

Wireless Bluetooth Controller for DC Motor

ECE 445
Final Report
May 1, 2007

Team Members:

Abhay Jain

Reid Vaccari

TA: Brian Raczkowski
Professor Gary Swenson

TABLE OF CONTENTS

1. INTRODUCTION	3
1.1 Motivation.....	3
1.2 Objectives	3
1.2.1 Goals.....	3
1.2.2 Functions.....	3
2. DESIGN	3
2.1 Block Diagram.....	4
2.2 Block / Subproject Descriptions	4
2.2.1 PC / User Input	4
2.2.2 USB Bluetooth Adapter.....	5
2.2.3 BlueSMIRF Bluetooth Module	5
2.2.4 Microcontroller	6
2.2.5 12 V Battery	8
2.2.6 Voltage Regulators	8
2.2.7 H-Bridge.....	8
2.2.8 12 Vdc Motor.....	9
2.3 Performance Requirements.....	10
3. DESIGN VERIFICATION	11
3.1 Testing Procedures and Results	11
3.1.1 PC to Bluetooth Adapter Module.....	11
3.1.2 Control Unit.....	11
3.1.3 H-Bridge	11
3.1.4 12 Vdc Motor.....	12
3.2 Thoroughness.....	12
3.3 Tolerance Analysis	12
4. COST ANALYSIS	14
4.1 Labor.....	14
4.2 Parts	14
5. CONCLUSIONS.....	15
5.1 Accomplishments	15
5.2 Uncertainties	15
5.3 Ethical Considerations	16
5.4 Future Work.....	16
5.5 Alternatives.....	16
6. APPENDIX.....	17
6.1 Current Analysis	17
6.2 In Motor Current vs. Applied Motor Voltage Plots.....	17
6.3 H-Bridge Simulation.....	19
7. REFERENCES	23

I) Introduction

1.1) Motivation

We were drawn to the idea of this project from a similar idea in the suggested Power project ideas as provided by the TAs. We found it to be particularly interesting because we are both interested in power and the idea of a wirelessly controlled motor seemed fascinating to us. We also feel that this could be a very practical idea as wireless technology is becoming increasingly more available. So, being able to control a motor through a wireless connection on a laptop could considerably enhance flexibility. The idea of using Bluetooth technology was also a big motivation for our project as we are both familiar with the technology but are very interested in obtaining a much more comprehensive understanding of how it works.

1.2) Objectives

1.2.1) Goals

The goal of this project was to design a controller that would be able to run a DC motor wirelessly using Bluetooth technology. This controller functions through a software application on a laptop or desktop computer from within 60 feet of the motor.

1.2.2) Functions

The motor has the functionality to start, stop, accelerate, and decelerate through the push of commands on the computer. The software is a simple one-screen Windows based application.

Benefits:

- Practical
- Provides Flexibility
- Economical
- User-friendly
- Can be ran from any PC running Windows

Features:

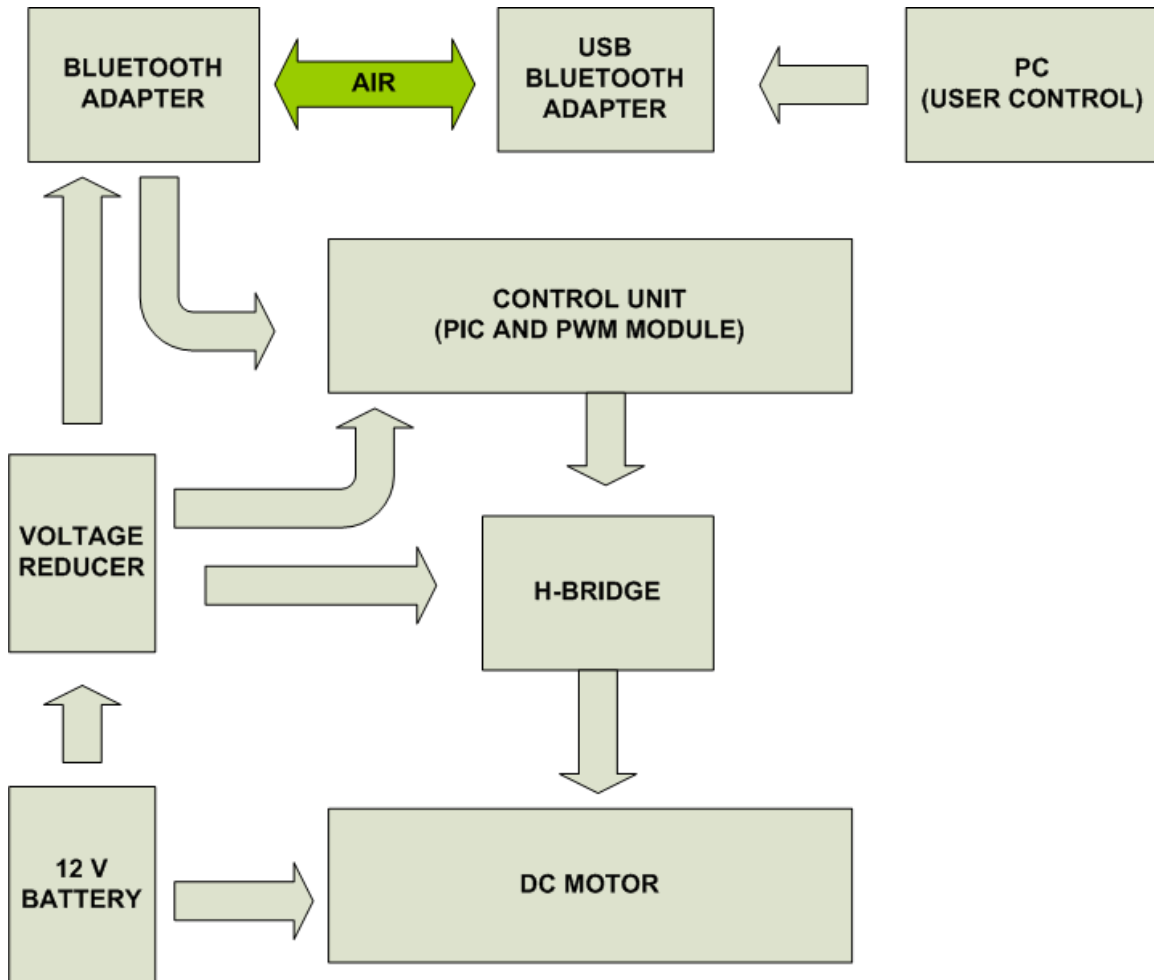
- Windows based
- Battery operated
- Wireless control via Bluetooth
- Adjustable Speed
- User has ability to start, stop, accelerate or decelerate the DC motor

II) Design

Over the course of time between the proposal and the design review, the design was modified to be more efficient by removal of the DC to DC step down converter. The purpose of this was to physically alter the amount of voltage that reaches the motor using analog components. However, it was proven to be unnecessary as the PWM module can adopt this functionality by altering the pulse width that reaches the h-bridge, and thus the

duty cycle, translating to the percent voltage that reaches the motor. Thus, the block diagram and other relevant design specifications have been updated. This design remained unchanged for the remainder of the project.

