

LAB 3 Report

Team Members: Kameron Bryant and Kevin Ward

Date: 2/17/2020

Introduction

The third lab in microcomputer systems focused designing a circuit and programming the microcontroller to behave as an up-down counter using the given seven segment display. We implemented code to increment a variable that represents the hex value within an array of with 9. When that specific hex value was chosen the then a group of cathodes lit on the seven segment display. Then we used the joystick on the microcontroller to increase or decrease the number. The prelab focused on completing the pin configuration because the correct configurations would be instrumental in completing the lab.

Design

A small portion of the design of this lab relied on building a small internal circuit to correctly display the cathodes on the seven segment display . Another part of the design was based on utilizing masks and bitwise operators to set and reset bit. A larger part of the design relied on creating an algorithm to get the right cathodes to light when displaying a number (0-9) when the up or down button was pressed on the joystick.

Functionality and Correctness

The functionality of the lab primarily relied on the code. The code written is made up of functions that make it possible for the cathodes to produce the correct number. The internaClock function I made Enables the clock, waits until the clock is ready, selects the clock as the systems source, then waits until the clock is used. The Btn_Init function was made to enable PA3 and PA5 on the joystick since those were the buttons used to control the count. The ButtonPressed function let us press PA3 or PA5 on the joystick. The Mode function set and reset the bits for both the GPIO H and E ports. The setODRsfunction controls which Cathodes on the seven segment display are activated . The function countup and countdown functions incremented and decremented the count based on if which button was pressed on the joystick . The outcome of the lab was half correct because the correct cathodes were displayed when the up button was pressed but for some reason nothing would happen when the down button was pressed. The down button was tested and it worked. It just wouldn't work if the up button was pressed and vice versa.

Lab Demo


Pre-Lab Assignment (10 points)

1. Suppose seven segments a-g connected to PE10-PE15 and PH0, respectively. Please demonstrate how to display digital '0' on the seven-segment LEDs.
2. Up and down buttons in the joystick connected to PA3 and PA5. How to initialize PA3 and PA5 (Hint: set PA3 and PA5 as input mode. Don't forget to set both as pull-down)

Lab Demo

You should be able to demonstrate your up-down counter to TAs.

1. The counter should display decimal numbers between 0 and 9, and should "roll over" when incrementing past 9 or decrementing past 0.
2. Use the push buttons or Joystick to increment/decrement the number displayed on the seven segment display.
3. Clearly explain your code functions.

TA: 
2/17/2020

Post-Lab Assignment (10 point)

1. What's the difference between software and hardware debouncing? Use examples to explain how to implement both of them with the STM32L4 discovery board.

Pre-lab

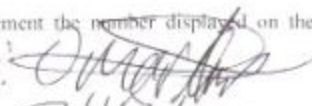
Pre-Lab Assignment (10 points)

1. Suppose seven segments a-g connected to PE10-PE15 and PH0, respectively. Please demonstrate how to display digital '0' on the seven-segment LEDs.
2. Up and down buttons in the joystick connected to PA3 and PA5. How to initialize PA3 and PA5 (Hint: set PA3 and PA5 as input mode. Don't forget to set both as pull-down)

Lab Demo

You should be able to demonstrate your up-down counter to TAs.

1. The counter should display decimal numbers between 0 and 9, and should "roll over" when incrementing past 9 or decrementing past 0.
2. Use the push buttons or Joystick to increment/decrement the number displayed on the seven segment display.
3. Clearly explain your code functions.

TA: 
2/17/2020

Post-Lab Assignment (10 point)

1. What's the difference between software and hardware debouncing? Use examples to explain how to implement both of them with the STM32L4 discovery board.

Post-lab assignment

Software debouncing refers to coding that analyses the value at a manual switch and filters out any miniscule electrical signals coming from the switch during/after it's been pushed, which if left undealt with can be mistakenly recognized as a full "push" signal by the CPU. Afterwards, all that's left is one binary signal, pressed or not pressed, that is dependent on whether or not the button was completely pressed down. Utilizing software is usually the easiest way to perform this function and is extremely common.

Hardware debouncing, on the other hand, is often included in the circuitry of the component and is implemented by hand far less frequently. This involves connecting a capacitor in parallel with the manual switch, which causes the output current to stay 0 A except for when the switch is completely pressed and the capacitor discharges. Implementing this would be relatively straightforward, and would only involve the use of a couple wires and a breadboard to connect a capacitor of proper value in parallel with the manual switch being used.

Conclusions

Half of the objective of this lab was to design a circuit that displayed the proper cathodes to produce the correct number when the up or down button was pressed. The other half of the objective for this lab relied on creating an algorithm to get the right cathodes to light when displaying a number (0-9) when the up or down button was pressed on the joystick . The biggest problem we encountered was we had an issue programming the joystick in a way that would stop the display from changing if the button is held down. We also have what appears to be an inactive button which inhibits us from using the designated pins for the joystick from working in the way we desire, we found this out by testing some of the code on a peer's board. We stopped the program from running continuously when held down by creating a while loop that has a flag, and the flag requires the button to be pressed again to either increment or decrement.