The IED H-Bridge Circuit Board (v.3)
Last Rev. 9/28/05 Stephen Rock & Matthew Rosmarin

You should thoroughly read and understand this document before attempting to use the IED H-Bridge Board. Else, you may find that fixing a damaged board is a miserable experience.

Introduction

A circuit known as an H-bridge can be used to achieve speed and direction control when using permanent magnet DC (direct-current) motors such as those common in toys, cell phones, and cars. (See: http://electronics.howstuffworks.com/motor1.htm for a good introduction on motors.)

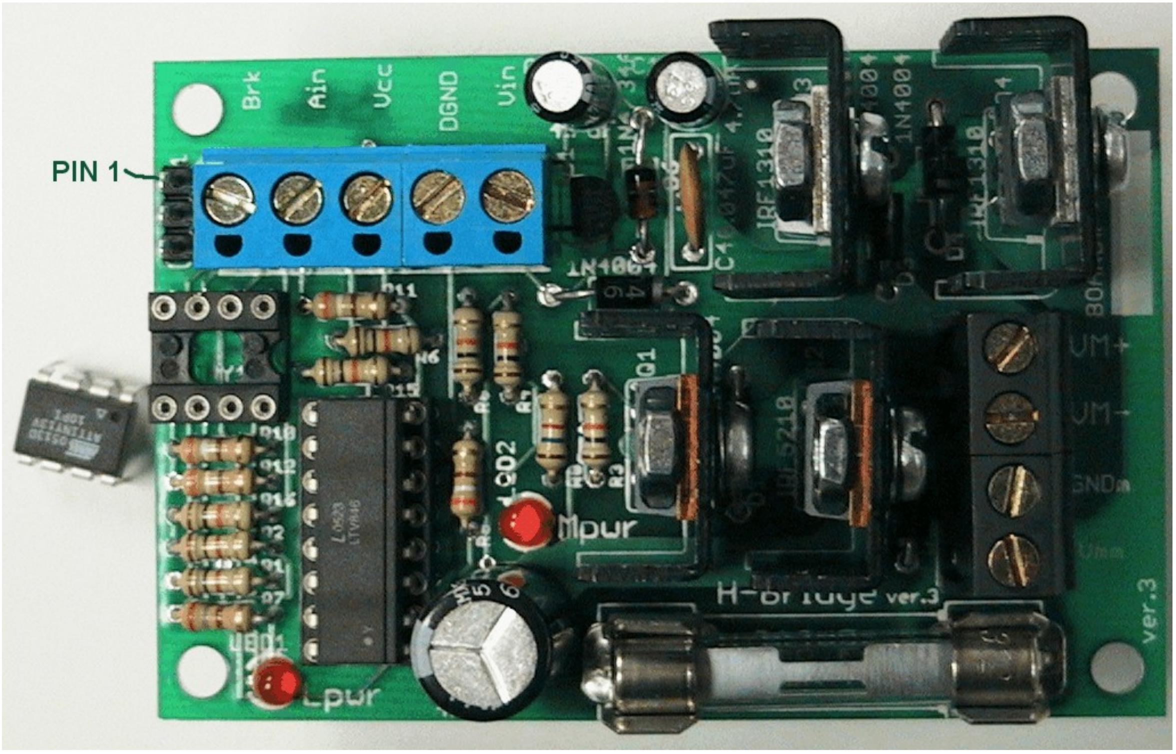
IED students have historically had a need to control a range of motor sizes, and since voltage ranges allowable in the class are limited to 24VDC, permanent magnet brushed DC motors are the usual choice for rotary motion. If on-off control is all that is required, a relay could be used. If forward-off-reverse control is required, one relay can be used to control the on-off function, and a second relay (or a DPDT, that's double pole double throw, switch) can be used to reverse the direction of the motor. (See the powerpoint slides on reserve in the library for more detail and beware that when an energized coil goes open circuit it can generate a large voltage spike capable of shocking you or destroying transistors that are not protected by a diode as shown in the slides.)

The same forward-off-reverse functionality can be achieved using a transistor-based H-bridge circuit. However, since the transistors can switch considerably faster than relays, they can also be switched in such a way that only partial power (on average) is applied to the motor. The motor is turned on for a fraction of a second, and then back off, and this is repeated thousands of times a second to achieve a particular speed. This is known as pulse width modulation, and more on this subject can be found at http://www.embedded.com/story/OEG20010821S0096 or from a search of this term or "PWM" on your favorite web search engine.

For very small motors, it may be practical to use an integrated circuit chip such as the L293 (see: http://www.st.com/stonline/products/literature/ds/1330/l293d.pdf or http://rocky.digikey.com/WebLib/Texas%20Instruments/Web%20data/L293,%20L293D.pdf). However, these are limited to relatively low currents (max of 1 amp, or 2 amps if combined), which can easily be exceeded when a motor is under heavy load (or stalled – where the rotor is kept from spinning). Exceeding the current can destroy the driver chip, and this usually happens at the last possible minute when some unforeseen event occurs. (Hint: It is a very good idea to test the stall current of your motor before selecting a driver.)

Overview

We have developed an H-Bridge driver circuit board for IED that is capable of switching considerably higher currents than chips such as the L293. This circuit board is shown in the photo below (with one chip removed – more on this later).



IED H Bridge Board – shown with microcontroller removed

The IED H-Bridge board (HB, hereafter) has been designed to switch loads of 25Amps at 24VDC, but the installed fuse is 20A and this current should not be exceeded without consulting with your instructor and increasing the fuse amperage rating.
IMPORTANT: Under no circumstances should you bypass the fuse or replace it with one of a different type or capacity without your instructor's permission – you are responsible for all damage to the HB boards! (They are not fun to repair!) The gearmotor supplied for IED projects draws less than an ampere when free running, but it has a stall current of almost 10A. If you decide to use an alternate motor, be sure the STALL CURRENT does not exceed 20A. Likewise, if you will be attempting to quickly reverse large inertial loads, it is possible that you can even exceed the stall current of the motor because the load will cause the motor to act like a generator and resist the rapid direction change.

Additionally, it is important to note that the AC power supplies provided in IED for bench testing are limited at 6.5A and will fault (and remain off until they are power cycled) should this current be exceeded. This, too, serves to protect you and the HB board from destruction. Be aware that batteries can source *MUCH LARGER* currents, at least for a short time. Car batteries can source hundreds of amperes if the terminals are shorted (think arc welder, see: http://www.safari4x4.com.au/80scool/tech/batweld/batweld.html).