2.1) Updated Block Diagram

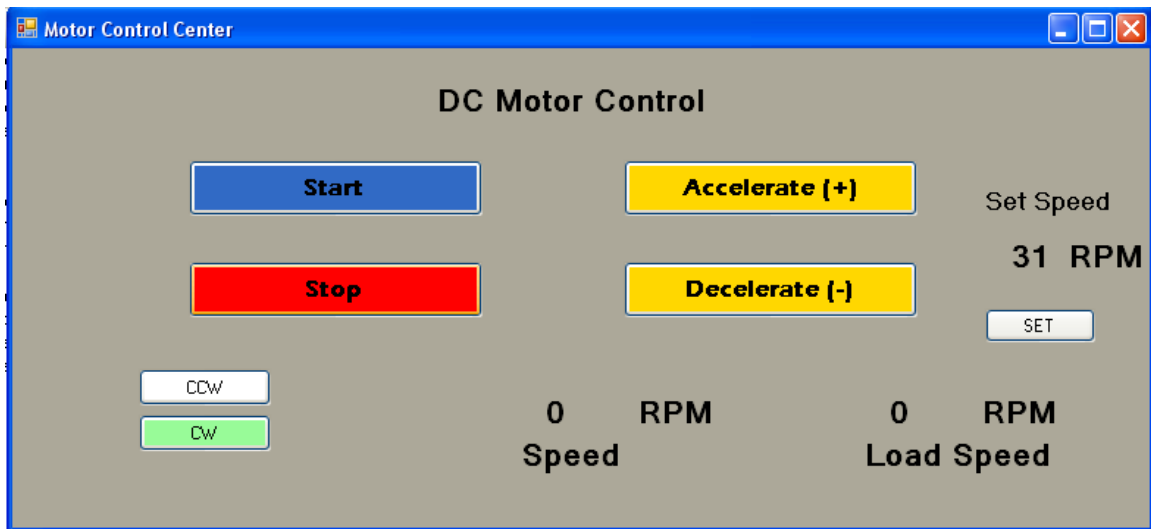
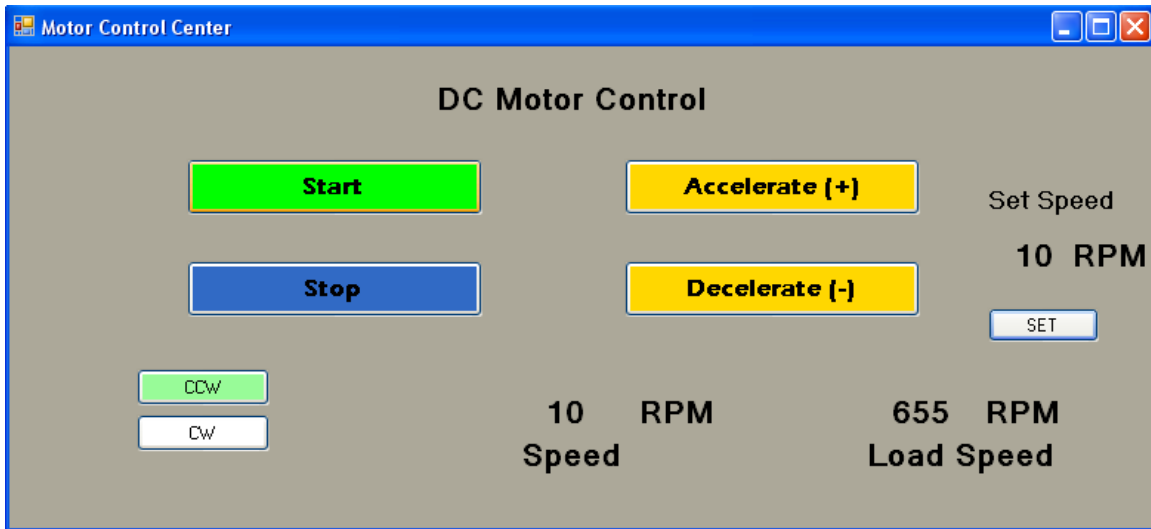


2.2) Block / Subproject Descriptions

2.2.1) PC / User Input: This block is the only point where the system accepts user input. It is a Windows-based software application that can be run on any Windows PC or laptop. Here the user is able to manipulate the various functions of the motor; run, stop, accelerate, decelerate and switch directions using easy to learn onscreen controls. The software was programmed using Visual C++ programming language. A control scheme is generated to send out a serial control signal that will represent the desired speed of the motor, in RPMs. The motor used has a gear ratio of 65.5:1, and both speeds are displayed on the screen. The speed is set relevant to the lower geared side of the motor, thus the user has the option of setting the speed from 0-95 RPM.

Once the user has selected a speed, it is outputted to the Bluetooth via a serial port connection. This is a string representation of the integer. If the motor is chosen to be operated in the counter-clockwise direction, the string outputted is just the speed.

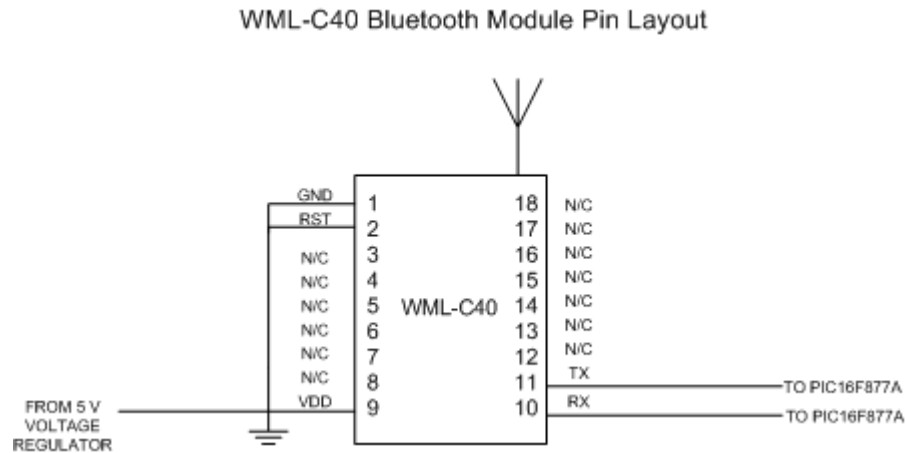
However, if the direction chosen is clockwise, then a “-” character is appended to the beginning of the control string. Thus, operating at 50 RPM in the CCW direction has a control string of “50”, whereas in the CW direction it would be “-50”.



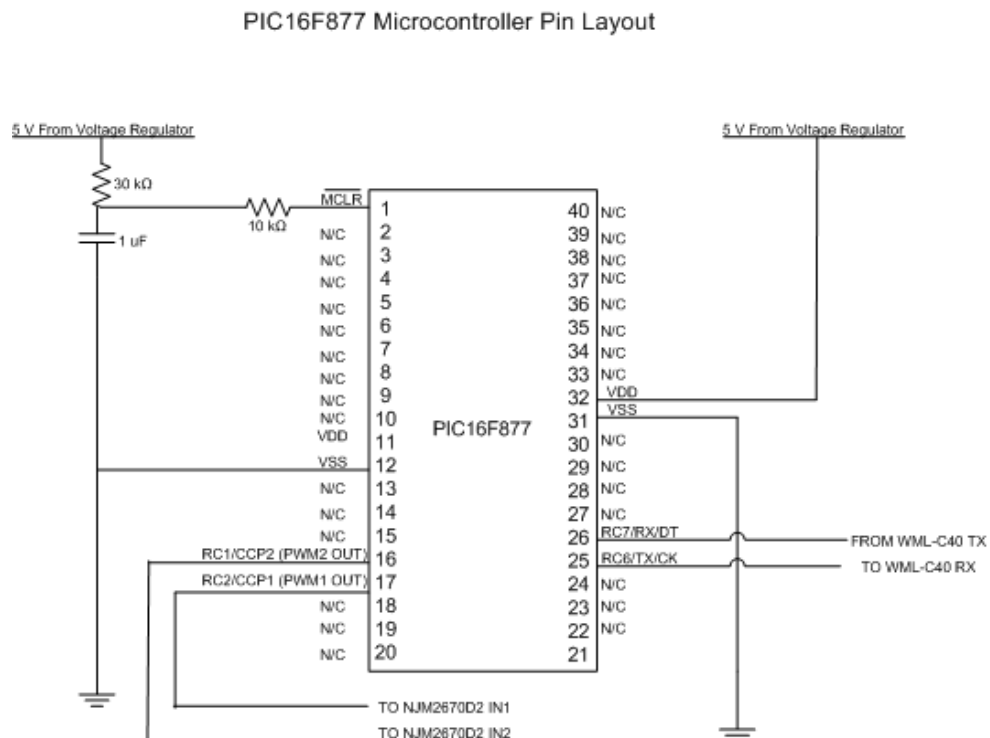
2.2.2) USB Bluetooth Adapter (SKU#: RF-BT-USB): To bridge the connection between the PIC and the PC, there are Bluetooth modules connected to both sides. The PC side is implemented using this common adapter that lets a Bluetooth connection be made as a serial link. The USB adapter can be installed easily in Windows, just as any other USB device would. The signals are received and manipulated from the motor control software.

2.2.3) BlueSMIRF Bluetooth Module (WML-C40): This is the other end of the Bluetooth wireless connection; it is the module that receives the wireless signals from the USB transmitter and sends them to the control unit. It is capable of communication via UART, PCM and USB interfaces, and we utilize the UART (serial) communication. The control signals are received from the USB Bluetooth transmitter and pass through the

WML-C40 to the PIC16F877 microcontroller via TX / RX serial communication which are pins 10 and 11 on the module. It is fed with 5 V from the regulator, which it internally steps down to 3.3 V.



2.2.4) Microcontroller (PIC16F877): The control unit consists of the PIC microcontroller and the pulse width modulator. The PIC16F877 was chosen because it was available in the ECE445 lab and it included two onboard PWMs. The PWMs are used to control the duty cycle of the motor by regulating the power output. Once the control signal from the PC reaches the PIC through the Bluetooth transmitters, its values are interpreted in the program and generate the necessary outputs. The PIC is programmed in a simplified C instruction set. The duty cycle of the motor is set by changing the length of the pulse that the PWM outputs.



There are several options for control that could have been explored, including feedback or feed forward. We chose a feed forward control scheme to be used in applications where the load is already known.

