Digital Design Lab
EEN 315 Section 3G

Lab 1
Seven Segment Display and Sequence Detector

Group 5
Nathan Paternoster
(Partner: Michael Finale, Daniel Bennett)

Sayan Maity, TA

University of Miami

September 26th, 2013

Abstract

The purpose of this lab is to design and build a seven segment display and a basic sequence detector. Both elements must be built using only NAND gates, D flip-flops, and LED's. This lab is used as an introduction to common lab equipment such as the breadboard, digital probe, IDL 800 digital lab trainer kit, and TTL chips. Minimal design is a key point in designing the seven segment display in order to efficiently build its circuit.

Table of Contents

Overview

A seven segment display is a very commonly used circuit that uses 7 separate LED's to represent a single decimal digit. To design one you must consider which LED's must be lit to display each digit. The sequence detector must be built to detect a specified sequence of bits and will use the seven segment display to signal when that sequence is seen.

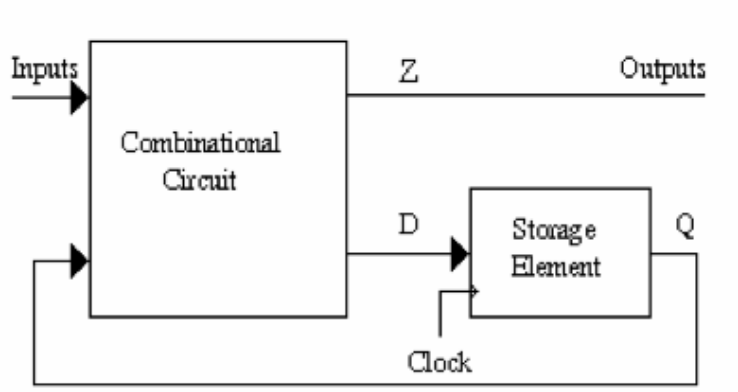


Figure 1
Block Diagram of a Basic Sequential Machine.

In this figure the combinational circuit represents the seven segment display. Adding a storage element (flip-flops) to the circuit allows the use of a sequence detector.

Objectives

To illustrate the principles of both combinational and sequential logic design and provide an understanding of logic circuitry components.

Equipment

Description	Chip Number	Quantity
2-Input NAND gate	7400	6
3-Input NAND gate	7410	7
4-Input NAND gate	7420	4
Dual positive-edge triggered D flip-flops	7474	2

Description

The first step to design the seven segment display is to write a truth table that contains all 10 digits (0-9) as the binary encoded input. This will require four bits to represent. The output will be all seven segments a through g. You must determine which segments are to be lit up for each decimal number and mark those segments with a 1 accordingly.

The second step is to find a logical output equation for each segment. Use a 4-variable K-map for each output function a-g to solve for the minimal equation. With the seven output equations you may sketch out the circuit. Use only NAND logic to achieve all the NOT/AND/OR operations required.

The first step to design the sequence detector is to draw a next state diagram. We will need four states to detect the input sequence '101', repeats allowed. After the state diagram is drawn, the second step is to write a next state table with a single input (our sequence detector will light up a single LED to show that the sequence has been detected).

The third step is to give each state a binary-coded decimal representation and write the transition table. Two input variables will be required to represent the four states. Using Moore method, this table will have a third column for the output.

Next, two flip-flop equations must be created (one for each input variable). Use the D flip-flop characteristic table to fill the K-map for each flip-flop. After the flip-flop equations have been found, a circuit may be drawn.

Specifications

Must use only NAND gates and only D flip-flops.

Design Synthesis

Step 1 of designing the seven segment display: the truth table. The numbers on the left represent the digits to be displayed, the X input variables represent the binary coded decimal versions of the digits, and the a-g output variables represent each segment of the seven segment display. After 9 there should be no segments lighting up because all other inputs are invalid.

	X_0	X_1	X_2	X_3	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1
10	1	0	1	0	0	0	0	0	0	0	0
11	1	0	1	1				…			

Step 2 of designing the seven segment display: the output equations. Each vertical column, a, b, c, etc., will have its own output equation and will require its own K-map. Groups of 1's must be made in the most minimal fashion possible to create the most efficient circuit.

