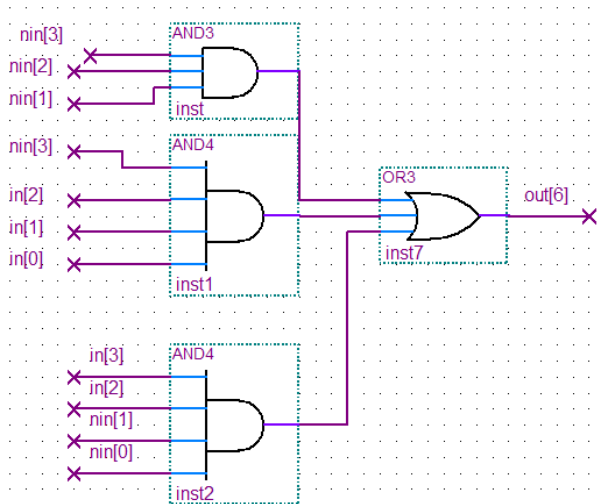# Imperial College London

Lecture 7

Digital Design with FPGA

Peter Cheung
Imperial College London

Course webpage: www.ee.ic.ac.uk/pcheung/teaching/EE2_CAS/
E-mail: p.cheung@imperial.ac.uk

How to describe/specify digital circuits?

Schematic diagram & gates

Boolean equation

$$out6 = /in3*/in2*/in1 + in3*in2*/in1*/in0 + /in3*in2*in1*in0$$

Truth table

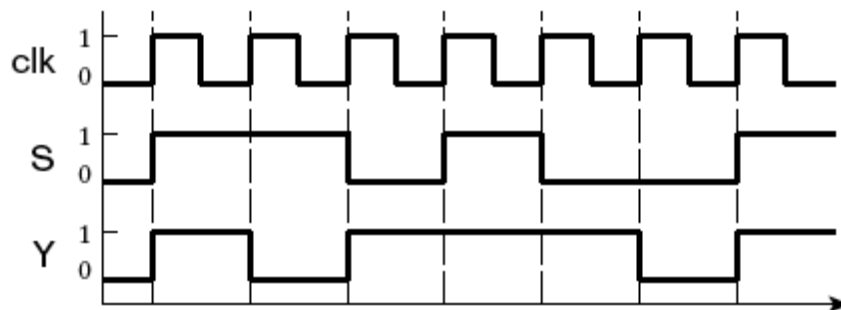| in[3..0]	out[6:0]
0000	1000000
0001	1111001
0010	0100100
0011	0110000
0100	0011001
0101	0010010
0110	0000010
0111	1111000
1000	0000000
1001	0010000

Hardware Description Language (HDL)

```
3-to-1 MUX

module mux32three(i0,i1,i2,sel,out);
input [31:0] i0,i1,i2;
input [1:0] sel;
output [31:0] out;
reg [31:0] out;

always @ (i0 or i1 or i2 or sel)
begin
  case (sel)
    2'b00: out = i0;
    2'b01: out = i1;
    2'b10: out = i2;
    default: out = 32'bx;
  endcase
end
endmodule
```
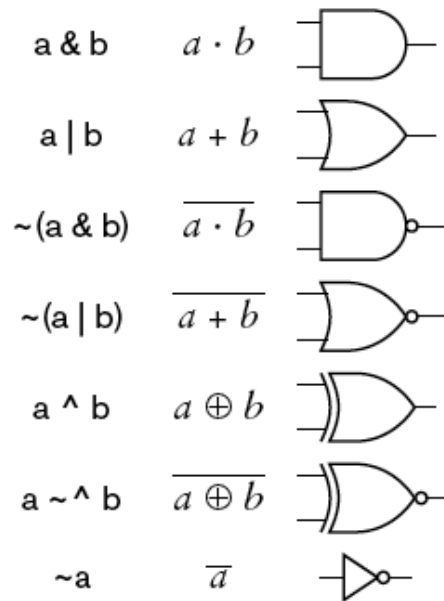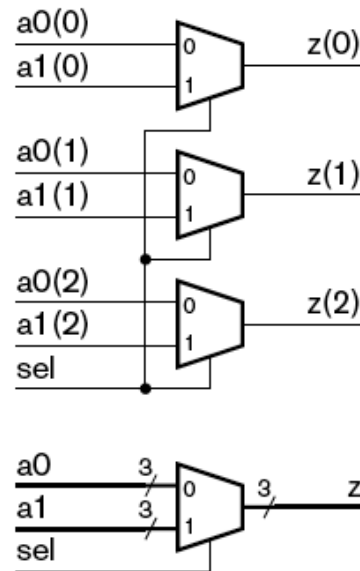
Timing Diagram
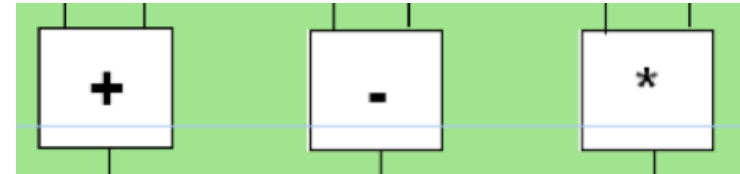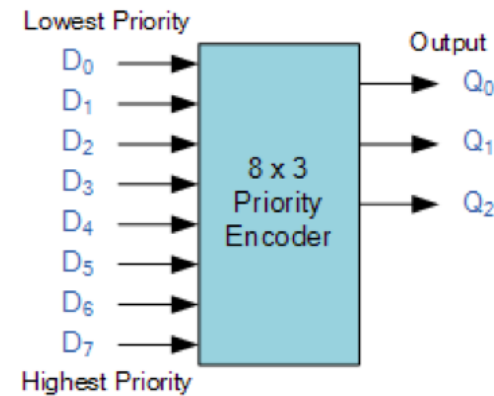
Basic digital building blocks

Cause & Effect

A ———[&]——▷— X

(B input)

Input B going high causes X to go low

Input A going low causes X to go high

> **Propagation Delay:**
> The time delay between a cause (an input changing) and its effect (an output changing), assuming output load capacitance of 30pF.

Example: 74AC00: Advanced CMOS 2-input NAND gate

	min	typ	max	
A↑ to X↓ (t_{PHL})	1.5	4.5	6.5	ns
A↓ to X↑ (t_{PLH})	1.5	6.0	8.0	ns

t_{PHL} and t_{PLH} refer to the direction that the output changes: high-to-low or low-to-high.

