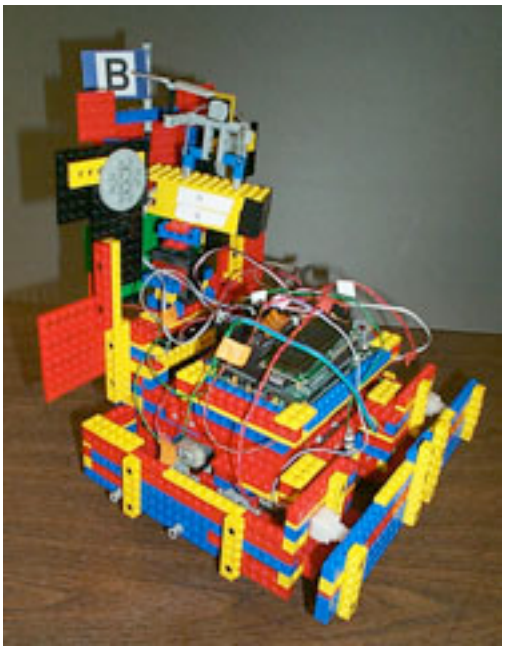
Integrating Bluetooth onto LEGO robots

Final Project Report
April 25, 2003

Prepared for
Prof. Sree N. Sreenath and Prof. Behnam Malakooti



Prepared by

Team Members

Andrew Jones

Mike Krofcheck

Faculty Advisor(s)

Frank Merat
Associate Professor, EECS

Richard Drushel
Senior Research Associate, Biology

Table of Contents

Table of Figures

Table of Figures

Abstract

The “Integrating Bluetooth onto LEGO Robots” project team consisted of Andrew Jones and Mike Krofcheck, was supervised by Professor Frank Merat and Professor Richard Drushel, and sponsored by PCBExpress. The project team setout to build a wireless communication board and corresponding driver software utilizing Bluetooth technology that would be interfaced with the LEGO robot microcontroller boards used in EECS 375. The wireless communication capabilities provided by Bluetooth will allow robots designed by students to communicate with each other and with a computer to download program updates. It will also eliminate the use of serial cables currently used to program the robots.

Bluetooth was chosen for wireless communication due to its lower power consumption, operation in the unlicensed ISM band, ability to interface with serial devices, and low cost in high quantity. Other competing communication protocols like WiFi (802.11b) and HomeRF do not provide the integrated all-in-one solution, low power consumption, or cost savings of Bluetooth. The communication capabilities provided by Bluetooth will allow robots designed by students to communicate with each other and with a computer wirelessly.

The project team has obtained an integrated Bluetooth Module (BTM), designed a high frequency Integrated Inverted F Antenna (IIFA), designed and fabricated a Printed Circuit Board to support a Bluetooth module with serial communication, tested and verified operation of the PCB, and implemented a Bluetooth communication driver.

Executive Summary

The project team of Andrew Jones and Mike Krofcheck setout to build a wireless communication board and corresponding driver software utilizing Bluetooth technology that would be interfaced with the LEGO robot microcontroller boards used in EECS 375. Bluetooth was chosen for wireless communication due to its lower power consumption, operation in the unlicensed ISM band, ability to interface with serial devices, and low cost in high quantity. Other competing communication protocols like WiFi (802.11b) and HomeRF do not provide the integrated all-in-one solution, low power consumption, or cost savings of Bluetooth. The communication capabilities provided by Bluetooth will allow robots designed by students to communicate with each other and with a computer wirelessly.

An integrated Bluetooth Module (BTM) was utilized as opposed to one of the many available single chip Bluetooth modules due to the difficulty in acquiring single chip modules and the inclusion of the Bluetooth Stack software on local Flash memory. The Bluetooth Stack controls the baseband controller and radio transmitter/receiver chips, making it vital to any Bluetooth implementation. A single chip module typically does not include the Bluetooth Stack, significantly increasing development time.

To date, the project team has obtained an integrated BTM, designed a high frequency Integrated Inverted F Antenna (IIFA), designed and fabricated a Bluetooth communication board with serial communication links and IIFA, tested and verified operation of the Bluetooth communication board, and implemented a Bluetooth communication driver.

Introduction

EECS 375, commonly referred to as the "LEGO lab class", has become a popular course offering in the EECS department. In fact, the course has become so popular, that the course instructors, Dr. Richard F. Drushel, Dr. Hillel J. Chiel, and Dr. Randall D. Beer, have to keep a waiting list for the course.

Modeled after the original LEGO lab course, MIT 6.270, designed by Fred Martin, Randy Sargent, Anne Wright, P.K. Oberoi, *et al.* at Massachusetts Institute of Technology, EECS 375 employees a C-programmable 68HC11 microcontroller board which provides significantly more control and functionality than the LEGO Mindstorms set utilized in ENGR 131. After being used by 411 students over the last 15 semesters of the class's existence, the general functional limitations of the boards have resulted in several "copy cat" designs from semester to semester. Due to this apparent need, the student group consisting of Andrew Jones and Mike Krofcheck, have decided to build a Bluetooth communication board and driver software that could be interfaced with the microcontroller board used in EECS 375. The wireless communication capabilities provided by Bluetooth will allow robots designed by students to communicate with each other and with a computer to download program updates. It will also eliminate the use of serial cables currently used to program the robots.

Bluetooth, also referred to as IEEE 802.15, is a short-range wireless communication protocol operating in the Industrial, Science, and Medical (ISM) band at 2.45 GHz. Bluetooth establishes Wireless Personal Area Networks (WPANs) with devices in the protocols 30 ft. communication range and allows for data to be transmitted at up to 1 Mbps.

Due to the amount of work involved in such an undertaking, the project was originally broken into two semesters. The first part, conducted during the Fall '02 semester, involved investigating the wireless communication options available, deciding on a standard wireless protocol (or an original protocol), and designing a module to demonstrate the wireless capabilities of the standard chosen (Bluetooth). Additional work was conducted during the recess between the Fall '02 and Spring '03 semester to locate and acquire the integrated Bluetooth Module (BTM) necessary to begin fabrication of a Bluetooth communication board. At the conclusion of the Spring '03 semester, the project group planned to provide the following deliverables:

- A Printed Circuit Board (PCB) with Bluetooth communication capabilities that accepts commands via a serial cable
- Bluetooth driver software for microcontroller to control Bluetooth communication capabilities
- Implementation plans to integrate Bluetooth PCB with the microcontroller board used in EECS 375

No prior work on integrating Bluetooth communications onto the microcontroller board used in EECS 375 has taken place. However, a previous senior project group consisting of Bruce E. Brown and Dan W. Baker made an attempt at implementing a proprietary wireless protocol for the robots used in EECS 375. Professor Richard Drushel, instructor for EECS 375, indicated that restrictions of the design included the complicated circuit needed to implement the protocol, power requirements, and the inability to quickly start communication between multiple robots. Another senior project group, lead by Phillip Fultz and Ryan Hollinger, worked on "Medical Astronaut Monitor

System (MAMS)", exploring the use of Bluetooth technology for wireless medical monitoring equipment.

The senior project team consists of two members: Andrew Jones and Mike Krofcheck. Both Andrew and Mike will partake in the hardware and software roles of the project, although Andrew will have a larger percentage contribution to the software required for the project. Andrew Jones has been selected to act as the team leader.

Methodology

The success of the "Integrating Bluetooth onto LEGO robots" project hinged on building and successfully operating a Bluetooth module Printed Circuit Board (PCB) and Integrated Inverted F Antenna (IIFA) that would later be integrated with the LEGO robot boards used in EECS 375. This was the largest portion of the project and required a great deal of work by both team members. The second major portion of the project was the development of a Bluetooth driver using the Host Controller Interface (HCI) supported by the Bluetooth protocol. The HCI allows for a serial link to be established between the host (computer or microcontroller) and Bluetooth module, providing a method of controlling the Bluetooth module's operation. Finally, after a driver had been developed, the Bluetooth module was to be integrated with the existing microcontroller board used in EECS 375. However, due to the time required to successfully design, build, and verify the operation of the Bluetooth module PCB, the final phase was not completed.

During work on the "Integrating Bluetooth onto LEGO Robots" project, it became apparent that the original methodology outlined in the project proposal did not reflect a realistic perspective of the complexity and difficulty in creating a custom Bluetooth

module PCB, writing and testing a Bluetooth driver, and interfacing the Bluetooth module with a host microcontroller. As a result, the original methodology outline was revised to reflect a realistic level of achievement. Due to the level of changes, the original methodology section has been placed in Appendix A.

1. Research and obtain necessary project components
    1.1. Identify and purchase readily available Bluetooth module
    1.2. Purchase microcontroller development environment
2. Build Bluetooth module Printed Circuit Board (PCB)
    2.1. Determine hardware design requirements for Bluetooth controller (i.e high frequency concerns and considerations)
    2.2. Research high frequency antenna design and options (integrated antenna vs. external antenna)
    2.3. Research Printed Circuit Board (PCB) fabrication companies available
    2.4. Research existing high frequency PCB layout designs
    2.5. Design circuit supporting Bluetooth module
    2.6. Test circuit in PSpice
    2.7. Layout PCB for Bluetooth module in CAD software
    2.8. Consult with project advisor and "high frequency PCB expert" in EECS Department
    2.9. Fabricate PCB
3. Verify operation of PCB (see Appendix B for hardware verification procedure)
4. Generate Bluetooth driver

4.1. Install and setup microcontroller development environment

4.1.1. Install Open Source GNU package for 68HC12

4.1.2. Purchase POD for Background Debug Mode use

4.2. Read trade articles related to Hardware Control Interface (HCI) microcontroller programming

4.3. Obtain open source version of Bluetooth Stack

4.4. Modify open source Bluetooth Stack as needed for microcontroller environment

4.5. Write any additional HCI commands

5. Integrate Bluetooth module PCB with current microcontroller board

5.1. Remove existing serial connection on microcontroller board utilized in EECS 375

5.2. Connect Bluetooth PCB with serial connection on microcontroller board

5.3. Test Bluetooth driver software operation with Bluetooth PCB

The major changes between the revised methodology (above) and the original methodology are as follows:

- Section 2 of the original methodology report, "Integrate Bluetooth onto microcontroller development board to test optimal integration methods" was modified to "Build Bluetooth module Printed Circuit Board (PCB)" and many more intermediate steps were added that the senior project group had not earlier anticipated. In addition, the software development portion of section 2 was rolled into a separate section, "Generate Bluetooth communication protocol" to better reflect the magnitude and time needed for this step.

- All references to redesigning the EECS 375 microcontroller and building software for the host computer were removed from the original methodology. The original plans to accomplish these steps were overly optimistic and somewhat naïve. Bluetooth development resources are currently very limited. The Bluetooth module used by the senior project group alone was only 3 pages long.
- Generation of Bluetooth driver software for microcontroller board modified to reflect change in position. Creating the Bluetooth software needed for the project would be a senior project in and of itself. The senior project group decided to work with existing open source code and modify as needed.

The following is a detailed description of the senior project team's implementation of the revised methodology.

Bluetooth module Printed Circuit Board (PCB)

Prior to the Spring '03 semester, extensive research was conducted by the senior project team to identify a viable integrated Bluetooth Module (BTM) that would provide the necessary Bluetooth chipset, Bluetooth Stack, and serial interface. A BTM from Windigo Systems, the BTM02C2XX-R, was identified and found for sale at a United Kingdom distributor, MangoCommerce, for about $90. This method of obtaining the necessary Bluetooth development tools was much more cost effective than some of the pre-made developer hardware; Teleca sells a Bluetooth Application tool kit using the Ericsson Bluetooth chipset, but at a cost in excess of $2500. By using the Windigo

Systems BTM, the senior project group was also able to eliminate the complexity and restrictions of using a pre-maid host (i.e. PocketPC, Palm Pilot, etc…).

The Windigo Systems BTM02C2XX-R utilizes a BlueCore2 family of Bluetooth Integrated Circuits (IC's) from Cambridge Silicon Radio (CSR), a primary developer of single chip Bluetooth modules. The Windigo BTM integrates the CSR BlueCore2 –External single chip Bluetooth system with 8 Mb of external flash memory, a pre-qualified Bluetooth stack compatible with the Bluetooth v1.1 specifications, and the Universal Asynchronous Receiver/Transmitter (UART) interfaces needed to connect the BTM to other peripherals or a host controller (i.e. computer or microcontroller). However, the BTM02C2XX-R modules lack a built-in antenna, needed to transmit the 2.45 GHz RF signal (see Figure 1).

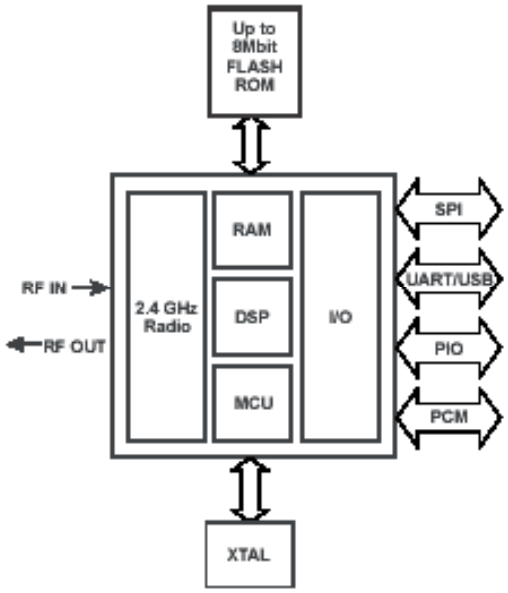


Figure 1: Block diagram of Windigo BTM

To use the BTM from Windigo Systems, a Printed Circuit Board (PCB) was needed that provided the necessary regulated 3.3 volts input, decoupling capacitors, high

frequency antenna, reset circuitry, and UART pin connectors. High frequency concerns necessitated research into proper antenna and PCB design practices.

