Abdullah Aljandali
EE 454 - UTPIDPWMOC
4/30/2019

Statement of the project

Using STM32L432KC, build a PID controller for a 1-D helicopter. Use PWM to control the speed of the propeller, and an LCD display to allow the user to adjust the coefficients.

Possible utility or purpose

The purpose of this project is to get familiarized with LCD displays and PID control.

Identification of specifications

The project satisfies the requirements for an A-level project, as it controls a 1-D helicopter. It also has a method for externally adjusting and displaying the PID coefficients. The target position is both adjustable in software and externally.

Extra Features

I changed the colors on the LCD :)

Identification of design issues and solutions

I had two ideas on how to allow the user to control the PID coefficients. The first idea is with using buttons, the software loops between the four coefficients and the user has the option of increasing, decreasing, or submitting and moving on to the next coefficient. The other idea was to use five buttons, up and down to increase or decrease values, right and left to switch between coefficients, and a submit button to submit all the values. I decided to do the second one. Although it is more complicated in terms of hardware and requires more buttons, it allows the user to change their values if they changed their mind instead of having to reset the microcontroller.

As for setting the target externally, I could either do it by using buttons and the LCD display just like we did for the coefficients, or to allow them to move the arm and capture the angle it is at. I decided to use the LCD because I am already doing that for the coefficients, so the user will only have to learn one method for adjusting values. Though the value the target is set in the code is not very user friendly, as it is an ADC values (0-4096) depending on the

potentiometer. So I had two ideas to make it more user friendly, that is to either use degrees (0-180) or percentage (0-100). Degrees would make more sense for an arm, but since I am using percentages for the coefficients, I found it less confusing to just use percentage for the target as well. This way 90 degree angle would be 50%.

I was not sure if the project was asking me to allow the user to adjust the coefficients and the target at any point while the system is running, or only once before the system starts. I tried allowing the user to control the coefficients and target during run, but that wasn't very helpful and resulted in a mess as the starting overshoot is not accounted for. Thus, I decided to only allow the user to adjust the values at the start but allow them to see, and not adjust, these values when the propeller is on.

Another reason why I did not want to allow adjusting the values while the propeller is on is because that would require communication with the LCD. The issue with that is that I was using UART and communicating with the LCD was slowing down the program and I was getting a slow response with the propeller.

To power the motor, I used an H-Bridge. The input of the H-bride was from the voltage source and the output goes to the motor, with the PWM signal coming from the microcontroller to the enable pin of the H-Bridge. This way, I can control the motor with the PWM coming from the microcontroller but with bigger voltage and amps than the microcontroller could handle.

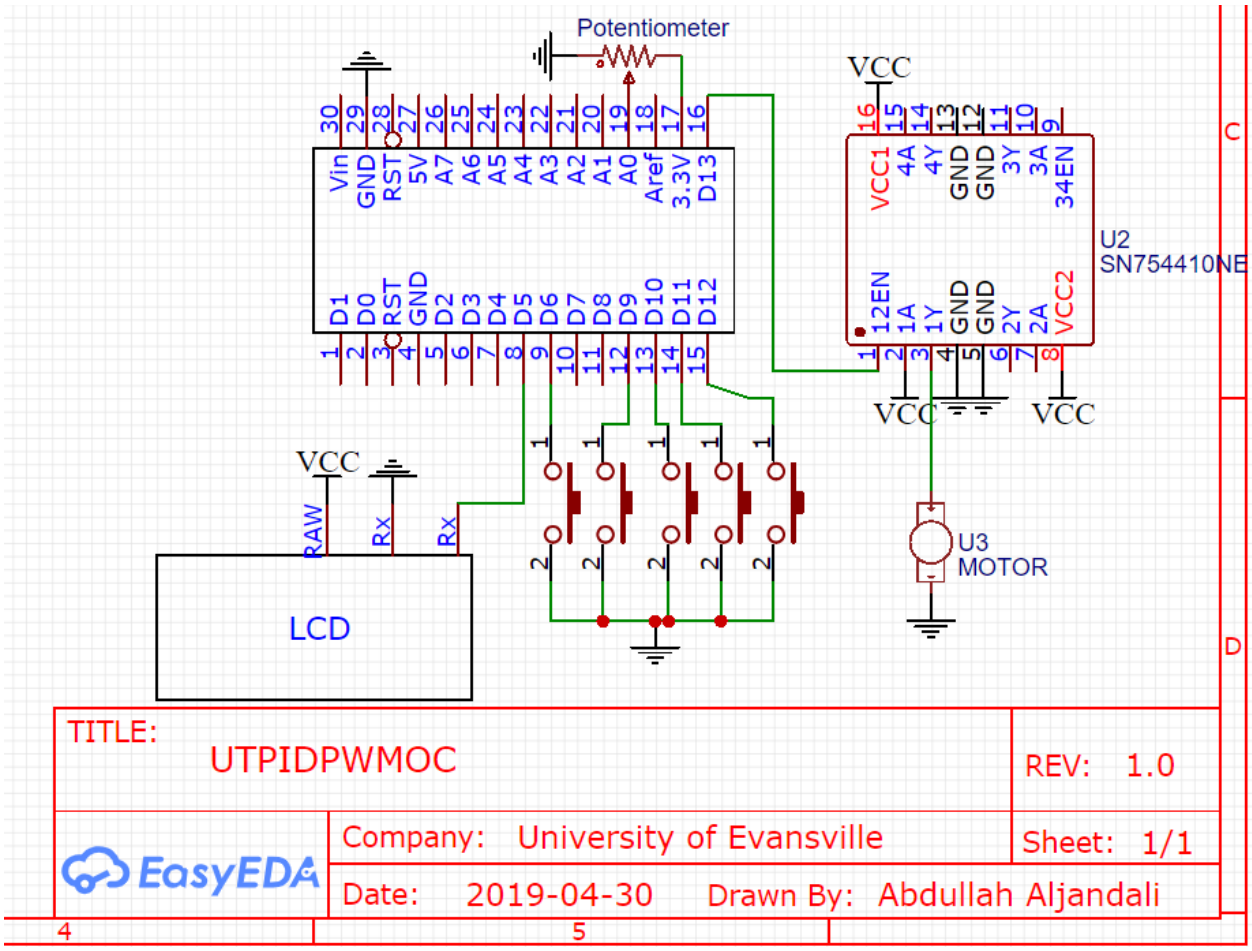
The way I found my coefficients is by setting my target low, ki to 0, kd to 0 and changing kp until it oscillated around my target with a reasonable speed. Then I slowly increased my kd value so that it does not oscillate anymore.

