University of Pennsylvania
Department of Electrical and Systems Engineering

ESE 206: Electrical Circuits and Systems II – Lab

H-Bridge Motor Control

Objective:

The objectives of this lab are:

1. To construct an H-bridge using 6 enhancement MOSFETS.

2. To use this H-bridge to control a motor using a Q-Drive and an Inverted Q-Drive. You will be able to switch the motor ON and OFF, and control its direction.

3. To use an Arduino Microcontroller to control the Q-Drive and Inverted Q-Drive.

Introduction:

A MOSFET transistor is a three terminal semiconductor device in which current, flowing from the drain-source terminals, is controlled by the voltage on the gate terminal. A MOSFET can be used as a switch, by controlling the voltage provided to the gate. For an enhancement n-MOSFET (NMOS), if the gate voltage is sufficiently high, current flows from the drain to the source. Conversely, for a p-MOSFET (PMOS), with a sufficiently high gate voltage, current flows from source to drain. **Figure 1** shows the terminals of a NMOS (a) and a PMOS (a).

Figure 1: Terminal Names for NMOS (a) and PMOS (b)

We will be using the IRF630 for NMOS and the IRF9630 for the PMOS. Take a look at the datasheets and make sure to know the pin configurations of both.

Motivation for the H-Bridge:

A D.C. Motor requires a voltage difference between its terminals to rotate. The direction in which a motor rotates in determined by which side of the motor is connected to the positive and negative terminals. Swapping the positive and negative terminals will cause the motor to rotate in the

opposite direction. An H-Bridge is used to control the direction of the motor and to also provide enough current for the motor to run.

Operation:

1. To force a motor to switch in two directions, one requires a minimum of 4 switching elements. We will use 4 MOSFET to control the direction of the motor. Consider **Figure 2** shown below.

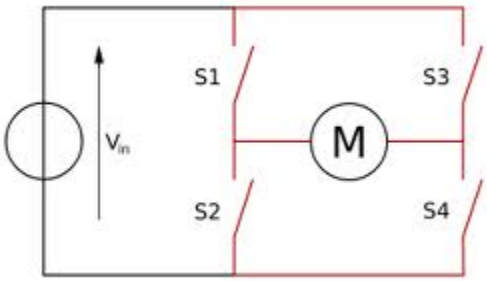

Figure 1: H-Bridge Configuration

Source: http://en.wikipedia.org/wiki/H_bridge

Initially, all switches are open. There is no potential difference across the ends of the motor. It will therefore, not rotate in any direction. If only switches 1 and 4 are ON (Q-Drive), there will be a voltage drop across the motor and it will run. This also occurs when only switches 2 and 3 are ON (Inverted Q-Drive), however the motor runs in the opposite direction. If only S1 and S3 are ON, or if only S2 and S4 are on, there will be no voltage drop across the motor and it will brake.

Note that if S1 and S2 are open at the same time, the circuit will short and you will have very high current and rapid heating of the MOSFETs. This also occurs if S3 and S4 are open at the same time. Be sure to never let this situation occur!

Prelab Assignment:

1. Find some literature on H-Bridges and take some time to learn the basics.
2. Write out a truth table like the one you see below. Put 0 (OFF) or 1 (ON) for inputs, and for output write the response of the motor or the circuit. There are 16 possible situations. Which scenarios result in a dangerous circuit output?

S1	S2	S3	S4	Result
0	0	0	0	…
1	0	0	0	…
0	1	0	0	…
etc.				

2. We will be controlling these four MOSFETS with only two inputs – 2 more MOSFETS. We will be shorting the gates of S1 and S4 and controlling these two MOSFETs with a fifth MOSFET (S5), and also shorting the gates of S2 and S3 and controlling it with the sixth MOSFET (S6). So, for example, if S5 is ON, then S1 and S4 will also be. In this configuration, S5 (Input 1) controls *both* S1 and S4, while S6 (Input) controls S2 and S3.

Prelab Assignment:

3. Create another truth table – this time with only 2 inputs. List the resulting motor or circuit behavior. There are four possible situations. Which scenarios result in a dangerous circuit output?

Procedure:

Part I: Implementation using MOSFETS

We will use 2 PMOS and 4 NMOS to construct our initial H-Bridge. The 2 PMOS (S1 and S3) are used as the "source" for the current and the NMOS (S2 and S4) are used as the "sinks". NMOS S5 and S6 will be connected as shown in order to control the H-Bridge. We will then use a 5 V power supply to supply the inputs (Gates of S5 and S6) in configurations of your choosing. **Figure 3** shows a schematic of the circuit.