D-Flipflop (1)

Notation:

> \>   input effect happens on the rising edge

> C1   C $\Rightarrow$ Clock input, 1 $\Rightarrow$ This input _is_ input number 1.
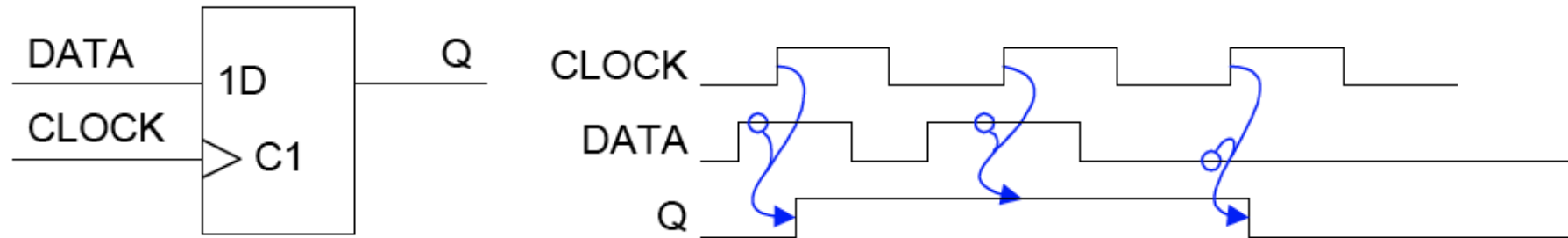
> 1D   D $\Rightarrow$ Data input,
>   1 $\Rightarrow$ This input _is controlled by_ input number 1.

The meaning of a number depends on its position:

     A number _after_ a letter is used to identify a particular input.
     A number _before_ a letter means that this input is controlled
     by one of the other inputs.

D-Flipflop (2)
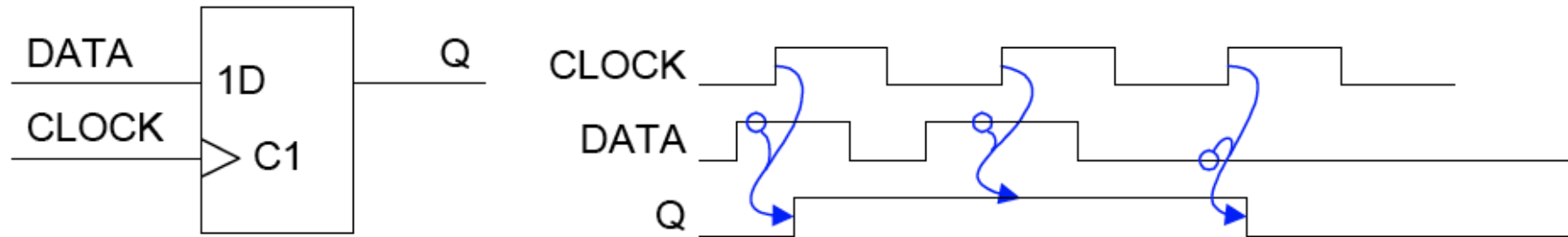
Cause and Effect:

- CLOCK↑ causes Q to change after a short delay. This is the *only* time Q ever changes.

- The value of D *just before* CLOCK↑ is the new Q.

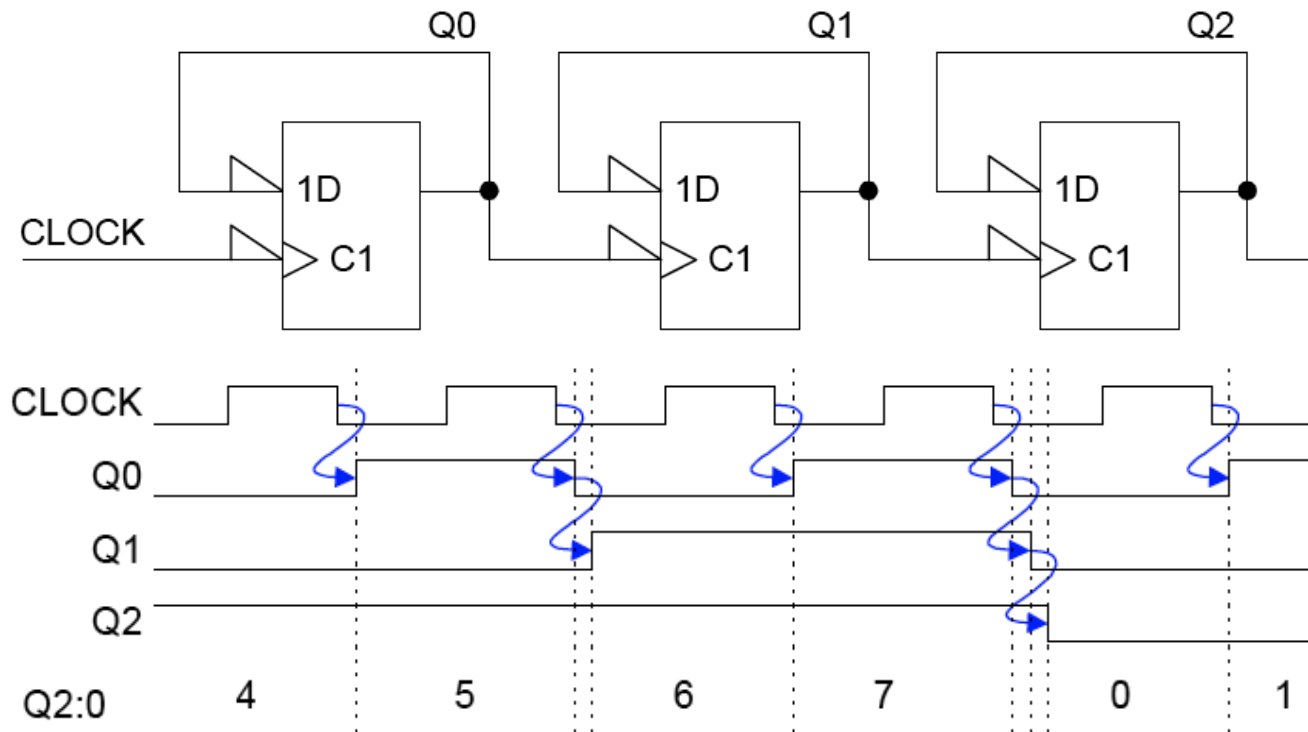- Propagation delay CLOCK↑ to Q is typically 1 ns.

- Propagation delay DATA to Q **does not make sense** since DATA changing does not cause Q to change.
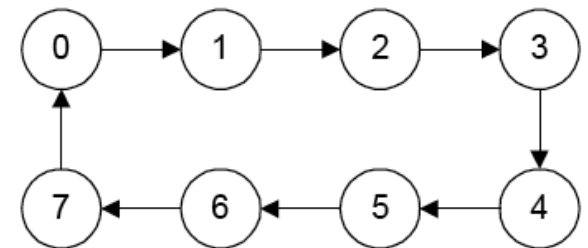
Ripple Counter

- Notice inverters on the CLOCK and DATA inputs
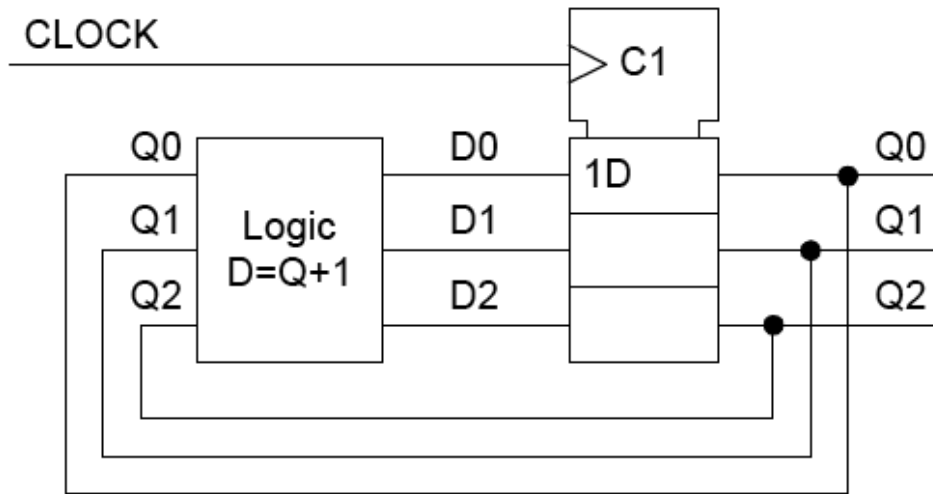- Least significant bit of a number is always labelled 0

Propagation Delay: CLOCK⬇ to Q2 = 3 x 1ns = 3ns

State Diagram
(not including transient states):

Synchronous Counter

The logic block must add 1 onto the current value of the counter, Q, to generate the next value of the counter, D. Suppose it has a propagation delay of 1ns.

