VLSI Design
Practice Problems for Verilog

(Instuctor: Prof. Abhishek Srivastava, CVEST IIIT Hyderabad)

Write Verilog code and testbench for the following and simulate using Icarus Verilog and viusalize waveforms using GTKWave

Exercise 1

Combinational gates: AND, OR, NOR, NAND, XOR, NOT and XNOR

Exercise 2

Implement the following Compound Combinational circuits using universal gates:

- Y = AB + CD
- Y = (ABC + DE).F
- Y = ((A+B')(CD + E))'
- F(A,B,C,D) = Π(1,3,5,7,13,15)
- F(A,B,C,D) = Σ(0,6,8,13,14), d(A,B,C,D) = Σ(2,4,10) where d(.) is don't care conditions

Exercise 3

Digital Blocks:

- **MUX** - 2*1 , 4*1 , 8*1 Design 2*1 MUX using Logic gates and use it to create 4*1 MUX then use 4*1 MUX to create 8*1 MUX
- **DEMUX** - 1*4 , 1*8 Design 1*2 DEMUX using logic gates and use it to create 1*4 DEMUX then use 1*4 DEMUX to create 1*8
- **DECODER** - 2*1 , 2*4 , 3*8 Design 2*1 DECODER using logic gates and use it to create 2*4 DECODER then use 2*4 DECODER to create 3*8 DECODER. Use enable pin also
- **PRIORITY ENCODER** - 8*3. Design 8*3 priority encoder using logic gates. Use enable pin also.

Exercise 4

Adders:

- **1 bit Full Adder:** Design using Logic gates.
- **4 bit Carry Ripple Adder:** Design using 1 bit Full adder
- **16 Bit Carry Ripple Adder:** The 16-bit adder will use 4 - bit ripple carry adders as components. The 16 – bit adder has two inputs ‘a’ and ‘b’ of type bit vector representing the addend and augend; and 1-bit input signal c_in of type bit representing the carryin. The adder produces one output signal s of type bit vector representing the sum word and a 1-bit output signal c_out of type bit representing the carryout.
- **4 bit Carry Look-Ahead (CLA)**
- **16 bit Carry Look-Ahead:** The 16-bit adder will use 4 - bit CLA as components. The 16 – bit adder has two inputs ‘a’ and ‘b’ of type bit vector representing the addend and augend; and 1-bit input signal c_in of type bit representing the carryin. The adder produces one output signal s of type bit vector representing the sum word and a 1-bit output signal c_out of type bit representing the carryout.

Exercise 5

Flip Flops:

- D FLIP-FLOP
- J-K FLIP-FLOP
- R-S FLIP-FLOP
- T FLIP-FLOP

Exercise 6

Design 4 bit universal shift register

Exercise 7

Design 3 bit Serial Adder

Exercise 8

Counters:

- 4 bit synchronous Binary Counter
- 4 bit up-down binary counter
- 4 bit Ripple counter

Exercise 9

Sequential Circuits:

- A sequential circuit with 2 D-flip flops A and B, two inputs x,y and one o/p z is specified as
  - A(t+1) = xy' + xB
  - B(t+1) = xA+xB'
  - z = A
- Write state diagram and verilog code for the following circuit

