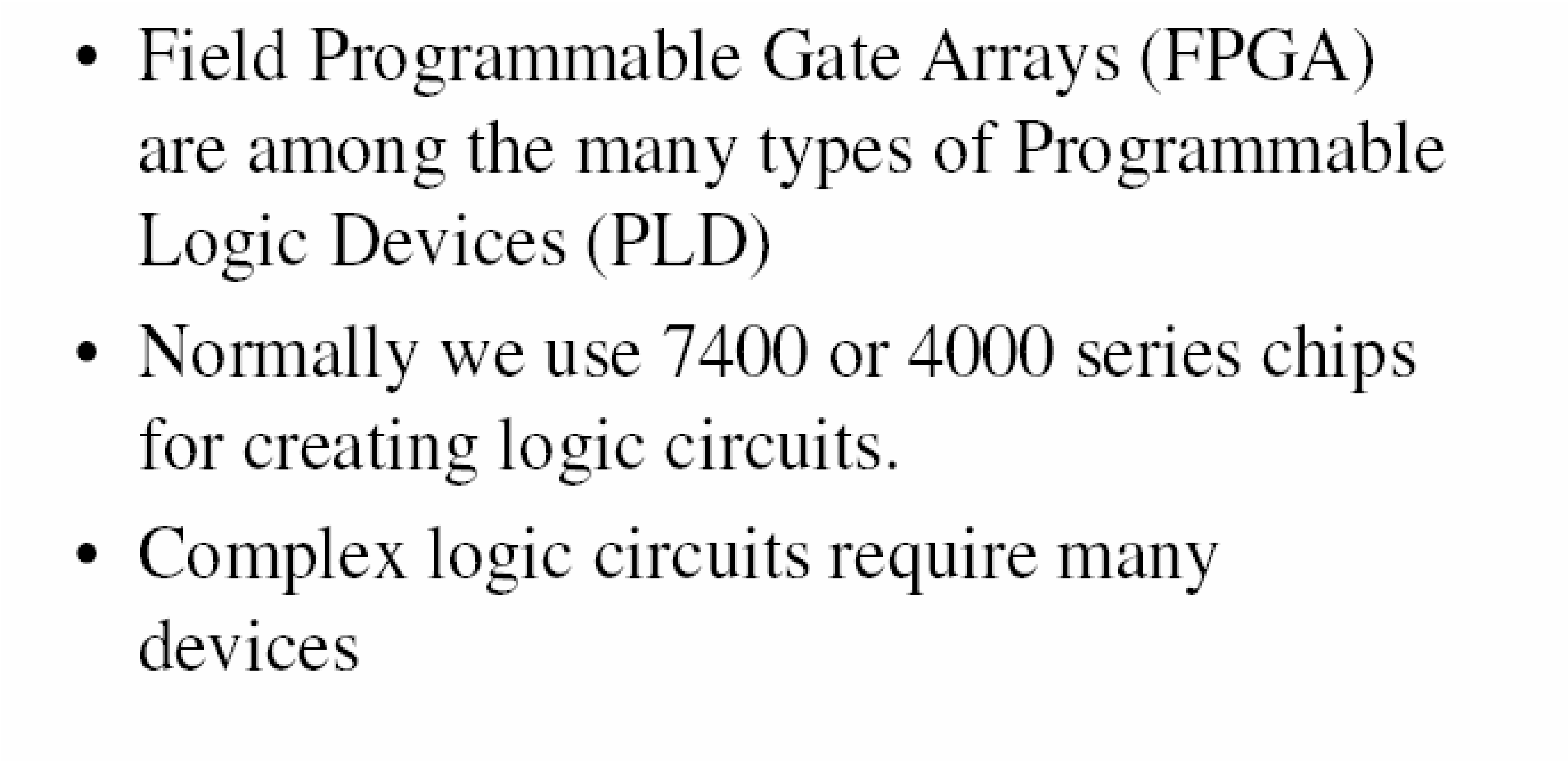
An Introduction to FPGA

Dr. Shubhajit Roy Chowdhury,

Centre for VLSI and Embedded Systems Technology,

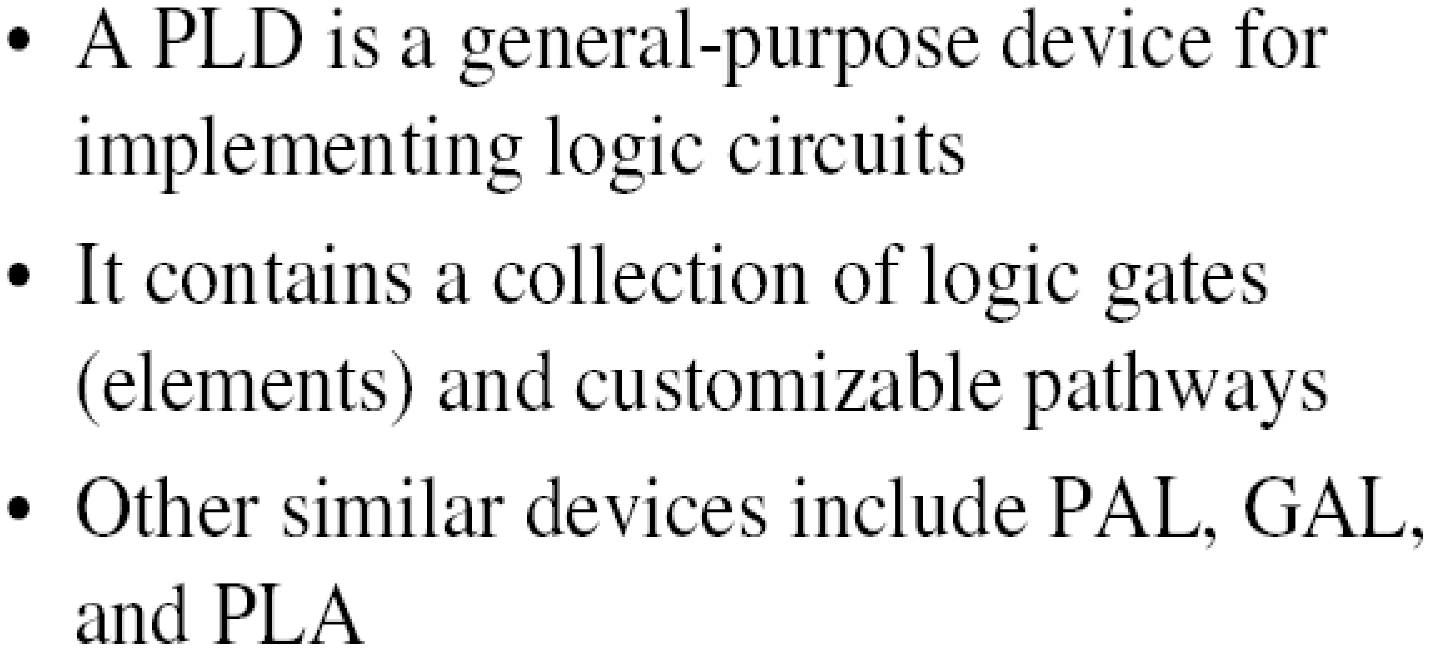
IIIT Hyderabad

Introduction



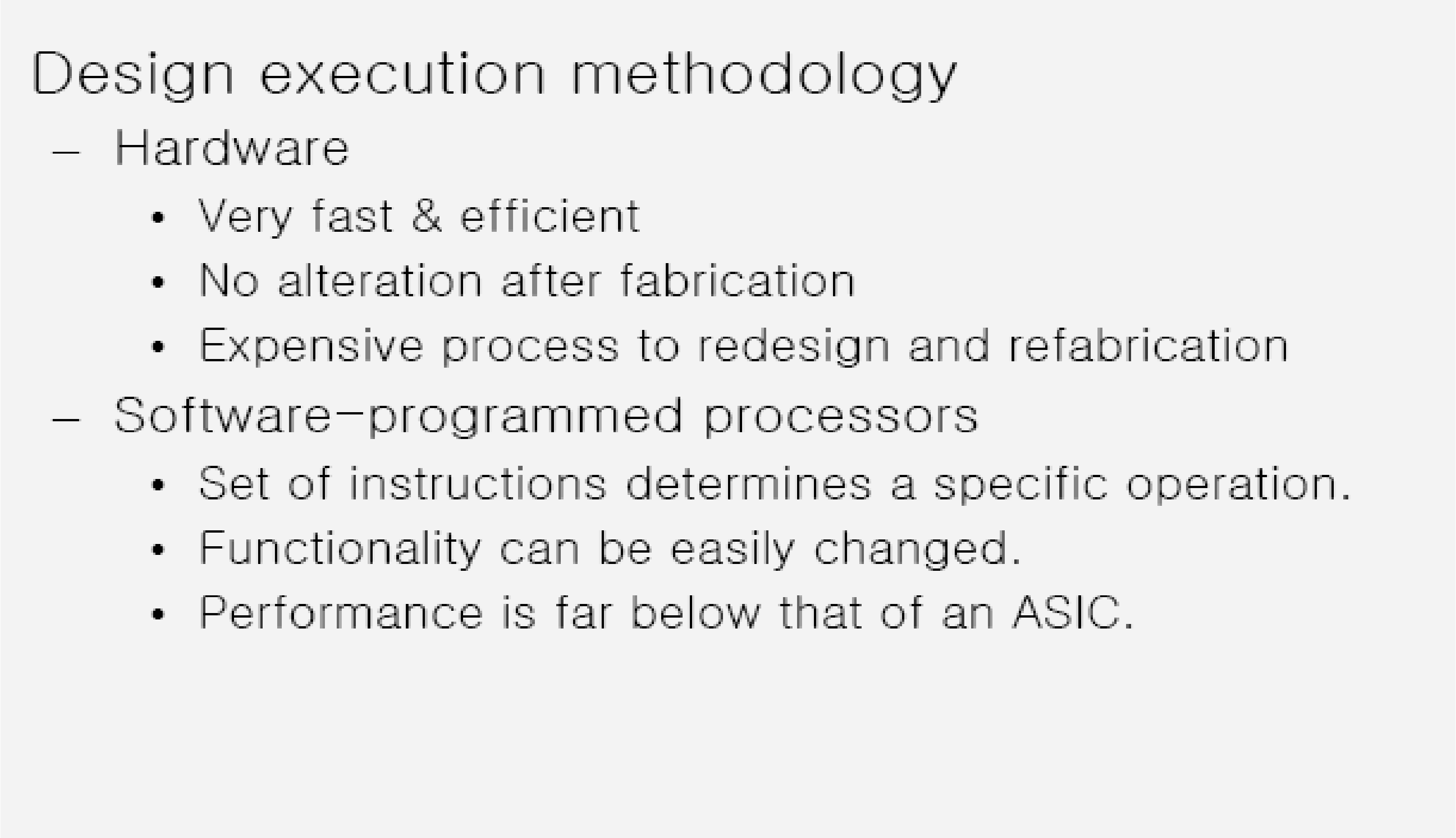
- Field Programmable Gate Arrays (FPGA) are among the many types of Programmable Logic Devices (PLD)
- Normally we use 7400 or 4000 series chips for creating logic circuits.
- Complex logic circuits require many devices

Programmable Logic Devices



- A PLD is a general-purpose device for implementing logic circuits
- It contains a collection of logic gates (elements) and customizable pathways
- Other similar devices include PAL, GAL, and PLA

