Mechatronics Lab 1

Name: ______________________________

Due date: 09 / 11 Sep 14

Reading:

- Arduino getting started

Objectives:

- To gain experience programming the Arduino. All coding is to be done individually so that each student gains experience with the tools upon which all subsequent labs are based.
- To verify that software is installed and functional on students laptops
- To gain experience constructing circuits using a solderless breadboard
- To briefly introduce the wire wrap method
- To construct a functional Atmel ATmega328 (Sparkfun Arduino Pro Mini) breadboard circuit. This will be used in subsequent labs as well as homework assignments.

Deliverables:

- Source code is to be placed into a GitHub.com repository for the instructor to review. No credit will be given without this code.
- This is a task based lab assignment similar to shipboard PQS you have completed in the past. Complete the listed Alpha and Bravo tasks and obtain signatures from the instructor. When you are complete submit the completed sheet so that instructor may enter grades into D2L.

Grading: This is a task based lab. Grading is based on the following equation:

$$Grade = \left[\frac{num\ completed\ A\ tasks}{total\ num\ A\ tasks}(75) + \frac{num\ completed\ B\ tasks}{total\ num\ B\ tasks}(25)\right] * is\_code\_posted\_GitHub()$$

Pre lab Discussion Points

This exercise assumes that you have had at least some exposure to the solderless breadboarding in the past. Still, there many be aspects that you are unfamiliar with. Consider the following points as you begin to construct the circuit. Ask your instructor for assistance if you are unfamiliar with any of the concepts.

- Layout of the breadboard's interconnects
- Resistor color code
- Orientation of the breadboard; there is an "up" and "down" side
- Pin 1 identification
- Diode direction

Recommended Procedure:

Breadboarding an electronic circuit is an art. Like all skills, it takes time to develop. Ideally, you would construct the circuit several times striving to improve the layout with each iteration. Unfortunately, this activity would consume excessive amounts of time. To facilitate your construction efforts an instructor built board is available for you to copy.

The circuit should be constructed in stages as outlined below. Each stage should be tested before other stages are added. DO NOT attempt to construct the entire circuit before testing the individual sections.

- **Arduino Pro Mini** The Spark Fun Arduino Pro Mini has already been installed on your breadboard. As a first step you should test the device by opening the Arduino "blink" sketch and downloading the program to the Arduino. This is a good time to obtain signatures for Alpha tasks 1.1 to 1.3.
- **Power supply** The 5 VDC power supply is used for all electronics on the breadboard. Here a 7805 linear regulator is used to regulate the voltage taken from the 9 VDC battery. Construct the lower portion of the circuit as shown in the enclosed schematic. Connect the ground and 5 VDC output to the breadboard's horizontal power bus. Don't forget to connect the upper and lower rails. Use a voltmeter to verify that 5 VDC is present and that the regulator is cool to the touch.
- **Demo code** Sample code to use for this lab is available from:

  https://github.com/macee/mechatronics.git

  Work with your instructor to download the code, place in the appropriate directories, and flash it to the Arduino.
- **LCD** Connect the Liquid Quartz Display (LCD) module as shown on the schematic using a wirewrap method. When functioning the LCD should flash "Welcome to Mecha" every time the Arduino is reset. After the initial start up the LCD will display ”V = XXXX, H = XXXX". Where XXXX is a fluctuating number.

- **Buzzer** Install the buzzer. When complete you should hear a series of tones when the Arduino is reset.
- **Joystick** When the joystick is installed the LCD XXXX numbers will be stable and respond to the vertical and horizontal joystick commands. The LED mounted on the Arduino (pin 13) will light when the joystick is pressed.
- **Tri colored LED** The Arduino will sequence the tri-colored LED during the initial startup.
- **Serial control** Work with your instructor to control the tri-colored LED via the Arduino IDE's "serial monitor" .

Alpha Tasks:

- A1.1 ______________________________ Install the Arduino IDE on your laptop.
- A1.2 ______________________________ Install the FTDI USB driver software on your laptop.
- A1.3 ______________________________ Demonstrate the ability to program the Arduino by installing and running the example blink program.
- A1.4 ______________________________ Construct the circuit presented in this lab using a combination of breadboard and wire wrap methods.
- A1.5 ______________________________ Modify the code so that your name is printed to LCD line 1 and your home town appears on line 2. This should appear for 2 seconds upon start-up.
- A1.6 ______________________________ Modify the code so that the LCD displays repeatedly counts from 0 to 9 with 0.5 seconds between counts. No other data should appear on the screen. Use of a FOR loop is recommended.
- A1.7 ______________________________ Modify the code so that the Arduino beeps for 0.25 seconds every time the LCD count resets to zero.
- A1.8 ______________________________ Establish a user account on GitHub.com. Email instructor your user name.
- A1.9 ______________________________ Work with instructor to post this lab's code to your newly established https://github.com/macee account.

