Lab 02: Counters (Sep 26, 2019)

Submission deadlines:

Source Code:	18:30, Oct 1, 2019
Report	23:59, Oct 6, 2019

Objective

- ➢ Be familiar with different counter designs in Verilog
- ➢ Bonus: LFSR (linear-feedback shift register) counter

Action Items

1. (lab2_1.v) Design a **positive-edge-**triggered-clock 4-bit counter.

- ♦ If **rst_n** == 1’b0: the counter resets its output value to 4’b0000
- ♦ If **en** == 1’b0: the counter should hold its current output value
- ♦ If **en** == 1’b1:
  - ● If in == 1’b0:
    - ■ If **dir** == 1’b1: the output value should increase by 1 at every **clk** triggered
    - ■ If **dir** == 1’b0: the output value should decrease by 1 at every **clk** triggered
    - ■ When your output value is 4’b1111 and dir == 1’b1, the next output value will be 4’b0000.
    - ■ When your output value is 4’b0000 and dir == 1’b0, the next output value will be 4’b1111.
  - ● If in == 1’b1: set the value equal to the data (input).

You have to use the following template for your design.

```
module lab2_1(clk, rst_n, en, dir, in, data, out);
    input clk, rst_n, en, dir, in;
    input [3:0] data;
    output [3:0] out;

    // add your design here

endmodule
```

2. (lab2_1_t.v) You have to create a testbench, which named lab2_1_t.v by yourself to verify your design. In your testbench, you should try the possible input combinations, and check if your design is correct or not. During the demo, TA will check your waveform and ask you some questions about how you build your testbench.

3. (lab2_2.v) Design a **negative-edge**-triggered-clock 2-digit gray-code counter. TAs supply the testbench lab2_2_t.v for this lab. You have to use the following template for your design. You have to follow the hint to design your circuit.

```
module lab2_2(clk, rst_n, en, dir, gray, cout);
    input clk, rst_n, en, dir;
    output [7:0] gray;
    output cout;

    // add your design here

endmodule
```

The table below shows the relationship between binary code and gray code.

Decimal Value	**Binary code**	**Gray code**	**Decimal value**	**Binary code**	**Gray code**
0	0000	0000	**8**	1000	1100
1	0001	0001	**9**	1001	1101
2	0010	0011	**10**	1010	1111
3	0011	0010	**11**	1011	1110
4	0100	0110	**12**	1100	1010
5	0101	0111	**13**	1101	1011
6	0110	0101	**14**	1110	1001
7	0111	0100	**15**	1111	1000

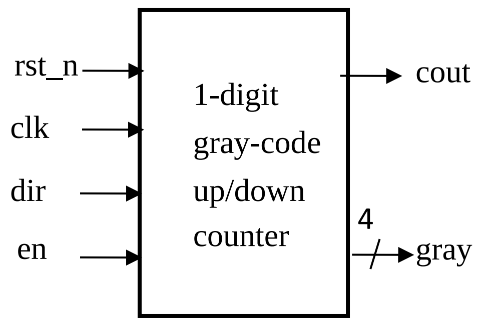
For example, 17(dec) = 11(hex) = 0001 0001(gray-code)

Reference: https://en.wikipedia.org/wiki/Gray_code

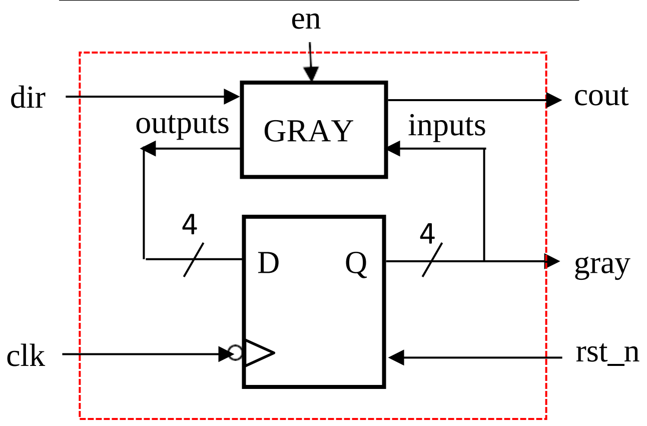
- If **rst_n** == 1'b0: the counter reset its output value to 8'h0.
- If **en** == 1'b0: the counter should hold its current output value.
- If **en** == 1'b1:
  - If **dir** == 1'b1: the output value should increase by 1 (in the gray-code order) at every **clk** triggered.
  - If **dir** == 1'b0: the output value should decrease by 1 (in the gray-code order) at every **clk** triggered.
  - The output **cout**=1'b1 when ① counting to 15 (binary code 1111, gray code 1000) with **dir** == 1 or ② counting to 0 (binary code 0000, gray code 0000) with **dir** == 0

Hint: You can first design a 1-digit gray-code counter and then use it to design the 2-digit gray-code counter.