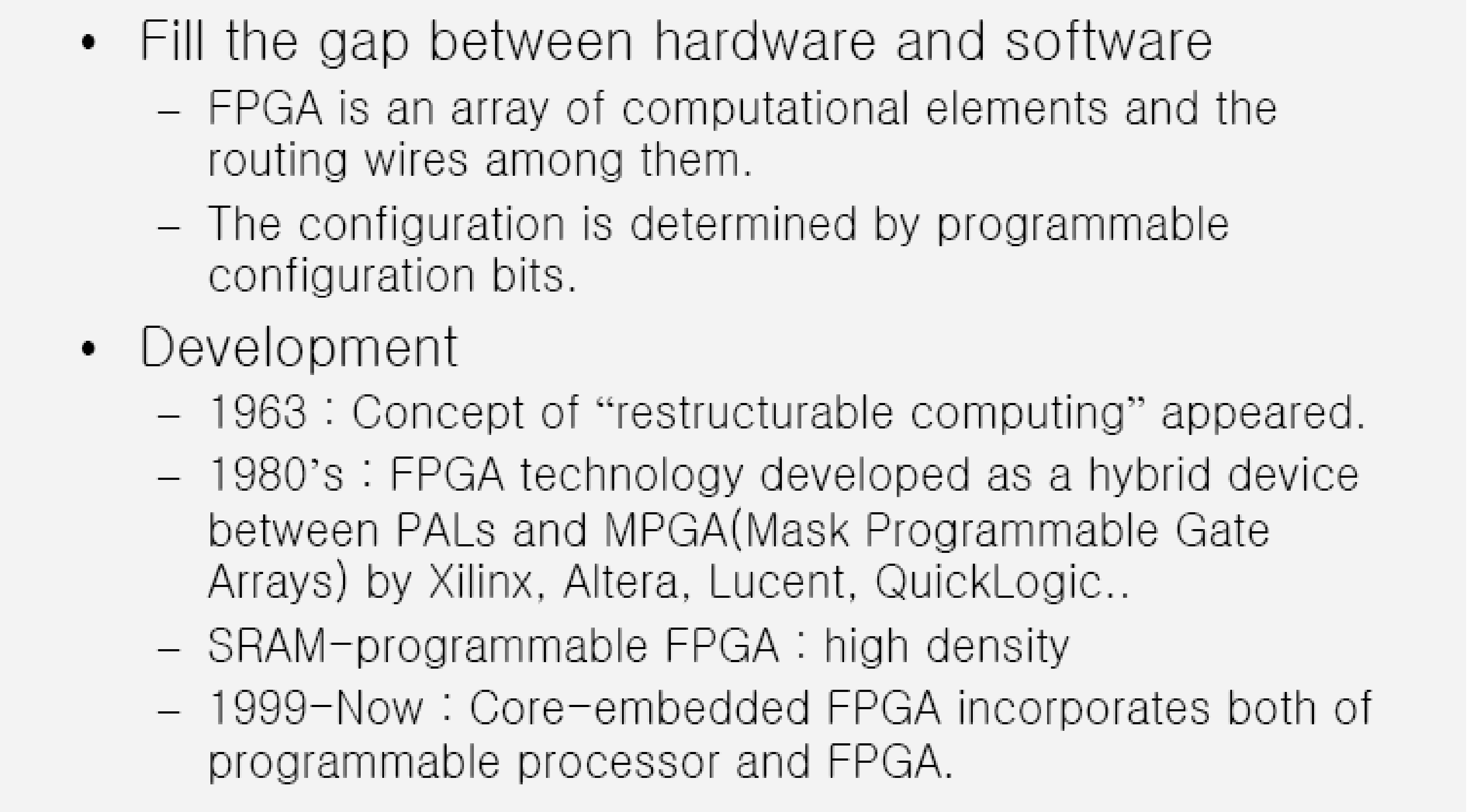
Choice of execution methodology



Design execution methodology

- Hardware
  - Very fast & efficient
  - No alteration after fabrication
  - Expensive process to redesign and refabrication
- Software-programmed processors
  - Set of instructions determines a specific operation.
  - Functionality can be easily changed.
  - Performance is far below that of an ASIC.

Reconfigurable Computing



- Fill the gap between hardware and software
  - FPGA is an array of computational elements and the routing wires among them.
  - The configuration is determined by programmable configuration bits.
- Development
  - 1963 : Concept of “restructurable computing” appeared.
  - 1980’s : FPGA technology developed as a hybrid device between PALs and MPGA(Mask Programmable Gate Arrays) by Xilinx, Altera, Lucent, QuickLogic..
  - SRAM-programmable FPGA : high density
  - 1999-Now : Core-embedded FPGA incorporates both of programmable processor and FPGA.

INTRODUCTION TO FPGA

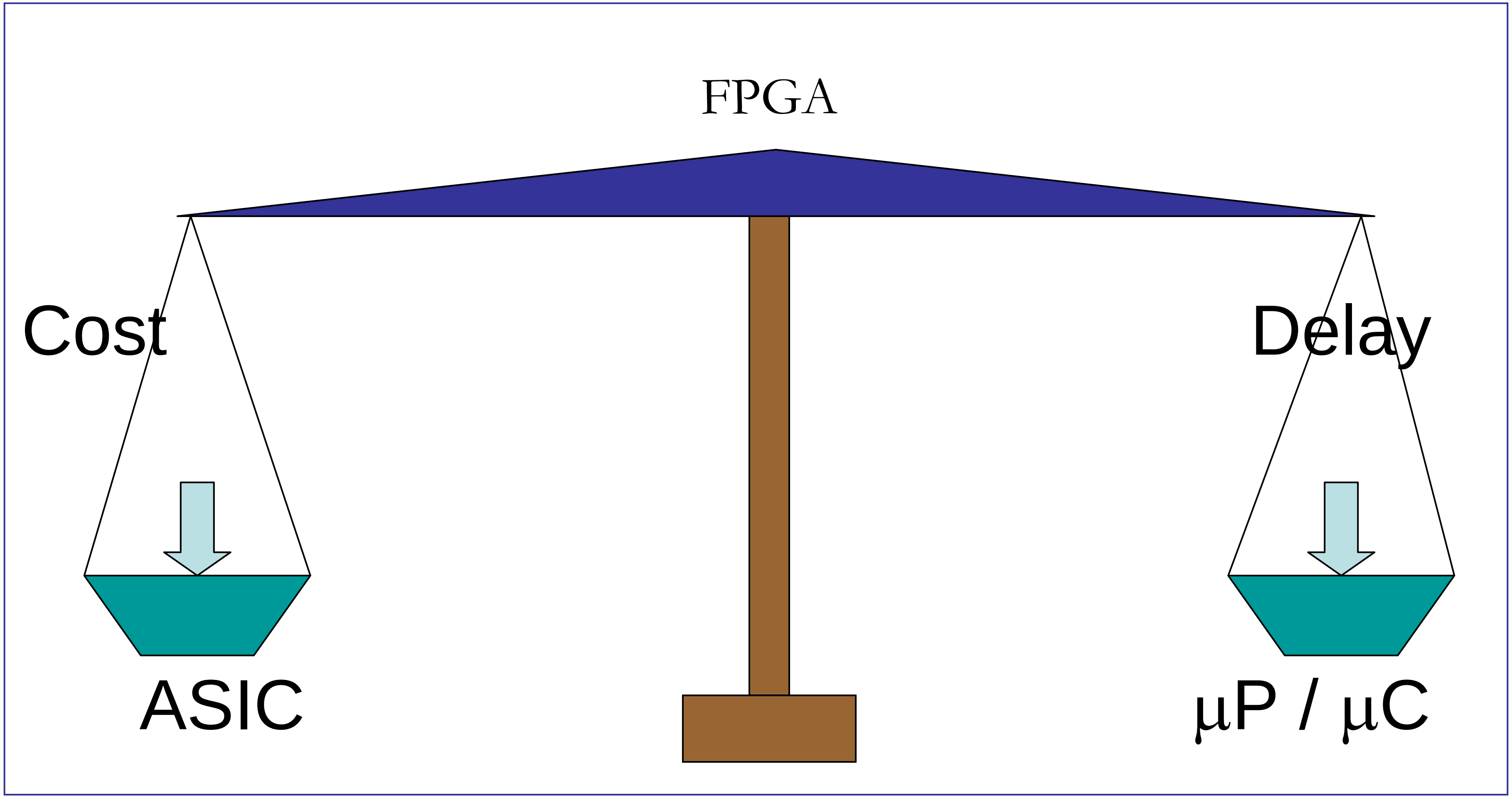
FPGA - *What it is?*

- Field Programmable Gate Array (FPGA) is a semiconductor device containing programmable logic components and programmable interconnects.

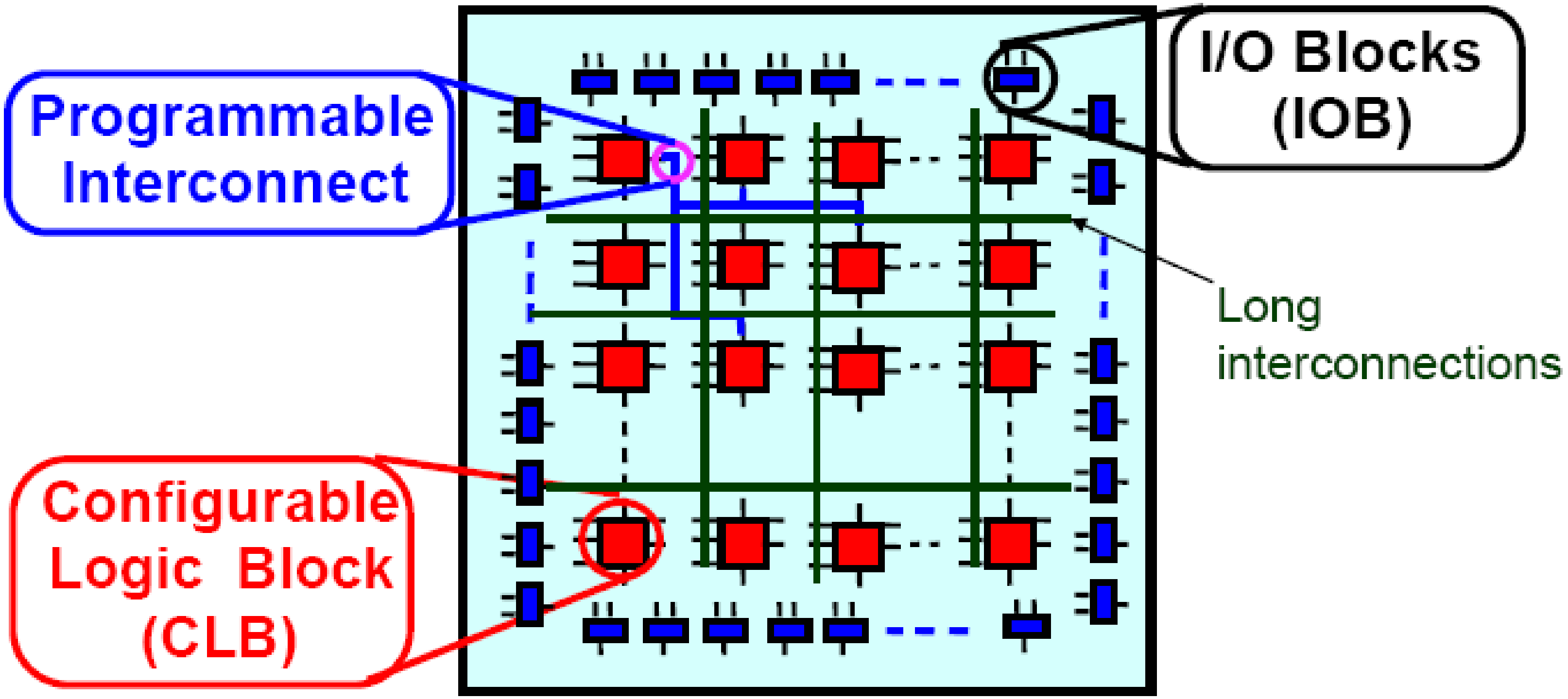
- The programmable logic components can be programmed to duplicate the functionality of basic logic gates (such as AND, OR, XOR, NOT) or more complex combinatorial functions such as decoders or simple math functions.