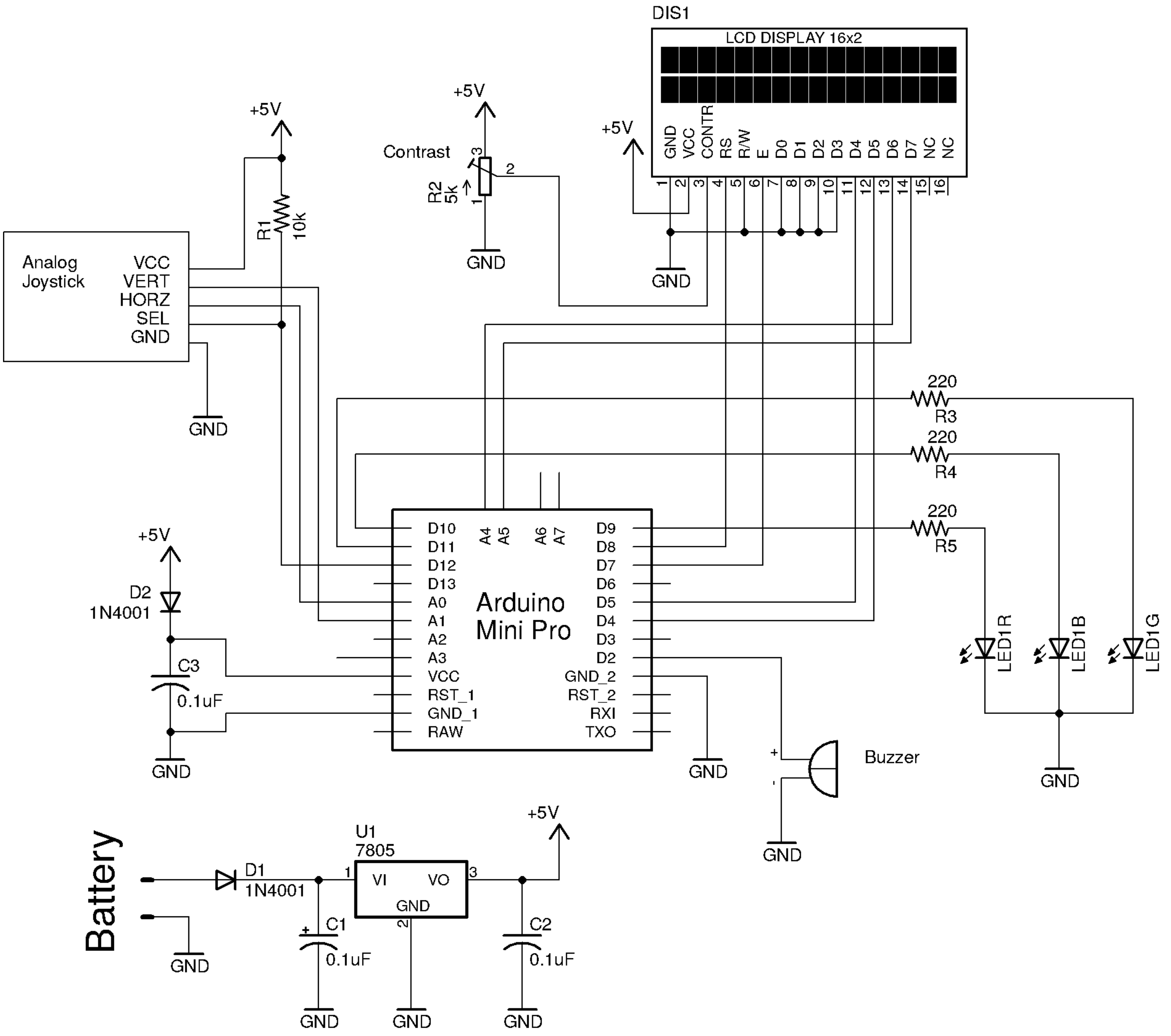
Bravo Tasks:

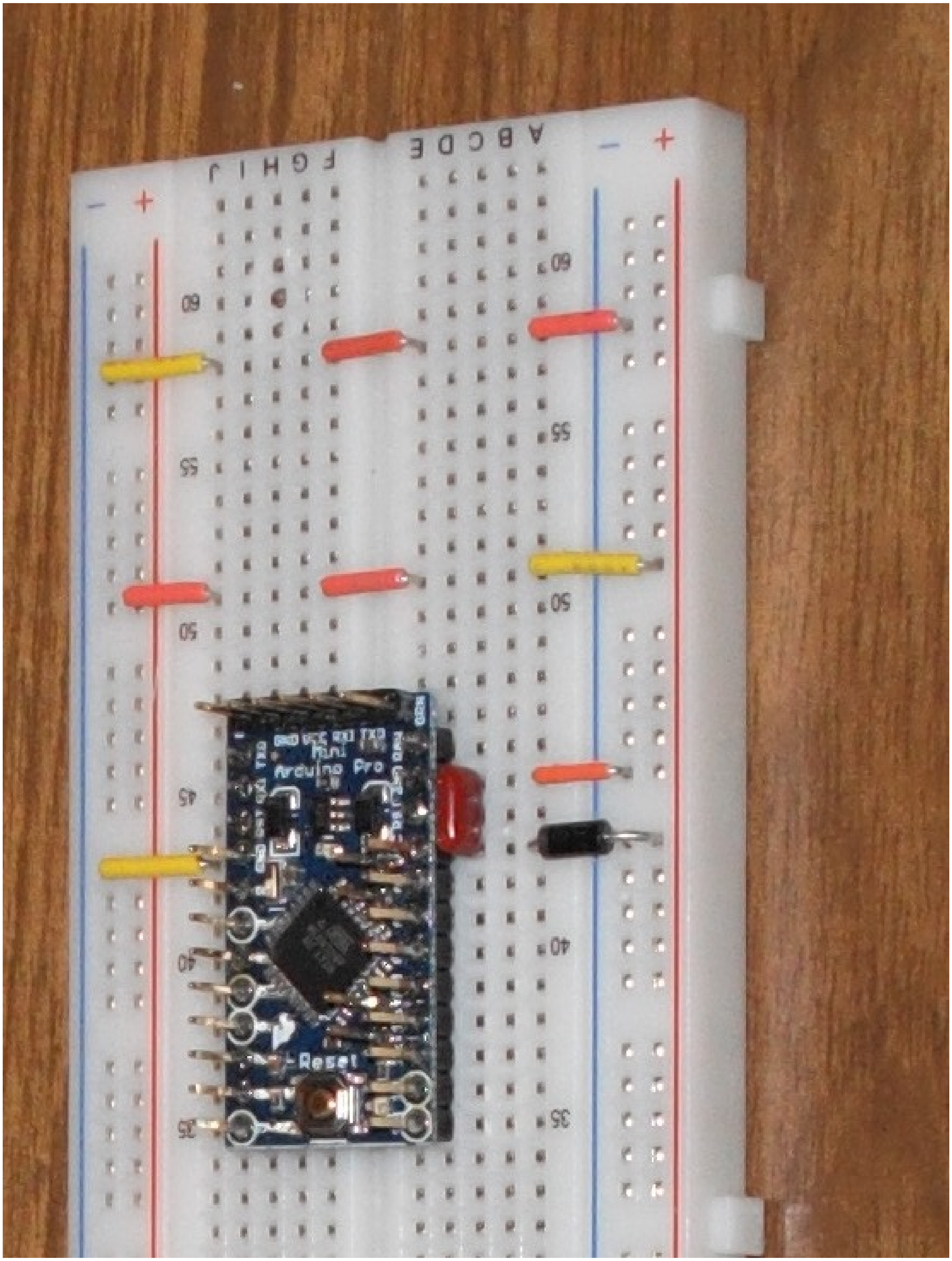
Locate the data sheets for the ATMEL ATmega328 and the Sparkfun Arduino Pro mini to answer B1.1 through B1.4:

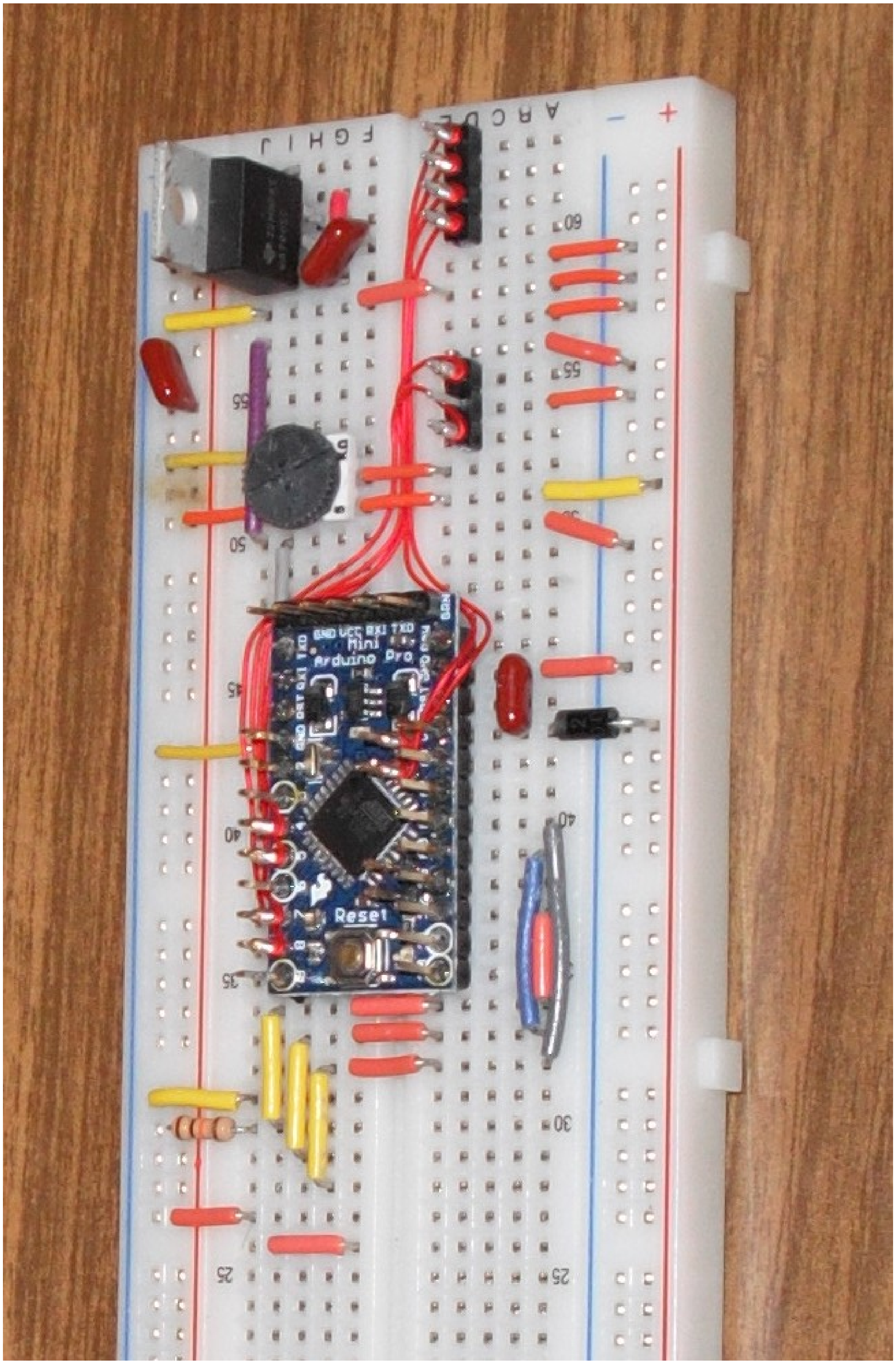
- B1.1 ______________________________ What is the minimum and maximum supply voltage for the ATMEL ATmega328?
- B1.2 ______________________________ What is the maximum current any individual Arduino pin may source and sink? Also, what is the maximum combined current?
- B1.3 ______________________________ What is the size of the ATmega328's RAM?
- B1.4 ______________________________ Like nearly all data sheets, the ATMEL ATmega328's data sheet contains mechanical drawings for the device packages. For signature, show this drawing to your instructor and identify the acronym describing the package for the particular device used on the Sparkfun Arduino pro mini. Hint - it is not a UFBGA, MLF, PDIP, or a VQFN.
- B1.5 ______________________________ State the purpose of Diodes D1 and D2 as shown in this lab's schematic.

Tips and Hints:

- Please resist the urge to remove your Arduino Pro Mini from the breadboard as it is easily damaged. Note that it has been positioned to allow clearance for parts in future labs.


DIS1
LCD DISPLAY 16x2
GND
VCC
CONTR
RS
R/W
E
D0
D1
D2
D3
D4
D5
D6
D7
NC
NC
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
+5V
GND
+5V
Contrast
3
2
1
R2
5k
GND
+5V
R1
10k
Analog
Joystick
VCC
VERT
HORZ
SEL
GND
GND
220
R3
220
R4
220
R5
D10
D11
D12
D13
A0
A1
A2
A3
VCC
RST_1
GND_1
RAW
A4
A5
A6
A7
Arduino
Mini Pro
D9
D8
D7
D6
D5
D4
D3
D2
GND_2
RST_2
RXI
TXO
LED1R
LED1B
LED1G
GND
+5V
D2
1N4001
C3
0.1uF
GND
GND
+
-
Buzzer
GND
Battery
D1
1N4001
U1
7805
1
VI
VO
3
GND
2
+5V
C1
0.1uF
C2
0.1uF
GND
GND
GND
GND


A B C D E
F G H I J
60
55
50
45
40
35
Mini
Arduino Pro
Reset


Arduino Pro
Mini
Reset
GND VCC RX1 TXO

Mechatronics Lab 2

Name: ______________________________

Due date: 16 / 18 Sep 14

Objective:

- To construct a wiring harness for a DC system
- To gain experience using a variety of solderless wire connectors and associated tools
- To introduce fuse protection circuits
- To interface the Arduino with high power solenoids
- To introduce the concept of a step-start with a current limiting resistor

Deliverables:

- Arduino coding is an individual effort - NO collaboration!
- Students may work together to construct the wire harness for their DC motor system.
- Source code is to be placed into a GitHub.com repository for the instructor to review. No credit will be given without this code.
- This is a task based lab assignment similar to shipboard PQS you have completed in the past. Complete the stated Alpha and Bravo tasks and obtain signatures from the instructor. Turn in completed sheet so that instructor may enter grades into D2L.

