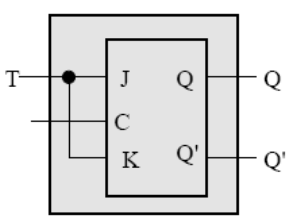
Laboratory 5 Design a Frequency Divider and Programming a FPGA

For your report:

- The problem written in English
- The flowchart or function table to solve the problem if it is necessary
- The design entry included (VHDL and Schematic)
- The RTL Viewer has to be included
- The simulation result for designed digital component
- The analysis for the simulation
- The pin assignment- the table for assigning the circuit inputs and outputs to specific pins on the FPGA
- The configuration for the FPGA Device (JTAG)
- The test table you designed to record and verify the designed circuit on hardware
- Picture taken for your test if it is necessary
- The conclusion

Background:

A frequency divider can be constructed from T flip-flops, where T-type flip-flop is shown in Figure 1. A T flip-flop is obtained from a JK flip-flop by tying the J and K inputs together to form the T input.



T	Q(t+1)
0	Q
1	Q'

Function table

Figure 1

The input of each T flip-flops is set to 1 to produce a toggle at each cycle of the clock input. For each two toggles of the first cell, a toggle is produced in the second cell, so its output is at half the frequency of the first. The output of the third cell is 1/8 the clock frequency. The same device is useful as a counter. Figure 2 shows schematic design for three-bit frequency divider by using three T-type flip-flops.

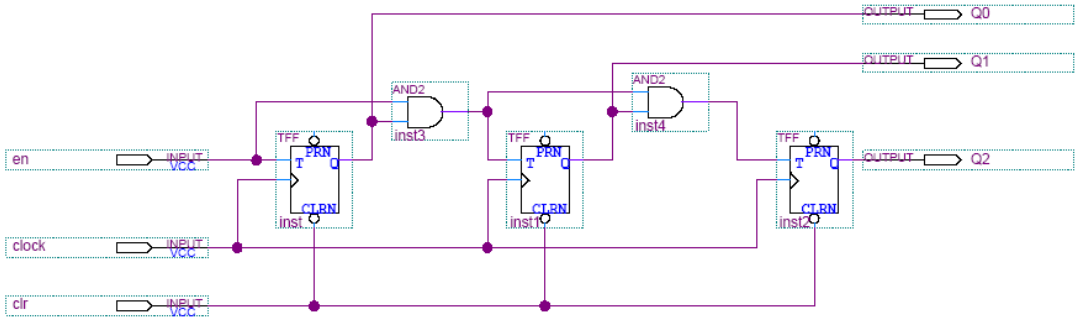


Figure 2

Figure 3 is to use the following library in your VHDL design entry

```
1   library IEEE;
2   use ieee.std_logic_1164.all;
3   use ieee.std_logic_arith.all;
4   entity FreqDivider is
5       port (
6               clk     : in  std_logic;
7               enable  : in  std_logic;
8                   Q0  : out std_logic;
9                   Q1  : out std_logic;
10                  Q2  : out std_logic
11            );
12      end FreqDivider;
13
14  architecture arc of FreqDivider  is
15      signal counter: unsigned (2 downto 0);
16  begin
17      process (clk, enable)
18      begin
19                      --positive triggering
20          if (clk' event and clk ='1') then
21              if (enable ='1') then
22                  counter <= counter + 1;
23              end if;
24          end if;
25      end process;
26
27          Q0 <= counter(0);
28          Q1 <= counter(1);
29          Q2 <= counter(2);
30  end arc;
31
```

Figure 3

Figure 4 is the simulation result. Check the relation among clock, Q0, Q1, and Q2

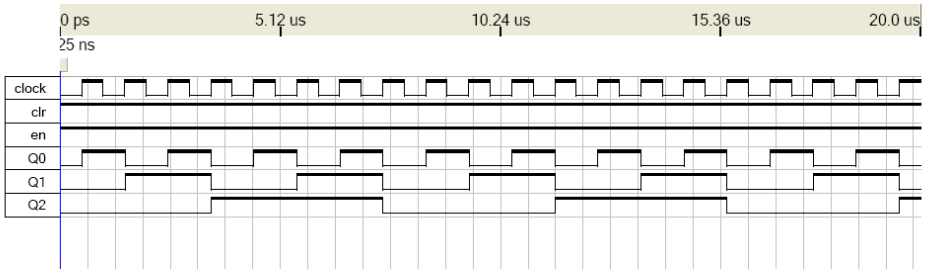


Figure 4

Suppose a clock frequency is clock = 24kHz (you can set different frequency by using function generator)
The frequency of the 2^0 output line (Q0) is 1/2 of the input clock line (12kHz)
The frequency of the 2^1 output line (Q1) is 1/4 of the input clock line (6kHz)
The frequency of the 2^2 output line (Q2) is 1/8 of the input clock line (3kHz)

Project:
Consider the extension of the circuit. Create a 8-bit synchronous counter which uses eight *T-type flip-flops* or VHDL The counter increments its count on each *positive edge* of the clock if the *Enable* signal is asserted. The counter is reset to 0 by using the *Reset* signal.

8-bit frequency divider

1. Write a *VHDL* file or create a *BDF* file that defines a **8-bit counter (8-bit frequency divider)** by using the structure depicted in Figure 2 (bdf) or extension from Figure 3 (vhdl) or LPM freqdiv (Figure 5), and compile the circuit. What is the frequency relation between the clock and eight outputs from eight *T-type flip-flops*, respectively?
2. Create a waveform vector and simulate your circuit to verify its correctness of the design of 8-bit frequency divider.
3. Augment your VHDL file or BDF file to use the GPIO as the *Clock* input (the input from function generator), switches SW_1 and SW_0 as *Enable* and *Reset* inputs, and 8 pins from GPIO to display the divided frequencies in oscilloscope.
4. All output signals from GPIO must be connected to your breadboard in order to be observed from oscilloscope or displayed by external LEDs
5. Make the necessary pin assignments and compile the circuit.
6. Programming FPGA and test your implementation. All input and output frequencies should be measured from oscilloscope. Develop a table and record the observable frequencies. Take photos from your experiments.

	Name	Component	Pin Location		Name	External Component	Pin Location	Component	Pin Location
Input	RST	SW0	PIN_N25	Output	Q7	Frequency observed on oscilloscope		LEDG7	
Input	..	..		Output	Q6	Frequency observed on oscilloscope		LEDG6	
Input	CLK	Function generator			:				
				Output	Q1	Frequency observed on oscilloscope		LEDG1	
				Output	Q0	Frequency observed on oscilloscope	GPIO	LEDG0	PIN_AE22

7. You may use the following library in your VHDL design entry:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
…
```

8. Quartus II software provide a library. It is called Library of Parameterized Modules and can be found in the /others. Right-click in a new block design (bdf) workspace and choose insert> symbol **freqdiv**

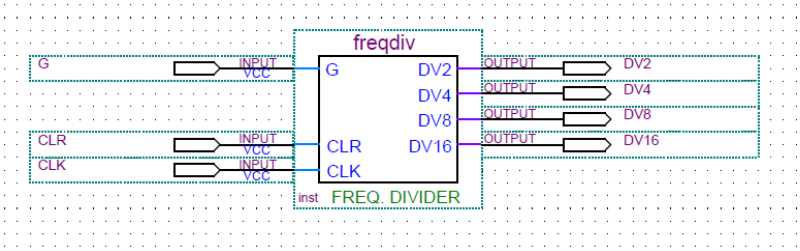


Figure 5