Combinational Division Circuit

Grading Sheet

Group #:_____ Name(s):_____ .

Grading

- Section 4.1(a): VHDL code (60 points):_____.
  Attach code printout (with proper header and comment)
- Section 4.2(b) RT-level simulation waveform (20 points):_____.
  Attach simulation waveform screen capture
- Section 4.3(e): post-synthesis simulation waveform (10 points): _____.
  Attach simulation waveform screen capture
- VHDL code format and comments (10 points):_____.

Total points: .

Experiment

Combinational Division Circuit

1 Purpose

To design an intermediate-sized combinational circuit

2 Reading

- Chapter 3 of *FPGA Prototyping by VHDL Examples 2nd edition.*

3 Project specification

The division is a complex operation. It is not supported by synthesis software and must be implemented manually. A "quick-and-dirty'' way to derive a combinational division circuit is to use a sequence of "comparison-and-subtraction" operations. Consider a 4-bit unsigned system. The dividend is $y_3y_2y_1y_0$ and the divisor is $d_3d_2d_1d_0$. After $y_3y_2y_1y_0 \div d_3d_2d_1d_0$ is performed, the quotient is $q_3q_2q_1q_0$ and the remainder is $r_3r_2r_1r_0$. The operation can be done in stages:

- 1st stage: Create a 7-bit partial dividend P4, $000y_3y_2y_1y_0$
- 2nd stage:
  - Create a partial divisor D4, $d_3d_2d_1d_0000$.
  - If P4≥D4, then set P3=P4–D4, q_3=1; otherwise, set P3=P4, q_3=0;
- 3rd stage:
  - Create a partial divisor D3, $0d_3d_2d_1d_000$.
  - If P3≥D3, then set P2=P3–D3, q_2=1; otherwise, set P2=P3, q_2=0;
- Repeat the stages two more times to obtain q_1 and q_0. The 4 LSBs of P0 is the remainder, $r_3r_2r_1r_0$.

Based on this algorithm, design a 5-bit division circuit. The input and output are

- input:
  - `y, d`: two 5-bit inputs representing dividend and divisor.
- output
  - `q, r`: two 5-bit outputs representing quotient and remainder.

4 Design and simulation

4.1 Division circuit design

(a) The entity declaration of this design is

```
entity division is
    port(
         y, d: in std_logic_vector(4 downto 0);
         q, r: out std_logic_vector(4 downto 0)
    );
end division;
```

Derive the architecture body. The VHDL code must be properly documented.

.

4.2 RTL ModelSim simulation

(a) Use the testbench (division_tb.vhd) to simulate your VHDL code. The testbench includes the test patterns of 27÷1, 27÷0, 1÷17, 0÷17, 27÷3, 27÷5, 27÷17, 27÷27, 27÷29, 27÷31.

(b) Develop a proper "layout" and "format" for the simulated waveform. To get full credits, the input and output signals should be arranged as four 5-bit buses and represented in a proper format so that the simulation result can be easily understood. Do a screen capture of the simulated result.

4.3 Post-synthesis ModelSim simulation

(a) Perform compiling (synthesis/placement and routing) and obtain the .vho file.

(b) Follow the format guideline to add a header to the file and verify the "structure" architecture body is generated.

(c) Revise the testbench to use the "structure" architecture for uut.

(d) Perform post-synthesis simulation

(e) Do a screen capture of the simulated result.

(f) Since the waveform is the same as the RT-level simulation, the snapshot should include the expanded uut on the left panel. No point will be given if the uut unit is not expanded in the snapshot.