- In most FPGAs, these programmable logic components (or logic blocks) also include memory elements, which may be simple flip-flops or more complete blocks of memories.

FPGA – A good compromise

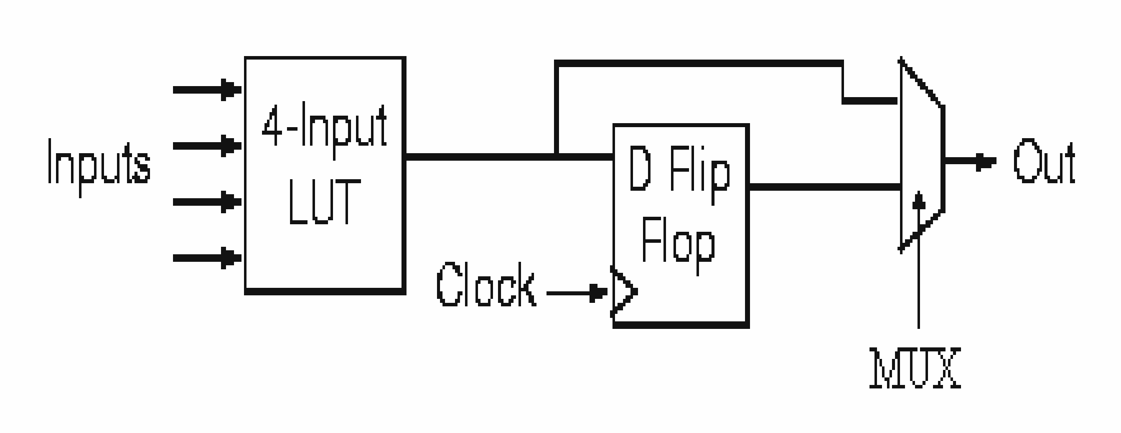


BASIC ARCHITECTURE OF FPGA

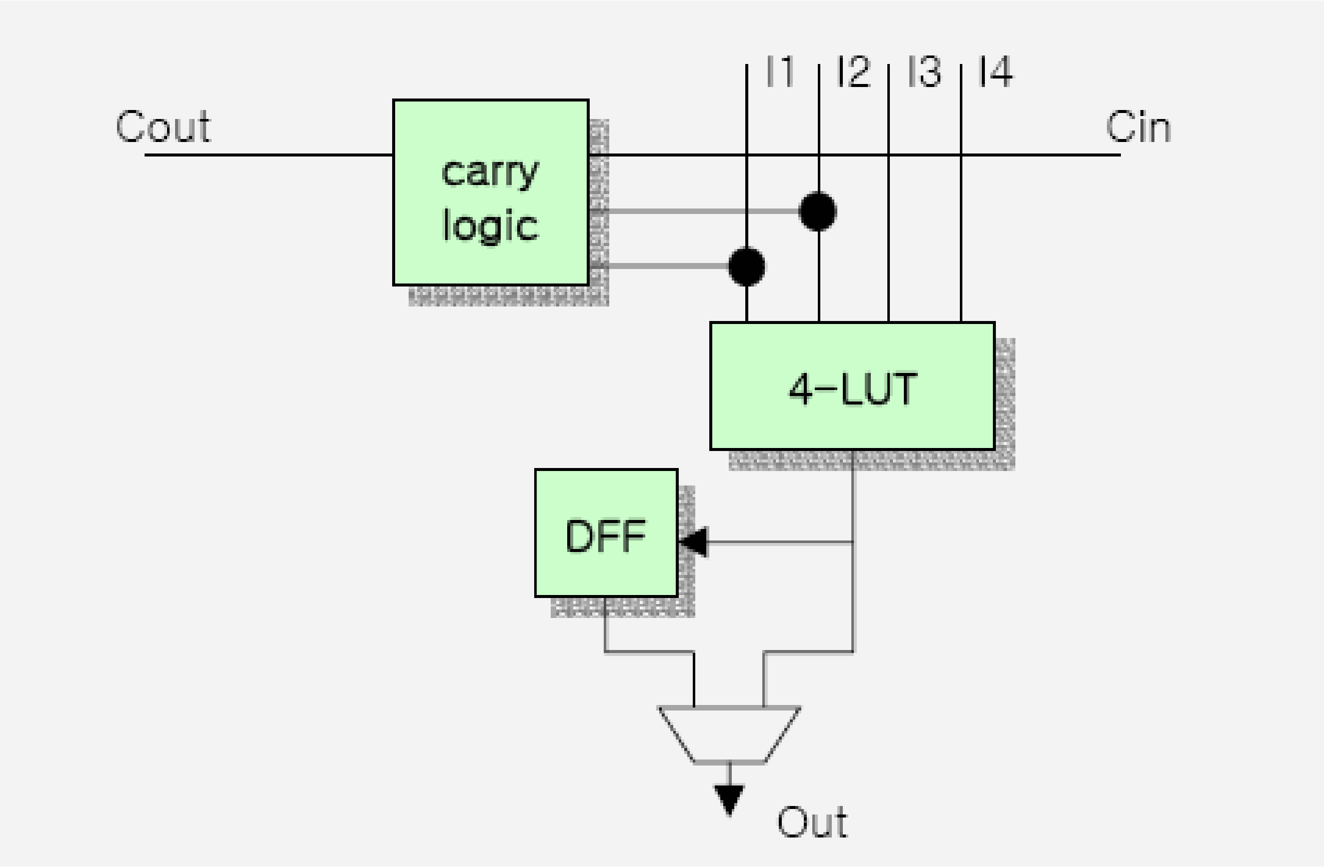


SRAMS cells throughout the FPGA determine the functionality of the device

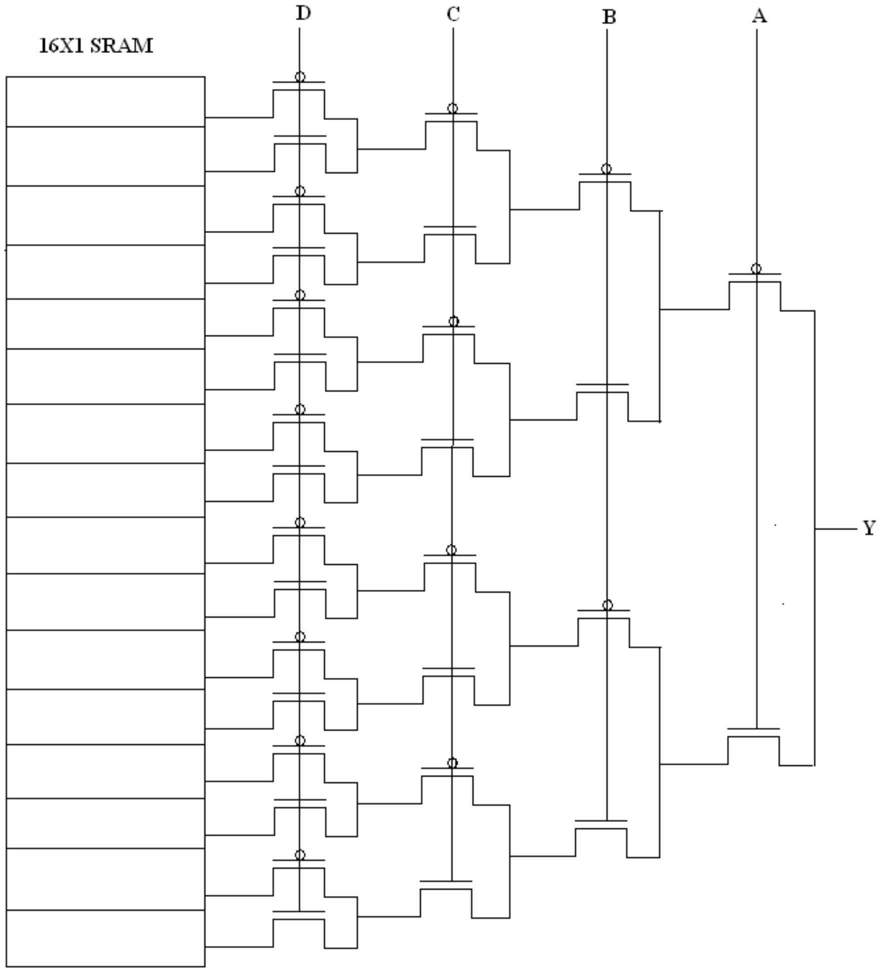
STRUCTURE OF A CONFIGURABLE LOGIC BLOCK



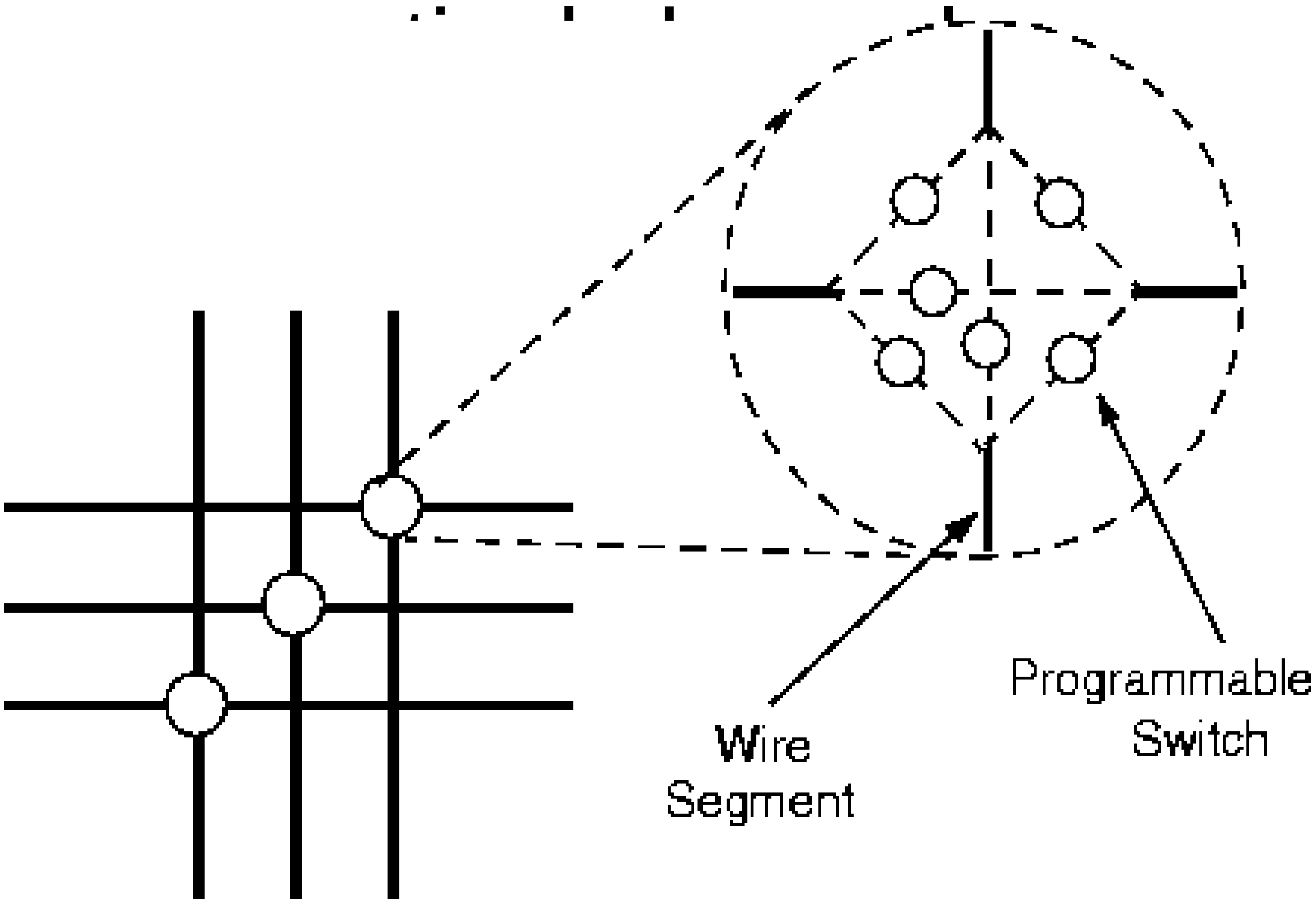
Structure of a Configurable Logic Block with Carry Logic



Structure of 4 input LUT



Switchbox topology at the intersection of horizontal and



Segmented Routing Architecture

•Local communication traffic by short wires

•Long wires are typically used to travel over long distances within the chip without passing through switches

•Research challenges:

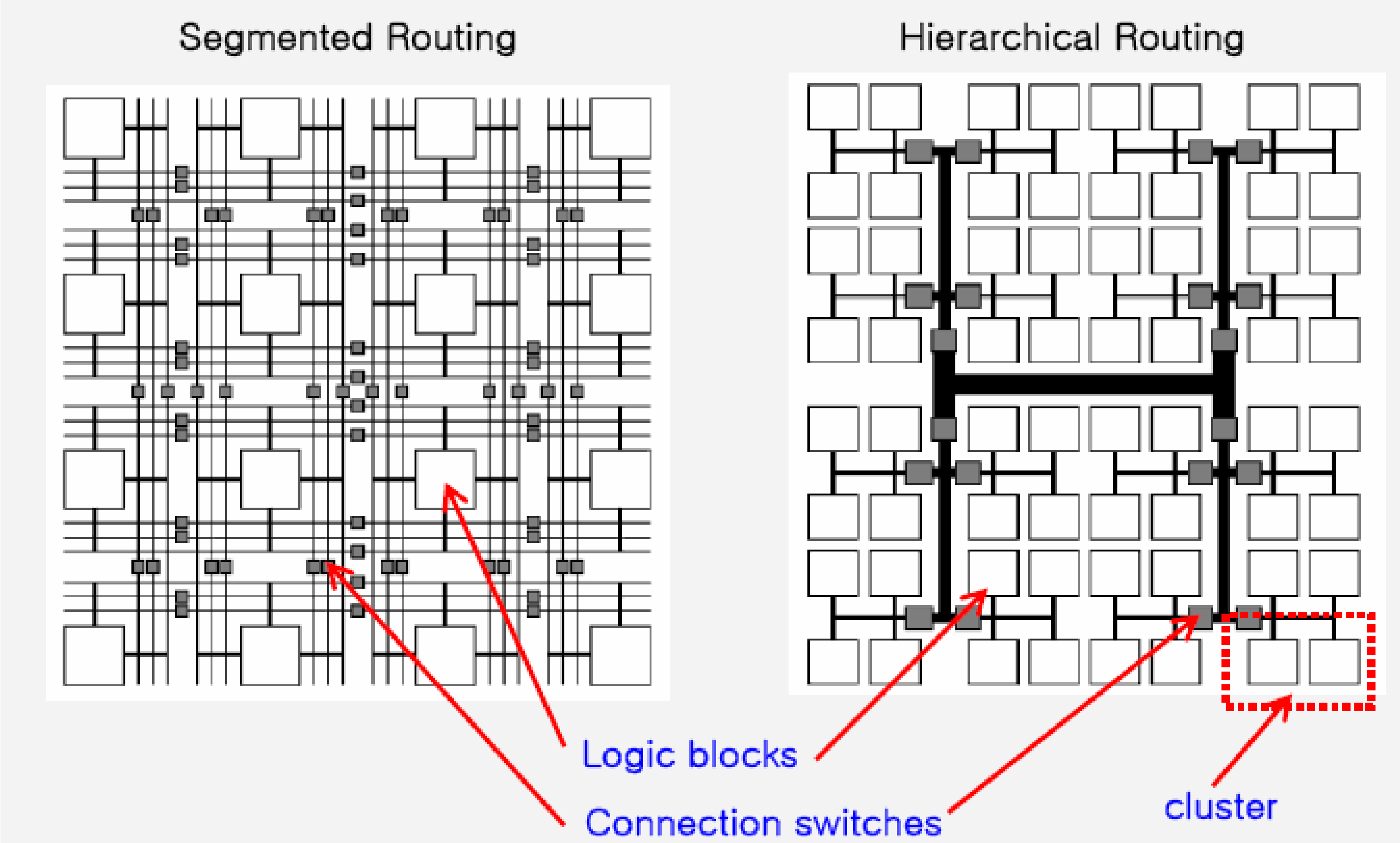