- A *register* is a bunch of flipflops with the same CLOCK.

- The individual flipflops are rectangles stacked on top of each other. Only the top one is labelled.

- All shared signals (e.g. the CLOCK input) go to the notched *common control block* at the top of the stack.

Propagation Delays:
CLOCK↑ to Qi = 1ns
CLOCK↑ to Di = 2ns
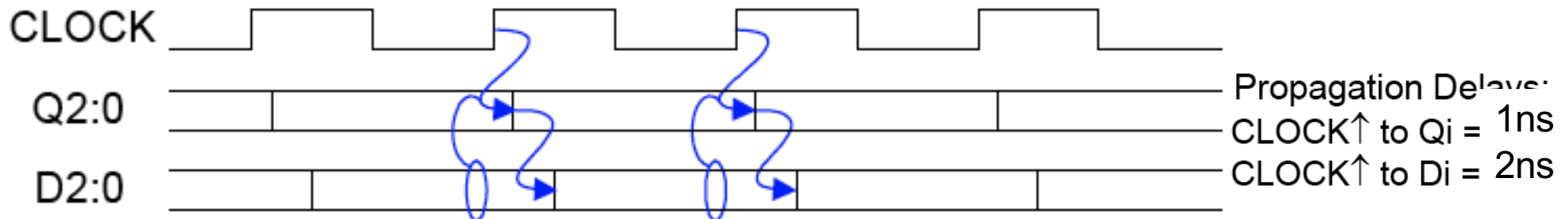
All flipflops change state within a fraction of nanosecond of each other.

Old ways of implementing digital circuits
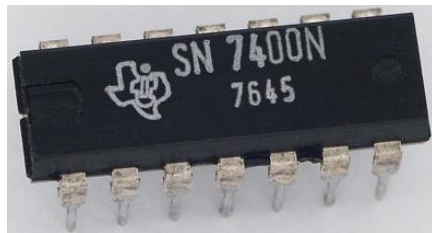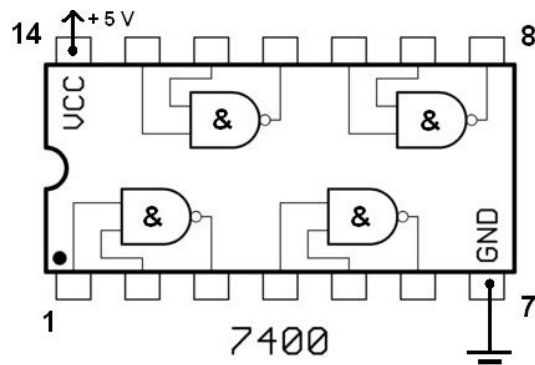
- ◆ Discrete logic – based on gates or small packages containing small digital building blocks (at most a 1-bit adder)
- ◆ De Morgan's theorem – theoretically we only need 2-input NAND or NOR gates to build anything
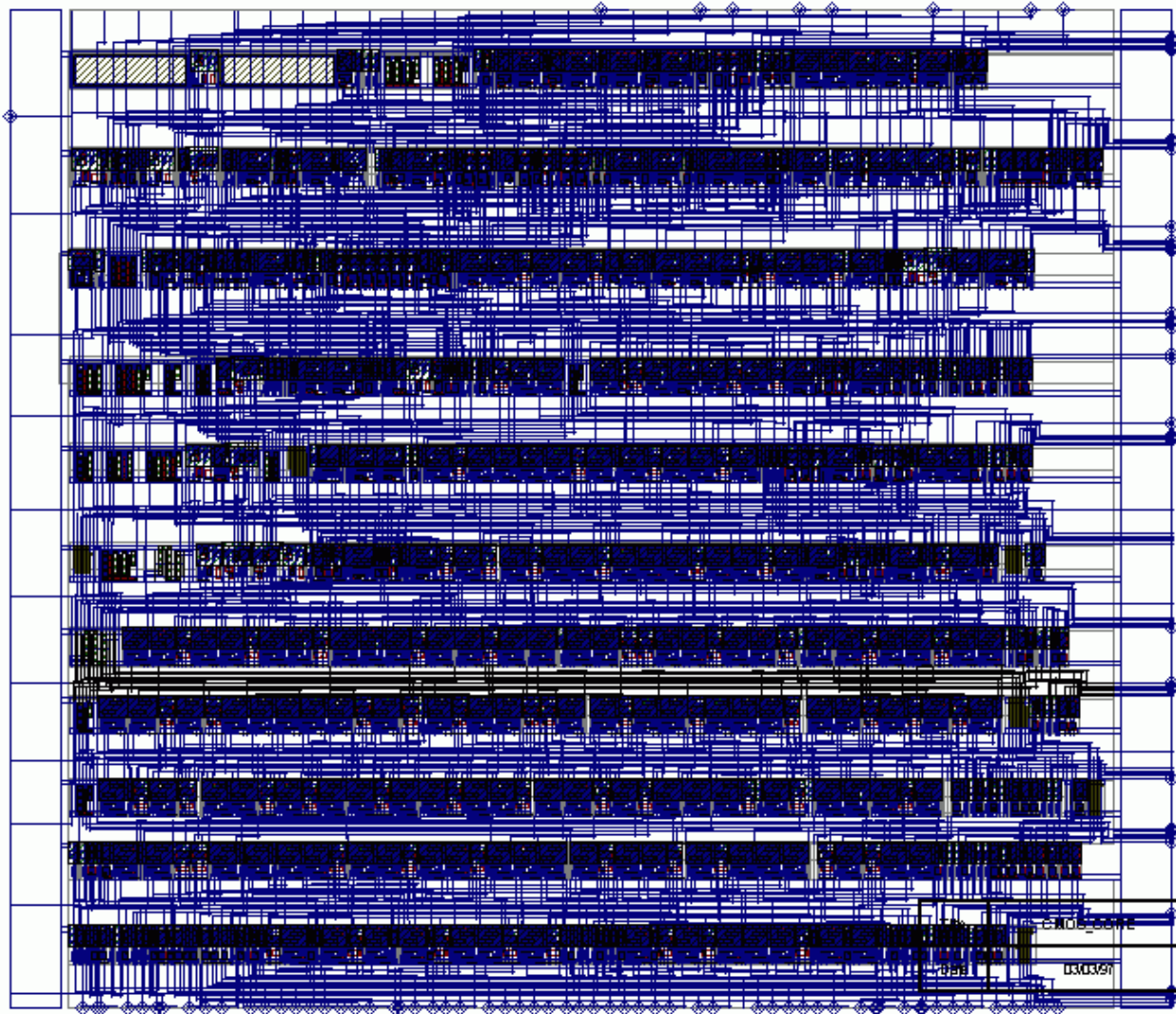- ◆ Tedious, expensive, slow, prone to wiring errors

