



# ECE210-Intro. Digital Logic Design

Fall 2024

## LAB4: Traffic Intersection Controller

### DATES

Date	Section	Time
Nov 26-2024	D21	08:00 - 10:50 AM
	D25	02:00 - 04:50 PM
Nov 27-2024	D31	08:00 - 10:50 AM
Nov 28-2024	D45	02:00 - 04:50 PM
Dec 03-2024	D22	08:00 - 10:50 AM
	D26	02:00 - 04:50 PM
Dec 04-2024	D32	08:00 - 10:50 AM
Dec 05-2024	D46	02:00 - 04:50 PM

### INTRODUCTION

Logic circuits are classified into two types, combinational and sequential. A combinational circuit is one whose outputs depend only on the current inputs. A sequential circuit is one whose outputs depend not only on its current inputs, but also on the past sequence of inputs. Until now, we have looked exclusively at combinational circuit design. Today we will focus on techniques for implementing sequential circuits.

### LEARNING OBJECTIVES

In this lab students will design and implement a traffic controlling system enabling Secondary Highway traffic crossing Primary Highway fulfilling specifications.



# ECE210-Intro. Digital Logic Design

## LAB4: Traffic Intersection Controller

Fall 2024

### PRE-LAB

You are expected to read the lab instructions and complete the pre-lab assignments before attending the lab sessions. You are responsible for uploading your pre-lab assignment to eClass.

- Design a Moore finite state machine to meet the specifications in this lab. It should be an 8-state machine using 3 D-Type flip-flops (One for Q0, one for Q1 and one for Q2). Please fill out the given tables and derive your solutions based on them. Include the following in your solution:
  - Complete State table (or State Transition Table) and Output table.
  - Solved K-maps for D0, D1 and D2 which describe the next state (Hint – 5 variable k-map).
  - Solved K-maps for the outputs (SG, SY, SR, HG, HY, HR, CD2, CD1 and CD0).
  - Flip-flop input and output equations (Equations for D0, D1 and D2 based on the inputs (EMV and Change) and current state (Q2, Q1 and Q0)).
  - Handwritten or electronic version of VHDL solution.

**Note:** You do not need to consider the “Reset” functionality in your tables. This function is already implemented for you in the code.

It is very important that you complete the pre-lab components before you come to the lab. Failing to do so will impede your ability to complete the lab within the allotted time.

### Outputs based on Current State

Current State				Outputs								
Symbol	Q			Secondary HW Lights			Primary HW Lights			Countdown Display (CD)		
	Q2	Q1	Q0	SG	SY	SR	HG	HY	HR	CD2	CD1	CD0
S0	0	0	0									
S1	0	0	1									
S2	0	1	0									
S3	0	1	1									
S4	1	0	0									
S5	1	0	1									
S6	1	1	0									
S7	1	1	1									



# ECE210-Intro. Digital Logic Design

## LAB4: Traffic Intersection Controller

Fall 2024

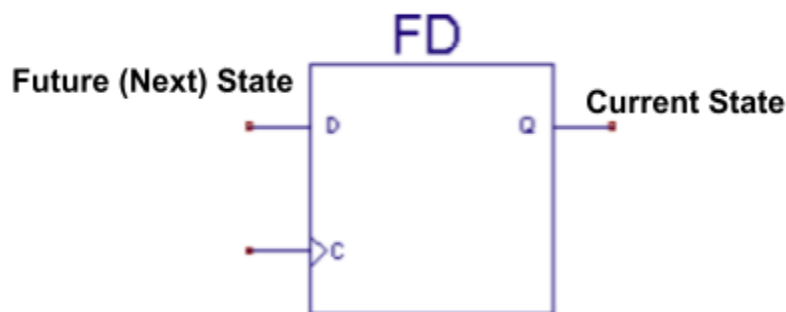
**State transition table**

Current State (Q)				Inputs		Next State (D)			
Symbol	Q2	Q1	Q0	Change	EMV	D2	D1	D0	Symbol
S0	0	0	0	0	0	0	0	0	S0
	0	0	0	0	1	0	0	0	S0
	0	0	0	1	0	0	0	1	S1
	0	0	0	1	1				
S1	0	0	1	0	0				
	0	0	1	0	1				
	0	0	1	1	0				
	0	0	1	1	1				
S2	0	1	0	0	0				
	0	1	0	0	1				
	0	1	0	1	0				
	0	1	0	1	1				
S3	0	1	1	0	0				
	0	1	1	0	1				
	0	1	1	1	0				
	0	1	1	1	1				
S4	1	0	0	0	0				
	1	0	0	0	1				
	1	0	0	1	0				
	1	0	0	1	1				
S5	1	0	1	0	0				
	1	0	1	0	1				
	1	0	1	1	0				
	1	0	1	1	1				
S6	1	1	0	0	0				
	1	1	0	0	1				
	1	1	0	1	0				
	1	1	0	1	1				
S7	1	1	1	0	0				
	1	1	1	0	1				
	1	1	1	1	0				
	1	1	1	1	1				