➢ How many wires should be contained in each channel?

➢How many local and global wires should be there in each channel?

Hierarchical Routing Architecture

- Cluster based routing architecture
- Routing within a cluster is only at the local level within the cluster
- Global wires connect the clusters together
- Most connection between the logic blocks is local with limited number of wires traversing over longer distances
- Intelligent design of placement algorithms needed

Two routing architectures



Segmented Routing

Hierarchical Routing

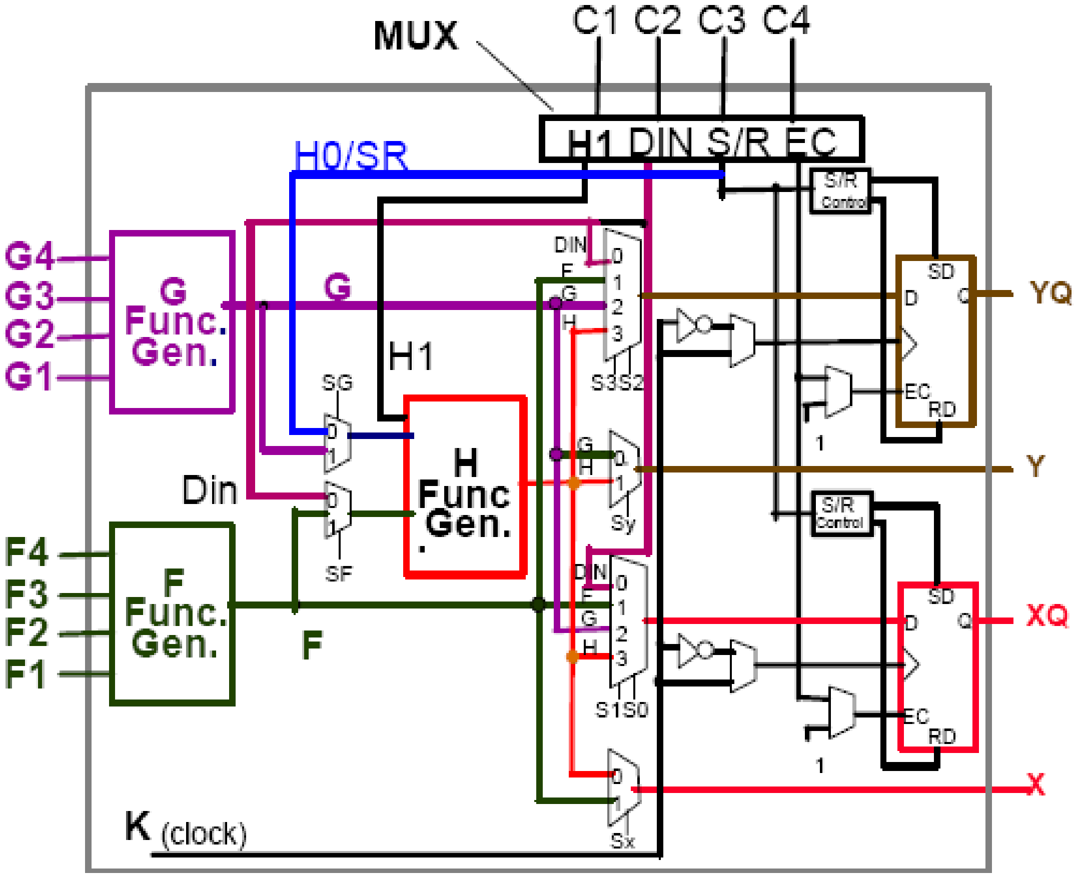
Logic blocks

Connection switches

cluster

Xilinx XC4000E CLB Architecture

- 2 Four-input function generators (Look Up Tables)
- 1 Three-input function
- 2 Registers:
  - Pos. or Neg. edge-trig. Synchronous and asynchr. Set/Reset
- Possible functions:
  - any fct of 5 var.
  - two fcts of 4 var. + one fct of 3 var.
  - some fct of 9 var.