The PIC has a clock of 20 MHz provided by an external Fox F1100E oscillator. For the PWM operations, this is stepped down to 16 kHz. At 16 kHz, 416 us would be representative half the period and a duty cycle of 50%. To translate this to an integer to set the duty cycle as in the program, we use the equation:

$$\frac{.000416}{16 \cdot \frac{1}{20E6}} = 520$$

Setting the PWM duty cycle to 520 should then allow the PWM to output a duty cycle of 50%, but when the output was measured with an oscilloscope, it was seen to give a duty cycle of 52%. This also held true for other values tried, so there is a 10:1 ratio from duty cycle variable (in integer form) to the actual produced duty cycle (in percent).

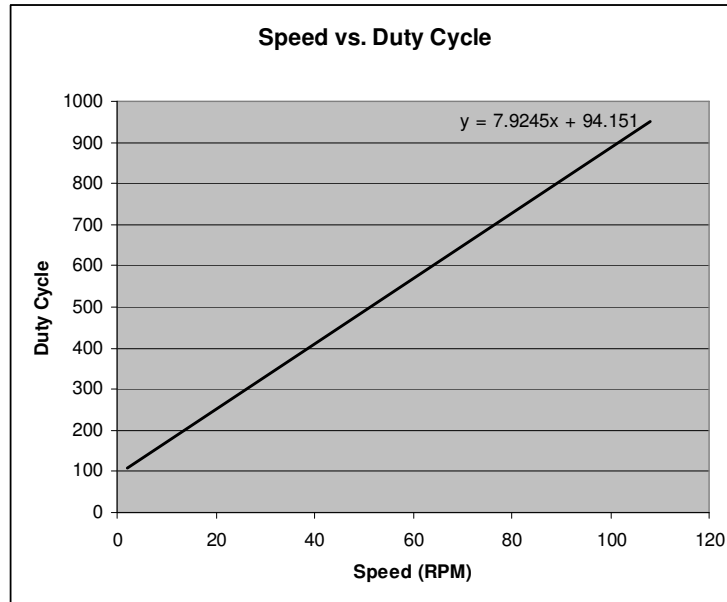
To implement this, the motor was run at no-load to measure what voltages were provided by various duty cycles. These voltages correspond to speeds, so we can construct a relationship between actual speed and the duty cycle. The data from this test are as follows in Table 1:

Duty Cycle	Speed (RPM)
110	2
475	48
667	72
844	95
950	108

Using the most extreme data points, we calculated a linear relationship between speed and duty cycle.

$$\begin{aligned}
 Y &= Mx + B \\
 \text{Duty cycle} &= M(\text{speed}) + B \\
 M &= (y_2 - y_1) / (x_2 - x_1) = (950 - 110) / (108 - 2) = 7.92 \\
 B &= 950 - (7.92 * 108) = 94.15 \\
 \text{Duty Cycle} &= 7.92(\text{speed}) + 94.15
 \end{aligned}$$

This equation was used to set the duty cycle, based on the speed that the user specified. This had to be altered slightly for each direction, due to different coefficients of friction in either direction. A plot of this linear fit is seen below.



2.2.5) 12 V Battery (BP5-12-T2): This is the power supply for the circuit. It is used to power the motor, as well as the logic chips of the system. The h-bridge, Bluetooth module and PIC microcontroller are supplied with a lower voltage, scaled down by voltage regulator to 5 V. The battery is connected to the motor through the h-bridge, which dictates when and how much (in junction with the PWM) voltage reaches the motor leads. A 12 V lead acid battery was used.

2.2.6) 5 V Voltage Regulators (LM7805A): The voltage regulators are used to convert the 12 V voltage from the battery down to the appropriate supply voltages for each component.

We decided to use the voltage regulators after our original voltage divider circuit (consisting of two resistors) was unable to protect our circuit from a voltage spike, which resulted in our original Bluetooth module receiving too much voltage, thus ruining it.

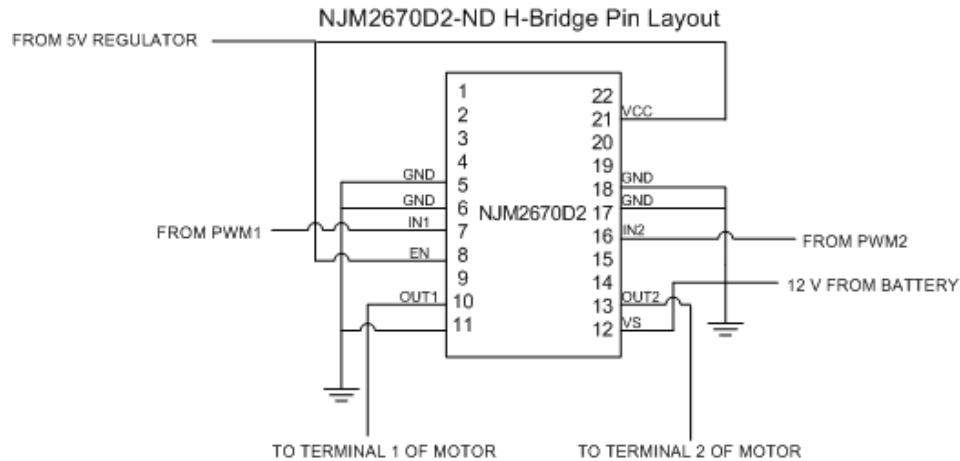
Three separate regulators were used to provide the 5 volts needed to each of the h-bridge, Bluetooth module and microprocessor.

2.2.7) H-Bridge (NJM2670D2-ND): The purpose of the h-bridge is to allow the user to choose the direction the motor rotates in, as well as bring the motor to a complete stop. It consists of four MOSFET transistors operating as switches. Two MOSFETS are connected to each lead of the motor, one connecting the lead to V_{CC} (12 V from the battery) and the other connecting the lead to system ground. When the top switch on one side is on at the same time as the bottom switch on the opposite side, current flows in that direction and causes the motor to turn. When these are turned off and the other two switches turned on, current flows between those two instead and the motor rotates in the opposite direction.

The chip chosen is the NJM2670D2-ND PWM Dual H-Bridge Driver. The two inputs to the H-bridge, IN1 and IN2, receive their signal from the PWMs. The duty cycle of the PWM signal specifies how long the MOSFET switches are active, thus controlling the average voltage that reaches the motor, which in turn controls the speed.

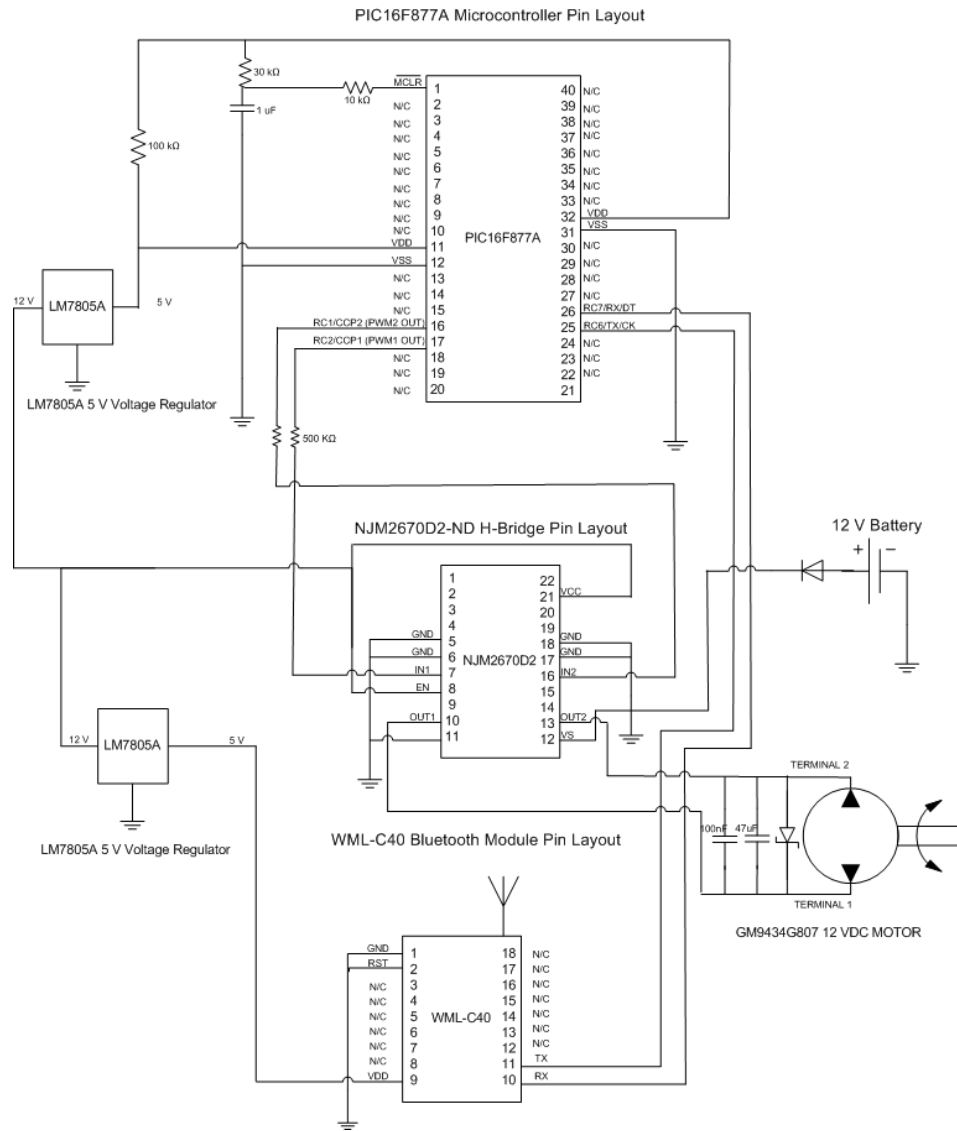
If the motor is to turn in the clockwise direction, then the duty cycle that reaches IN2 is set to 0, and IN1 receives the duty cycle for the given speed, and vice versa.

