Verilog Implementation

EE 241 - Session 1, Spring 2019

Introduction:

Line Decoders find widespread use in digital circuits especially in memory decoding or data routing applications. Figure 1 shows a simplified logic schematic of a 3 to 8 line decoder. It has 3 inputs x_0 through x_2 and 8 outputs namely y_0 through y_7. Inputs determine which output will be set to logic-1 and the remaining outputs will stay at logic-0.

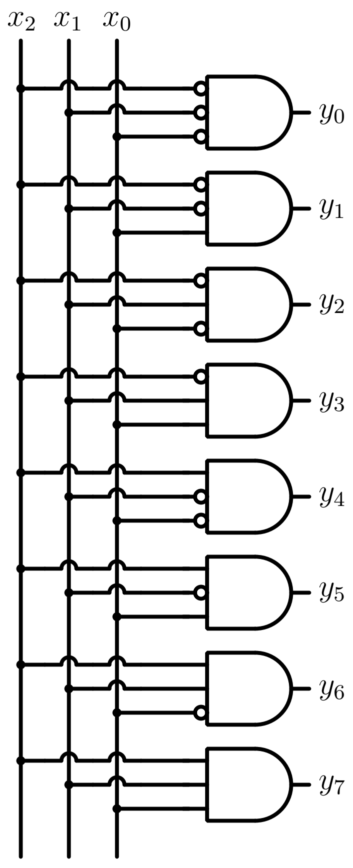

Figure 1: 3-8 Line Decoder.

Procedure:

- Using the schematic shown in Figure 1, implement 3-8 line decoder on Basys board. Use sw2 through sw0 as inputs and LED7 through LED0 as outputs.
- Apply De Morgan Rule to the Boolean Functions found in previous section and assign new functions to the outputs.

Arithmetic Operations

EE 241 - Session 2, Spring 2019

Preliminary Work:

- Write a verilog module which calculates the following function:

$$f_{(x,y,z)} = z + (x \times y)$$

  where inputs x,y are 2-bit numbers and z is a 4-bit number.

- Write a testbench for your module showing 5 different input combinations. Verify that your calculator works properly by checking out the outputs. Take a screenshot of your simulation graph, and upload to Coadsys Assignment named "Session 2 Preliminary Work".

Procedure:

Implement your module on Basys3 board. Use switches as inputs, LEDs as outputs. Input and output assignment is as follows:

- SW7..4 is used for input z.
- SW3..2 is used for input y.
- SW1..0 is used for input x.
- LED[?]..0 is used for output $f_{(x,y,z)}$. Determine how many LEDs should be used.

Temperature Conversion

EE 241 - Session 3, Spring 2019

Preliminary Work:

- Write a verilog module which converts temperature from *celcius* to $fahrenheit$ using the following formula.

$$\frac{celcius}{100} = \frac{fahrenheit - 32}{180}$$

- Temperature range is between 0° - 100°
- Determine the number of bits for defining input and output.

Procedure:

Implement your module on Basys3 board. Use switches as inputs, LEDs as outputs.

2-Bit Multiplier

EE 241 - Session 4, Spring 2019

In this experiment you will implement a binary multiplier circuit that takes two 2-bit binary numbers as input and provides the multiplication result as a binary number at its output.

Preliminary Work:

Design a multiplier circuit that matches the definition given in the Introduction section above:

- Decide how many inputs and outputs such a circuit should have (Hint: The number of required inputs is obvious. For the outputs, consider what the maximum possible product can be when two 2-bit binary numbers are multiplied, and accordingly determine how many bits will be needed for representing the output).
- Build a truth table containing the appropriate number of inputs and outputs, and write down the outputs for each input combination.
- Find the Boolean functions corresponding to the outputs. Each bit of the output will have a different Boolean function, but all functions will of course have the same inputs. These output functions together will make up the final multiplier circuit.
- Try to do some minimizations so that the final circuit will be as small as possible.
- Define 2-bit multiplier by using logic gates (you are not allow to use ‘ * ’ operator) and perform a simulation on Vivado. Compare your simulation results with the truth table you have created. If your simulation results match with the table, bring your module, testbench and simulation results to the lab.

Procedure:

Implement the multiplier circuit you have designed using Basys3 Board. The switches on Basys-3 board will provide the inputs to the circuit, and the outputs will be connected to the LEDs. Verify the operation of your circuit by applying all possible input combinations and checking the generated outputs.

Even-Odd Parity Generator

EE 241 - Session 5, Spring 2019

Preliminary Work:

- Write a verilog module which calculates the number of switches set to logic-1 is even or odd.
- Write a testbench which simulates your module. Use 16 different input combinations and observe the output.

