EEL3705L, Digital Logic Design Lab, Spring 2011

**Pre-Lab #3**

For

Lab Assignment #3:

Design a 2-Digit Binary-to-Decimal Display Driver

**Document Description:** This pre-lab report documents my design and though process to arriving at that design to complete the assignment. For this lab, students are required to design a circuit that will take a binary number from 0-99, and display that number on two seven-segment displays.

* **Introduction**

In this lab, students are required to design a circuit that will take a binary input, from 02 to 992, and translate it to a display on the board t hat basically any user will be able to read. The circuit will need to be a modular combinational logic circuit.

* **Requirements**

1. The circuit needs to “read” in the binary number that represents the decimal numbers that range from 0 to 99.
2. The circuit then needs to display the binary input in a form that a literate user will be able to understand.
3. The truth table for this circuit is given below.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Inputs | | | | | | | Outputs | | | | | | | | | | | | | |
| SW6 | SW5 | SW4 | SW3 | SW2 | SW1 | SW0 | HEX1 SEG6 | HEX1 SEG5 | HEX1 SEG4 | HEX1 SEG3 | HEX1 SEG2 | HEX1 SEG1 | HEX1 SEG0 | HEX0 SEG6 | HEX0 SEG5 | HEX0 SEG4 | HEX0 SEG3 | HEX0 SEG2 | HEX0 SEG1 | HEX0 SEG0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 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 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 1 | 0 | 1 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | d | d | d | d | d | d | d | d | d | d | d | d | d | d |

* **Theory and Design**

Firstly, the binary inputs need to be split into the numbers that represent the 100 and 101 place positions in their corresponding decimal number representation. This would be most ideally implemented by a device that would perform a modulus division on the binary input numbers. Although there is not a megafunction in Quartus II for this specific purpose, the megafunction lpm\_div with a binary ten as the denominator and the binary inputs as the numerator will place the numbers corresponding to the 101 place in the quotient bus, and the 100 numbers in the remainder bus. A very simple way to input the binary number would be to use seven switches on the DE2 board to represent the seven digits of the binary number.

Now that each digit of the number is separated, we need to create a circuit that will take each digit from 0-9 in its binary representation and translate it into a form that a user can read. Because the number is split into two digits, we can use the same circuit for each digit. In his lab, we will use the seven-segment display to show the number in its decimal notation. Because there are only four binary digits that can represent the decimal numbers from zero to nine, we can create a Karnaugh Map for each segment of the display, and create a sub-circuit that will translate the binary number to its decimal representation on the seven-segment displays using a combination of the simplest Product-of-Sums method.

K-map for segment 6 of the seven segment display:

i3 i2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i1 i0 | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | d | 0 |
| 01 | 1 | 0 | d | 0 |
| 11 | 0 | 1 | d | d |
| 10 | 0 | 0 | d | d |

K-map for segment 5 of the seven segment display:

i3 i2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i1 i0 | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | d | 1 |
| 01 | 1 | 0 | d | 0 |
| 11 | 1 | 0 | d | d |
| 10 | 1 | 0 | d | d |

K-map for segment 4 of the seven segment display:

i3 i2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i1 i0 | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | d | 0 |
| 01 | 1 | 1 | d | 1 |
| 11 | 1 | 1 | d | d |
| 10 | 0 | 0 | d | d |

K-map for segment 3 of the seven segment display:

i3 i2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i1 i0 | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | d | 0 |
| 01 | 1 | 0 | d | 0 |
| 11 | 0 | 1 | d | d |
| 10 | 0 | 0 | d | d |

K-map for segment 2 of the seven segment display:

i3 i2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i1 i0 | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | d | 0 |
| 01 | 0 | 0 | d | 0 |
| 11 | 0 | 0 | d | d |
| 10 | 1 | 0 | d | d |

K-map for segment 1 of the seven segment display:

i3 i2

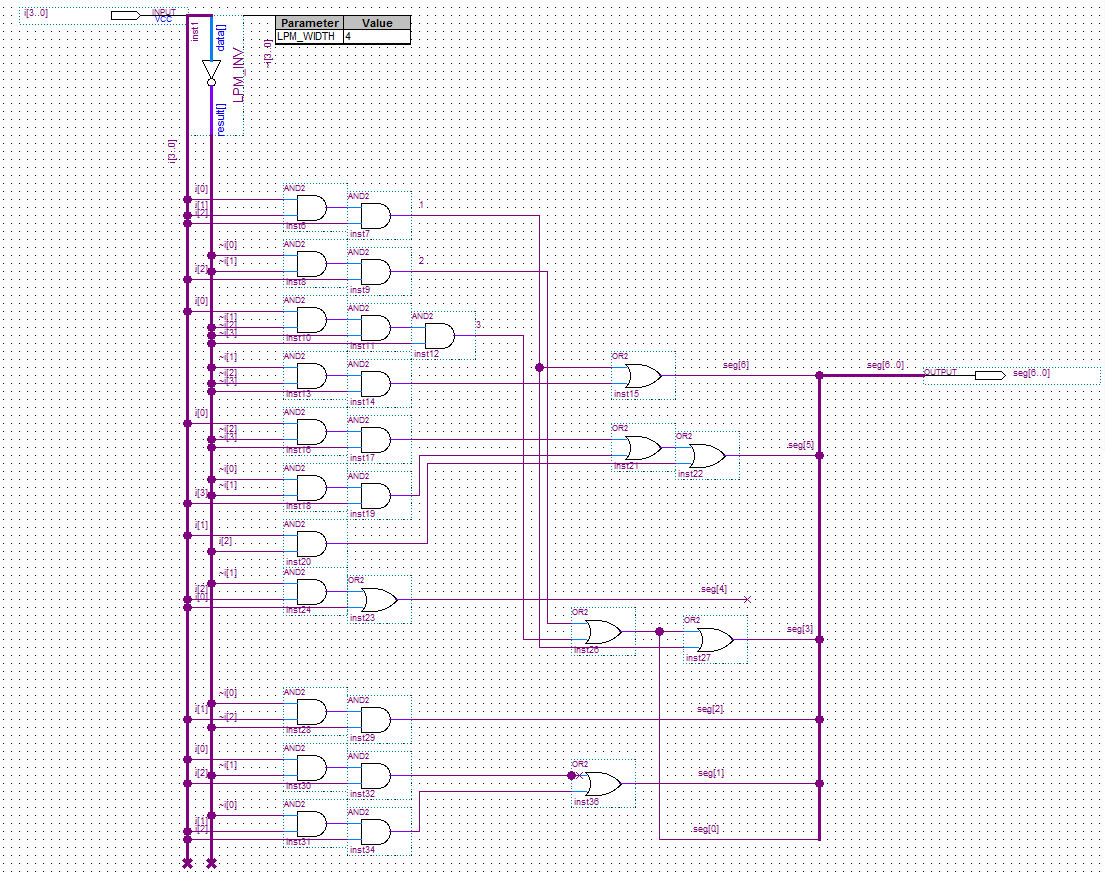
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i1 i0 | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | d | 0 |
| 01 | 0 | 1 | d | 0 |
| 11 | 0 | 0 | d | d |
| 10 | 0 | 1 | d | d |

K-map for segment 0 of the seven segment display:

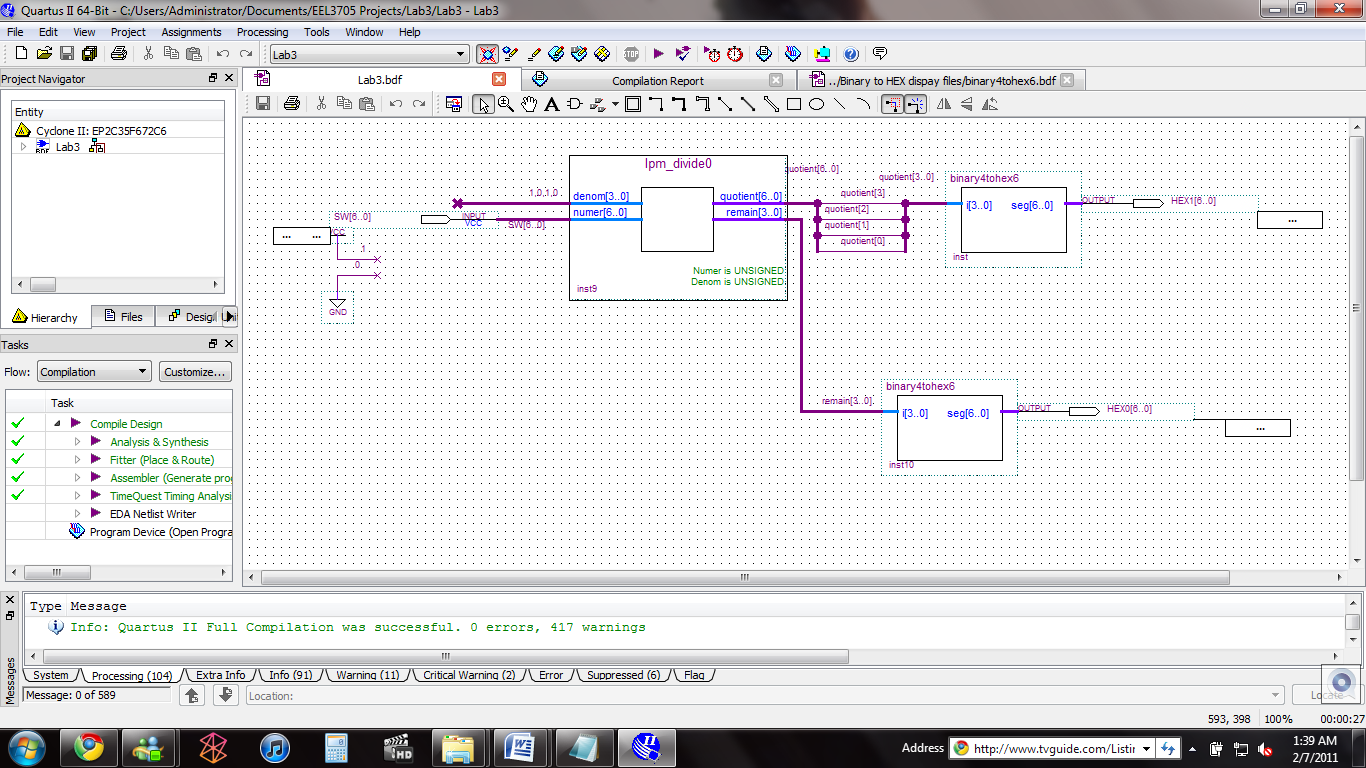
i3 i2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i1 i0 | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | d | 0 |
| 01 | 1 | 0 | d | 0 |
| 11 | 0 | 0 | d | d |
| 10 | 0 | 0 | d | d |

