Brunel University London

重慶郵電大學

CHONGQING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

TERM: Spring, 2022

Module: EE1616 Electronics Workshop

CLASS: 34092102

BRUNEL ID: 2161047

CQUPT ID: 2021215069

NAME: Xukang Liu

TUTOR: Zhipeng Wang

April 19, 2022

Introduction and aims:

This workshop we studied the asynchronous counter and used this principle to create a timer. After finished the circuit diagram, we compiled and programmed it into FPGA board. The final requirement is to realize the timing function by using the FPGA board and the output of LED lights under the artificial control.

Task description:

1. **Design a 4-bit Asynchronous Counter**. Create a new project and a new BDF, add 4 JKFF and connect them.
2. **Design modification to give 4-bit timer.** Create a new project wizard, remember you should select the **Cydone III** family and **EP3C16Q240C8.** Use the symbol TFF to complete this part. Add 4 TFF into the Quartus II and connect them follow the part 1. Note that use VCC as T input, it can keep the input of T always high level. In addition, use NOT gate between the input and the CLK input of TFF. This gate can invert the trigger to a falling egde and let the counter count up.
3. **Add a switch to make the counter more perfect**. Besides the CLK initial input, add a input which name is "SW". And add a AND gate to

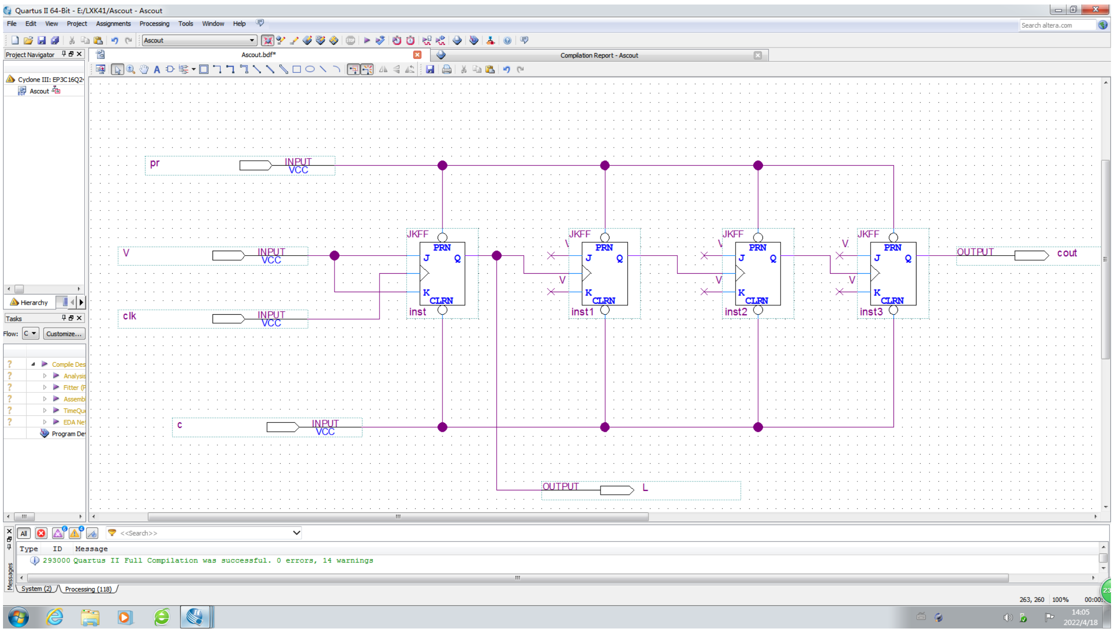
control “CLK” and “SW”. If SW=1, the timer will work. Otherwise, the counter always outputs 0.

Experimental method:

1. Create a new project wizard. Set the Family to “Cydone III” and the available devices to “EP3C16Q240C8”.
2. Add some symbols you need, connect them with a correct order.
3. Compile and create a waveform to simulate it. Check with truth table to make sure that your circuit diagram is no problem.
4. Connect the FPGA board to computer with USB cable, and power on it. Programme your design onto the FPGA board.
5. Find the switches or keys and LED on the FPGA board, check the operation of the timer by choosing the input signals.