Procedure:

Implement your module on Basys-3 board. Use all switches as inputs.

- If the number of switches set to logic-1 is even, led[0] is on.
- If the number of switches set to logic-1 is odd, led[15] is on.

Getting Familiarized with 7-Segment Display

EE 241 - Session 6, Spring 2019

Today, electronic devices are generally equipped with displays either to show necessary results or to make them more user-friendly. The most common component to display numerical values is 7-Segment Display. It usually consists of 7 LEDs for segments, which build up one BCD digit, and an additional LED for the dot. These segments (LEDs) are universally named from a to g as shown in Figure 1.

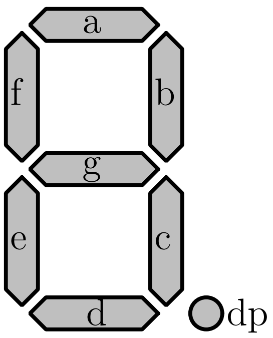

Figure 1: 7-Segment Display

Current flows through LEDs in one direction since they are actually diodes, and this leads to two types of LED displays: Common Cathode (Figure 2a) and Common Anode (Figure 2b).Usually, in order to limit the current flow through the LEDs, a serial resistor is connected to each segment, but a single resistor serially connected to the common node may be enough for an experiment.

The Basys 3 board contains one four-digit common-anode 7-segment LED display. Each of the four digits is composed of seven segments arranged in a “figure 8” pattern, with an LED embedded in each segment. Segment LEDs can be individually illuminated, so any one of 128 patterns can be displayed on a digit by illuminating certain LED segments and leaving the others dark.

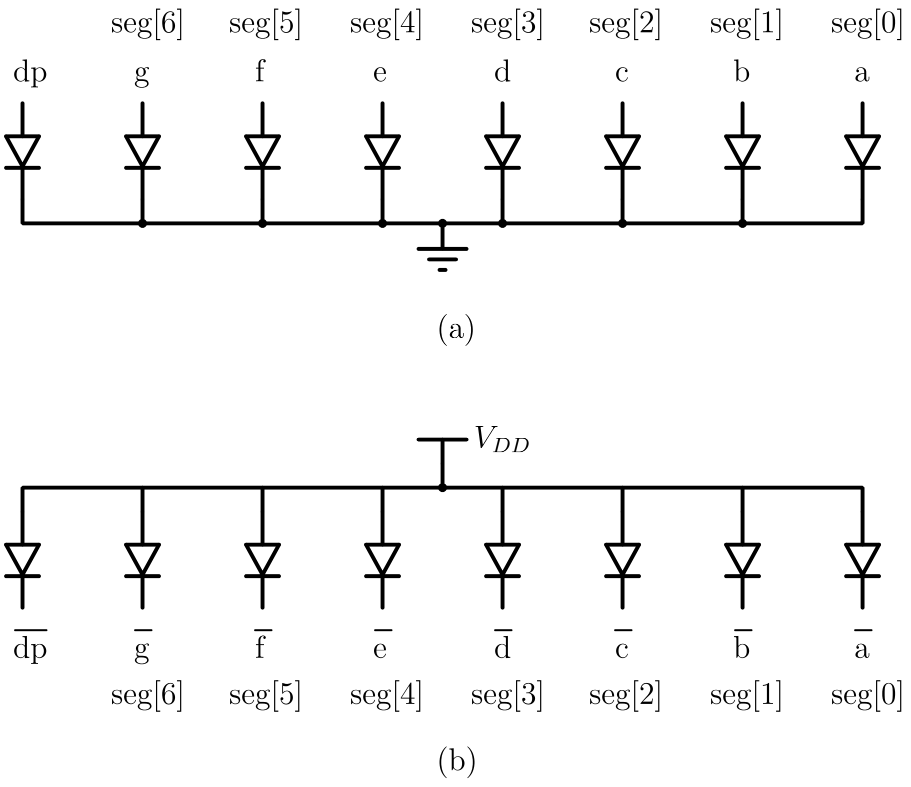

Figure 2: 7-Segment Display Configuration: a) Common Cathode. b) Common Anode.

The anodes of the seven LEDs forming each digit are tied together into one "Common Anode" circuit node, but the LED cathodes remain separate, as shown in Figure 2. The Common Anode signals are available as four "digit enable" input signals to the 4-digit display. The cathodes of similar segments on all four displays are connected into seven circuit nodes labeled CA through CG as shown in Figure 3 (for example, the four "D" cathodes from the four digits are grouped together into a single circuit node called "CD"). These seven cathode signals are available as inputs to the 4-digit display. This signal connection scheme creates a multiplexed display, where the cathode signals are common to all digits but they can only illuminate the segments of the digit whose corresponding anode signal is asserted.

