Introduction to FPGA

Honey Khindri, Pritam Palit, Ritu Devi

SERB School – 2019, TIFR

Outline of Talk

- Introduction
- What is FPGA and how it works?
- Why FPGA?
- Specification, Architecture and Max 10 board used for the project
- Installation and Set-up
- Exercise 1 – AND Gate
- Exercise 2 – Half Adder
- Exercise 3 – Hexagonal Counter

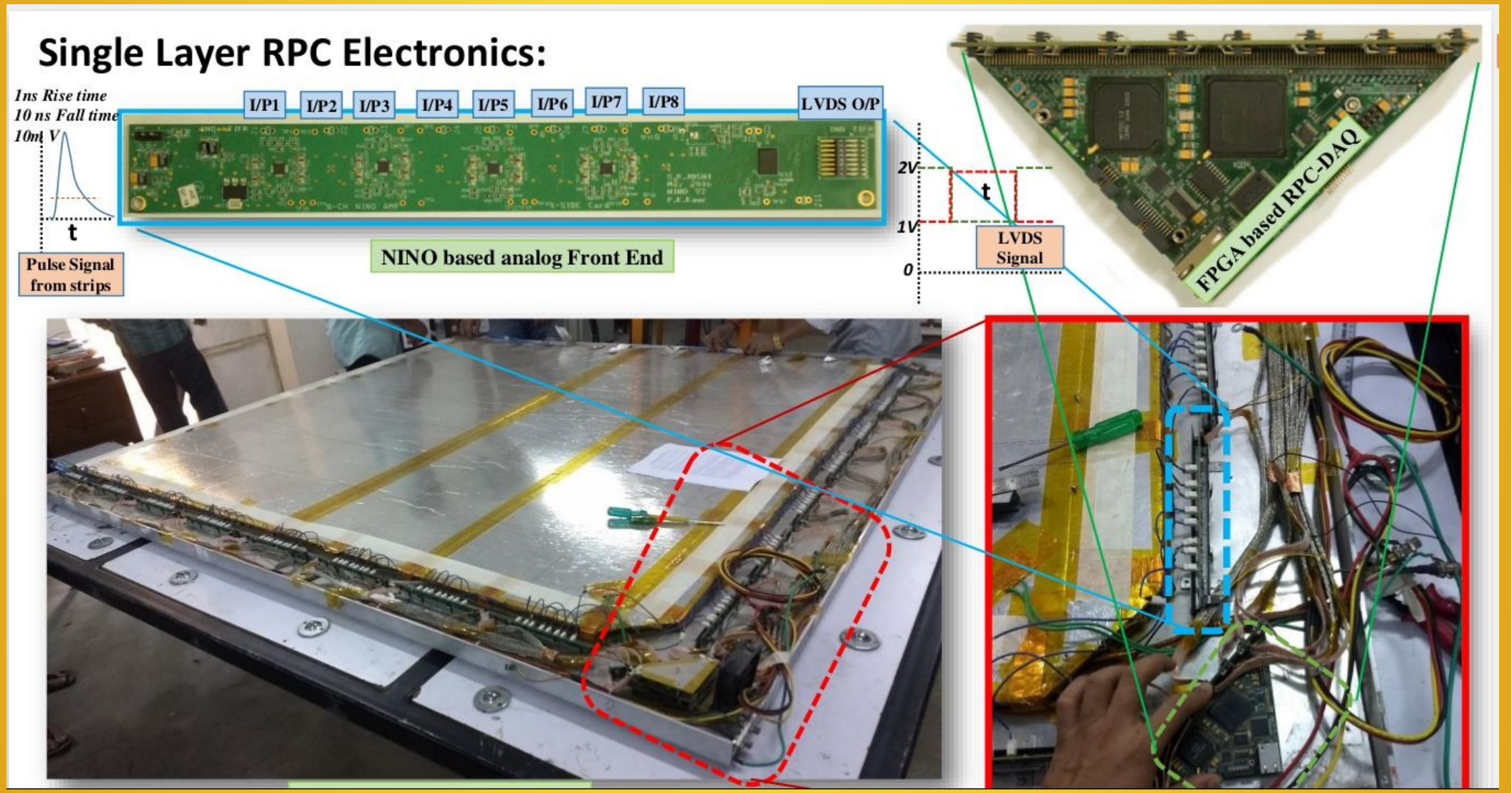
What is FPGA?

- FPGA is “Field Programmable Gate Array”
- Gate Array is prefabricated semiconductor device, like a silicon chip.
- FPGA, is an integrated circuit that can be configured ‘in the field’ by the designer to perform certain operations.
- When needed, the FPGA can be reprogrammed to perform a completely different task from its original one

Why FPGA?

- Reconfigurable
- Small Area
- Less power consumption
- Less cost
- Speed due to parallelism

Parallelism - Example



How FPGA Works

- In FPGAs, there is no defined hardware and we are the one designing the circuit. We can configure an FPGA as we want for our purpose.
- To create a design we write Hardware Description Language (HDL), which is of two types – Verilog and VHDL.
- Then the HDL is synthesized into a bit file using a BITGEN to configure the FPGA
- The FPGA stores the configuration in two type of files