The high current 24VDC side of the HB is optically isolated from the logic side of the circuit for added protection. Only light (inside the large IC on the board) signals when the H bridge transistors should function – there is no electrical connection. Motor power is supplied with the positive terminal of a battery or power supply connected to the Vmm screw terminal. The negative terminal of the battery or power supply can be connected to the GNDm terminal. Vmm should never be at a voltage lower than GNDm (e.g. don't accidentally reverse the power supply or battery leads – convention in engineering is that red wire is positive and black wire is negative or in this case ground).

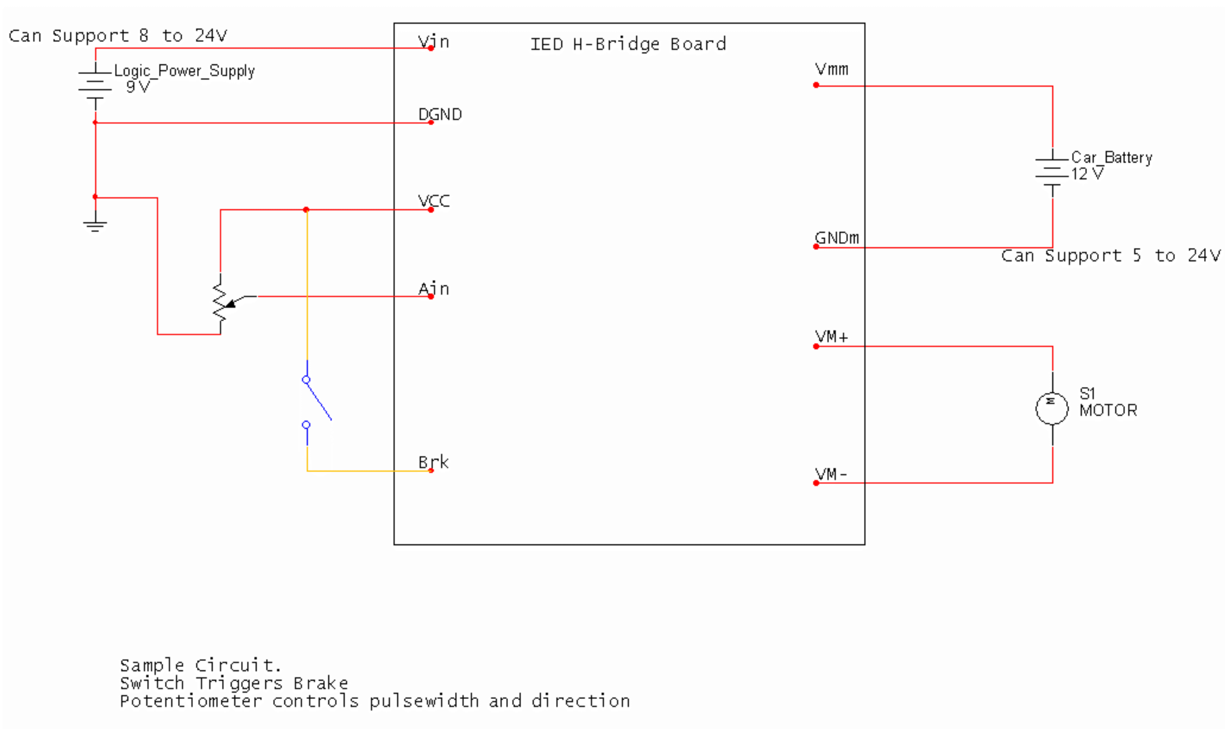
Note that the circuit board was designed so that one of the four screws (the one closest to a transistor heat sink) that mounts the board to the base plate connects with the GNDm signal. It should not be used to supply ground to your motor; however, you should take care to ensure this bolt does not come in contact with any other electrical signals or voltage sources.

Finally, the motor must be connected to terminals VM+ and VM-. If you connect the motor and then find you'd prefer forward and reverse to be swapped, simply swap the motor leads going to VM+ and VM-.

Modes of Operation

The IED HB can be used in three different ways. First, the board is equipped with a microcontroller that is pre-programmed to implement a PWM controller. This means that it accepts a DC voltage in and generates the PWM signals to control motor speed. This is the simplest method for using the HB. For those wishing to learn more about PWM, it is possible to remove the microcontroller (the 8 pin IC) and directly drive the optoisolator with TTL level (5V) logic signals. This is more complicated, as it is possible to present a combination of three signals that cause the bridge to short rail to rail (a direct short from Vmm to GNDm, not good, but if current limited by the lab supply to 6.5A, it will not destroy the bridge). Finally, it is possible to purchase an Atmel Attiny13 microcontroller of your own, program it, and substitute it on the board. This would also require considerable skill, and could prove limiting as there are few I/O pins on the microcontroller selected for this project. While this last option is not recommended, if you pursue this, please be sure to retain and protect the microcontroller furnished with the board (do not reprogram it!). You are responsible for returning a working HB at the end of the semester.

On-board PWM Generation *(the simplest option)*



The low power side of the circuit contains a microcontroller chip (the small IC) that has been programmed to accept an analog signal between 0-5V and generate Pulse Width-Modulated (PWM) motor control signals for the H-bridge based on this voltage. The motor should be off when a command voltage of approximately 2.5V is given, at full speed in one direction at 0V and fully reversed at 5V. **It is VERY IMPORTANT that the analog input not go below 0V (a negative voltage relative to DGND) or exceed 5V.** If this input is not connected, a voltage divider has been added to the circuit to set this at approximately 2.5 volts; however, due to manufacturing tolerances in the resistors used to create this voltage divider, the motor may not be completely off (but will be close). This ensures that were the wire carrying the analog speed signal to the board to become disconnected, the motor would not be commanded at full speed as would be the case when a zero volt signal is forced on this input.

There is a voltage regulator on the board so that the logic and microcontroller can be powered by a source between 8-24VDC. For this we suggest a 9V battery, or similar voltage wall-wart DC power supply (beware that some are simply transformers producing AC voltage). You need a DC power supply. It should say xVDC on the supply, where x is the voltage, and it should be capable of delivering at least 100mA of current. Note that this is *different* from the 24V 6.5A supply provided for bench testing in IED and it is each team's responsibility to furnish this power supply or battery. In theory, the regulator supplied could be used with the 24V supply provided for powering the motor; however, it is bad practice to share the motor power with the digital logic because of switching noise and the possibility for microcontroller malfunction.

The motor command voltage can be created manually by using a potentiometer (a variable resistor, often abbreviated 'pot'). The maximum value of the pot will determine how much current it draws, so since current is limited, we should stick with a pot having a resistance of at least 10k ohms. A pot between 10k and 20k ohms would be a good choice. The center terminal of the pot sweeps from one side of the resistor to the other, so by connecting this to Ain, you've created a variable voltage divider than can be used to adjust motor speed and direction.