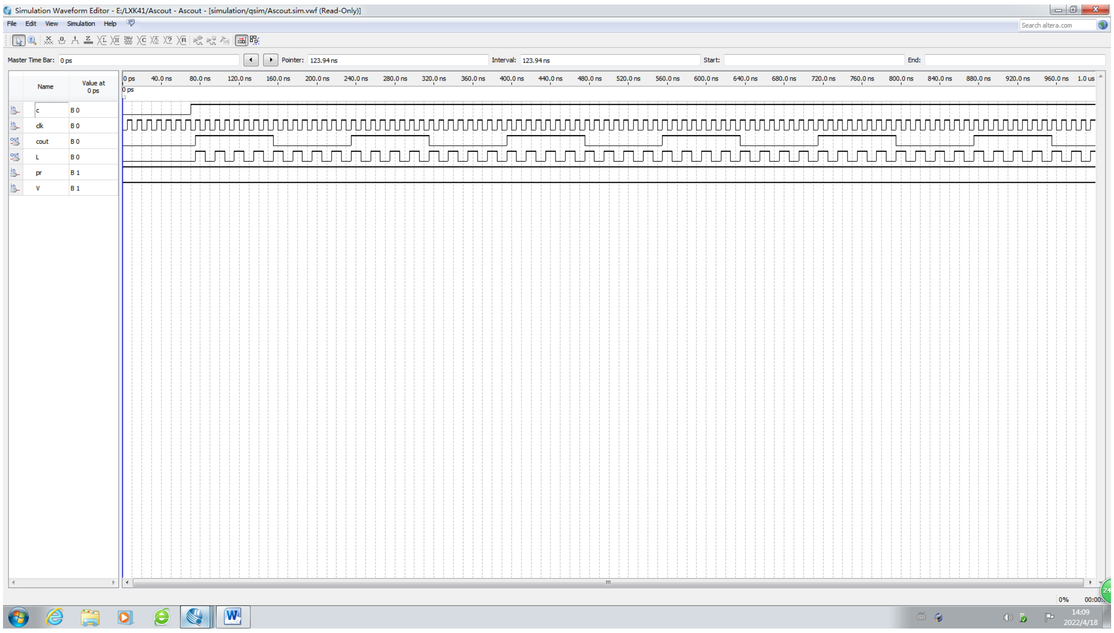
Result and Observations:

This is part 1, I create a asynchronous counter.



Quartus II 64-Bit - E:/LXK41/Ascout - Ascout

```
Type  ID     Message
      293000 Quartus II Full Compilation was successful. 0 errors, 14 warnings
```



Simulation Waveform Editor - E:/LXK41/Ascout - Ascout - [simulation/qsim/Ascout.sim.vwf (Read-Only)]

Master Time Bar: 0 ps Pointer: 123.94 ns Interval: 123.94 ns

Name	Value at 0 ps
c	B 0
clk	B 0
cout	B 0
L	B 0
pr	B 1
V	B 1

Check with the truth table:

Point 1: at 120.0ns

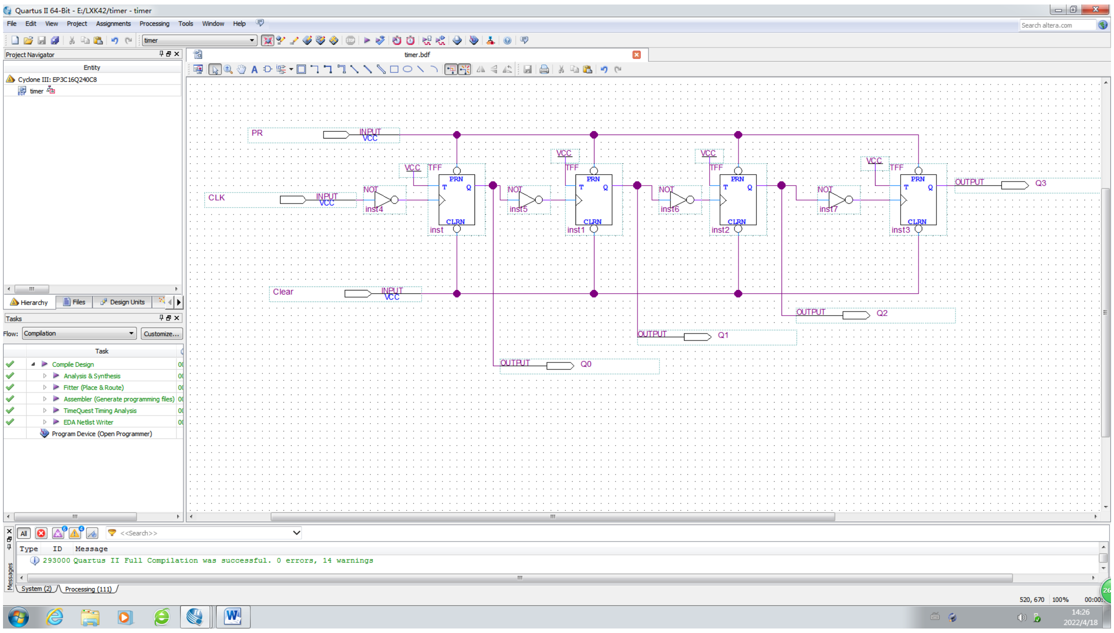
CLK 1→0 V=1 L=1 cout=1 Correct

Point 2: at 200.0ns

CLK 1→0 V=1 L=1 cout=0 Correct

……

Part 2: Preliminary design a 4-bit timer

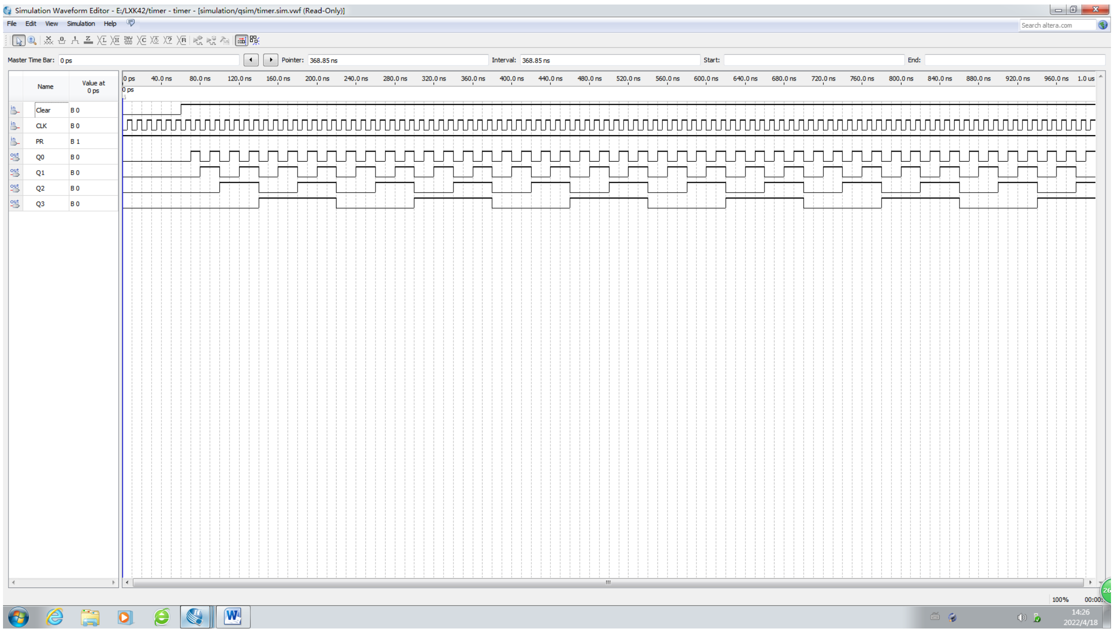

Quartus II 64-Bit - E:/LXK42/timer - timer

Cyclone III: EP3C16Q240C8
timer

Flow: Compilation

Task
✓ Compile Design
✓ Analysis & Synthesis
✓ Fitter (Place & Route)
✓ Assembler (Generate programming files)
✓ TimeQuest Timing Analysis
✓ EDA Netlist Writer
Program Device (Open Programmer)

```
Type  ID     Message
      293000 Quartus II Full Compilation was successful. 0 errors, 14 warnings
```