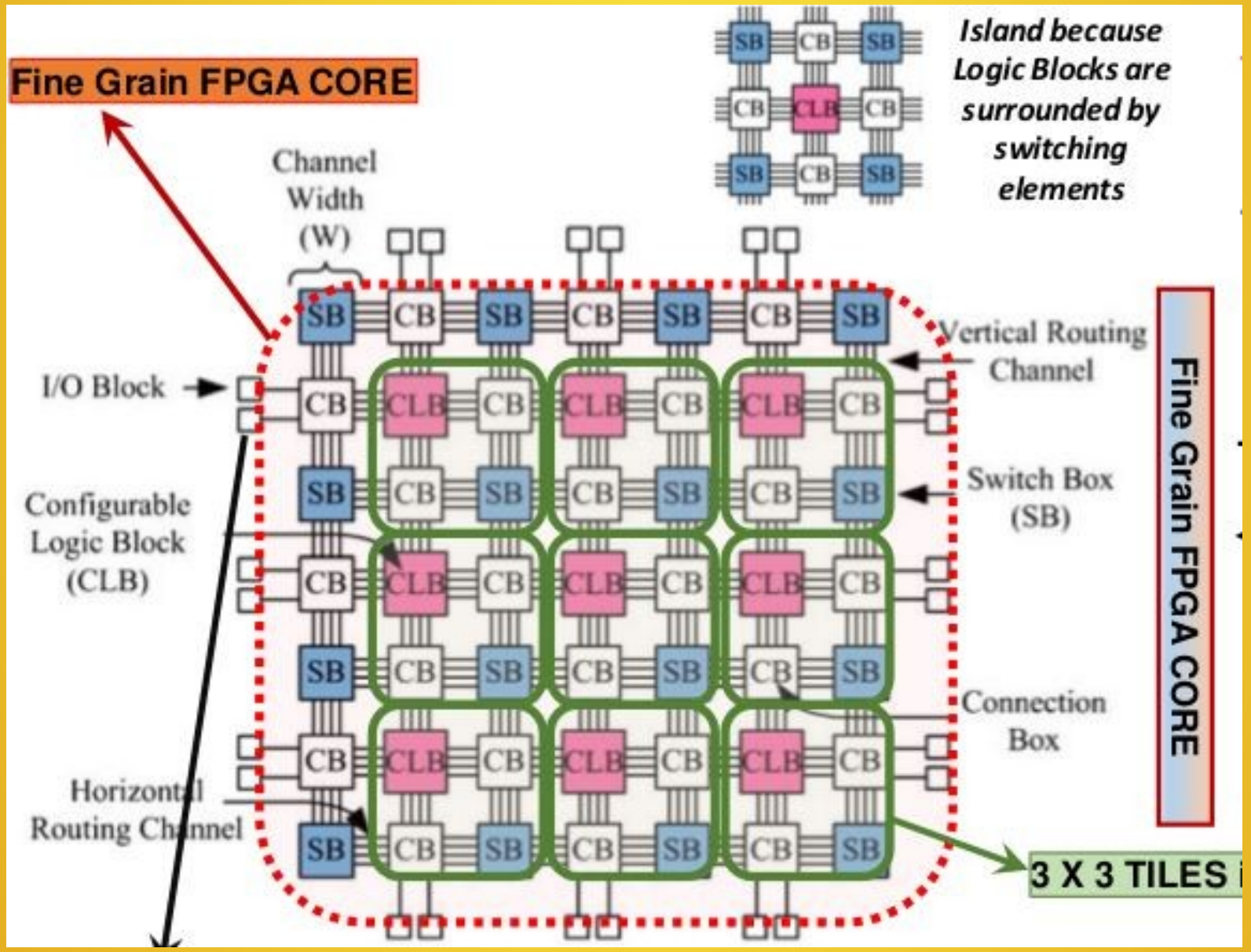
  - .sof (Volatile memory, lost after power off) : SRAM

  - .pof (Non-volatile memory) : EPROM

Specification and programming language

- There are several FPGA vendors, e.g. Xillinx, ALTERA, Lattice etc.
- We here used ALTERA
- Altera has different families , Max10, Cyclone4, Cyclone5 etc depending on requirements.
- We used Max10, due to its low cost.
- VHDL (VHSIC Hardware description language) is used for programming the FPGA