Early integrated circuits based on gate arrays

- ◆ Rows of gates – often identical in structure
- ◆ Connected to form customer specific circuits
- ◆ Can be full-custom (i.e. completely fabricated from scratch for a given design)
- ◆ Can be semi-custom (i.e. customisation on the metal layers only)
- ◆ Once fabricated, the design is fixed

Modern digital design – full custom IC

- Intel Core i7
- > ¾ billion trans.
- Very expensive to design
- Very expensive to manufacture
- Not viable unless the market is very large
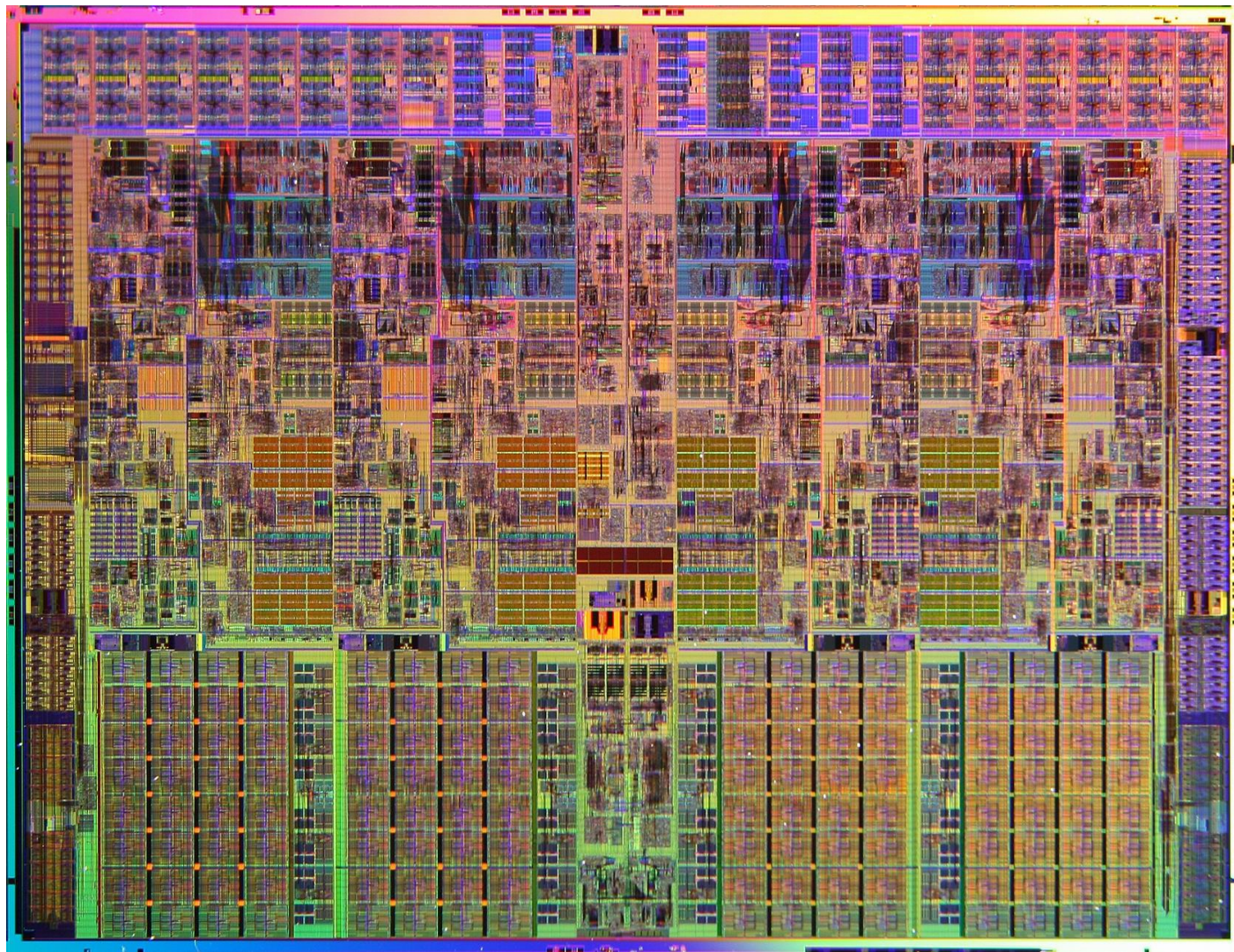
Field Programmable Gate Arrays (FPGAs)

- Combining idea from Programmable Logic Devices (PLDs – Yr 1 Lecture 8) and gate arrays
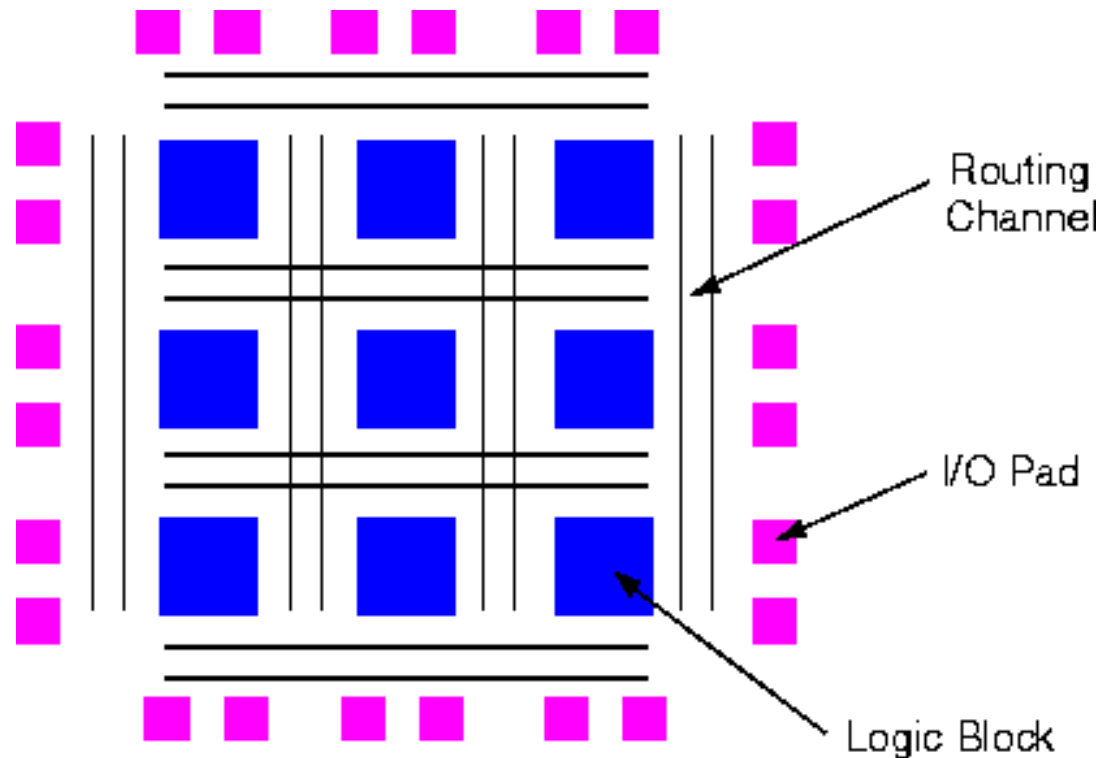
- First introduced by Xilinx in 1985