Alternatively, we could generate the analog motor speed command voltage for Ain using a computer with a digital to analog converter or an interface such as the Labjack. When doing this, it is very important to connect the ground from the labjack to the digital ground DGND on the HB because voltage is relative and the two grounds, if isolated, could be at different potentials (i.e. from electrical noise). If using a computer generated input to Ain, **it is critical that the programmed voltage not be allowed to exceed 5V**, and the output of any programmed device should be tested using a volt meter *prior* to connecting to the HB.

Please note that the regulated 5V supply on this board is of very limited capacity (100mA) and is provided on the terminal strip only for use by the board – it should not be taken from the board and used to power any additional hardware such as TTL integrated circuits or external indicator lamps or LEDs (light emitting diodes). This would quickly destroy the regulator IC on the HB board. Please construct your own 5V regulator if necessary by copying the design here or using a higher current regulator IC (see: http://rocky.digikey.com/WebLib/Fairchild/Web%20Data/LM7805_MC78xx_MC78xxA.pdf).

Last, but not least, one input on the microcontroller (labeled Brk on the logic terminal block) implements a brake for the motor by stopping generation of PWM signals and shorting the motor leads together. This causes what is known as "back EMF" (search on the term if curious) to make the motor slow down. It is not a true mechanical brake. However, braking action will be proportional to motor speed. This signal is active high, meaning that 5V (and no more) should be applied to this screw terminal to cause the braking action. If you do not wish to use the brake feature, just leave it disconnected, there is no need to ground it as it is pulled low on board by a pull-down resistor. This signal can be generated using a contact closure (switch or relay) to Vcc, or it can be connected to a logic output port on a computer or Labjack. Again, be sure to connect a ground between the HB and whatever is generating the Brk signal.

Direct H-Bridge Control

It is also possible to directly control the H-bridge by removing the microcontroller chip (the only 8 pin DIP packaged IC on the board – shown removed in the first photo) and connecting to the three header pins next to the logic terminal strip. The header pins are denoted pins 1,2, and 3, with pin 1 labeled on the circuit board silk screening and oriented furthest from the microcontroller IC.

These pins are standard 0.100” spacing pins, and you can find compatible pins and connector bodies by visiting www.digikey.com and searching for the part number provided on the bill of materials given in the appendix of this document.

Before attempting to drive the bridge yourself, we recommend that you observe the signals on these three pins while the microcontroller is still connected and operating. It is possible to view the PWM signals generated by the microcontroller at these pins using an oscilloscope.

Of the eight (2^3) possible signal combinations on these pins, only four are safe combinations that correspond to coast (off), brake, forward, and reverse as shown in the table below.

Logic			Optoisolator LEDs				Inverted				Corresponding				Bridge
(at 3 header pins)			(as wired)				Opto LEDs				Transistor State				Function
Fwd	Rev	Enable	A	C	B	D	~A	~C	~B	~D	A	C	B	D	
0	0	0	0	0	0	0	1	1	1	1	0	0	1	1	Brake
0	0	1	0	0	1	1	1	1	0	0	0	0	0	0	Coast
0	1	0	1	0	0	0	0	1	1	1	1	0	1	1	X
0	1	1	1	0	1	0	0	1	0	1	1	0	0	1	Reverse
1	0	0	0	1	0	0	1	0	1	1	0	1	1	1	X
1	0	1	0	1	0	1	1	0	1	0	0	1	1	0	Forward
1	1	0	1	1	0	0	0	0	1	1	1	1	1	1	X
1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	X

Inputs: Logic 0 = 0V, Logic 1 = 5V; Transistor State 0 = not conducting, 1 = conducting

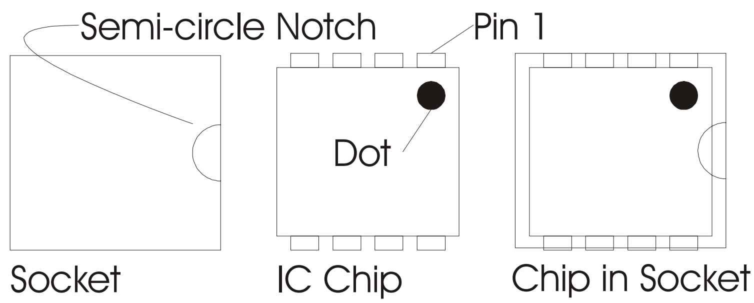
Note: Illegal bridge states denoted X short the bridge and should never be entered.

Truth table for H-bridge control signals and corresponding function.

The other illegal states will cause the bridge to short and care should be taken to ensure these states are never reached – even for very short durations. Aside from potential circuit damage, this would drain any batteries faster than otherwise necessary. When generating your own PWM signals, please be sure to use only a single power supply limited to 6.5A for initial testing. This way, if your programming or wiring does cause the bridge to enter an illegal state and short, it will fault the power supply and protect the bridge transistors from destruction. Incorrect operation when connected to a battery power source capable of delivering greater than 20 amps would likely have disastrous consequences.

If review of the schematic, attached program, and bill of materials (all in the appendix) is confusing, we recommend against attempting to directly drive the bridge. Please seek help *before* trying to implement your own PWM control if you are unsure about how this circuit works.

Before removing the microcontroller from the board, please note it's orientation. It appears "up side down" based on most board silk screen labeling and should be inserted in the socket so that the dot denoting pin 1 of the chip is toward the notched end of the IC socket as shown below.



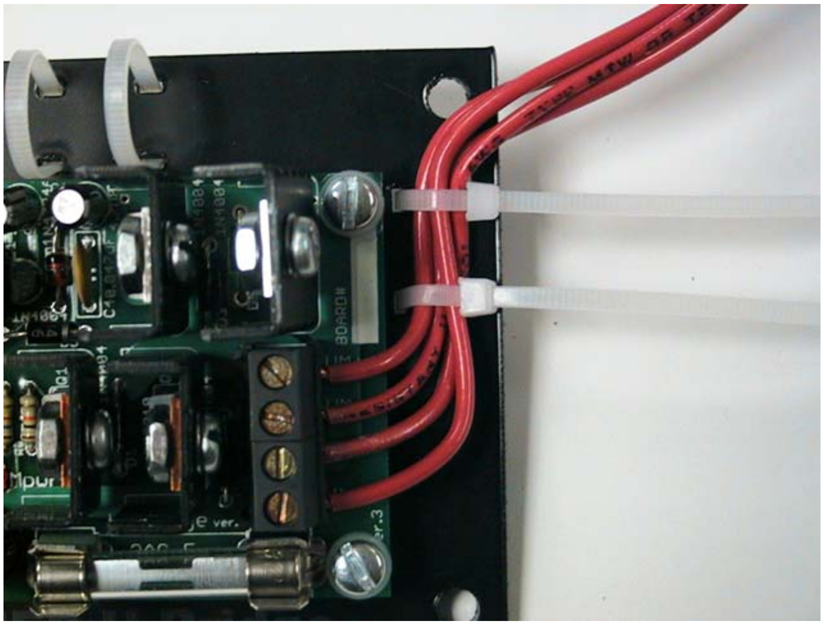
Microcontroller Programming

For those who may be interested in replacing the microcontroller with one having their own program, the following is provided; however, this is not recommended and will be of limited utility given the low I/O count of the controller selected.