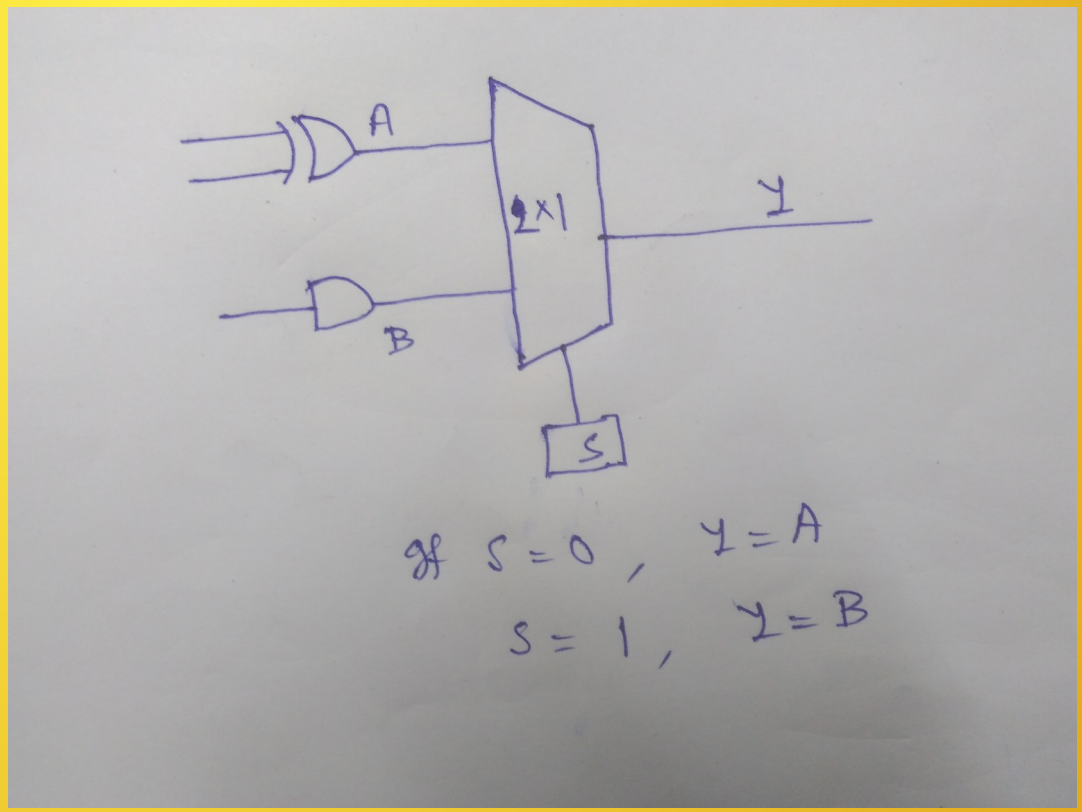
FPGA Core Structure



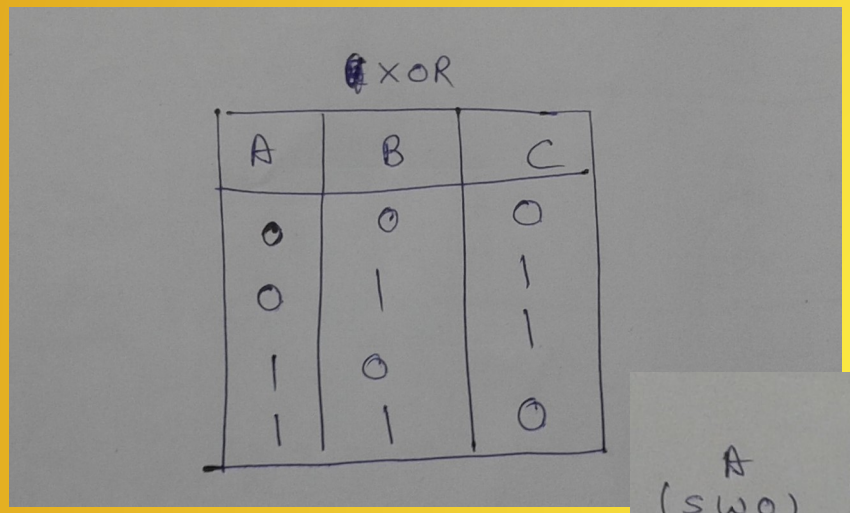
CB and SB

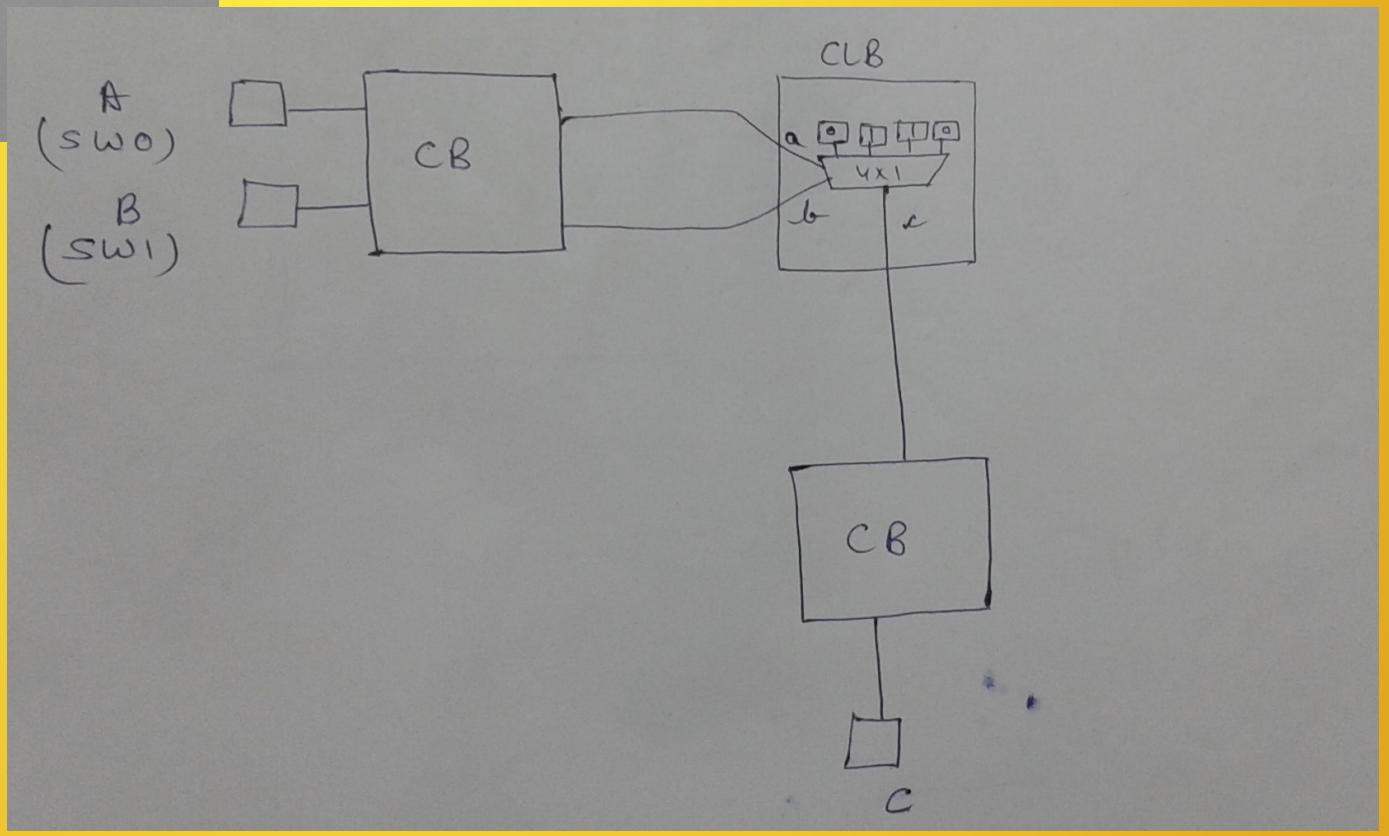
- MUX is used as a switch.