- Arrays of logic blocks (to implement logic functions)

- Lots of programmable wiring in routing channels

- Very flexible I/O interfacing logic core to outside world

- Two dominant FPGA makers:

  - Xilinx and Altera

- Other specialist makers e.g. Actel and Lattice Logic

Routing Channel

I/O Pad

Logic Block

Configurable Logic Block (or Logic Element)

◆ Based around Look-up Tables (LUTs), most common with 4-inputs

◆ Optional D-flipflop at the output of the LUT
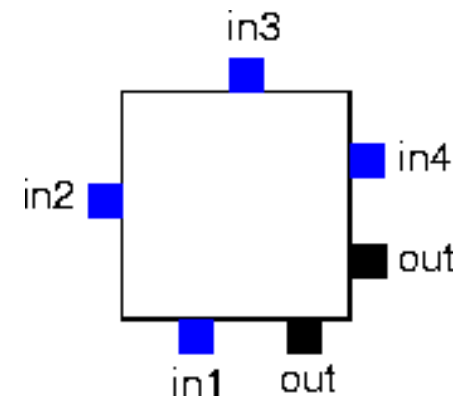
◆ 4-input LUT can implement ANY 4-input Boolean equation (truth-table)

◆ Special circuits for cascading logic blocks (e.g. carry-chain of a binary adder)
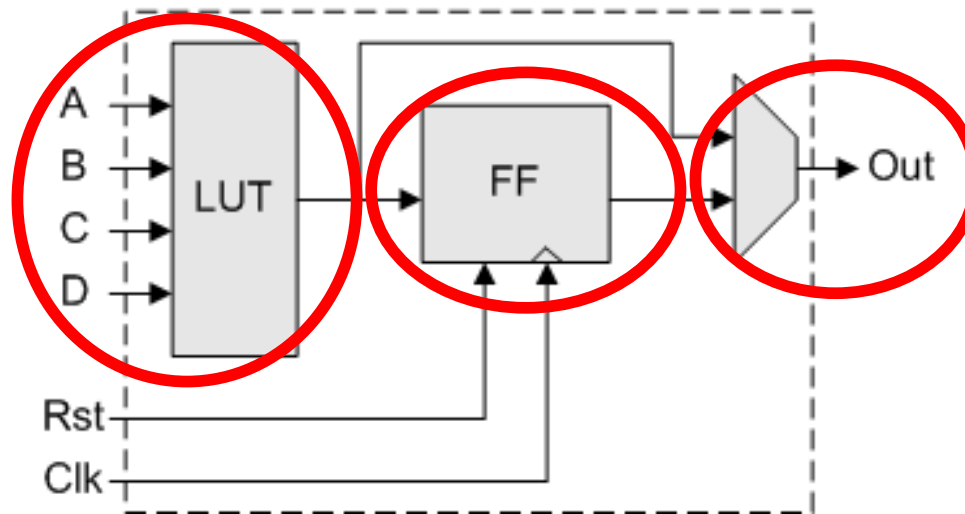
Logic Element (LE)

◆ Each logic block has pins located for easy access:

Programmable Routing

- ◆ Between rows and columns of logic blocks are wiring channels

- ◆ These are programmable – a logic block pin can be connected to one of many wiring tracks through a programmable switch

- ◆ Xilinx FPGAs have dedicated switch block circuits for routing (more flexible)

- ◆ Each wire segment can be connected in one of many ways:
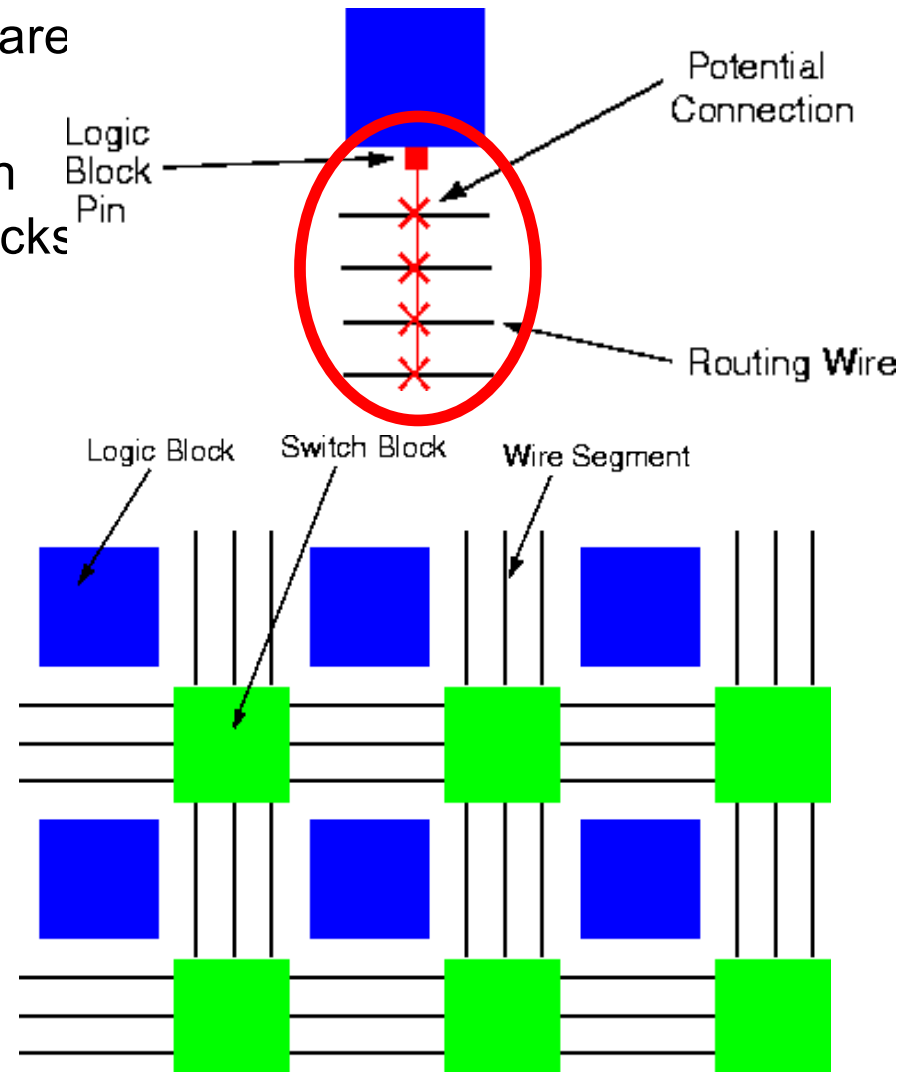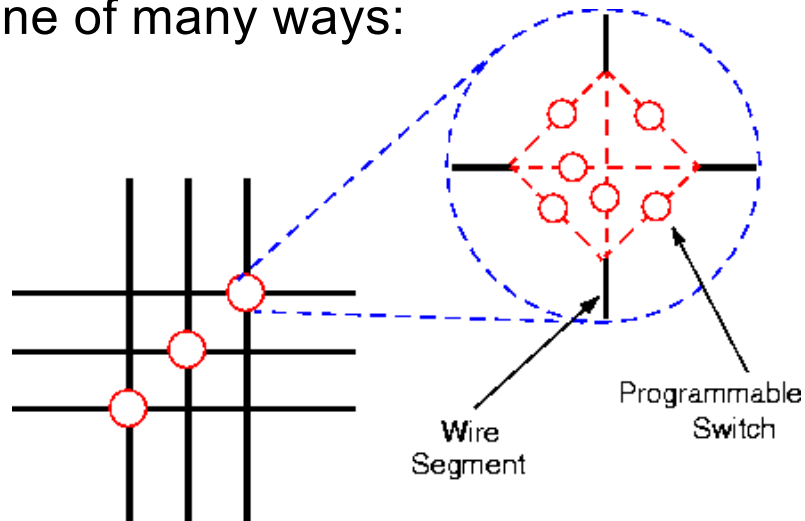
The Idea of Configuring the FPGA

