capacity at reasonable cost**
  - 100,000 to a several million gates
  - On-chip fast RAM
- **Clock speed**
  - 150 MHz and above, global clocks, clock management
- **Versatile I/O**
  - To accommodate a variety of standards
- **Design effort and time**
  - synthesis, fast compile times, tested and proven cores
- **Power consumption**
  - must stay within reasonable limits

Field Programmable Device

- **Basic Section of FPD:**
  - Logical Block
  - Routing (Switch Matrix)
  - Input Output Block
- **More Advanced FPD Contains:**
  - On-chip Memory
  - Embedded Processor
  - Clock Management
  - High-Speed Transceiver

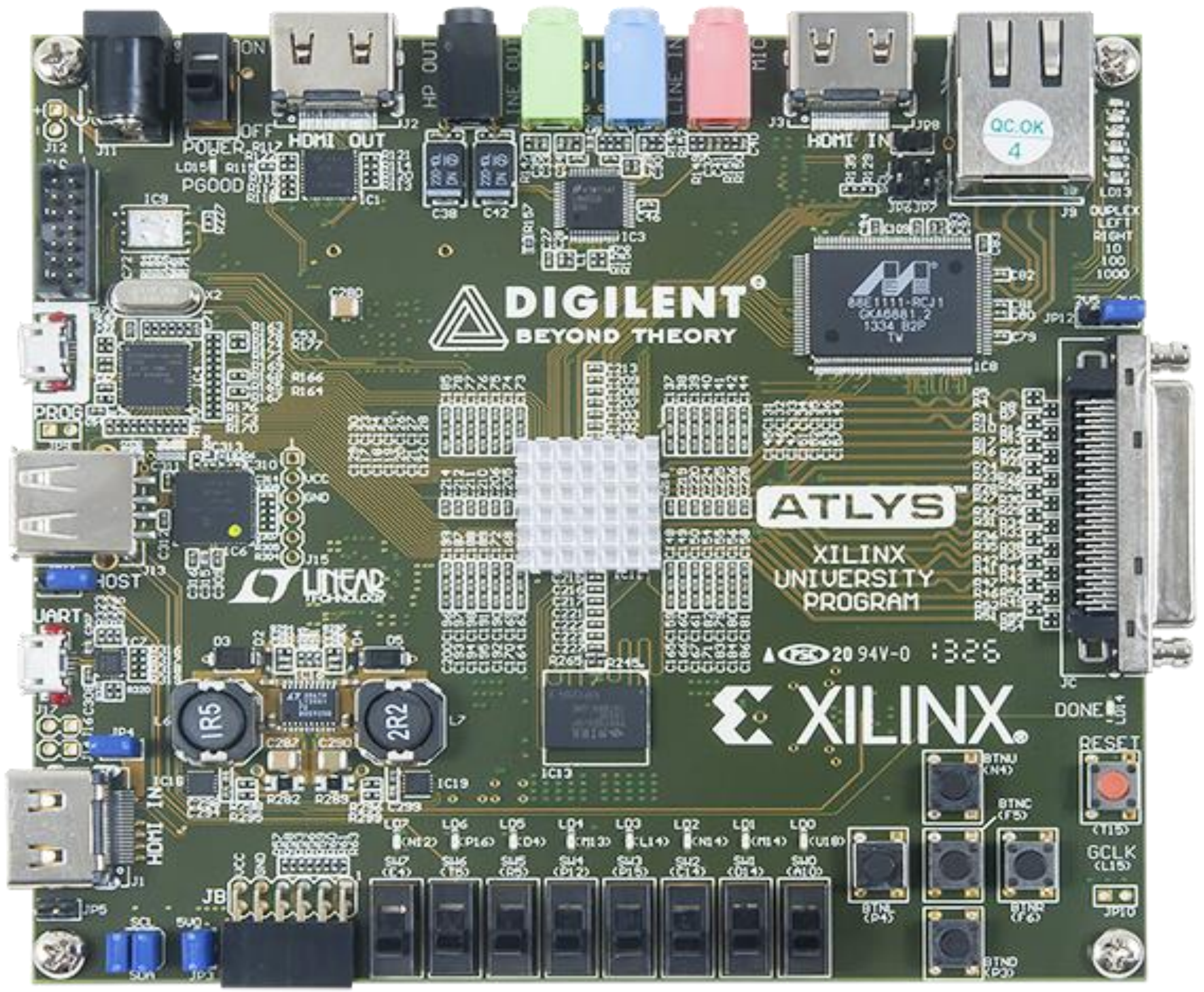
Special FPGA functions

- Internal SRAM
- Embedded Multipliers and DSP blocks
- Embedded logic analyzer
- **Embedded CPUs**
- High speed I/O (~10GHz)
- DDR/DDRII/DDRIII SDRAM interfaces
- PLLs

Digilent Xilinx Atlys FPGA Board

- Xilinx Spartan-6 LX45 FPGA,
- 6,822 slices : four 6-input LUTs and eight flip-flops
- 2.1Mbits of block RAM, 128 MB DDR2
- 58 DSP slices
- JTAG programming , RJ-45 Ethernet port
- 4 HDMI video ports
- AC-97 Audio Codec mic, & headphone
- Two on-board USB2
- USB-UART and USB-HID port
- GPIO includes 8 LEDs, 6 buttons, and 8 slide switches