  - Both CB, SB act as switch. CB can connect horizontally or vertically, while SB can do both.



CLB and LUT





Max10 Development Board Architecture

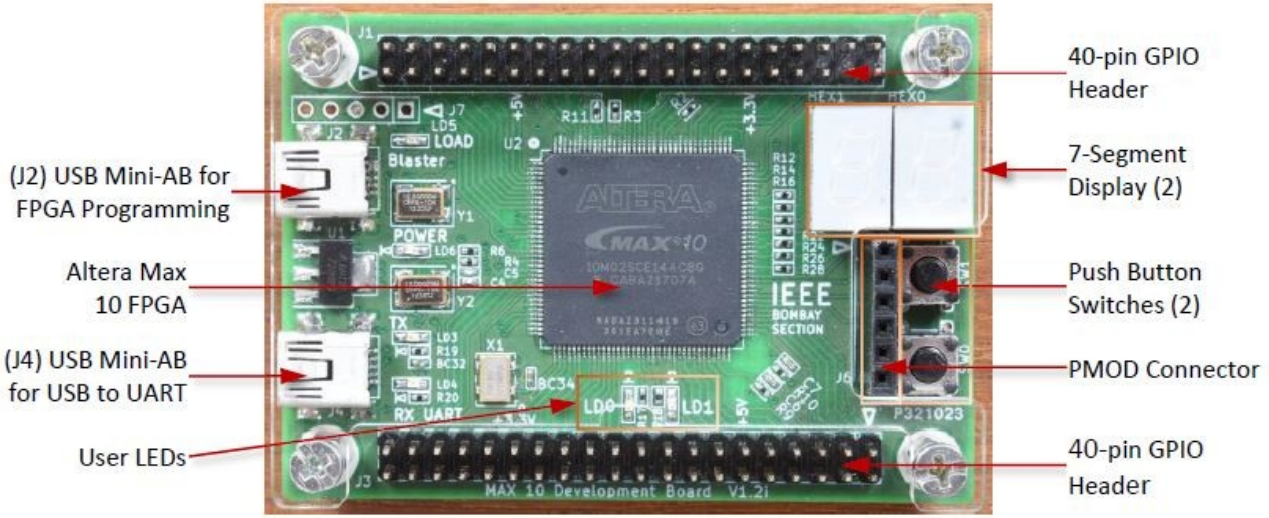


Figure 1. The Max_10_Dev Board Top Side

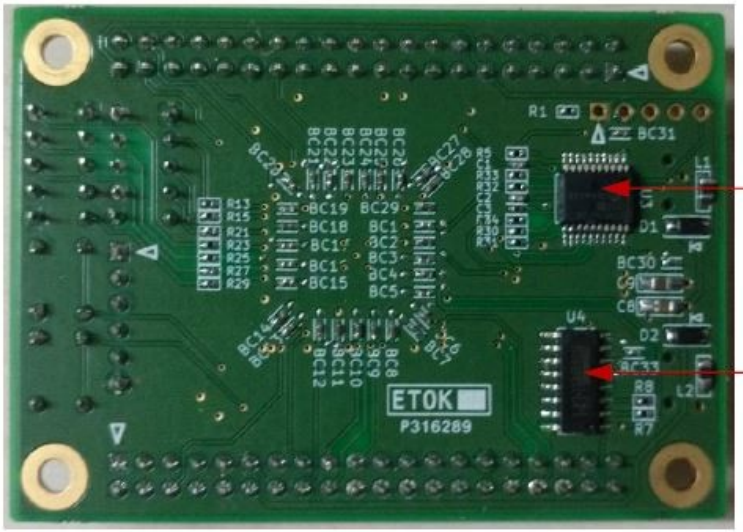


Figure 2. The Max_10_Dev Board bottom side

Software installation for the Project

- Quartus Prime (Custom Software by ALTERA)

