2/25/2022

Christopher Thomas

CST 231 – Digital System Design I

Lab 3

MUX Display

Christopher Thomas

Table of Contents

[Abstract 3](#_Toc96702310)

[Introduction 4](#_Toc96702311)

[Design 4](#_Toc96702312)

[Physical hardware design 5](#_Toc96702313)

[Synthesized hardware design 6](#_Toc96702315)

[Design Structure 6](#_Toc96702316)

[Modules 6](#_Toc96702317)

[Simulation and Testing 7](#_Toc96702318)

[Problems 7](#_Toc96702319)

[Results and Conclusion 7](#_Toc96702320)

[Appendix – A 8](#_Toc96702321)

[A-1 Pin Mapping Table 8](#_Toc96702322)

[A-2 RLT Diagrams 9](#_Toc96702323)

[A-2.a clk\_divider 9](#_Toc96702324)

[A-2.b counter 9](#_Toc96702326)

[A-2.c binary\_to\_bcd 9](#_Toc96702328)

[A-2.d mux\_display 9](#_Toc96702330)

[A-2e bcd\_to\_decimal 10](#_Toc96702331)

[A-2f control\_generator 10](#_Toc96702332)

[A-2g demux 11](#_Toc96702333)

[A-2h mux\_case 11](#_Toc96702334)

[A-2i multiplexed\_display 12](#_Toc96702335)

[A-3 Module Code 12](#_Toc96702336)

[A-3a clk\_divider 12](#_Toc96702337)

[A-3b counter 13](#_Toc96702338)

[A-3c binary\_to\_bcd 13](#_Toc96702339)

[A-3d mux\_display 14](#_Toc96702340)

[A-3e bcd\_to\_decimal 14](#_Toc96702341)

[A-3f control\_generator 15](#_Toc96702342)

[A-3g demux 15](#_Toc96702343)

[A-3h mux\_case 16](#_Toc96702344)

[A-3i multiplexed\_display 16](#_Toc96702345)

[Table of Figures 16](#_Toc96702346)

[References 17](#_Toc96702347)

# Abstract

A parameterized 12-bit up/down counter with enable and reset will be created. This will require a clock divider like we used in the last lab. You will then send your binary outputs through a binary to BCD converter. Each of these will be fed into a BCD to seven segment decoders. You will then use the CL3641AH muxed seven segment display to display it. It will be wired up using the GPIO 0 header on the DE1-SoC. All necessary mux control, encoding, and decoding will be created.

# Introduction

The purpose of this project is to get familiar with using a multiplexed seven segment display to save on hardware needs. As well as get experience using the GPIO pins of the DE1-SoC. Also, to get more familiar with binary to BCD and up down counters.

# Design

The design consists of two top levels one that handles the clock divider, up/down counter, and binary to BCD. That then feeds into the mux\_display top level which consists of a different clock divider, control generator, demux, BCD to decimal, and the mux\_case.Diagram

Description automatically generated

Diagram, schematic

Description automatically generated

## Physical hardware design

The physical hardware consisted of a circuit for the external multiplexed seven segment display. The CL3641AH muxed seven segment display was used and based off of the information of the data sheet it was calculated that using the 3.3V from our DE1-SoC that 150-ohms was the resistors needed. The CL3641AH is common cathode. Seven 150-ohm resistors were used.

Figure 1: Resistor Formula

## Chart, diagram Description automatically generated with medium confidence

Figure 2: Seven-Segment Circuit

## Synthesized hardware design

The synthesized hardware consists of a clock divider which divides the clock which divides the 50MHz signal to 2kHz for the control\_generator module and the demux and to 100Hz for the counter so it is visible to the human eye. The up/down counter module feeds the binary\_to\_bcd which then feeds the mux\_display top level. Which then controls the multiplexed seven segment display.

### Design Structure

Diagram

Description automatically generated

The clk\_divider module needs to be first so that each other module is running at the appropriate speed. The up/down counter feeds the binary\_bcd module so that off the clock the value BCD value will increase. Once that value is converted to BCD it is sent to mux\_display so that it can be properly displayed.

### Modules

#### Module: clk\_divider

This module utilizes a counter that increments each time it receives a positive edge signal from the onboard clock. A value is then determined that will be reached by the counter before it inverts the output signal so that the device connected to the clk\_divider is only receiving a fraction of the signals and thus dividing the clock signal to a slow rate.

#### Module: counter

The counter is parametrized and receives input from the clk\_divider module and increments count on each positive edge which is the output. The counter module also includes an enable and a reset\_n, the enable determines whether the counter will continue to count if enable is low it will count if not it will pause the count. The reset\_n will start the count over if it becomes a low. The up/down input determines if the count will be increasing or decreasing.

#### Module: binary\_to\_bcd

The binary\_to\_bcd module is parametrized and receives its input from counter. This module utilizes the double dabble algorithm for converting the binary value of the input to an appropriate bcd value. Once the binary value has been converted to the appropriate bcd value it is output.

#### Module: bcd\_to\_decimal

The bcd\_to\_decimal module decodes the bcd value to be used to output appropriate value for the seven-segment displays. This is achieved by using a case statement that takes the bcd value and outputs the appropriate values to the seven-segment display to make the decimal value needed.

