ELEC 204 Digital System Design

Laboratory Manual

Lab #5

Washing Machine Controller Design (State machines)

Special note for this lab: Make sure you do your own work and double-check it a few times before showing your solution to your teaching assistant. Plagiarism is strictly graded as zero.

1. Objectives

A state machine (or a finite-state machine) is a device or an algorithm, which can be in one of a set number of stable conditions depending on its previous condition and on the present values of its inputs. The main objective in this lab is to develop a state machine with timers and external asynchronous inputs. Then, we aim to use the state machine design for implementing its logic circuit design. In this experiment, you will:

- design a state diagram of a washing machine,
- implement its corresponding state diagram on FPGA using a modular design,
- experimentally demonstrate the operation of the state machine.

2. Procedure

1. Make sure you understand sequential logic circuit design and analysis principles. In particular, you need to understand how to design the state table, state equations, circuit schematics and state diagrams for a sequential logic circuit (also known as a state machine).
2. Watch the tutorial videos of this lab and answer the tutorial questions. Read the section 3 of this document for background information.
3. Prepare the preliminary report.
4. Write and compile your program on VHDL. Download the program on the board, present it in the lab.
5. Post your lab report and demo video on Blackboard until the Sunday 23:59 pm of the lab week.

3. Background Information

A sequential logic circuit is a type of circuit that uses inputs and previous states stored in the circuit to decide on its next state(s) and the output(s). A logic circuit is classified as a sequential logic circuit if it depends on a previous state, which is stored in a register or a flip-flop for use in future clock cycles. A complete description of a sequential logic circuit includes its state table, state equations, circuit schematics and state diagram. A state table summaries all input/output/state sequence relations for a sequential logic circuit, similar to a truth table summarizing the functionality of combinational logic circuit. Sequential logic circuits are also named as finite-state machines or state machines.

The key distinct property of sequential logic circuits from combinational logic circuits is that sequential circuits depend on the states or inputs or outputs from previous clock cycles, while combinational logic circuits only operate on the inputs provided with the current clock cycle. Combinational logic circuits do not have memory. In combinational logic circuits, the outputs are only functions of inputs of the current clock cycle, while in sequential logic circuits, the outputs and the states are functions of current and/or previous inputs and states.

The states of sequential logic circuit are defined as single or multiple bit binary logic values. These values are stored in flip-flops or registers. Flip-flops are digital circuit components that can store a digital bit value and deliver this bit as its output in the next clock cycle. A sequential circuit has one flip-flop for each distinct state variable.

When analyzing a sequential logic circuit, we first construct the state table that consists of inputs and present states as well as the corresponding next states and outputs. Once this table is constructed, the next states (flip-flop outputs; D_A, D_B, …) are represented using equations in terms of the inputs and the previous states (A, B,…). Once the next states are represented using equations as the sum of minterms

from the state table, standard Boolean algebra could be used to simplify the expressions as much as possible. These expressions are then used to construct the circuit schematics similar to the combinational logic circuit design steps. A summary and examples of these procedures are provided in Part 5.5 of the textbook (M. Morris Mano, Michael D. Ciletti, Digital Design with an Introduction to the Verilog HDL, VHDL and SystemVerilog Global Edition, Pearson, Harlow, Sixth Edition).

In order to draw the state diagram, you first list all of the possible binary state values using graph notation as shown in Fig. 5.19 of the textbook. To draw the state diagram, first write the state values such as 00, 01, 10, 11 and circle them. Then connect these values from previous state value to the next state value and write the inputs/outputs on the arrows. Repeat this until all states are represented.

Sequential Circuit Design

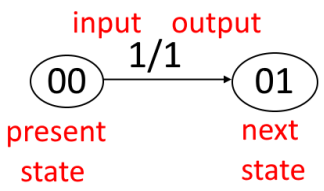
When designing the sequential logic circuit from a verbal description as required for this lab, we start by identifying the number of state variables that the circuit should have. The number of state variables is determined from the problem description. For instance, if a machine has three operational modes circulating one after another (A → B → C → A), then each state can be represented using two state variables (S_1S_0) whose values can span the state space (A = 00, B = 01, C = 10 and S_1S_0 = 11 is an unused state). Then, the state table is constructed based on the input/output and state transition definitions in the problem description. If the transition from A to B occurs when a START button is pressed and an indicator LED only for B state is lit, then we define an input bit START and an output bit for B state LED output Z_0 and indicate this transition on the state table below:

Table 1. State table entry for mapping a transition from one state to another. The table indicates the transition upon pressing the START button (START = 1) from present state A (S_1S_0 = 00) to the next state B (S_1S_0 = 01) and deliver Z_0 = 1 in this new state.