  - To compile and load the VHDL code and program the FPGA hardware accordingly

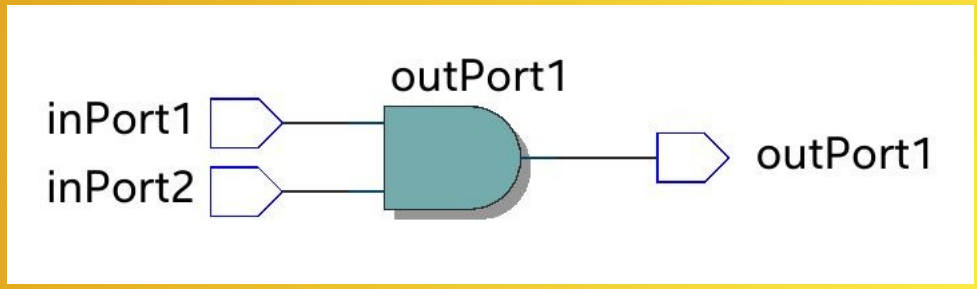
- ModelSim-Intel FPGA Edition

  - To emulate , i.e. to simulate the hardware using software

- MAX 10 FPGA device support

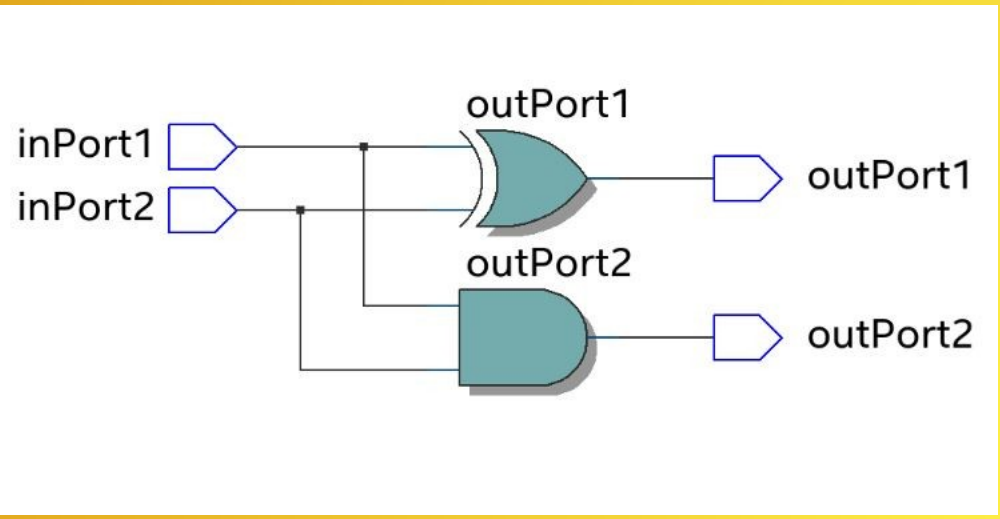
  - To create the environment of Max 10 family

Exercise 1 - AND



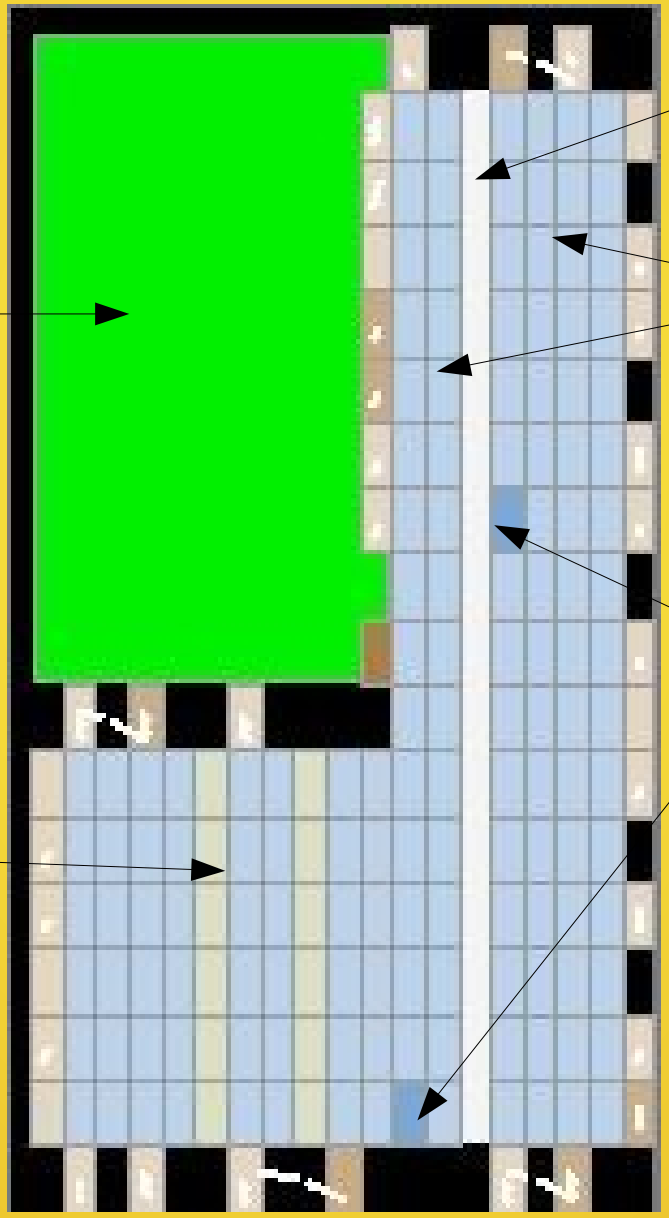

```
--import std_logic from the IEEE library
library ieee;
use ieee.std_logic_1164.all;

--ENTITY DECLARATION: name, inputs, outputs
entity AND_gate is
   port( inPort1  : in std_logic;
         inPort2  : in std_logic;

         outPort1 : out std_logic );

end AND_gate;

--FUNCTIONAL DESCRIPTION: how the Inverter works
architecture func of AND_gate is
begin
   outPort1 <= inPort1 and inPort2;


end func;
```

