BIRZEIT UNIVERSITY

Faculty of Engineering and Technology

Department of Electrical and Computer Engineering

DIGITAL ELECTRONICS AND COMPUTER ORGANIZATION LABORATORY (ENCS2110)

"(Experiment9)"

Prepared by:

Name: Eman Asfour **Number:** 1200206

Instructor: Dr. Jamal Seyam

Section: 4

Date: 8 / 14 / 2023

```
1  module priority_encoder_4x2 (input [3:0] A);
2    reg [1:0] Y;
3    always @* begin
4    Y[1:0] = 2'b00;
5    if (A[3] == 1'b1) Y[1:0] = 2'b11;
6    else if (A[2] == 1'b1) Y[1:0] = 2'b10;
7    else if (A[1] == 1'b1) Y[1:0] = 2'b01;
8    else if (A[0] == 1'b1) Y[1:0] = 2'b00;
9    end
10   endmodule
11
```

Figure 1: The code for priority_encoder_4x2

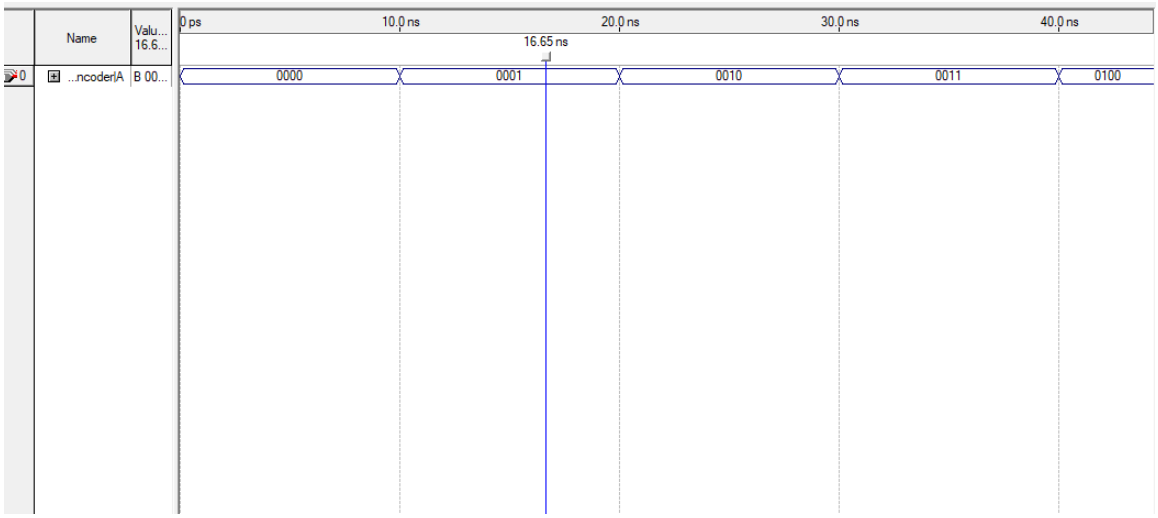


Figure 2: The waveform for priority_encoder_4x2

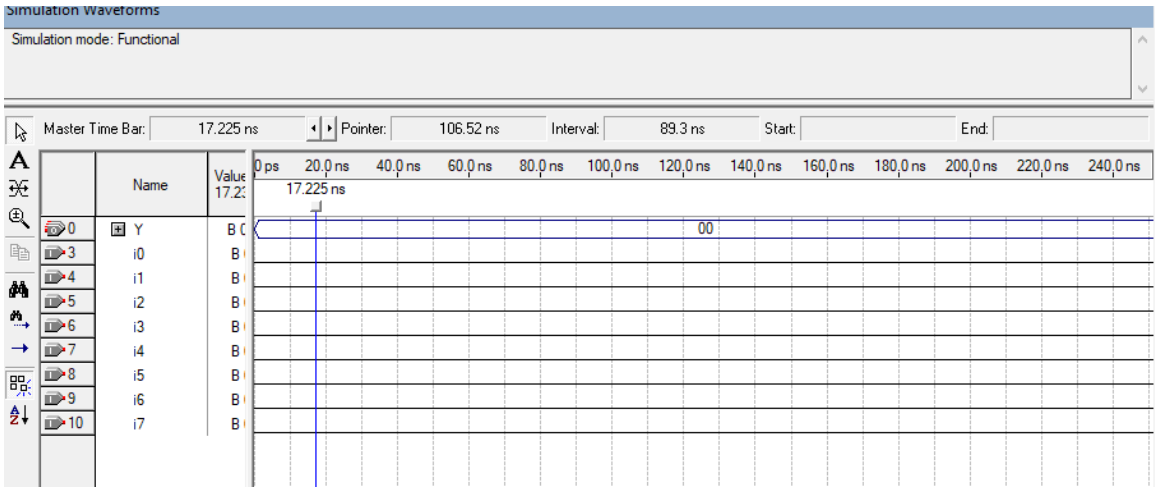


Figure 3: the waveform 4 to 2

```
1   module priority_encoder_4x2 (
2       input i0, i1, i2, i3, i4, i5, i6, i7,
3       output reg [1:0] Y
4   );
5
6       always @* begin
7           Y[1:0] = 2'b00;
8
9           if (i7) Y[1:0] = 2'b11; // Input = 7
10          else if (i6) Y[1:0] = 2'b11; // Input = 6
11          else if (i5) Y[1:0] = 2'b11; // Input = 5
12          else if (i4) Y[1:0] = 2'b11; // Input = 4
13          else if (i3) Y[1:0] = 2'b10; // Input = 3
14          else if (i2) Y[1:0] = 2'b10; // Input = 2
15          else if (i1) Y[1:0] = 2'b01; // Input = 1
16          else if (i0) Y[1:0] = 2'b00; // Input = 0
17      end
18
19  endmodule
20
```