(INPUTS) START	Present state S_1	Present state S_0	Next state S_1	Next state S_0	(OUTPUTS) Z_0
1	0	0	0	1	1

Next, we complete the rest of the states on the table based on the verbal description of the transitions since there are seven more rows that need to be mapped to a next state or an output. If there are unspecified state transitions, then assume the system returns to its original starting state. After the state table is complete, draw a graph notation representation as shown in Fig. 1. Add these arrows for all rows of the state table. After the state diagram is complete, use a K-map to describe each next state variable in terms of the inputs and all present state variables. After using Boolean algebra to obtain the simplest expressions for the state equations, use flip-flops for each state variable and draw the circuit schematics that implements the sequential logic problem.

Fig. 1. State diagram for the row shown on Table 1.



4. Preliminary Work

For the preliminary work, prepare a state diagram representation of the washing machine that has the functionality below:

There are 4 states indicated using the leftmost LED's (LED9-LED6) and the 2 output bits using LED1-LED0.

+ The washing machine has four states of operation: *WASH*, *SPIN*, *RINSE, IDLE.*
+ The washing machine must be a synchronous circuit with clock signal CK. There are two external 1-bit inputs: *START* and *RESET* buttons. The external output is a 2-bit signal S_1S_0 that shows which state the washing machine is in (*WASH*, *SPIN*, *RINSE, IDLE are 00, 01, 10, 11, respectively*).
+ *WASH*, *SPIN* and *RINSE* states use a down counter. The counters all use a 1-bit internal variable *DEC*. The counters synchronously decrement once each second for *DEC* = 1, but can be synchronously reset on any cycle of clock *CK*. It has a single internal output, *ZERO*, which is 1 whenever the counter reaches zero and is 0 otherwise.
+ The counters for *WASH, SPIN* and *RINSE* states count for 4, 3 and 2 seconds, respectively.
+ When the machine starts, it always starts with the *IDLE* state. When *START* button is pressed, the system transitions from *IDLE* to *WASH* state and DEC = 1.
+ Next, the counter for *WASH* state starts counting down (S_1S_0 = 00). As soon as this counter reaches zero, *ZERO* becomes 1, the counter for *WASH* state resets and the system transitions to *SPIN* state.
+ Then, the counter for *SPIN* state (S_1S_0 = 01) starts counting down. As soon as this counter reaches zero, *ZERO* becomes 1 again, the counter for *SPIN* state resets and the system transitions to *RINSE* state.
+ In this final operational state which is *RINSE* (S_1S_0 = 10), the counter for *RINSE* state starts counting down. As soon as this counter reaches zero, *ZERO* becomes 1 again, the counter for *SPIN* state resets and the system transitions to *RINSE* state.
+ *RESET* input aborts the entire washing procedure, sets DEC = 0, and the machine resets to the *IDLE* state (S_1S_0 = 11).
+ *For all other unspecified state and output conditions (i.e. implementation of asynchronous input), any solution is accepted*.

5. Experimental Work

For the experimental work, write the VHDL code, which demonstrates the washing machine controller using the necessary LED's and input buttons. In your lab reports, make sure include the photos of the FPGA boards indicating each state (WASH, SPIN, RINSE, IDLE) and upload a short video of the FPGA demo with your lab report.

6. Grading

Your labs are graded as follows:

Total lab grade = Preliminary work* (30%) + Lab interview and demo* (40%) + Lab report (30%)

- <u>You may not enter the lab if you do not finish the preliminary work before the lab.</u>
  * (asterisk) means these items are essential for passing the lab.
- **Deadlines for preliminary work:** The previous Thursday 5pm before the labs is the deadline to upload the preliminary work on Blackboard.
- **Deadlines for lab reports:** The first Sunday 23:59pm after the labs is the deadline to upload the lab reports on Blackboard.

Further reading:

1. Chapter 5 of the textbook for sequential logic circuit design and analysis (M. Morris Mano, Michael D. Ciletti, Digital Design with an Introduction to the Verilog HDL, VHDL and SystemVerilog Global Edition, Pearson, Harlow, Sixth Edition).
2. Parts 6.4 and 6.5 of the textbook for counter circuits (M. Morris Mano, Michael D. Ciletti, Digital Design with an Introduction to the Verilog HDL, VHDL and SystemVerilog Global Edition, Pearson, Harlow, Sixth Edition).
3. Chapters 1 and 3 on sequential logic circuit implementation using VHDL in Douglas L. Perry, VHDL Programming by Example, McGraw-Hill, New York.