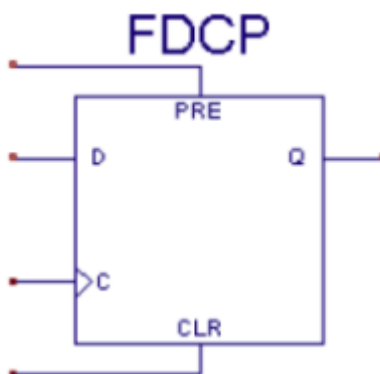
### BACKGROUND

In this lab you will use a common device known as a Flip-Flop, as the memory element in your state machine. Flip-Flops use feedback logic to store a digital value so long as there is power applied to the circuit. There are a number of different types of Flip-Flops, in this lab we will deal with the D-Type Flip-Flop. Below is the schematic symbol for a basic D Flip-Flop.



The inputs to the device are D (data) and C (clock). On each rising edge of the clock signal, the value present on the data input will be latched and held at the output, Q, until the next clock edge. At that time, whatever value is present at D will be transferred to Q.

A variation of the basic D-FF is shown below with PRE (preset) and CLR (clear) inputs. The preset and clear inputs are used to force the value of the output, Q, to a certain state. Activating the preset signal will cause the output to go to logic '1'. Asserting the clear signal will cause the output to go to logic '0'. Note: We will not use this variation in this lab.



Flip-Flops are typically used as storage elements, latch or buffer in digital electronics. An SRAM is made up of Flip-Flops with each device storing 1-bit of information.



### TRAFFIC LIGHT CONTROLLER

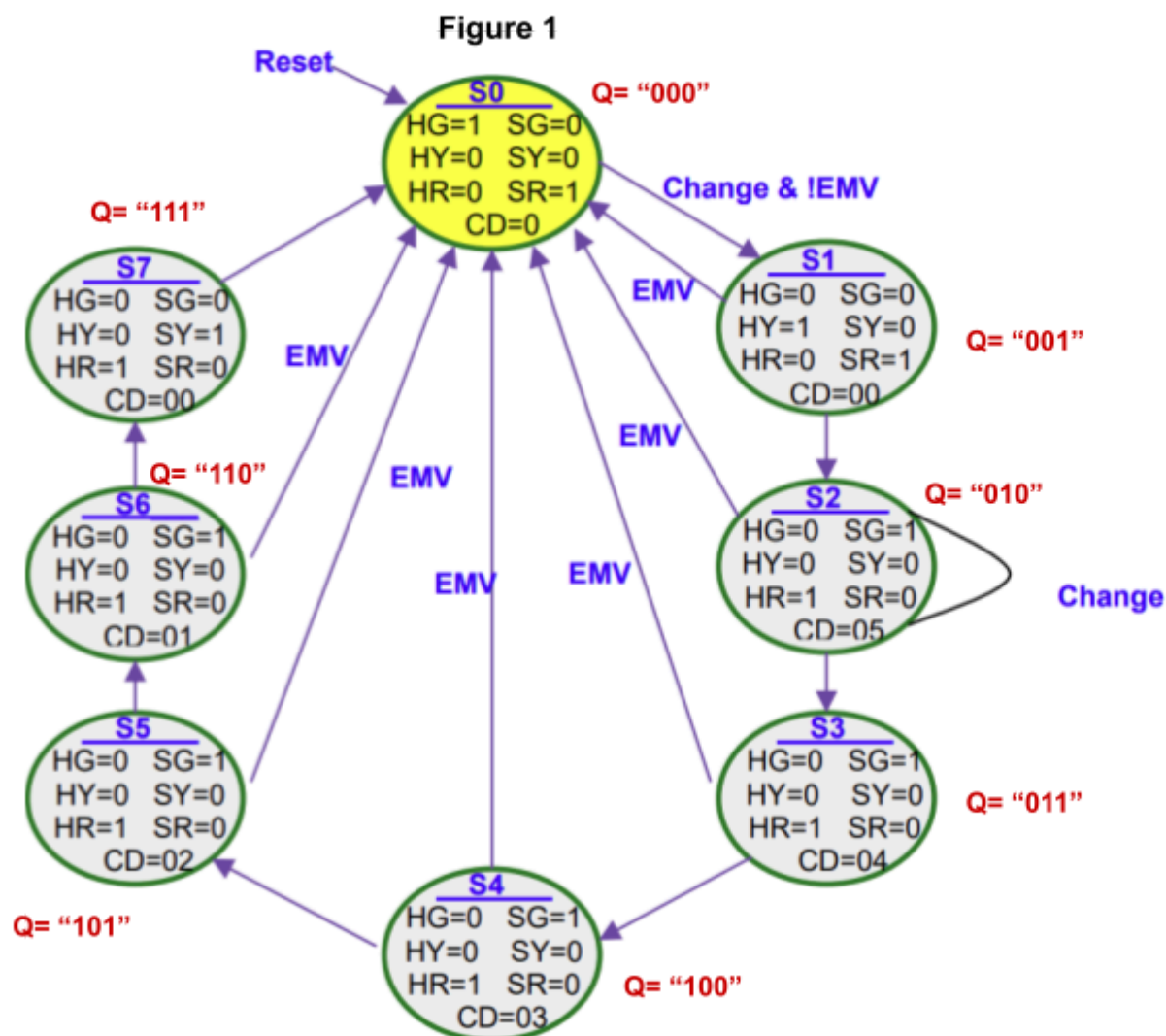
There has been a problem for those traveling through secondary roads across very busy primary highways. To alleviate this problem, a decision was made to design a traffic light controller for highways passing through rural areas. To achieve this feat, you have been hired in the hope that you can design and implement a solution before the end of winter.

