ECE 85L Digital Logic Design Laboratory

Fresno State, Lyles College of Engineering
Electrical and Computer Engineering Department
Spring 2015

Laboratory 14 – Synchronous State Machine Design

1. OBJECTIVES

- Understand the Basic Operation of a Synchronous State Machine
- Develop a State Diagram and Next-State Table from the Design Constraints
- Implement the Full Design using a PLD

2. DISCUSSION

2.1 Synchronous State Machines

This laboratory will entail the implementation of a complete synchronous state machine using a Programmable Logic Device (PLD). Specifically, you will design the tail light system for a 1965 Ford Thunderbird motor vehicle, which from an engineering standpoint, is considered to be one of the more interesting automotive electronic backsides ever created. Since the machine is relatively complex, implementing it in discrete components would necessitate a substantial amount of wiring, circuitry, and subsequent debugging. As a result, we will implement the entire machine using PLDs with the AMTEL ATF750C.

Recall that a synchronous state machine conceptually consists of three subsystems:

1. Memory – Flip-Flops
2. Input-Forming Logic (**IFL**) – Combinatorial
3. Output-Forming Logic (**OFL**) – Combinatorial

The subsystems are interconnected as shown below:

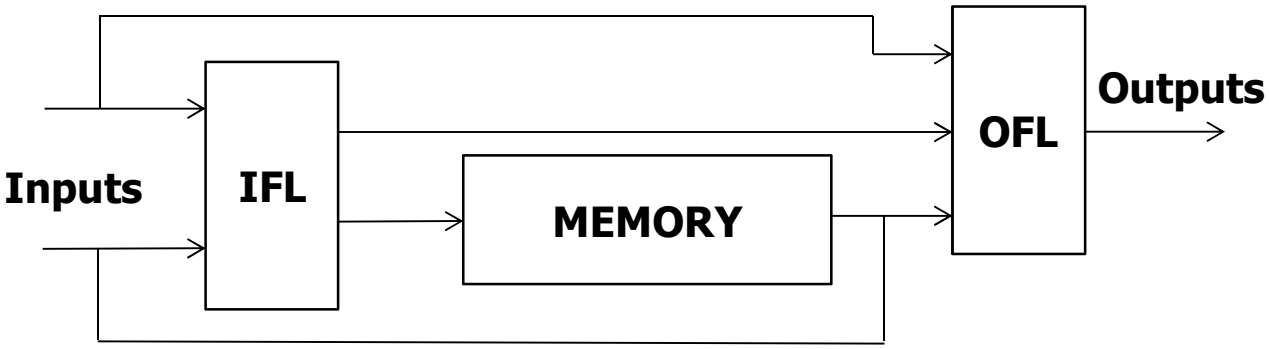

Figure 2.1: Block Diagram of a Synchronous State Machine

A State Machine can be categorized as Mealy or Moore depending upon whether the outputs change in direct response to input or state changes, or in response to state changes only (the Moore Machine eliminates paths from the Inputs or the Input Forming Logic directly to the Output Forming Logic).

2.2 T-Bird Tail Light Description

Figure 2.2 displays the rear of a 1965 Ford Thunderbird. As can be seen there are three lights on each side. The state machine which controls the lights has two input signals, **LEFT** and **RIGHT** that indicate the driver's request for a left turn or a right turn. The flashing sequence for a left turn (**LEFT = 1**) or a right turn (**RIGHT = 1**) also is shown in Figure 2.3. The machine also has an emergency flasher input **HAZ** (hazard) that requests the tail lights to be operated in hazard mode – all 6 lights flashing on and off in unison. Also, if by chance one of the turn signals is operated concurrently with the assertion of **HAZ**, the hazard input should dominate. And lastly in the event that **HAZ** is asserted while in the "middle" of a left or right flashing sequence, the machine should enter hazard mode as soon as possible (the next clock tick). Your Instructor will give you a demonstration of a working system.