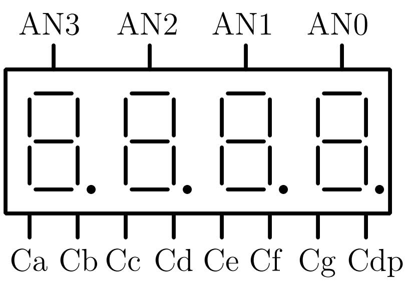

Figure 3: 4-Digit 7-Segment Display Configuration.

To illuminate a segment, the anode should be driven high while the cathode is driven low. However, since the Basys 3 uses transistors to drive enough current into the common anode point, the anode enables are inverted. **Therefore, both the AN0..3 and the CA..G/DP signals are driven low when active.**

Procedure:

In this experiment, you are going to control Basys Board's 7-segments by using buttons with a specific way. The specifications are as follows:

- If btnL is pressed, number "1" will appear on first digit of 7 segment display.
- If btnU is pressed, number "2" will appear on second digit of 7 segment display.
- If btnR is pressed, number "3" will appear on third digit of 7 segment display.
- If btnD is pressed, number "4" will appear on fourth digit of 7 segment display.
- If more than one **button** is pressed, or no **button** is pressed; nothing will be display on 7 segment display.

Implement your module on Basys-3 Board.

7-Segment Applications

EE 241 - Session 7, Spring 2019

Preliminary Work:

Write a testbench that simulates the experiment which is explained below. Show all valid switch combinations with corresponding outputs. Also, show 3 invalid switch combinations with its corresponding outputs.

Procedure:

In this experiment, you are going to control Basys Board's 7-segments by using switches with a specific way. The specifications are as follows:

- Each switch on Basys board has its own ID number namely; sw0=0, sw1=1, ..., sw10=A, sw11=b, ..., sw15=F.
- When a switch is set to logic-1, its ID number will appear on 7-segments.
- If more than one switch is set to logic-1 or no switches are set, no number will appear on 7-segments.
- ID numbers between 0 and 3 will be shown in segment 1.
- ID numbers between 4 and 7 will be shown in segment 2.
- ID numbers between 8 and B will be shown in segment 3.
- ID numbers between C and F will be shown in segment 4.

7-Segment Applications

EE 241 - Session 8, Spring 2019

Procedure:

In this experiment, you are going to control Basys Board's 7-segments by using switches and buttons with a specific way. The specifications are as follows: (Empty lines will be given in the experiment to prevent early attempts.)

- If .... is pressed, the binary number entered by .... will be displayed on .... of the 7 segment display in decimal form.
- If .... is pressed, the binary number entered by .... will be displayed on .... of the 7 segment display in decimal form.
- If .... is pressed, the binary number entered by .... will be displayed on .... of the 7 segment display in decimal form.
- If .... is pressed, the binary number entered by .... will be displayed on .... (rightmost) of the 7 segment display in decimal form.
- If more than one button is pressed, or no button pressed, nothing will be shown on any of displays.

State-Machine Application (Demo)

EE 241 - Session 9, Spring 2019

Procedure:

In this demo, we are going to construct the state diagram of the button application whose details are described below. Then, we are going to implement a state machine which describes the press and release events of a button. Specifications are as follows:

- Use LED[0] as the output.
- Initially, LED[0] is logic-0.
- Each time BtnC is pressed, LED[0] toggles its logic value.
- While releasing BtnC, LED[0] must not change its logic value.

Write verilog description of the machine using behavioral modelling and implement on Basys3 board.

Finite-State-Machine

EE 241 - Session 10, Spring 2019

Procedure:

Implement the state machine given in Figure 1 on Basys-3 board. State numbers will appear on rightmost 7 segment display and outputs will appear on LEDs. Inputs are implemented with btnL and btnR.

