Laurice Sattouf

Choi Tim Anthony Young

Dimitri Garcia

ECE4304- Lab 3

2/24/2021

Professor M. Aly

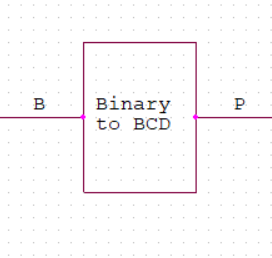
**Binary to BCD converter Displaying on 7-Segment.**

**Purpose:**

1. Design binary to BCD converter unit that takes four inputs and produce BCD values from (0-9).
2. Design a 7SEG decoder.
3. Use two units of the BCD to display the results.
4. Define all possible corner cases using a testbench.

**Procedure:**

1. **Designing Binary to BCD unit:**

To achieve this task, we created the design using combinational circuit and K-maps, where our input is Binary, and the output is BCD from 0 to 9:

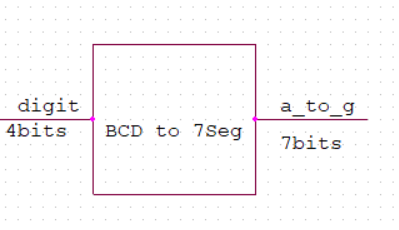
P(3) <= B(3) and not(B(2) or B(1));

P(2) <= B(2) and (not(B(3)) or B(1));

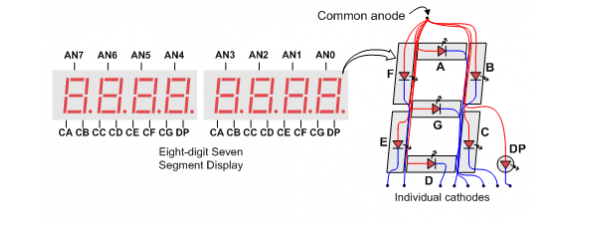
P(1) <= (B(3) and B(2) and not(B(1))) or (not(B(3)) and B(1));

P(0) <= B(0);

1. **BCD to 7-segment Decoder:**

The board we are using to perform this lab, Nexys A7-100T, has 7-segment display with common anode as we could see in Figure (2). Therefore, the segment will be on when logic “0” is applied.

**Figure 2**



|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Digit3** | **Digit2** | **Digit1** | **Digit 0** | **a** | **b** | **c** | **d** | **e** | **f** | **g** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

1. **Using two displays to show results:**

To enable a specific Display from the 8 Display provided on the board, we used top\_an that selects which display will be on. We are asked to turn on Display 0 and 4.

top\_an(7 downto 5) <= (others => '1'); -- off

top\_an(3 downto 1) <= (others => '1'); -- off

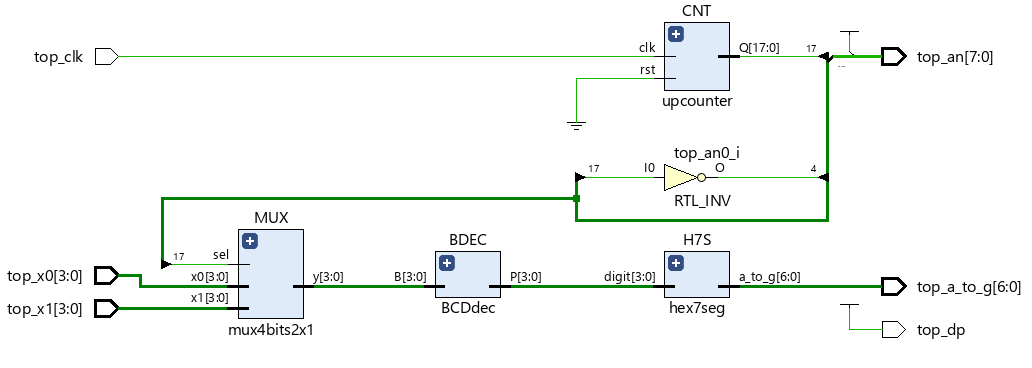
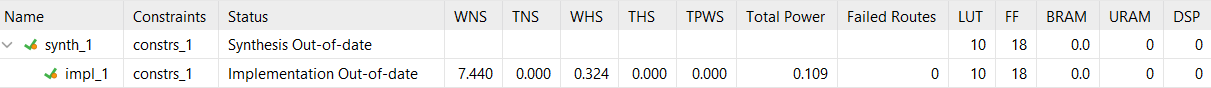
top\_an(4) <= not(top\_sel); --on

top\_an(0) <= top\_sel; --on

1. **Wrapping the circuit with Time Muxing and up-counter:**

the following schematic show the result of our design:

**Figure 3: RTL Schematic using Vivado**

**Power and Resources Used:**

10 lookup tables and 0.109 total power used to implement this design using Vivado. We had three different codes and this was the most efficient with the least amount of power and lookup tables.

**Difficulties:**

This lab was easy to implement since all the syntax that needed are clear and basics.

**Work Contribution:**

* We had a meeting to brainstorm and explain the main idea of the lab, and we created the schematic so that everyone would work individually to achieve the most optimized design.
* We had a zoom meeting to choose the design with less power consuming, then we were able to demo our implemented design and cover all the corner cases.
* Documentation and reports were evenly distributed, and it covered all the steps of our successfully implemented design.