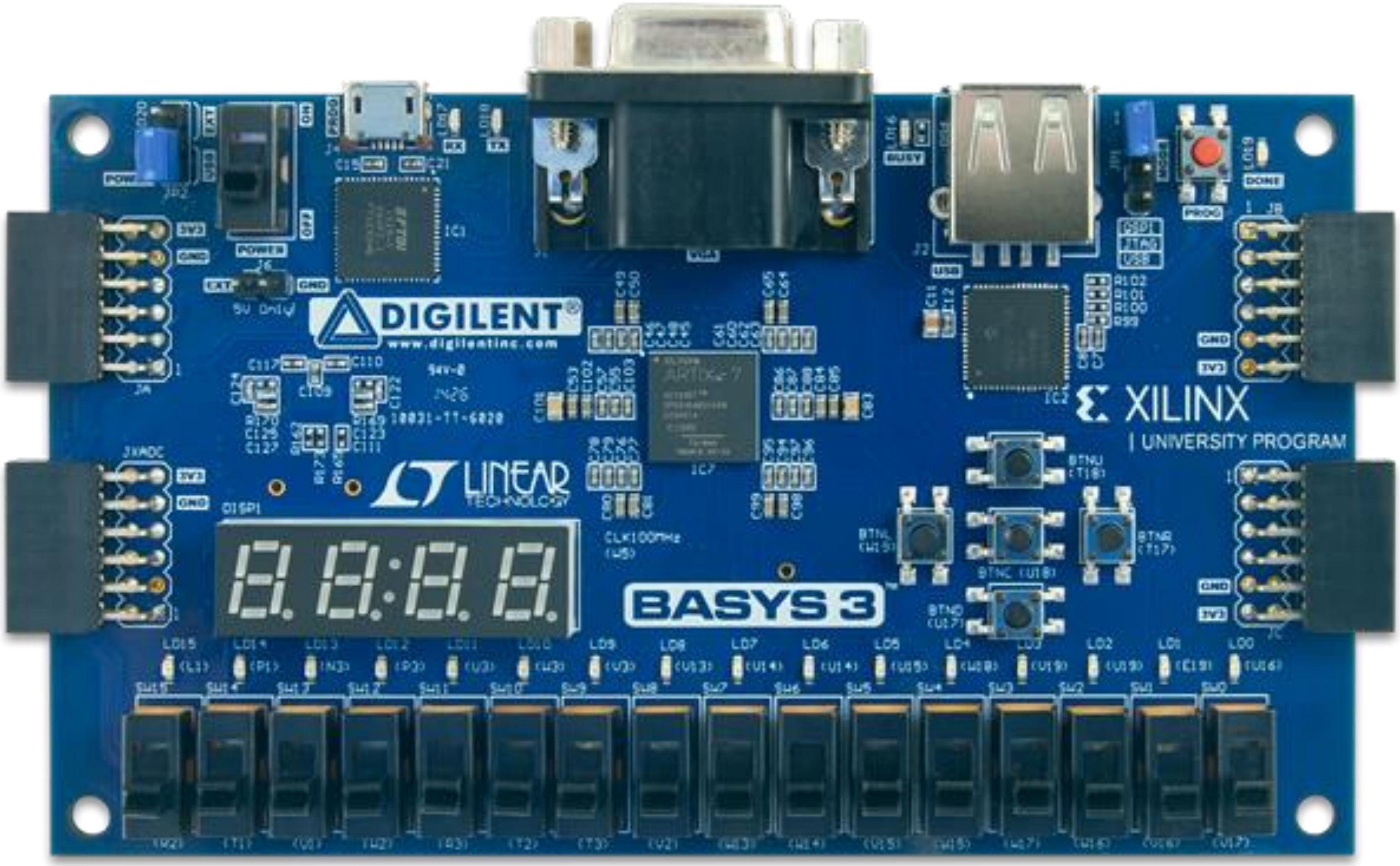
Digilent Xilinx Atlys FPGA Board



Digilent Xilinx Basys 3 FPGA Board

- Xilinx Artix-7 FPGA
- 33,280 logic cells in 5200 slices (four 6-input LUTs and 8 flip-flops)
  1,800 Kbits of fast block RAM
- 90 DSP slices
- ADC, USB-JTAG port, USB-UART Bridge , 12-bit VGA output
- USB HID Host for mice, keyboards
- 16 user switches, 16 user LEDs, 5 user pushbuttons , 4-digit 7-segment display

