VHDL 4250

Lab #7 - CLOCK GENERATION

Brandon Gallo and Samuel Bishop

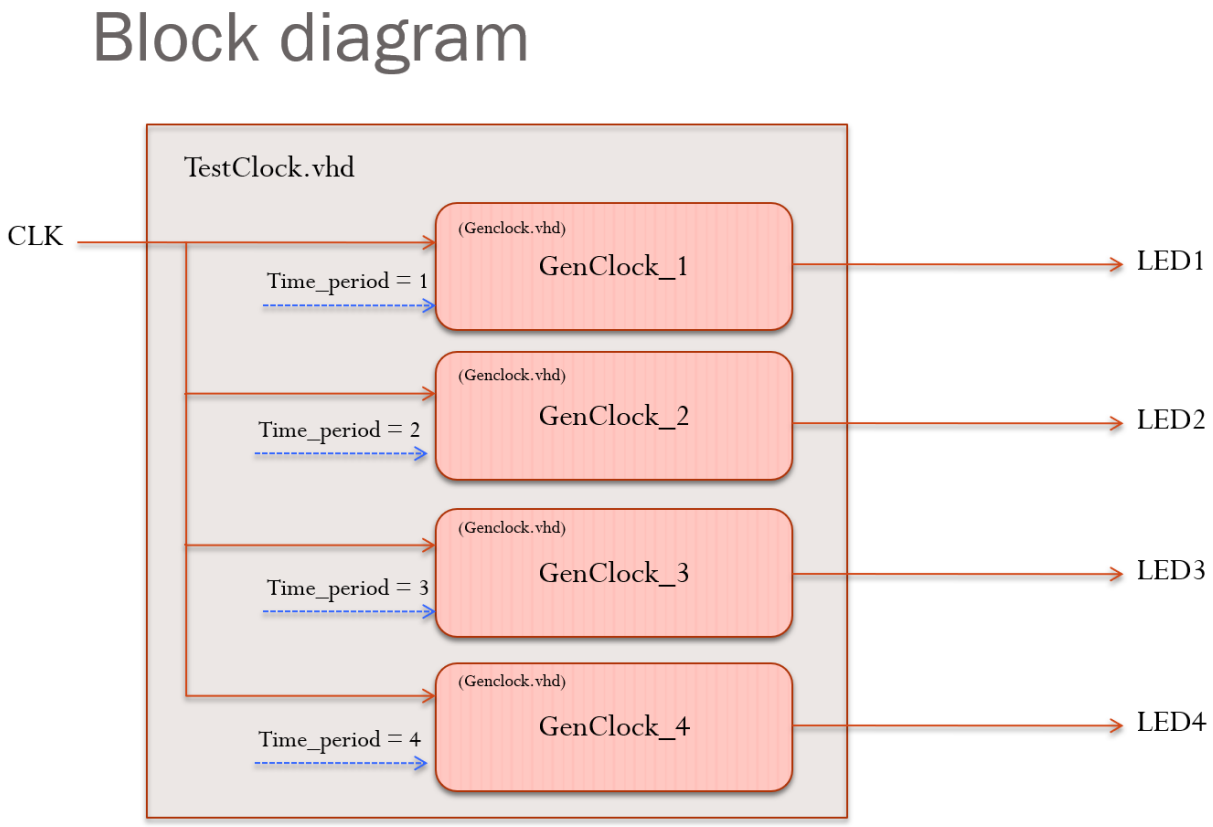
Group 5

04/16/20

a. Objective

The objective of this lab is to implement a clock that generates signals on selectable periods of .5s, 1s, 2s, and 4s from hypothetically to be 50MHz clock speed on the Xilinx Spartan-3 FPGA prototyping board.

Here students had to implement a Generic Clock circuit within the hardware description language by using the properties of the Generic data type as described in Section 8-10 of our textbook.



b. Lab Work

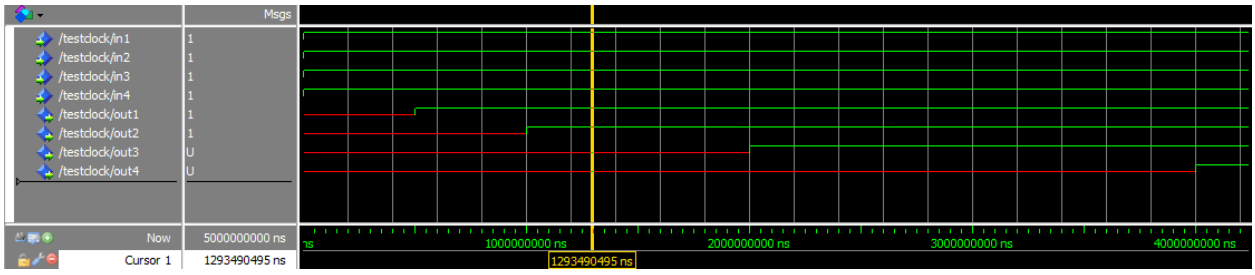
The first step in the assembly of this circuit was to design the Generic Clock circuit to use in the testing later. In order to do so, a file named GenClock.vhd was created. Inside of GenColock.vhd the IEEE library was utilized and all of the standard logic data types as well as numeric data types were imported from that library. Next the entity GenClock was created with an embedded Generic variable named time_period with a range of inputs from 1 to 4. Then the group ported this entity's inputs as input clk, as well as ported an output Clock both of std_logic types. A behavioral architecture for the entity was described within the VHDL code.