Due to the high frequency nature of Bluetooth, a simple bread boarded circuit would not be adequate. With the Bluetooth signal at a frequency of about 2.45 GHz, the wavelength, λ, is about 122mm ($\lambda f = C$, where C is the speed of light, 299,792,458 m/s and f is the frequency) and its quarter wavelength, $\lambda_{1/4}$, is about 30mm. This poses great problems if the signal is to be used on a breadboard due to the fact the signal will have significantly different wavelength values at different points along the circuit as a result of the large circuitry size. It creates a domino effect of sorts. This varying wavelengths leads further to undesirable capacitances in many areas. The capacitances will cause cross-talking between the different wires causing catastrophic signal distortion effects on the desired signals. In addition, all wires on the breadboard will operate as antennas picking up stray signals from anything nearby. This adds an enormous amount of noise into the Bluetooth module making signal deciphering impossible. Due to these effects and concerns, a PCB needed to be designed and fabricated for the BTM.

The PCB for the BTM needed to include all of the circuitry necessary to enable full functionality of the BTM. This included a power supply circuit, reset circuitry, an RF antenna, and output pins for the UART signals for the host interface. The power supply and reset circuitry are fairly simple as shown here in Figure 2. The BTM is powered from a 9 volt battery so it will be mobile for testing and easily integrated into the LEGO robot microcontroller board at a future date. The battery runs into a UA78M33C voltage regulator which takes the 9 volt input voltage and converts it into the 3.3 volt output that is required by the BTM for operation. Decoupling capacitors are added (C1, C2, and C3)

to minimize the noise in the power line. This is done by using the capacitors to provide a low impedance path between the BTM's power line and the ground plane on the PCB. This removes any unwanted RF power that is generated by the operation of the module caused by rapid changes in the current draw of the BTM. Without these decoupling capacitors there will be crosstalk and excessive noise on the power line. C3 is located as close to the BTM as possible on the PCB layout to provide the most effective 'filtering'. A 100 Ω resistor (R1) was also added to act as a simple frequency filtering device.

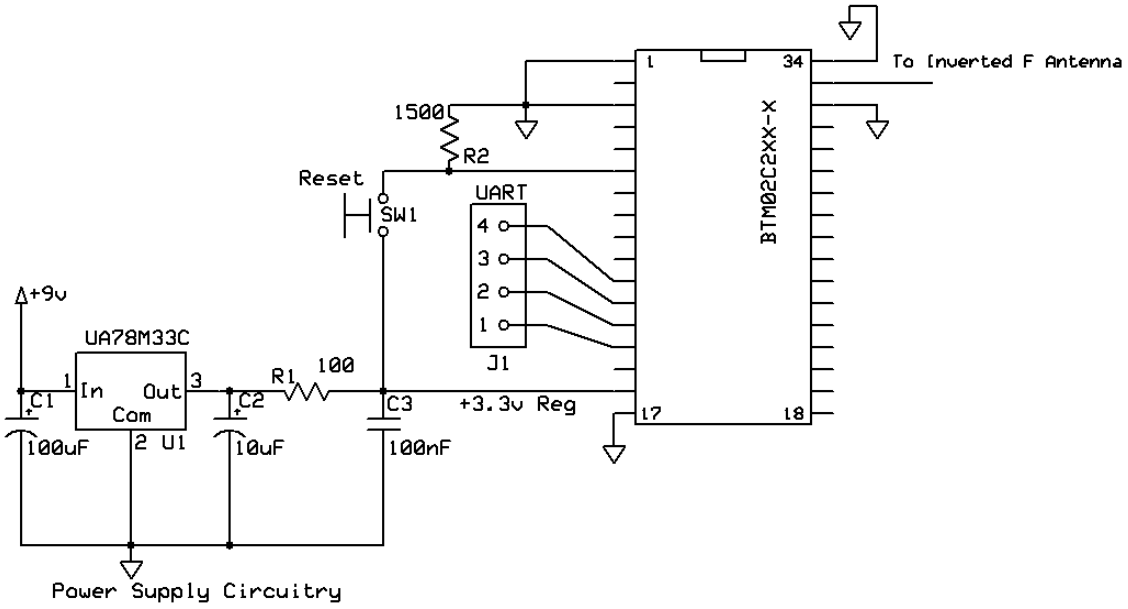


Figure 2: Schematic layout of BTM Printed Circuit Board

The reset circuitry consists of a single pole momentary switch (SW1) and a 1.5 kΩ resistor (R2). This is due to the fact that the project team was unable to acquire a double pole momentary switch. When the switch is not depressed, the BTM's reset pin is tied to the ground through R2. When the switch is depressed, the reset pin is tied to the

supply voltage for the chip causing the chip to reset, and R2 acts to block the power supply from being shorted to the ground.

Extensive research was conducted to determine the type of antenna that would be needed to transmit the 2.45 GHz signal from the module. After some preliminary research, the field was narrowed down to two antennas; a standard dipole antenna and an integrated inverted-F antenna (IIFA). Table 1 shows some of the key characteristics of each antenna design.

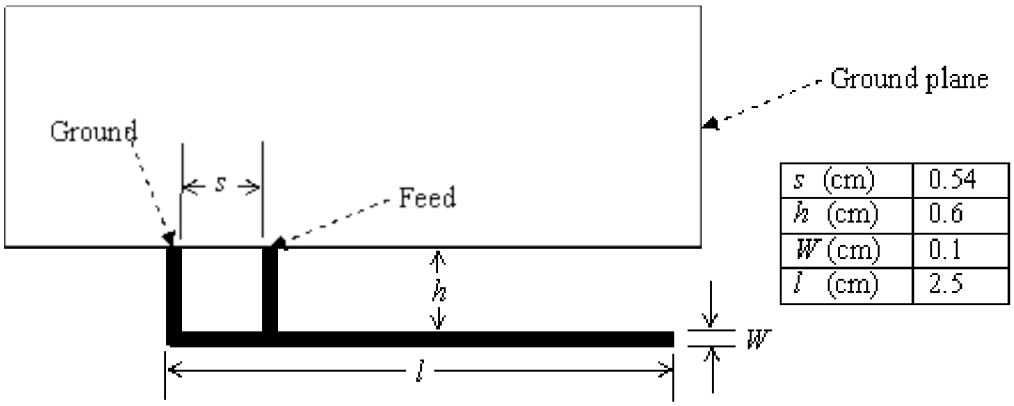
	Dipole Antenna	Integrated Inverted-F Antenna
Size	Large	Tiny
Design	Very Simple	Complex/Difficult
Ease of Addition to PCB	Simple	Integrated into PCB
Practical in End Result	No	Yes
Range	Varies with Design	Varies with Design
Materials	Coax Cable	Traces on PCB, Ground Plane

Table 1: Comparison Dipole Antenna and Integrated Inverted F Antenna

Using the dipole antenna would be by far the simplest approach to the solution, but in the long run it is not the most practical; the only step for using a dipole antenna is to solder the coax cable so it is connected to the RF output of the BTM. However, the antenna for the projects design needs to be as small and as unobtrusive as possible. If the LEGO robots were to have an antenna sticking out of one of their boards, it would cause many design problems for students as they would have to build around the antenna.

Designing the IIFA was considerable work, as many variables must be considered to obtain the correct antenna dimensions. Incorrect dimensions would have a detrimental effect on the antenna's ability to transmits and receives efficiently. The dielectric of the PCB board needed to be taken into consideration and this generally varies from PCB to

PCB. A ground plane is needed to complete the 'antenna loop,' which makes the PCB design thicker, as another layer is needed in the design. Despite the work, the IIFA provided the best design choice for the senior project team, as the antenna would be designed in the PCB, removing the need for external components and saving valuable PCB space. In addition, the IIFA cuts down on material costs since no additional components are needed. However, as a failsafe, two through holes were created; one in the ground plane and another in the trace connecting the RF output to the antenna. This way if the IIFA does not work, it may be excluded from the circuit and a dipole antenna can be attached without difficulty. The IIFA that the project team went with is shown in Figure 3.



s (cm)	0.54
h (cm)	0.6
W (cm)	0.1
l (cm)	2.5

Figure 3: Integrated Inverted F Antenna design

To understand how the signal is physically transmitted and received over the length of a regular antenna and an antenna with a ground plane, refer to Figure 4 below. A quarter wave antenna and appropriate ground plane is comparable to a half wave antenna; the ground plane forms the other quarter wave of the signal completing the antenna. The ground plane serves this purpose in the IIFA.

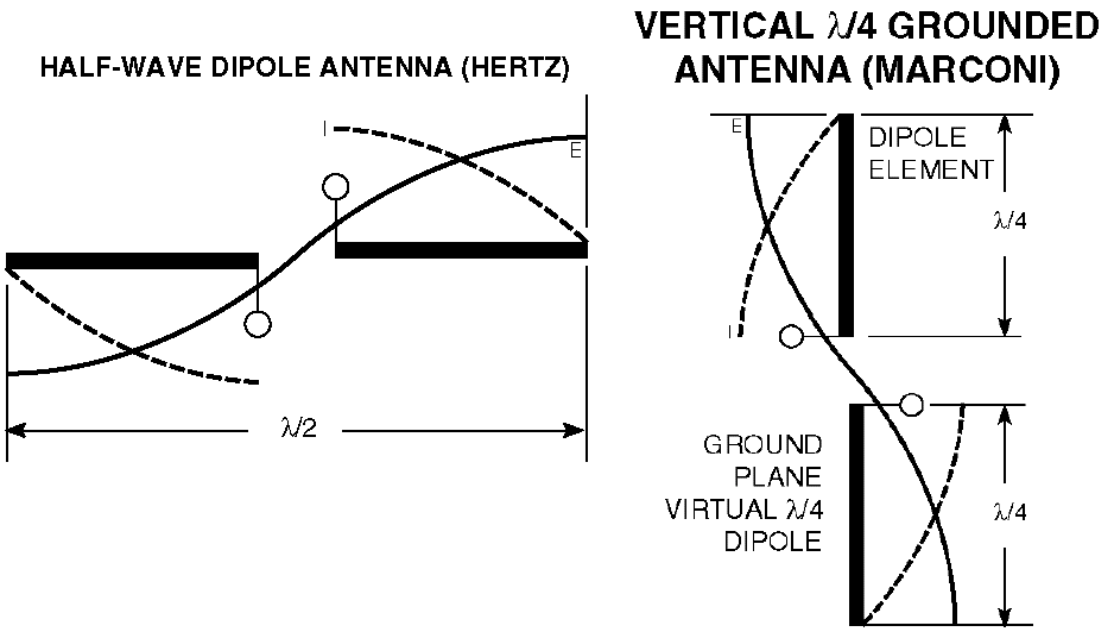


Figure 4: Signal transmission over grounded and non-grounded antenna

The transmission line on the BTM has an impedance of 50 Ω. In order to achieve the maximum transfer of energy from the transmission line to the antenna, the transmission line and antenna need to be matched. This means they both need to have an impedance of 50 Ω at 2.45 GHz. To achieve this, the basic IIFA design is modified and then tested, changing the s, and h values. The length of l is varied to compensate for the changing variable to keep the resonate frequency of the antenna at 2.45 GHz.

Due to the lack of equipment at the project team's disposal, the project team was unable to complete this process. The process of fabricating all the different antenna designs and testing them would have taken longer than the entire semester, not to mention that the equipment for testing the antennas was not available for project use. The project team found many different IIFA designs while researching antennas and selected

the best one for this project. The IIFA selected was designed by Mohammed Rana Basheer, a graduate student from the University of Missouri-Rolla[i]

The following graphs show how the antenna properties changes as different lengths are modified in an attempt to match the transmission line and antenna impedance at 50 Ω.

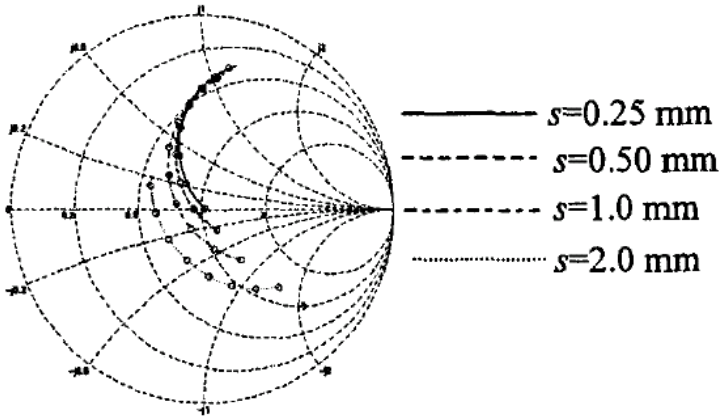


Figure 5: Antenna characteristic diagram as function of value s[ii]

Figure 5 shows the input impedance of the IIFA as a function of the feed to ground distance, s. As the feed to ground distance varies, both the impedance and the resonate frequency of the antenna changes. The dots on each line are the measurements taken at 40 MHz intervals from 2300 MHz to 2600 MHz. As s increases the impedance decreases and the antenna is shortened electrically. In order to keep the antenna effective and better match the impedance, the length of the IIFA must be adjusted to compensate for the shift in resonate frequency. This can be seen in Figure 6 below.