Figure 4: the edit for 4 to 2

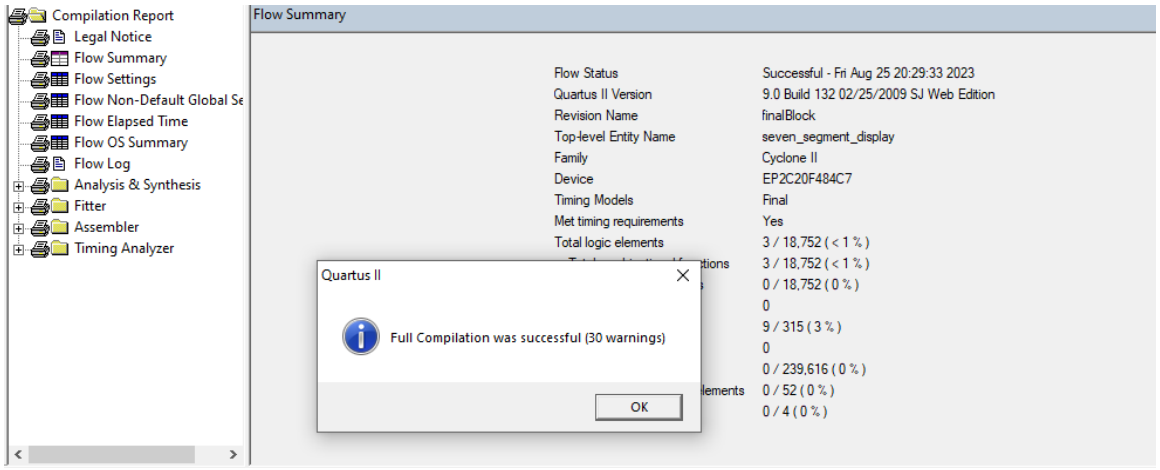


Figure 5: start running

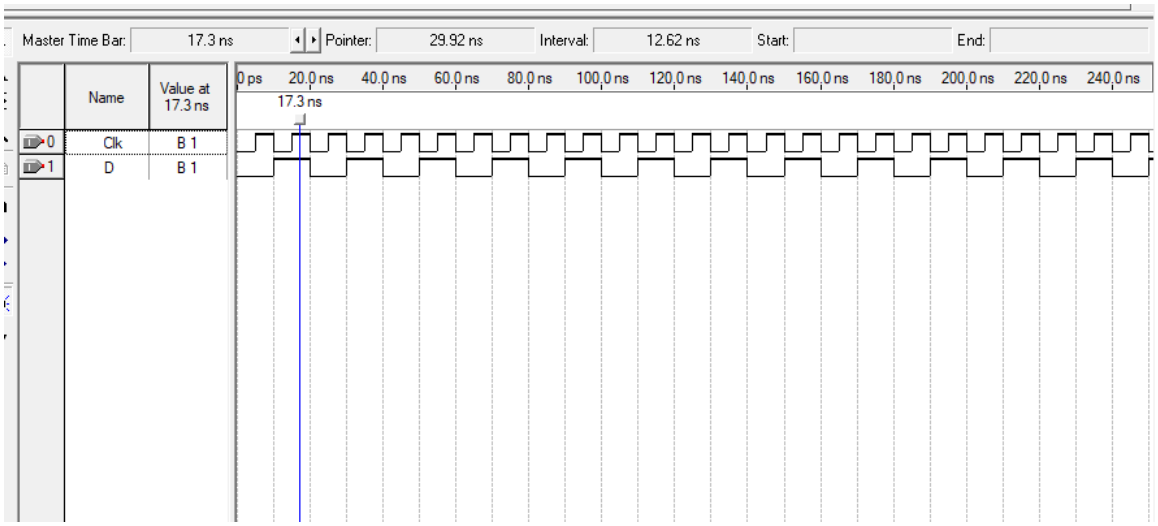


Figure 6: D-flip flop

```
1  module D_Flip_Flop (input D,Clk);
2   reg Q;
3   always @(posedge Clk)
4   begin
5   Q <= D;
6   end
7  endmodule
8
```

Figure 7: D_filp_flop

```
    module mux_2x1 (input sel, input a, input b, output y);