Simulation Waveform Editor - E:/LXK42/timer - timer - [simulation/qsim/timer.sim.vwf (Read-Only)]

Master Time Bar: 0 ps   Pointer: 368.85 ns   Interval: 368.85 ns

Name	Value at 0 ps
Clear	B 0
CLK	B 0
PR	B 1
Q0	B 0
Q1	B 0
Q2	B 0
Q3	B 0

Check with the truth table:

Point 1: at 120.0ns

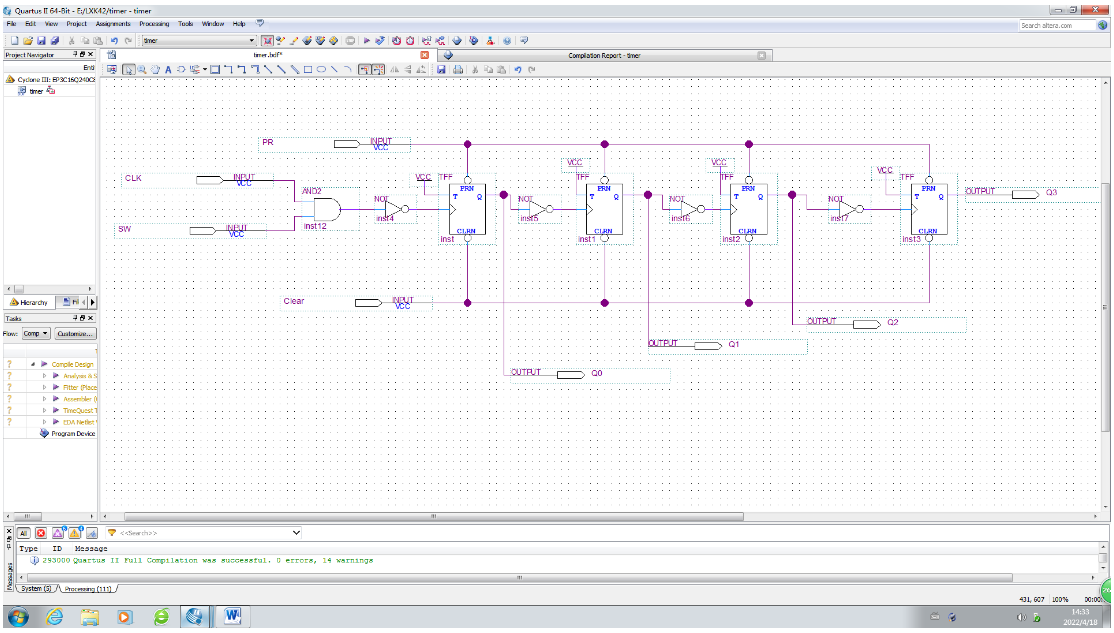
CLK 1→0　Q0 1→0　Q1 0→1　Q2=1　Q3=0　Correct

Point 2:at 190.0ns

CLK 1→0　Q0 0→1　Q1=0　Q2=1　Q3=1　Correct

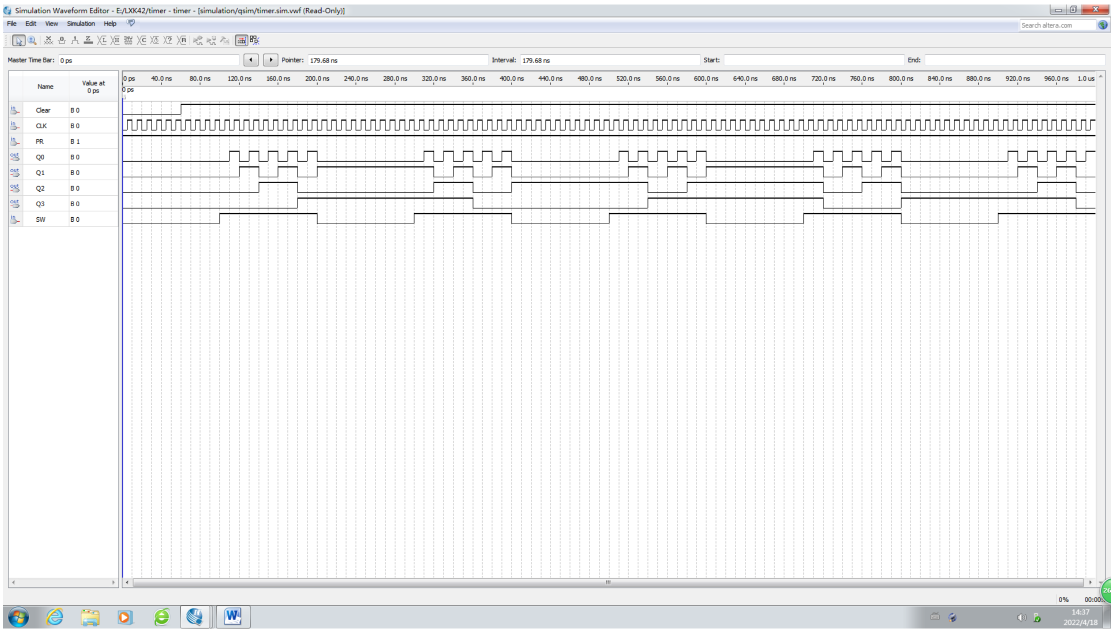
……

Part 3: Devise the timer, add a switch to make it perfect



Type ID Message

293000 Quartus II Full Compilation was successful. 0 errors, 14 warnings



Simulation Waveform Editor - E:/LXK42/timer - timer - [simulation/qsim/timer.sim.vwf (Read-Only)]

Master Time Bar: 0 ps

Pointer: 179.68 ns

Interval: 179.68 ns

Start:

End:

Name	Value at 0 ps
Clear	B 0
CLK	B 0
PR	B 1
Q0	B 0
Q1	B 0
Q2	B 0
Q3	B 0
SW	B 0

Check with the truth table:

Point 1: at 120.0ns

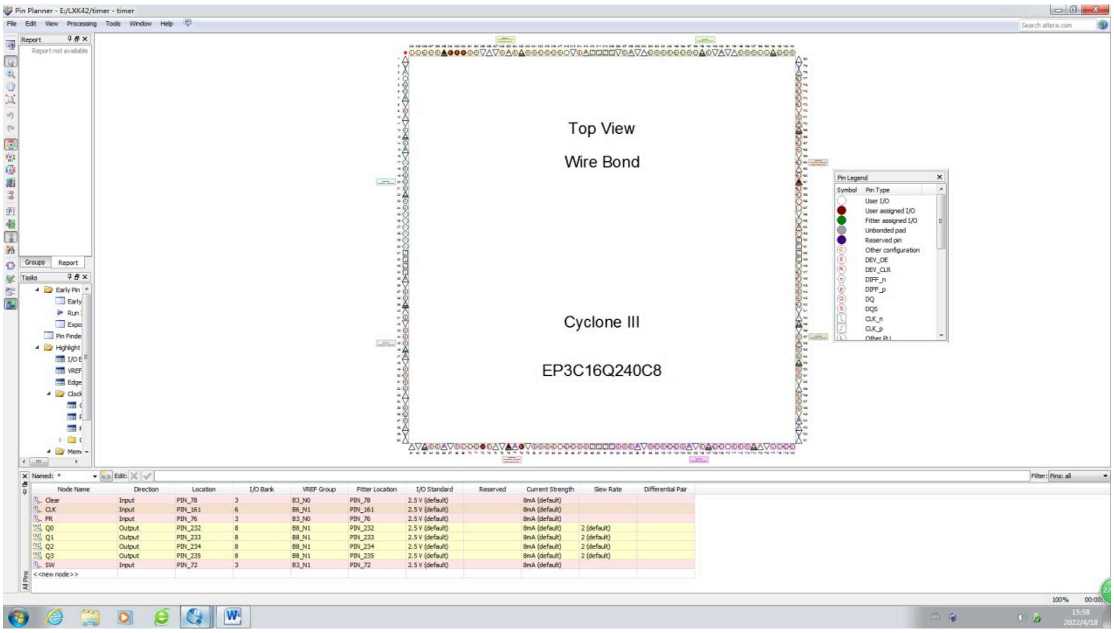
CLK 1→0 SW=1 Q0 1→0 Q1 0→1 Q3=0 Correct

Point 2: at 240.0ns

CLK 1→0 SW=0 Q0=0 Q1=0 Q2=0 Q3=0 Correct

……

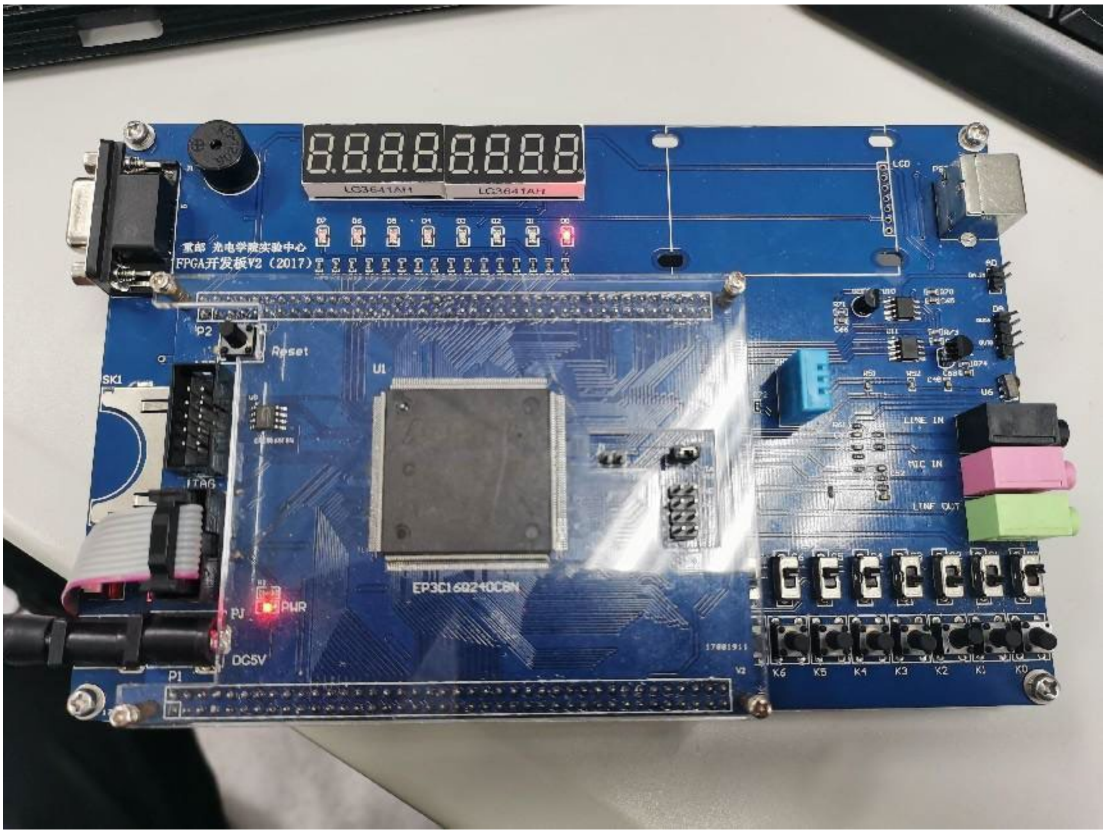
Then connect the FPGA board, choose appropriate pins with FPGA board.