Grading: This is a task based lab. Grading is based on the following equation:

$$Grade = \left[\frac{num\ completed\ A\ tasks}{total\ num\ A\ tasks}(75) + \frac{num\ completed\ B\ tasks}{total\ num\ B\ tasks}(25)\right] * is\_code\_posted\_GitHub()$$

Pre lab Discussion Points

Consider the following points as you begin to construct the board and ask your instructor for assistance if you are unfamiliar with any of the concepts.

- Fuse
- Color of wire
- Diode direction
- Wire crimp
- Wire stripper

Tips and Hints:

This lab is a continuation of the lab #1. You will use the Arduino, LCD, and buzzer that was constructed in the previous lab.

There are a limited number of hand tools available for circuits construction. Students are encouraged to start this lab by by focusing on different aspects of the construction. For example:

- battery connections
- high power solenoid connections
- low power solenoid connections
- breadboard BJT transistor / MOSFET
- Arduino code

Note that code development can be performed independent of the DC motors i.e., back in the Chase Hall. LEDs 1 and 2 will indicate the state of the motor relays - see schematic attached to this handout.

Alpha Tasks:

- A2.1 ______________________________ Construct and debug a wire harness as shown in the schematic attached to this handout.
- A2.2 (Neatness counts as 4 tasks)

  Neat short wire runs __________________.
  Wires bundled together and bound using tie wraps __________________.
  Terminal lugs used on all wires __________________.
  No loose connections __________________.

- A2.3 ______________________________
  Measure the resistance of K1's coil __________________.
  Calculate the current flow associated with a 24 VDC source __________________.
  Measure the actual current __________________.

- A2.4 ______________________________ Demonstrate ability to control the high power solenoids using the Arduino.
- A2.5 ______________________________ Code system to perform these tasks at startup.
  - Relays K1 and K2 de-energized
  - Send to USART - "This lab demonstrates step start control over DC motors."
  - Send to LCD:
    Step Start Lab
    your name
- A2.6 ______________________________ Code system to perform these tasks at time equals 3 seconds:
  - Send to USART - "Standby, a motor start sequence has been initiated"
  - Send to LCD:
    CAUTION!
    Motor Starting
  - Beep three times alerting the operator that the motor is going to start
- A2.7 ______________________________ Code system to perform these tasks at time equals 6 seconds:
  - Send to USART - "Motor is accelerating."
  - Send to LCD
    Motor ramping
  - Activate relay K2

- A2.8 ______________________________ Code system to perform these tasks at time equals 9 seconds:
  - Send to USART - “Full power engaged.”
  - Send to LCD
    Motor running
  - Energize relay K1
  - De-energize relay K2
- A2.9 ______________________________ Code system to perform these tasks at time equals 12 seconds:
  - Send to USART “Motor secured and coasting to a stop.”
  - Send to LCD
    Motor secured
  - de-energize relay K1
  - de-energize relay K2

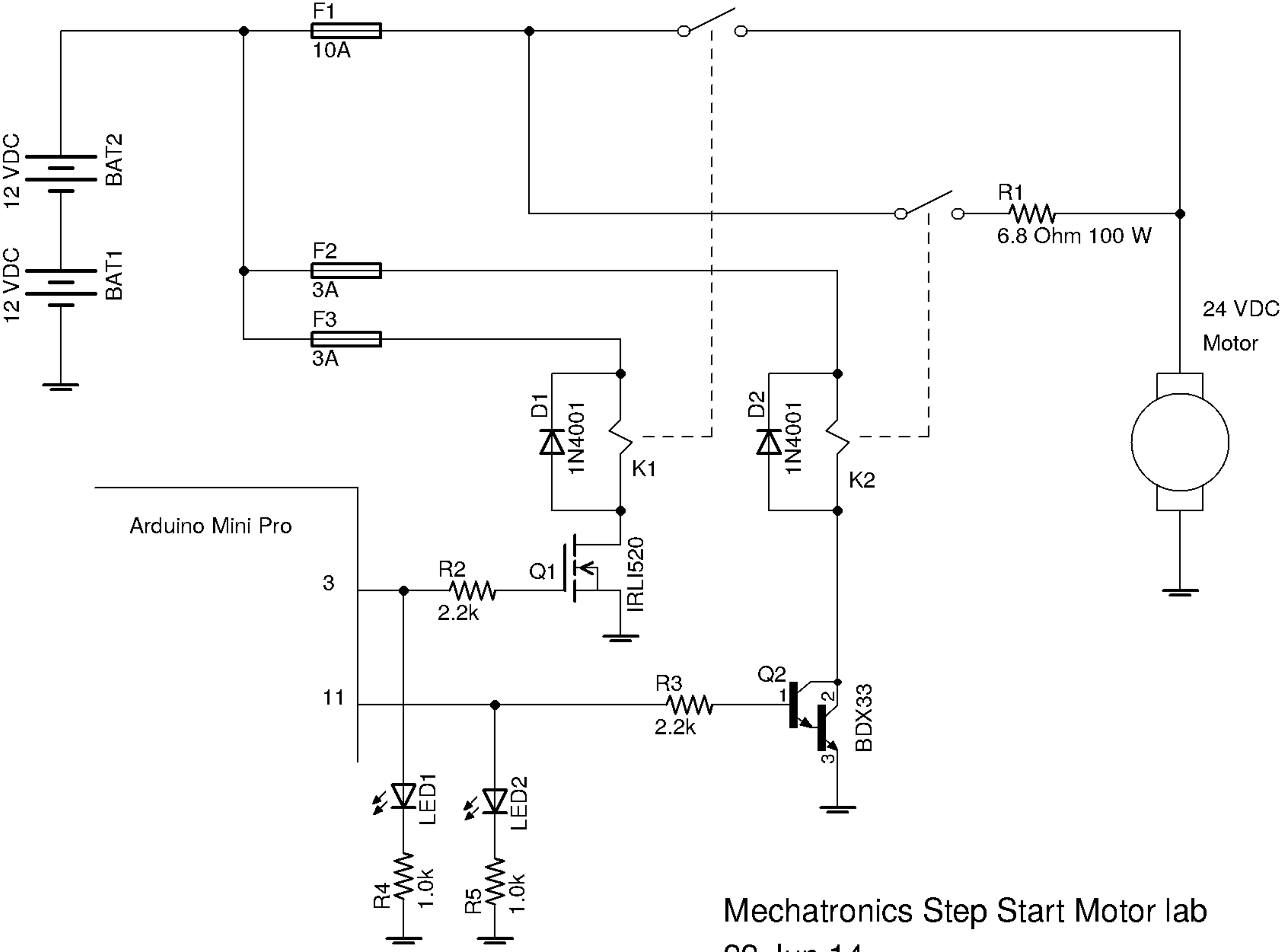
Bravo Tasks:

- B2.1 ______________________________ State the need for establishing software control of relays K1 and K2 in the Arduino’s setup routine.
- B2.2 ______________________________ Describe what would happen if D1 were installed backwards?
- B2.3 ______________________________ Sketch the current flow through D1 for the sequence described in the alpha tasks.
- B2.4 ______________________________

  Measure the voltage drop across R2 while K1 is energized ________________.

  Then Measure the voltage drop across R3 while K2 is energized ________________.

  Based on these reading classify the MOSFET and BJT transistor as either voltage or current controlled.



Mechatronics Step Start Motor lab
22 Jun 14

Note: don't forget to attach the breadboard ground to the motor ground

Mechatronics Lab 3

Name: ____________________________

Due date: 23 - 25 Sep 14

Objectives:

- To explore applications of the Arduino Analog to Digital Converter (ADC)
- To explore use of a low pass filter for signal conditioning
- To introduce serial control via the Arduino serial monitor
- To explore a rudimentary feedback system
- To explore an Arduino to MATLAB / Python serial interface

Deliverables:

- Arduino coding is an individual effort - NO collaboration!
- Source code is to be placed into a GitHub.com repository for the instructor to review. No credit will be given without this code.
- This is a task based lab assignment similar to shipboard PQS you have completed in the past. Complete the stated Alpha and Bravo tasks and obtain signatures from the instructor. Turn in completed sheet so that instructor may enter grades into D2L.

Grading: This is a task based lab. Grading is based on the following equation:

$$Grade = \left[\frac{num\ completed\ A\ tasks}{total\ num\ A\ tasks}(75) + \frac{num\ completed\ B\ tasks}{total\ num\ B\ tasks}(25)\right] * is\_code\_posted\_GitHub()$$

Tips and Hints:

- This lab is a continuation of lab #2. You will use the Arduino, LCD, buzzer, and scooter motors as assembled in the previous labs.
- The second scooter motor functions as a tachometer.
- Code development can be performed independent of the DC motors i.e., back in the Chase Hall. A variable resistor such as the joystick may be substituted for the tachometer. LEDs 1 and 2 will indicate the state of the motor relays - see schematics from lab #2 as well as the schematic attached to this handout.

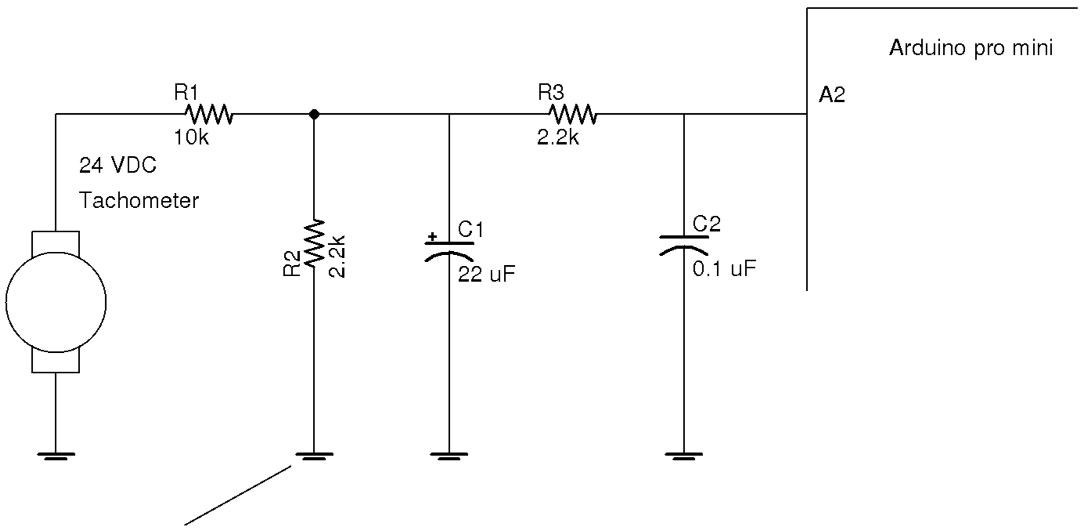
Alpha Tasks:

- A3.1 ____________________ Construct the tachometer circuit as shown on the last page of this document.
- A3.2 ____________________ Demonstrate ability to display the raw tachometer values on the Arduino LCD.
- A3.3 ____________________ Demonstrate ability to display the raw tachometer values on the Arduino serial monitor.
- A3.4 ____________________ Code system to perform these tasks at startup.
  - Send to USART: "This lab demonstrates serial control of the Arduino. It also changes motor operating states based on feedback from a tachometer. Type "start" then strike enter to initiate a motor start sequence"
  - Configure the LCD to scroll the message **Please type start to initiate a motor start**. Hint use snprintf with n equal size LCD and line + i offset...
  - Relays K1 and K2 de-energized
- A3.5 ____________________ Code system to perform these tasks in response to the user command "start":
  - Send to USART: "Standby, a motor start sequence has been initiated"
  - Configure the LCD to scroll the message **CAUTION! motor starting**
  - Beep three times alerting the operator that the motor is going to start
  - After the beeping Activate relay K2
  - Send to USART: "Motor is accelerating."
  - Send to LCD: **Motor ramping**
- A3.6 ____________________ Add a polite handler that prompts the user to enter the correct command when an improper command is entered.
- A3.7 ____________________ Code system to perform these tasks when the motor has reached 75% of idle speed:
  - Send to USART: "Full power at your command. Type "engage" to initiate."
  - Configure the LCD to scroll the message **motor idling, type engage to continue**
- A3.8 ____________________ When operator types "engage"
  - Send to USART: "By your command - full power engaged!"
  - Send to LCD: **Full power!**
  - Energize relay K1
  - De-energize relay K2