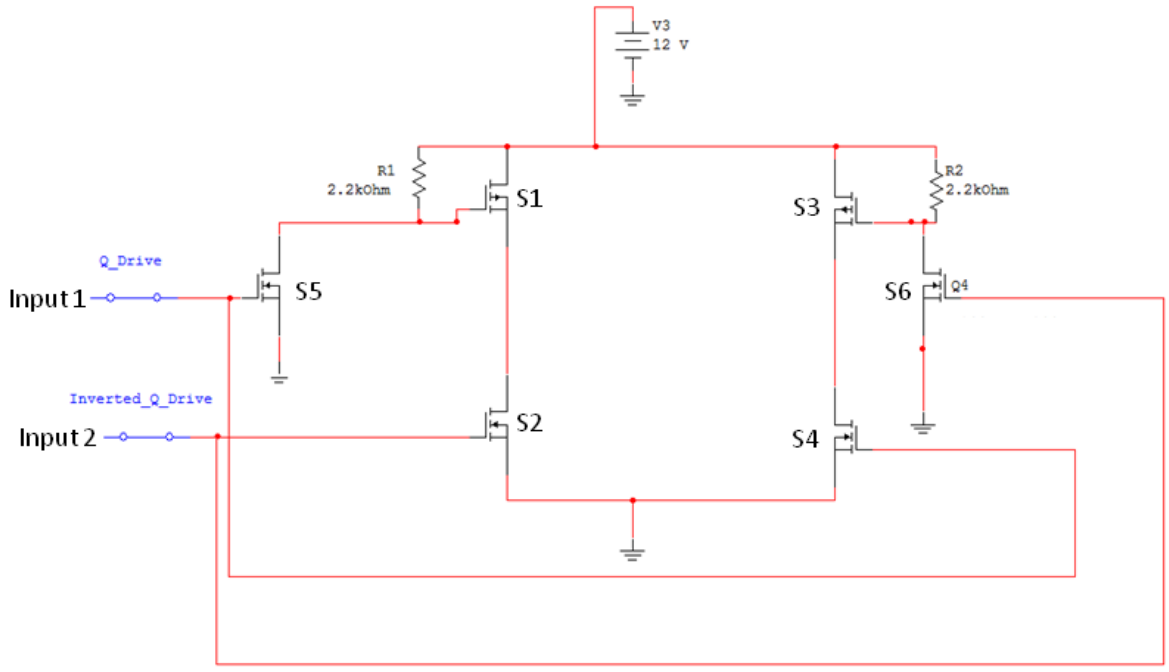

Figure 3: Circuit Schematic of H-Bridge

Build the initial H-Bridge circuit as shown above. Remember that PMOS conduct from source to drain, and NMOS conduct from drain to source (See **Figure 1** above).

The source terminal of both the PMOS connect to the power supply (12 V), and their drains are connected to the motor and to the drains of the NMOS directly below them. The source terminals of the NMOS are grounded. S5 is triggered by Input 1 and S6 is triggered by Input 2.

To test the working of the circuit, you will use the Tri-State switch, as shown in **Figure 4**.

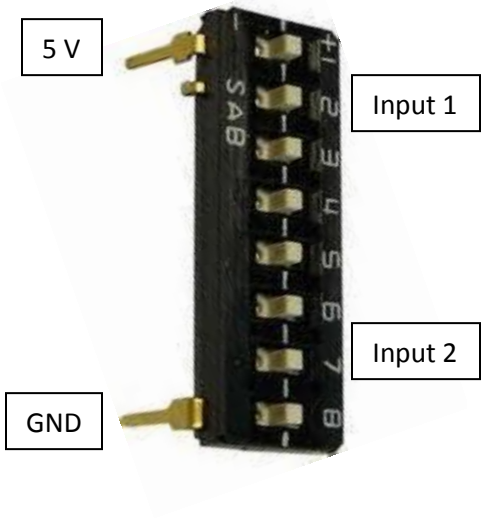

Figure 4: Tri-State Switch in H-Bridge Circuit

Connect a 5 V power supply (NOT V_{CC}!) to the positive terminal of the switch, and GND to the negative terminal. Connect Input 1 and 2 to two lines of the switch, and take turns switching the inputs to 5 V (ON) and GND (OFF). Make sure your motor is connected so that it performs as listed in **Table 2** below.

Input 1	Input 2	Motor Description
ON	OFF	Rotates in clockwise direction.
OFF	ON	Rotates in the counter-clockwise direction.
ON*	ON*	Motor "brakes" rapidly*
OFF	OFF	Motor does not rotate.

Table 2: Truth table for H-Bridge control

***DON'T LEAVE BOTH INPUTS ON FOR MORE THAN 10 SECONDS –THIS SHORTS THE POWER SUPPLY.**
Now test your H-bridge against the truth table shown in **Table 2**. If your circuit satisfies all four cases, show your TA, and proceed to the next part.

