# Lab 03: Clock Divider and LED Controller

# (Oct 3, 2019)

Submission deadlines:

| | |
|---|---|
| Source Code: | 18:30, Oct 8, 2019 |
| Report | 23:59, Oct 13, 2019 |

## Objective

Get familiar with the clock divider and LED control on the FPGA demo board.

## Action Items

1. (20%) Write a Verilog module for the clock divider that divides the frequency of the input clock by $2^{26}$ to get the output clock.

You should use the following template for your design and name the file as "clock_divider.v". (The parameter statement is mandatory.)

```
module clock_divider(clk, clk_div);
    parameter n = 26;
    input clk;
    output clk_div;

    // add your design here

endmodule
```

2. (20%) Write a Verilog module of the LED Controller which is synchronous with the clock whose frequency is obtained by dividing the frequency of Basys3's clock, 100MHz, by $2^{26}$. Also, program your LED Controller to the

FPGA demo board. You should use the clock divider you design in Action Item 1.

**Here are the inputs and outputs constraints:**

- ✓ When the rst is 1, the left-most LED (LD15) is ON, and others are OFF.
- ✓ When the en is 1 and dir is 1, the LED will begin to shift (to turn on and off one by one) from the right to the left synchronized to the clock.
- ✓ When the en is 1 and dir is 0, the LED will begin to shift (to turn on and off one by one) from the left to the right synchronized to the clock.
- ✓ When the en is 0, hold the LEDs unchanged.

**IO Connection:**

| | |
|---|---|
| clk | connected to W5 |
| rst | connected to W16 |
| en | connected to V17 |
| dir | connected to V16 |
| led | connected to LD15-LD0 |

You should use the following template for your design and name the file as " lab03_1.v".

Demo: https://youtu.be/dkAfj2vEp5o

```
module lab03_1(clk, rst, en, dir, led);
    input clk;
    input rst;
    input en;
    input dir;
    output[15:0] led;

    //add your design here
endmodule
```

3. (30%) Write a Verilog module of the LED Controller which is synchronous with the clock whose frequency is obtained by dividing the frequency of Basys3's clock, 100MHz, by $2^{23}$ or $2^{26}$.

   There are two LED runners, called Mr. 1 and Mr. 3.

   ✓ Mr. 1 and Mr. 3 race on the 16 LEDs of the FPGA Demo board.

   ✓ Represent Mr. 1 by one single LED.

   ✓ Represent Mr. 3 by three consecutive LEDs.
   (His position is determined by the middle LED.)

   ✓ When the rst is 1, set Mr. 1 and Mr. 3's positions to LD15 and LD14, respectively.
   i.e. Initial state: (●: LED on, ○: LED off)
   Mr. 1: (LD15) ●○○○○○○○○○○○○○○○ (LD0)
   Mr. 3: (LD15) ●●●○○○○○○○○○○○○○ (LD0)

   ✓ When the en is 1 and dir is 1, Mr. 1 and Mr. 3 will begin to shift from right to left and Mr. 3 runs at the clock rate of (100 MHz / $2^{23}$); Mr. 1 runs at the clock rate of (100 MHz / $2^{26}$), respectively.

   ✓ When the en is 1 and dir is 0, Mr. 1 and Mr. 3 will begin to shift from left to right. Mr. 3 runs at the clock rate of (100 MHz / $2^{26}$); Mr. 1 runs at the clock rate of (100 MHz / $2^{23}$), respectively.

   ✓ When the en is 0, Mr. 1 and Mr. 3 hold at their current positions.

**IO Connection:**

| | |
|---|---|
| clk | connected to W5 |
| rst | connected to W16 |
| en | connected to V17 |
| dir | connected to V16 |
| led | connected to LD15-LD0 |

You should use the following template for your design and name the file as "lab03_2.v".

Demo: https://youtu.be/nRBcSAGV8Q4

```
module lab03_2(clk, rst, en, dir, led);
    input clk;
    input rst;
    input en;
    input dir;
    output[15:0] led;

    //add your design here

endmodule
```

4. (30%) Write a Verilog module of the LED Controller which is synchronous at the clock rates of (100MHz / $2^{23}$) or (100MHz / $2^{26}$), dividing from Basys3's 100MHz clock.

   Here are two LED runners, called Mr. 1 and Mr. 3 (similar to lab03_2).

   ✓ When the rst is 1, set Mr. 1 and Mr. 3's positions to LD15 and LD14, respectively.
   i.e. Initial state: (●: LED on, ○: LED off)
   Mr. 1: (LD15) ●○○○○○○○○○○○○○○○ (LD0)
   Mr. 3: (LD15) ●●●○○○○○○○○○○○○○ (LD0)

   ✓ When the en is 1, Mr. 1 and Mr. 3 will begin to shift from left to right and Mr. 3 runs synchronized to the clock rate of (100 MHz / $2^{26}$); Mr. 1 runs at the clock rate of (100 MHz / $2^{23}$), respectively.

   ✓ When they collide with each other, Mr. 1 will change its direction.
   ➢ Collision Situation 1: Mr. 1 touches the tail of Mr.3
   e.g.
   Mr. 1: (LD15) ○○  ●○○○○○○○○○○○○○(LD0)
   Mr. 3: (LD15) ○○○●●●○○○○○○○○○○○○ (LD0)
   ➢ Collision Situation 2: Mr. 1 touches the middle of Mr.3
   e.g.
   Mr. 1: (LD15) ○○○○○○○●○○○○○○○○ (LD0)
   Mr. 3: (LD15) ○○○○○○●●●○○○○○○○ (LD0)

**(Hint: At what condition will Collision Situation 2 happen? Given the different clock rates of the two runners.)**

➢ Collision Situation 3: Mr. 1 touches the head of Mr.3

e.g.

Mr. 1: (LD15) ○○○○○●  ○○○○○○○○○ (LD0)

Mr. 3: (LD15) ○○○●●●○○○○○○○○○ (LD0)

✓ When the en is 0, Mr. 1 and Mr. 3 will hold at their current position.

**IO Connection:**

| | |
|---|---|
| clk | connected to W5 |
| rst | connected to W16 |
| en | connected to V17 |
| dir | connected to V16 |
| led | connected to LD15-LD0 |

You should use the following template for your design and name the file as " lab03_3.v".

Demo: https://youtu.be/KKwR2Kcw_wk

```
module lab03_3(clk, rst, en, led);
    input clk;
    input rst;
    input en;
    output[15:0] led;

    //add your design here

endmodule
```

**Note:**

1. You may have to change your runtime (ns) in “Simulation Settings” to fit the testbench settings before the simulation.
2. You should hand in clock_divider.v, lab03_1.v, lab03_2.v, lab03_3.v, and any other design files you create.
   (1) You should describe it clearly in the report if you need additional design files.
   (2) **DO NOT hand in a compressed ZIP or RAR file!**
3. You should also hand in your report as lab03_report_StudentID.pdf (i.e., lab03_report_107080001.pdf).
4. You will need to answer questions about this lab by TA during the demo.
5. You need to generate the bitstream files before the lab demo. Prepare three separate bitstream files for lab03_1, lab03_2, and lab03_3, respectively, to smooth the demo process.