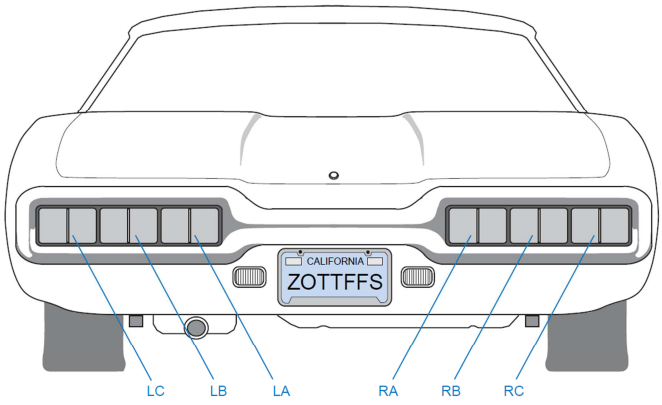

Figure 2.2: Tail Light System of a 1965 Ford Thunderbird

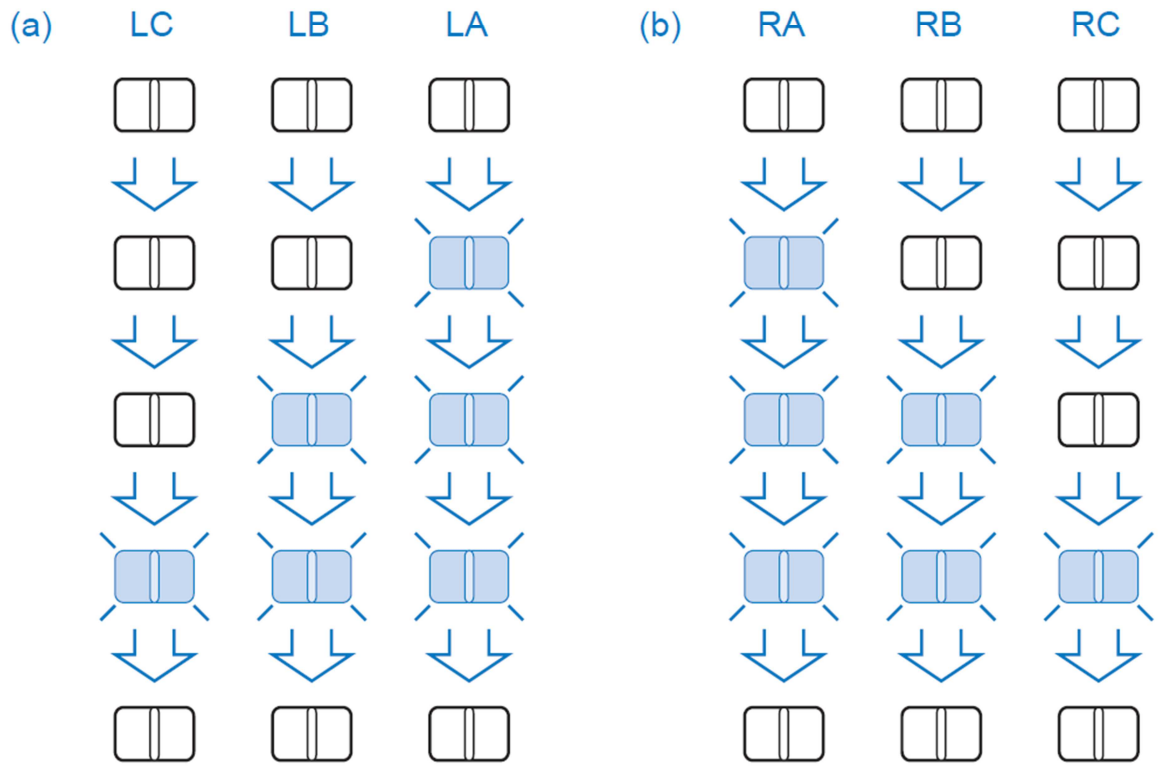

Figure 2.3: The 1965 Thunderbird Tail-Light System. (a) Represents the sequence for a left turn, and (b) Represents the sequence for a right turn. In both cases, the Left/Right Turn Sequence repeats regularly as long as the signal is active. When the Hazard input is asserted, all 6 lights should repeatedly blink on and off.

2.3 PLD Implementation of a Synchronous State Machine

Referring to the PLD modes of operation on the ATMEL ATF750C specification sheet, it should be clear that the device can be configured to implement D Flip-Flops (see **Registered Output** under **Section 6. Logic Options**) to provide the required memory functions of a Synchronous State Machine. The D input to each Flip-Flop is formed by AND-OR logic (thereby creating SOP expressions), thereby providing the Input Forming Logic. The inputs to the AND-OR array may include external inputs and/or the output states Q_0, Q_1, etc. from each of the D Flip-Flops. Lastly, the Q and Q* outputs from the D Flip-Flops may also be returned to the AND-OR array to provide the Output Forming Logic.

As PLDs are limited in size based upon the number of Flip-Flops, interconnections, product terms etc. that are available within the Integrated Circuit, if the design will not “fit” onto the selected device, the designer has two options:

1. Select a larger device from the same family.
2. Partition the design such that it can be implemented in multiple PLDs.

Because this design project is limited to the ATMEL ATF750C, only Option 2 is viable (if necessary). If your design does not fit on a single PLD, referring to Figure 2.1, it is recommended that the IFL and Memory blocks are partitioned into one chip, while the OFL is partitioned into a second PLD. The outputs from the first PLD can serve as external inputs to the second PLD, as necessary. However, if you minimize your IFL and OFL equations appropriately, the entire design can fit into a single ATMEL ATF750C PLD.