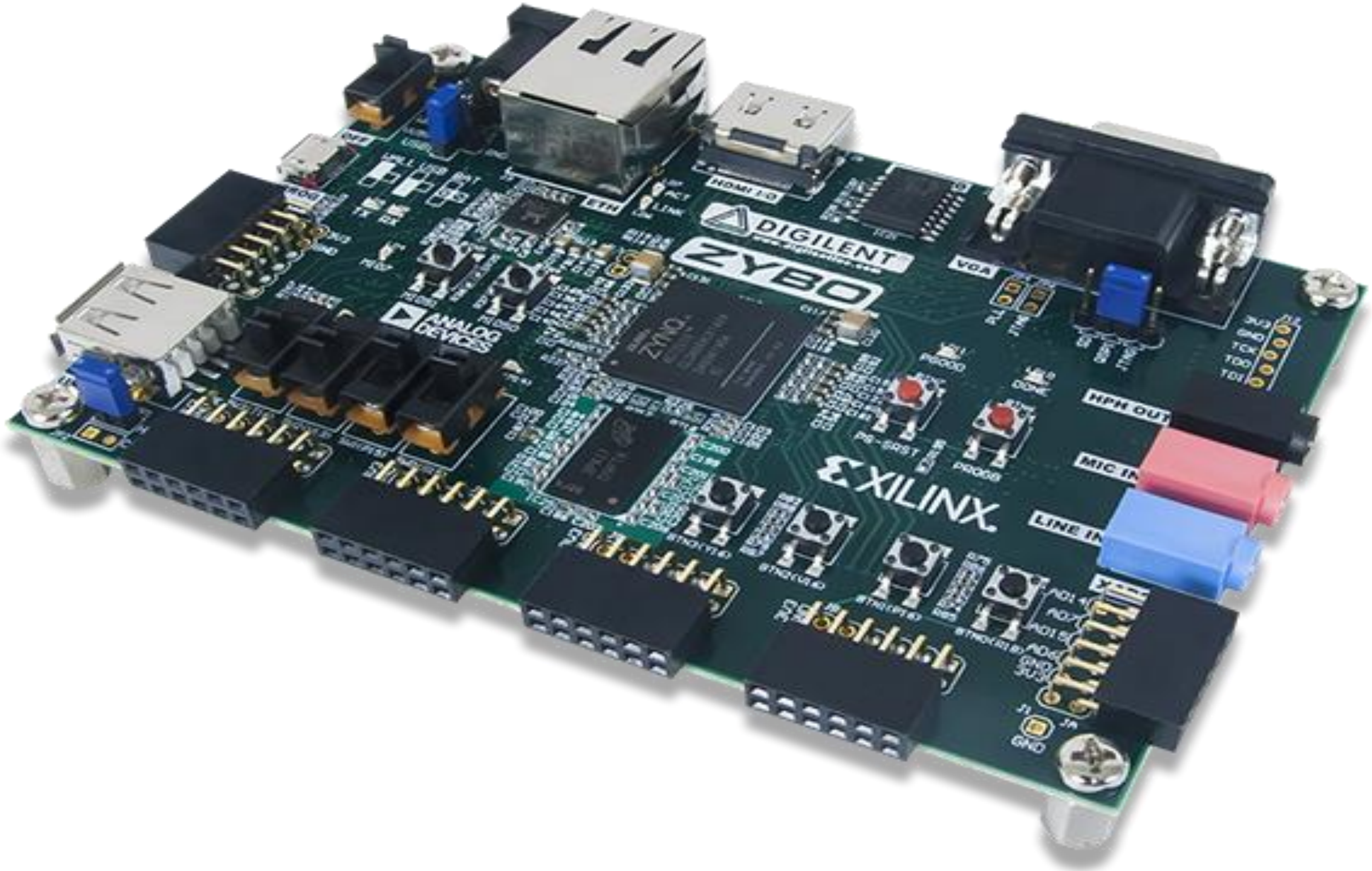
Digilent Xilinx Basys 3 FPGA Board



Digilent Xilinx Zybo FPGA Board

- Xilinx Zynq-7000 (XC7Z010-1CLG400C)
- 28,000 logic cells
- 240 KB Block RAM
- 80 DSP slices , Dual channel, 12-bit, ADC
- **650 MHz dual-core Cortex™-A9 processor**
- JTAG programming and UART to USB converter
- 1G Ethernet, USB 2.0, SDIO, SPI, UART, I2C
- Dual-role HDMI port , VGA port, Ethernet PHY
- OTG USB 2.0 PHY (supports host and device)
- GPIO: 6 pushbuttons, 4 slide switches, 5 LEDs

Digilent Xilinx ZYBO FPGA Board



Modeling Digital Systems

- VHDL is for coding models of a digital system...
- Reasons for modeling
  - requirements specification
  - documentation
  - testing using simulation
  - formal verification
  - synthesis
- Goal
  - most ‘reliable’ design process, with minimum cost and time
  - avoid design errors!

HDL

- HDL is NOT C ...
  There are some similarities, as with any programming language, but syntax and logic are quite different; so get over it !!

Requirement of any HDL -1

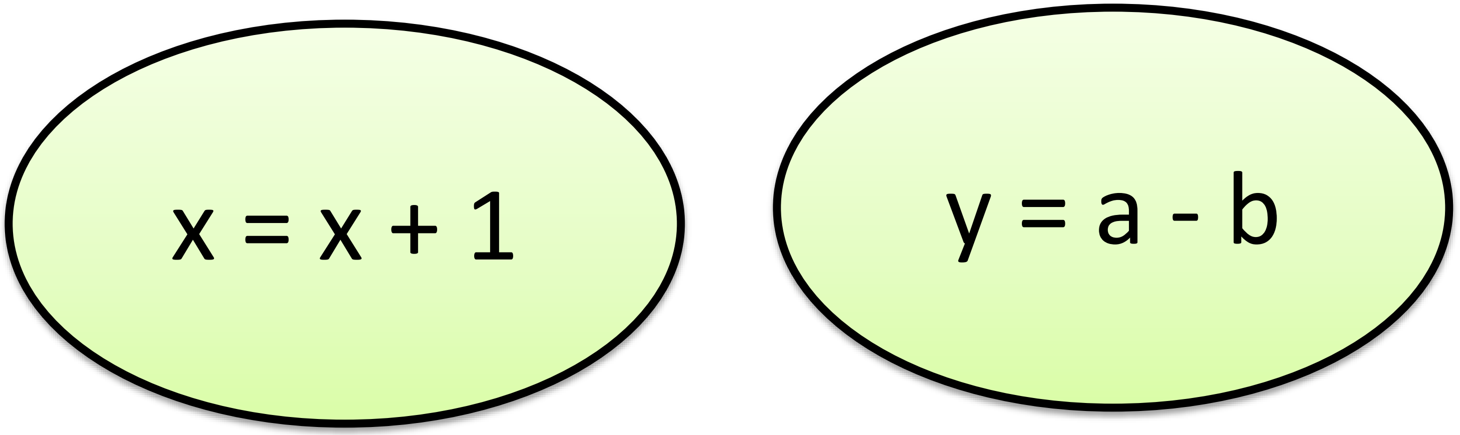
- Time
  - How the behavior of the system changes with time
  - Creating waveforms