#### Module: control\_generator

The control\_generator module rotates through different values to be used by the mux\_case and demux modules.

#### Module: mux\_case

The mux\_case module takes the value from the control\_generator module to know which BCD value to be displayed currently.

#### Module: demux

The demux module takes the value from the control\_generator module to know which of the four seven segment displays to display on.

# Simulation and Testing

The testing was done by using a testbench for each module in the design other than the clk\_divider. The counter was tested incrementing the value with each clock pulse. The output that is expected is for each time the clock goes high the count would increase by 1. The binary\_to\_bcd was tested by having the input incrementing every 5 cycles. The output should be 0-9 and after 9 is reached it should turn back over to 0-9 again. The bcd\_to\_decimal was tested in the same was the bcd module was tested but the corresponding outputs should be a 7-bit value that represents the segments of the display. The output should directly correspond to the case statements in the module. The control\_generator was tested like the counter. The mux\_case was tested by iterating through all the sel values and seeing that the value is changing each time. The demux was tested by also iterating through all the sel values and making sure the output value changes each time.

# Problems

During testing after simulation my tens digit was being displayed in the hundreds digit. This was a hardware issue as I had mixed up the wires when hooking up the circuit.

# Results and Conclusion

The lab was comprised of seven individual modules. The lock divider which was used twice. Once to control the multiplexed display and again to control the up/down counter. Since we require persistence of vision for the multiplexed display it must be ran much faster than the counter which needs to be slow enough to allow the human eye to see the numbers change. The multiplexed display is running at 2kHz while the counter is running at 100Hz. The counter generates a binary value that depending on up/down input either counts up or down. That binary value is then passed to the binary\_to\_bcd module so that it can be separated into the necessary digit value. That value is then sent to the bcd\_to\_decimal module which lights up the appropriate lights on the seven-segment display. This is then fed into the mux\_display top level. The control\_generator, demux, and mux\_case modules all run off of the same clk\_divider so they are in sync. With those modules in sync each section of the multiplexed display gets lit up with the proper value.

# Appendix – A

## A-1 Pin Mapping Table

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Name | FPGA Pin No. | Description | I/O Standard |
| SW[0] | PIN\_AB12 | Slide Switch [0] | 3.3V |
| SW[1] | PIN\_AC12 | Slide Switch [1] | 3.3V |
| SW[2] | PIN\_AF9 | Slide Switch [2] | 3.3V |
| GPIO [0] | PIN\_AC18 | Seven-Segment LED[0] | 3.3V |
| GPIO [1] | PIN\_Y17 | Seven-Segment LED[1] | 3.3V |
| GPIO [2] | PIN\_AD17 | Seven-Segment LED[2] | 3.3V |
| GPIO [3] | PIN\_Y18 | Seven-Segment LED[3] | 3.3V |
| GPIO [4] | PIN\_AK16 | Seven-Segment LED[4] | 3.3V |
| GPIO [5] | PIN\_AK18 | Seven-Segment LED[5] | 3.3V |
| GPIO [6] | PIN\_AK19 | Seven-Segment LED[6] | 3.3V |
| GPIO [7] | PIN\_AJ19 | Sel[0] | 3.3V |
| GPIO [8] | PIN\_AJ17 | Sel[1] | 3.3V |
| GPIO [9] | PIN\_AJ16 | Sel[2] | 3.3V |
| GPIO [11] | PIN\_AH18 | Sel[3] | 3.3V |

## A-2 RLT Diagrams

### A-2.a clk\_divider

### Diagram, schematic Description automatically generated

### A-2.b counter

### Diagram Description automatically generated

### A-2.c binary\_to\_bcd

### Diagram, schematic Description automatically generated

### A-2.d mux\_display

Diagram, schematic

Description automatically generated

### A-2e bcd\_to\_decimal

Diagram

Description automatically generated

### A-2f control\_generator

Diagram

Description automatically generated

### A-2g demux

Diagram

Description automatically generated

### A-2h mux\_case

Diagram, schematic

Description automatically generated

### A-2i multiplexed\_display

Chart, waterfall chart

Description automatically generated

## A-3 Module Code

### A-3a clk\_divider

Graphical user interface, text, application

Description automatically generated

### A-3b counter

Text

Description automatically generated

### A-3c binary\_to\_bcd

A picture containing text

Description automatically generated

### A-3d mux\_display

Text

Description automatically generated

### A-3e bcd\_to\_decimal

Text

Description automatically generated

### A-3f control\_generator

Text

Description automatically generated

### A-3g demux

Text

Description automatically generated

### A-3h mux\_case

Text

Description automatically generated

### A-3i multiplexed\_display

Text

Description automatically generated

# Table of Figures

[Figure 1: Power Equation 4](#_Toc91856676)

# References

[*"Binary-to-BCD Converter: "Double-Dabble Binary-to-BCD Conversion Algorithm"*](https://web.archive.org/web/20120131075956/http:/edda.csie.dyu.edu.tw/course/fpga/Binary2BCD.pdf)

DE1-SoC User Manual