An Example

Implement the following function function on a LUT based FPGA like XC 4000E

X=A’B’(C+D)

Y=AK+BK+C’D’K+AEJL

To start with the solution

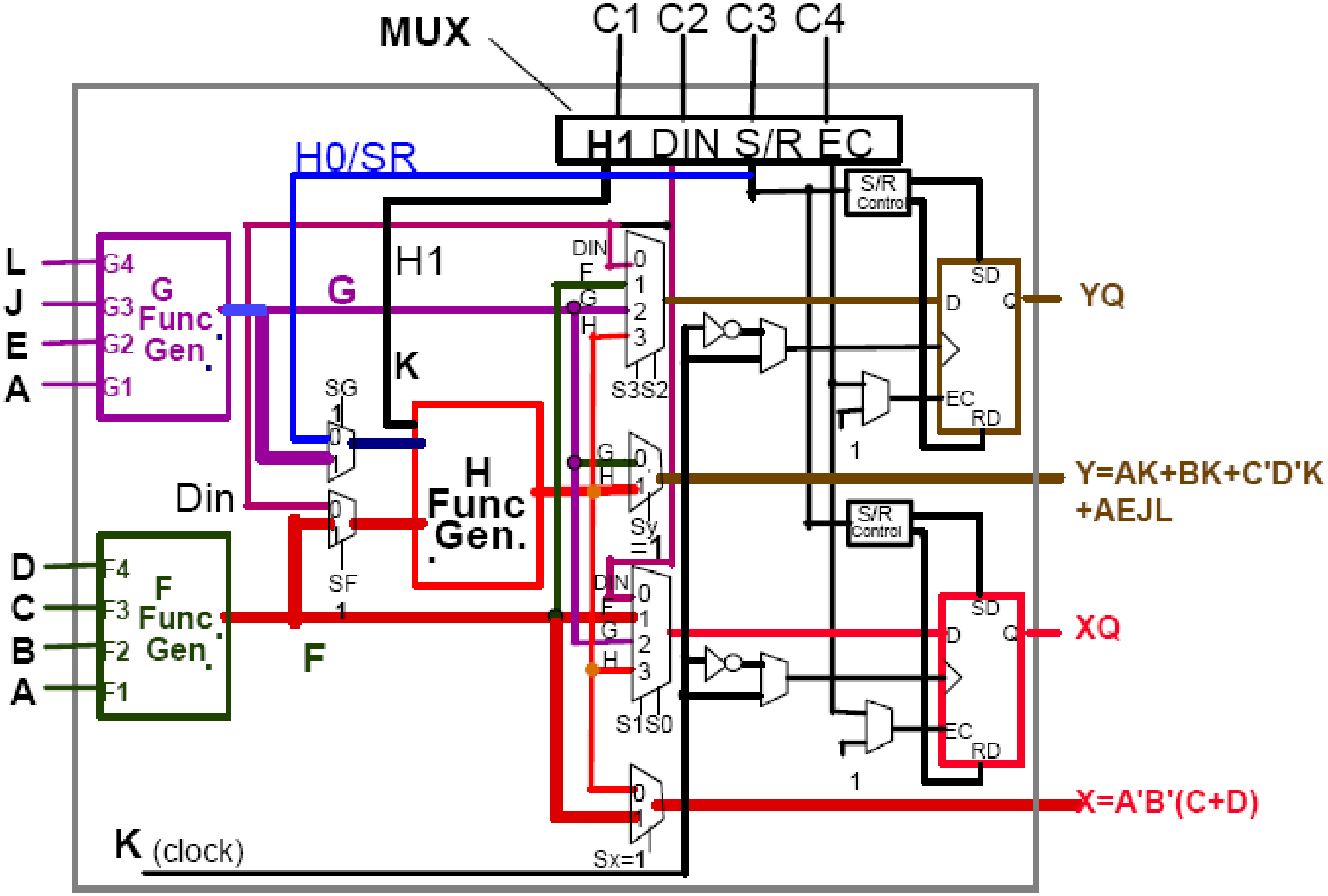
- Implement X on LUT F
- Implement AEJL on LUT G
- Use F,G,H for Y

Take Y=K(A+B+C’D’)+AEJL

=KX’+AEJL

=KF’+G

The solution



XC 4000E Interconnections

Simplified diagram

3 types:

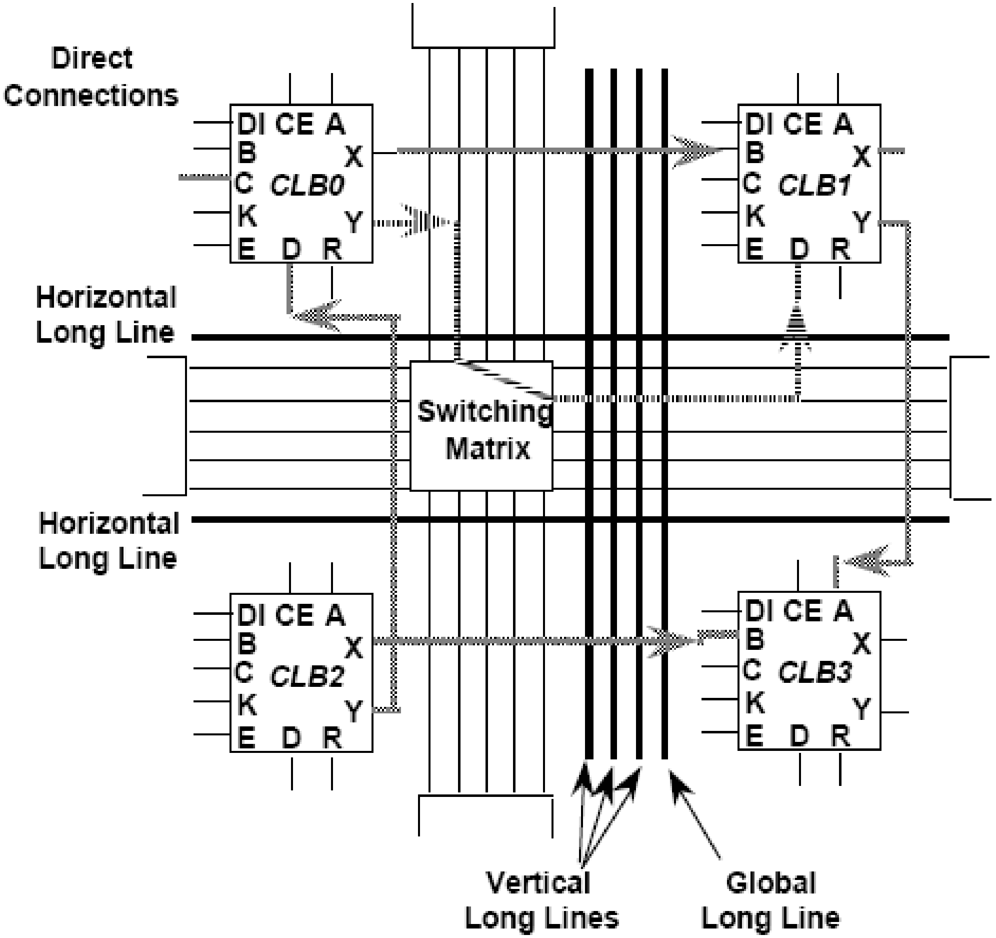
* Fast Direct Connections

* General Purpose Connections with Switching Matrix

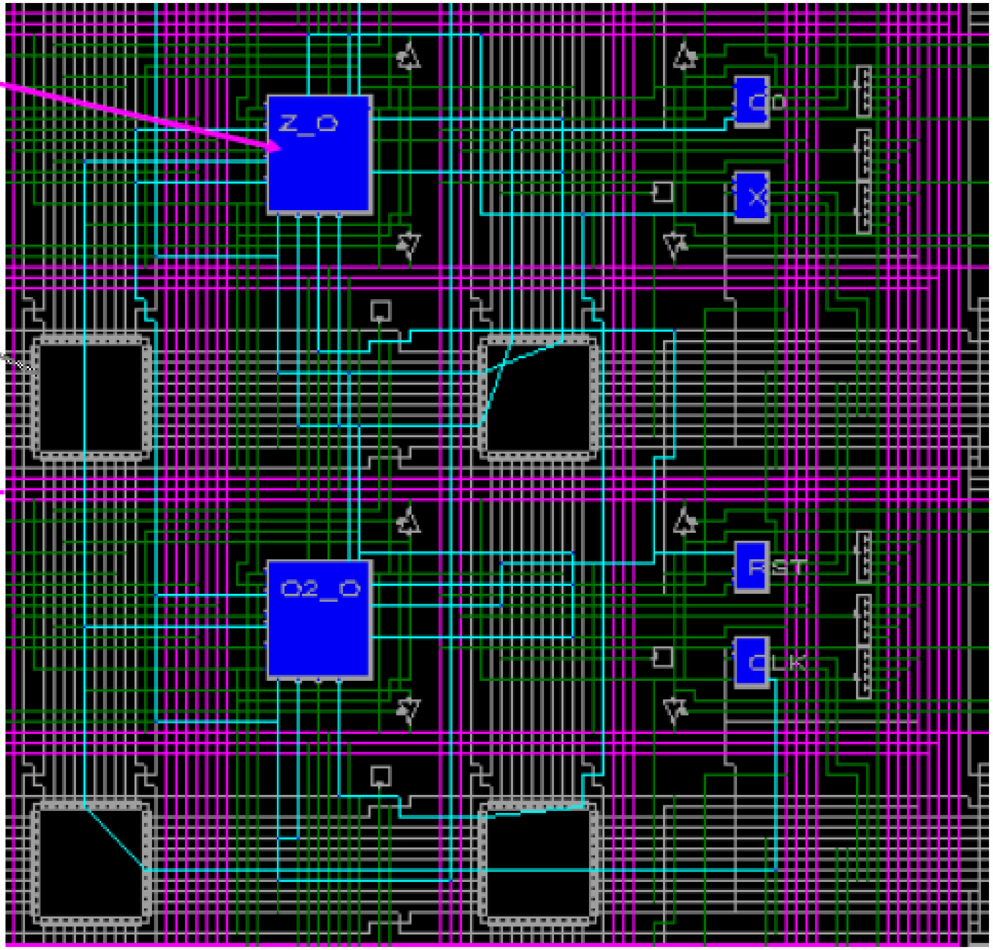
* Horizontal/Vertical Long Lines

Types of lines:

* Single length (8)
* Double length (4)
* Long lines (6)
* Global lines (4)



Detailed view of Internal wiring



CLB (blue)

Direct lines (green)

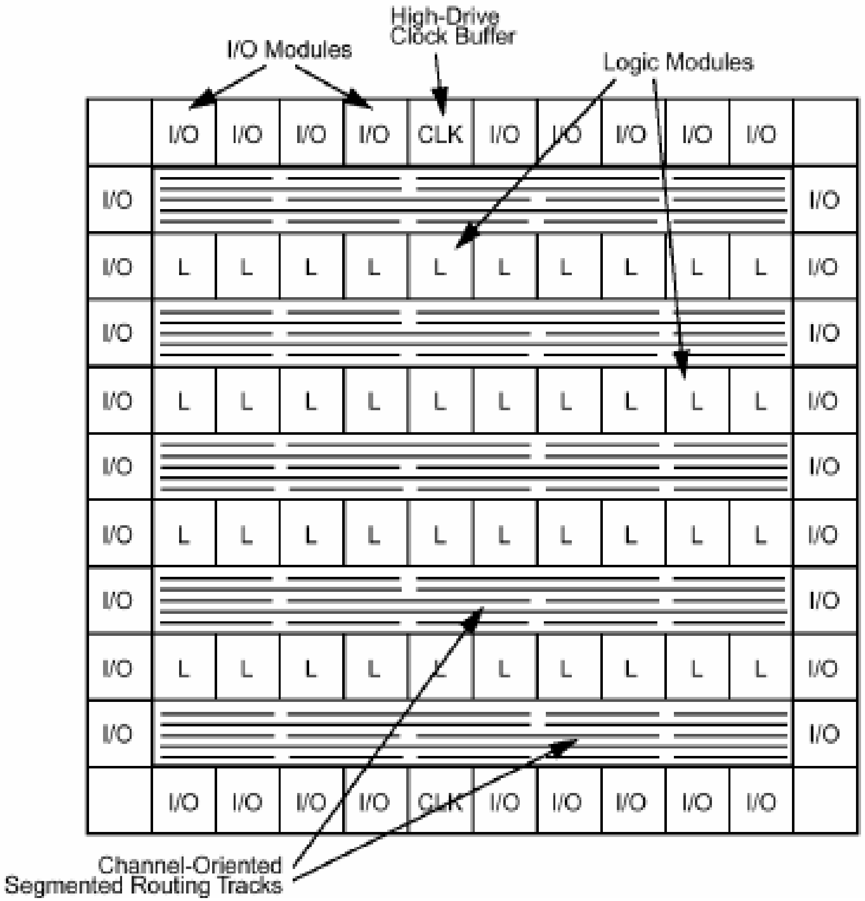
Switch Matrix

Long lines (purple)