Exercise 2 – Half Adder

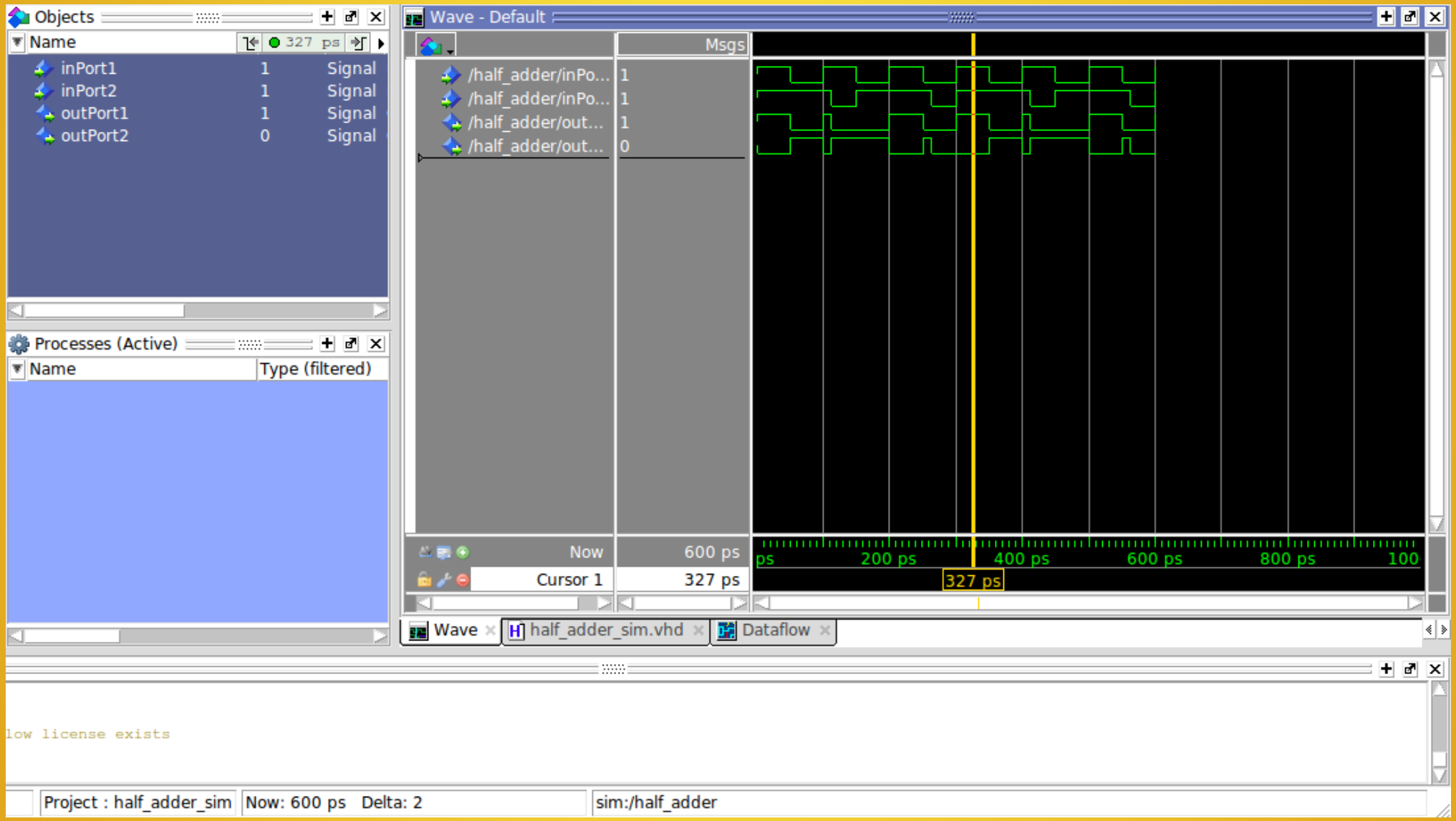



```
--import std_logic from the IEEE library
library ieee;
use ieee.std_logic_1164.all;

--ENTITY DECLARATION: name, inputs, outputs
entity Half_adder is
   port( inPort1  : in std_logic;
         inPort2  : in std_logic;

         outPort1 : out std_logic;
                    outPort2 : out std_logic);
end Half_adder;

--FUNCTIONAL DESCRIPTION: how the Inverter works
architecture func of Half_adder is
begin
   outPort1 <= inPort1 xor inPort2;
        outPort2 <= inPort1 and inPort2;

end func;
```