◆ Programming an FPGA is NOT the same as programming a microprocessor

◆ We download a **BITSTREAM** (not a program) to an FPGA

◆ Programming an FPGA is known as **CONFIGURATION**

◆ All LUTs are configured using the BITSTREAM so that they contain the correct values to implement the Boolean logic

◆ Shown here is a typical implementation of a 4-LUT circuit

- ABCD are the FOUR inputs
- There are four levels of 2-to-1 multiplexer circuits
- The 16-inputs to the mux tree determine the Boolean function to be implemented as in a truth-table
- These 16 binary values are stored in registers (DFF)
- Configuration = setting the 16 registers to 1 or 0
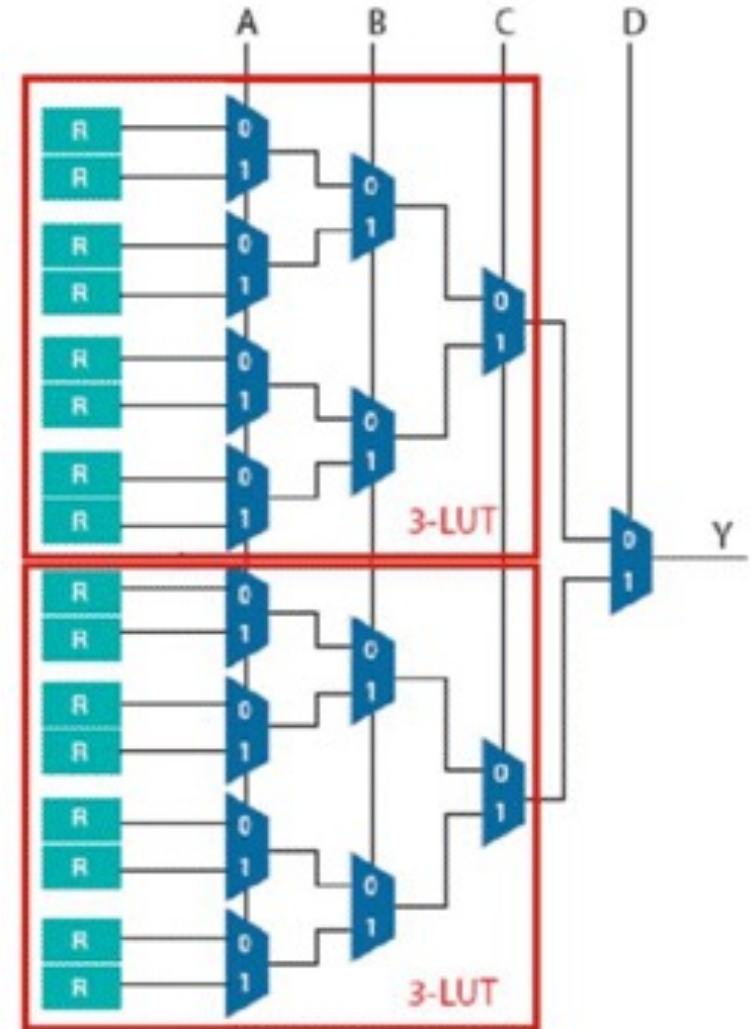
Configuring the routing in an FPGA

- At each interconnect site, there is a transistor switch which is default OFF (not conducting)

- Each switch is controlled by the output of a 1-bit configuration register

- Configuring the routing is simply to put a '1' or '0' in this register to control the routing switches