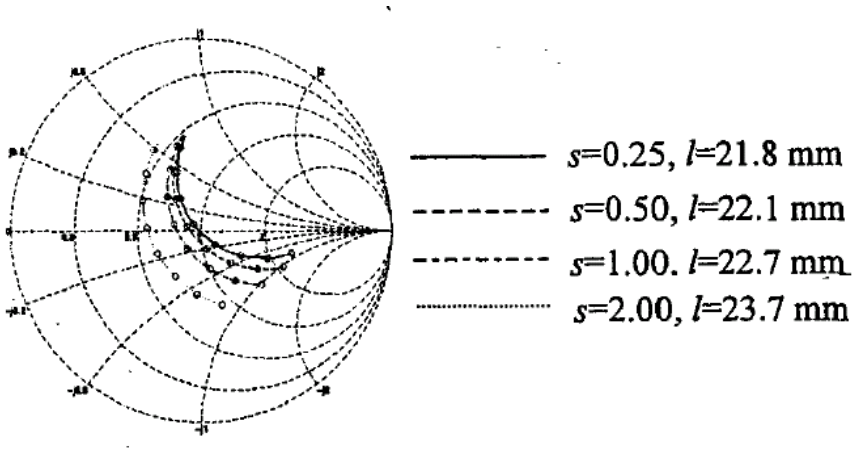


Figure 6: Antenna characteristic diagram as function of value s and l[iii]

In Figure 7, as the h distance increases, the input impedance increases. Therefore the smaller h is, the smaller s has to be and the smaller the antenna is overall.

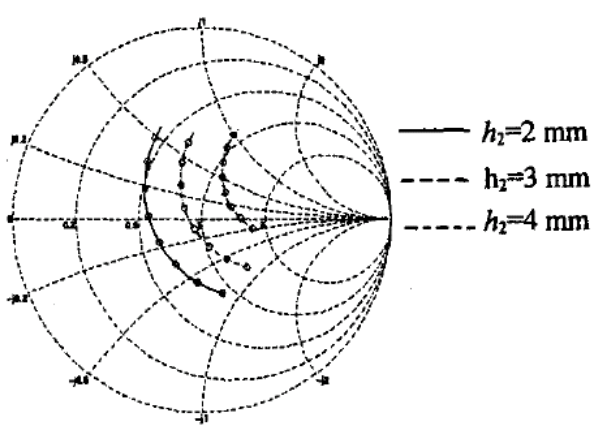


Figure 7: Antenna characteristic diagram as function of value h[iv]

The PCB was designed in Osmond (version 1.0b5), a free CAD program for Mac OS X. Nearly half a dozen PCB programs from companies like PCBExpress and ExpressPCB were tested, but Osmond was found to be the simplest layout software while also offering the most versatility and control over the PCB design. In addition, Osmond outputs layout files to the popular Gerber file format used by nearly all PCB Fabrication

companies. Due to the high frequency nature of the Bluetooth module, impedance constraints of the IIFA, and limitations of the PCB layout software several layout considerations/modifications were made:

- Care had to be taken to ensure that no power, ground or signal traces went beneath the antenna design. A power line beneath the antenna would disrupt the antennas functionality by inducing a charge on the antenna.
- The antenna was placed at the far end of the PCB edge to avoid interference with the other components
- To properly connect the antenna to the ground plane, a small through hole was made at the end of the ground branch and a 'star' pattern was used to connect it to the ground plane. This was the only way Osmond would let a connection from a trace to a ground place.
- Unlike initial thoughts, no special trace designs are needed to connect the RF output to the antenna.

Osmond provides a limited trace verification function that identifies stranded traces that are not connected between two points. This feature was used before exporting the layout to Gerber file format.

The completed board layout was reviewed by Prof. Frank Merat, the senior project team's advisor, and Robert Leskovec, a high frequency PCB layout specialist in the CWRU EECS department. The PCB layout received extremely high praise from Robert Leskovec, and was met with equal praise from Prof. Merat. Figure 8 below shows the completed PCB layout produced by Osmond.

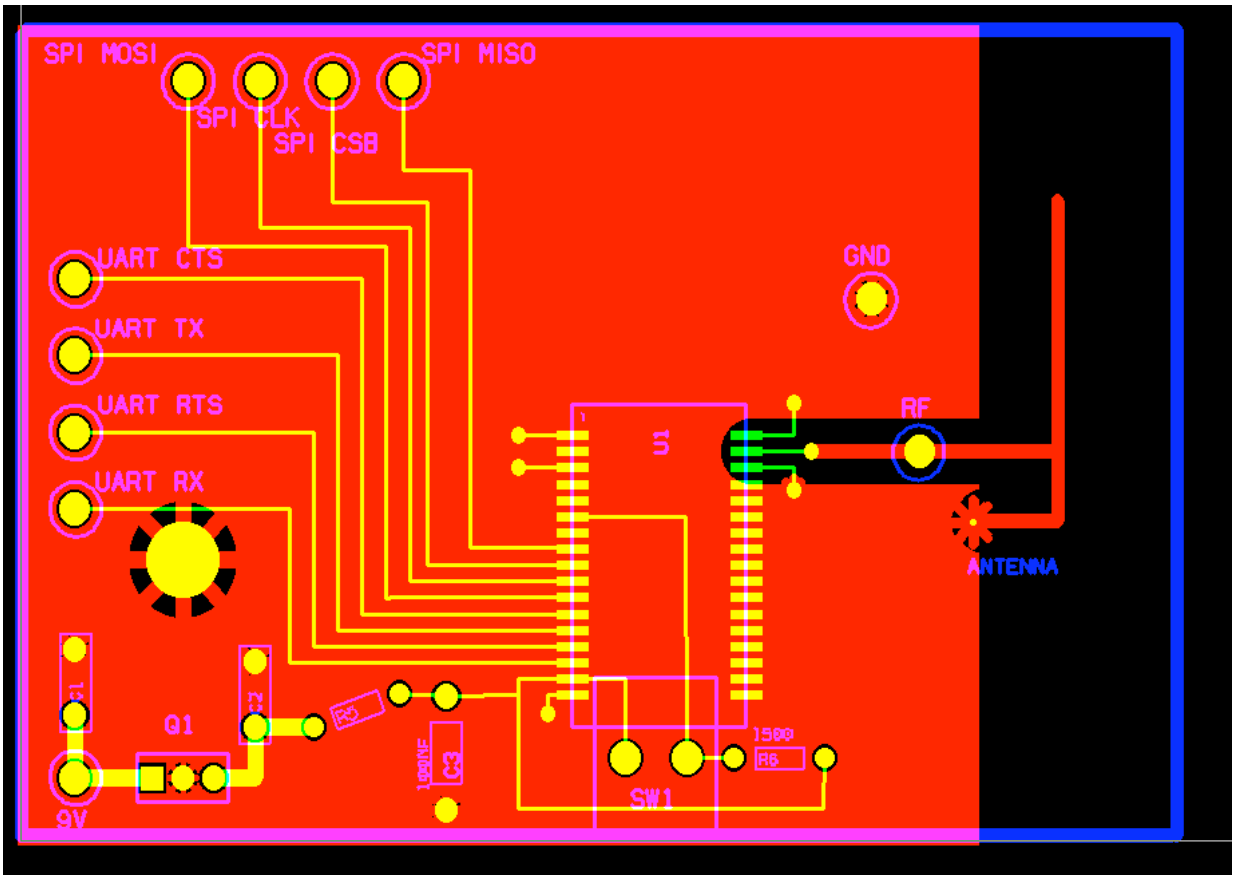


Figure 8: PCB layout for BTM created in Osmond

PCB Fabrication options were explored after successful completion of the Bluetooth PCB layout. Several companies are available to fabricate PCB's, but they vary in price, layers per board requirements/restrictions, and layout file formats supported. The two large companies considered, PCBExpress and ExpressPCB, are both highly regarded fabrication companies, but each has their own policies. ExpressPCB requires that you use their free software to create your designs. PCBExpress, on the other hand, allows use of any layout program that produces Gerber files. In addition, PCBExpress allows student groups to receive free fabrication (up to $300 per academic year) on all project related boards. For this reason, PCBExpress was used for fabrication. The finished PCB from PCBExpress can be found in Figure 9 and 10 below.

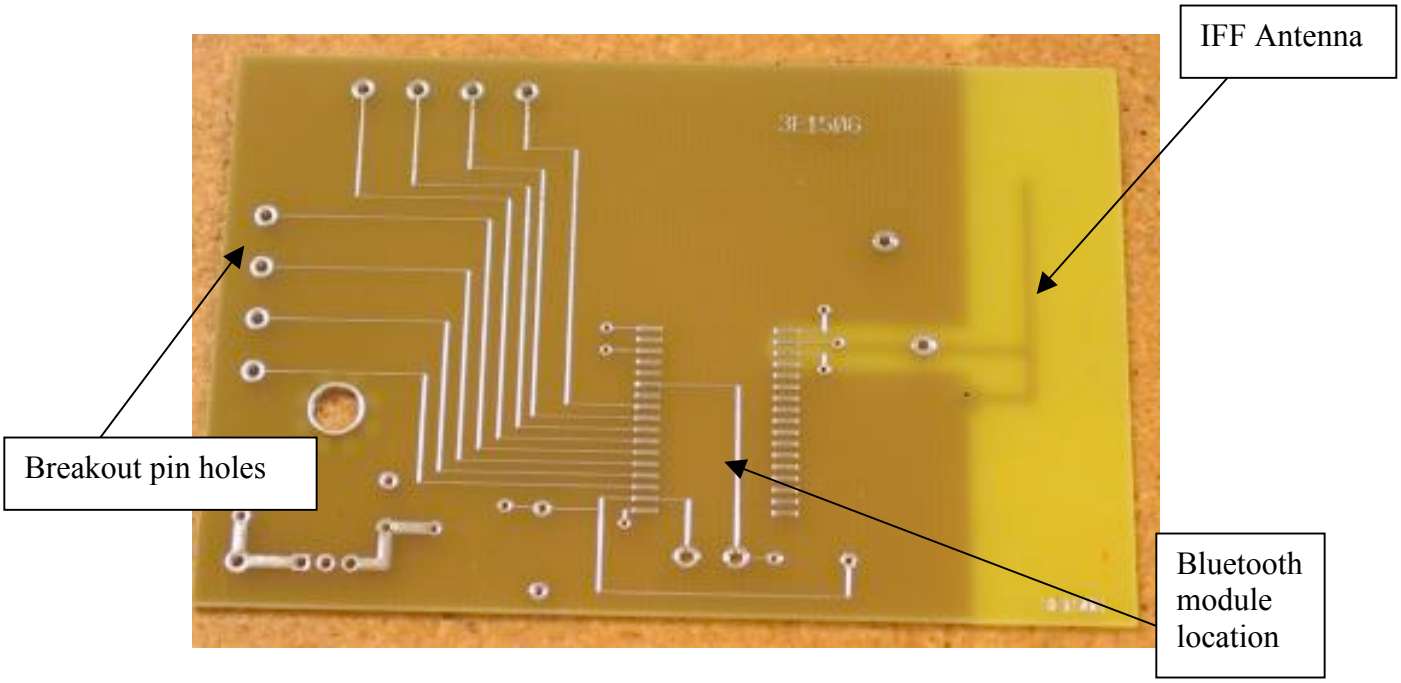


Figure 9: Top Printed Circuit Board

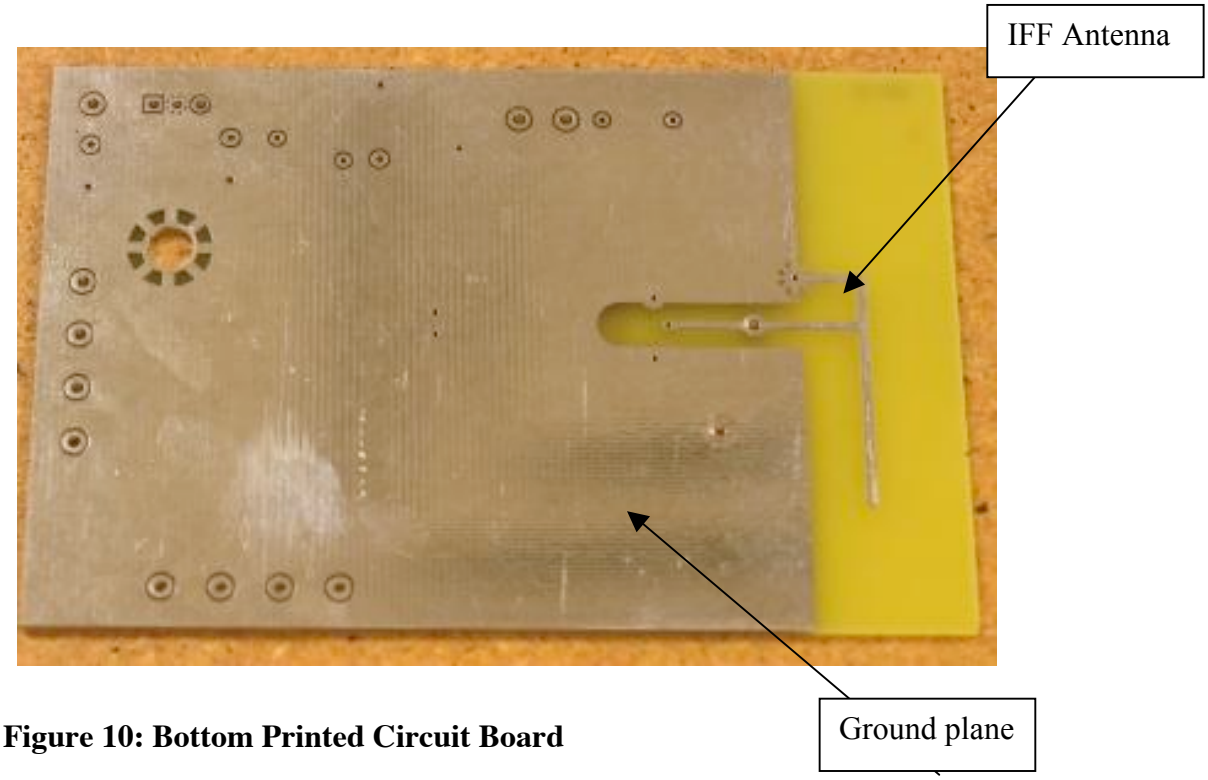


Figure 10: Bottom Printed Circuit Board

Following shipment of the PCB, all components were soldered onto the PCB. The BTM was by far the most expensive and important component to be soldered, so extra

care was used while soldering it to the PCB. The project team's contact at Windigo Systems, Quinton S.Q. Yuan, recommended soldering the BTM using a temperature below 240 Celsius to avoid damage to the internal circuitry of the BTM. After the BTM was successfully soldered, the remaining components (reset switch, voltage regulator, and supporting components) were soldered onto the PCB. The PCB with all components installed and output pins labeled can be found in Figure 11.