3. PRELAB

Your project involves the design and implementation of a synchronous state machine using 1 or more PLDs to implement the functionality of the Tail-Light System of the 1965 Thunderbird. Since your system will require a clock frequency of approximately 1 Hz, you must design an astable multivibrator with the appropriate frequency using a 555 Counter-Timer chip in advance of the Synchronous State Machine design. Because the 555 timer is not capable of driving a large fan-out, you must buffer the 555 output using a Schmitt trigger buffer (74LS14) prior to its use in your circuit to provide a jitter free, noise-immune clock signal. The output of your system should drive 6 LEDs to emulate the tail section of the Thunderbird.

Though you are free to use any design process, the following approach is suggested:

3. Develop your 555 Timer to create a 1 Hz clock signal with a Duty Cycle as close to 50% as possible. Determine the required resistor and capacitor values necessary to implement your design based upon the components in your lab kit.

4. Examine the specification sheet for the 74LS14 Schmitt Trigger Inverter. Draw a circuit diagram (with all necessary pinouts) that implements your 555 Timer and applies the Output of the Timer to the input of the 74LS14. Since the 7414 inverts your signal, use 2 inverters in series to buffer the clock.

5. After examining Figure 2.3, quantify the number of states necessary for the state machine, as well as the number of inputs and outputs to the system. Categorize the system as Mealy or Moore, again by observation.

6. Develop a State Diagram.

7. From the State Diagram, compute the (minimum) number of Flip-Flops required.

8. Assign each of the observable external states to an internal machine state (a unique Flip-Flop output combination). Although there are procedures for optimizing this assignment, for this exercise you may assign them arbitrarily.

9. Verify the State Diagram with your Instructor before proceeding any further.

4. LAB ASSIGNMENT

4.1 Synchronous State Machine Implementation

1. Once your State Diagram has been verified by your Instructor, produce a Next-State Table. Ensure that you design your machine to be self-starting, such that any used states have a valid and reasonable next state. As in lecture, the left-hand side of your Next State Table should possess the same number of columns as the Sum of the Present State Variables (Q0, Q1, etc.) and the number of inputs. The right-hand side of your Next State Table should possess the same number of columns as the Sum of the Next State Variables (Q0, Q1, etc.).

2. Assuming D Flip-Flops are used to create the memory cells, develop the Sum-of-Product Expressions (SOP) necessary to implement the Input Forming Logic to each of the Flip-Flops. The Input Variables used for your Input Forming Logic should include the external Inputs (Left, Right, and Hazard) as well as the Q/Q* outputs from each of the Flip-Flops, as necessary. The outputs for the IFL are the D inputs to the Flip-Flops.

3. Minimize the IFL SOP expressions using K-Maps to simplify the logic needed to implement the Input Forming Logic.

4. Develop the necessary Output Forming Logic from the output of your States to drive 6 LEDs. Minimize the expressions, as necessary.

5. Examine the ATMEL ATF750C Specification Sheet, ATMEL WinCUPL User's Manual, and WinCUPL Tutorial to determine how to create the necessary source code to implement your Synchronous State Machine.

6. Compile your source code to create a JEDEC output file, which can then be used to program the ATMEL ATF750C using the ATEMEL Device Programmer. When programming your devices, be sure to use the anti-static wrist band to ground yourself, since the CMOS chips are easily damaged by static electricity.

7. Develop a Complete Circuit Diagram (with full pinouts for each chip) for your Synchronous State Machine design.

8. Build and test your system, using your circuit diagram as a guide. Note that the final output of your Synchronous State Machine should drive 6 LEDs to emulate the Tail-Light Section of the 1965 Thunderbird. Use switches for the Boolean

Inputs (LEFT, RIGHT, HAZARD), and the output of the 555 Timer (buffered by the 7414 Inverters) as your Input Clock.

9. Demonstrate your working circuit to your instructor.

10. Document your entire design process.

4.2 Extra Credit

The actual tail lights of many vehicles require a 12 V supply voltage. Since TTL output logic levels are limited to a maximum of 5 V, open-collector buffers and load resistors may be used to interface the output of your digital design to step the maximum voltage up to 12 V. Examine the 74LS06 Open Collector Buffer specification sheet, and modify your design to produce the necessary 12 V output signals. Your circuit may then be interfaced with a transistor circuit (supplied by the Instructor) to supply the large current needed to drive an array of real tail lights on a mock-up board. Demonstrate this to your instructor.

5. REVIEW QUESTIONS

1. What is the difference between a Mealy and a Moore Machine?

2. How can D Flip-Flops be used to implement memory cells?

3. How many potential input variables are required for a system with N Inputs and M States?