Part II: Control with Arduino

As you may know, the Arduino can be used for many different purposes, one of them supplying input voltages. We will be using the Arduino to supply the two H-Bridge inputs. Use the Arduino

Duemilanove provided and connect it via USB to your computer. Open up the Arduino 0022 IDE program on your desktop. Here is a reference if you are unfamiliar with Arduino.

Open up a new sketch and copy the code listed below into the sketch. Read over the code carefully. How does it function? What do the variables 'leftstatus' and 'rightstatus' monitor? Put your name in the 'lcd.print("My name is___")' line of code.

```
#include <LiquidCrystal.h>
int gate1 = 11;
int gate2 = 12;

LiquidCrystal lcd(8,9,4,5,6,7);
int analogPin = A0;
int adc_key_old;
int adc_key_in;
int NUM_KEYS = 5;
int key = -1;
int adc_key_val[5] = {30, 150, 360, 535, 760};
int leftstatus = 0;
int rightstatus = 0;
int timeout = 120;

int freq = 50;
int duty = 100;

void setup() {
  pinMode(gate1, OUTPUT);
  pinMode(gate2, OUTPUT);
  // set up the LCD's number of columns and rows:
  Serial.begin(9600);
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.clear();
  lcd.print("hello, world!");
  lcd.setCursor(0,1);
  lcd.print("My name is ____");
  delay(3000);
  lcd.clear();
  lcd.setCursor(0,0);
  adc_key_old = analogRead(analogPin);
  digitalWrite(gate1, LOW);
  digitalWrite(gate2, LOW);
}

void loop() {

 while(true) {
 constantout(leftstatus, rightstatus, freq);  // <--- Comment Out in Part III
 adc_key_in = analogRead(analogPin);
 adc_key_in = get_key(adc_key_in);
 if (adc_key_in == 3){
   if (leftstatus == 0) {
```

```
    leftstatus = 1;
    Serial.print("Left Status = ");
    Serial.println(leftstatus);
    delay(300);
    break;
   }

  else if (leftstatus == 1) {
    leftstatus = 0;
    Serial.print("Left Status = ");
    Serial.println(leftstatus);
    delay(300);
    break;
  }
 }
  else if (adc_key_in == 0){
   if (rightstatus == 0) {
    rightstatus = 1;
    Serial.print("Right Status = ");
    Serial.println(rightstatus);
    delay(300);
    break;
   }
   else if (rightstatus == 1) {
    rightstatus = 0;
    Serial.print("Right Status = ");
    Serial.println(rightstatus);
    delay(300);
    break;
   }
  }
lcd.clear();
lcd.setCursor(0,0);
lcd.print(leftstatus);
lcd.print(" ");
lcd.print(rightstatus);
lcd.print(" - ");
lcd.setCursor(0,1);
if (leftstatus == 0 && rightstatus == 0){
  lcd.print("Brake");
}
else if (leftstatus == 1 && rightstatus == 0) {
  lcd.print("Clockwise");
}
else if (leftstatus == 0 && rightstatus == 1) {
  lcd.print("Counterclockwise");
}
else {
  lcd.print("Brake (");
  lcd.print(timeout);
  lcd.print(")");
  timeout--;
  if (timeout < 1) {
```

```
   timeout = 120;
   rightstatus = 0;
   leftstatus = 0;
   break;
  }
 }
 } //end while loop
} // end loop()


int get_key(unsigned int input) {
 int k;
 for (k = 0; k < NUM_KEYS; k++) {
  if (input < adc_key_val[k]) {
   Serial.print("k = ");
   Serial.println(k);
   return k;
  }
 }
 if (k >= NUM_KEYS) {
  k = -1;
 }
 return k;
}


void constantout(int left, int right, int frequency){
 float totalperiod = 1000/frequency;
 if (left == 0) {
  digitalWrite(gate1, LOW); //if left is OFF, output ground
  delay(totalperiod);
 }
 if (right == 0) {
  digitalWrite(gate2, LOW); //if right is OFF, output ground
  delay(totalperiod);
 }

 if (left == 1 && right != 1){
  digitalWrite(gate1, HIGH); //if 1 0, have pulse for fraction of time then off
  delay(totalperiod);
 }
 if (right == 1 && left != 1) {
  digitalWrite(gate2, HIGH);
  delay(totalperiod);
 }
 if (right == 1 && left == 1) {
  digitalWrite(gate1, HIGH);
  digitalWrite(gate2, HIGH);
  delay(totalperiod);
 }
} //end constantout
```

You will also need an Arduino LCD board to display your output. Because the LCD board may or may not have external output pins, we may have to create space for us to take the output from the Arduino (Pins 11 and 12) while still using the LCD. If the LCD has no external output pins, you need to use two 8-pin sockets and two 6-pin sockets and mount them on the Arduino, followed by the LCD on top of the pin sockets. See **Figure 5** below for a final look at what your controller will look like.