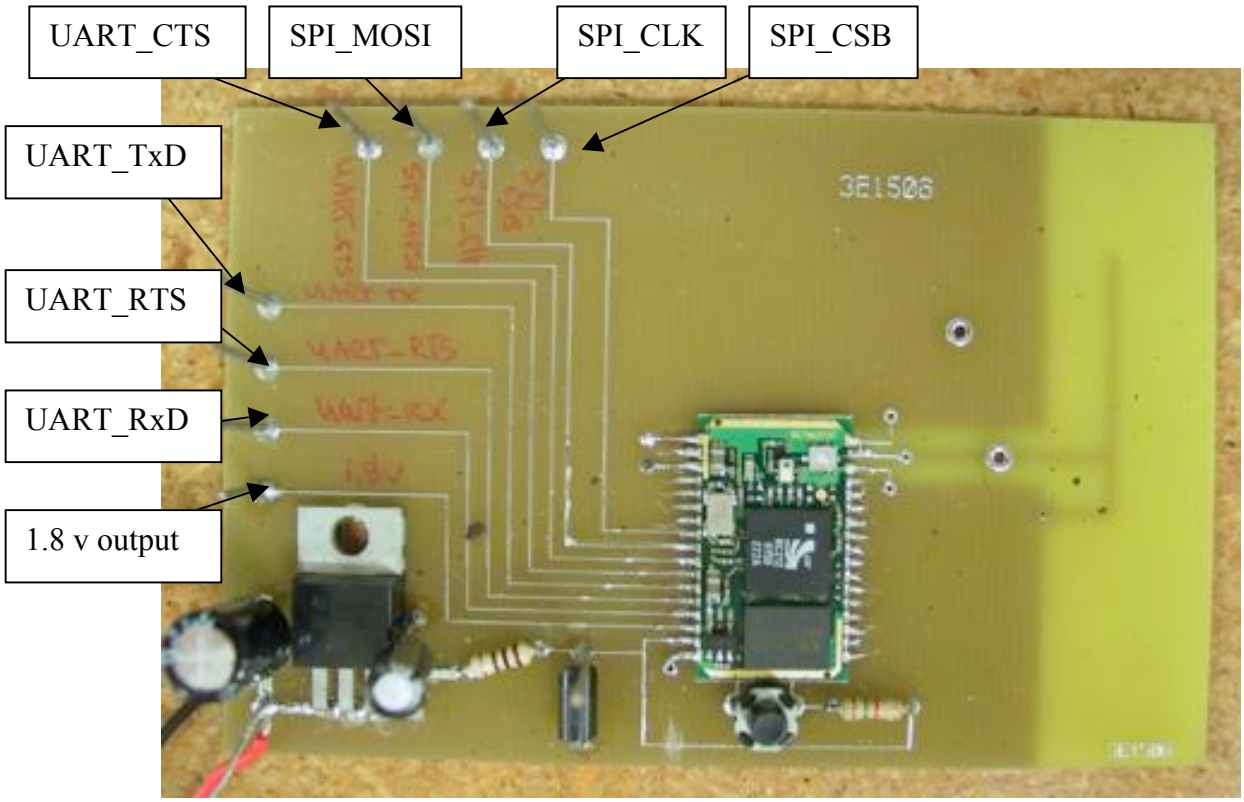


Figure 11: Printed Circuit Board with components

After construction of the Bluetooth PCB and placement of the components, extensive testing was conducted to ensure proper functionality of the BTM. A hardware verification procedure was first established a consistent and reproducible testing plan.

The verification procedure consists of four major sections (the complete document is available in Appendix B):

1. Visual Inspection
2. Limited functional inspection
3. Extensive functional tests
4. Code testing

Initial hardware verification included performing a visual spot check of the PCB to ensure that all components were properly placed and that all traces were valid. Only one problem was found during the visual inspection of the board; both sides of the reset switch had been wired to Vcc, when only one side should have been wired to Vcc and the other to ground. The incorrect trace was corrected immediately by rubbing away the exposed copper trace on one side of the reset switch, effectively tying that side of the switch to the board's ground plane. The project team verified this by monitoring the current drawn by the BTM. The current sits at 4 mA while the module is in normal operation, and upon reset the current fluctuates between 10 mA and 22 mA for a brief period before returning to a steady 4 mA.

Limited functional testing involved checking for RF output in the 2.4 GHz. range when the Bluetooth module was powered—but not controlled by a host controller (i.e. computer or microcontroller)—using both a Simpson model 380 Microwave Leakage Tester and Advantest R4131B Spectrum Analyzer capable of gigahertz level detection. For reference purposes and to verify proper operation of the measurement devices, a known operation Bluetooth device (Sony Ericsson T68i cell phone) was activated and detected using the measurement equipment before testing of the BTM took place. No appreciable leakage from either the reference Bluetooth device or the project team's PCB

was detected with the Simpson Microwave Leakage Tester. In all likelihood, this is because the Simpson instrument is designed to pick up large microwave leakage; that outside of FCC regulations. Sine both the reference Bluetooth device and Windigo Systems BTM are FCC certified, no significant leakage should have been expected. The Advantest R4131B on the other hand, was able to detect the reference Bluetooth device when that device was placed in "discoverable" mode, attempting to find other Bluetooth devices. However, when the Windigo Systems BTM was powered with the regulated +3.3v and placed near the Spectrum Analyzer, no RF output was detected. During this observation, the BTM current drain was noted to be around 3-4 mA—consistent with BTM02C2XX-R's power specifications for default manufacturer mode (Asynchronous Communication Link, UART mode, 38.4 kbps baud rate). This indicated that the BTM was powered correctly and that there was a high probability that the BTM was safely soldered to the PCB. Further research revealed that the BTM does not transmit Bluetooth signals until commanded via Host Controller Interface (HCI) commands; it remains in an idle state until otherwise instructed.

Additional verification involved connecting the Bluetooth PCB to host controller using the UART interface. To remove outside unknowns as much as possible, the Bluetooth module was first interfaced with a computer using a serial connection. Due to the voltage level differences between a computer's RS232 (+/- 30 V) connection and the BTM's UART connections (+5 V), a RS232-to-UART conversion device was designed (see Figure 12) and bread boarded. This device utilized a MAX232 dual EIA-232 Driver/Receiver chip from Texas Instruments. Each receiver converts EIA-232 inputs to 5-V TTL/CMOS levels acceptable for the BTM's UART interface. Each driver converts

TTL/CMOS input levels into EIA 232 levels for the RS232 interface of the computer. A Texas Instruments UA7805C was also included to regulate the 9 volt input voltage and converts it into the 5 volts needed for the MAX232. The voltage regulator has two decoupling capacitors, required by the MAX232 datasheet.

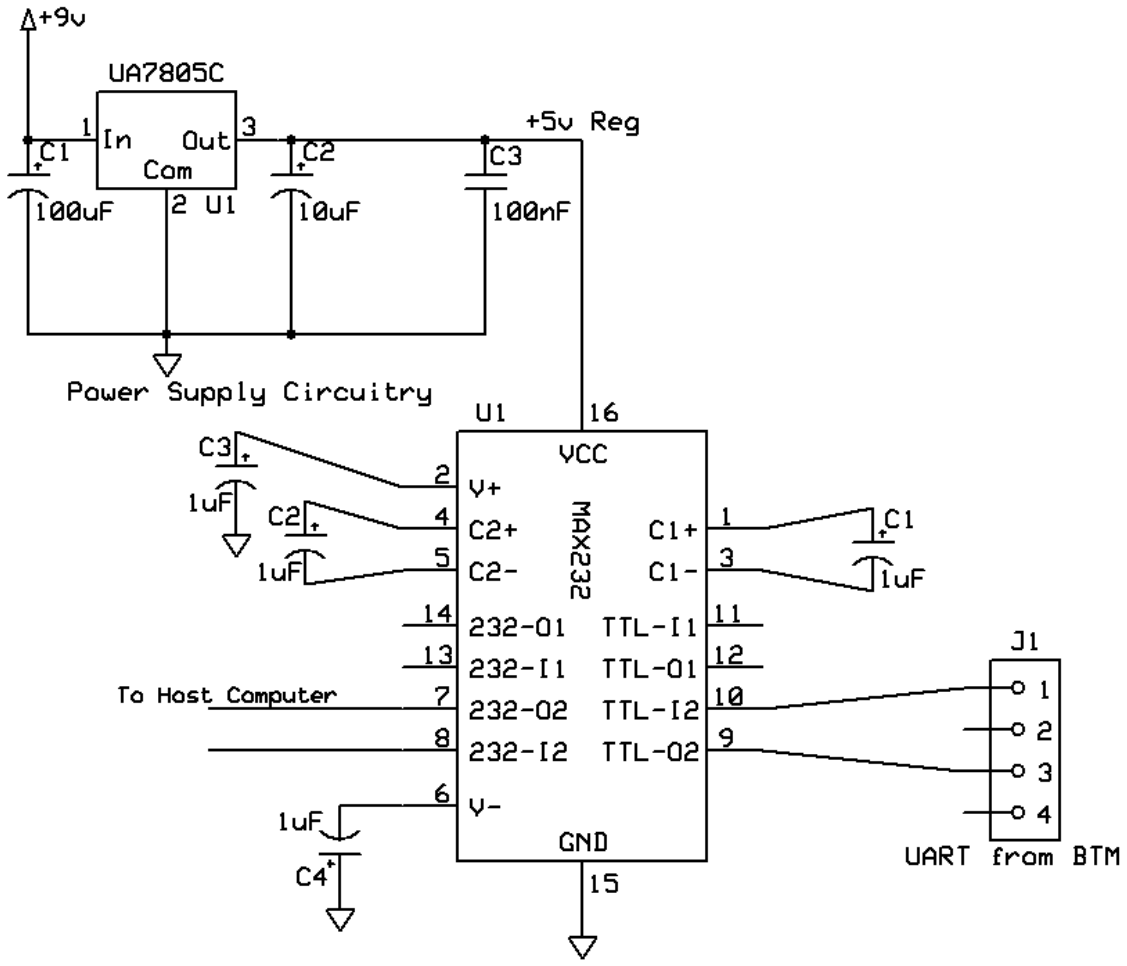


Figure 12: Schematic diagram for UART-to-RS232 converter

The Windigo BTM02C2XX-R contained four UART signals that needed to be interfaced correctly with the nine signals in a D9 RS232 connector. The UART signals included the standard receive (RxD) and transmit (TxD) signals, but also included the

clear to send (CTS) and ready to send (RTS) signals for hardware flow control. The flow control is used to prevent buffer overflow and is not used for HCI flow control, as HCI has it's own mechanism of sending HCI commands, HCI events, and HCI data. The RS232 signals included the same four signals, but also included five additional signals that were not required for communication with the Bluetooth PCB. Table 2 below outlines the connections between the UART and RS232 signals. See Appendix C for complete RS232 pin naming information.

UART Pin No.	Connection Direction	RS232 Pin No.
N/A	GND ←	1
2	←→	2
3	←→	3
N/A	GND ←	4
N/A	GND ←	5
N/A	GND ←	6
7	→	8
8	←	7
N/A	GND ←	9

NOTE: Pin 2 = RxD, Pin 3 = TxD, Pin 7 = RTS, Pin 8 = CTS

Table 2: Signal connections between BTM and Computer

Once the BTM was physically connected to the computer as shown in Table 2, HCI commands were sent from within a terminal program called Docklight to the BTM. This terminal program provides much more control and monitoring capabilities than

many terminal programs included with MacOS X and Windows XP, plus it allows you to create “send sequences” in either ASCII, hexadecimal, decimal, or binary form. For communication testing purposes, a simple HCI command was sent: `0x01 0x03 0x0C 0x00`, where `01` specified the start of an HCI command packet over UART/RS232, `03 0C` is the HCI opcode for a Bluetooth reset, and `00` is the required parameter length, but since the reset HCI command has no parameters, the value is null. More information on HCI command format and Bluetooth module operation is provided in the next section discussing the Bluetooth driver. Docklight indicated that the HCI command had been placed on the RS232 bus for transmission to BTM, however, the expected return HCI Command Complete Event `0x04 0x0E 0x04 0x01 0x03 0x0C 0x00` was not observed. To troubleshoot the communication problems, the signals sent on the four connected RS232 lines and the four UART lines were monitored using a Hewlett Packard Oscilloscope while a BTM reset was sent.