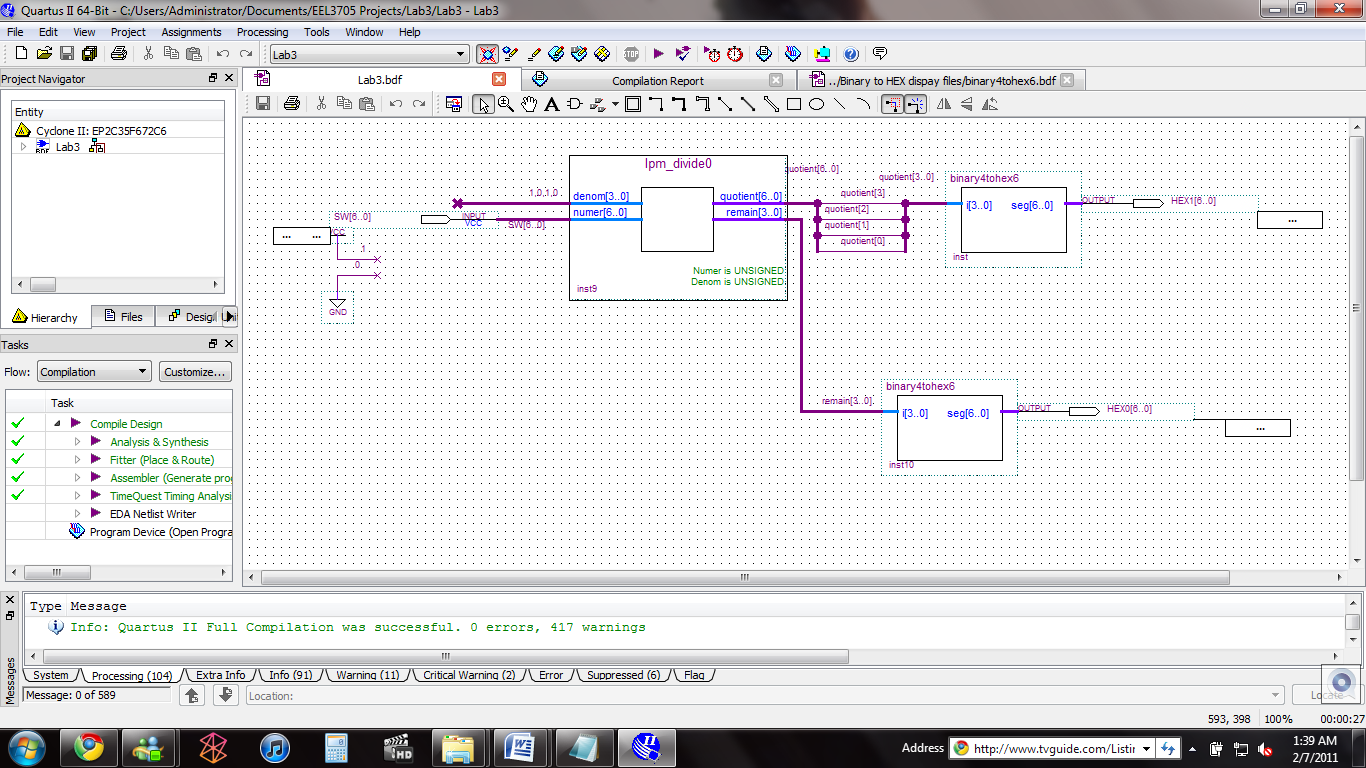
In all of these Karnaugh Maps, i3represented the left-most bit of the input binary number, and descended down to i0, which represented the right-most bit of the binary input number. The simplest product-of-sums formula for each of these segments is:. As we can see, some of these expressions have shared terms that we can recycle for some of the segment outputs. The Quartus representation of the circuit that will implement these formulas is given below.