Requirement of any HDL -2

- Periodic Signals
  - Clocks

Requirement of any HDL -3

- Concurrency

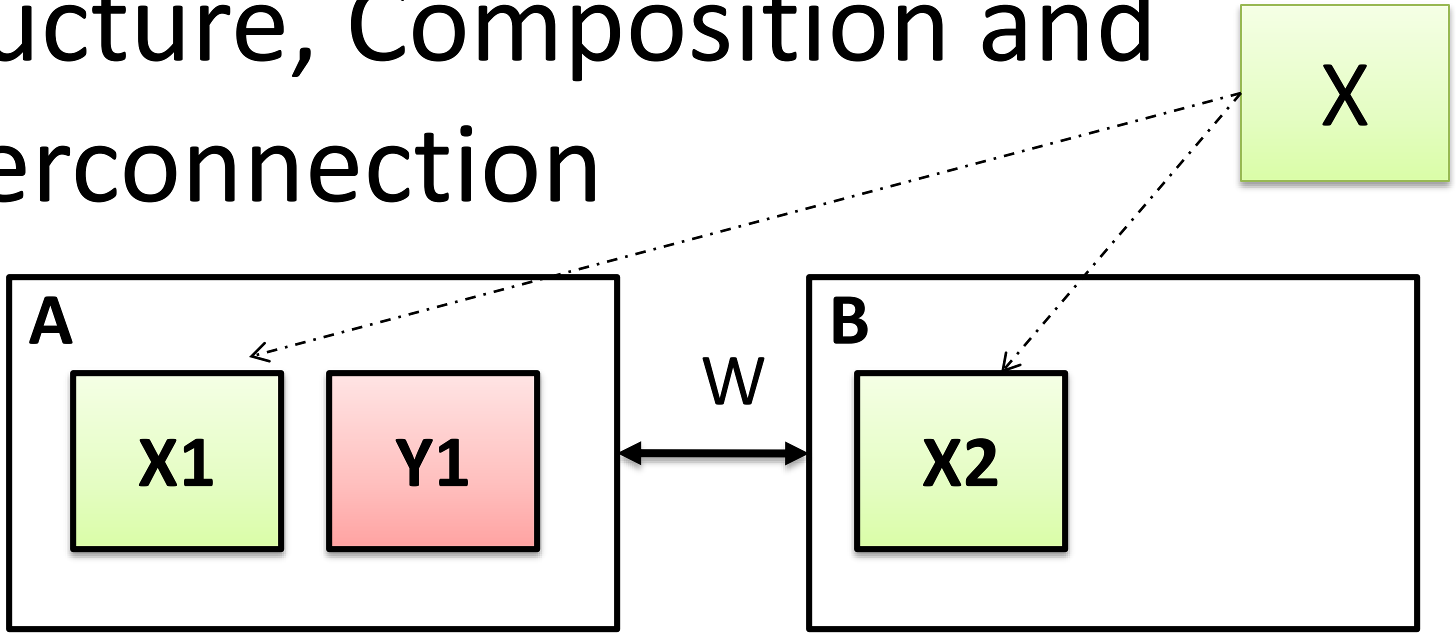


P1 P2

- Specify: Processes P1 and P2 execute in parallel

Requirement of any HDL -4

- Structure, Composition and Interconnection



  - Block A consists of two blocks: X1 and Y1
  - Block X is duplicated
  - Wire W connects A and B

Requirement of any HDL -5

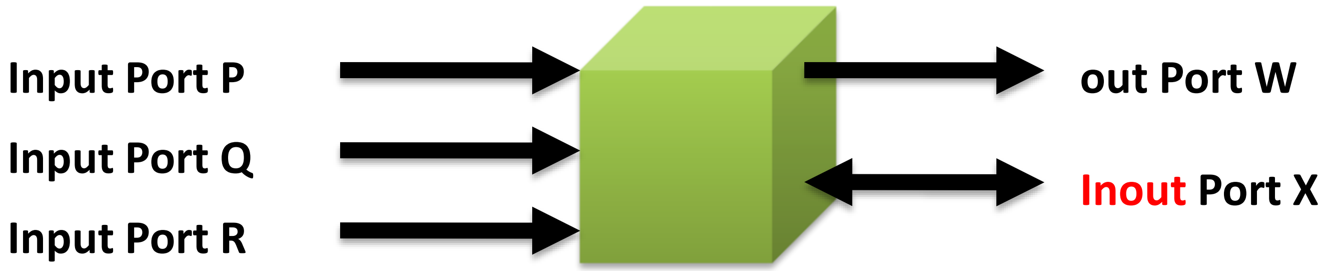
- Bit-true data types
  - Not so important in SW
  - Important in HW

```
int<6:0> var;
```