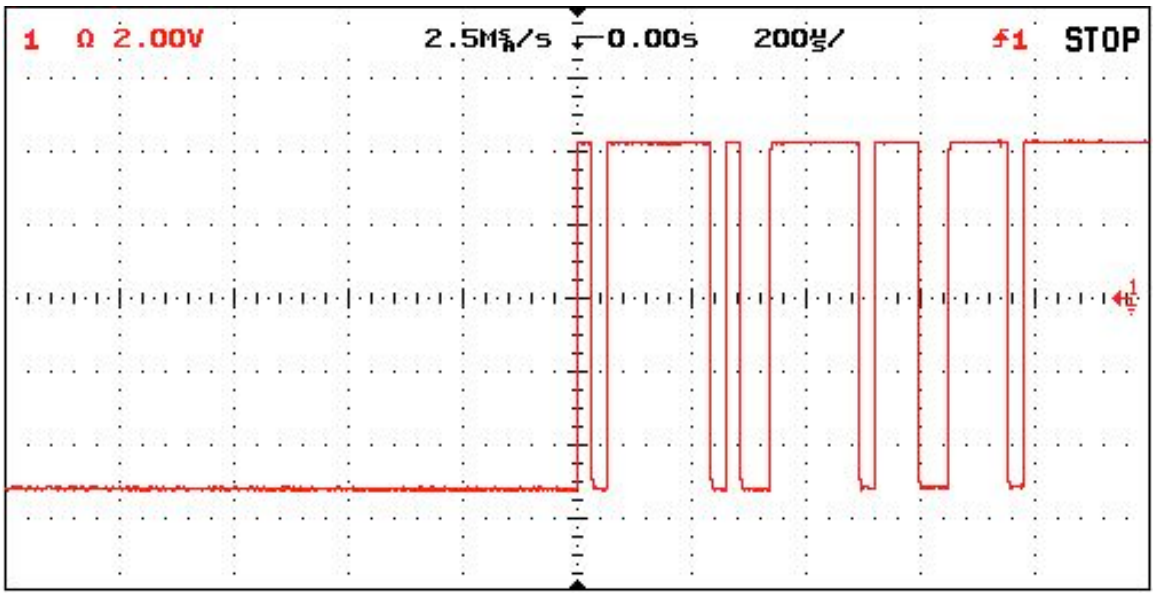


Figure 13: Computer's TxD pin

Figure 13 shows the Bluetooth reset HCI command being sent over the transmit line from the computer. The signal is active when data is transmitted from the computer to the device. When no data is transmitted, the signal is held in the mark condition (logic "1", negative voltage). The signal is then fed into the RS232-to-UART converter, where the voltage is inverted and lowered to fall within the 0 – 4v range of UART signals (Figure 14). The output from the converter goes into the BTM's receive pin.

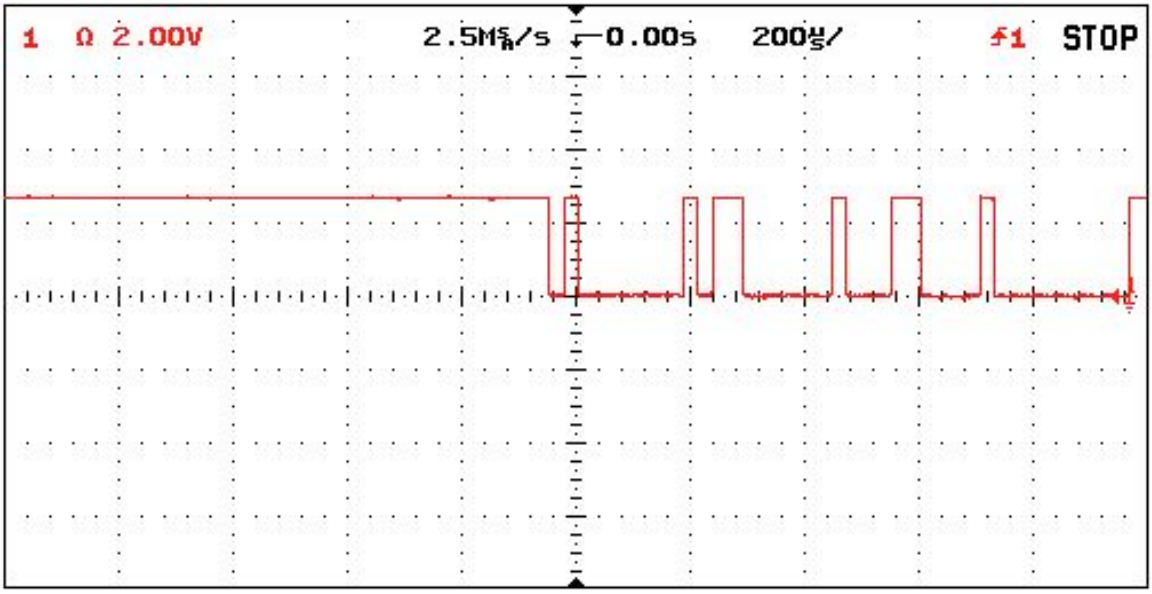


Figure 14: BTM's RxD pin

Figure 15 and 16 show the connection between the Request To Send (RTS) of the computer and the Clear To Send (CTS) of the BTM. The RTS signal is asserted (logic '0', positive voltage) to prepare the BTM for accepting transmitted data from the computer. When the BTM is ready, it acknowledges by asserting CTS.

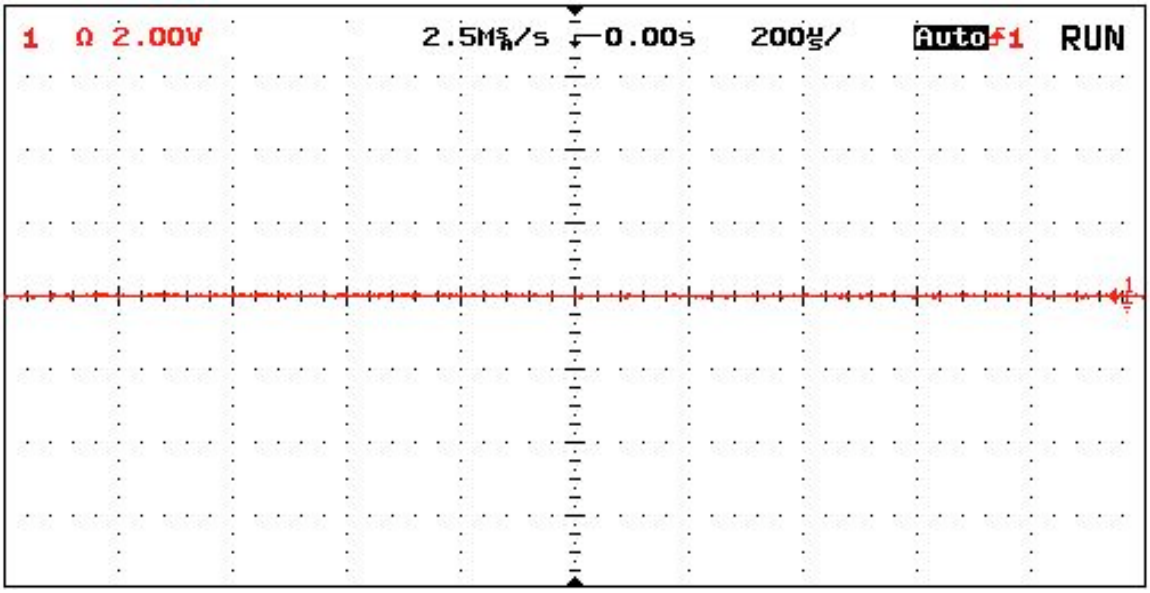


Figure 15: Computer's RTS pin

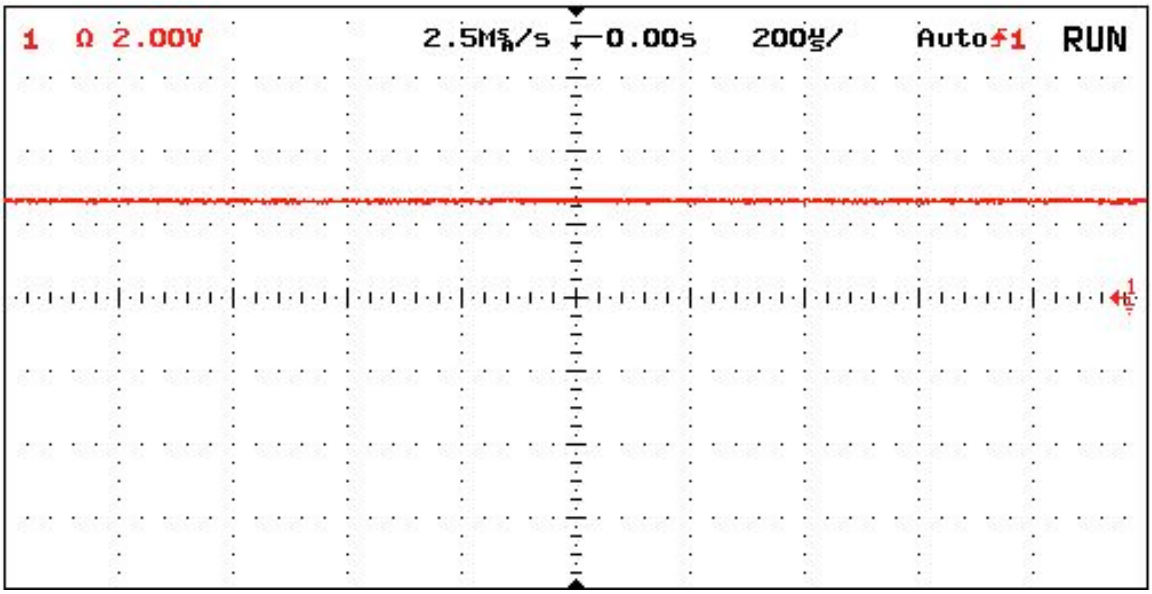


Figure 16: BTM's CTS pin

Figure 17 and 18 show the opposite connection; the connection between the RTS of the BTM and the CTS of the computer. The RTS from the BTM appears to be at zero volts, suggesting that it is not attempting to send data back to the computer. This would be consistent with current behavior on the computer. A Bluetooth reset HCI command is

sent, but nothing is returned. As a result, the TxD line from the BTM is forever in the mark condition and does not send data.

Contact with Rob Barris, who has been helpful with microcontroller development, suggested that sense and mark logic levels were incorrect on the Bluetooth side of the communication and thought a inverter between the signal coming out of the RS232-to-UART converter might help matters. The senior project team attempted that setup using a 74LS04 Hex Inverter to invert the RxD, TxD, CTS, and RTS lines going into and out of the BTM in different combinations, but the results were the same.

