Task 4

David Shmailov, Aviram Lachmani

May 2021

Contents

1 Introduction

In this task we were required to do a digital verification of our design from lab 2, a modulus counter, using an FPGA learning kit and the Quartus software. The assignment is divided into two parts: a synthesis and performance verification and than a hardware verification via the FPGA.

2 Performance verification

Before we could properly measure the maximal frequency this design is able to work, we had to make sure all inputs and outputs are buffered by registers. we found out that although this design is suppose to be sequential, some inputs in our original design are not synthesized in a way that they are buffered by a register. also we had to add an enable line for our design which was not previously required. The following are the changes we made to our design: for every process we added an enable port to its sensitivity list, added the port with an AND gate together with the clock. this made the register to be synthesized with an enable line and not an external AND gate. for example:

```
proc1 : process(clk,rst,enable)
        VARIABLE ount : std_logic_vector (n-1 downto 0);
        begin
                IF (rst='1')THEN
                                ount:=(others=>'0');
                                countOut<= (others=>'0');
                ELSIF (clk'EVENT AND clk='1' and enable = '1') THEN
                                if (connection=ount)then
                                        countOut<=(others=>'0');
                                        ount:=(others=>'0');
                                ELSE
                                        ount:= ount+addone;
                                        countOut<=ount;
                                END IF;
                END IF;
        end process;
```

another modification we made is wrap our UpperBound input into a D-Flip-Flop with the following process:

```
upperReg: process(clk,rst,enable)
                variable d : std_logic_vector(n-1 downto 0);
        begin
                if(rst='1') then
                        d:=(others => '0');
                elsif (clk'event and clk='1' and enable='1') THEN
                        d:=upperBound;
                end if;
                upperConnection <= d;
        end process;
```

the maximum frequency for this design by timing analyzer report is: 237.47 MHz. we have set the limit in the SDC file as required.

The following is an RTL Viewer result of the design. The implementation of our uses only the top level and is not separated into smaller blocks:

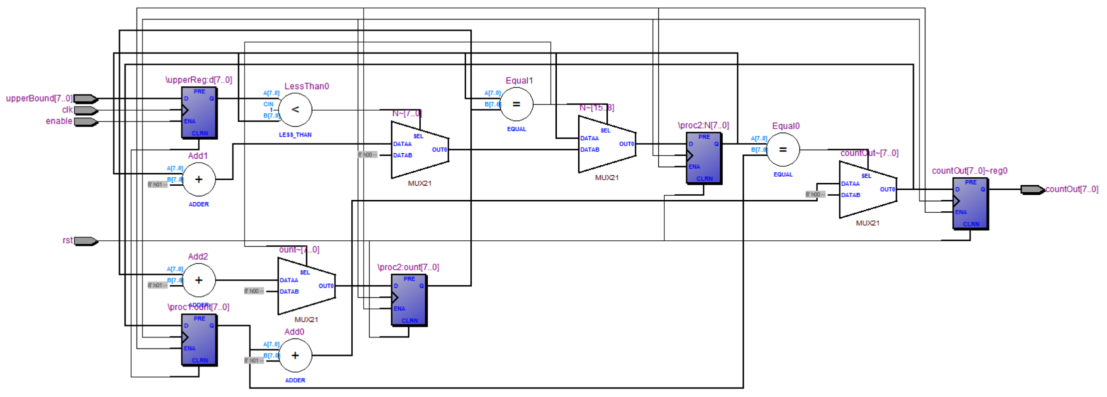


Figure 1: RTL view of top

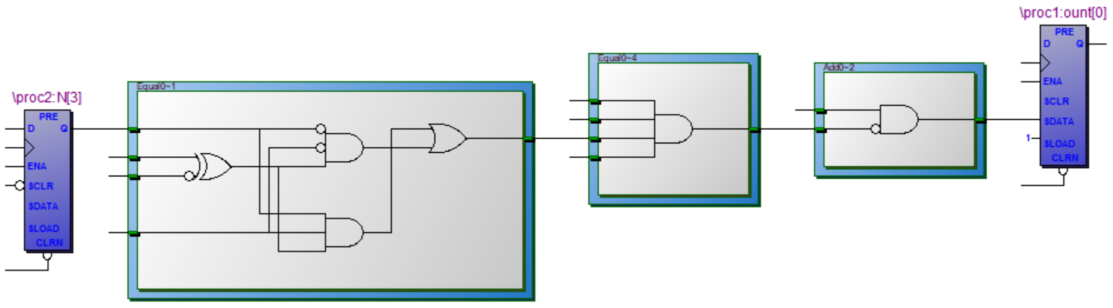


Figure 2: longest critical path of top

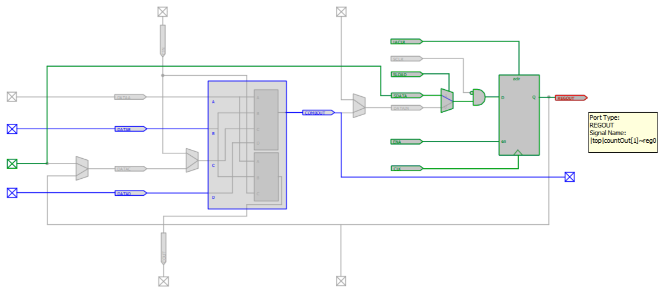


Figure 3: Critical path by resource property

3 Hardware Test Case

We developed the verification test-bench in the files ”Digital_System” and ”Digital_System_Top”, the design under test is the file ”top”.

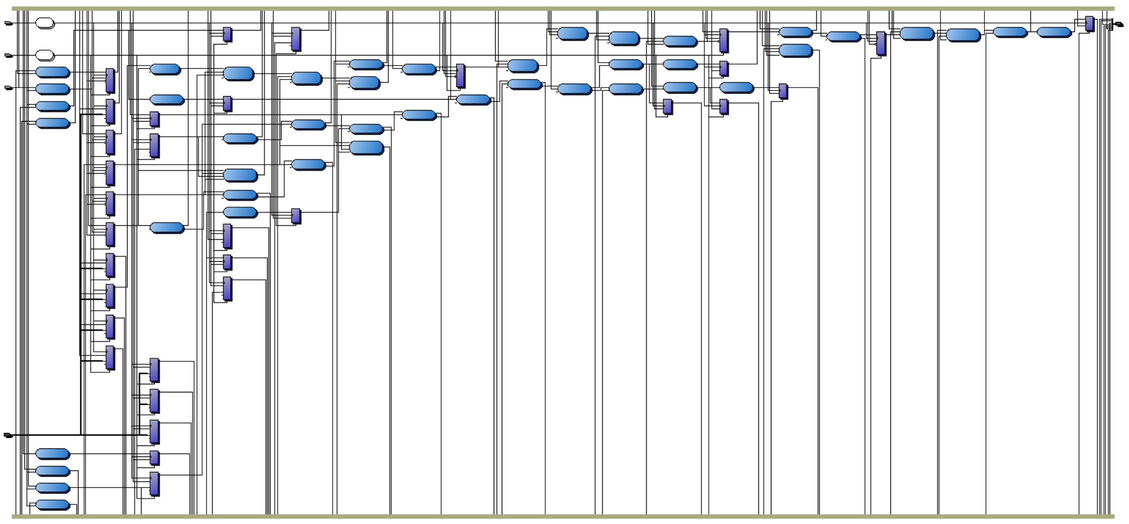
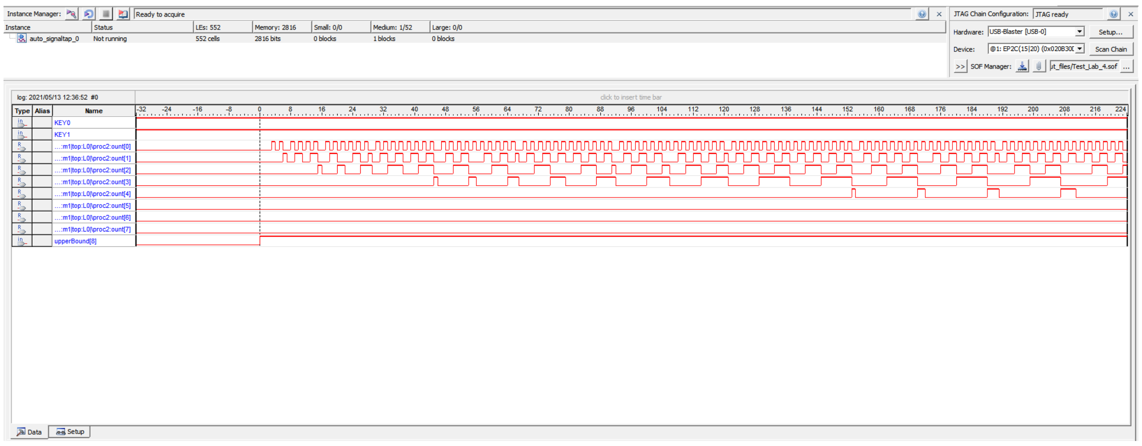