Memory Allocation in FPGA

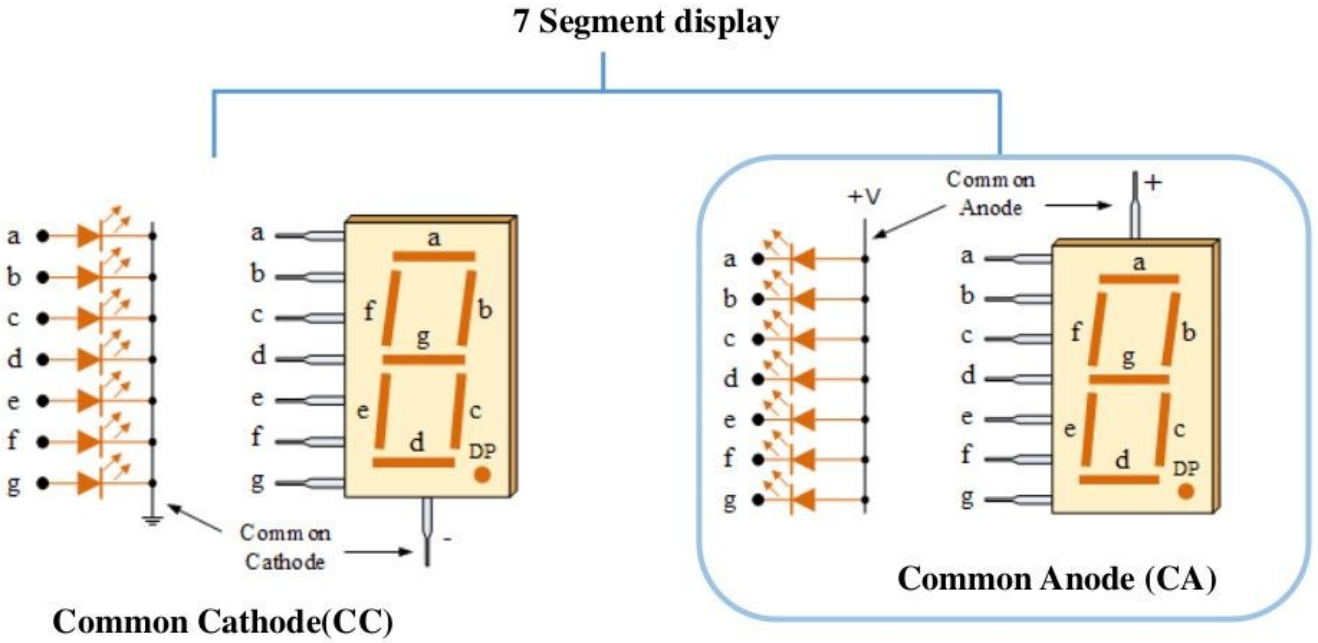


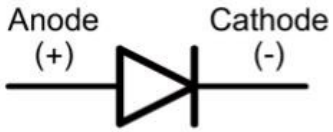
Emulation of Half Adder by ModelSim



Hexagonal Counter – Display types

SEVEN SEGMENT DISPLAY TYPES





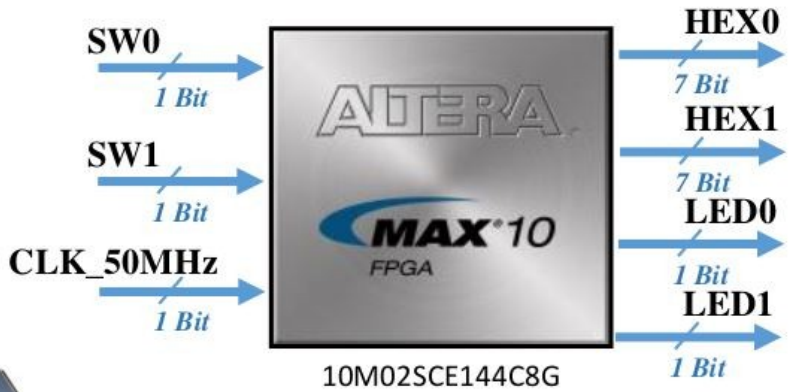
Exercise 3 – Hexagonal Counter

PIN OUT REQUIRED

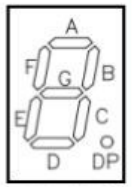
SW0 (PIN_70)

SW1 (PIN_71)

CLK_50MHz (PIN_28)

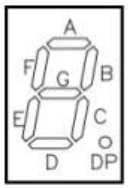


10M02SCE144C8G



HEX0

HEX0[0]	PIN_101	Segment A
HEX0[1]	PIN_99	Segment B
HEX0[2]	PIN_97	Segment C
HEX0[3]	PIN_94	Segment D
HEX0[4]	PIN_95	Segment E
HEX0[5]	PIN_100	Segment F
HEX0[6]	PIN_98	Segment G



HEX1

HEX1[0]	PIN_92	Segment A
HEX1[1]	PIN_90	Segment B
HEX1[2]	PIN_88	Segment C
HEX1[3]	PIN_86	Segment D
HEX1[4]	PIN_87	Segment E
HEX1[5]	PIN_91	Segment F
HEX1[6]	PIN_89	Segment G

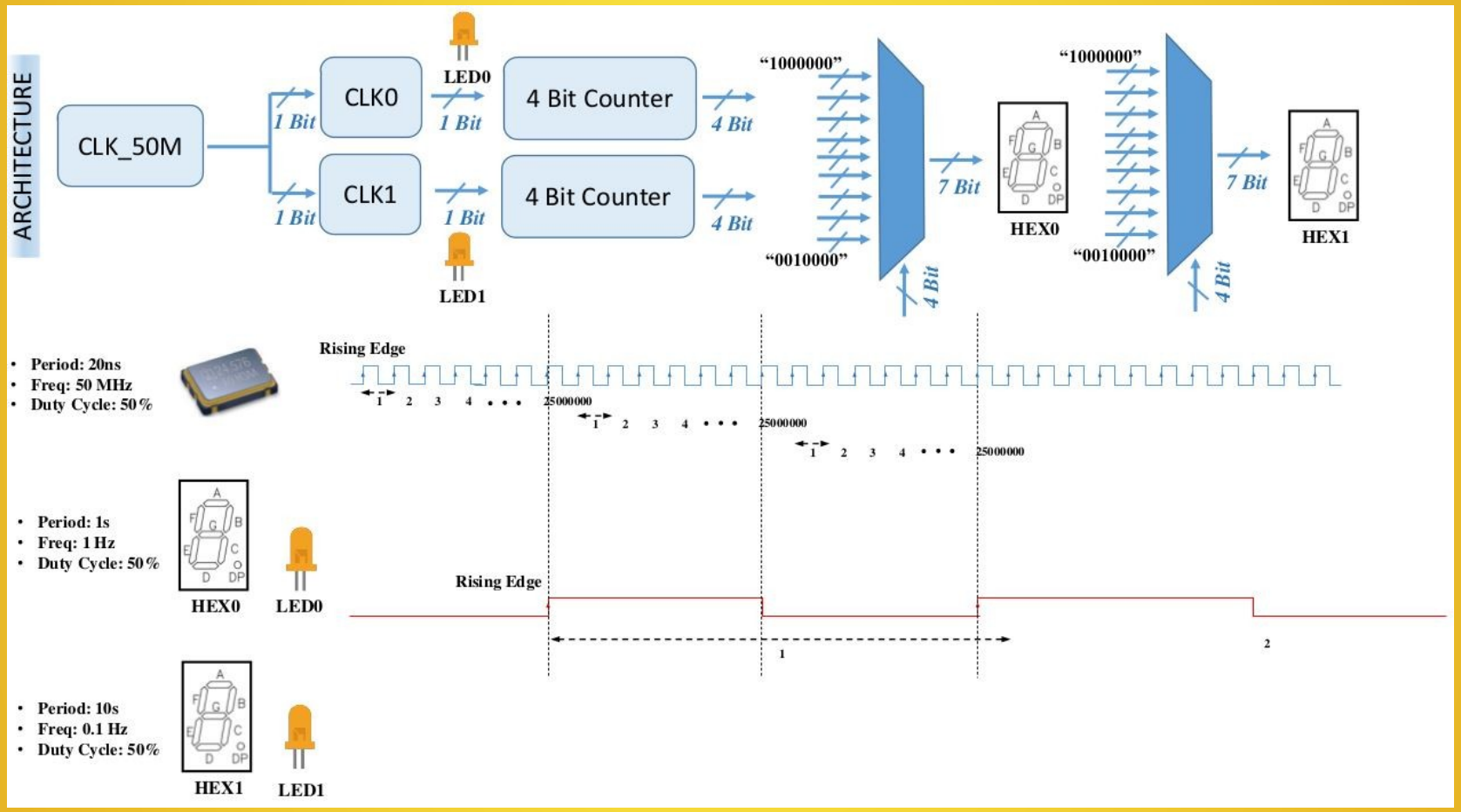
LED0 (PIN_54)