X_2X_3 \ X_0X_1	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	0

$a = X_0'X_2 + X_0'X_1X_3 + X_0X_1'X_2' + X_0'X_1'X_3'$

X_2X_3 \ X_0X_1	00	01	11	10
00	1	1	0	1
01	1	0	0	1
11	1	1	0	0
10	1	0	0	0

$b = X_0'X_1' + X_1'X_2' + X_0'X_2'X_3' + X_0'X_2X_3$

X_2X_3 \ X_0X_1	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	0	0
10	0	1	0	0

$c = X_0'X_1 + X_0'X_3 + X_1'X_2'$

X_2X_3 \ X_0X_1	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	0	0	0
10	1	1	0	0

$d = X_0'X_1'X_2 + X_0'X_2X_3' + X_0'X_1'X_2' + X_1'X_2'X_3' + X_0'X_1X_2'X_3$

X_2X_3 \ X_0X_1	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	0	0	0	0
10	1	1	0	0

$e = X_1'X_2'X_3' + X_0'X_2X_3'$

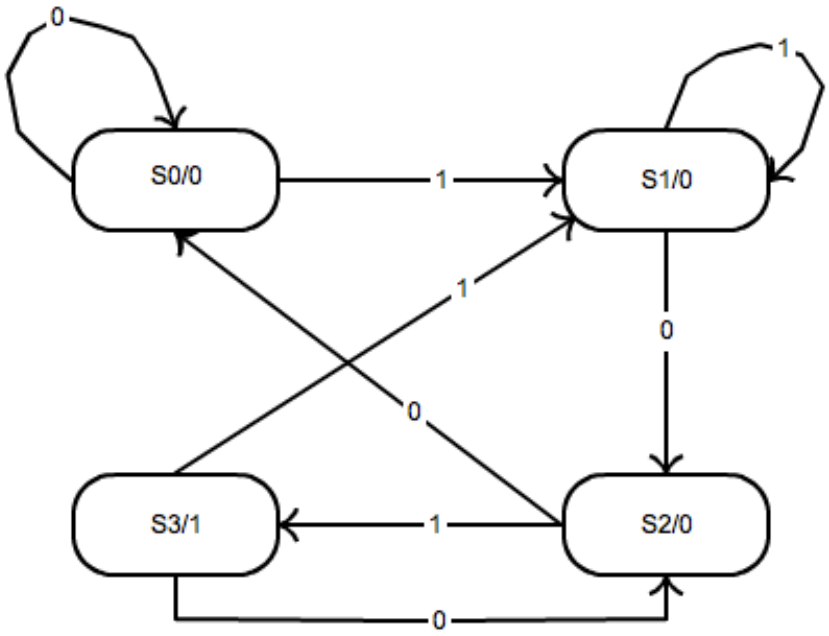
X_2X_3 \ X_0X_1	00	01	11	10
00	1	1	0	1
01	0	1	0	1
11	0	0	0	0
10	0	1	0	0

$f = X_0'X_1X_3' + X_0'X_1X_2' + X_0X_1'X_2' + X_1'X_2'X_3'$

X_2X_3 \ X_0X_1	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	1	0	0	0
10	1	1	0	0

$g = X_0'X_1X_2' + X_0X_1'X_2' + X_0'X_1'X_2 + X_0'X_2X_3'$

Step 1 of designing the sequence detector: the next state diagram.



After the diagram has been drawn, step 2 is to write the next state table:

	X = 0	X = 1	Z
S_0	S_0	S_1	0
S_1	S_2	S_1	0
S_2	S_0	S_3	0
S_3	S_2	S_1	1

The next state table. Z is the output at each state. The Moore Model is used.

Step 3 of designing the sequence detector: the transition table. Each state is given a binary-coded decimal. Z is again the output.

	Y_0	Y_1	X = 0	X = 1	Z
S_0 =	0	0	00	01	0
S_1 =	0	1	10	01	0
S_2 =	1	0	00	11	0
S_3 =	1	1	10	01	1

Step 4 of designing the sequence: the output equations. There will be two output equations D_0 and D_1 for the two flip-flops and one output equation Z for the entire sequence detector. For D_0 the first input column, Y_0, will be compared to the first bit of both X=0 and X=1 and compared to the D flip-flop characteristic table. D_1 will be solved the same way using Y_1 and the second bit of both X=0 and X=1. The output equation Z contains only one minterm so it may be written immediately. K-maps will once again be used.