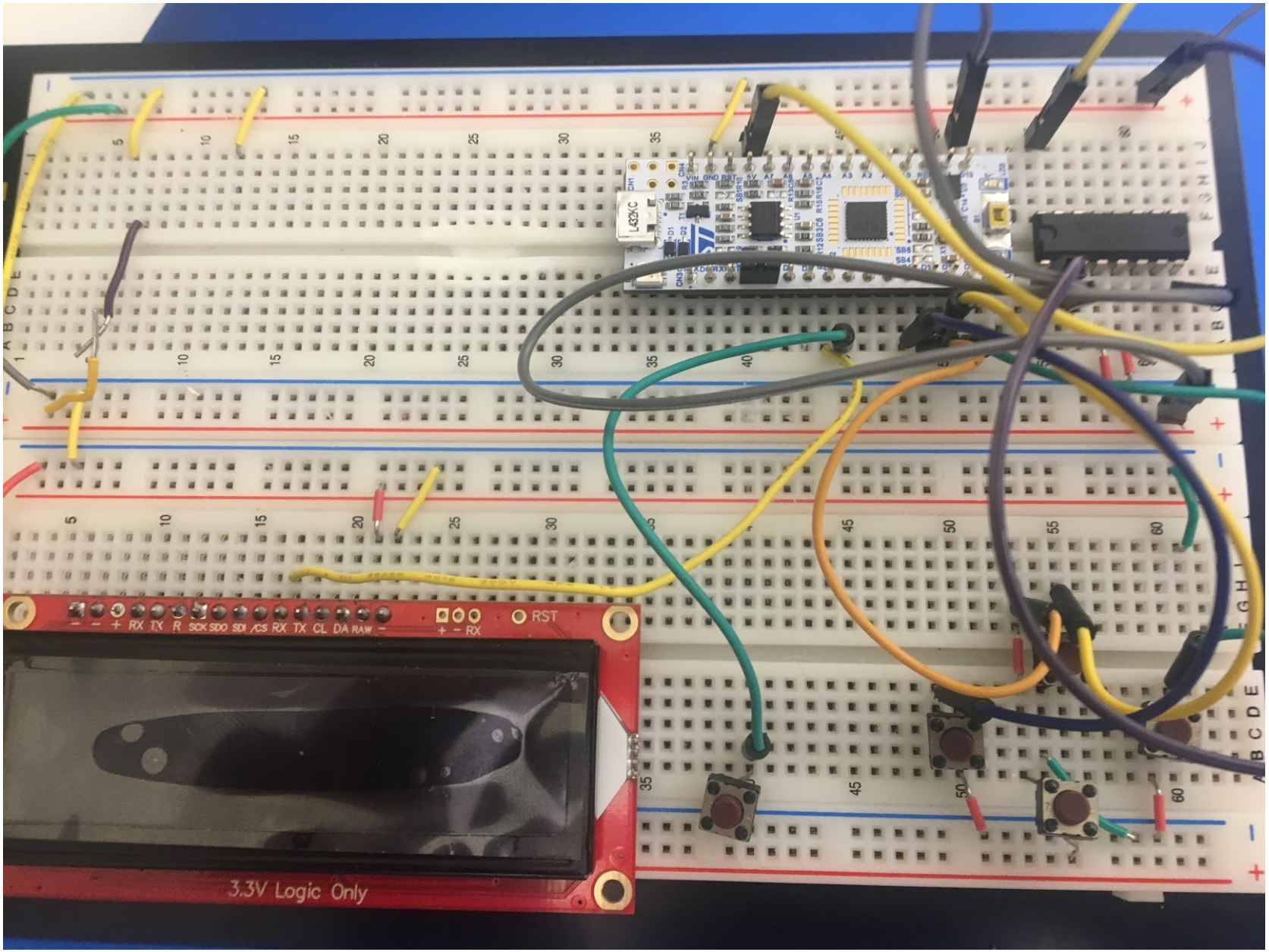
Hardware Details

<u>Components used:</u>

- STM32L432KC Microcontroller - x1
- Buttons - x5
- SN7544NE -x1
- Serial LCD -x1
- Motor -x1

Circuit Schematic




RST
3.3V Logic Only