 assign y = sel ? b : a;
endmodule
```

*Figure 8: mux 2*1*

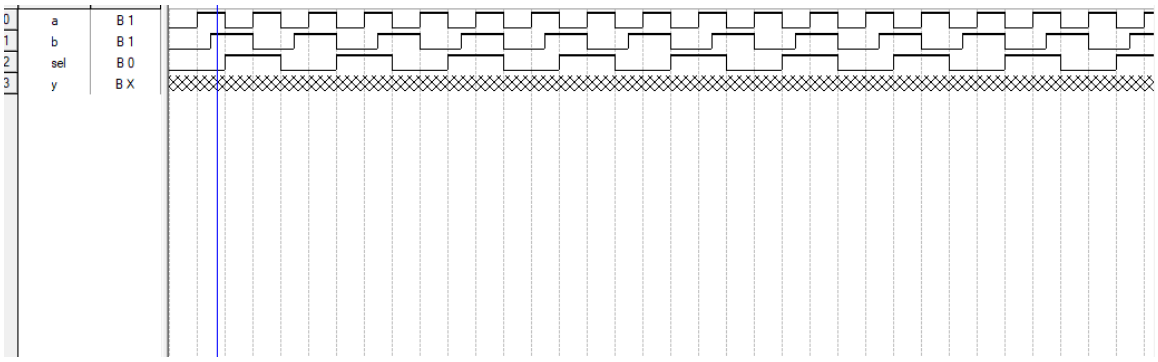


*Figure 9: waveform for 2*1 mux*

```
1  module comparator(out, in);
2      input [6:0] in;
3      output reg out;
4
5      always @(in)
6      begin
7          if (in == 7'b0100100)
8              out = 1'b1;
9          else
10             out = 1'b0;
11     end
12 endmodule
13
```

