Indian Institute of Technology Kharagpur

AUTUMN Semester, 2021
COMPUTER SCIENCE AND ENGINEERING

Computer Organization Laboratory

Assignment-5: Verilog Design and Implementation of Sequential Circuits

Full Marks: 60

Time allowed: 6 hours

INSTRUCTIONS: Make one submission per group in the form of a single zipped folder containing your Verilog source code files(s) and Verilog testbench(es). Name your submitted zipped folder as `Assgn_5_Grp_<Group_no>.zip` and (e.g. `Assgn_5__Grp_25.zip`). Inside each submitted source and testbench files, there should be a clear header describing the assignment no., problem no., semester, group no., and names of group members. Liberally comment your code to improve its comprehensibility.

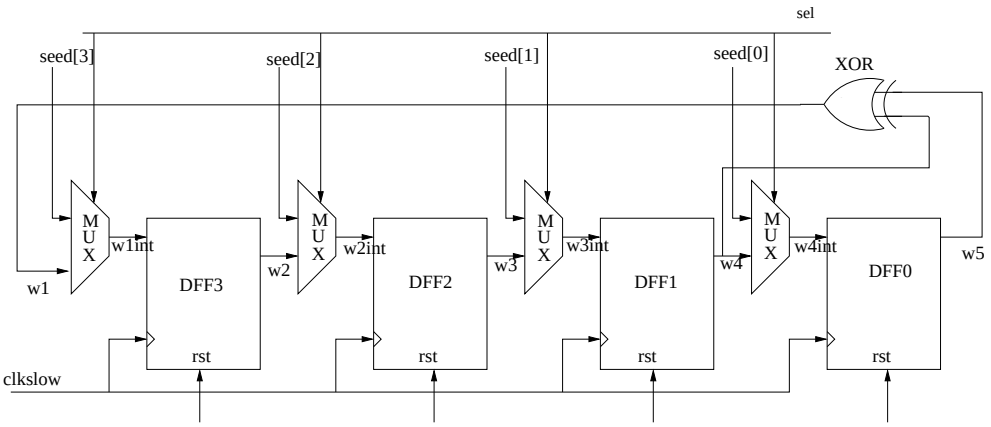

Figure 1: A 4-bit LFSR circuit.

1. **Linear Feedback Shift Register (LFSR) Design** The objective is to design a synchronous sequential circuit using four D Flip-flops, which when loaded by an initial non-zero vector called *seed*, cycles through all the 15 non-zero binary combinations through its state transitions, before returning to the initial vector (the seed). Thus, this circuit can be used as a modulo-15 counter, although it does not count in exact binary sequence. The only combinational gates required are XORs. These sequential circuits are commonly termed as *Linear Feedback Shift Registers* (LFSRs), and have extensive use in digital circuit design and testing, communication theory, etc.

   Design (using Verilog), simulate (using an appropriate Verilog testbench), and implement (on a FPGA platform supported by your CAD software tool), the 4-bit LFSR shown in Fig. 1. The four stages of the sequential circuit can be loaded by an arbitrary non-zero initialization seed by using the four multiplexers as shown in the diagram. In your testbench, verify that for an initialization seed 1111, the state transition sequence followed by this circuit is as follows: `1111` $\rightarrow$ `0111` $\rightarrow$ `0011` $\rightarrow$ `0001` $\rightarrow$ `1000` $\rightarrow$ `0100` $\rightarrow$ `0010` $\rightarrow$ `1001` $\rightarrow$ `1100` $\rightarrow$ `0110` $\rightarrow$ `1011` $\rightarrow$ `0101` $\rightarrow$ `1010` $\rightarrow$ `1101` $\rightarrow$ `1110` $\rightarrow$ `1111`. (20 marks)

2. **[Two's Complement Converter FSM]** Design (using Verilog), simulate (using an appropriate Verilog testbench), and implement (on a FPGA platform supported by your CAD software tool), a simple finite state machine (FSM) that in every clock cycle reads an input bit, and outputs a bit such that the bitstring output till that point is the two's complement of the binary number read till that point, including the most recently read bit (the number is considered to be input from the LSB side). The FSM has one input control signal which resets it to the initial state. Come up with an appropriate interface for your circuit. [Hint: consider a Mealy machine.] (10 marks)

3. **[Multiple-of-three Detector FSM]** Design (using Verilog), simulate (using an appropriate Verilog testbench), and implement (on a FPGA platform supported by your CAD software tool), a simple finite state machine (FSM) that in every clock cycle reads an input bit, and outputs a bit which indicates whether the binary number read till that point, including the most recently read bit (the number is considered to be input from the MSB side), is divisible by three. The input number is to be considered an unsigned integer. The FSM has one input control signal which resets it to the initial state. Come up with an appropriate interface for your circuit. [Hint: implement a Mealy machine.] (10 marks)

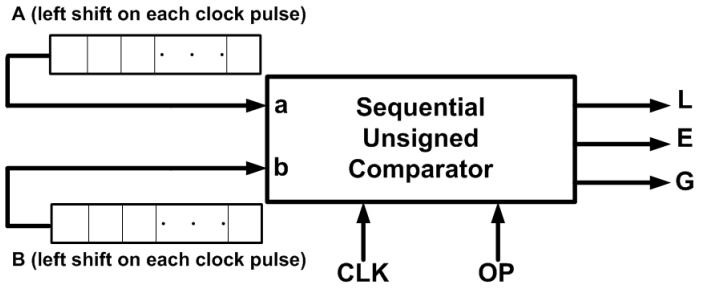

Figure 2: Sequential unsigned comparator schematic.

4. **[Sequential Unsigned Comparator]** Consider the task of designing a sequential circuit that can compare two unsigned integers A and B, each of which is 32-bit long. The block-level schematic is shown in Fig. 2, where in every clock cycle, the circuit serially reads one bit each of A and B **from their MSB sides** at its input ports a and b respectively, and performs state transition of an internal finite state machine (FSM). In every clock cycle, the contents of the registers containing A and B are left-shifted. The circuit has three output lines L, E and G, exactly one of which becomes logic-1 at the end of the session indicating the result. Thus, if $A < B$, then $\{L, E, G\} = 1\,0\,0$; if $A = B$, then $\{L, E, G\} = 0\,1\,0$; if $A > B$, then $\{L, E, G\} = 0\,0\,1$. For simplicity, assume that the machine is in an initial all-zero (reset) state before the beginning of the session (the reset signal has not been shown in the above schematic), and the input control signal OP is at logic-0; as long as OP remains at logic-0, $L = E = G = 0$. At the end of the session (i.e., after the arrival of least significant bits), OP flips to logic-1, indicating the end of the operation, and that the values of L, E and G are now ready to be read.

Design (using Verilog), simulate (using an appropriate Verilog testbench), and implement (on a FPGA platform supported by your CAD software tool), the above circuit, which in 32 clock cycles generate the comparison results. The circuit has an input control signal to load the shift registers by *Parallel Load* (instantaneous asynchronous load) operation, and another input control signal to reset the FSM. Come up with an appropriate interface for your implementation. (20 marks)