- Bitstream is either stored on local flash memory or download via a computer

- Configuration happens on power-up

Before Programming

After Programming

configuration control

READ or WRITE
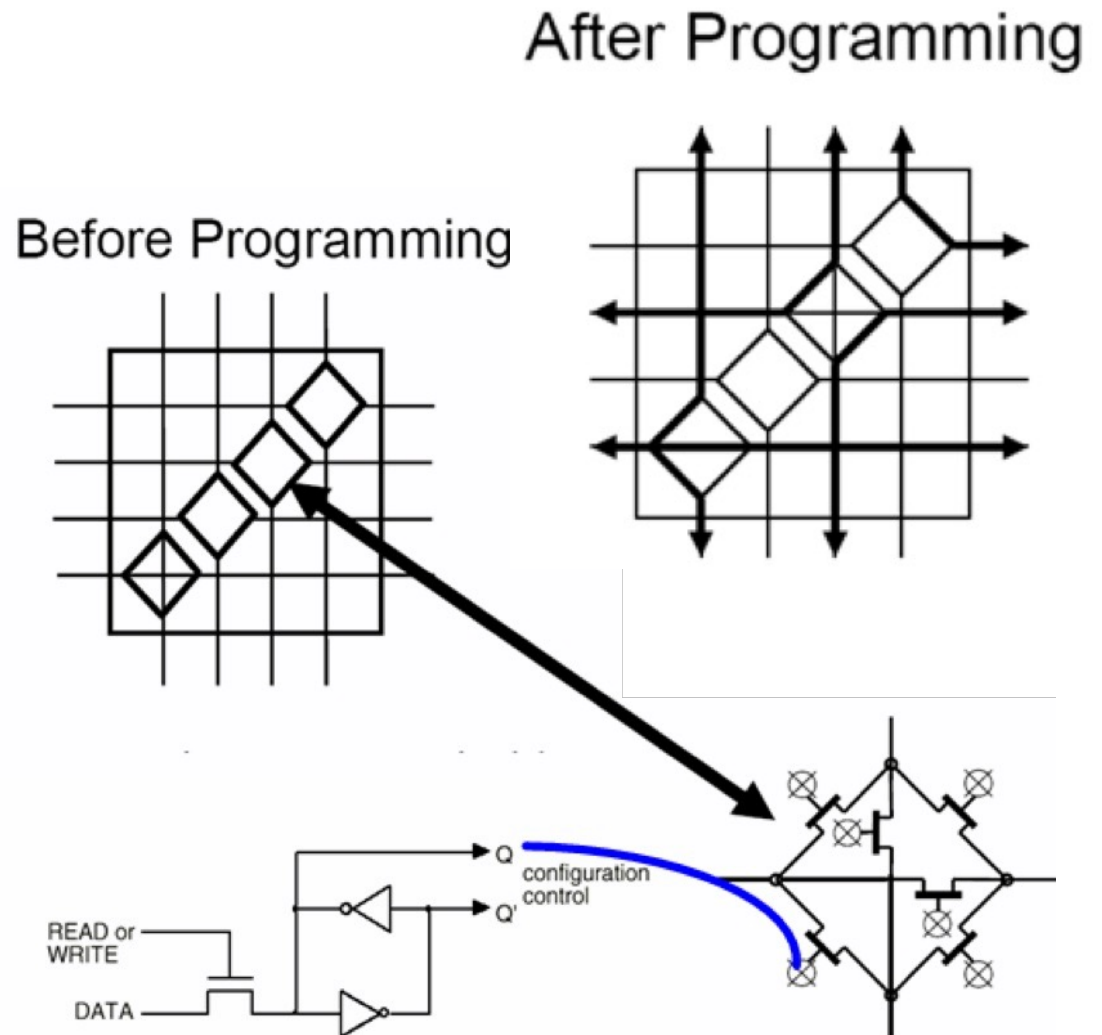
DATA

Max 10 FPGA – chip level view
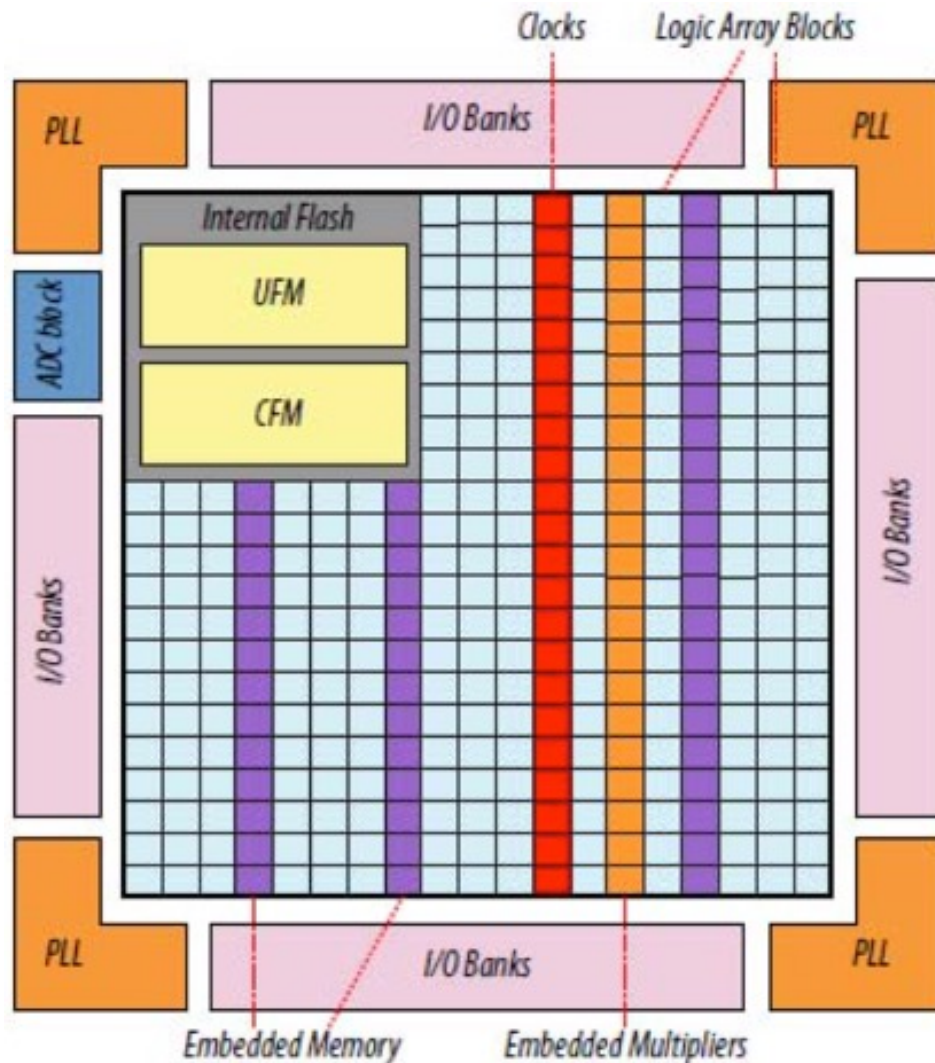
- ❖ Intel/Altera's Max 10 FPGA
- ❖ Low cost family

- ❖ Inside FPGA, it has:
  - Logic Array Blocks (LABs) containing Logic Elements (LEs)
  - Embedded Memory Blocks
  - Embedded Multiplier Blocks
  - Clock Blocks
  - Internal Flash (program/data)
  - Configurable I/O circuits
  - Phase-locked Loops (PLL)
  - Analog-to-Digital converter Block

Max 10 FPGA Logic Element
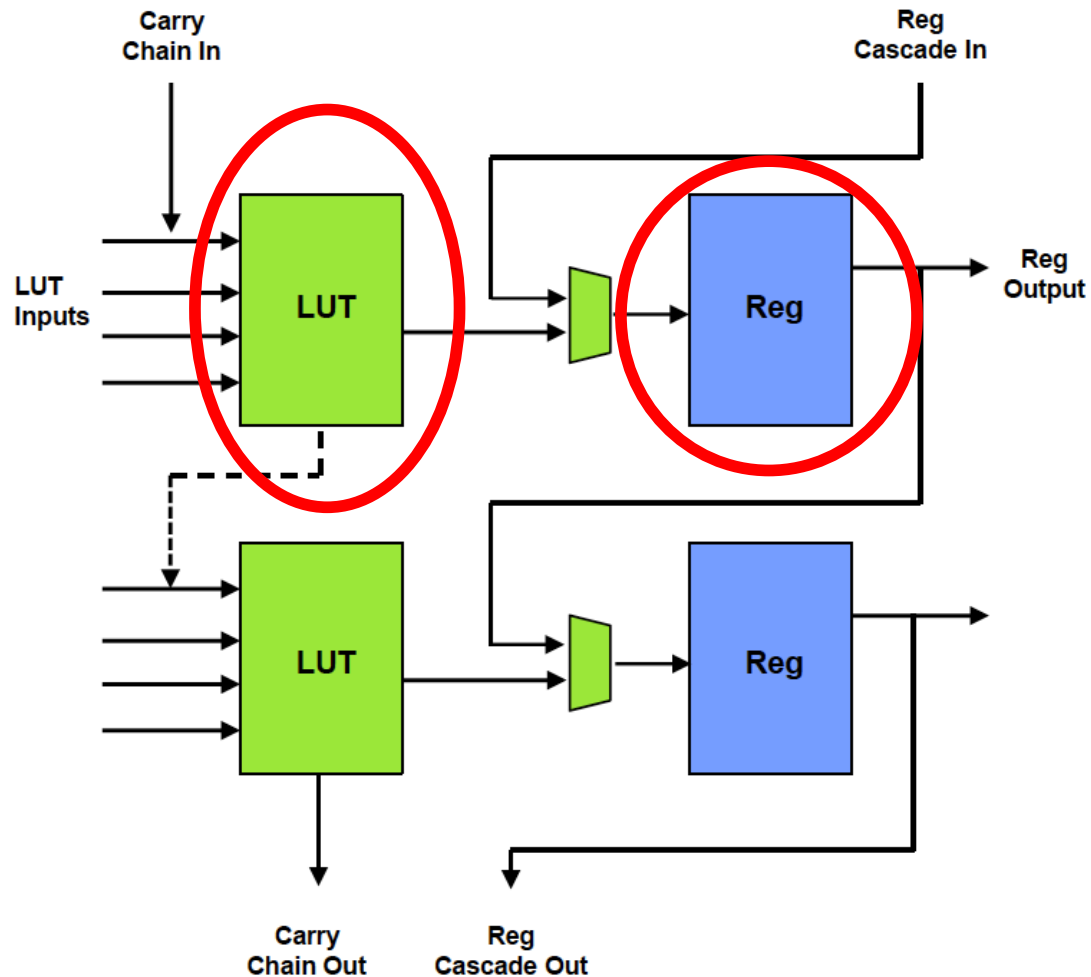
- ❖ Intel/Altera's Max 10 FPGA
- ❖ Device is **10M50DAF484C7G**
- ❖ 50,000 Logic Elements (LEs)
- ❖ Classical 4-LUT with D-FF

