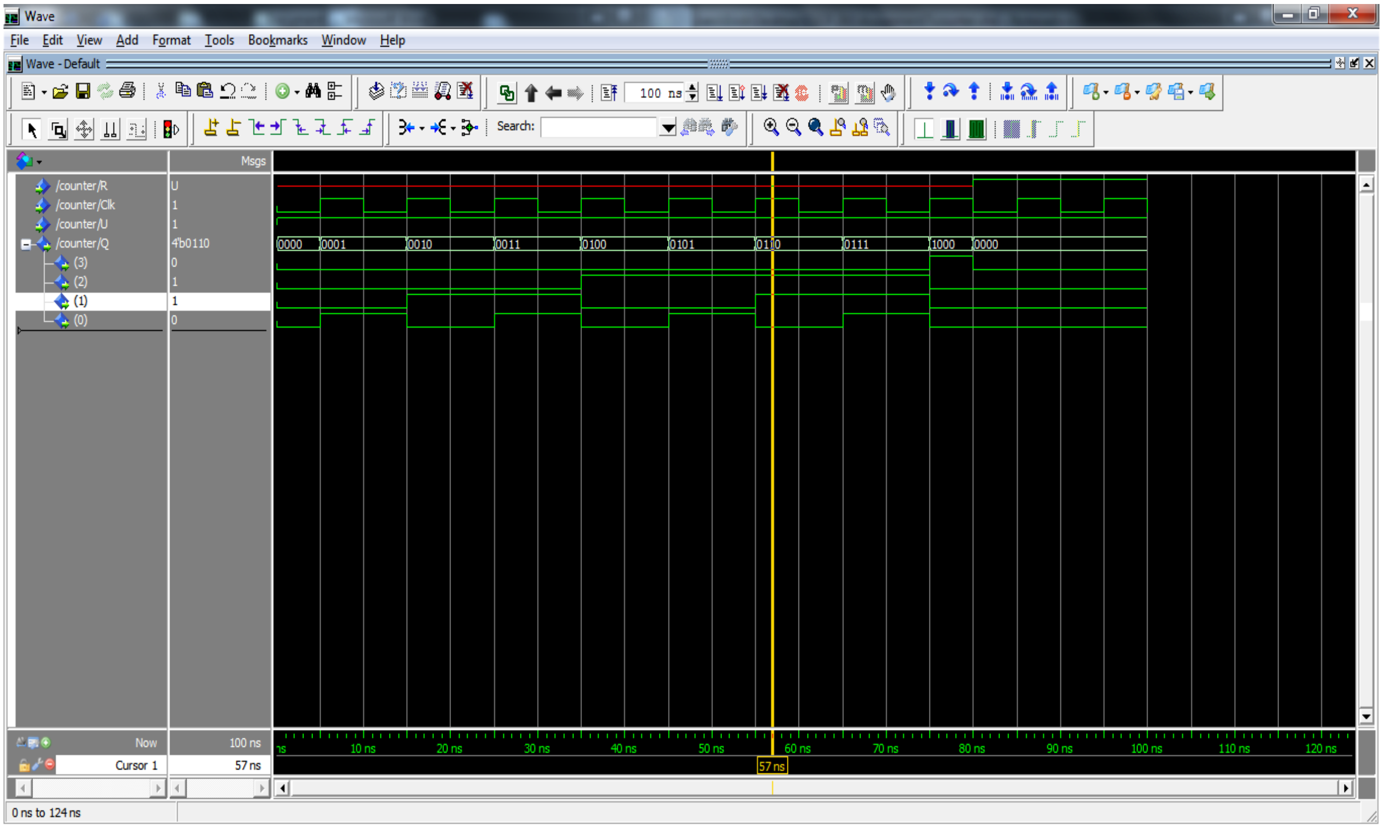
Part 1:

- In part 1 of assignment 7 the task was to model the behavior of a 4-bit up/down counter with reset using VHDL. The design approached to solve the task was first, add the IEEE library and the std logic library as well as the std unsigned library which is used to performed unsigned arithmetic operations. Then the entity of the counter was declared as counter with input signals R (reset), Clk (clock), and U (input). And with the output signal Q which is a vector output signal of width of 4 bits. Afterwards, the architecture declared as behavior of the counter was declared, with a process using the entity declared at the beginning. Also a new internal vector signal variable was declared named Q_val of width 4-bits with an initial value of 0. Secondly, the behavior of the counter was described by first declaring the reset behavior, when signal is 1 the internal signal Q_val will be set to 0. Then if the select signal input U is 1 then the counter will count up and if the input is 0 then the counter will count down at the rising edge of the clock. After that the output value of the entity Q was set equal to the internal output value Q_val and the process as well as the definition of the architecture was finalize.
- Snapshot of part one, at 80 ns the reset was activated and the output was set to 0, which indicates that the reset work and the counter is counting up

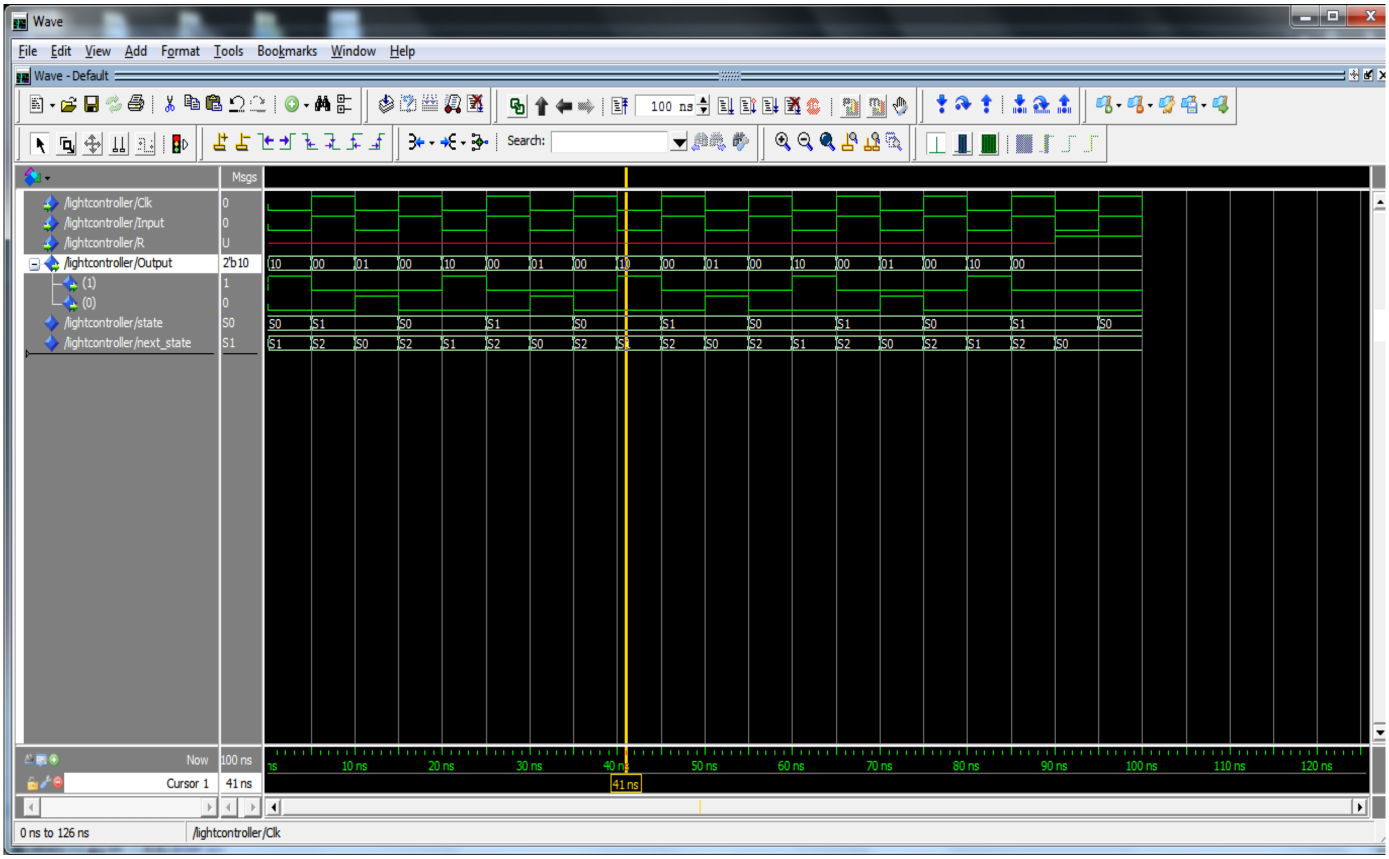


Signal	Msgs
/counter/R	U
/counter/Clk	1
/counter/U	1
/counter/Q	4'b0110
(3)	0
(2)	1
(1)	1
(0)	0

Now 100 ns
Cursor 1 57 ns
0 ns to 124 ns

Part 2.

- In part 2 of the assignment the task was to model the behavior of a traffic-light controller with reset. The traffic light controller can be interpreted as a Mealy state machine. The design approached to solve the task was first, add the IEEE library and the std logic library as well as the std unsigned library which is used to performed unsigned arithmetic operations. Then the entity of the light controller was declared with input signals Clk (clock), Input, and R (reset), and also an output vector signal Output. Then the architecture of the light controller was declared with internal signals S0, S1 and S2 of type statetype which represent the state of the light controller, and internal signals state which