Before physically implementing the H-bridge, we tested its functionality using a PSPICE simulation. This simulation diagram and resulting waveforms are attached in the Appendix, section 6.3.



2.2.8) 12 Vdc Motor (GM9434G807): This is a Pittman 12 V permanent magnet geared DC motor. It will be powered by the 12 V lead acid battery, from which the voltage level that reaches the motor will be controlled via the control unit, through the h-bridge. It is capable of rotating in both directions, which is controlled with the h-bridge. The motor has a gear ratio of 65.5: 1. In order to isolate the supply voltage, protective capacitors are added in parallel with the motor, as well as a zener diode to regulate the voltage.

Full Schematic



2.3) Performance Requirements

- Maximum input voltage of 12 V DC
- Output voltage varying from 0 – 12 V DC depending on desired speed
- Adjustable speed from 0 to 93.9 RPM on lower geared side
- Adjustable speed from 0 to 6222 RPM on load side
- Maximum rated motor torque of 2.12 N m
- Motor can turn in both directions
- Can handle continuous motor load up to 15 W

- Wireless control via Bluetooth up to 60 feet
- Motor control software runs on any Windows-based PC

III) Design Verification

3.1) Testing Procedures and Results

3.1.1) PC to Bluetooth Adapter Module

In order to test the connection from the GUI and USB Transmitter to the WML-C40 on the motor side, we first connected the RX and TX of the WML-C40 to each other, so that the chip receives a signal from the transmitting PC and sends it back. This is used to verify that both the RX and TX configurations are working properly. Using the application HyperTerminal, ASCII characters are typed with no local echo on, so that if they are not sent back by the Bluetooth module, they will not be displayed on the screen.

By doing this test, we were able to verify that the Bluetooth module was correctly receiving and transmitting the control signal.

Once this was working, we connected the TX of the Bluetooth module to a RS232 level translator, which was then connected to a second PC running HyperTerminal. This was used to represent the PIC. To make sure the signal would be correctly received by the PIC, we ran the GUI, executing all the commands, and observed the speed received on the second PC. We could observe the speed changing in accordance with what was being selected on the GUI, verifying the output was correct.

3.1.2) Control Unit

The PIC microprocessor and PWM combination were tested in several steps. Various serial inputs were sent to the PIC using HyperTerminal and our motor control GUI, and outputs were assigned based on input. The serial data had to be translated from a string variable to an integer, so that it could be numerically evaluated. The integer equivalent of the string was set by the string character's ASCII decimal value, which means we had to translate the ASCII decimal equivalent to its integer value.

Once a speed integer was produced, we ran a test by first turning one output high if the incoming speed was greater than zero, and a second output high if the speed was less than zero. We were able to accomplish this, meaning we could run the motor in both directions at this point, but not at variable speeds.

The onboard PWMs were tested by assigning various duty cycles to the outputs, and observing the resulting waveform with an oscilloscope.

3.1.3) H-Bridge

The H-bridge was tested first by hardwiring one of the two input pins to ground, and the other to 5 volts. Depending on which orientation they were connected in, the H-bridge would either produce 12 V or -12 V to the motor leads. This is how the direction of the motor is determined.

Once this was verified, we were able to test the H-bridge by wiring it to the outputs of the PIC. Using the setup before the PWMS were implemented, we were able to observe the H-bridge outputting 0 V, 12 V and -12 V depending on what was specified

by the user input. After this was functional, we were able to test it with the PWM inputs instead, allowing for variable speed control.

3.1.4) 12 Vdc Motor

The motor was first tested by connecting 12 V and -12 V to the motor leads, to verify that it ran in both directions. Once that was verified, we were able to connect it to the output of the H-bridge, and observe the resulting motor activity for a given user input.

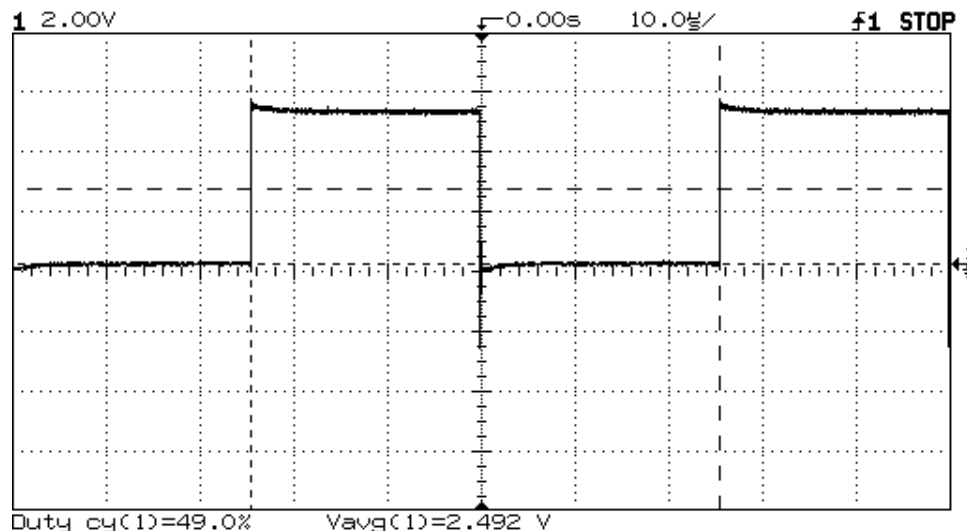
3.2) Thoroughness

Since all of our parts function correctly independently we were able to determine that the product will meet the performance requirements. We knew that if the product did not function correctly, the problem would be due to implementation and connection error and should be relatively easy to identify and resolve.

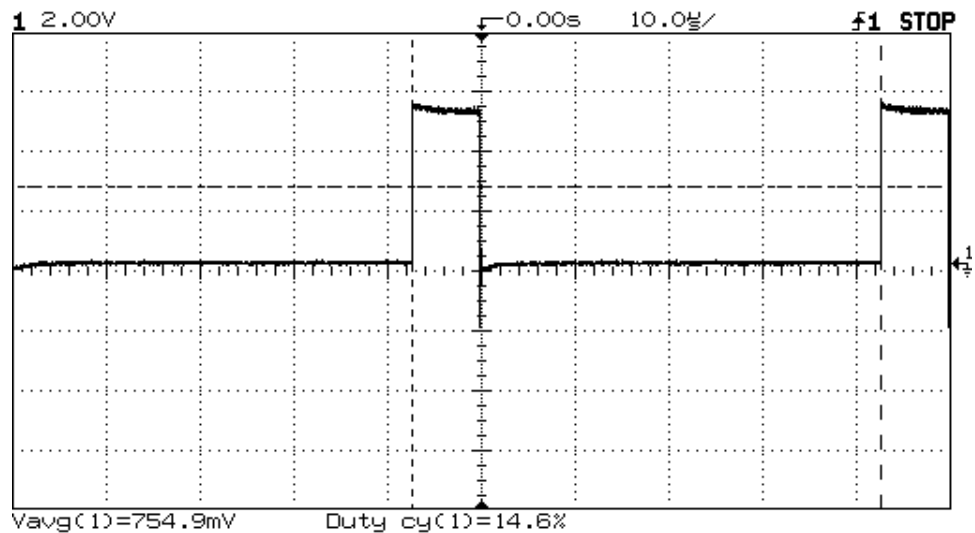
3.3) Tolerance Analysis

Since the DC to DC step down converter was removed from the design, the most important aspect is the PWM that is built into the PIC. This unit is especially important, as it is the one that controls the exact amount of power that the motor receives; thus controlling the functionality of the motor. Making sure that the PWM is delivering the correct duty cycle to the h-bridge ensures that the user will be able to control the exact speed of the motor. This was tested by defining different duty cycles and making sure that these are in fact the signal that reaches the h-bridge, and once that is confirmed, making sure these duty cycles correspond to the proper output voltage. More specifically, a known duty cycle was applied, and an oscilloscope was used to view the waveform from the PWM to the h-bridge to make sure that the correct duty cycle was observed. When that was working properly, the oscilloscope was placed over the motor leads to make sure it was receiving the corresponding correct voltage.