The Atmel Attiny13 can be programmed using the Atmel AVR programmer. This programmer comes with a complete development environment for a very reasonable single unit price of $29/ea and is available at Digikey.com. A C compiler for various AVR microcontrollers is available for free (when used for non-commercial purposes) from: http://www.hpinfotech.ro/html/download.htm and the code size limit does not impact use with the Attiny13 chip. The Atmel Attiny13 microcontroller in the HB was programmed in C using the program shown in the appendix. Please do not reprogram the microcontroller – purchase and work with your own if you wish to experiment with microcontroller programming on the HB board. Note that the HB does not facilitate in circuit programming of the microcontroller – the chip will have to be removed from the board for programming.

Physical Integration

The HB circuit boards are attached to a plastic backer and should not be removed. This serves three purposes. First, it protects the boards with live electrical connections exposed on the back from being inadvertently set on a wrench or other conductive surface (yes, this has been done in IED in the past with far more expensive circuit boards). Second, it provides a more durable point at which to attach the HB to your projects and minimizes risk that the printed circuit board of the HB itself becomes damaged (e.g. over-tightening a screw, or using a single screw and snapping a corner off the board). Finally, and most important for the continued success of your circuits in the 11th hour, it provides a means of strain relief for all cables going to the board. The photo below shows how small tie wraps can be used to secure wires going to the HB board, thus making them less likely to be torn out of the terminal strips.


Partial view of H-bridge board backer and strain reliefs

WARNINGS

1. The HB boards are sensitive to static electricity (e.g. in the dryer air of winter, you take off a sweater, or scuff your feet across some carpet, then touch a refrigerator or other grounded surface and feel/see/hear a spark jump). To prevent damage to the board, please touch something grounded, such as a water faucet, or electrical conduit, before handling or working with your board when the relative humidity in the air is low (especially in the winter).

2. The components rising above the HB board are not made for rough handling. In particular, the heat sinks attached to the bridge transistors will easily catch and be destroyed by shoving this board in a backpack, etc. It is important to protect the board as though it's going to save you 10's of hours of work – a working board can do this. Please find a box or case to transport the board in and consider how you will protect the board once it is attached to your IED project.

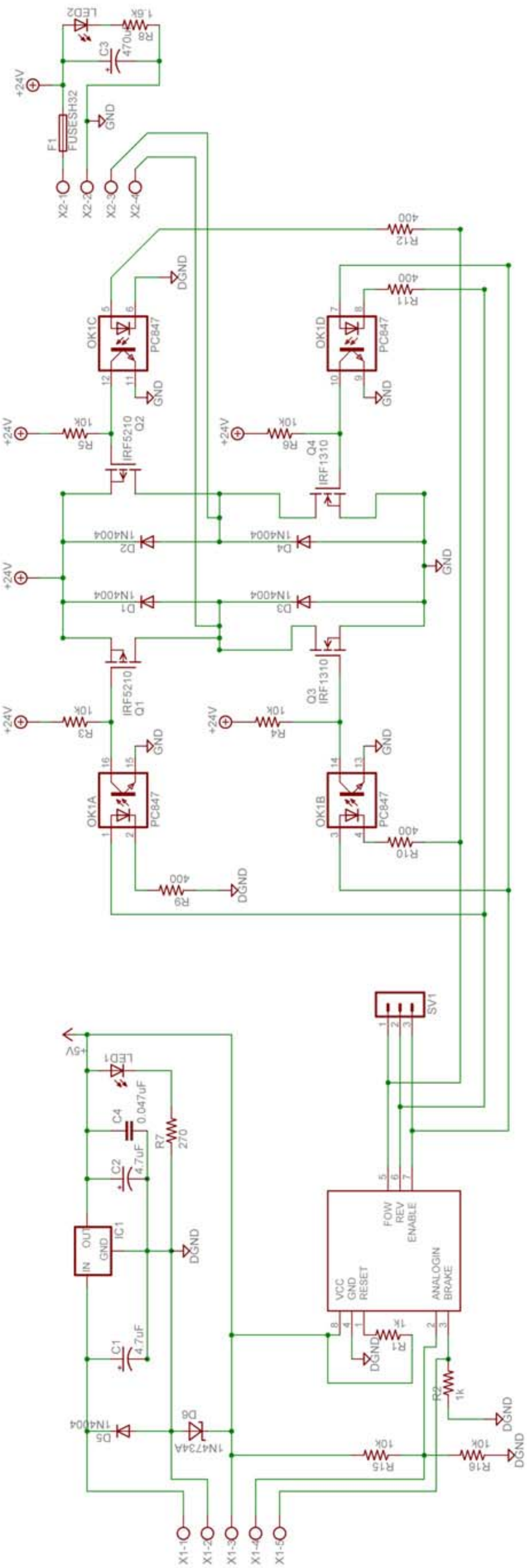
3. The bridge transistor heat sinks are *not* electrically insulated and must not be touching one another or any other conductor. They can easily be folded into one another by less than careful handling of the board.

4. Heat sink temperature should be periodically monitored and if the motor will run under heavy load and draw close to the 25ampere design limit, it may be necessary to supply a forced air cooling fan to more efficiently dissipate heat from the transistors.