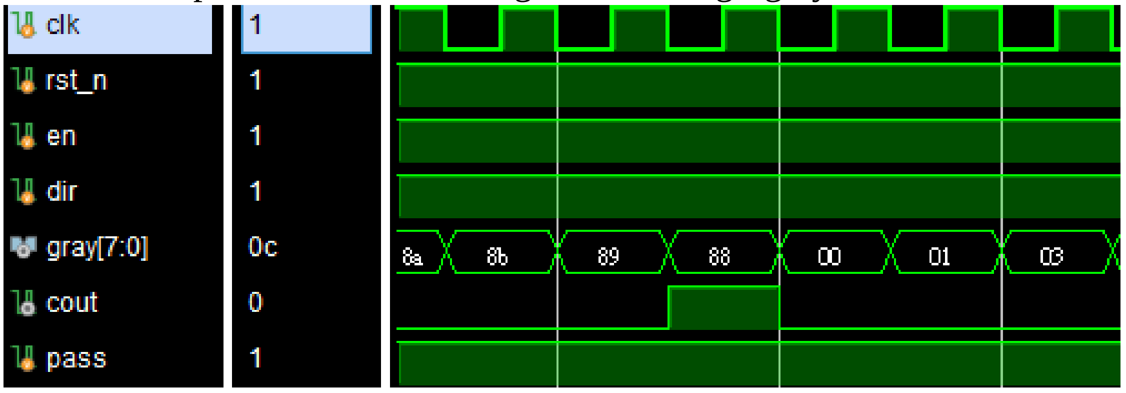
Block diagram of
1-digit gray code counter



1-digit gray code counter
in further details



The relationship of **cout** and other signals for 2-digit gray code counter.



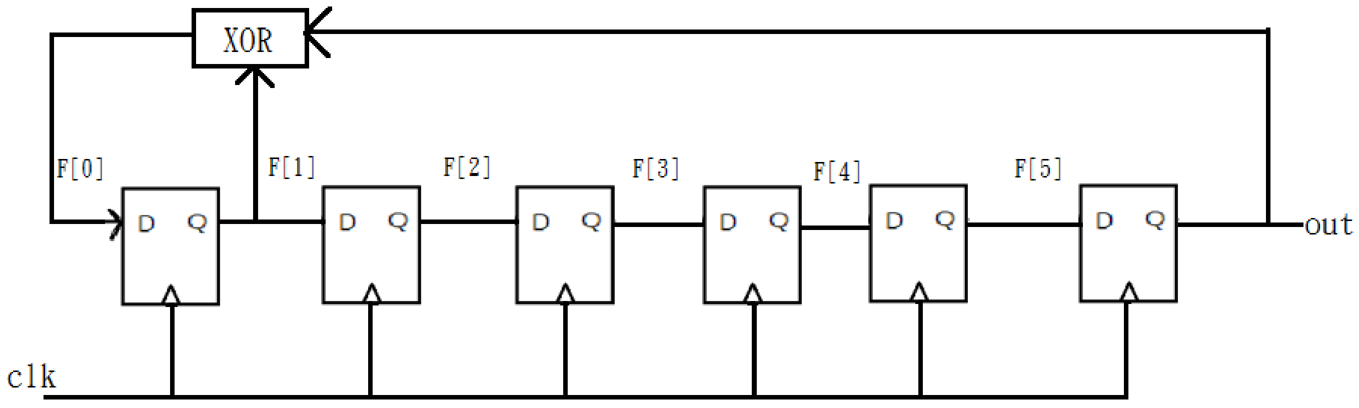
4. **Bonus:** (lab2_3.v, lab2_3_t.v) Design a 6-bit Linear feedback shift register (LFSR). You have to create a testbench, which named lab2_3_t.v by yourself to verify your design. In your testbench, you should try the possible input combinations, and check if your design is correct or not. During the demo, TA will check your waveform and ask you some questions about how you build your testbench. You have to use the following template for your design.

```
module lab2_3 (clk, rst_n, out);
    input clk, rst_n;
    output out;
    reg [5:0] F;

    // add your design here

endmodule
```

You should use **positive-edge**-triggered clock. You **must** design LFSR based on the logic diagram below.
If **rst_n** == 1'b0, reset **F** to 6'b000001. Observe the output **out** and 6-bit **F** for a number of cycles. Is there any deterministic order of the output sequence?
(Hint: https://www.eetimes.com/document.asp?doc_id=1274550)



Attention

✓ You can add a $monitor in your testbench to show all the information of your inputs and outputs when you run the simulation.
✓ You may have to change your runtime (ns) in "Simulation Settings" to fit your testbench settings before you run the simulation.
✓ You should hand in lab2_1.v, lab2_1_t.v, lab2_2.v, (and lab2_3.v, lab2_3_t.v, optional) and any other design files you create (you should describe it clearly in the report if you need additional design files).
(Upload each source file separately! DO NOT hand in a compressed ZIP file!)
✓ You should also hand in your report as lab02_report_StudentID.pdf (i.e., lab02_report_107456789.pdf).
✓ You should be able to answer questions of this lab from TA during the demo.