- ❖ Two dedicated paths between LEs:
  - Carry chain
  - Register cascade chain

Max 10 Logic Array Block (LAB)
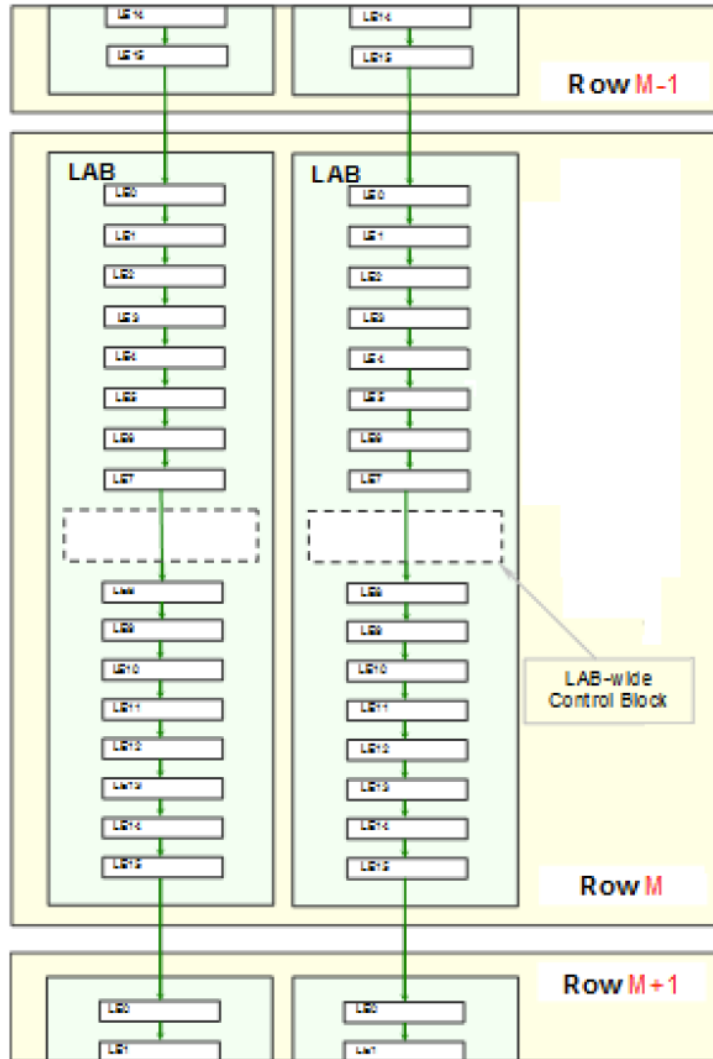
- ❖ Logic Element (LE)
  - 4 LUT
  - Register
  - Output routing logic

- ❖ Logic Array Block
  - 16 Logic Elements
  - Local Lab Control
  - Routing inside LAB

Design Tools – Quartus Prime Lite
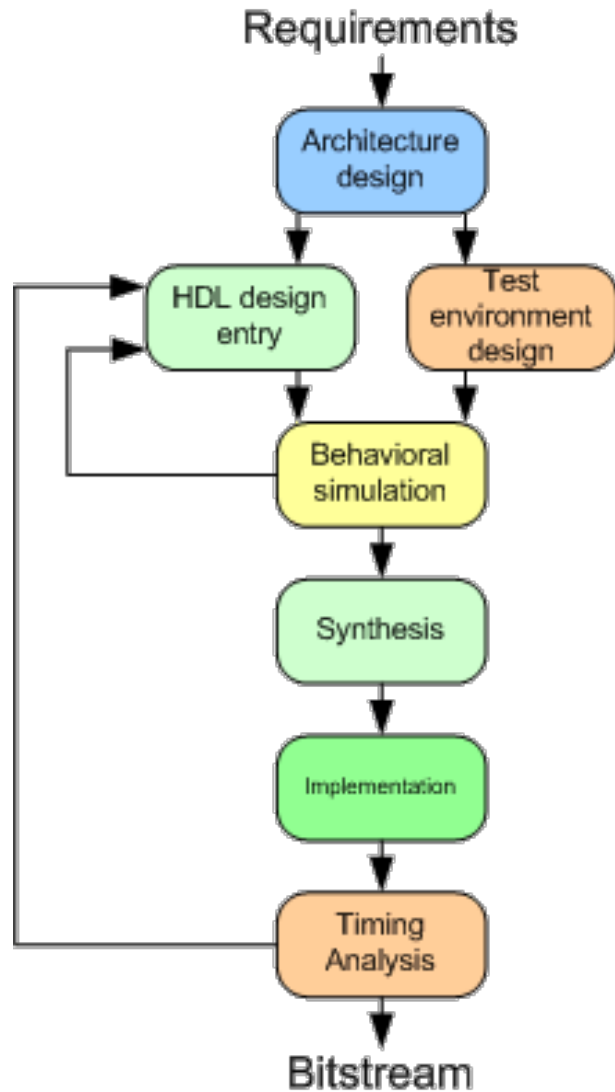
- ◆ Quartus Prime Lite – a comprehensive design tools for Intel/Altera FPGAs
- ◆ Special web edition free to download from (need registration):
  - https://fpgasoftware.intel.com/18.0/?edition=lite&platform=windows
  - Features include (see introduction to Quartus II):
    - design entry
    - compilation from Hardware Description Languages (HDL)
    - synthesis
    - simulation
    - timing analysis
    - power analysis
    - project management

DE10-Lite FPGA Board