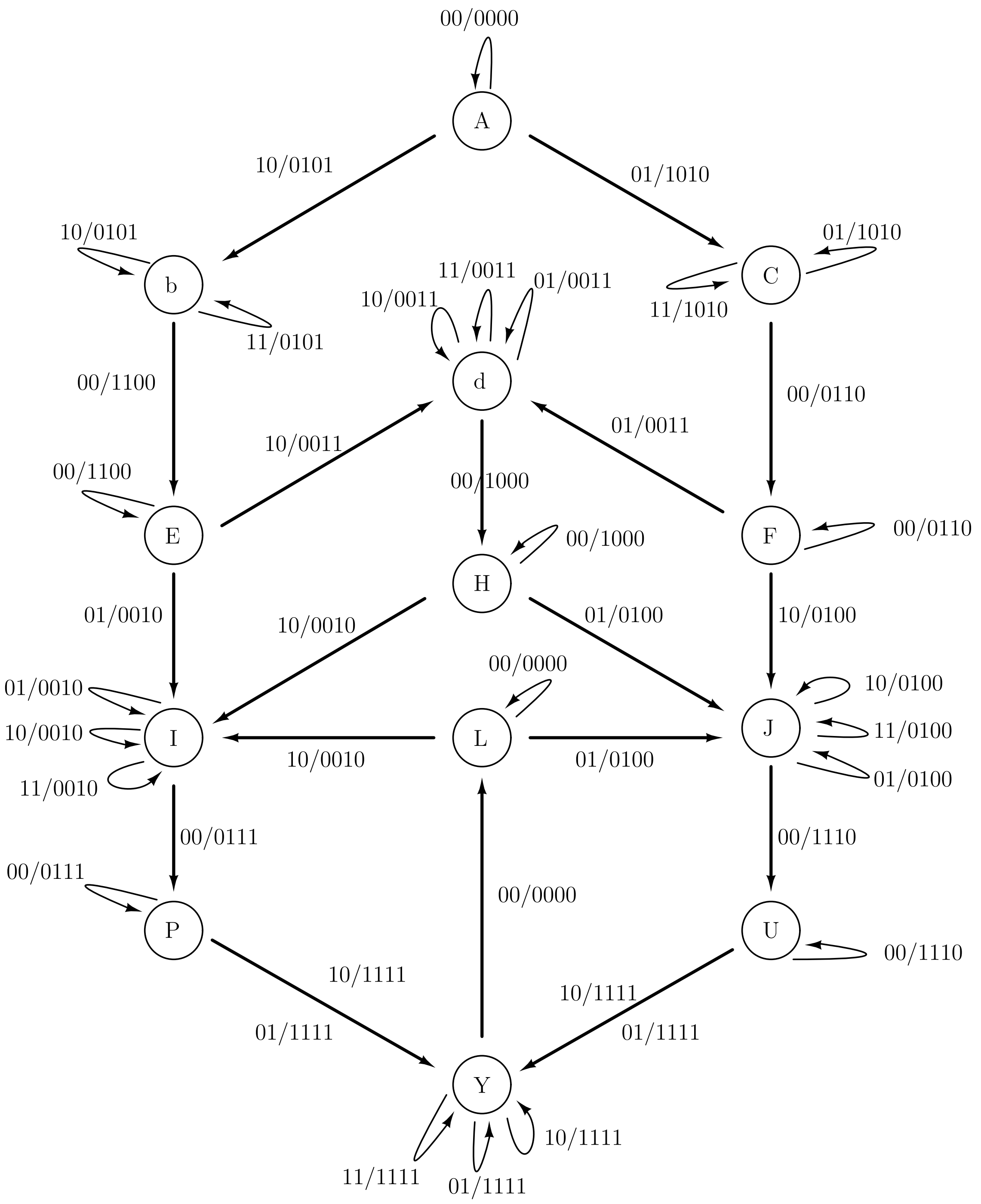

Figure 1: State Diagram

Scanning Display Controller

EE 241 - Session 11, Spring 2019

Procedure:

In this experiment, you are going to upgrade the system in session 8. This time, all digits will show the numbers simultaneously. This system drives the anode signals and corresponding cathode patterns of each digit in a repeating, continuous succession at an update rate that is faster than the human eye can detect. Each digit is illuminated just one-fourth of the time, but because the eye cannot perceive the darkening of a digit before it is illuminated again, the digit appears continuously illuminated. If the update, or "refresh", rate is slowed to around 45Hz, a flicker can be noticed in the display. The specifications are as follows:

- The maximum clock frequency for the scanning display controller is 1 kHz. Therefore, write a verilog module which generates a 1 kHz clock. (Remember the master clock frequency is 100 MHz).
- Write the verilog description of the state machine which scans the anode signals of the 7 segment display.
- The binary number entered by sw[15:12] will be displayed on digit-1 of the 7 segment display.
- The binary number entered by sw[11:8] will be displayed on digit-2 of the 7 segment display.
- The binary number entered by sw[7:4] will be displayed on digit-3 of the 7 segment display.
- The binary number entered by sw[3:0] will be displayed on digit-4 of the 7 segment display.