Several locations along Highway 16 have been targeted for testing the controller. In each location, detectors are to be placed along the secondary road coupled with some simple timer logic to raise a signal when a vehicle is waiting to cross the highway. A second set of detectors, intelligent enough to discern between a regular vehicle and an Emergency Vehicle, will be placed on the main highway. A special signal will be raised when an emergency vehicle is approaching (see Figure 1). In addition, 7 Segment counters are placed on the secondary highway which will count down from five to zero before cycling the light to red.

The traffic light controller should operate with the following criteria:

1. If no vehicle is detected on the secondary road, the highway lights should be green, while the secondary road lights should be red. The 7-segment should display a 0.
2. If a vehicle is detected on the secondary road, the change signal will be asserted and the highway lights should cycle through yellow, then red. The secondary road lights will then turn green. Upon turning green, the 7-Segment will display a 5.
3. The secondary road lights will remain green so long as a vehicle is detected on the secondary road. The 7-segment will remain at displaying a 5.
4. When no further cars are detected on the secondary road, the 7-Segment will start counting down in 1 second intervals from 5 to 0 in descending order. Upon hitting 0, the light will cycle from yellow to red. The highway light will then turn green.
5. If an emergency vehicle is detected on the primary highway at any time, the EMV signal will be asserted and the highway lights should immediately (next clock cycle) switch to green, while the secondary road lights should turn red. The 7-segment will reset to 0.

To help you understand the existing design, the following summary is provided. Your task is to design, test and implement the Moore Finite State Machine (FSM) circuitry, which drives the traffic light cycles. Figure 1 illustrates the state diagram for your project (an explanation of these figures follows):




In this figure, the condition needed for a transition is written on the corresponding arrow. For example, we expect a transition from S0 to S1 if "Change & !EVM", or in other words "Change=1" and "EVM=0". Here is the description of the state diagram inputs and outputs:

### Inputs:

Reset	Places controller in its initial state (Primary highway green, Secondary road red)
Change	Goes high when a vehicle is detected on the secondary road. [SW0].
EMV	Goes high when an emergency vehicle is detected on the primary highway. [SW3]



	ECE210-Intro. Digital Logic Design
	LAB4: Traffic Intersection Controller
Fall 2024	

### Outputs:

HG, HY, HR	Green, yellow and red Primary Highway lights, respectively
SG, SY, SR	Green, yellow and red Secondary Highway lights, respectively.
CD	Countdown displayed on 7-segments.