- A3.9 ______________________________ Reduce the energy consumed by K1 one second after is has been activated. Determine the lowest PWM duty cycle that reliably “holds” the relay.
    - Send to USART: Relay K1 duty cycle reduced to save power.”
    - Send to LCD: **PWM K1**
- A3.10 ______________________________ Code system to perform these tasks 3 seconds after K1 is energized:
    - Send to USART: “Motor secured and coasting to a stop.”
    - Send to LCD: **Motor secured**
    - De-energize relay K1 and K2
    - After a three second delay start over again at step A3.4.

Bravo Tasks:

- B3.1 ______________________________ Demonstrate ability to use MATLAB or Python to display the raw motor speed. http://www.mathworks.com/help/matlab/ref/serial.html
- B3.2 ______________________________ It has been suggested that power may be saved by using a “pick and hold” operation for the relay. Measure the K1 relay current with and without PWM to prove this assertion. Set the PWM be to the minimum amount that will keep the relay activated.
- B3.3 ______________________________ A PWM signal may be developed using direct software control. Construct a software based PWM without using the analogWrite() function. This can be done using a loop containing digitalWrite() and delay() functions very similar to your first blinky sketch. Consider using the Arduino delayMicroseconds() function for higher frequency operation.
- B3.4 ______________________________ Suppose a laser safety screen is to be added to the mechanism and you need to add code that secures the machine in fast controlled manner. Estimate the time required to modify your code so that the shutdown will occur for all states.
- B3.5 ______________________________ Calculate, simulate, Laplace, or just physically measure the time constant of the tachometer filter.



Caution - this ground is critical to prevent damage to the Arduino

Mechatronics Tachometer Feedback

01 Sep 13

Mechatronics Lab 4

Name: ______________________________

Due date: 30 Sep / 02 Oct 14

Objectives:

- To construct code using a Finite State Machine (FSM)
- To introduce Interrupt Service Routines
- To introduce the ATmega328p special function registers
- To explore blocking vs non-blocking code

Deliverables:

- Arduino coding is an individual effort - NO collaboration!
- Fully functional motor controller operating as specified in this assignment's
- Source code is to be placed into a GitHub.com repository for the instructor to review. No credit will be given without this code.
- This is a task based lab assignment similar to shipboard PQS you have completed in the past. Complete the stated Alpha and Bravo tasks and obtain signatures from the instructor. Turn in completed sheet so that instructor may enter grades into D2L.

Grading: This is a task based lab. Grading is based on the following equation:

$$Grade = \left[\frac{num\ completed\ A\ tasks}{total\ num\ A\ tasks}(75) + \frac{num\ completed\ B\ tasks}{total\ num\ B\ tasks}(25)\right] * is\_code\_posted\_GitHub()$$

Tips and Hints:

- This lab is a continuation of lab #2 and #3. You will use the Arduino, LCD, buzzer, and scooter motors that were constructed in the previous labs.
- Code development can be performed independent of the DC motors i.e., back in the Chase Hall. A variable resistor such as the joystick may be substituted for the tachometer. LEDs 1 and 2 will indicate the state of the motor relays - see schematics from lab #2 as well as the schematic attached to this handout.

Alpha Tasks:

- A4.1 ______________________________ Add code to the ISR so that that Arduino LED on pin 13 blinks once a second. You are not allowed to use blocking code such as delay() from within the ISR.

- A4.2 ______________________________ State the importance of the C keyword "volatile" as related to sending data from the ISR to main and from main to ISR.

- A4.3 to A4.9

  Using the code template supplied by instructor, modify lab #3 to use a FSM construct. You are not allowed to use any delays, serial writes, or LCD writes inside the ISR FSM.

  A4.3 is equivalent A3.4 ______________________________

  A4.4 is equivalent A3.5 ______________________________

  A4.5 is equivalent A3.6 ______________________________

  A4.6 is equivalent A3.7 ______________________________

  A4.7 is equivalent A3.8 ______________________________

  A4.8 is equivalent A3.9 ______________________________

  A4.9 is equivalent A3.10 ______________________________

- A4.10 ______________________________ Add a fault state. This state is entered if the motor does not achieve 75% idle speed within 4 seconds.

Bravo Tasks:

- B4.1 ______________________________ The supplied code template is a departure from Arduino land. It uses a timer based interrupt service routine. The code contains the words “foreground” and “background” in big letters. Describe the significance of these terms.
- B4.2 ______________________________ You have decided to adapt the FSM code used in this lab for use in a slow process control system. Modify the code so that it operates at a rate of 4 times a second.
- B4.3 ______________________________ Analyze the USART library provided by your instructor. Explain how it operates with particular emphasis on the circular buffer.
- B4.4 ______________________________ Retrieve the datasheet for the ATmega328p. Identify the different interrupt sources. Describe three of them to your instructor.
- AB.5 ______________________________ Install Paul Zimmer’s ”Fizzim” Finite State Machine (FSM) software on your laptop. This Java based software is available from http://www.fizzim.com/.
- AB.6 ______________________________ Use the Fizzim software to construct a “bubbles and arrow” diagram of the FSM described in this lab.

Mechatronics Lab 5

Name: ______________________________

Due date: 14 / 16 Oct 14

Objectives:

- To introduce transistor operation
- To introduce data capture and plotting using Arduino and MATLAB
- To explore analog and digital filtering
- To leverage social coding

Deliverables:

- Operational device to measure transistor β and produce the family of operating curves.
- Teamwork to produce beautiful well-documented code.
- Students ready to proceed to an independent Mechatronics project.
- Source code is to be placed into a GitHub.com repository for the instructor to review. No credit will be given without this code.
- This is a task based lab assignment similar to shipboard PQS you have completed in the past. Complete the stated Alpha and Bravo tasks and obtain signatures from the instructor. Turn in completed sheet so that instructor may enter grades into D2L.