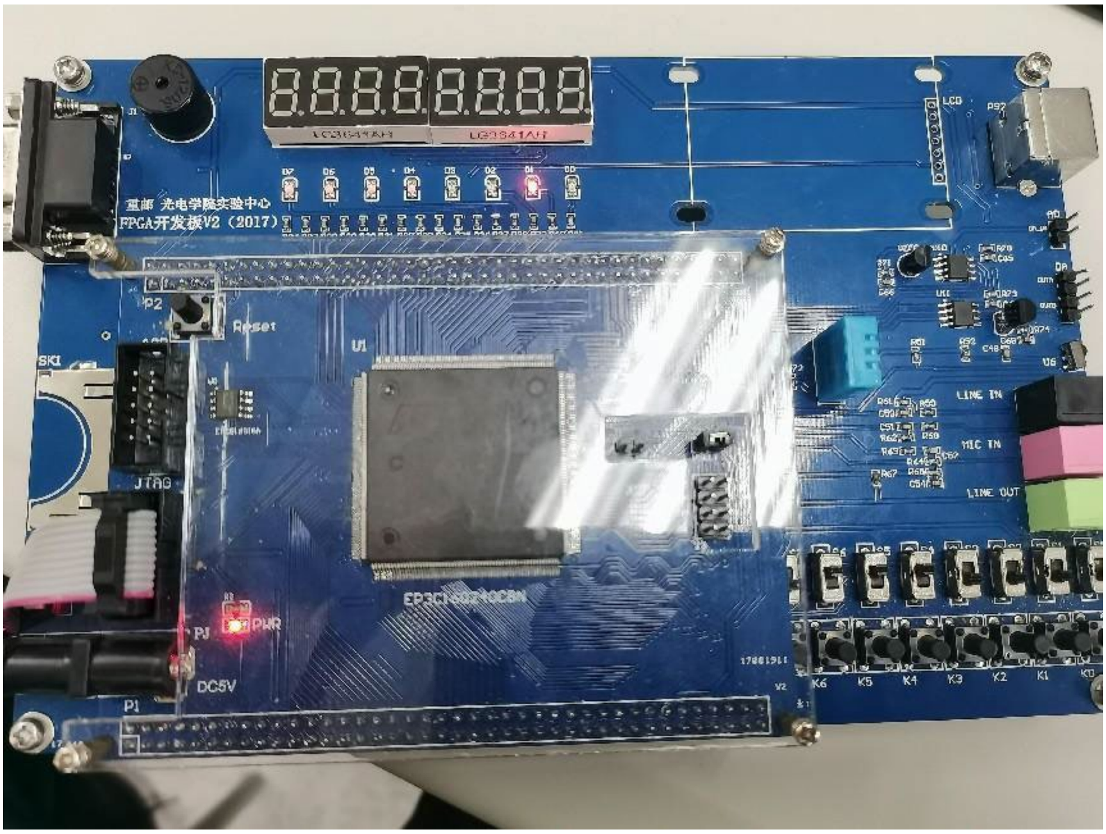
Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair
Clear	Input	PIN_78	3	B3_N0	PIN_78	2.5 V (default)		8mA (default)		
CLK	Input	PIN_161	6	B6_N1	PIN_161	2.5 V (default)		8mA (default)		
PR	Input	PIN_76	3	B3_N0	PIN_76	2.5 V (default)		8mA (default)		
Q0	Output	PIN_232	8	B8_N1	PIN_232	2.5 V (default)		8mA (default)	2 (default)	
Q1	Output	PIN_233	8	B8_N1	PIN_233	2.5 V (default)		8mA (default)	2 (default)	
Q2	Output	PIN_234	8	B8_N1	PIN_234	2.5 V (default)		8mA (default)	2 (default)	
Q3	Output	PIN_235	8	B8_N1	PIN_235	2.5 V (default)		8mA (default)	2 (default)	
SW	Input	PIN_72	3	B3_N1	PIN_72	2.5 V (default)		8mA (default)		
<<new node>>										

Finally, the timer is completed. This is the final result.