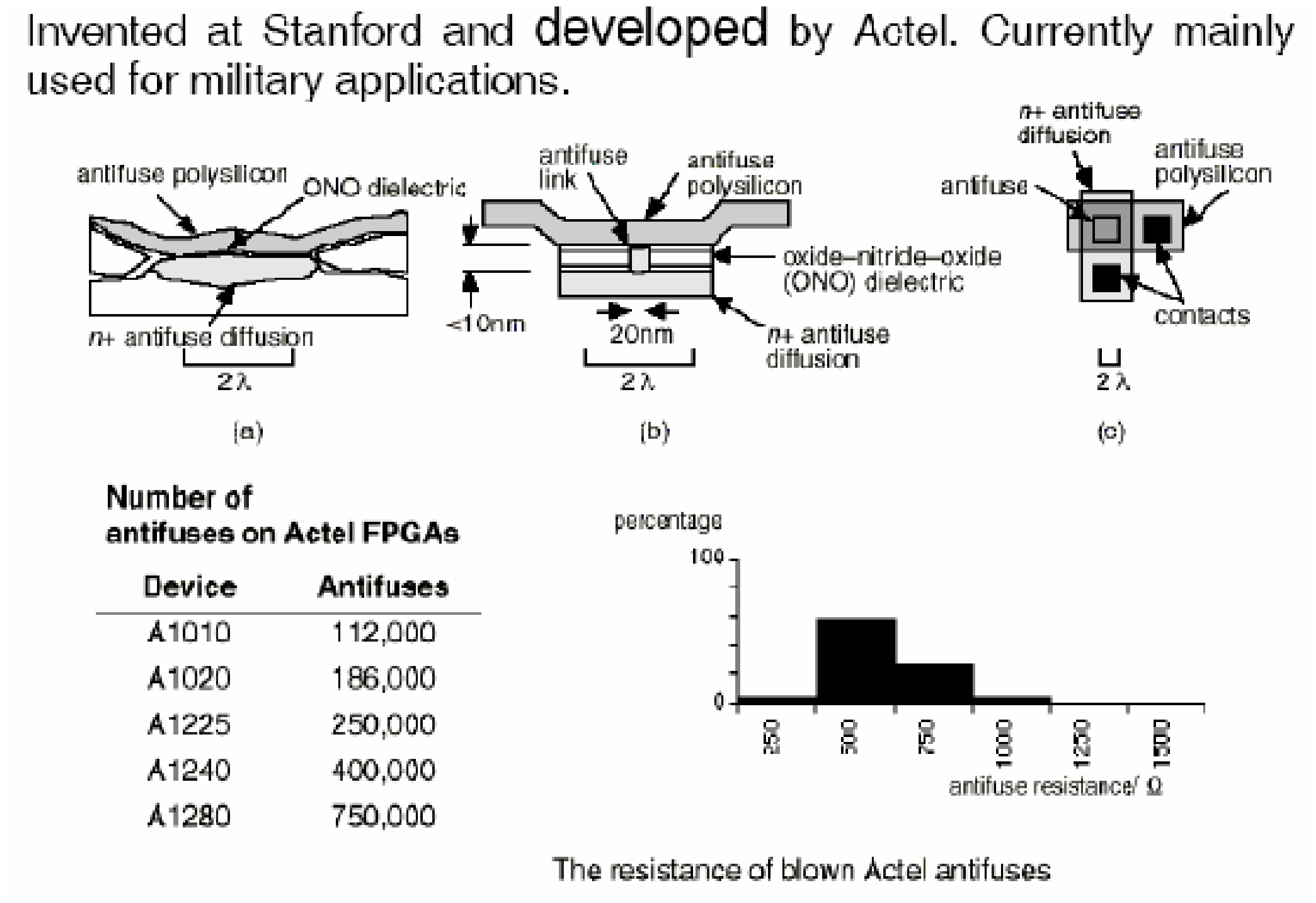
Screen clip from Xilinx Foundation XACTstep(TM) software

Non volatile FPGAs

- Uses antifuse technology
- Based on **channelled gate array** architecture as shown below
- Each logic element (labelled ‘L’) is a combination of multiplexers which can be configured as a multi-input gate



Antifuse Technology



Invented at Stanford and developed by Actel. Currently mainly used for military applications.

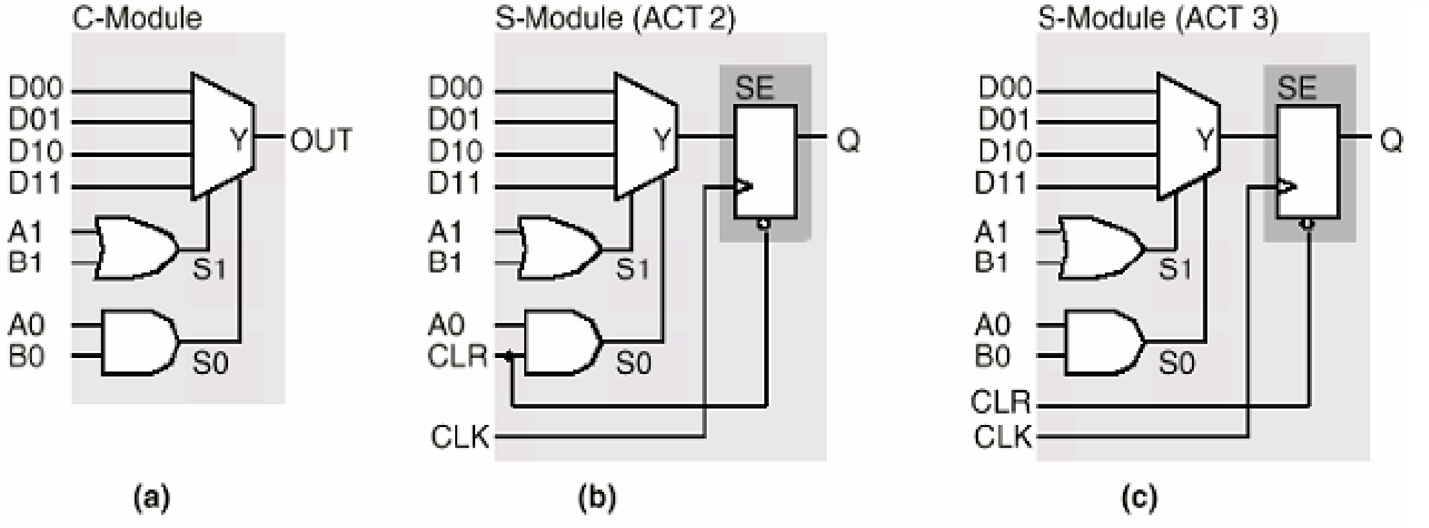
antifuse polysilicon
ONO dielectric
n+ antifuse diffusion
2λ
(a)
antifuse link
antifuse polysilicon
oxide–nitride–oxide (ONO) dielectric
<10nm
20nm
n+ antifuse diffusion
2λ
(b)
n+ antifuse diffusion
antifuse
antifuse polysilicon
contacts
2λ
(c)
percentage
100
0
250
500
750
1000
1250
1500
antifuse resistance/ Ω

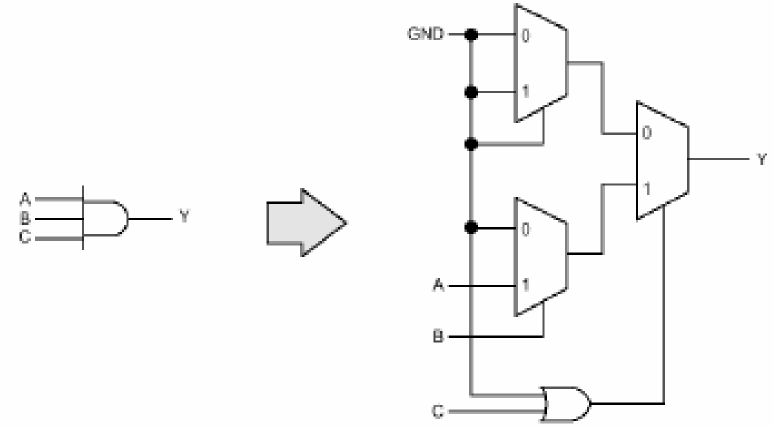
Number of antifuses on Actel FPGAs

Device	Antifuses
A1010	112,000
A1020	186,000
A1225	250,000
A1240	400,000
A1280	750,000

The resistance of blown Actel antifuses

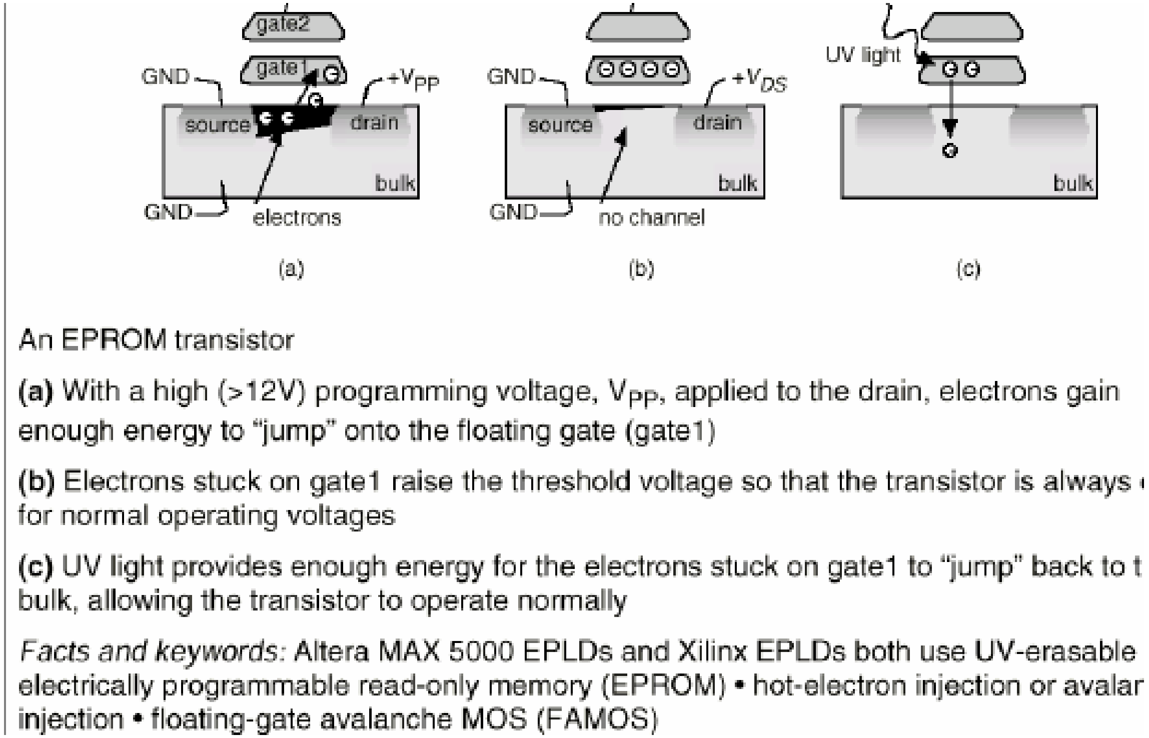
Actel CLB Architecture





EPROM and EEPROM based FPGAs

- Generally used in product-term type of PLDs.
- Non-volatile and reprogrammable.
- Good for FSM, less good for arithmetic circuits.