		Y_0Y_1 00	01	11	10
X	0	0	1	1	0
	1	0	0	0	1

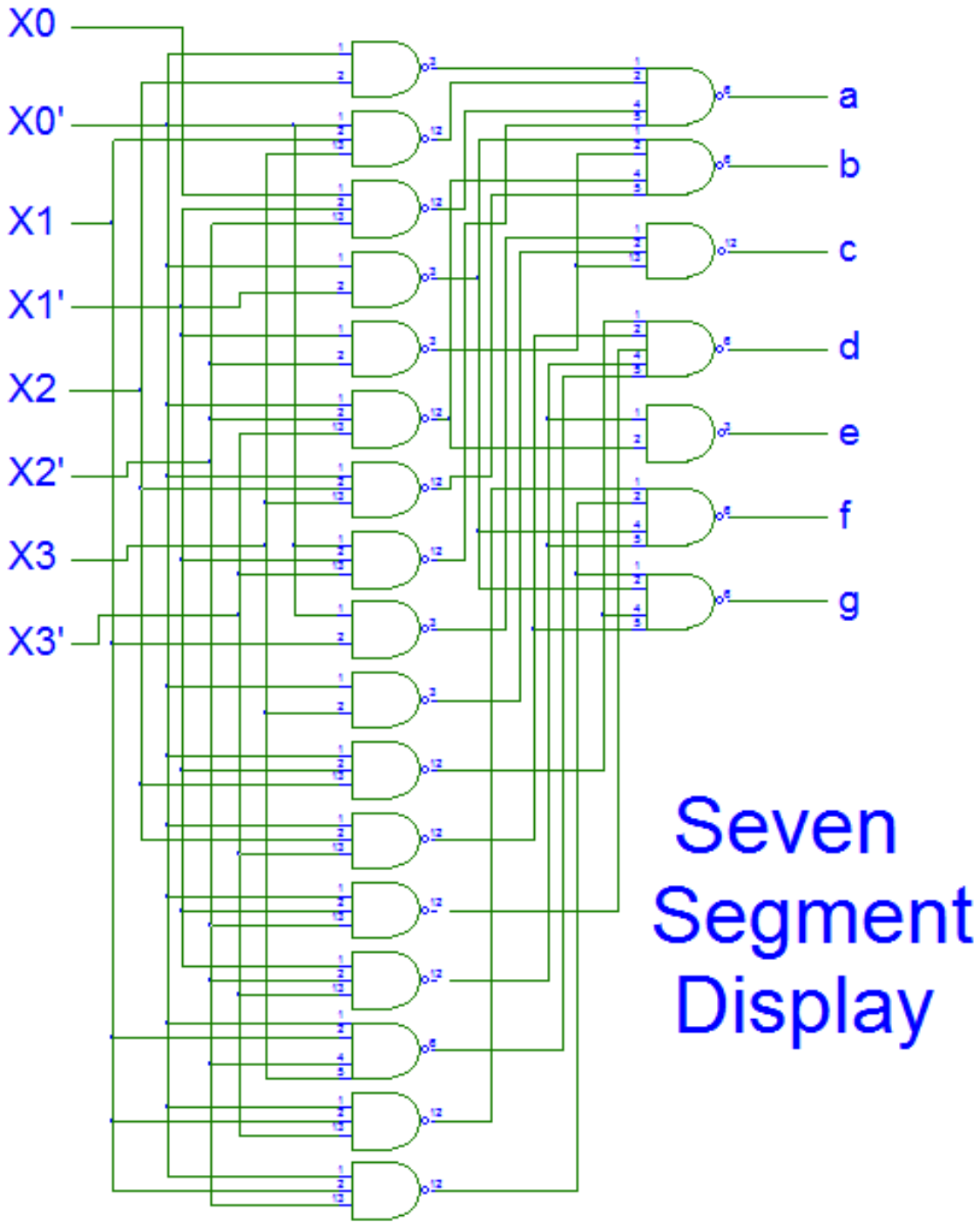
$D_0 = Y_1'X' + Y_0Y_1'X$

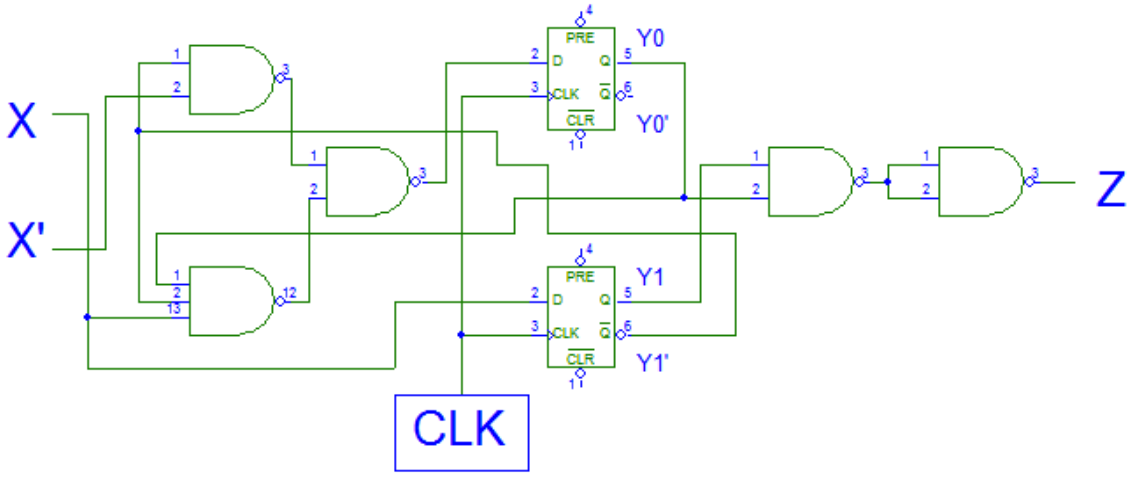
		Y_0Y_1 00	01	11	10
X	0	0	0	0	0
	1	1	1	1	1