Appendix

A – Circuit Schematic
B – PWM Program
C – PWM and Motor Voltage Example Plots
D – Board Layout
E – Bill of Materials
F – So you want to make your own printed circuit board…

Appendix A – Circuit Schematic



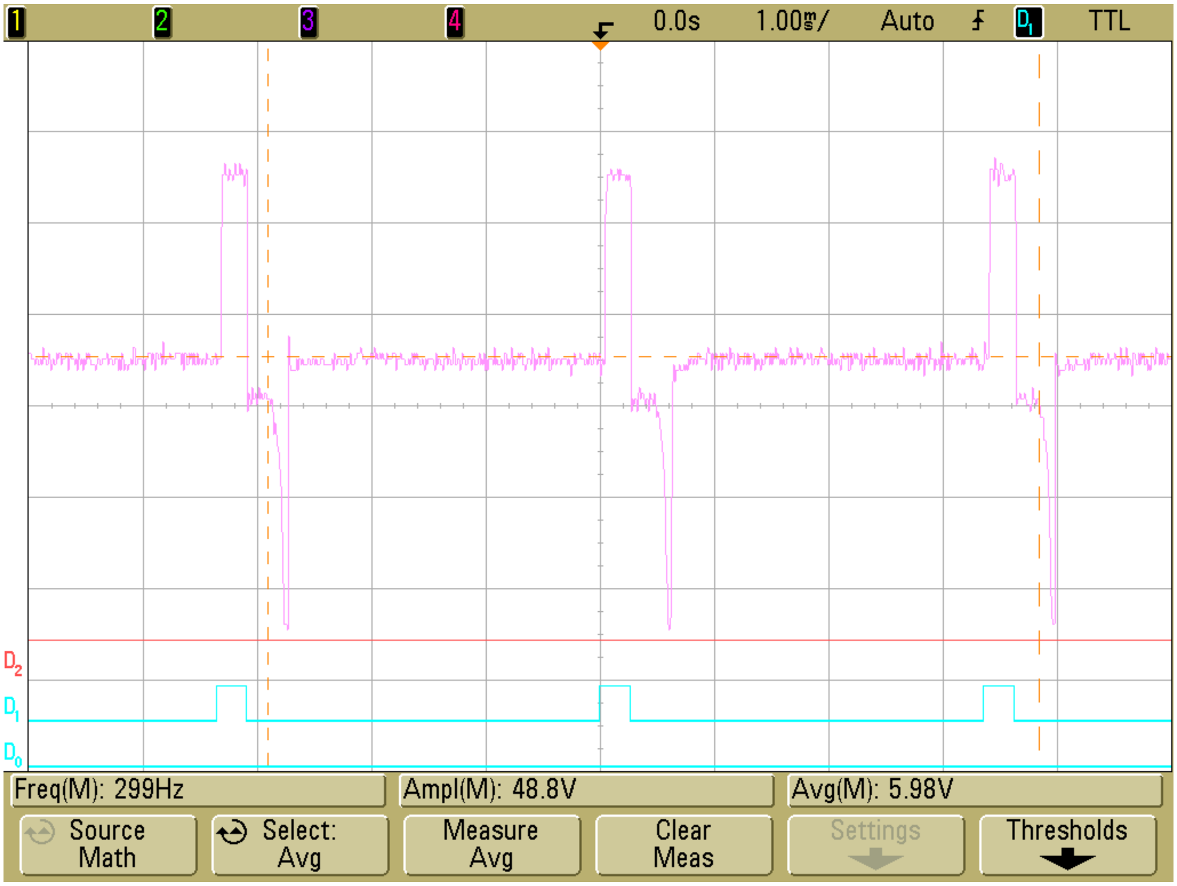
Appendix B – PWM Program

```c
/************************************
IED H-Bridge PWM Firmware (v.3)
By Matthew Rosmarin and Stephen Rock
Last Updated on Sept 27, 2005
************************************/

//Include Statements
#include <tiny13.h>
#include <delay.h>
#define ADC_VREF_TYPE 0x00
//*****************************

// Analog to Digital Conversion Function
// Converts 0V to +5V into a numeric value from 0 to 1024
// "adc_input" is the ADC pin number you wish to convert
// Use: read_adc(3); // would return the result for ADC pin 3
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input|ADC_VREF_TYPE;
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}
//******************************

// Start of program
void main(void)
{
// Declare Variables
int command;
int pa,pb ;
//******************

//Initialization Code (this code sets up the microcontroller)
              // Crystal Oscillator division factor: 1
              CLKPR=0x80;
              CLKPR=0x00;
              // Input/Output Ports initialization
              // Port B initialization
              // Func5=In Func4=In Func3=In Func2=Out Func1=Out Func0=Out
              // State5=T State4=T State3=T State2=0 State1=0 State0=0
              PORTB=0x00;
              DDRB=0x07;
              // Timer/Counter 0 initialization
              // Clock source: System Clock
              // Clock value: 150.000 kHz
              // Mode: Phase correct PWM top=FFh
              // OC0A output: Non-Inverted PWM
              // OC0B output: Non-Inverted PWM
              TCCR0A=0xA1;
              TCCR0B=0x03;
              TCNT0=0x00;
              OCR0A=0x00;
              OCR0B=0x00;
              // External Interrupt(s) initialization
              // INT0: Off
              // Interrupt on any change on pins PCINT0-5: Off
              GIMSK=0x00;
              MCUCR=0x00;
              // Timer/Counter 0 Interrupt(s) initialization
              TIMSK0=0x00;
              // Analog Comparator initialization
              // Analog Comparator: Off
```

```c
            ACSR=0x80;
            ADCSRB=0x00;
            // ADC initialization
            // ADC Clock frequency: 150.000 kHz
            // ADC Bandgap Voltage Reference: Off
            // ADC Auto Trigger Source: None
            // Digital input buffers on ADC0: On, ADC1: On, ADC2: On, ADC3: On,
            // ADC4: On
            DIDR0=0x00;
            ADMUX=ADC_VREF_TYPE;
            ADCSRA=0x86;
//***************************************************************
// End of Initialization code

// Main Program Loop (runs forever)
while (1)
      {
      //Compute Desired Command Values
      command=read_adc(3);     // Get command value
      command=command-512;    // Scale Value
      command=command/2;
         //********************************

                  if(read_adc(2)>100) //check to activate break
                  {
                        pa=0x00;
                        pb=0x00;
                        OCR0A=0x00;
                        OCR0B=0x00;
                        delay_ms(8); //Ensures no illegal state while switching
                        PORTB=0x00;
                  }
                  else  //break mode not selected
                  {
                        if(command>0)        // Forward mode
                        {
                              PORTB=0x04;
                              pa=command;
                              pb=0x00;
                        }
                        else                 //Reverse mode
                        {
                              PORTB=0x04;
                              pa=0x00;
                              pb=-command;
                        }
                  }
            //Check to make sure output value is valid
            if(pa>255) pa=255;
            if(pa<0)pa=0;
            if(pb>255) pb=255;
            if(pb<0)pb=0;
            //*****************************************

            //Set Output
            OCR0A=(char)pa;
            OCR0B=(char)pb;
            //********************************************
      };
}
```