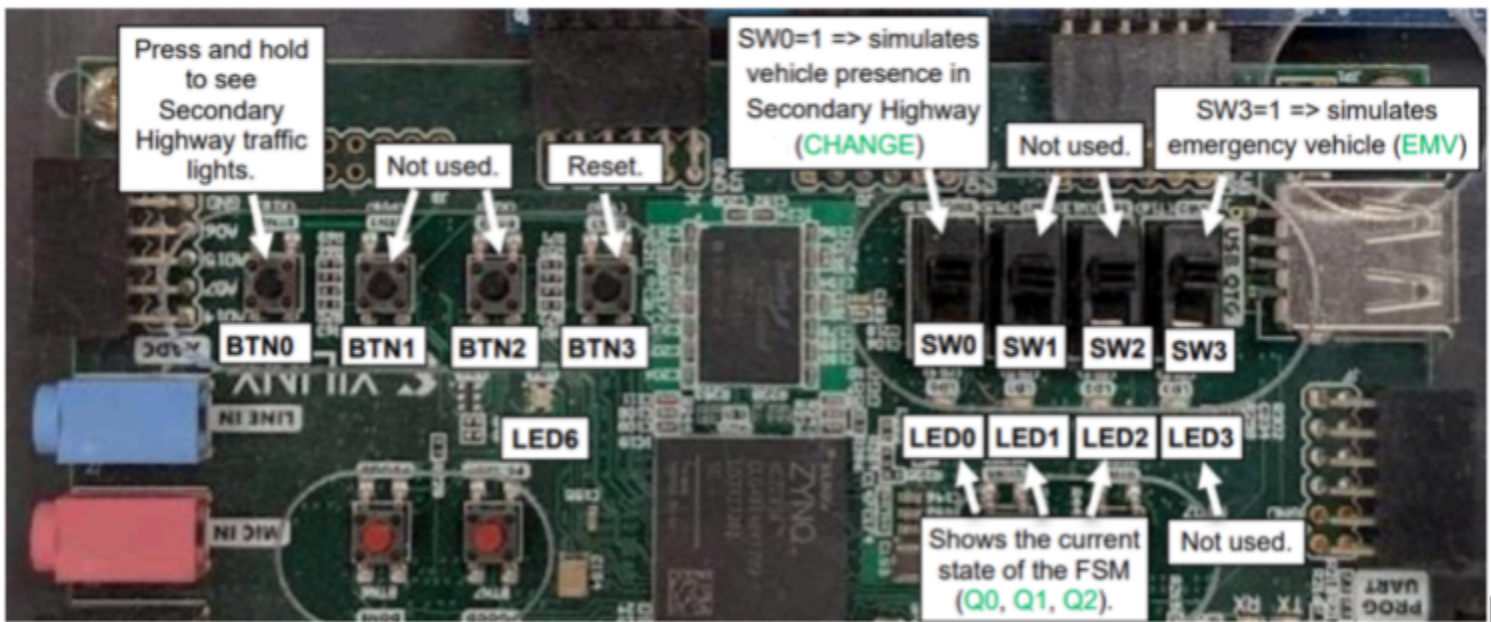


Figure 2

### Figure 2 - Details:

- **BTN0:** Since on this Zybo Z7 board there is only one RGB led (LED6), so the given design requires pressing and holding this button to see traffic lights status of Secondary Highway.
- **BTN1 and BTN2:** Not used in this design.
- **BTN3:** Reset back to S0.
- **SW0:** Simulates vehicles presence on Secondary Highway.
- **SW1 and SW2:** Not used in this design.
- **SW3:** Simulates emergency vehicle presence on Primary Highway.
- **LED6:** Showing intersection lights status, default=>Primary, BTN0 pressed=>Secondary.
- **LED0 to LED2:** Shows the current state of the FSM.
- **LED3:** Not used in this design.



# ECE210-Intro. Digital Logic Design

## LAB4: Traffic Intersection Controller

Fall 2024

### REQUIREMENTS

---

Download the project template from eClass and complete the design. Figure 2 shows the resources used in this project. Demonstrate your simulation and working implementation to a TA to receive a check-off.

**Note:** This lab does not require a formal report

### REFERENCES

---

- [Zybo Z7 Documentation, Tutorials and Example Projects](#)
- [Zybo Z7 Reference Manual](#)
- M.M. Mano. Computer Engineering: Hardware Design, (Prentice Hall, 1988).
- M Zwolinski, Digital System Design with VHDL, (Prentice Hall, 2000).





# ECE210-Intro. Digital Logic Design

## LAB4: Traffic Intersection Controller

Fall 2024

### MARKING GUIDELINES

---

Student Names:\_\_\_\_\_

\_\_\_\_\_

#### Pre-Lab

- (10) State Table (or State Transition Table)
- (10) K-Maps
- (15) Next State & Output Equations
- (05) Handwritten or electronic version of VHDL solution(s)

Sub-total\_\_\_\_\_/40

#### In-Lab

- (25) Traffic Light Changes Correctly
- (13) Side Road Light Change Countdown
- (12) Correct EMV Detection
- (10) State Reset

TA Signatures:\_\_\_\_\_ Sub-total\_\_\_\_\_/60

Lab Report Total\_\_\_\_\_/100