Seen below are two control signals from the PWM output to the H-bridge input. These are 5 volt logic, and have a duty cycle corresponding to the requested speed. These duty cycles were determined to provide the requested speed by the function tests that were described above.

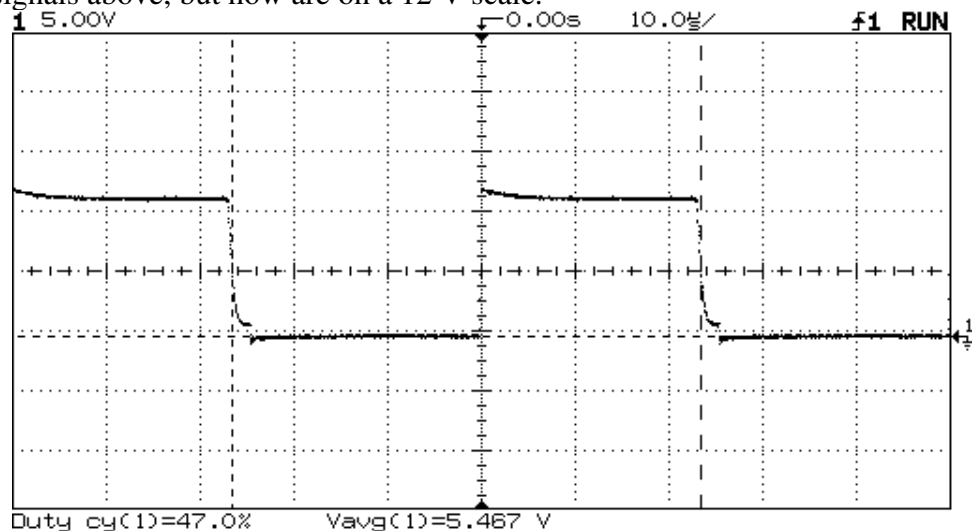


Control Signal from PIC to H-bridge at 48 RPM

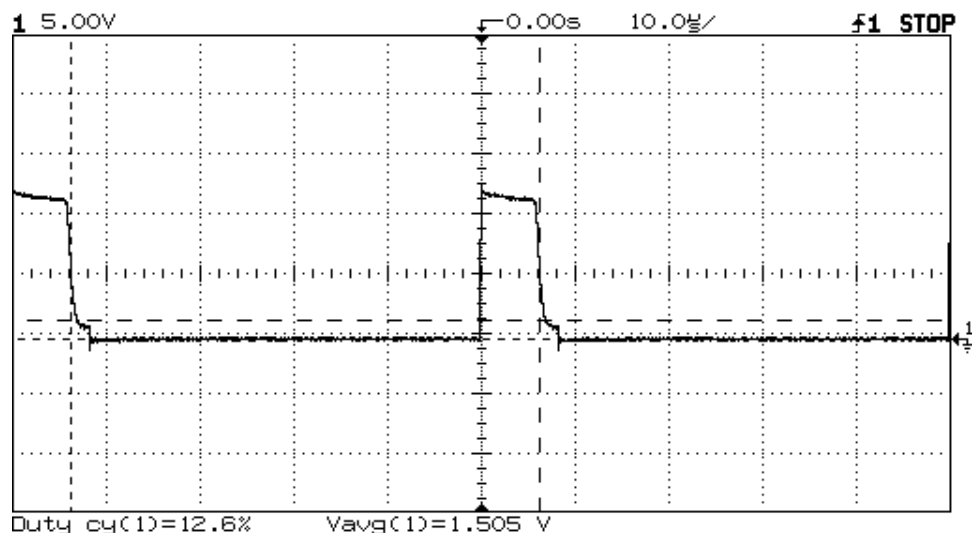


Control Signal from PIC to H-bridge at 5 RPM

To verify that the correct voltage is provided to the motor, oscilloscope plots of the motor voltage are also provided. As can be seen, these have close to the same duty cycle as the control signals above, but now are on a 12 V scale.



Motor Voltage at 48 RPM



Motor Voltage at 5 RPM

IV) Cost Breakdown

4.1) Labor Costs:

The provided formula for labor cost, per person, is:

$$\text{Labor Cost Per Person} = \text{Hourly Salary} * \text{Actual Hours} * 2.5$$

Over the course of the project, there were a total of 394 man hours worked between both team members. The estimated wage for this is \$25 / hour.

So the total labor cost, including both team members, is $394 * 25 * 2.5$

Total Labor Cost: \$24,625

4.2) Part Costs:

Part	Model #	Manufacturer	Price	Quantity	Total Price	Total Price (if 10,000 purchased)
Wireless Receiver for PC	WRL-00150	Sparkfun	\$16.95	1	\$16.95	\$13.56
2.4GHz Duck Antenna	WRL-00145	Sparkfun	\$7.95	1	\$7.95	\$6.36
BlueSMIRF WML-C40 Module	WRL-00158	Mitsumi	\$64.95	1	\$64.95	\$51.96
H-Bridge	NJM2670D2-ND	TI	\$4.83	1	\$4.83	\$3.22
12 V Battery	BP5-12-T2	B&B Battery	\$15.93	1	\$15.93	\$8.50
500 K Ω Resistor	TK20P500RJE	Ohmite Mfg.	\$4.21	2	\$8.42	\$3.78
20 MHz Oscillator	F1100E	Fox	\$2.27	1	\$2.27	\$1.75
5V Regulator	LM7805A	Fairchild	\$0.39	3	\$1.17	\$0.93
PIC	PIC16F877	Microchip	\$10	1	\$10	\$7.83
12 V DC Motor	GM9434G807	Pittman	\$19.99	1	19.99	\$14.95
4.7 M Ω Resistor	CRCW08054M70JNEA	Vishay/Dale	\$0.04	2	\$0.08	\$0.006
1.8 M Ω Resistor	CRCW08051M80FKEA	Vishay/Dale	\$0.04	1	\$0.04	\$0.005
3.3 M Ω Resistor	CRCW08053M30FKEA	Vishay/Dale	\$0.04	1	\$0.04	\$0.005
100 K Ω Resistor	CRCW1210100KFKEA	Vishay/Dale	\$0.11	1	\$0.11	\$0.027
30 K Ω Resistor	CRCW080530K0FKEA	Vishay/Dale	\$0.04	1	\$0.04	\$0.005
10 K Ω	CRCW121010	Vishay/Dale	\$0.11	1	\$0.11	\$0.027

Resistor	K0FKEA					
1 uF Capacitor	2222 021 38108	BC Components	\$0.50	1	\$0.50	\$0.22
100 nF Capacitor	UVR1H0R1MD D	Nichicon	\$0.20	1	\$0.20	\$0.035
47 uF Capacitor	2222 138 35479	BC Components	\$1.47	1	\$1.47	\$0.63
Zener Diode	2EZ12D5D041	MicroSemi	\$1.42	1	\$1.42	\$1.09
Diode	1n4004	General Semiconductor	\$0.05	1	\$0.05	\$0.028

Total Part Cost: \$156.52

Total Cost (Labor + Parts) = \$24,781

Total Part Cost (if 10,000 purchased): \$114.92

Total Cost (Labor + Parts (if 10,000 purchased)) = \$24,739

V) Conclusion

5.1) Accomplishments

With our allotted time this semester, we were able to successfully integrate everything we promised in our design proposal. The user was successfully able to control the motor wirelessly from the GUI running on a laptop.

The user was able to start, stop, accelerate and decelerate the motor by setting the speed (in RPM, referenced to the lower geared side). The motor was able to run 0-95 RPM on the lower geared side, and 0-6222 RPM on the load side. The direction of the motor was also selectable, using the CW and CCW buttons. This functionality all worked as desired. When the motor was running and the user chose to switch directions, the motor slowly ramped down its speed to zero, then switched directions and ramped back up to the desired speed. The same ramping effect was used when Stop was selected, to bring it slowly to 0 RPM. The purpose of this was to protect the motor from sudden voltage and speed changes.

We were able to run the motor at a maximum of 15 W, which was the desired range. The desired torque of 2.12 N-m could not be achieved however, because we were limited by our H-bridge output current of 1.2 A. We were able to achieve a maximum of torque of 1.33 N-m.