Within the behavioral architecture for our synchronous counter a signal clock_value of type std_logic was created as the intermediary assignment variable. After beginning the

architecture signal clock_value is immediately set to Clock, then the process checks for updates on the variable clk. In the process block the Generic variable time_period is evaluated and if it is equal to 1 then the clock_value variable will be set to HIGH (‘1’) after half a second. Likewise if time_period is set to 2, 3, or 4 then the clock_value variable will be set to HIGH after one, two, or four seconds respectively.

After the GenClock.vhd file is complete students implement the file TestClock.vhd to test the Generic Clock component and insure functionality. Within this new file the IEEE library was utilized and all of the standard logic data types as well as numeric data types were imported from that library. Next the entity TestClock was created and ported inputs in1, in2, in3, in4 as well as outputs out1, out2, out3, out4 as std_logic types. Next a structural architecture of TestClock was created which references our GenClock.vhd file to get the GenClock component. Within the structural architecture four different Generic Clock components were port mapped using all four different generic settings as well as different inputs and outputs to insure functionality.

TestClock.vhd was then tested within model sim to ensure that HIGH signals were sent after .5s, 1s, 2s, and 4s dependant on the generic signal provided to the time_period variable.



Generic Clock Test Values Waveform

c. Conclusion

This lab was slightly easier than some of our previous labs up to this point, but we still ran into some complications. Implementing the generic data type was slightly confusing at first because we did not know how to deal with this new concept. However, after troubleshooting and messing around with the VHDL code, we were able to get the solution that we wanted. This lab was a good demonstration of a program that used the “Generic” function. We realized that this function is similar to an enumeration in other programming languages like C, or C++. By making this comparison to a programming language that we have a better background in, the concept of “Generic” became relatively simple. We now see the usefulness of using this “Generic” function and can see the practical use of applying it to the CLK to perform a task at various time periods.

As we progress into next week's lab, we now have yet another programming technique that will help us complete the desired task.

```
1  library IEEE;
2  use IEEE.std_logic_1164.all;
3  use IEEE.numeric_std.all;
4
5  Entity GenClock is
6          -- Generic is like an enumeration of the time period type.
7          Generic(time_period : integer range 1 to 4 ); -- time period variable has 4 ranges (.5s, 1s, 2s, 4s)
8          Port (
9                  clk: in std_logic;
10                 Clock: out std_logic
11         );
12 End GenClock;
13
14 Architecture Behavior of GenClock is
15 signal clock_value: std_logic;
16 begin
17         Clock <= clock_value;
18         process (clk)
19         begin
20                 if time_period = 1 then
21                         clock_value <= '1' after 500 ms;
22                 elsif time_period = 2 then
23                         clock_value <= '1' after 1000 ms;
24                 elsif time_period = 3 then
25                         clock_value <= '1' after 2000 ms;
26                 elsif time_period = 4 then
27                         clock_value <= '1' after 4000 ms;
28                 end if;
29         end process;
30  end Behavior;
```

GenClock.vhd Source Code

27 lines (24 sloc) | 660 Bytes

```
1  library IEEE;
2  use IEEE.std_logic_1164.all;
3  use IEEE.numeric_std.all;
4
5  Entity TestClock is
6          Port (
7                  in1, in2, in3, in4: in std_logic;
8                  out1, out2, out3, out4: out std_logic
9          );
10 end TestClock;
11
12 Architecture Execute of TestClock is
13         Component GenClock
14                 Generic(time_period : integer range 1 to 4 );
15                 Port (
16                         clk: in std_logic;
17                         Clock: out std_logic
18                 );
19         end Component;
20
21         -- Declaring test bench cases for GenClock
22         begin
23                 GC1: GenClock generic map (1) port map (in1, out1);
24                 GC2: GenClock generic map (2) port map (in2, out2);
25                 GC3: GenClock generic map (3) port map (in3, out3);
26                 GC4: GenClock generic map (4) port map (in4, out4);
27 end execute;
```

TestClock.vhd Source Code

References

Roth, Charles H., and Lizy Kurian John. *Digital Systems Design Using VHDL*. Cenage Learning P82, 2008.