$D_1 = X$

$Z = Y_0Y_1$

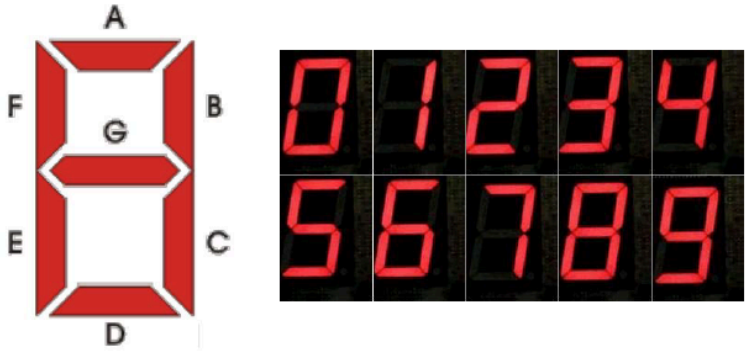
Complete Logic Diagram



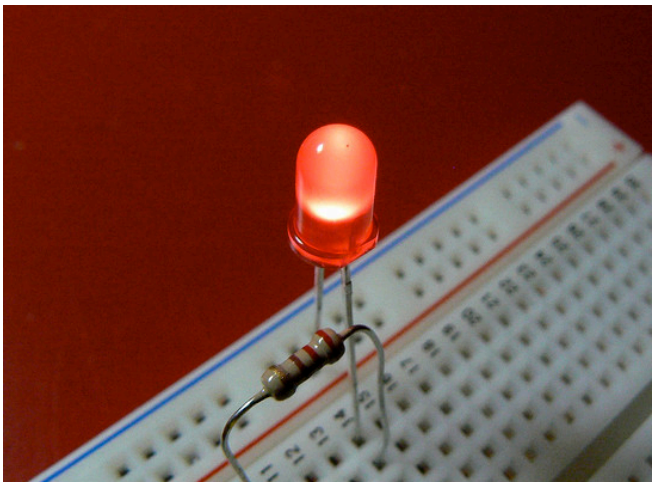


Sequence Detector

Results and Simulations



The final seven segment display will be able to display all digits as shown using four input variables (four switches). When any other combination beyond 9 is entered, there will be nothing displayed.

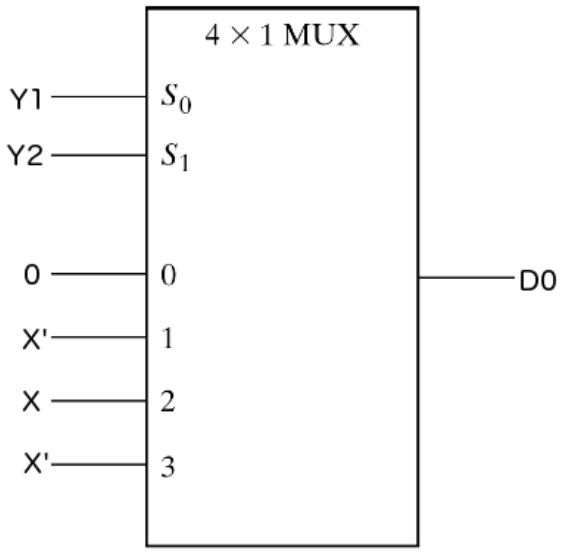


The sequence detector will show a light red diode when ‘101’ is detected.

Answers to the questions in the lab handout

1. A multiplexer may be used to implement Boolean functions in a combinational circuit. To create D_0, we should use a truth table to construct our multiplexer. Y_1 and Y_2 will be the two select lines and X will be the input.

Y_1	Y_2	X	D_0	
0	0	0	0	$D_0 = 0$
0	0	1	0	
0	1	0	1	$D_0 = X'$
0	1	1	0	
1	0	0	0	$D_0 = X$
1	0	1	1	
1	1	0	1	$D_0 = X'$
1	1	1	0	



The D_1 flip-flop will take just the input X. The output equation will be created in the same way.

2. A vending machine may use a sequence detector to detect what combination of buttons you input when you make your purchase decision. Similarly, a remote control may use a sequence detector to detect what channel you are trying to change to.

Conclusion

This lab provided us with new experience using TTL's and the digital lab trainer kit. It gives me a much better understanding of how circuits use logic to achieve the desired result. The difference between combinational and sequential circuits was made clearer by this lab.

Works Cited

None

Signed OFF

[Attached]