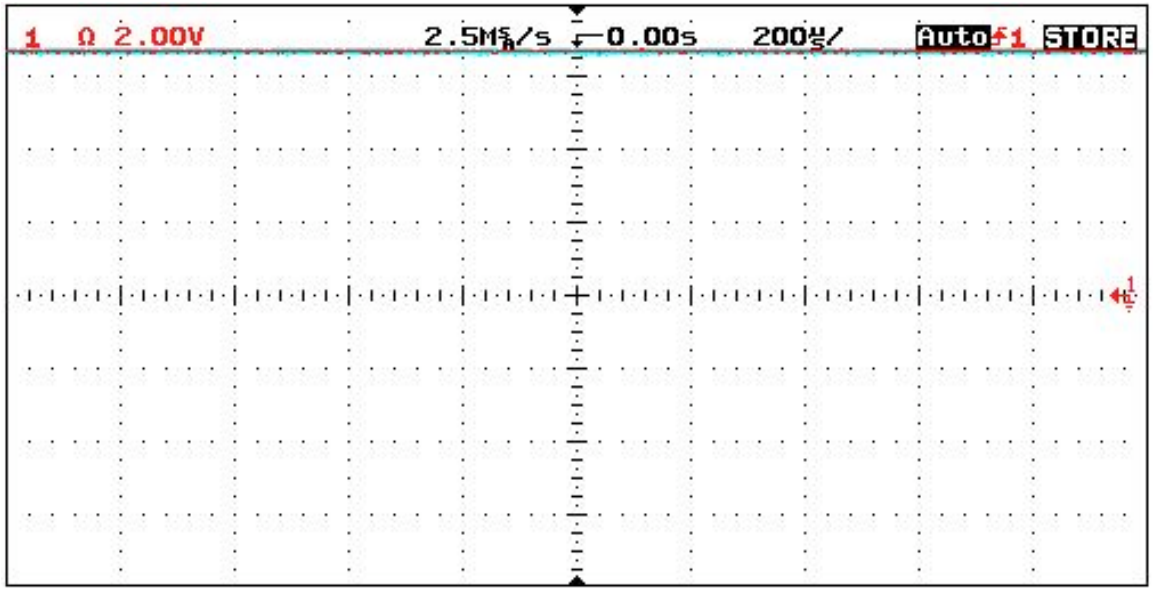


Figure 17: Computer's CTS pin

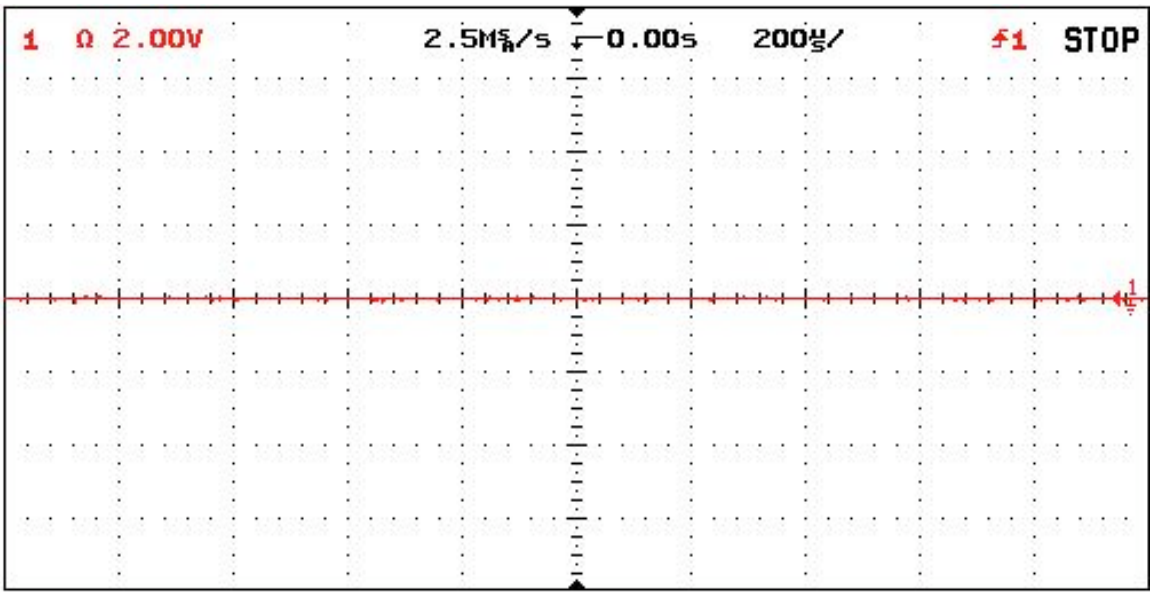


Figure 18: BTM's RTS pin

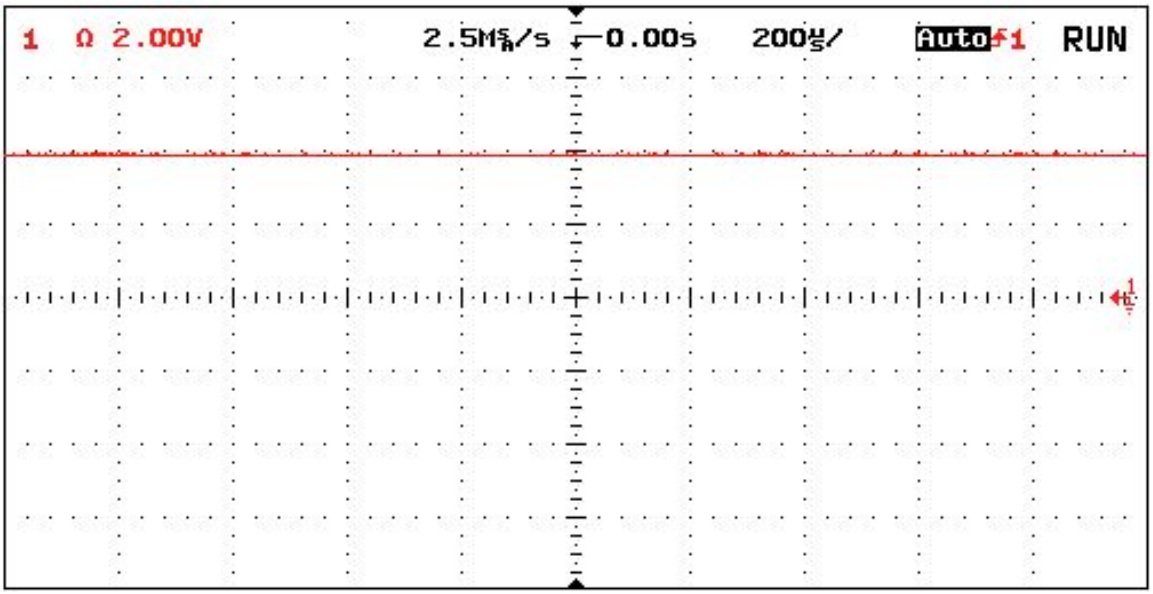


Figure 19: BTM's TxD pin

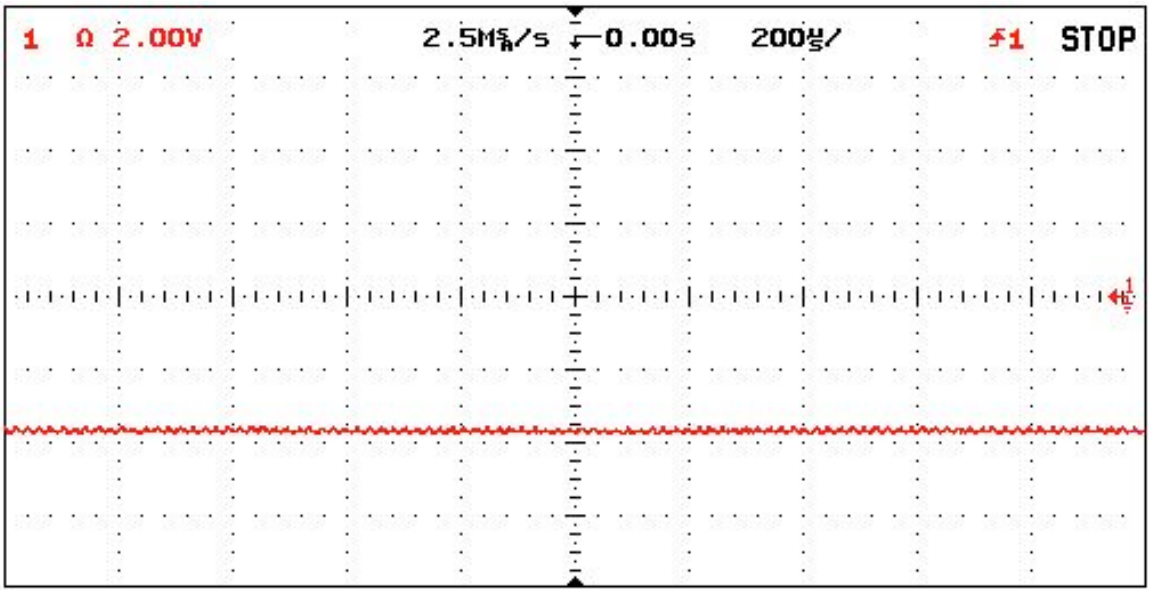


Figure 20: Computer's RxD pin

Results from experiments indicate that the BTM from Windigo Systems is drawing the correct current and voltage and reacts as expected during a manual restart, yet communication between the BTM and host computer has still not been established. To fully troubleshoot this problem, another BTM from Windigo System will have to be obtained to eliminate the possibility of a Dead On Arrival (DOA) unit.

Bluetooth driver

The Bluetooth module contains a piece of firmware code known as the Bluetooth stack. This stack contains all the low level operations and commands needed to operate the base band controller and radio receiver/transmitter of the Bluetooth chip. A developer interacts with the Bluetooth stack using Host Controller Interface (HCI) commands and writes driver software for the host environment that sends various HCI commands in a

certain sequence to set the state of the Bluetooth module. The program that defines the sequences of commands sent is called a Bluetooth driver.

The Motorola 68HC12 microcontroller was chosen as the development platform for the Bluetooth driver for various reasons. Because the project team envisioned the Bluetooth communication board being integrated with the microcontroller board in EECS 375, any Bluetooth driver implemented needed to be compatible with the microcontroller board currently in use (Motorola 68HC11). The 68HC12 is fully backwards compatible with the 68HC11 microcontroller used in EECS 375, but provides a significant speed increase, a 16-bit architecture versus the 68HC11's 8-bit architecture, lower power consumption, a rich superset of the 68HC11 commands, and an easy to use single-wire background debug mode for development. A M68EVB912B32 68HC12 development board was provided by Professor Frank Merat. The M68EVB912B32, pictured in Figure 21, contains the 68B912B32 model in the 68HC12 series along with an RS232 port, Background Debug Mode (BDM) terminals, the DBug12 monitor/debugger program with four modes of operation, prototype space, the necessary power circuitry, and pin blocks for easy connection to the microcontroller.

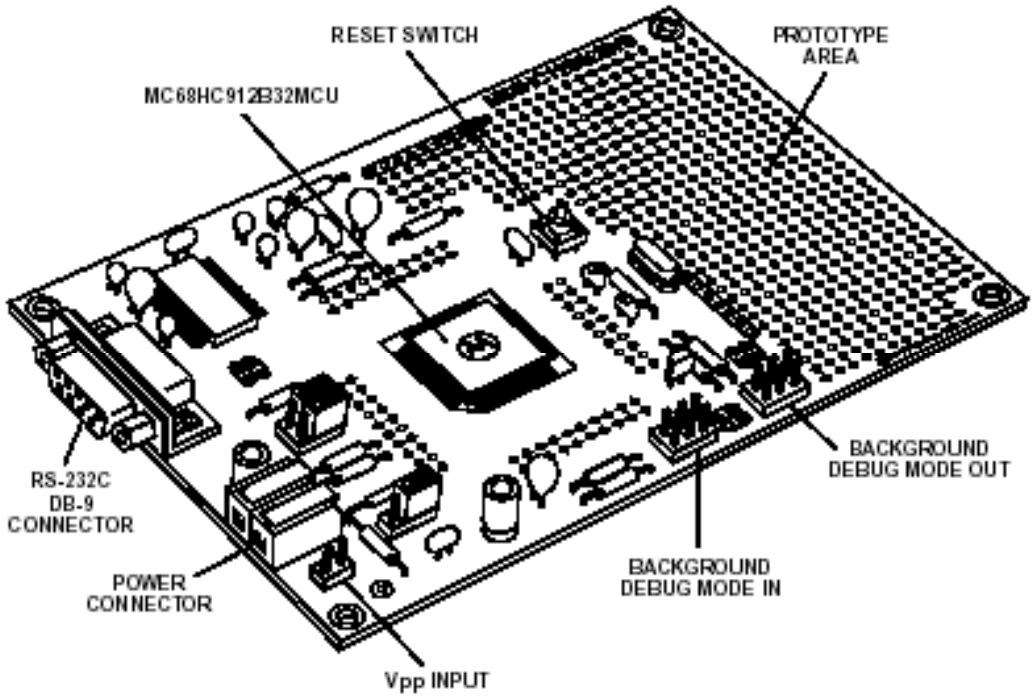


Figure 21: Motorola M68EVB912B32 68HC12 Development board

The M68EVB912B32 development board included software from HIWARE and P&G Microcomputer Systems, Inc. for compiling code for the microcontroller, but these programs were outdated and difficult to use. In place of the included pieces of software, the senior project team used the GNU Development Chain for 68HC11&68HC12 microcontrollers, an open source cross compiler that works on any host computer that supports the "GNU's Not Unix" (GNU) tools. For Bluetooth driver development, the GNU Development Chain was installed on a Mac OS X v10.2.3 system using a Perl script made available by 68HC12 developer Rob Barris. A Keyspan USB-to-Serial PDA Adapter was purchased to allow code developed on the Mac OS X system to be downloaded to the 68HC12 microcontroller via the RS232 port on the development board.

The development chain for the 68HC12 follows the standard software generation path (See Figure 22), but creates an S-record after the linker stage instead of an executable file (aka .exe file). The S-record format is an encoded program format used by the 68HC12 microcontroller; all code targeted for a 68HC12 must be converted to S-record format. An S-record consists of five fields of information: Type, Record Length, Address, Code/Data, Checksum. Eight S-record type codes are defined to accommodate the various functions to operate the microcontroller, but programs typically only implement the S-record types appropriate to their purpose. For example, the DBug12 program that runs on the microcontroller only supports two S-record types, S1 and S9, since it is only concerned with executing code/data (S1 type) and terminating the program (S9 type) when needed.

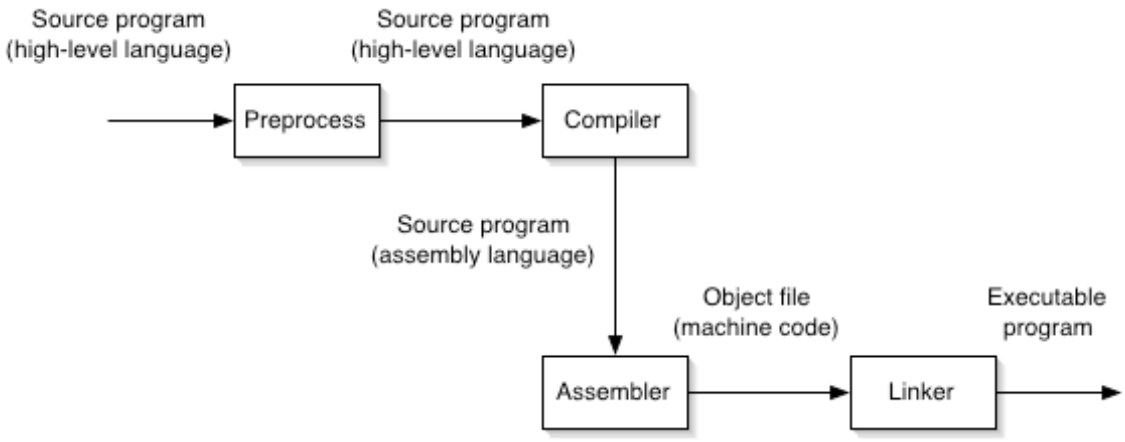


Figure 22: Software Generation Process

Some problems were encountered when initially attempting to create a simple S-record file using the GNU cross-compiler. Code refused to complete the compilation process when the command <CMD> was used as instructed in the cross-compiler

manual. Assistance from users on the GNU Development Chain mail list provided feedback, which helped resolve the problem. <PROBLEM RESOLUTION>.