Appendix C – PWM and Motor Voltage Example Plots

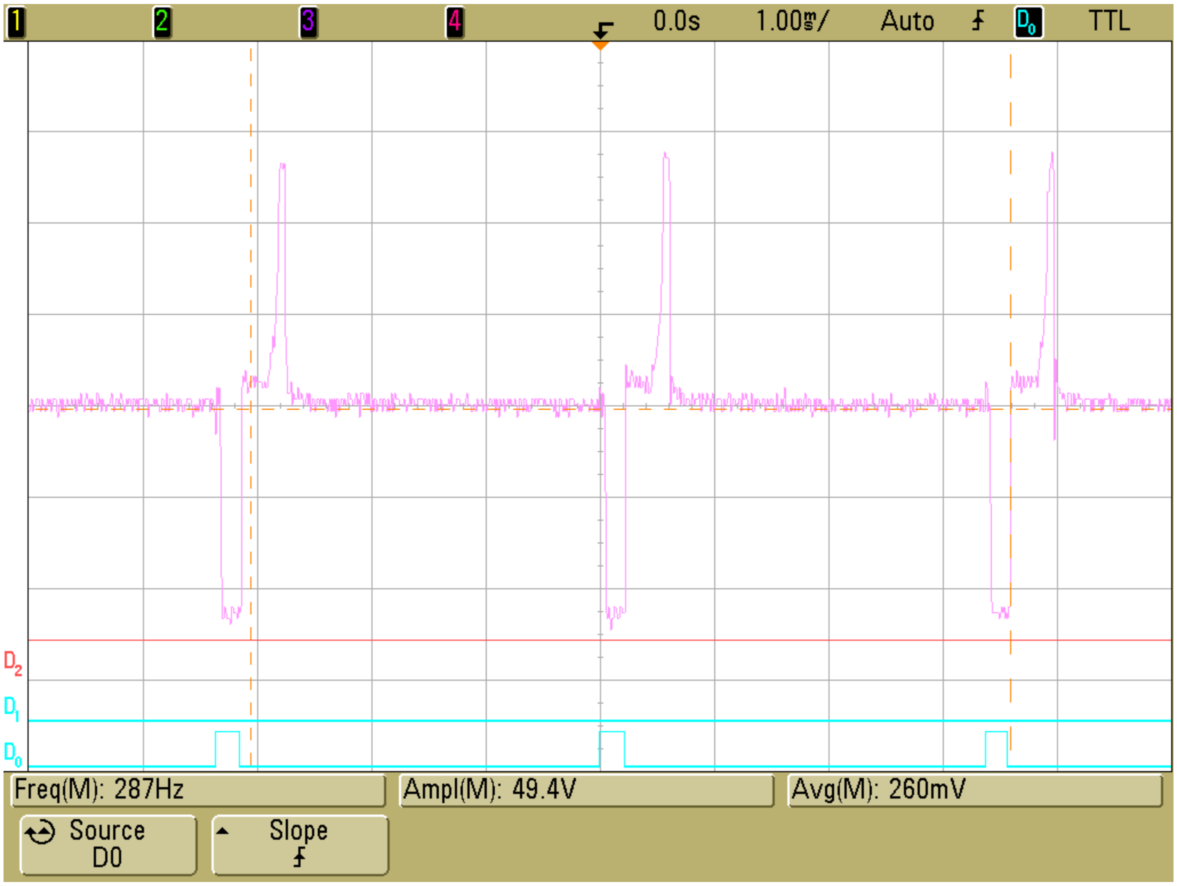
The following plots show the voltage across the motor on the top (analog) trace and also show the state of the three bridge inputs where D0=FWD, D1=REV, and D2=ENABLE. In this case the motor is being driven in "reverse" with respect to the bridge as the FWD line is logic low, the enable signal on D2 is high, and the pulses occur on the REV line shown by D1.



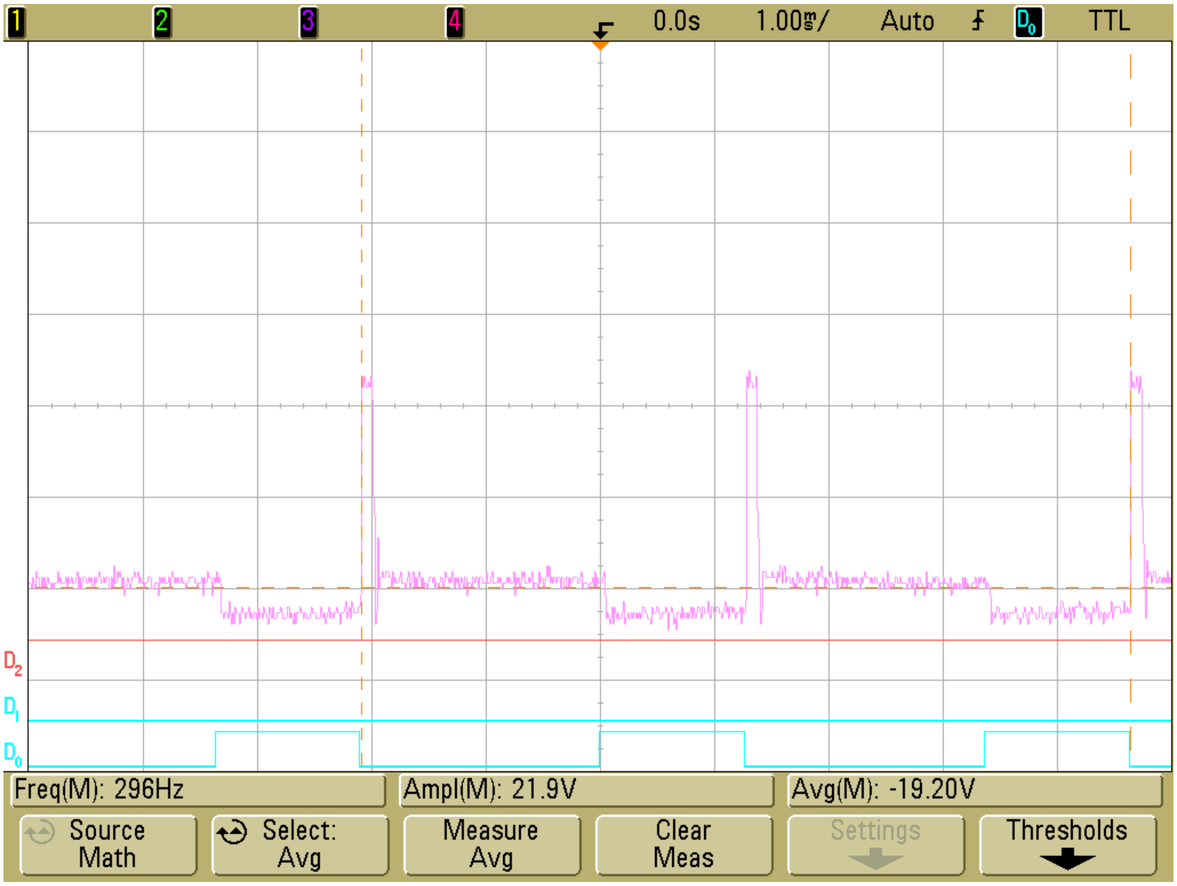
This plot shows a running motor (not stalled) that has some voltage above zero (the centerline of the plot) because the motor is free to spin and generates a voltage proportional to its speed. When the PWM signal to the bridge energizes the transistors, the voltage spikes up for the duration of the PWM pulse. The large negative going pulse observed after the bridge shuts off is due to back EMF in the motor. The motor voltage then returns to the level corresponding to its speed of rotation. Note that as the motor becomes more heavily loaded, the average voltage will decrease for a given PWM duty cycle.

