Due Date: 28.03.2024

EHB425 VLSI2
Homework 4

The exponentiation operation $C=A^B \bmod 2^{16}$, where $A=(a_{k-1},a_{k-2},\ldots,a_1,a_0)_2$, $B=(b_{k-1},b_{k-2},\ldots,b_1,b_0)_2$, will be implemented by using the following algorithm

Left-to-right exponentiation algorithm

```
Input:     $A=(a_{k-1},a_{k-2},\cdots,a_1,a_0)_2$, $B=(b_{k-1},b_{k-2},\cdots,b_1,b_0)_2$, $A, B < 2^{16}$
Output:    $C=A^B \bmod 2^{16}=(c_{k-1},c_{k-2},\cdots,c_1,c_0)_2$
Step1:  $C=1$
Step2:  for $i=k-1:0$
Step3:         $C=C^2 \bmod 2^{16}$
Step4:         if $b_i=1$
Step5:              $C=C*A \bmod 2^{16}$
```

1) Draw the algorithmic state machine (ASM) of “Left to right exponentiation algorithm”.

2) Draw the data path of the circuit for implementation of “Left to right exponentiation algorithm” using your ASM from (1).

3) Determine the control signals from the control path to the datapath. Draw the control signal sequence table.

4)

a. Write a Verilog code for definition of the state transitions of the control path and control signals.

b. Produce the RTL schematic from Vivado and dot schematic from OpenLane for your controller that you have designed in 4.a.

c. Behaviorally simulate the circuit that you have designed in 4.a using **iverilog** and **gtkwave**. Show all possible cases of state transitions.

5)

a. Write a Verilog code of your datapath in (2) by describing the bit length, k, as parameter. You will use your multiplier design in Homework 2 for MUL block.

b. Produce the RTL schematic from Vivado and dot schematic from OpenLane for the circuit that you have designed in 5.a for k=16.

c. Behaviorally simulate the circuit that you have designed in 5.a using **iverilog** and **gtkwave**.

6)

a. Connect control and data path that you have designed in 4 and 5 in order to build a single purpose processor for “Left-to-right exponentiation algorithm”.

b. Produce test vectors for your design by using Matlab.

c. Write a self-checking testbench for simulation of your single purpose processor in order to verify that all the test vectors are satisfied. You can find some examples for reading test vectors from files, using $display, $monitor, etc. commands for assessment of the results and writing the results to a file from in the following web pages. But of course you can find more references for this purpose.

https://veriloggguide.readthedocs.io/en/latest/verilog/testbench.html

https://hardwarebee.com/ultimate-guide-verilog-test-bench/

http://www.testbench.in/TS_08_SELF_CHECKING_TESTBENCHS.html

d. Perform behavioral simulation for the circuit you obtained in 6.a using the testbench you have written in 6.c using **iverilog** and **gtkwave**.

7)

a. Apply the full **OpenLane** chip-flow on the design you created at 6.a, using k=16. Report the errors you have encountered (if any) while applying the flow, and state how you’ve solved each one. Try to obtain a design with no timing (setup or hold violations), DRC, LVS and antenna violations. Show the warnings you get at the end of the flow and briefly discuss why do you think they occur.

b. Report the maximum clock frequency and chip area of the final design. What is the worst positive setup slack value?

c. Perform post-place&route functional simulation on the circuit that you have obtained in 7.a using the testbench you written in 6.c using **iverilog** and **gtkwave**.

References

1) Frank Vahid, **Digital design, with RTL design, VHDL, and Verilog,** Hoboken, NJ : John Wiley, 2010.
2) Peter D Minns, **FSM-based digital design using Verilog HDL,** Chichester, England : J. Wiley & Sons , c2008.
3) Pong P. Chu, **FPGA prototyping by Verilog examples Xilinx Spartan -3 version**, Hoboken, N.J. : J. Wiley & Sons, c2008.
4) Morris Mano, Michael Ciletti, **Digital Design With an Introduction to the Verilog HDL**, FIFTH EDITION, Pearson.