5.2) Uncertainties

With the feed forward control approach that was used, we remain uncertain how accurate the speed control would be for an unknown or varying load.

Another uncertainty would be how the circuit would respond if the load exceeded the maximum load of 15 W. We did not observe this because we did not want to exceed the rated currents and destroy our chips; however it is very possible that this could happen in a practical application.

5.3) Ethical Considerations

As our design was not constructed entirely from scratch and used existing products purchased from suppliers, we may need to have their consent before commercially marketing our product.

Other than that, this project does not raise any other ethical issues.

5.4) Future Work

Once we had achieved all the aspects that we had originally promised in our Design Review, we decided to take the next step and incorporate a feedback loop into our circuit. The purpose of this was to confirm that the actual speed achieved by the motor matched the desired speed specified by the user. If not, a correction algorithm would be implemented, to compensate and reduce the error.

The practical applications for such a circuit with speed feedback would be a situation where the load is varying or unknown. One example of this would be an off-road remote control car that encounters various terrains.

With a feed forward loop, an increase in load would not be compensated for, and thus the desired speed would not be achieved. However with feedback, it would be compensated for, and the voltage to the motor could be increased to achieve the desired speed.

We were able to begin implementation of this feedback control, by measuring actual RPMS achieved by the motor using a rotary encoder that we designed. We were then able to send this signal to the microcontroller and read it, but did not have enough time to finish implementing the code. This would be a great place for someone picking the project up to start with.

5.5) Alternatives

An alternative to using the PWM outputs to set the duty cycle of the motor, a buck converter circuit could have been constructed. However, we found that the PWM technique was much more efficient because all the scaling could be done digitally in the PIC code, rather than via analog using a buck converter. This simplifies the circuit greatly.

An H-bridge with a higher current rating could also have been used, in order to allow a higher input current to the motor, thus increasing the maximum torque and power produced.

Lastly, a battery with a higher Amperes / Hour rating could be chosen which would increase the length of time the circuit can be run. This would be very useful for many applications, such as the remote control car.

VI) Appendix

6.1) Current Analysis

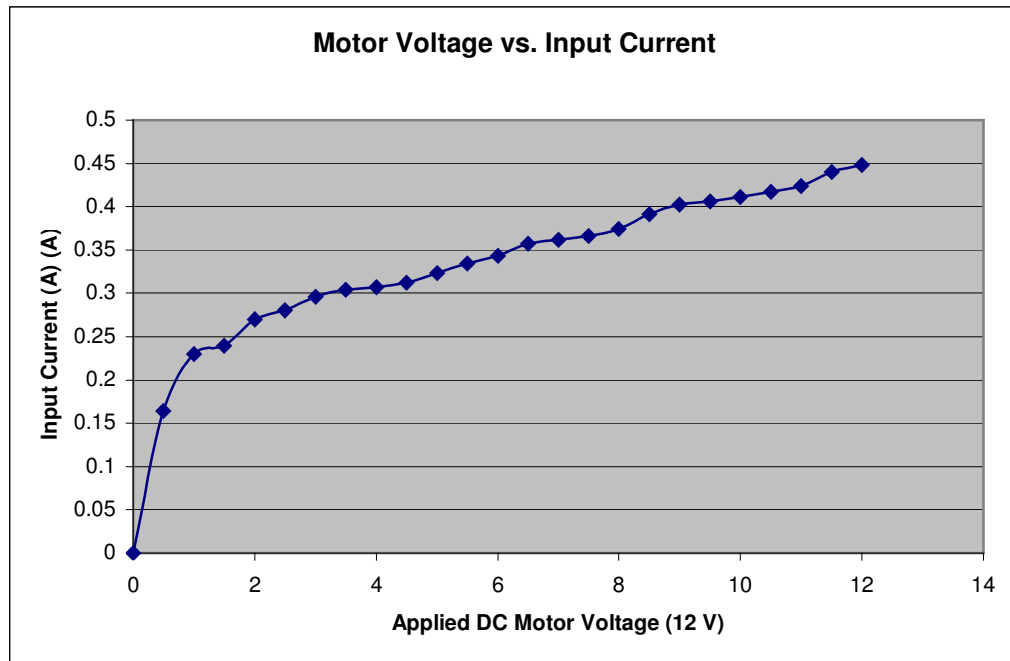
Pin Currents

Measurement	Value
DC Motor No-Load Current (Measured)	0.45 A
DC Motor No-Load Current (Spec Sheet)	0.33 A
Motor Peak Current	14.5 A
Maximum Current Output from H-Bridge	3 A
H-Bridge High Level Logic Input Current	2 to 50 uA, 10 uA typical
H-Bridge Max Output Current	1.2 A
WML-C40 Current Consumption	90 mA
PIC16F877 I/O Pin Current (Source and Sunk)	25 mA

6.2) Input Motor Current vs. Applied Motor Voltage Plots

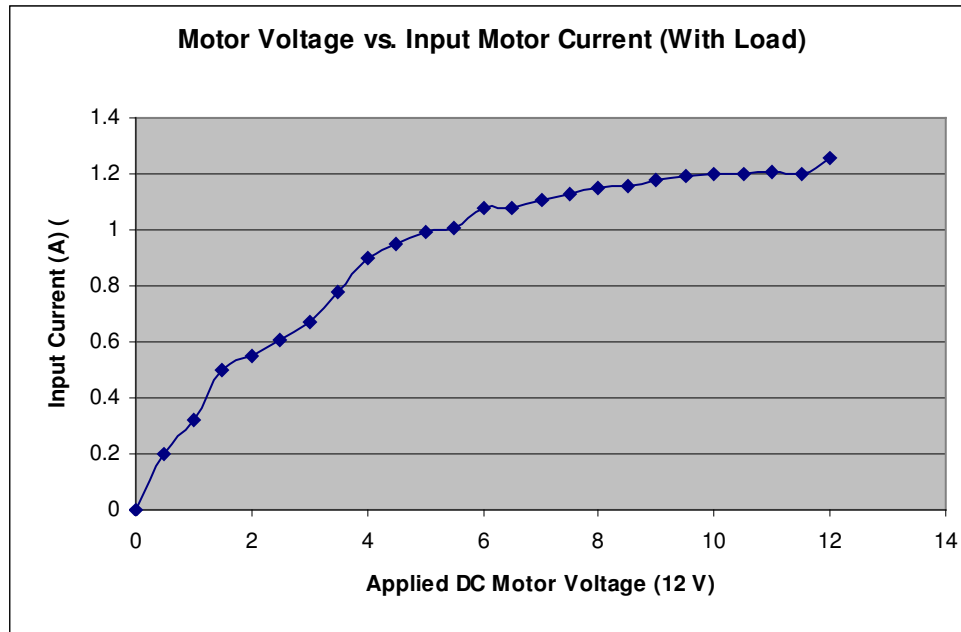
DC Motor Input Current (No-Load)

Motor Voltage	Measured Input Current (A)
0 V	0
0.5 V	0.164
1.0 V	0.230
1.5 V	0.240
2.0 V	0.270
2.5 V	0.280
3.0 V	0.296
3.5 V	0.304
4.0 V	0.307
4.5 V	0.312
5.0 V	0.323
5.5 V	0.334
6.0 V	0.343
6.5 V	0.357
7.0 V	0.362
7.5 V	0.366
8.0 V	0.374
8.5 V	0.391
9.0 V	0.402
9.5 V	0.406
10.0 V	0.411
10.5 V	0.417
11.0 V	0.424
11.5 V	0.440
12.0 V	0.448



DC Motor Input Current (Max Load)

Motor Voltage	Measured Input Current (A)
0 V	0
0.5 V	0.2
1.0 V	0.32
1.5 V	0.5
2.0 V	0.55
2.5 V	0.61
3.0 V	0.67
3.5 V	0.78
4.0 V	0.9
4.5 V	0.95
5.0 V	0.99
5.5 V	1.01
6.0 V	1.08
6.5 V	1.08
7.0 V	1.11
7.5 V	1.13
8.0 V	1.15
8.5 V	1.157
9.0 V	1.18
9.5 V	1.19
10.0 V	1.198
10.5 V	1.2
11.0 V	1.207
11.5 V	1.2
12.0 V	1.26