LED1 (PIN_55)

```
Entity HEX_COUNTER is
        port
        (
                CLK_50M :    in std_logic;
                SW0             :    in std_logic;
                HEX0    :   out std_logic_vector(6 downto 0);-- A->0 || B
                HEX1    :   out std_logic_vector(6 downto 0);
                LED0    :   out std_logic;
                LED1    :   out std_logic

        );
end HEX_COUNTER;



architecture behav of HEX_COUNTER is

    signal Rst: STD_LOGIC;
         signal Pause: STD_LOGIC;
```

https://www.mouse

Hexagonal Counter Architecture



Hexagonal Counter Code

```
HEX0 <= "1000000" when counter0_SHARE = "0000" else
   "1111001" when counter0_SHARE = "0001" else
                "0100100" when counter0_SHARE = "0010" else
                "0110000" when counter0_SHARE = "0011" else
                "0011001" when counter0_SHARE = "0100" else
                "0010010" when counter0_SHARE = "0101" else
                "0000010" when counter0_SHARE = "0110" else
                "1111000" when counter0_SHARE = "0111" else
                "0000000" when counter0_SHARE = "1000" else
                "0010000" when counter0_SHARE = "1001" else
   "1000000";

HEX1 <= "1000000" when counter1_SHARE = "0000" else
   "1111001" when counter1_SHARE = "0001" else
                "0100100" when counter1_SHARE = "0010" else
                "0110000" when counter1_SHARE = "0011" else
                "0011001" when counter1_SHARE = "0100" else
                "0010010" when counter1_SHARE = "0101" else
                "0000010" when counter1_SHARE = "0110" else
                "1111000" when counter1_SHARE = "0111" else
                "0000000" when counter1_SHARE = "1000" else
                "0010000" when counter1_SHARE = "1001" else
   "1000000";
```

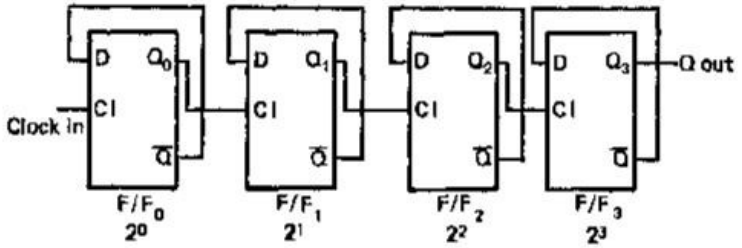
Acknowledgement

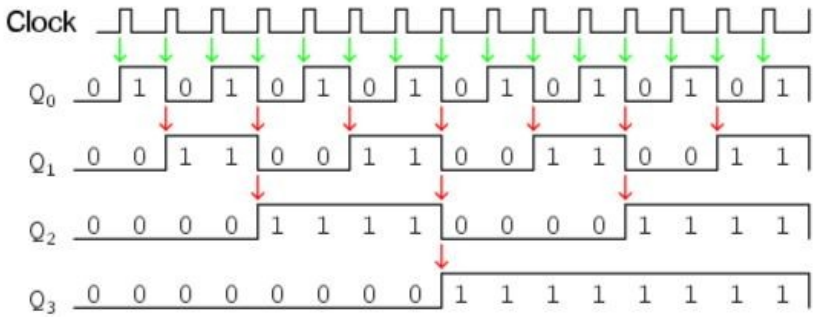
- We would like to thank our project supervisor, Salam Thoithoi Singh and also the EHEP organizers for this wonderful project.

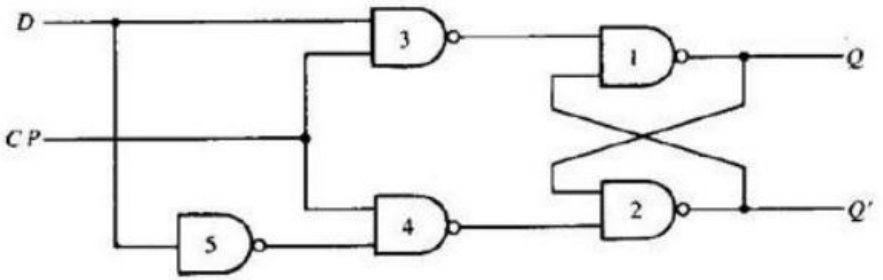
Thank you!

Hexagonal Counter

COUNTER USING D-FLIPFLOP



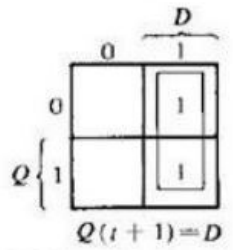




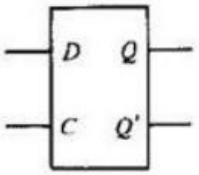
(a) Logic diagram

Q	D	Q(t + 1)
0	0	0
0	1	1
1	0	0
1	1	1

(b) Characteristic table



(c) Characteristic equation



(d) Graphic symbol