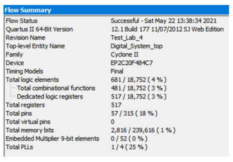
Figure 4: Synthesis of Top and verification test bench



Instance	Status	LEs: 552	Memory: 2816	Small: 0/0	Medium: 1/52	Large: 0/0
auto_signaltap_0	Not running	552 cells	2816 bits	0 blocks	1 blocks	0 blocks

Instance Manager:
Ready to acquire
JTAG Chain Configuration:
JTAG ready
Hardware: USB-Blaster [USB-0]
Setup...
Device: @1: EP2C(15|20) (0x020B30D
Scan Chain
SOF Manager:
ut_files/Test_Lab_4.sof
log: 2021/05/13 12:36:52 #0
click to insert time bar
Type
Alias
Name
KEY0
KEY1
...:m1|top:L0|\proc2:ount[0]
...:m1|top:L0|\proc2:ount[1]
...:m1|top:L0|\proc2:ount[2]
...:m1|top:L0|\proc2:ount[3]
...:m1|top:L0|\proc2:ount[4]
...:m1|top:L0|\proc2:ount[5]
...:m1|top:L0|\proc2:ount[6]
...:m1|top:L0|\proc2:ount[7]
upperBound[8]
-32 -24 -16 -8 0 8 16 24 32 40 48 56 64 72 80 88 96 104 112 120 128 136 144 152 160 168 176 184 192 200 208 216 224
Data
Setup

Figure 5: signal tap sampling



Flow Summary	
Flow Status	Successful - Sat May 22 13:38:34 2021
Quartus II 64-Bit Version	12.1 Build 177 11/07/2012 SJ Web Edition
Revision Name	Test_Lab_4
Top-level Entity Name	Digital_System_top
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Total logic elements	681 / 18,752 (4 %)
Total combinational functions	481 / 18,752 (3 %)
Dedicated logic registers	517 / 18,752 (3 %)
Total registers	517
Total pins	57 / 315 (18 %)
Total virtual pins	0
Total memory bits	2,816 / 239,616 (1 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	1 / 4 (25 %)

Figure 6: resource usage of the Quartus chip

4 ModelSIM

Here we will show our simulation analysis from LAB2:

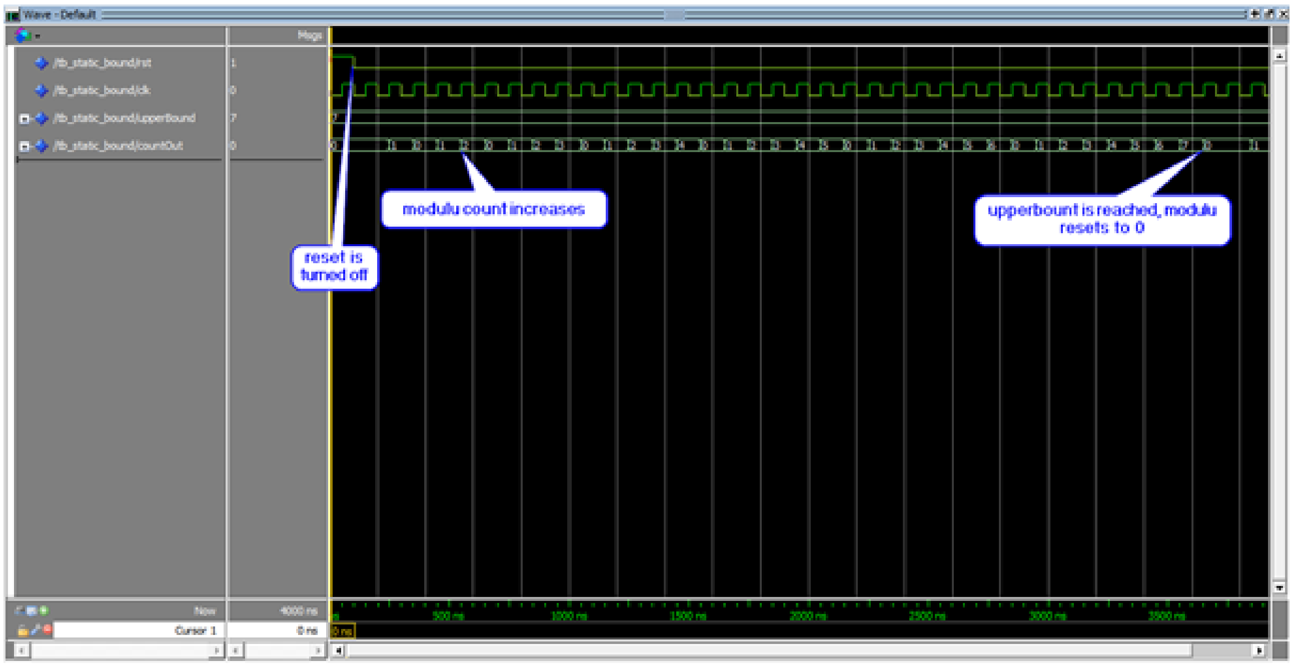


Figure 7: static upperbound test

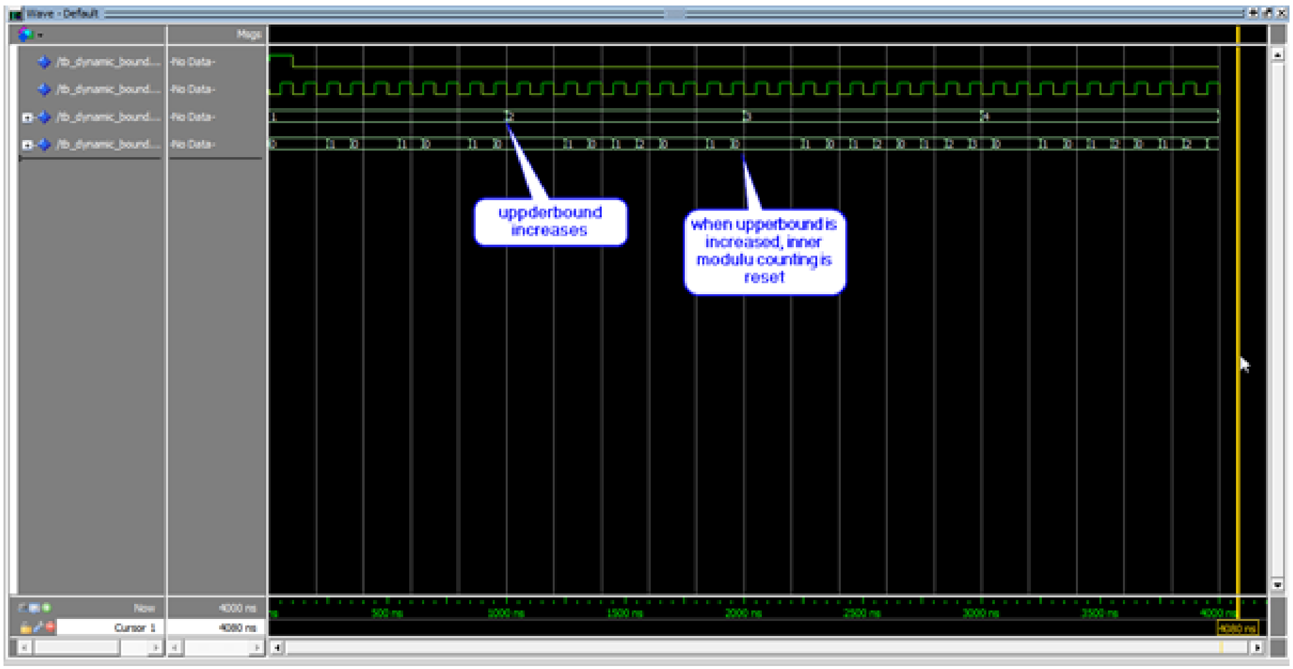


Figure 8: dynamic upperbound test

5 Conclusions

This is the first time we are exposed to post synthesis verification. Up until this point we only could simulate our design, but were not exposed to how it will be synthesized. We learned how things that logically wouldn't change the output of our design, do affect the synthesis of it.

We also noticed that the carry ripple adder that is implicitly implemented in the design is the longest critical path of the system. The carry ripple adder is added because of the way we implemented the clock divider. We learned a convenient way to divide by powers of 2 the clock by connecting the original clock to an adder and using the nth overflow bit to divide the clock by 2^n. the downside of this method is the necessity of an adder, but since this is implemented inside our verification test-bench and not inside the DUT, its ok to make this compromise.