Lab02 – Modeling, synthesis, simulation & implementation of sequential (synchronous) logic circuits using FPGAs

Dragomir MILOJEVIC, v3.3

1 Objectives

Learn how to model digital sequential logic circuits in Verilog.

2 Exercises

2.1 8-bit parallel/serial load shift register

Write Verilog model of an 8-bit parallel/serial load shift register with a synchronous reset. The operation of the register is defined as follows:

- Load operation is enabled when input signal `LD` is set to 1
- Parallel load is enabled when `LD` & `P_in` are both set to 1; 8-bit word at input `PD_in` is loaded in a single clock cycle (`PD_in` is a vector)
- Serial load is enabled when `LD` is set to 1 and `P_in` to 0; 8 bits are presented one at a time (one per clock cycle) on input `SD_in`
- If `LD=0` & `SHIFT=1`, the content of the register is shifted right, that is LSB is forgotten, bit MSB-1 is set to MSB, and MSB to 0
- For any other input combination, the register will simply hold the stored value

2.2 n-bit parallel/serial load shift register

Modify Verilog model of the register above to enable implementation of arbitrary sized input word. You should use `generic` for this.

2.3 Receiver/decoder module

Digital receiver module receives 4-bits word encoded using Gray code (see table below). Transmission & reception are done serially using one bit-line `D_in`. Beginning of the transmission is indicated with `start=1`, after which the MSB is sent first, followed by three other bits. Incorrect codes might appear due to transmission errors (codes not listed in the table). Once

Decimal	Gray	BCD
0	0000	0000
1	0001	0001
2	0011	0010
3	0010	0011
4	0110	0100
5	0111	0101
6	0101	0110
7	0100	0111
8	1100	1000
9	1101	1001

all 4 bits are transmitted, receiver module outputs decimal value of the word on output `D_out`. If error(s) occurred during transmission, output `E` must be set to 1, and `D_out` must be set to `1111`. `D_out` and `E` should change at the same time.

2.4 State machines (FSM)

Design a very simple Moore state machines with at least 4 states. Pick transitions freely. Do the implementation using minimal encoding (can you explain what this is?), and one-hot. What can you conclude from 2 different implementations?

2.5 To do

For each circuit:

1. Write Verilog model of the circuit.
2. For each Verilog module write a test-bench.
3. Perform functional simulation using Model/QuestaSim and show correct functional behavior of the circuit.
4. Do synthesis, place and route (P&R) and analyze various generated reports. Make a (very) short summary of performance using these reports. Derive minimum period, and maximum operating frequency.
5. Do post P&R simulation and show impact of switching and propagation delays on timing.