In order to "control" speed, we would require feedback and a control system, so although our final motor speed is a function of our PWM duty cycle, it is also dependent on the motor loading. Feedback control would be required to maintain a precise speed under varying loads.

The following plot shows the motor operating with pulses on the FWD line. Note that the motor voltage is reversed, and that is why the waveform appears to be inverted.

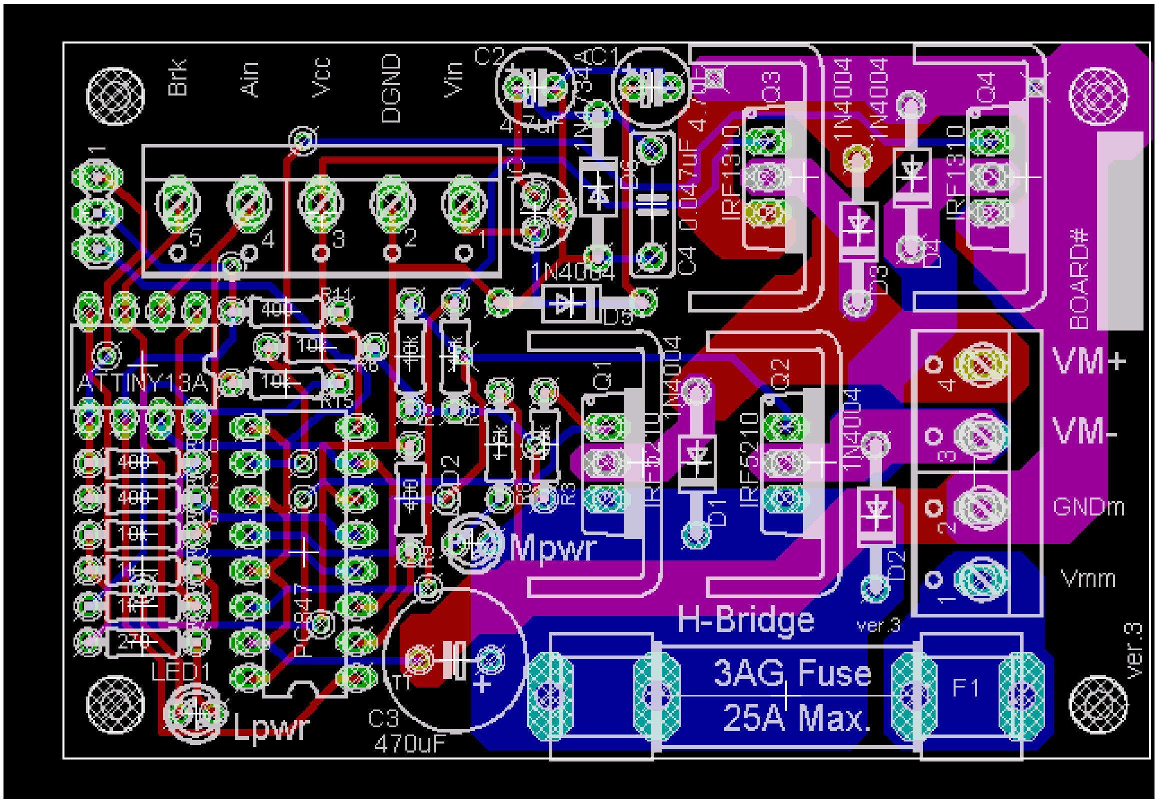


This plot shows an average voltage of only 0.260V and corresponds to a motor that just barely has enough power to rotate. If a higher speed were required, the duty cycle of the PWM pulses would be increased as shown in the following plot.