represents the present state of the light controller, and a next_state variable which represents the next state. The initial state of the controller is set to the S0 state. Then the process is named as processOne_Nextstate_and_Output, with signals state, Input and R. Afterwards, the behavior of the reset signal is described and when R is 1 then the output should be 0 and the next state will be the initial state. Else if the input is 0 then the output will be 10 (2) and the next state will be S1, else if the input is 1 the output will be 00 and the next state will be S2. The rest of the input, output and next state behavior is described as described in the assignment description for the next two states. Then the case process is ended along with the process definition for the cases. Moreover, a second process is described which defines the behavior of the clock and defines each state to be processed every rising edge of the clock, and the internal input signal state is set to the next state. And the process is ended as well as the architecture of the machine.
- Snapshot of part two, at 90 ns the reset was activated and the output was set to 0 as well as the next state was reset to the initial state S0, which indicates that the reset work and the light controller works. Because for each input the next state and the output change one every clock rising edge.


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
100 ns
Search:
Msgs
10 00 01 00 10 00 01 00 10 00 01 00 10 00 01 00 10 00
S0 S1 S0 S1 S0 S1 S0 S1 S0 S1 S0
S1 S2 S0 S2 S1 S2 S0 S2 S1 S2 S0 S2 S1 S2 S0 S2 S1 S2 S0
ns 10 ns 20 ns 30 ns 40 ns 50 ns 60 ns 70 ns 80 ns 90 ns 100 ns 110 ns 120 ns
41 ns

```
/lightcontroller/Clk         0
/lightcontroller/Input       0
/lightcontroller/R           U
/lightcontroller/Output      2'b10
  (1)                        1
  (0)                        0
/lightcontroller/state       S0
/lightcontroller/next_state  S1
Now      100 ns
Cursor 1 41 ns
0 ns to 126 ns    /lightcontroller/Clk
```

- In conclusion, in both parts the design work as desired and the output was the same as in the description of the assignment. In the first part the counter counted up if the input U was 1 and down if the input of U was 0. In part two every input of determined the state of the machine.