- Specify the bit-width of variables

Requirement of any HDL -6

- Modules and Interfaces
  - Ports



Requirement of any HDL -7

- Electrical Characteristics
  - Current Levels
  - Tristating
- Sensitivity
  - Rising edge/falling edge

Requirement of any HDL -8

- Other programming constructs
  - Text and File I/O
- useful in simulation/debugging

Requirement of any HDL -9

- Bit-true data types
  - Not so important in SW
  - Important in HW

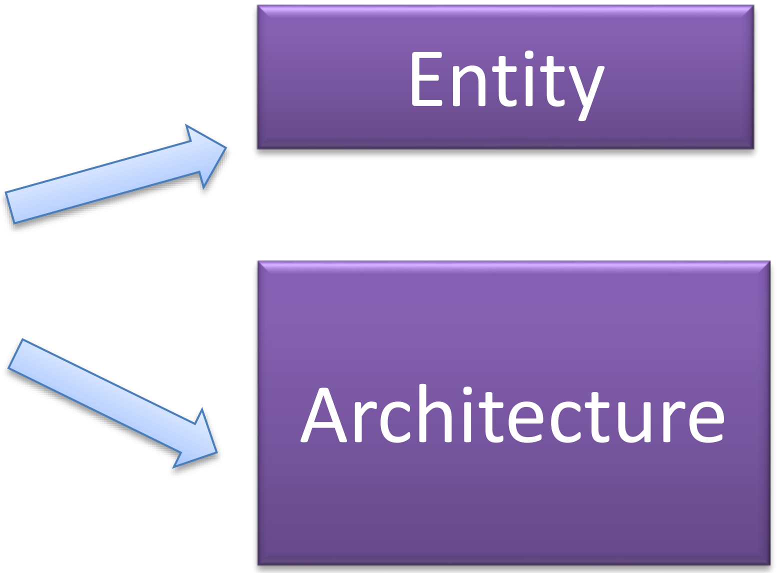
```
int<6:0> var;
```

- Specify the bit-width of variables

Fundamental VHDL Objects

- Entity and Architecture Pair

VHDL Model Consists of Two Parts



Entity Represent External Interface

Arch Represent Contents/Functionality

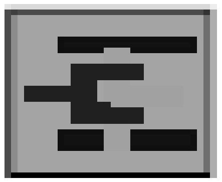
FPGA and ASIC Design Flow

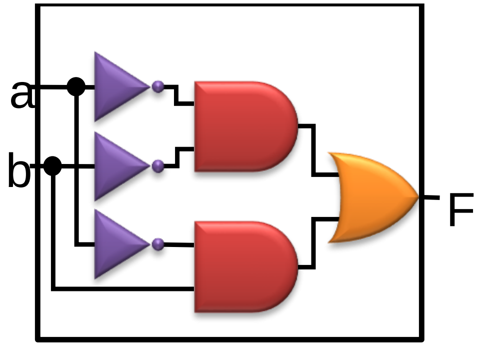
Model

- Representation of abstract view of the System
- Varying abstractions
  - functional only
  - timing only
  - functional + timing

Hardware Specification

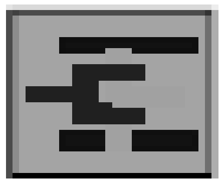
- Layout editor
  - directly enter layout
  - Up to ~10^2 of unique transistors
  - Complex circuits
  - Memory, aided by generators
- Schematic Capture
  - Enter gates and interconnections
  - Up to ~10^4 transistors
- Hardware Description Languages
  - Enter text description
  - 10^7 transistors