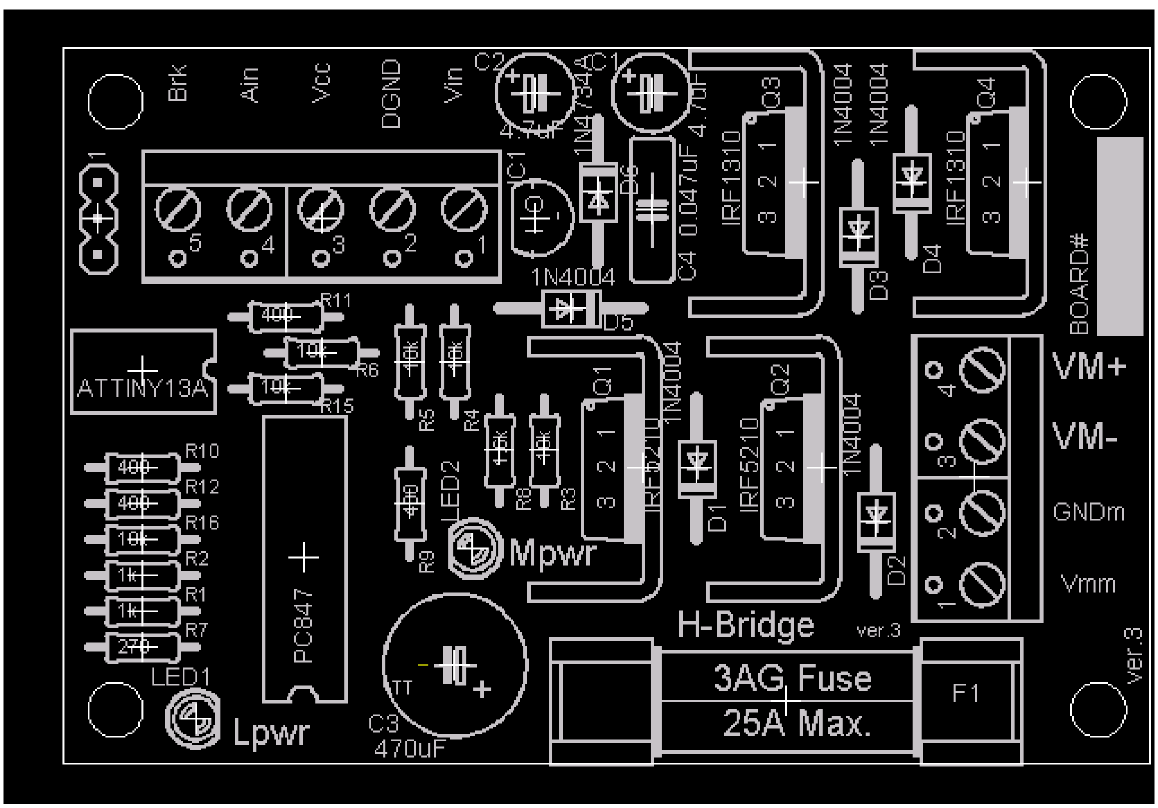


In this case the average voltage is considerably greater (in magnitude) because the motor is running. Also of note is that the PWM frequency is very near the constant 300Hz programmed, but it is the duty cycle, or length of time the pulse is on vs. off, that is changed to vary the speed of the motor.

Appendix D – Board Layout



Circuit board with top and bottom traces shown.



Circuit board silk screen only (for component ID and placement)

Appendix E – Bill of Materials

IED/Mechatronics H-Bridge Board Bill of Materials
Last Rev. 9/26/05

Line	Item	Manufacturer	Mfg. Part No.	Source	Source Part No.
1	HEX/MOS N-CH 100V 42A TO-220AB	IRF	IRF1310N	Digikey	IRF1310N-ND
2	HEX/MOS P-CH -100V -40A TO-220AB	IRF	IRF5210	Digikey	IRF5210-ND
3	Heatsink TO-220 Bolt On	Wakefield	273-AB	Digikey	345-1022-ND
4	SCREW MACHINE SLOTTED 6-32X1/4	Building Fastners	H154-ND	Digikey	PMS 632 0025 SL
5	NUT HEX 6-32 ZINC PLATED	Building Fastners	H220-ND	Digikey	HNZ632
6	#6 internal tooth lockwasher	Building Fastners	H240-ND	Digikey	INT LWZ 006
7	*Diode*	*General Semiconductor*	*1N4004*	*Mouser*	*512-1N4004*
8	Terminal block (motors)PCB 5.08MM 4POS	Weidmuller	999392	Digikey	281-1437-ND
9	*FUSE 20A 32V 3AG FAST ACT*	*Littelfuse Inc*	*0312020.H*	*Mouser*	*576-0312020.H*
10	CLIP FUSE 1/4" EAR PC MOUNT	Littelfuse	01020071H	Digikey	F040-ND
11	Quad optoisolator	LITE-ON INC	LTV-846	Digikey	160-1364-5-ND
12	Dip socket	Mill-Max Manufacturing	110-99-316-41-001000	Digikey	ED3116-ND
13	RES 270 OHM 1/4W 5% CARBON FILM	Yageo	CFR-25JB-270R	Digikey	270QBK-ND
14	RES 390 OHM 1/4W 5% CARBON FILM	Yageo	CFR-25JB-390R	Digikey	390QBK-ND
15	RES 1.0K OHM 1/4W 5% CARBON FILM	Yageo	CFR-25JB-1K0	Digikey	1.0KQBK-ND
16	RES 1.6K OHM 1/4W 5% CARBON FILM	Yageo	CFR-25JB-1K6	Digikey	1.6KQBK-ND
17	RES 10K OHM 1/4W 5% CARBON FILM	Yageo	CFR-25JB-10K	Digikey	10KQBK-ND
18	CAP 470UF 50V ELECT SMG RAD	United Chemi-Con	ESMG500ELL471MJ2	Digikey	565-1113-ND
19	*Molex .100 K.K. Connectors 3pos*	*Molex*	*22-28-4030*	*Mouser*	*538-22-28-4030*
20	STANDOFF HEX 6-32THR ALUM .250"	Keystone Electronics	2208	Digikey	2208K-ND
21	6-32x3/16 pan head mach screw	?		Fastenal	28781
22	6-32x5/16 pan head mach screw	?		Fastenal	28789
23	#6 nylon washers	?		Fastenal	76058
24	4" wire ties (for strain relief on backers)	?		Fastenal	63121
25	Terminal block (controller)			Digikey	ED2228-ND
26	Atmel ATtiny13 Microcontroller	Atmel	ATTINY13V-10PI-ND	Digikey	ATTINY13V-10PI-ND
27	DIP socket - 8 pin	Mill-Max Manufacturing	110-99-308-41-001000	Digikey	ED3108-ND
28	5V regulator (100ma LIMIT)	TI	UA78L05ACLP	Digikey	296-1365-ND
29	CAP 4.7UF 50V ELECT SMG RAD	United Chemi-Con	ESMG500ELL4R7ME	Digikey	565-1105-ND
30	.047 UFD 25V DISC CAP	Panasonic - ECG	ECK-F1E473ZVE	Digikey	P4307-ND
31	*5.6V zener*	*Vishay*	*1N4734A*	*Mouser*	*78-1N4734A*
32	LED 3.1MM 650NM RED DIFFUSED	Rohm	SLR-342VR3F	Digikey	511-1250-ND

Appendix F – So you want to make your own printed circuit board…

There are a variety of board houses that will turn printed circuit boards in very short times and at a far higher quality than one can ever hope to manually produce. We used Advanced Circuits for this project. They offer a service for prototypes that has an especially attractive price for students, see: www.barebonespcb.com and when it was time to order production boards (www.4pcb.com) they did an excellent job.

The circuit boards on which the HB is built have copper traces that are 3x the typical thickness (specified as 3oz copper per square foot, instead of the more typical 1oz copper board). This was necessary to carry the 25A maximum current for which the boards were designed.

These boards are 2 sided (no internal layers in the boards), have plated through holes, a solder mask, and silk screening.

We used Eagle schematic capture and board layout software from CadSoft (http://www.cadsoft.de/freeware.htm), which is free for non-commercial use on boards below a certain size. We downloaded the software on a Wednesday morning and had the first round of boards out for prototyping by Friday night. The software was very easy to learn (but it helped that we had previous experience with other software and the board design process). It took two additional iterations before we had all components properly defined and the final board design ready for a production order. The design process is often iterative.

Finally, this board was soldered using lead free solder. Kester (www.kester.com) recommended and generously supplied two lead free solders, both of which worked very well. To those experienced with soldering using conventional solder, the joints look like “cold joints” due to their dull appearance; however, this appearance is the norm for lead free solder. We’d recommend using lead-free solder for all future projects – it’s better for your health, better for the environment, and it’s the future.

Good luck!