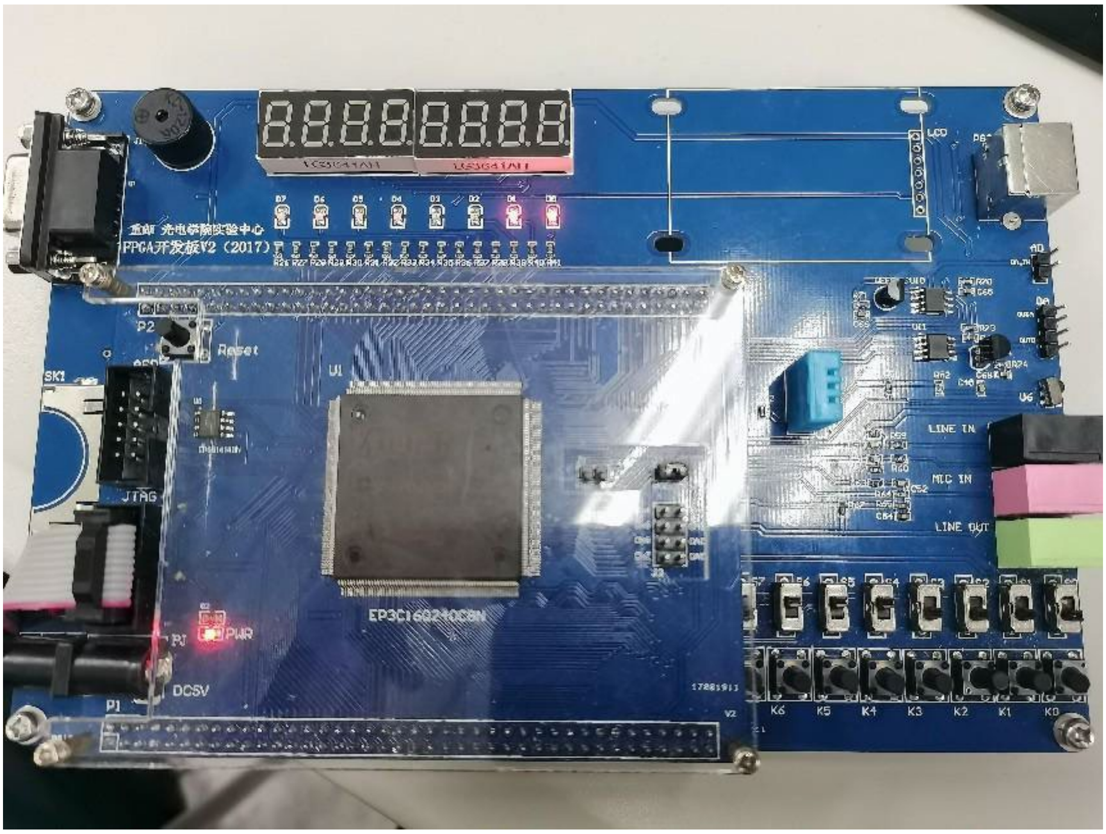
(1) Turn the preset, clear and SW to high level, change the switch1 which control the clock signal. Then we can find that D0 is on. It means this time is “0001”.



(2) Switch “switch1” to high level and then back to low level. At this moment, the clock signal is falling edge, so we can see that D1 is on and D0 is off. It means the time is “0010”.



(3) Same as the second step, we can see that D1 is on and D0 is on. It means the time is “0011”.



Conclusion:

This workshop I use Quartus II to design a timer and use FPGA board to realize my design. I further learn the principle of asynchronous binary counter. In this seminar, I create a asynchronous counter to be used as a timer. Because in this software, TFF is triggered at rising edge, if I don't change this, the counter will count down. So I to add a NOT gate to invert it to falling edge to let it count up. From the result, I realize the timing function by controlling the switch on the FPGA board. In the process of compiling to FPGA board, I also find a difficult. Because the frequency of clock signal is 50MHZ, so if I use it the timer will work vary fast, so that I can’t find the result. So I changed the pin of the clock signal to an on switch on the FPGA board. Then I can manually control the clock signal for better observation.

Overall, in these four integrated circuit design courses, I have a preliminary understanding of Quartus II and Altera device. Its a good software for digital circuit design. Moreover, I learned how to design a circuit, simulate and programme it onto the FPGA board. This is very helpful for our future work.