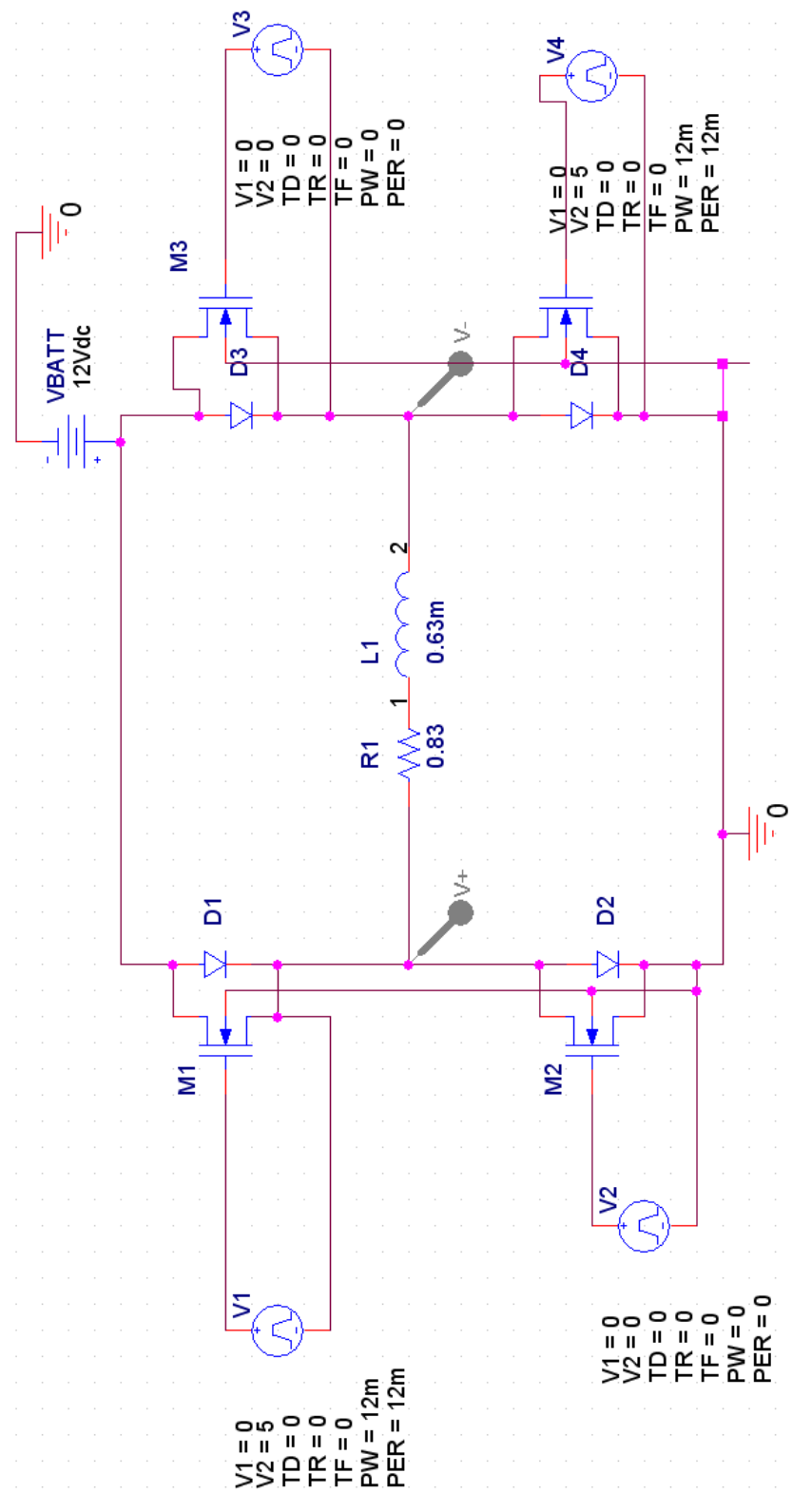


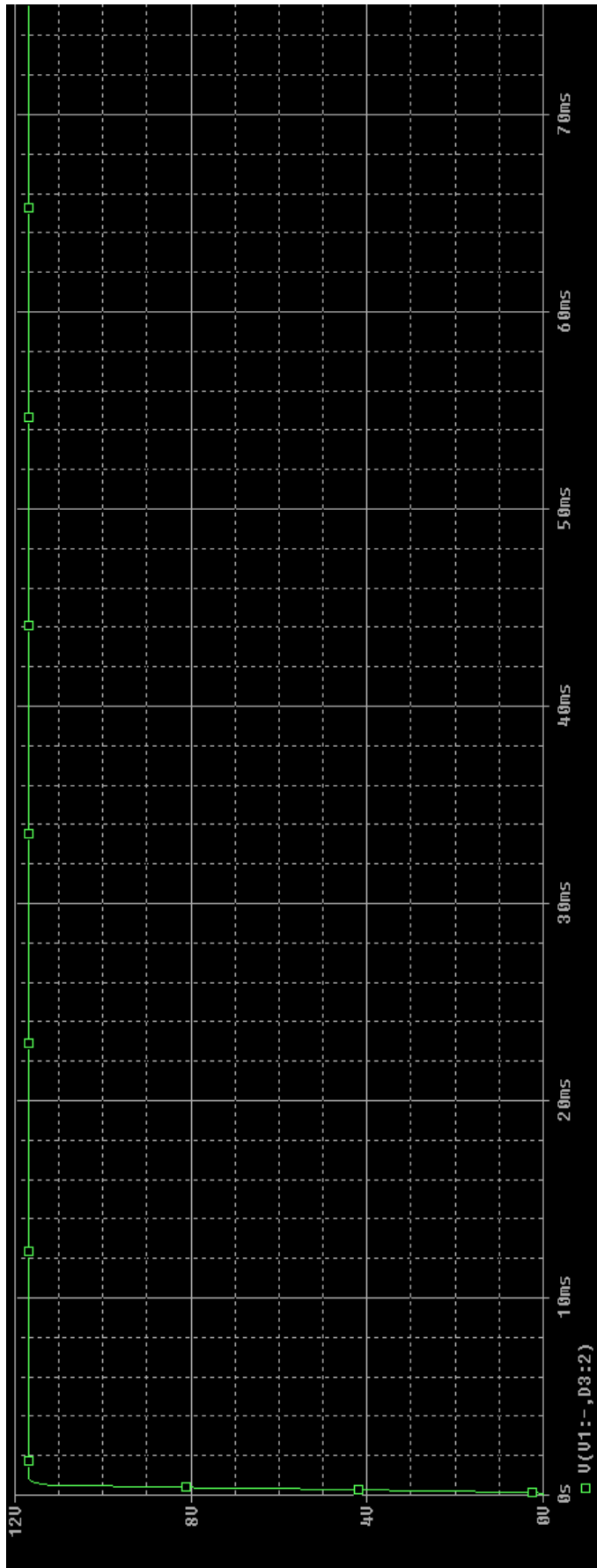
6.3) H-Bridge Simulation

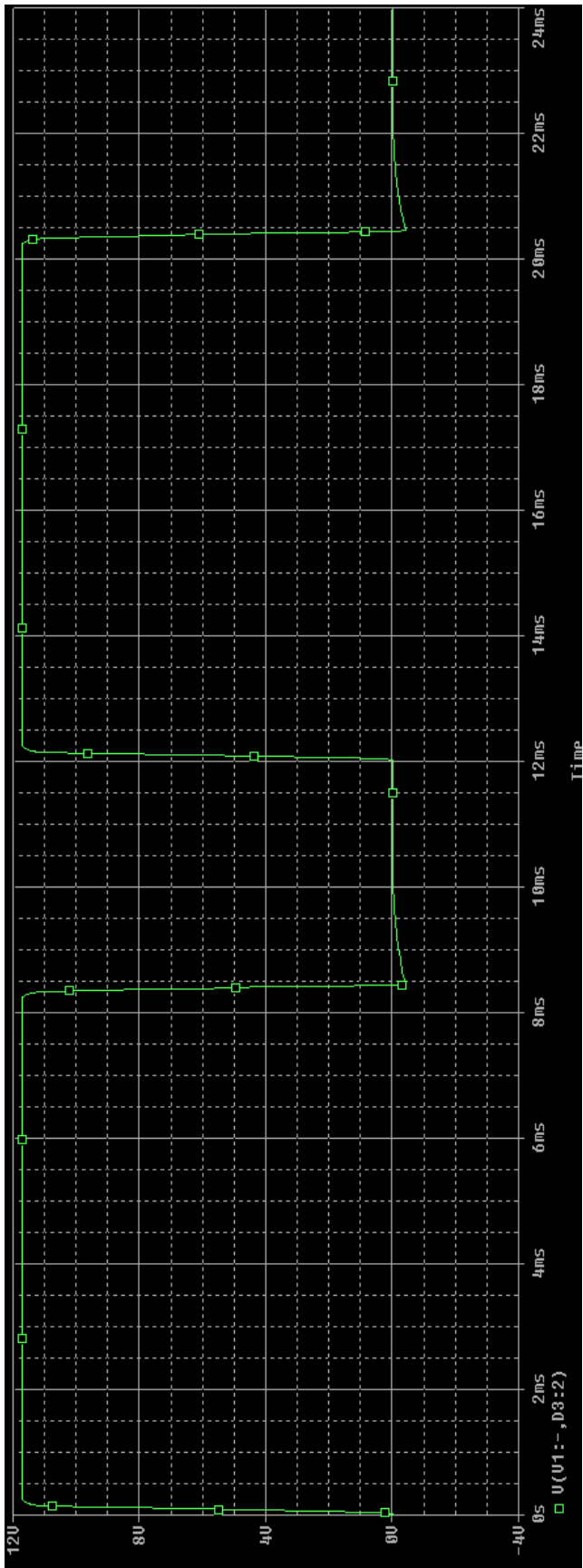
A simulation of the H-bridge using PSPICE is seen on the next page. This gave a verification that the H-bridge chip would respond in the correct manner.