An EPROM transistor

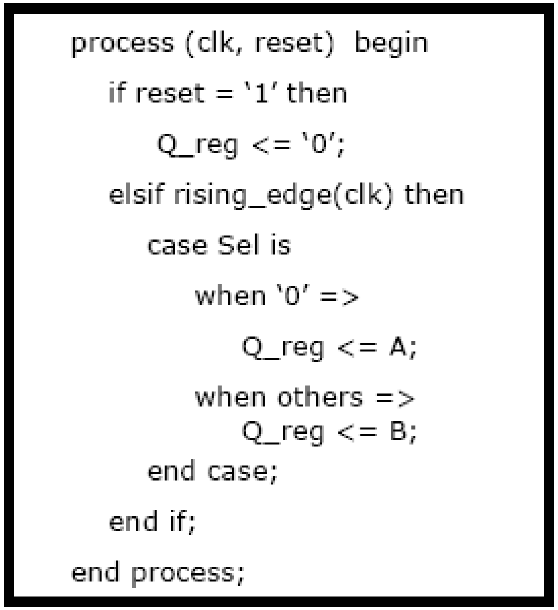
(a) With a high (>12V) programming voltage, V_{PP}, applied to the drain, electrons gain enough energy to “jump” onto the floating gate (gate1)

(b) Electrons stuck on gate1 raise the threshold voltage so that the transistor is always for normal operating voltages

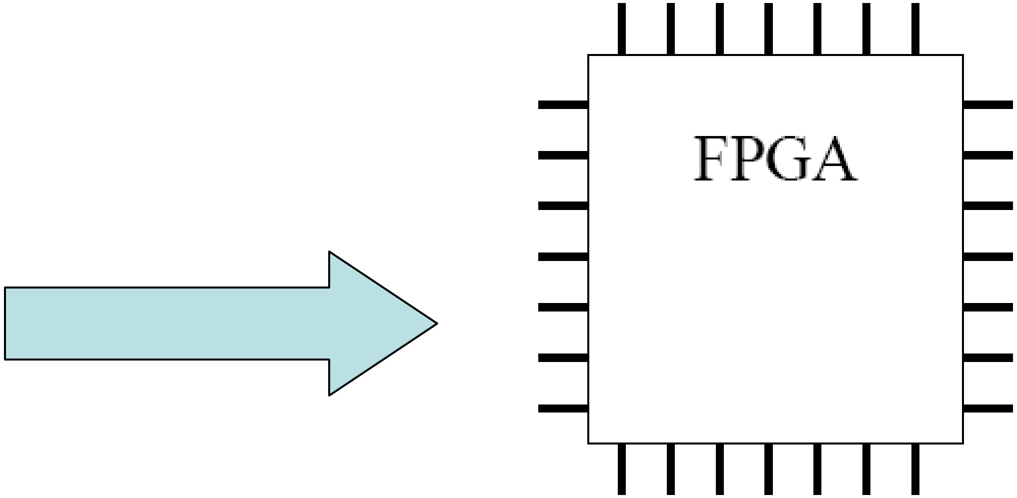
(c) UV light provides enough energy for the electrons stuck on gate1 to “jump” back to t bulk, allowing the transistor to operate normally

Facts and keywords: Altera MAX 5000 EPLDs and Xilinx EPLDs both use UV-erasable electrically programmable read-only memory (EPROM) • hot-electron injection or avalar injection • floating-gate avalanche MOS (FAMOS)

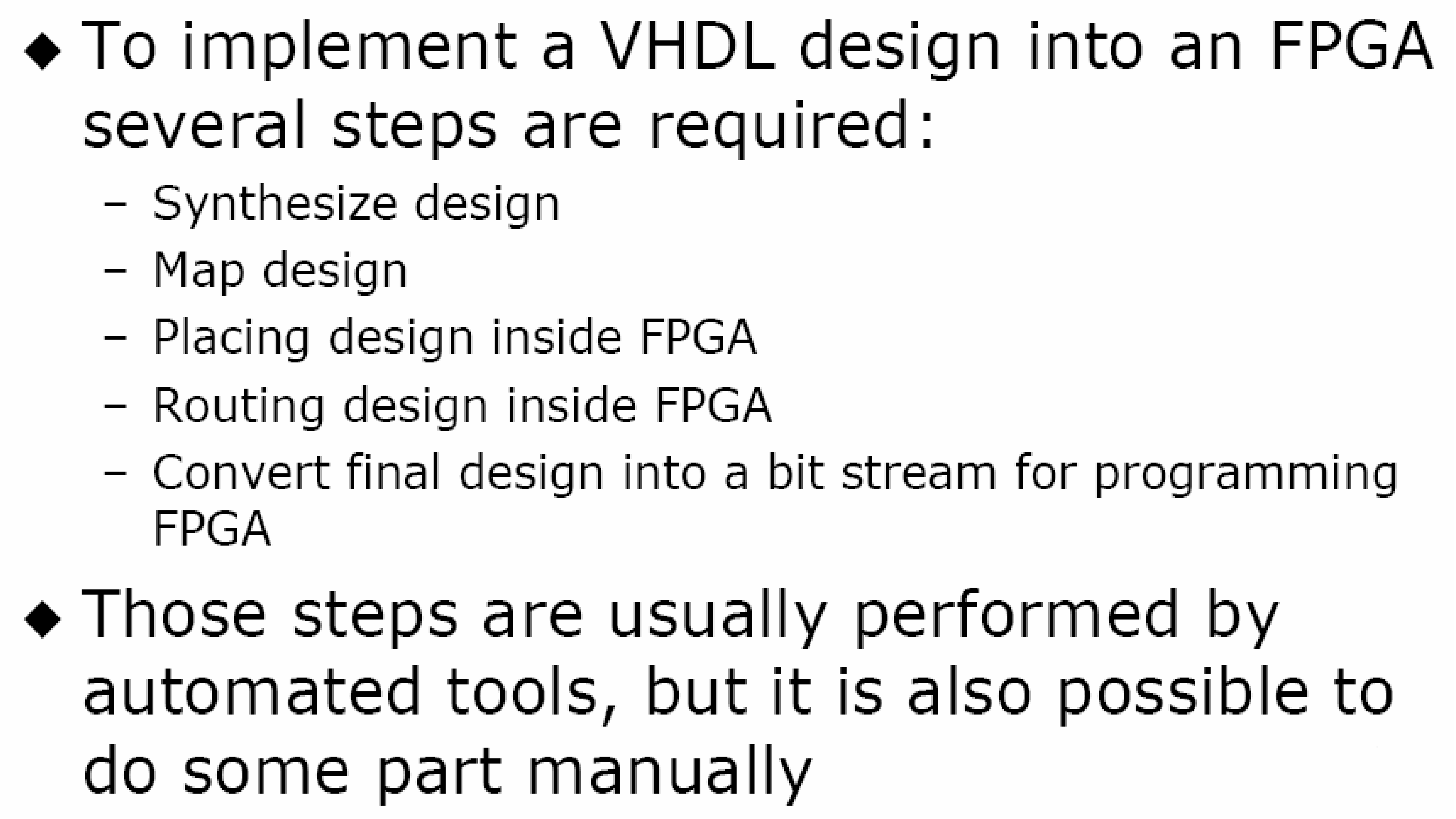
From VHDL to FPGA



```
process (clk, reset)  begin
   if reset = ‘1’ then
       Q_reg <= ‘0’;
   elsif rising_edge(clk) then
      case Sel is
          when ‘0’ =>
              Q_reg <= A;
          when others =>
              Q_reg <= B;
      end case;
   end if;
end process;
```

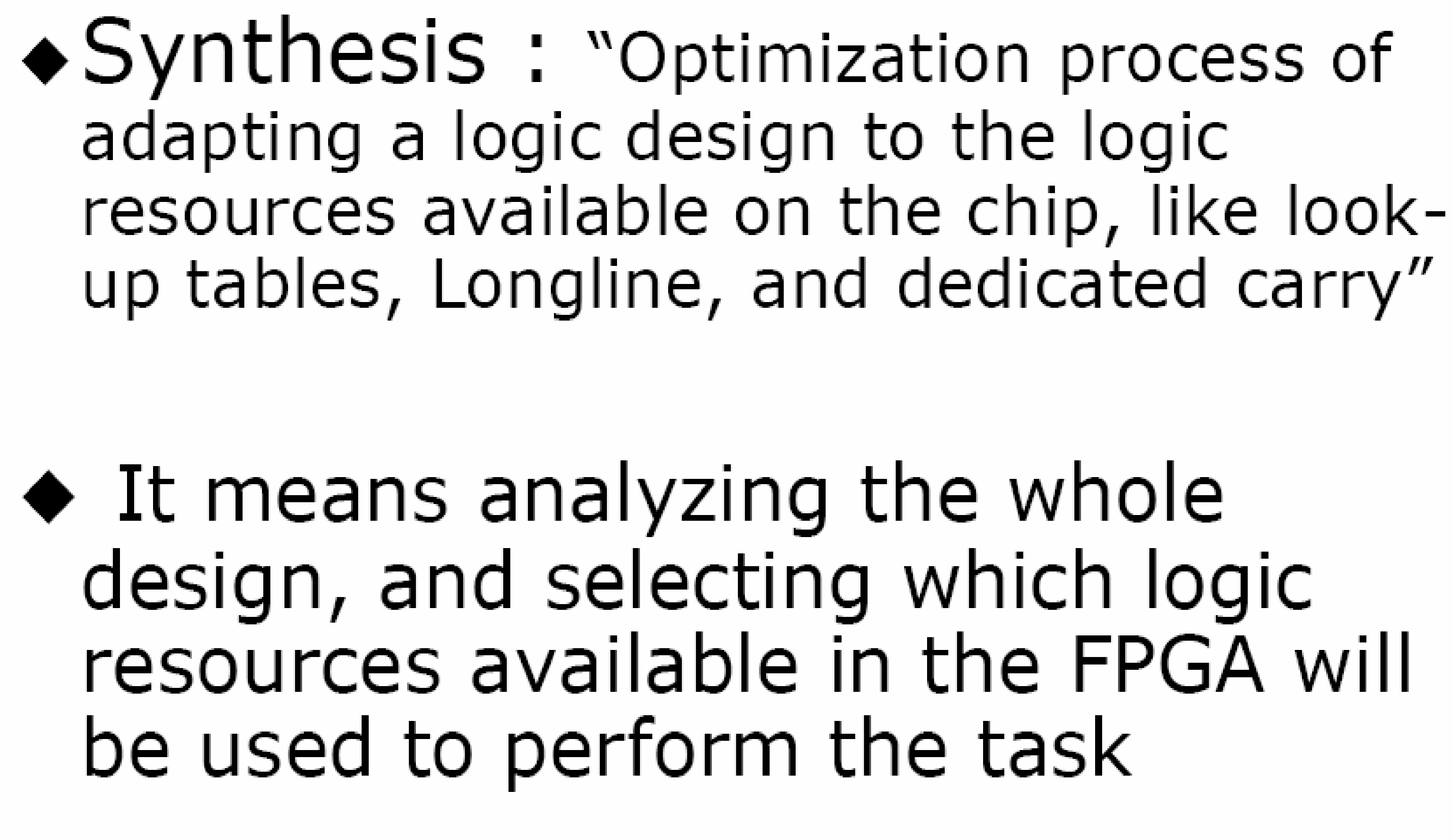


Steps Taken



- To implement a VHDL design into an FPGA several steps are required:
  - Synthesize design
  - Map design
  - Placing design inside FPGA
  - Routing design inside FPGA
  - Convert final design into a bit stream for programming FPGA
- Those steps are usually performed by automated tools, but it is also possible to do some part manually