Figure 10:comparator

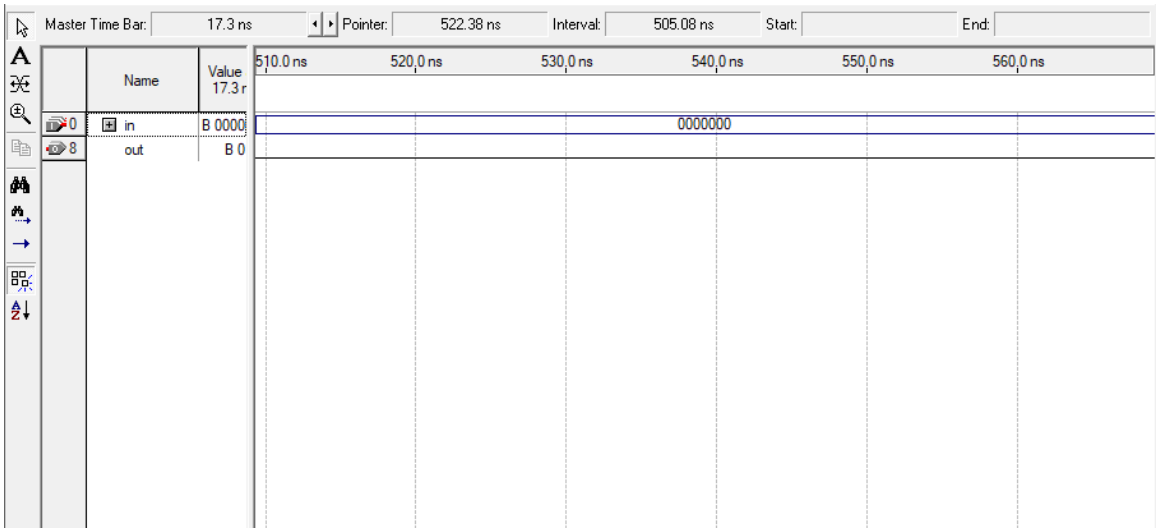


Figure 11: waveform for comparator

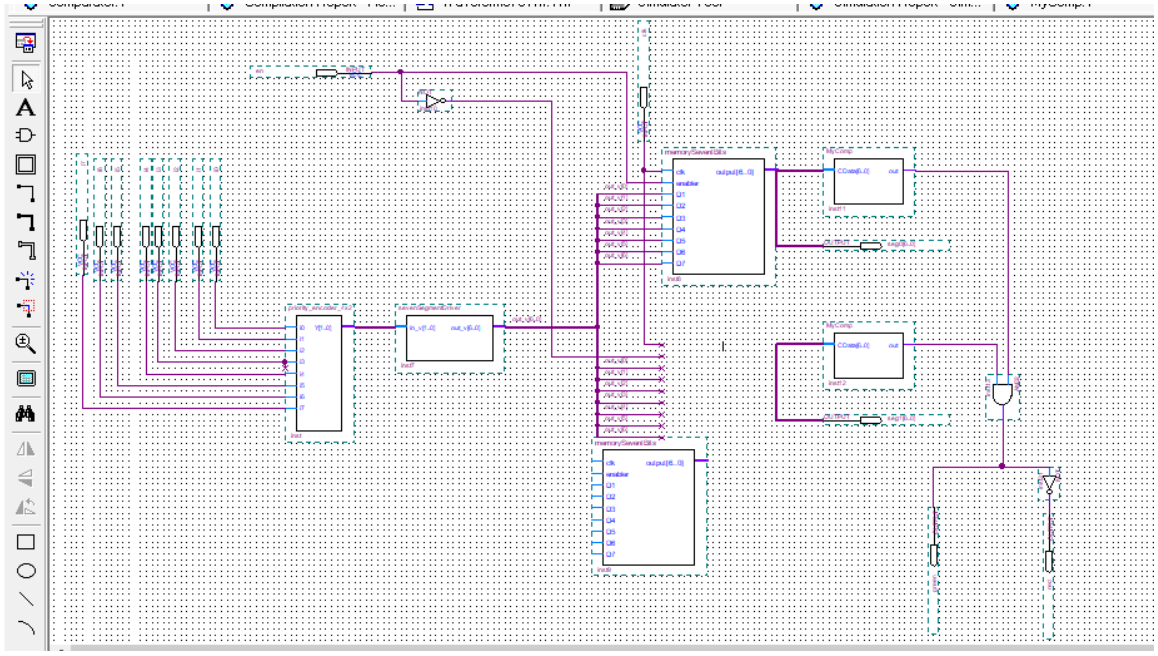


Figure 12: final diagram

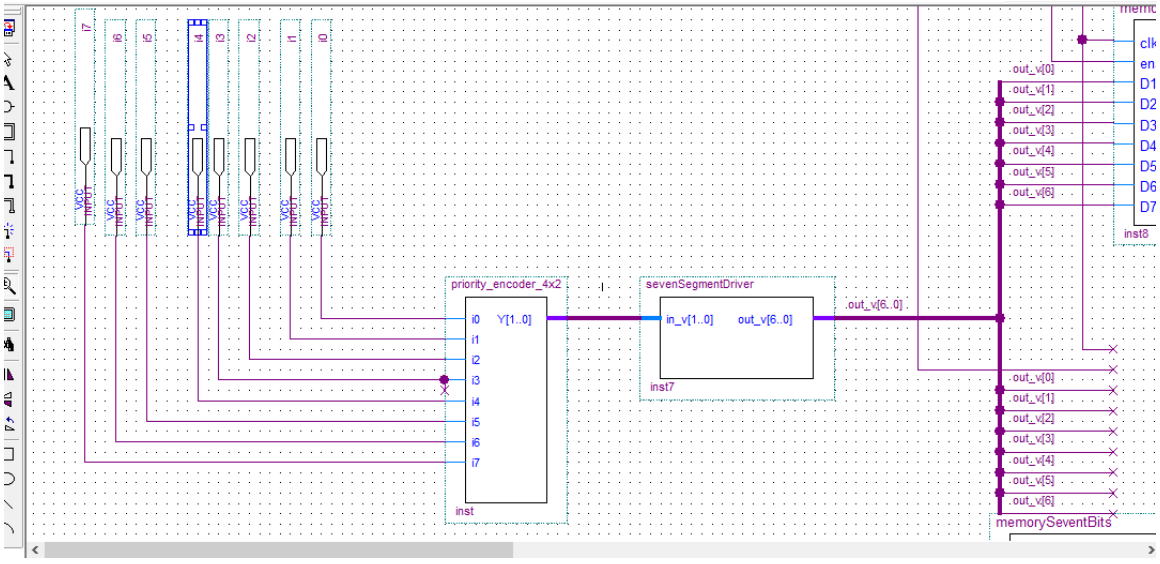
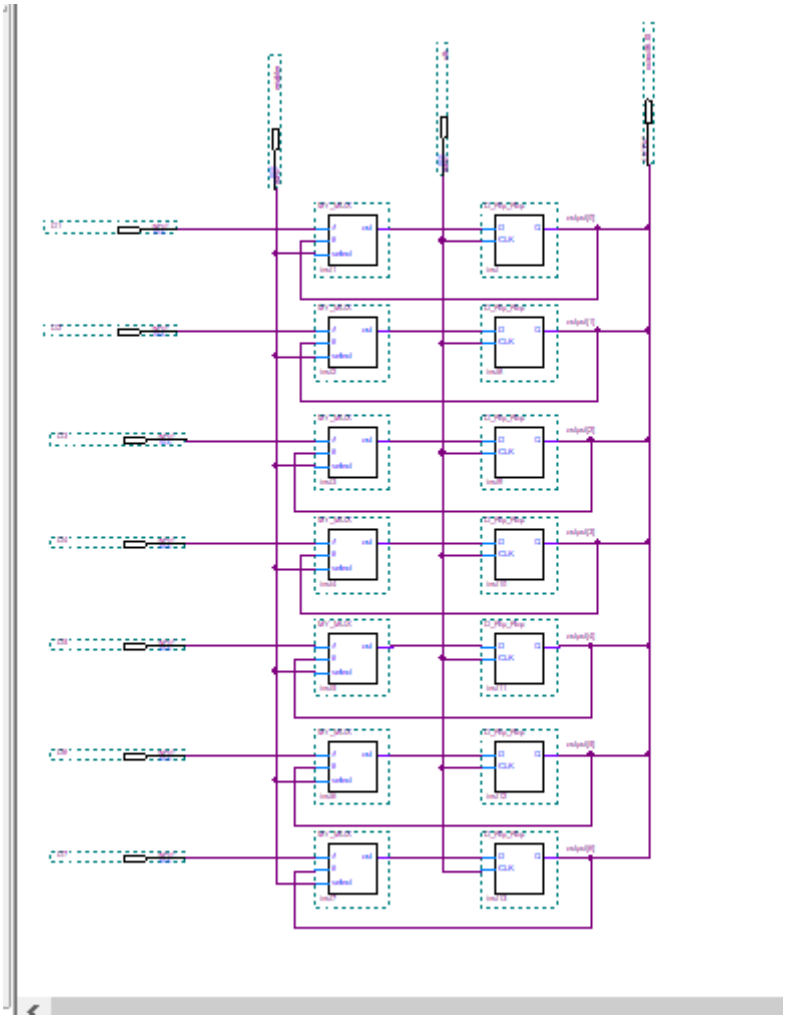
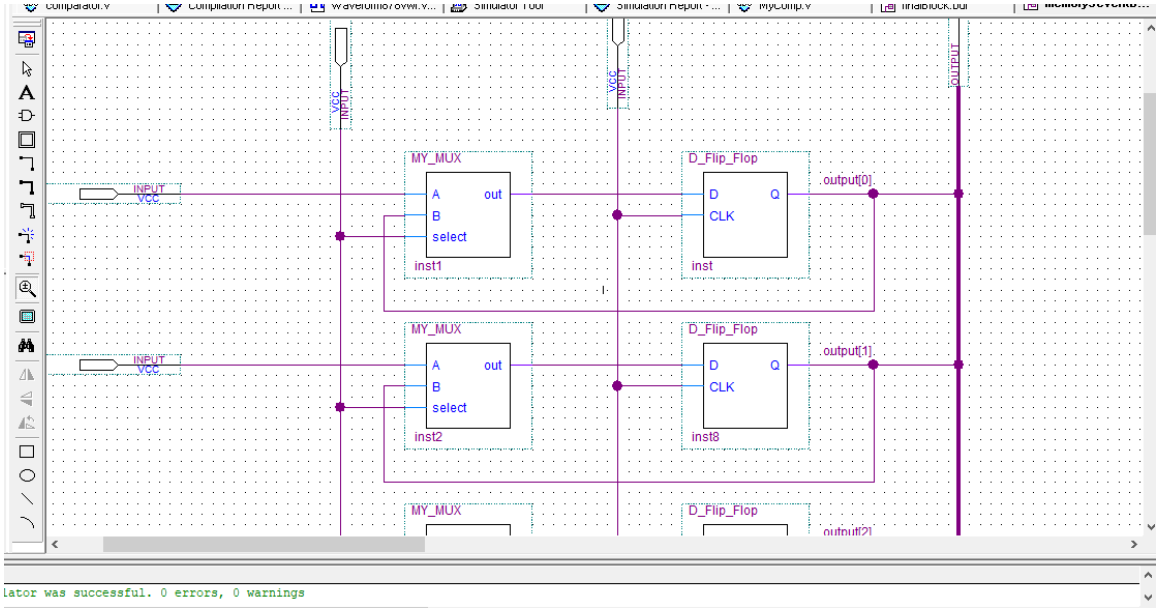


Figure 13: the edit




MY_MUX
A
B
select
out
inst1
D_Flip_Flop
D
CLK
Q
inst
output[0]
INPUT
VCC
MY_MUX
A
B
select
out
inst2
D_Flip_Flop
D
CLK
Q
inst8
output[1]
lator was successful. 0 errors, 0 warnings

In this experiment, we will create a basic security system using the Altera Quartus software, and program it onto an FPGA board. The primary function of this security system is to act as a 2-digit digital lock. Users can input a two-digit number where each digit ranges from 0 to 3. The input is made through a keypad utilizing the 91 switch keys available on our FPGA boards.

Each individual digit is displayed on a 7-segment display. If the combined number displayed on these segments matches the specified code "XX," signifying both digits are the same and between 0 and 3, a green LED indicator will activate, granting access. Conversely, if the entered number does not meet these criteria, a red LED will remain lit, effectively denying access.

In essence, this experiment demonstrates the development of a simple security mechanism using FPGA technology, enhancing our understanding of digital systems and their practical applications.