After compiling the test source code to a S-record file, the S-record was downloaded to the 68HC12 microcontroller via a terminal program; Zterm was used on the Mac OS X system, although there are numerous other programs available for this purpose. Programming speeds at 9600 baud were reasonable for small files, but excruciatingly slow for larger pieces of code (30+ minutes for 72 Kilobyte file). It was also learned that when user code was written to Flash EEPROM, it erased the Dbug12 monitor program. Upon reset of the microcontroller, the user code would execute. However, since little visual feedback is possible when microcontroller code is running without a debugger monitoring memory locations, the development process was nearly impossible.

Due to the time needed to download code and the difficulty in debugging user code, a Background Debug Module "POD" was purchased. The Technological Arts microBDM12SX+ POD connects to the BDM IN connector on the M68EVB912B32 board (see Figure 21) and to the host computer via a RS232 cable. The POD allows the developer to use all the memory on the target microcontroller while still being able to run an interactive debugger while user code is executing. Inside of the POD is a minimal microcontroller setup running the DBUG12 program that acts as a non-intrusive controller for the target system. The complete setup is shown in Figure 23 below. With this setup, development of the Bluetooth driver could commence.

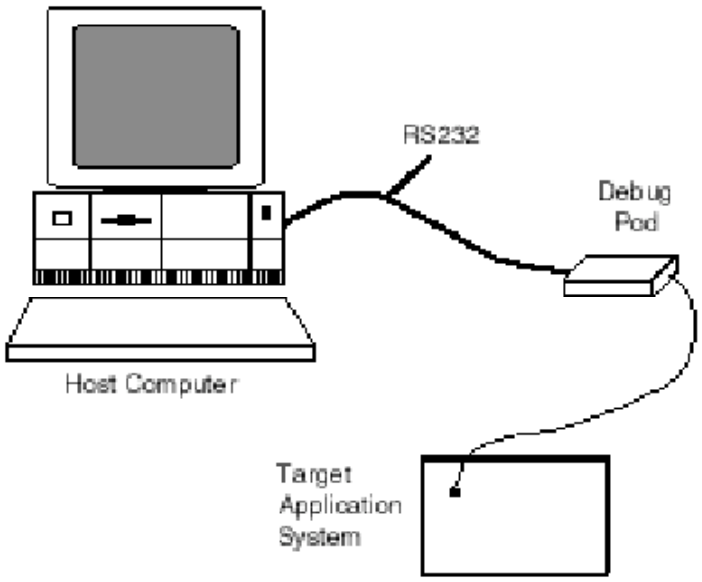


Figure 23: Development setup with POD

Originally, the senior project group planned to write their own Bluetooth driver, but due to the amount of time needed to complete the Bluetooth PCB and the time associated with creating a Bluetooth driver, it was decided to instead work with "MicroBlue", a project started by the University of Karlsruhe in Germany to create a Bluetooth stack that would operate on a wide range of microcontrollers. "MicroBlue" includes a great deal of functionality that is not necessary for use in the "Integrating Bluetooth onto LEGO robots" project, so changes have been made with the approval of the "MicroBlue" developers. However, testing of the modified stack could not take place since the integration with the microcontroller phase of the project did not take place.

A Bluetooth driver is basically a state machine that controls the BTM via Host Controller Interface (HCI) commands and reacts to HCI commands received from the BTM. The HCI commands sent from a host controller allow the host to interact directly

with the Bluetooth stack (aka “HCI firmware”) on the BTM. Figure 13 below shows the hardware and software layers that sit above and below the Bluetooth stack on the BTM. The HCI commands travel over the physical interface bus between the BTM and the host controller (i.e. computer or microcontroller). The Host Control Transport Layer of the Bluetooth stack provides for the host to send HCI commands, Synchronous data (SCO) and Asynchronous data (ACL) to the BTM. It also provides for the host to receive HCI events, SCO data, and ACL data from the BTM.

HCI command packets have a rigid format specified by the Bluetooth v1.1 specifications. HCI commands consist of three main sections: opcode, parameter length, and the required parameters. Figure 14 shows the contents of a HCI command packet. The 2 byte opcode identifies the command to be performed while the parameter and parameter length fields are used to indicate how many pieces of data are being send with the command. When the BTM completes most of the HCI commands, a Command Complete event is sent back to the host. Some HCI commands do not have Command Complete events, but instead send Command Status events back to the host periodically depending on the state of the BTM.

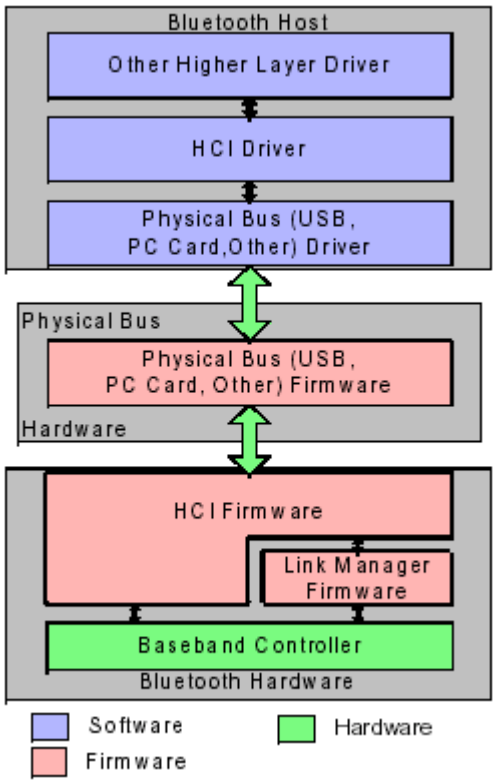


Figure 24: Interaction of host and BTM using HCI commands

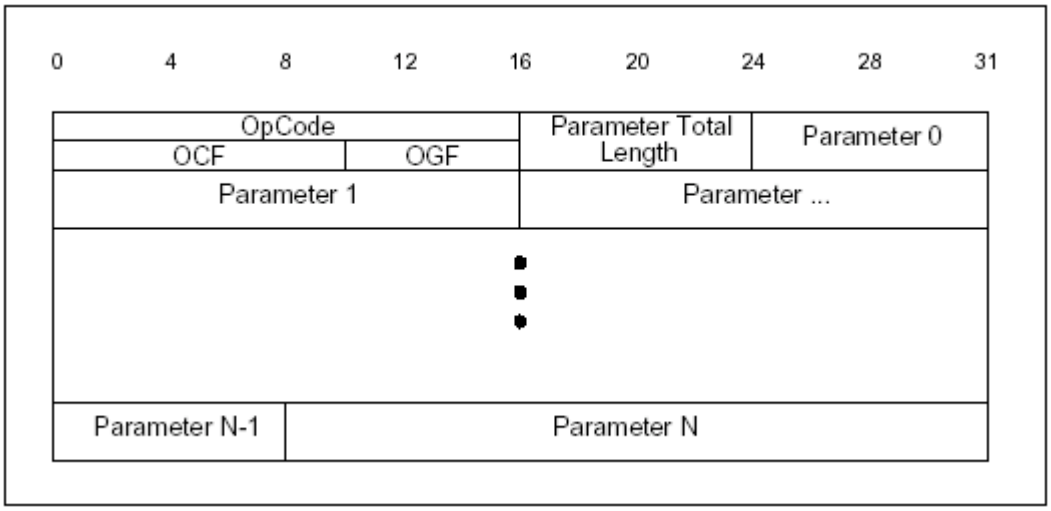


Figure 25: HCI Command Packet

The "MicroBlue" Bluetooth stack was already setup to send HCI commands using the format described above, but needed to be modified to operate using the opcode of the 68HC12 microcontroller.

Since the senior project group was unable to integrate the Bluetooth communication board with the Motorola 68HC12 microcontroller, testing of the Bluetooth driver with the BTM could not be conducted.

Results

The "Integrating Bluetooth onto LEGO robots" senior project team was able to successfully complete five out of the seven major tasks of their project and meet all original performance requirements and specifications with our design. The major phases that were planned to be completed at this stage are as follows:

- Acquire an integrated BTM
- Design a high frequency Integrated Inverted F Antenna (IIFA)
- Design and fabricate a Bluetooth communication board with serial communication links and IIFA
- Test and verify operation of the Bluetooth communication board
- Design a Bluetooth communication driver
- Achieve communication between the Bluetooth module and a host computer
- Integrate Bluetooth communication board with microcontroller

The project team has acquired a Bluetooth module (BTM) and integrated it onto the communication board with an Integrated Inverted F Antenna (IIFA). This can be verified in Figures 8 through 11, which illustrate the layout of the PCB and the

completely constructed PCB with components. The Bluetooth communication PCB was verified using several methods. First, continuity of traces was verified using a Digital Multimeter. During this process, a few errors were noted and corrected. Subsequent testing involved establishing communication between the BTM and a host (i.e. computer).

Communication between the Bluetooth module and the host computer has not yet been established. Since communication with the computer has not been achieved, the Bluetooth communication board has not been interfaced with the microcontroller.

A Bluetooth driver has been established by working with code from the "MicroBlue" project. It was determined that working with existing code was more efficient that writing everything from scratch. The "MicroBlue" code has been modified as needed to accommodate the 68HC12 microcontroller.

All of hardware specifications for the PCB and its integrated components have been met. The specifications are listed below.

PCB specifications:

- Mobile power source (9v battery with supporting voltage regulator circuitry for +3.3v power to BTM)
    - Used Texas Instruments UA78M33C voltage regular to regular power.
    - Verified using a Digital Multimeter on both input and output of power regulation circuit.
- Decoupling capacitors on power supply lines to eliminate noise/charge buildup. Need to be as close to BTM chip as possible for greatest effectiveness
    - See Figure 8 and Figure 11 for placement of decoupling capacitors.

- Integrated Inverted F-Antenna (IIFA) size- 25mm long, 1mm wide trace, 6mm distance between antenna and ground plane, 5.4mm between antenna feed and antenna ground.
    - See Figure 8 and Figure 10 for the IIFA on the PCB.
- IIFA placement- no ground or power lines may be routed beneath the antenna
    - See Figures 8, 9, and 10 for the IIFA. The IIFA was placed away from other components, power lines, and other traces.
- IIFA must be connected to the BTM through a small gap in the ground plane
    - See Figure 8 and 10 for gap in ground plane.
- Ground plane and IIFA on bottom side of board
    - See Figure 10 for bottom side of PCB.
- Output pins for computer/microcontroller UART connection
    - Test points were traced to the BTM's UART connection for easy access and integration with other serial devices. See Figure 11.
- Reset switch and supporting circuitry
    - See Figure 2 and 11 for schematic and completed PCB.
- Board size limited to 3.8" X 2.5" by manufacturer

The BTM could not be fully tested since communication was not established between the Bluetooth communication board and the host device (i.e. computer). However, the Windigo Systems BTM02C2XX-R has been certified to meet all Bluetooth version 1.1 specifications.

Bluetooth Module specifications:

- Transmission range of at least 20 feet
  - Class 2 Bluetooth devices like the BTM02C2XX-X have a transmission distance of 30 feet.
- Data rate between host and BTM of more than 32K baud
  - BTM02C2XX-X can transmit up to 34.8k baud
- Data transmission between two BT's greater than 500 Kbps
  - Bluetooth version 1.1 specification calls for Class 2 devices to transmit at up to 1 Mbps.
- Allow more than two BT devices to communicate
  - BTM02C2XX-X operates at full Class 2 speeds, allowing for full 7 slave piconet support (i.e. up to seven devices can communicate at same time).
- Mountable onto Printed Circuit Board using normal soldering techniques (i.e. non ball grid array component)
  - BTM02C2XX-X is a surface mount device with exposed pins on either side, allowing for placement on PCB using standard soldering iron at low temperatures.
- UART serial communication interface
  - BTM02C2XX-X contains standard UART pin configuration (RxD, TxD, RTS, CTS).
- Antenna input with 50 Ohm or greater input impedance

- The BTM02C2XX-X does not have an antenna built into the integrated circuit, but does contain one antenna RF port (pin 33) which has 50 Ohm impedance.

- Low chip power consumption
  - The BTM02C2XX-X's power specifications are listed in the datasheet. During typical synchronous communication (SCO), the BTM consumes on average 28 mA. While in lower power mode (aka "Deep Sleep Mode"), the BTM consumes only 15 uA of current.

RS232 converter specifications:

- 9v battery regulated to +5v
  - Used Texas Instruments UA7805C voltage regular to regular power
- Pin connections
  - RxD on BTM connected to TxD on host
  - TxD on BTM connected to RxD on host
  - RTS on BTM connected to CTS on host
  - CTS on BTM connected to RTS on host
- Decoupling capacitors on power supply lines to reduce noise and charge buildup
  - See Figure 12 for schematic of RS232 converter. The voltage regulator IC contains decoupling capacitors according to the UA7805C's datasheets recommendations.
- Non utilized pins are no connects (NC)

- All pins not used are left as NC. This is consistent with recommended procedure in the UA7805C datasheet.