Grading: This is a task based lab. Grading is based on the following equation:

$$Grade = \left[\frac{num\ completed\ A\ tasks}{total\ num\ A\ tasks}(75) + \frac{num\ completed\ B\ tasks}{total\ num\ B\ tasks}(25)\right] * is\_code\_posted\_GitHub()$$

Alpha Tasks:

- A5.1 ______________________________ Construct the Arduino circuit shown in the attached schematic.

- A5.2 ______________________________ Demonstrate the ability to set the base and collector power supplies via the serial port using a comma separated commands “DB, XXX” and “DC, XXX” where D is the duty cycle of the respective power supply.

- A5.3 ______________________________ Use the Arduino to calculate I_B and I_C - display the results on LCD line #1 and line #2. Use type float to represent the values. Recall that the AVR snprintf() function does not directly support type float. One option is to use the dtostre() function.

- A5.4 ______________________________ Calculate transistor beta and display the value on line #1 of the LCD in the space after I_B .

- A5.5 ______________________________ Demonstrate the ability to retrieve I_C and I_B via the serial port using the commands “?IB” and “?IC”.

- A5.6 ______________________________ The displayed numbers are likely to be noisy (jump around). To improve this situation construct a crude digital filter that computes the average I_C and I_B based on an average of 100 samples.

- A5.7 (counts as 5 tasks) ______________________________ Each lab section has been assigned to a team - you should have received an invitation from github.com. Work together to code and document a MATLAB interface to the Arduino. The Arduino / MATLAB combination shall draw the family of transistor curves as shown on textbook Figure 10.19. The Arduino shall contain the code as defined in tasks A5.1 through A5.6. Finally, the team shall select one individual’s Arduino code and perfect it as well is it’s documentation.

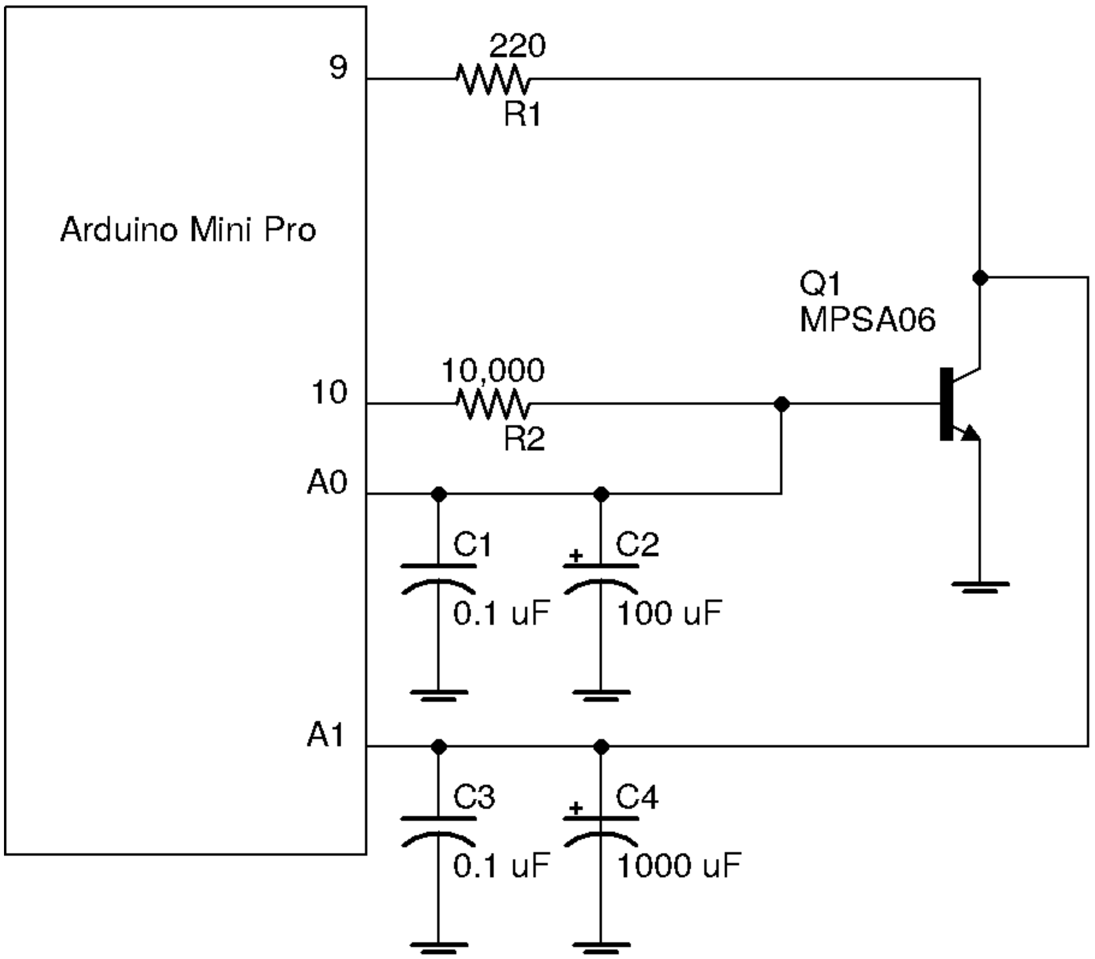
  Grading for this tasks consists of three parts.

  - 33.3% of grade is from peers. Each member will be asked to anonymously rate the performance of each teammate.
  - 33.3% of grade is based on instructor’s observations of the code and documentation submitted via github.
  - 33.3% of grade is from other EE staff members and their assessment of the quality of the code documentation.

  Recommend the initial MATLAB design be done in groups of two. After each team has the code marginally working come together to produce a third improved version.

Bravo Tasks:

- B5.1 ______________________________ State the changes necessary to modify the circuit / code to accommodate a PNP transistor.
- B5.2 ______________________________ Calculate the corner frequency of the low pass filter attached to Arduino PWM pin 10. How does this compare to the Arduino's PWM frequency?
- B5.3 ______________________________ Similar curves may be constructed to develop for a MOSFET. What, if any, changes need to be made to the hardware and software.