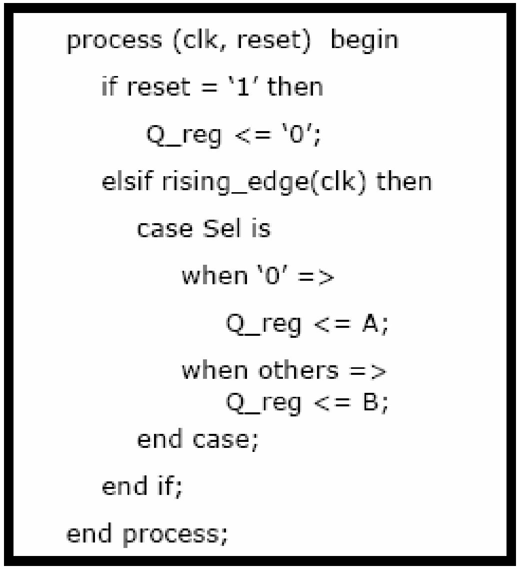
Synthesis



- Synthesis : “Optimization process of adapting a logic design to the logic resources available on the chip, like look-up tables, Longline, and dedicated carry”

- It means analyzing the whole design, and selecting which logic resources available in the FPGA will be used to perform the task

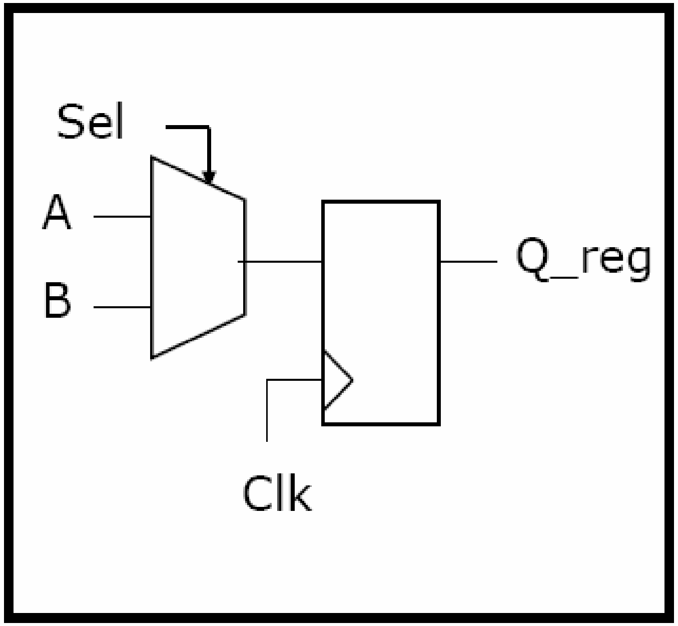
The Synthesis Process



```
process (clk, reset)  begin
    if reset = ‘1’ then
         Q_reg <= ‘0’;
    elsif rising_edge(clk) then
        case Sel is
             when ‘0’ =>
                  Q_reg <= A;
             when others =>
                  Q_reg <= B;
        end case;
    end if;
end process;
```

After synthesis,

we get:



Mapping the design

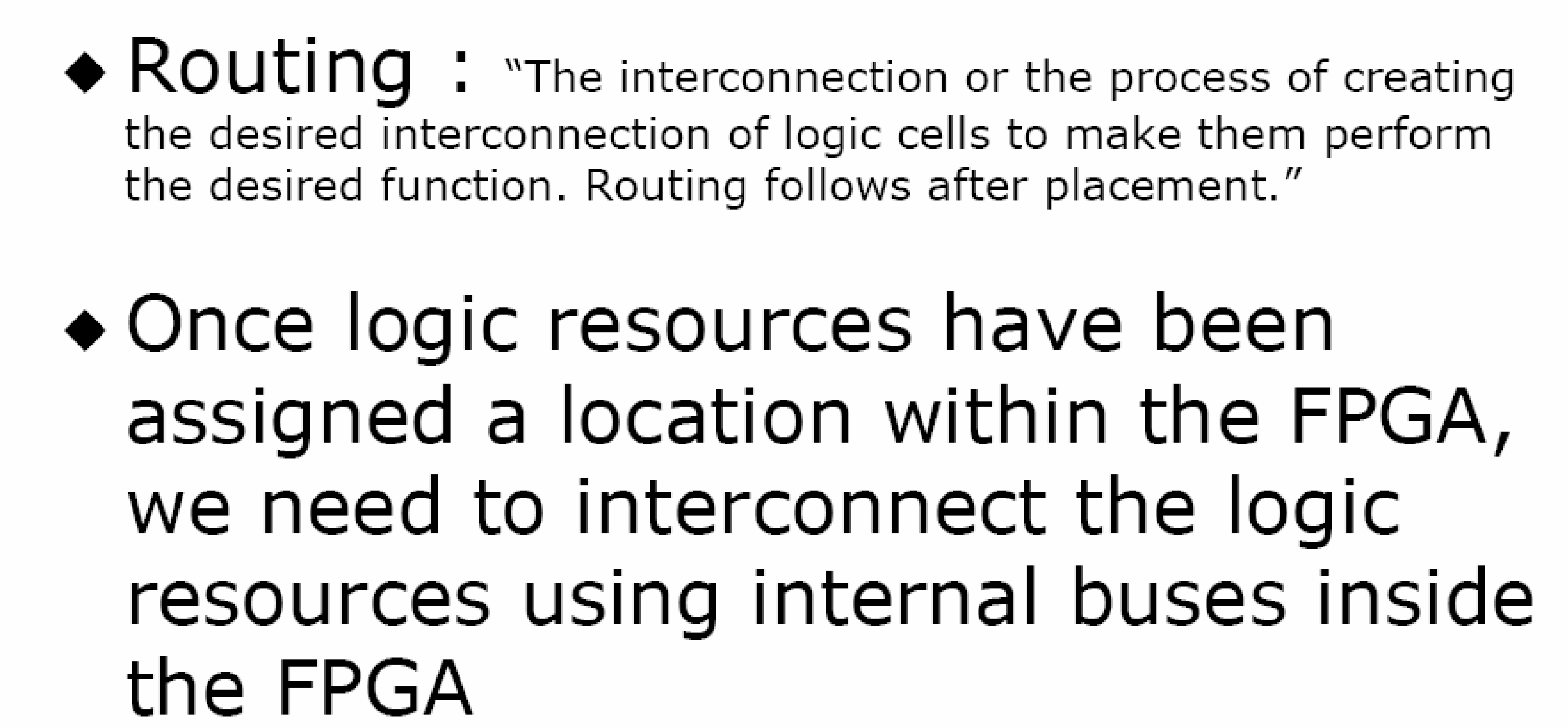
- Mapping : ”Process of assigning portions of the logic design to the physical chip resources (CLBs). With FPGAs, mapping is more demanding and more important a process than with gate arrays.”
- Function similar to synthesizing
- Synthesizing is not necessarily specific to a particular FPGA, but mapping usually is.

Placing the design

- Placing : ” In FPGAs, the process of assigning specific parts of the design to specific locations (CLBs) on the chip. Usually done automatically.”

- Once the design has been converted into the logic resources of the FPGA, we find a location for these within the FPGA

Routing the design



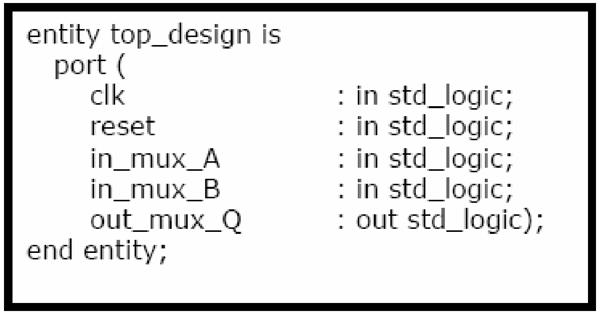
- Routing : “The interconnection or the process of creating the desired interconnection of logic cells to make them perform the desired function. Routing follows after placement.”
- Once logic resources have been assigned a location within the FPGA, we need to interconnect the logic resources using internal buses inside the FPGA

Convert design into bit stream

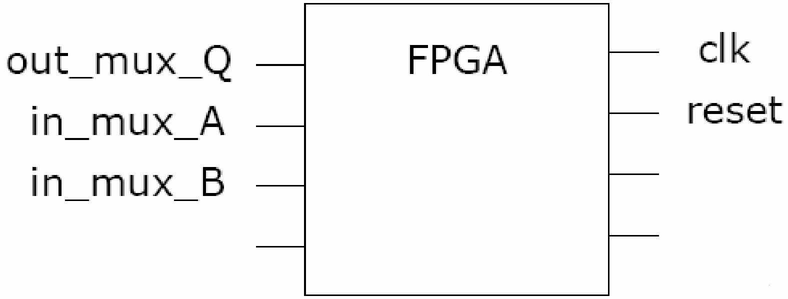
- This always done using an automated tools.
- The placed and routed designed is converted into a bit-stream that is downloaded into the FPGA to configure it

Assigning package pins

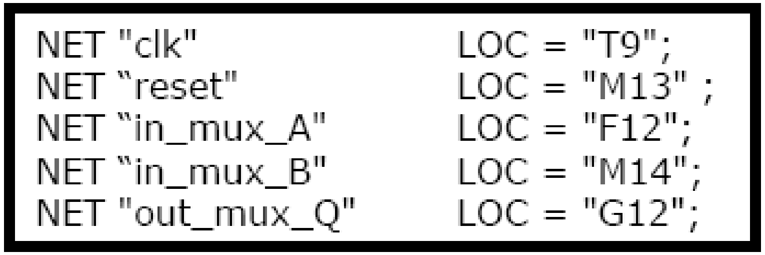
- When implementing an entity in FPGA, the input and output ports are mapped to pins of the FPGA



```
entity top_design is
   port (
       clk                    : in std_logic;
       reset                  : in std_logic;
       in_mux_A               : in std_logic;
       in_mux_B               : in std_logic;
       out_mux_Q              : out std_logic);
end entity;
```



UCF File ⟹



```
NET "clk"               LOC = "T9";
NET “reset"             LOC = "M13" ;
NET “in_mux_A"          LOC = "F12";
NET “in_mux_B"          LOC = "M14";
NET "out_mux_Q"         LOC = "G12";
```

Future Works

- Power-delay minimization during run time
- Area, power and delay efficient CLB allocation during compilation time

Acknowledgements

- Xilinx Inc
- Altera Corporation
- Actel Corporation

THANK YOU