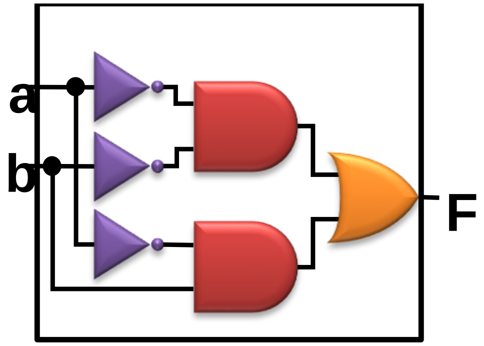




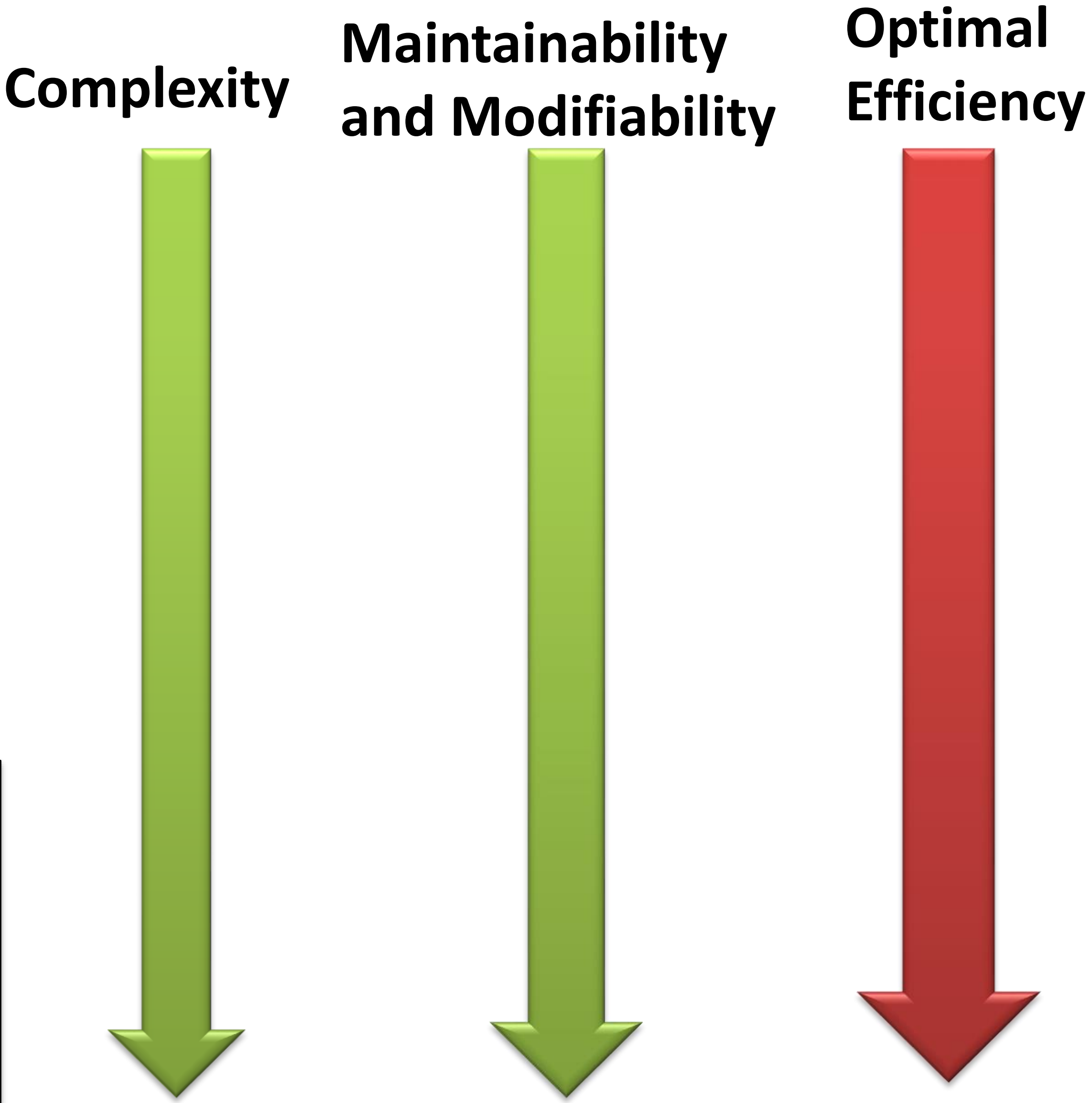
```
module ..
If (x < y) then
Y=x and z;
....
```

Hardware Specification



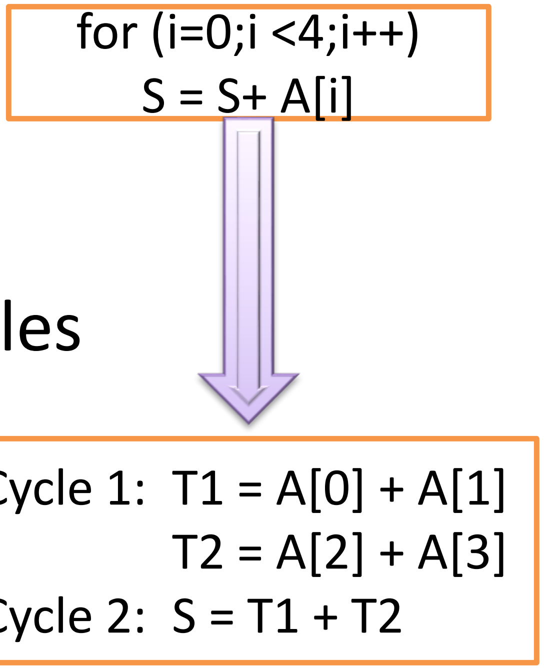


Entity ..
If (x < y) then
Y=x and z;
….



Modelling: level of detail

- Behavioral Level
  - no clock cycle level commitment
- Register-Transfer Level (RTL)
  - Operations committed to clock cycles



```
for (i=0;i <4;i++)
   S = S+ A[i]
```

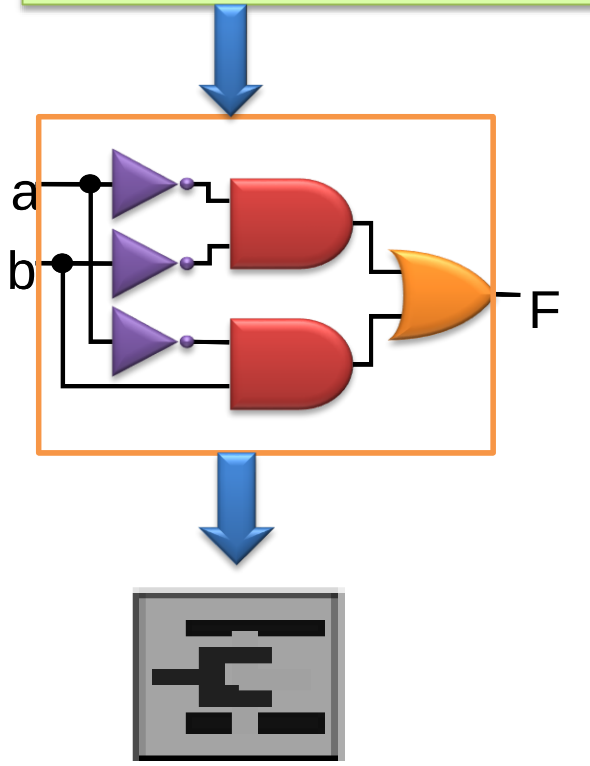
```
Cycle 1:  T1 = A[0] + A[1]
          T2 = A[2] + A[3]
Cycle 2:  S = T1 + T2
```

