EE 454
Dr. Mitchell

Project 3— Due Thursday 4/25/19

UTPIDPWMOC (Universal Tunable Proportional Integral Derivative Pulse Width Modulated Output Controller)

This project is designed to gain experience with data displays (likely LCD) and PID control.

The output: There are many ways the control system can be outputted. PWM is one of the most common as it only requires one data line and the duty cycle often correlates nicely to a "target" position. The output should be in the form of a PWM signal with a duty cycle in the range of 0% to 100%. An external driver should be used so that voltage of the PWM can be easily changed (using a variable power supply for example). Also, the external driver should be capable of delivering up to 1 A to the load.

The input: Since this is a universal controller the input can vary. However, at minimum you should be able to receive a 0-3V analog signal as your feedback for the control system.

Tunable: PID control algorithms can be tricky to optimize. As such the coefficients for the PID control algorithm should be externally modifiable. This means you can change the coefficients without reprogramming the microcontroller. It may be a good idea to display the coefficients as a percentage of max. For example, each of the coefficients could be displayed in the range from $-100 \leq K \leq 100$.

Display: There are several display options:

1. 16x2 character LCDs will be available for purchase from Jeff Cron for $20. You are welcome to find you own if you prefer. The LCD uses a simplified I2C, SPI, or UART interface that simply requires ASCII characters be sent though one of those serial protocols.
2. 7-segment LED displays can be used as well. There are a few options available. These will require hooking up a mux and 7-segment drivers. The coefficients can be changed to $0 \leq K \leq 9$.
3. Students can create their own display mechanism, but should be approved by the instructor before the final demonstration.

Setting Target: There should be some method to externally trigger the controller to read the current input and set the position as the target of the PID controller. In other words, a button that when pressed sets the current analog input as the target.

PWM period: If necessary the period of the PWM signal can be set up so that it is easily reprogrammable in the code. However, a nice additional feature would be to have an externally selectable set of PWM periods. Not all systems work best with the same period. Setting a period too low can cause the electronics to switch faster than it can handle, but can allow faster sampling and faster response to changes. Longer periods are often more efficient, but too low causes slow response and possibly "rough" performance. It is up to the student to decide a good range for the PWM period.

Demonstration = connected system can quickly achieve and maintain target position when proper PID coefficients are applied. Also, when PID constants are changed the system should go unstable. If system cannot go unstable then it does not need a PID controller.

Pass requirements:

1. No display
2. Create a basic PID controller and demonstrate it can control a physical system like the 1-D helicopter (will be shown in class).

C-level requirements:

1. No display
2. Create a basic PID controller with externally adjustable coefficients and demonstrate it can control a physical system like the 1-D helicopter (will be shown in class).

B-level requirements:

1. Some method of display for the PID coefficients
2. Create a basic PID controller with externally adjustable coefficients and demonstrate it can control a physical system like the 1-D helicopter (will be shown in class).
3. Target position can be set in software.

A-level requirements:

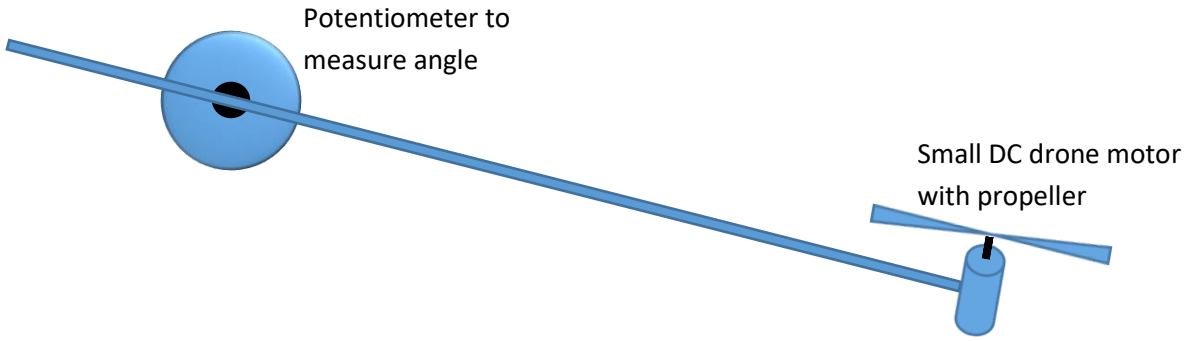
1. USER FRINDLY, WELL DESIGNED method of display for the PID coefficients and possibly PWM period and target position (again using a normalized scale).
2. Create a basic PID controller with externally adjustable coefficients.
3. Method for externally triggering a capture of the target position.
4. Demonstrate it can control a physical system like the 1-D helicopter (will be shown in class).
5. Target position can be set in software.

Optional: externally adjustable PWM period.

Recommendation: final design does not use a bread board.

Example of a connected system: 1-D helicopter (several platforms are available, but you may build your own system).

Confine a small DC drone motor to 1 dimension by attaching it to a pivoting arm as seen below. The microprocessor should make the pivot arm "hover" in the horizontal position.



The minimum for completing this task is as follows:

1. Read the position (angle) using a potentiometer connected to the ADC of the microcontroller.
2. Check for "error" in the position.
3. Adjust a PWM signal that controls the DC motor speed and therefore its height.

Evaluation will be based on at LEAST the following criteria:

1. Can the "helicopter" hover in the horizontal position?
2. Does the helicopter hover with minimal oscillation about the target?

How fast does the helicopter recover from perturbations? In other words, if I push down on the arm how fast can it recover to the hover position.

Grading and Verification:

1. This is an individual project. NO TEAMS.
2. Project verification (by Dr. Mitchell) is due by 2pm on the due date.
3. Students MUST demonstrate a working knowledge of the code and hardware for verification.
4. A penalty of 10% grade deduction per "school day" (MTWTF) will be assessed on late projects (2pm is the cutoff time for each day).
5. Projects verified after 5 school days will be assessed a 50% grade deduction.
6. Final project grades are calculated as:
   60% technical (based on verification of the project as outlined above)
   10% refinement (i.e. error handling, user interface, packaging, etc.)
   15% project description
   15% code organization
7. Technical grade depends on quality of final project at time of verification.

Project description should include:

1. Statement of the project
2. Possible utility or purpose
3. Identification of specifications
4. Identification of design issues and solutions
5. Schematic of components external to the STM32F446 board
6. Code
7. Informational resource used (researching things online is allowed, just tell me about what you find)
8. Be concise, don't write a book

Code organization:

1. Code should be well documented
2. Use descriptive constant, variable, and function names
3. Use functions when appropriate
4. Minimize use of global variables