Experiment 9: Clock Divider

Debasish Panda 21D070021

October 18, 2022

1 Overview of the experiment

The purpose of the experiment is to design a clock divider using which has been described in VHDL using Behavioral-Dataflow modelling. There is a 50 MHz on-board clock and we have to divide it to generate 0.5 MHz. We also have to demonstrate the divider successfully on Xenon board.

The method to design the divider is as follows: Suppose we need to generate f = 5 MHz from 50 MHz master clock. For this, we need a counter such that the clock out remains HIGH for 5 Input (master) Clock Cycles and LOW for next 5 Clock Cycles. In order to do this, we set-up a counter that starts from 1 and increments at every positive edge of the Input Clock (master) till the count reaches its maximum value, which is 5 in this case.

count = 50 MHz/(2 ∗ f) = 5

After the count reaches 5, count will be initialized back to 1. And clock output will go LOW till count reaches maximum again. Note: Here we are counting from 1 to maximum count. (Not from 0 to maximum count -1).

The appropriate code for Behavioral-Dataflow modelling of the system is written using VHDL on Quartus Prime software. Next, we have to demonstrate the Clock Divider practically by using the Xen10 FPGA board. In order to run the VHDL code on the board, we also need to determine the input and output pins correctly as per the specifications of the circuit board.

2 Experimental setup

Note that we need to describe the divider using behavioral modelling in order to execute the required operations, so we do not need to design any hardware design as such.

2.1 Design Documentation and VHDL Code

The VHDL code associated with the digital circuit is as follows:

```
library ieee;
use ieee.std_logic_1164.all;

entity clock_divider is
port(clk_out : out std_logic;
clk_50, resetn : in std_logic);
end entity clock_divider;

architecture Struct of clock_divider is

 signal i: integer := 0;
 signal outp: std_logic := '1';

begin
clock_proc: process(clk_50, i, resetn)
begin
if(resetn='1') then
i <= 0;
outp <= '0';
elsif(clk_50='1' and clk_50'event) then
if(i=5e7) then
outp <= not outp;
i <= 0;
else
i <= i+1;
end if;
end if;
```

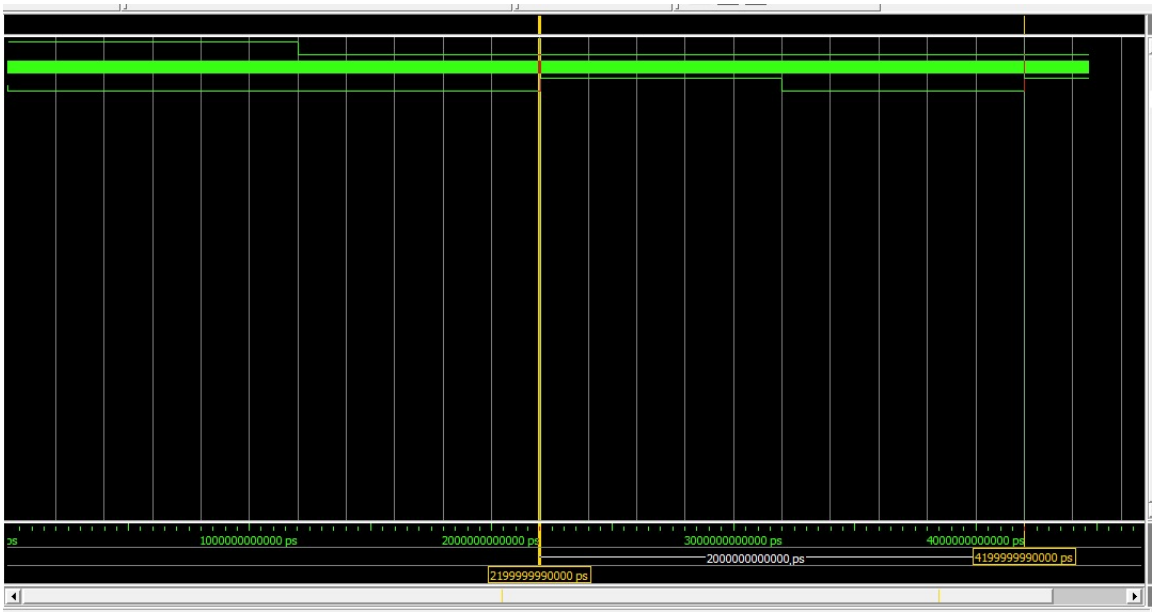
```
end process;

clk_out <= outp;

end Struct;
```

3 Observations

The experiment was conducted successfully, and the expected output was obtained using the RTL simulation. The Xenon board also gave the desired result, with the clock working as per specifications. The expected output of the Clock Divider is shown in the form of the final RTL Simulation result of the design.



RTL Simulation of Clock Divider

Following this, we have to do the pin planning correctly and convert our VHDL file to a .svf file, which is executable by the Xen10 board and is also needed for running the Scanchain program.

4 References

[1] EE214 Github page

[2] Overleaf LaTeX tutorial for beginners

[3] Introduction to Xen10 Board