Mechatronics Transistor Introduction Lab
30 Sep 14

Mechatronics Lab 6

Name: ______________________________

Due date: 08 Dec 14

Objective:

- to introduce motor control using an H-bridge
- to introduce rotary encoder
- to introduce FPGA
- to introduce the PID (Proportional Integral Derivative) controller

Deliverables from each team:

- Source code is to be placed into a GitHub.com repository for the instructor to review. No credit will be given without this code.
- This is a task based lab assignment similar to shipboard PQS you have completed in the past. Complete the stated Alpha and Bravo tasks and obtain signatures from the instructor. Turn in completed sheet so that instructor may enter grades into D2L.

Grading: This is a task based lab. Grading is based on the following equation:

$$Grade = \left[\frac{num\ completed\ A\ tasks}{total\ num\ A\ tasks}(75) + \frac{num\ completed\ B\ tasks}{total\ num\ B\ tasks}(25)\right] * is\_code\_posted\_GitHub()$$

Alpha Tasks:

- A6.1 ______________________________ Download the instructor provided lab-6 code from: https://github.com/macee/mechatronics.git

  Examine the code and identify:

  - Sampling time: __________________

  - Setpoint(s): __________________

  - Proportional gain: __________________

- A6.2 ______________________________ Using the instructor provided hardware and software present a working proportional controller.
- A6.3 ______________________________ Modify the Arduino code to implement a PID controller with minimal overshoot and rise time.
- A6.4 ______________________________ Plot the system response as well as the individual P, I, and D components using a format similar to that shown at the end of this document. Measure the following attributes:

  - rise time: __________________

  - percent overshot: __________________

  - settling time: __________________

Bravo Tasks:

- B6.1 ______________________________ Integral windup degrades the performance of a PID controller. Code an integral windup preventer that "locks" out the integral if the error is larger than some value of your choosing. Hint: examine your plots from step A6.4 to determine your lockout parameter. Consider using an ABS() function. Adjust parameters for minimal overshoot and minimal risetime.

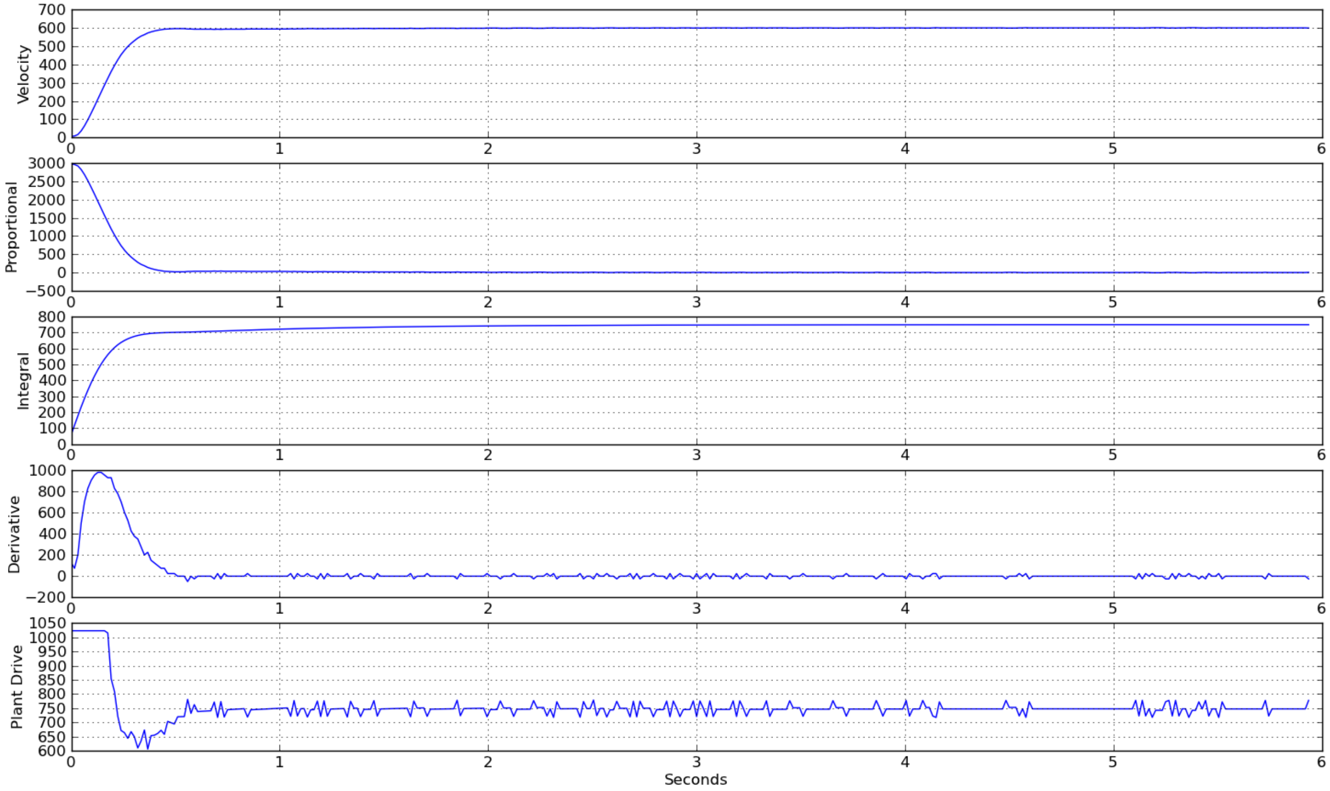
- B6.2 ______________________________ In the plot developed in step A9.5 you may have noticed that the derivative term is noisy. Code a feature to vary Kd based on error. If error is large Kd is large. If error is small Kd is turned off.

- B6.3 ______________________________ Plot the individual P, I, and D components for your improved PID. Measure the following attributes:

  - rise time: __________________

  - percent overshot: __________________

  - settling time: __________________


Velocity
0
100
200
300
400
500
600
700
Proportional
−500
0
500
1000
1500
2000
2500
3000
Integral
0
100
200
300
400
500
600
700
800
Derivative
−200
0
200
400
600
800
1000
Plant Drive
600
650
700
750
800
850
900
950
1000
1050
0
1
2
3
4
5
6
Seconds