Because this is just a subsystem that will be implemented multiple times in the circuit, it will be represented by a symbol block in the main circuit schematic, shown below.



The Quartus schematic for the circuit that will implement the entire problem is given in the picture just below.



Because the width of the quotient bus is the same width as the numerator bus, some of the digits in the quotient bus will not be needed, specifically, the first three.

* **Design of Prototype Testing Experiments**

The actual testing of this circuit is quite simple. All that is needed is to flip every possible combination of switches, and see if the corresponding binary numbers are displayed correctly in their decimal form on the displays. Because there are seven input bits, which means there are 128 possible combinations of switch positions, which will take some time.

|  |  |  |
| --- | --- | --- |
|  | Did every possible switch position produce the correct output? (Y/N) | If not, what numbers were displayed incorrectly? |
| Switch positions corresponding to decimal numbers 0 through 127 in binary |  |  |

* **Answers to Questions**



Display Driver

Output Device(s)

Input Device(s)

Output Signals

Binary signals for the input numbers

1. The minimum number of bits in binary needed to represent the numbers 0-99 in decimal is seven. Six bits will only represent numbers up to 63, and seven bits can represent numbers up to 127. Seven bits will allow the range needed for all these numbers, with the minimum amount of numbers left unused.
2. The slider switches can easily be used as the input devices, with one switch representing a single bit in the number.
3. A good way to represent the number to the user would be the seven segment displays. But because each one can only represent number 0-9, two will need to be used in this circuit.
4. For each seven segment display, seven signals will need to be sent to it, one controlling each segment.
5. On the seven segment displays, the number zero through nine will look like 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 respectively.
6. The complete truth table for this circuit is given as number three in the requirements section of this report.
7. There are seven inputs into this circuit, making it extremely too difficult to derive a simplified circuit from a Karnaugh map of each output. Instead, a modular approach needs to be taken.

Signals for each segment

Binary signals for first decimal digit



First seven segment display

Display Driver

Seven signals for each input binary bit

Component to split binary number into two decimal digits

Signals for each segment

Binary signals for second decimal digit

Seven switches representing the seven bits of the input number.

Display Driver

Second seven segment display

1. The signals that go into the component that splits the number into two decimal digits are in binary format, with each bit being controlled by one of seven switches. The output of this component is also a binary number, but should never be larger than a binary twelve. Each “Display Driver” has seven outputs, with each one controlling one segment of the seven segment display it is connected to.
2. Ideally, a modulus division would break up the input number into two components. In this application, it would be the input number mod(%) ten will do what we need.
3. After looking at the megafunction libraries, I saw that LPM\_DIV seems like it will do the modulus division desired, without any modifications. Both of its output numbers are whole numbers, in binary, and can easily be implemented in the rest of the circuit. For application on this lab, all that we have to do is set the input binary number as the numerator, and a binary ten as the denominator. Then, the quotient will be the most significant decimal digit, given in binary, and the remainder will be the least significant digit, also given in binary.
4. Because the maximum number that either the quotient or remainder can output is twelve, we can use just the first four bits for the input to the display drivers. Therefore, we can use a K-map for each segment of the output. Those K-maps are given above in the Theory section.
5. The detailed Quartus schematic and symbol representation of the display drivers are given in the Theory section above.
6. For simulation, we would at least need to have the switches range from 0-99 in binary. Because I don’t think the simulator will be able to display the outputs to the seven segment displays as numbers, we will have to look at the individual waveforms at each switch position to determine the number being output.
7. For this design, we will use switches SW[7..0] and seven segment displays HEX1[6..0] and HEX0[6..0]. We have used these components all semester and their pin assignments can be found in previous lab reports or the user manual for the DE2 board.

* **Conclusion**

This lab, once again, demonstrated a simple problem in which we see the implementation is quite lengthy. Although, one important lesson learned is that using already pre-made modules in your design can really cut down on the time it takes to build and design a circuit. I also noticed that this lab followed the pattern of the last lab, where adding multiple outputs increases the difficultly of the problem, using fourteen output signals here.

I also learned in this lab that representing something in a way that is easy to understand for humans can be difficult, because binary is a hard way understand what a number is.