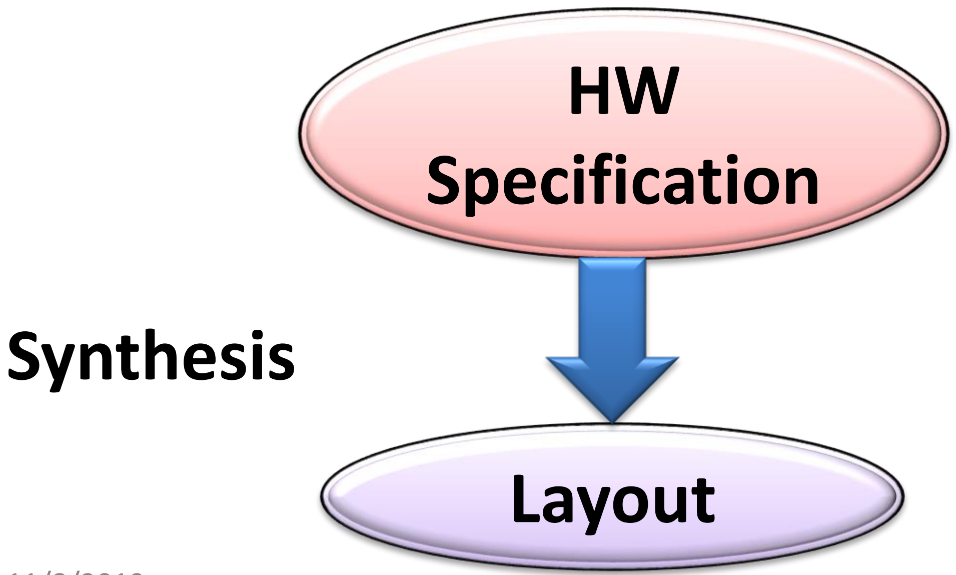
- Gate level
  - structural netlist

Synthesis

- HDL → Layout
  - HDL → Gates
  - Gates → Layout

```
module ..
If (x < y) then
Y=x and z;
….
```

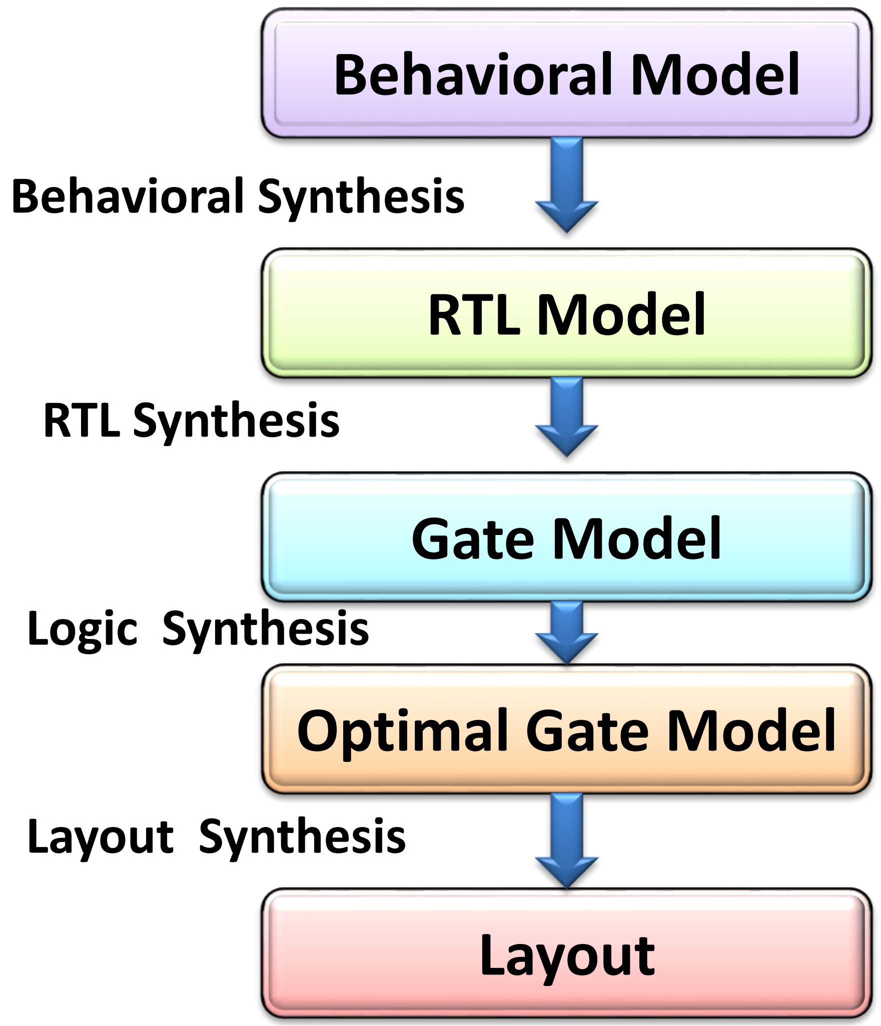




Synthesis

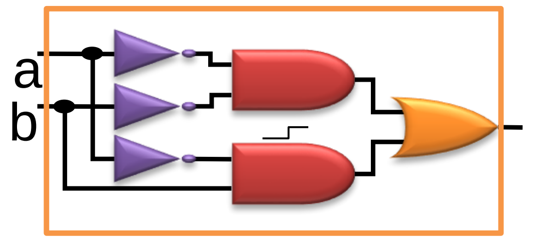
- Behavioral Synthesis (Process & Sequential)
  - Behavioral HDL → RTL HDL
  - No notion of clock to Clocked
- RTL Synthesis
  - RTL HDL → Gates
- Layout Synthesis
  - Gates → Layout