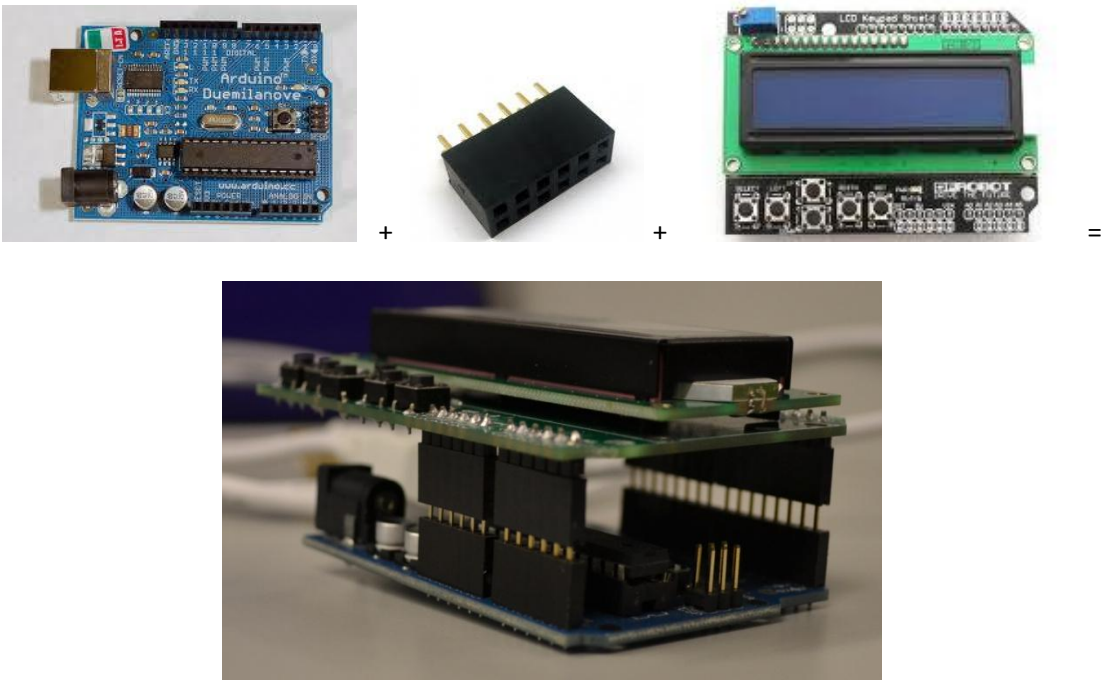

Figure 5: Controller Components and Final Configuration

Connect wires with clips to Digital pins 11, 12 and GND at the bottom of your pin sockets. Be sure that they only touch one pin and there are no accidental shorts. Connect those wires to your H-Bridge circuit inputs (Input 1 and Input 2 – bypass the switch in this case). Upload your code and test out the controller! Explore the buttons on the LCD shield. How does the LCD shield control the circuit?

Part III: Speed Control

Input Variation

We have been putting in as inputs constant DC signals. However, you can control the DC motor by changing the type of input signal. With the H-bridge inputs connected directly to your function generator again (back through the switch - not the Arduino for now), input a 5 V, 2.5 V offset square wave as the input for the 4 different circuit configurations. Experiment with the frequency and duty cycle of the square wave. What are the ranges of both parameters such that the motor still runs? How do these parameters affect the speed of the motor?

Arduino Control

Finally, as you may have noticed in the code, the output of the Arduino is controlled by a function called 'constantout'. This function simply makes the outputs HIGH or LOW depending on the user input on the LCD board, and keeps them constant at this level.

Create a new function called 'pulse' that fulfills these criteria:

1. Output is void

2. Four input variables - two input statuses 'leftstatus' and 'rightstatus', a duty cycle variable, and a frequency variable.

3. Depending on the state of 'leftstatus' and 'rightstatus', outputs to one of the digital pins a one period square wave with period determined by the frequency and time on HIGH or LOW determined by the duty cycle. (*Hint: Remember your if-statements and the Arduino function 'digitalWrite'*). We only need to create a square wave for one period because the while-loop in the code repeats this function indefinitely.

Comment out the old 'constantout' function call in the code, and instead insert your 'pulse' function. Be sure to create duty cycle and frequency variables. Upload the code, connect the output pins to the H-Bridge, and test to see how the motor performs. Show your TA as your exit ticket.

Written by Vignesh Subramanian, Matthew Kaye, and Parth Chopra, Mar. 2012.