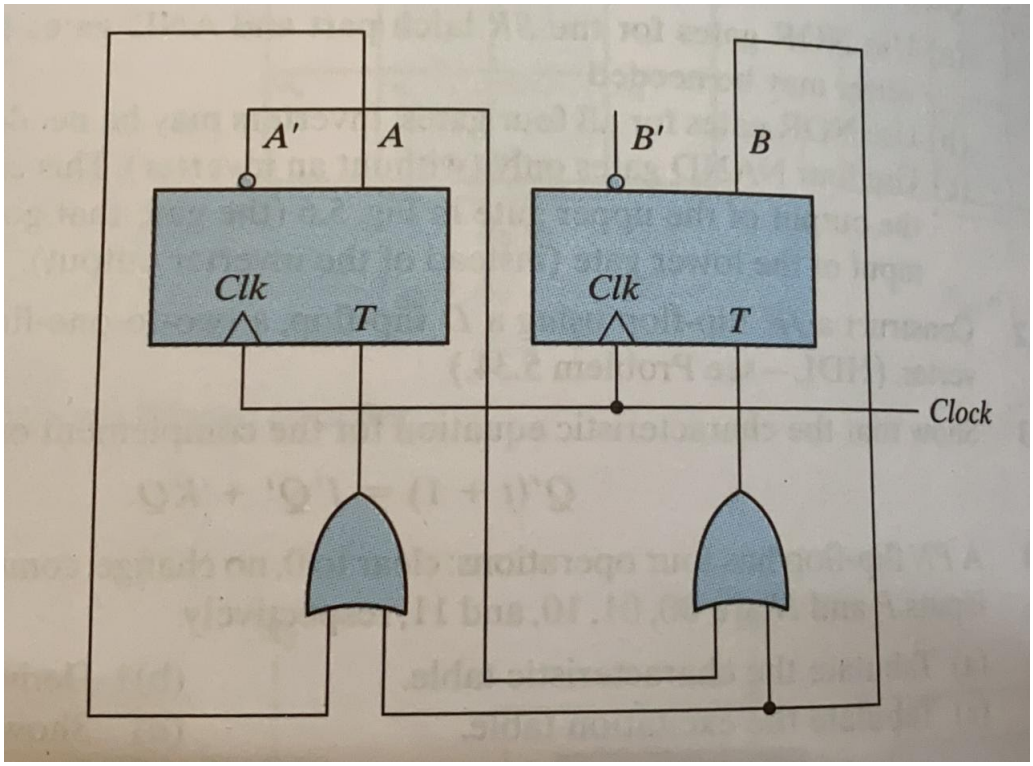

Figure 1: Circuit

Exercise 10

Finite State Machine:

- Implement Moore FSM for given FSM diagram

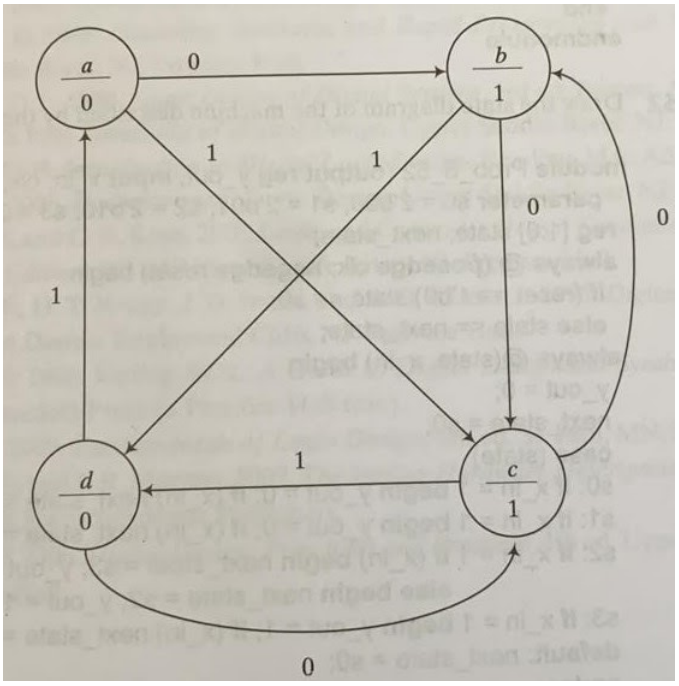

Figure 2: Circuit

- Implement Mealy FSM for given FSM diagram

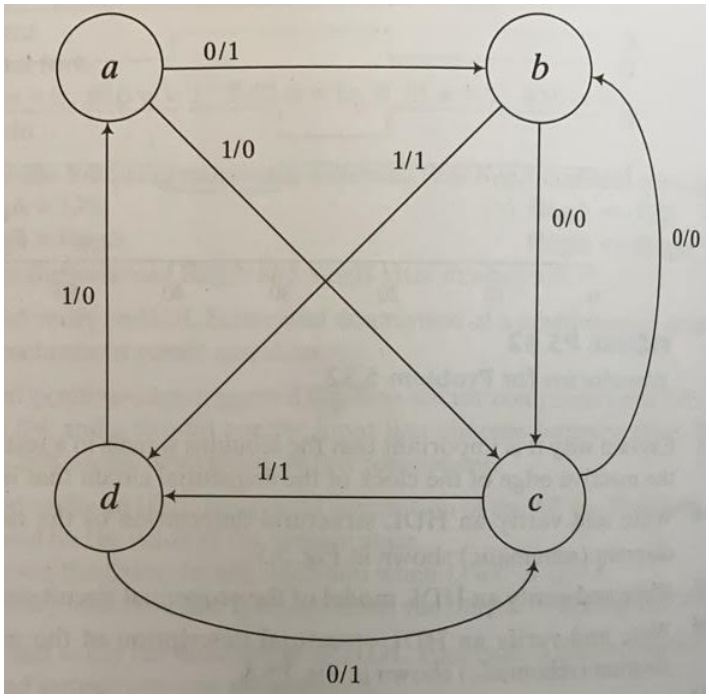

Figure 3: Circuit

Exercise 11

Write a Verilog module that implements the machine. Your module should have the following inputs/outputs. CLK used to clock the FSM RESET asserted high to reset the FSM to its initial state ONE asserted high to input a "1". Note that this signal may stay high for many cycles (eg, it's generated by a button press) before returning low. Each high period should count as a single "1" input, ie, to inputs two "1"s in series, the signal must return low in between the first and second "1". ZERO asserted high to input a "0". This signal has the same timing protocol as ONE above. OUT asserted high when at least two "0"s and two "1"s have occurred as inputs. Assume that all inputs have been externally synchronized with clk.