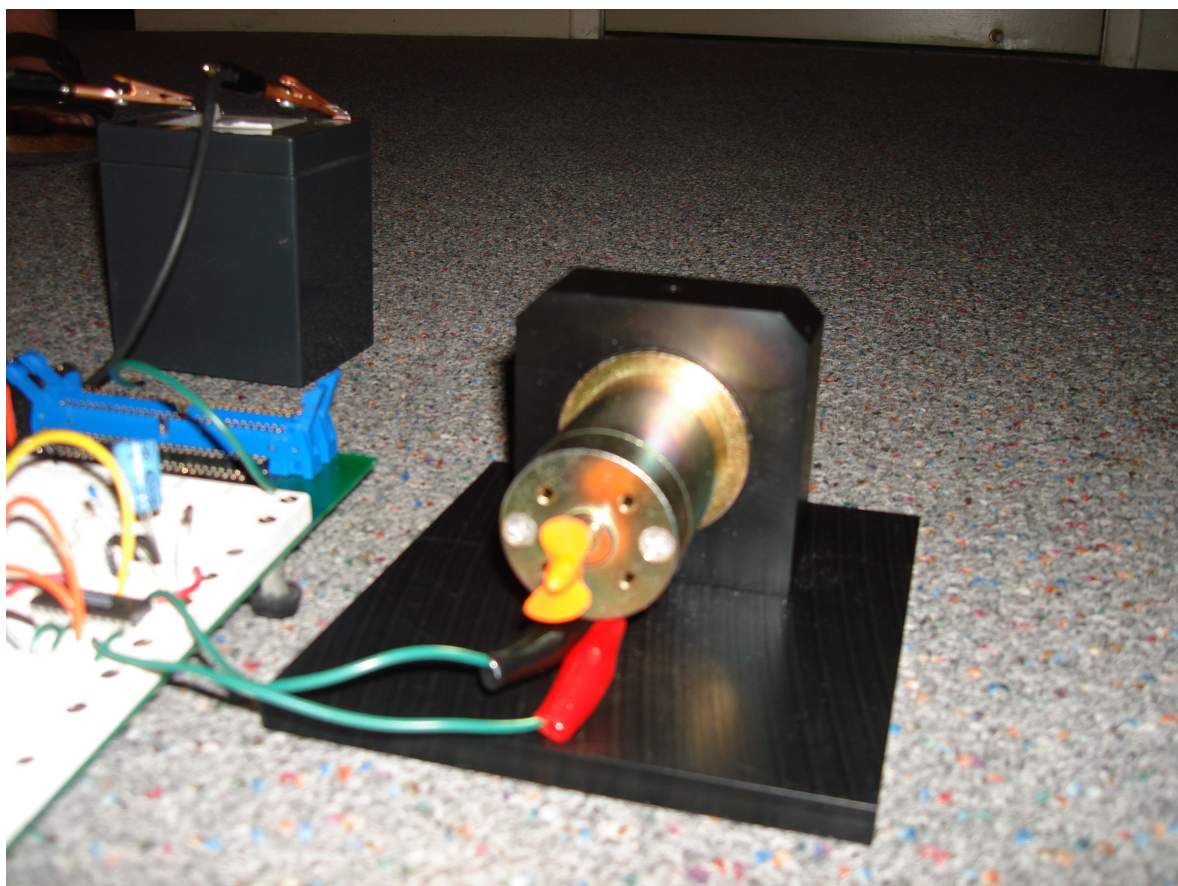
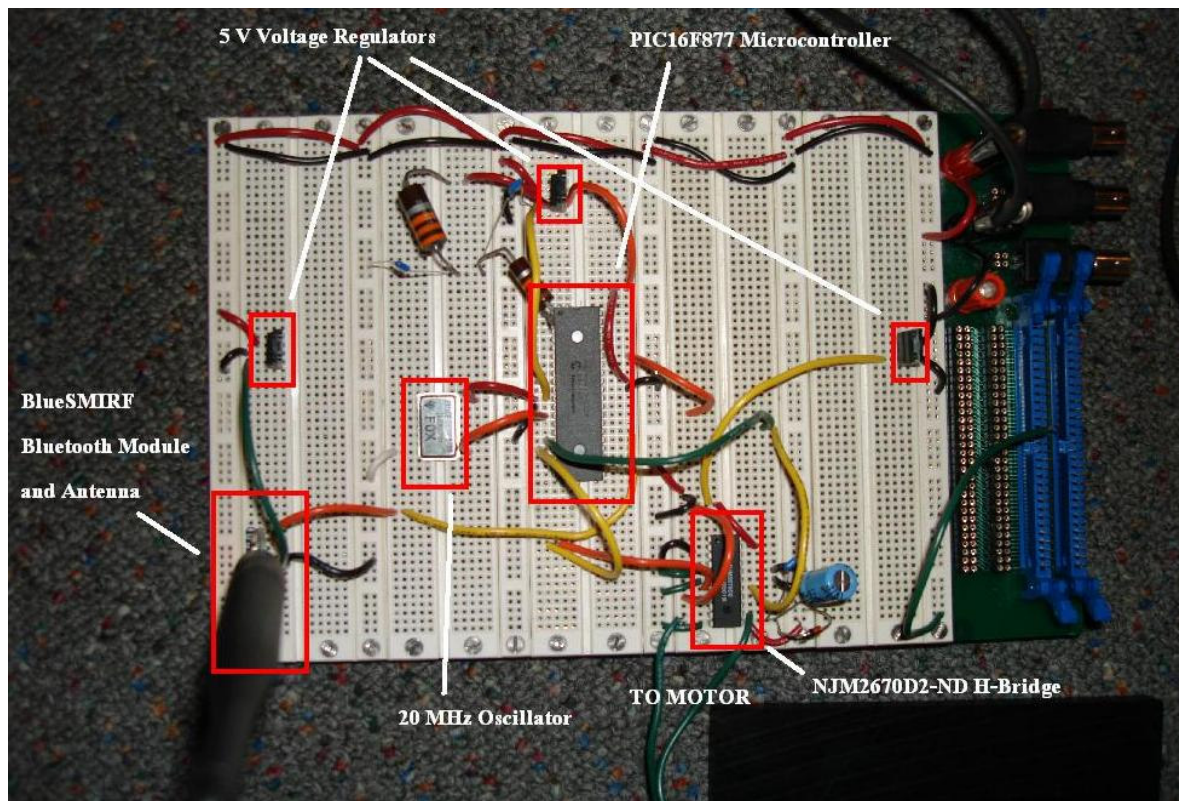
Using pulse voltage sources to model the duty cycle signal from the PWM, the pulse width can be changed to different percentages of the total period of the signal, which in turn changes the average voltage that the motor receives. With a PW of 12 ms and a period of 12 ms, the motor gets the full 12 volts, as seen in the first simulation PSPICE plot. However, if the period remains at 12 ms and the PW is changed to 8 ms (or two thirds), it can be seen that the average of the DC voltage that reaches the motor is $\frac{2}{3}$ of the total 12 V. This is seen in the second plot.

H-Bridge PSPICE Simulation









REFERENCES

- [1] Microchip Technologies, Inc, “PIC16F87X Data Sheet,” [Online Document], 2001, [cited 4 March 2007], Available HTTP: <http://ww1.microchip.com/downloads/en/DeviceDoc/30292c.pdf>
- [2] New Japan Radio Co., Ltd., “Dual H Bridge Driver.” [Online Document], 3 Oct 2003, [cited 25 Feb 2007], Available HTTP: http://semicon.njr.co.jp/njr/hp/fileDownloadMedia.do?_mediaId=3700
- [3] Mitsumi Electric, “Bluetooth Module, WML-C40 Class 1,” [Online Document], [cited 2 Feb 2007], Available HTTP: <http://www.sparkfun.com/datasheets/Wireless/Bluetooth/Bluetooth-SMD-Module.pdf>
- [4] PennEngineering Motion Technologies, Pittman, “LO-COG DC Gearmotors,” [Online Document], 2003, [cited 13 Feb 2007], Available HTTP: http://www.pittmannet.com/pdf/lcg_bulletin.pdf