Microcontroller specifications:

- Backwards compatibility with 68HC11 microcontroller
  - Feature of 68HC12 family of microcontrollers.
- 4 Mhz. or faster
  - The 68HC12 family operates at 8 Mhz.
- 10 kb of built in memory for user code
  - The MC68HC912B32 contains a 32 kb flash EEPROM for program memory, 1 kb of static RAM, and 768 bytes of byte-eraseable EEPROM.
- UART serial communication interface
  - The MC68HC912B32 contains UART connections.
- 10 digital I/O pins
  - The MC68HC912B32 has 80 pins total, 64 of which can be used for general purpose I/O

Software specifications have been met in the implementation of the Bluetooth driver.

- Bluetooth driver should meet Bluetooth v1.1 specifications
  - All HCI commands used are compliant with v1.1 of the Bluetooth specifications
- Bluetooth driver should operate on Motorola 68HC12 microcontroller

    - Testing was not completed due to fact that Bluetooth communication board was not fully integrated with 68HC12 microcontroller host.
- Microcontroller should be able to initiate Bluetooth communication over UART interface using HCI commands
    - Testing was not completed on the 68HC12 microcontroller for reasons stated above.
- Bluetooth driver should allow for discovery of other devices, connection to other devices, and being discovered by other devices.
    - Testing was not completed on the 68HC12 microcontroller for reasons stated above.

Review of Other Implications

The senior project team addressed each implication raised during the initial project proposal phase and a few that had not originally been considered.

- Economics: The senior project team chose to utilize a readily available integrated Bluetooth module from Windigo Systems instead of borrowing or purchasing a several thousand dollar Bluetooth development system. The BTM02C2XX-R was only $90 and easily fit within the projects budget constraints.
- Education: The experience gained from this project will greatly benefit those involved with the project even though all phases were not completed. The project team gained experience with high frequency antenna design, PCB layout software, microcontroller programming, and serial communication interfacing. The work performed on this project can act as a springboard for future Bluetooth related projects.

- Environment: The Windigo BTM02C2XX-R and Motorola M68EVB912B32 68HC12 development board are both FCC approved devices and do not emit unsafe levels of radiation, RF signals that may interfere with other devices, or any type of environmental pollution.
- Health and Safety: The Bluetooth module PCB has been designed to be safe for student use. All devices are properly grounded and connected. The Bluetooth module does not emit at a high level of power, reducing radiation concerns.
- Manufacturability: All PCB layout files have been archived and are available for additional production needs. Parts used on the Bluetooth module PCB are readily available.
- Sustainability: The PCB has been carefully designed and could be used in other Bluetooth related projects. The software is not as robust or full featured as desired, but provides a good backbone for future projects.

Conclusions

The senior project team completed all but one of the projects deliverables and completed 5 out of the 7 phases of the project. Given the lack of adequate developer support, the amount of work involved in creating a functional PCB for a Bluetooth module, and the amount both team members learned during the course of the project, the senior project team feels satisfied with the results.

The original project plan was overly optimistic given the amount of work to be completed. As a result, the project plan and methodology were modified mid-semester to better reflect achievable goals.

Recommendations

Headway has been made to achieve the final goal of this project—Bluetooth communications on the LEGO robots in EECS 375. A Bluetooth module PCB was designed and fabricated, an integrated high frequency antenna was created, and communication was close to being established with the Bluetooth module from a host computer.

It is the hope of this senior project team that a future senior project team will pick up this project. The next step would require the Bluetooth module to be integrated with the 68HC12 microprocessor and continuing the Bluetooth driver development process. After that process has been completed, another team could redesign the EECS 375 microcontroller board and insert the Bluetooth PCB design created this semester in place of the current RJ45 serial connection.

Appendix A

Original Methodology

1. Research microcontroller and Bluetooth module architecture.
    1.1. Read application notes provided by chip suppliers
    1.2. Read trade articles related to Bluetooth and high frequency chip design
    1.3. Install and setup microcontroller development environment
2. Integrate Bluetooth onto microcontroller development board to test optimal integration methods.
    2.1. Determine hardware design requirements for Bluetooth controller (i.e high frequency concerns and considerations)
    2.2. Research Printed Circuit Board (PCB) fabrication companies for high frequency board fabrication
    2.3. Research existing high frequency PCB layout designs
    2.4. Read trade articles related to Hardware Control Interface (HCI) microcontroller programming
    2.5. Write HCI code for microcontroller
    2.6. Design PCB for Bluetooth controller
    2.7. Build PCB for Bluetooth controller
3. Integrate Bluetooth module with current microcontroller board; writing the necessary software for the microcontroller.
    3.1. Remove existing serial connection on microcontroller board utilized in EECS 375
    3.2. Connect Bluetooth PCB with serial connection on microcontroller board

3.3. Write software libraries needed for student programs to interact with Bluetooth chip

3.4. Test software libraries needed for Bluetooth connectivity

4. Write supporting software for the computer allowing for student written code to be downloaded to the microcontroller board.

4.1. Learn Cocoa programming language

4.2. Become familiar with Interface Builder and Project Builder development software

4.3. Create Graphical User Interface (GUI) for microcontroller development software

4.4. Create GUI interface to interact with microcontroller board via Bluetooth (send and receive)

5. Redesign the current microcontroller board to include the Bluetooth module, a USB port, the updated microcontroller, and replace out of date parts.
6. Verify the redesigned microcontroller board using simulation software (PSpice).
7. Build redesigned microcontroller board(if time permits)

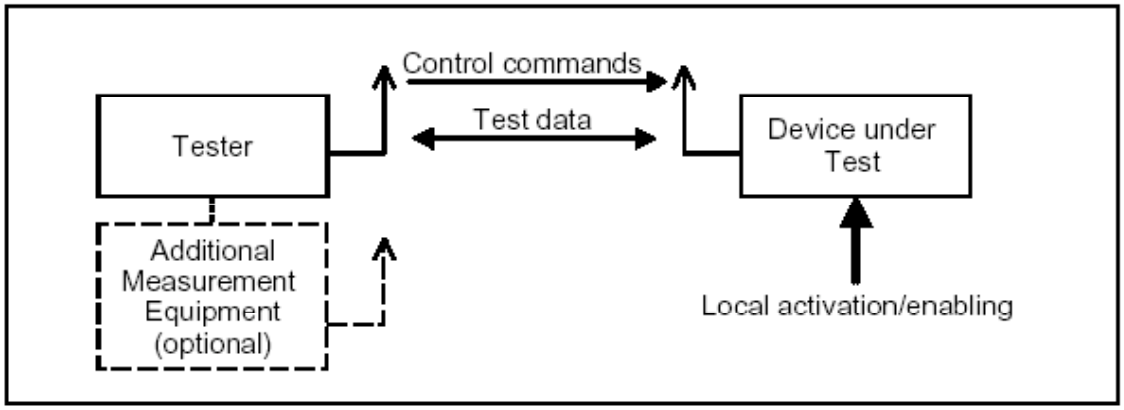
Appendix B

Hardware Verification procedure for BT PCB

5. Visual Inspection
    a. Verify that all components are properly connected to the PCB
        i. Correct polarity on capacitors and diodes
        ii. All IC's connected to ground
        iii. No s older run-off between soldered connections on IC's
        iv. Complete solder connections between component pins and PCB mask
    b. Verify proper clearance provided between components (i.e. no components touching)
    c. Verify traces connect to proper pins on components
    d. RF interference/distortion problems in design

6. Limited functional inspection
    a. Test for continuity between connections using DMM continuity check
    b. Test for switch operation (open/close) on PCB
    c. Microwave leakage test using Simpson model 380 Microwave Leakage Tester
        i. Test known Bluetooth module (D-Link DWB-120M)
        ii. Test Bluetooth PCB
    d. Spectrum Analyzer test for 2.4 GHz signal

7. Extensive functional tests
    a. Connect BT module to MCU or BT module to computer via RS232 interface
        i. Construct RS232-to-UART converter using MAX232 IC
        ii. Test RS232-to-UART converter for proper voltage/current characteristics
        iii. Attach BT module to UART side and computer to RS232 side via RX and TX signal lines (tie RTS and CTS signals low)
        iv. Attempt communication between BT module and computer via terminal program (38.4k baud, 8 bit, 1 stop bit, no parity, no hardware flow control)
    b. Enable device test mode on BT module (see CSR documentation)
        i. <notes from CSR documentation on enabling test mode>
    c. Create test environment with DUT and Tester BT device (see Figure below)



d. Perform following certification tests

   i. Transmitter Test

   ii. LoopBack Test

   iii. Pause Test

8. Code testing

   a. Write simple Host Control Interface (HCI) driver for MCU

   b. Exercise HCI driver on MCU by…

      i. Performing Inquiry and Page of MCU/BT combination

      ii. Pair with MCU/BT combination

      iii. Send text data to MCU/BT combination

Appendix C

RS232 Pin Configuration

RS232 Pin No.	Signal Name	Description
1	DCD	Data Carrier Detect
2	RxD	Receive Data
3	TxD	Transmit Data
4	DTR	Data Terminal Ready
5	SGND	Signal Ground
6	DSR	Data Set Ready
7	RTS	Request To Send
8	CTS	Clear To Send
9	RI	Ring Indicator

Appendix D

Communication Log with Windigo Systems

Dialog with Quinton S.Q. Yuan

Q: I recently purchased a BTM02C2XX-R. I was wondering of you can provide me with the power supply specifications for it. I did not notice them on your website. What are the max current draws for the 3.3v and 1.8v supplies? What voltage range can the module handle for the two supplies?

A: Please be aware that 1.8V is output on our module, and 3.3V is input, since we have one 1.8V regulator on module, so this module requires single 3.3V power supply. For the 3.3V power supply, it ranges from 3.3V to 5V.

Q: What is the best way to mount it to a PCB? The data sheet says that it is surface mountable, but I noticed that there are no leads coming off of it. There are just divots. Should it be placed in a socket? If so, where can we get a socket?

A: Yes, it's surface mountable, you can just mount them like any standard SMD parts, say SMD type crystal. It's hard to find a socket.

Q: For the Bluetooth module integration, we are only using the UART interface along with the RF port. Where should the other pins be tied (3.3v, or GND)?

A: You can leave these unused pin as unconnected, however, suggest you route the SPI interface out to a connector or testing pad if you need to check/reconfigure the module. Also it's better to have access the reset pin; especially your host system can reset the module when necessary.

Q: Is there a more detailed datasheet available for the Bluetooth Module than the 5 page brief one that is on the Windigo Systems website? Where would I be able to find the HCL commands that the module recognizes.

A: As the HCI is not controlled by us, actually by CSR, so normally we didn't include the HCI implementation in our own documentation, here is the release note for the first release on BC02 firmware. *Document: *HCIStack1.1v14.3 Software Release Note March 2002*

Q: I was wondering at what temperature the BTM should be soldered at.

A: Should below 240 C Degree

Q: Is there a specified testing procedure to double check that the BTM is working properly without having it integrated with a micro?

A: In order to do that, you need Casira with a suitable fixture, then you can use either your own software or software from CSR to verify.

Q: What I am looking for is a way to see that the chips itself is not burnt out. If it is just powered, is it transmitting anything? is there a pin configuration that would have it emit specific info? Does it have something like a self-test I guess is what we are asking. Something where we can verify that the chip functions (to a degree) without have to use software.

A: For normal operation, PIO0 and PIO 1 will indicate transmit or receive individually, but for -R, there is no "built-in" self-test by using hardware switch, you have to send special command to enable these testing routing.

Appendix E

Communication Log with Robert Leskovec

Dialog with Robert Leskovec

Q: Andrew and I were wondering what you though of our PCB layout for the Bluetooth module for our project.

A: Neat work! I would imagine the output line might need to have "stripline" dimensions.

Also, it looks like a lot of pins get grounded!

The large ground plane will soak up the heat and make it hard to solder the pins. Unless there is some other consideration such as lead length, you will probably have and easier time of it if you cut back the ground area from the chip to beyond the lead length, and instead have "lines" coming inward to the chip from the surrounding ground plane for each lead. This way you only have to heat the line under each lead.

But ask some other people like Frank Merat, and ask Steve Garverick what may be the best way to handle this if you are making the board on his machine.

Also, since I first talked to you, I found a grad Student, Malcolm Wightman mxw52 has experience with laying out boards for surfaces mount and may have some thoughts. He is working in Glennan 708B.

Appendix F

Resource List

Bluetooth Resources
Mango Commerce
http://www.mangocommerce.com/btdesigner/department.cfm?dcode=74

Windigo Systems
http://www.windigosys.com/

CSR
http://www.csr.com/

Bluetooth.com
http://www.bluetooth.com/

Microcontroller Resources
GNU Development Chain for 68HC11&68HC12
http://stephane.carrez.free.fr/m68hc11_port.php

Mac OS X installer script for GNU Development Chain for 68HC11&68HC12
http://homepage.mac.com/rbarris/FileSharing1.html

Keyspan USB PDA Adpater
http://www.keyspan.com/products/usb/pdaadapter/

Technological Arts BDM POD module
http://www.technologicalarts.com/myfiles/ubdm12.html

PCB Resources
PCBExpress
http://www.pcbexpress.com

Other Resources
Docklight RS232 terminal software
http://www.docklight.de/

Antenna Design Resources

[i] "Wireless Data Acquisition using Bluetooth," M. R. Basheer, University of Missouri-Rolla.
[ii] "Analysis of Integrated Inverted-F Antennas for Bluetooth Applications," M. Ali, G. J. Hayes, Ericsson Inc.
[iii] "Analysis of Integrated Inverted-F Antennas for Bluetooth Applications," M. Ali, G. J. Hayes, Ericsson Inc.
[iv] "Analysis of Integrated Inverted-F Antennas for Bluetooth Applications," M. Ali, G. J. Hayes, Ericsson Inc.