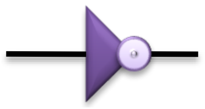
Design Flow

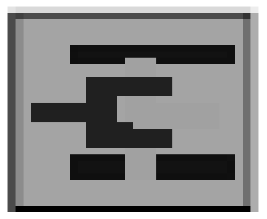


```
for (i=0;i <4;i++)
  S = S+ A[i]
```

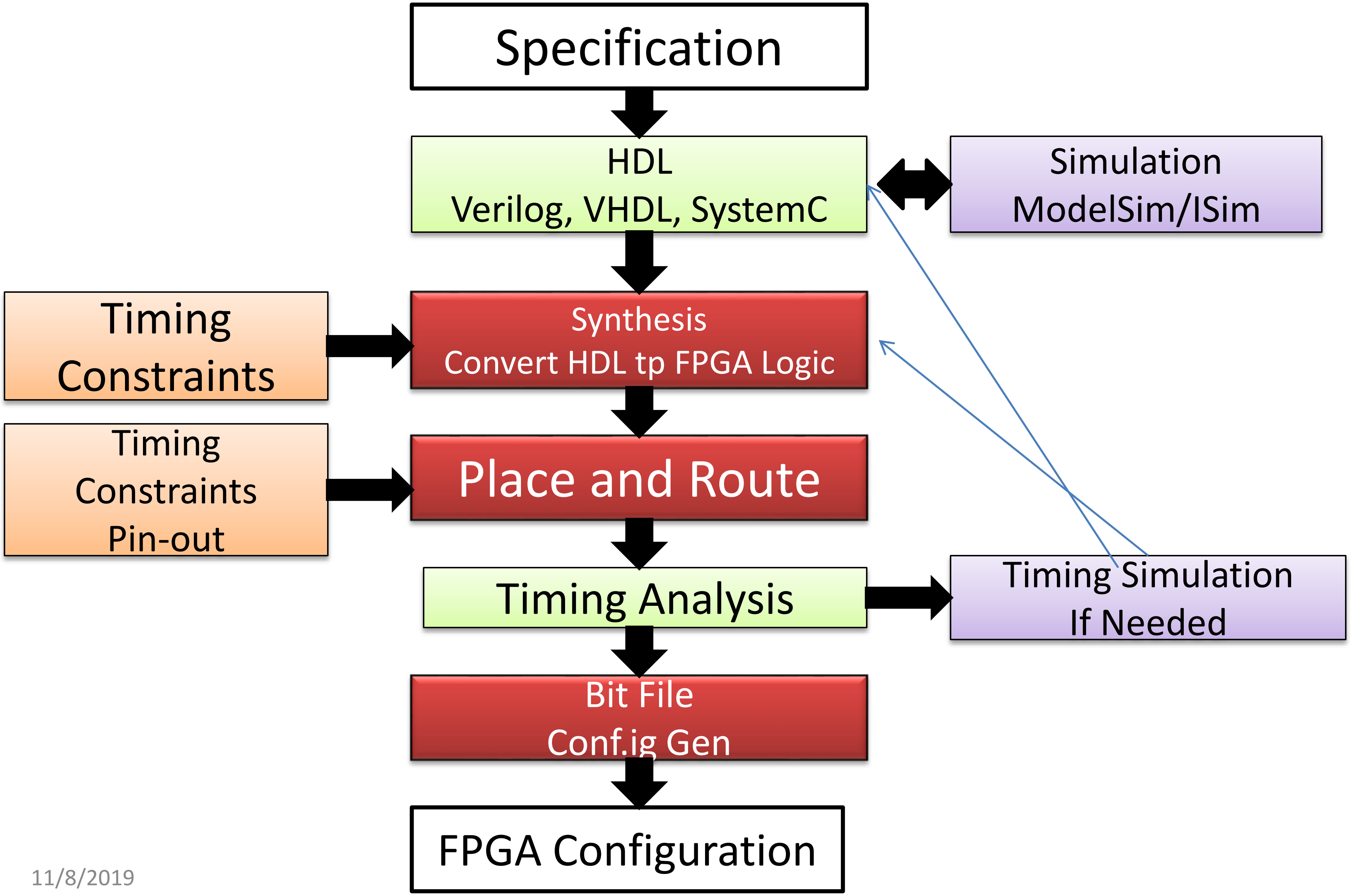
```
Cycle 1:  T1 = A[0] + A[1]
          T2 = A[2] + A[3]
Cycle 2:  S = T1 + T2
```







FPGA Design flow


Specification
HDL
Verilog, VHDL, SystemC
Simulation
ModelSim/ISim
Timing
Constraints
Synthesis
Convert HDL tp FPGA Logic
Timing
Constraints
Pin-out
Place and Route
Timing Analysis
Timing Simulation
If Needed
Bit File
Conf.ig Gen
FPGA Configuration