Army Research Laboratory

Designing an Ultrasonic Modem for Robotic Communications

by Justin Shumaker

ARL-TR-4754 March 2009

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5066

ARL-TR-4754 **March 2009**

Designing an Ultrasonic Modem for Robotic Communications

Justin Shumaker
Vehicle Technology Directorate, ARL

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**			
1. REPORT DATE *(DD-MM-YYYY)* March 2009	**2. REPORT TYPE** Final		**3. DATES COVERED (From - To)** January 2008–March 2008
4. TITLE AND SUBTITLE Designing an Ultrasonic Modem for Robotic Communications			**5a. CONTRACT NUMBER**
			5b. GRANT NUMBER
			5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Justin Shumaker			**5d. PROJECT NUMBER** 1L162618AH80
			5e. TASK NUMBER
			5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: AMSRD-ARL-VT-UV Aberdeen Proving Ground, MD 21005-5066			**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-TR-4754
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)			**10. SPONSOR/MONITOR'S ACRONYM(S)**
			11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			
13. SUPPLEMENTARY NOTES			
14. ABSTRACT An ultrasonic modem is a device that can transmit information acoustically. One was designed for the 2008 FIRST Robotics League competition to provide short-range communications with the robot. The creation of this device was the result of a regulation that inhibited any use of electromagnetic-based communications technology in the competition. The ultrasonic modem consists of easily attainable and low-cost parts and was designed to attenuate ambient noise in the audible frequency domain and reliably send a 2-bit message at a rate of 10 Hz or higher. Each message referred to a high-level control command that the robot would perform.			
15. SUBJECT TERMS ultrasonic, modem, communications, robot, audio, sound			

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Justin Shumaker
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UU	26	**19b. TELEPHONE NUMBER** (*Include area code*) 410-278-2834

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

1. Introduction

A modem consists of a modulator and demodulator that transmit and receive data, respectively. The impetus for building such a device stemmed from the 2008 FIRST Robotics League (FRL) competition rules that stated only acoustic and electromagnetic (EM) frequencies ranging from infrared to ultraviolet could be utilized to communicate with the robot. Common inexpensive ultrasonic piezoelectric transducers range in frequency from 24 to 40 KHz. A typical 40-KHz transducer requires 10 or more oscillations to reach peak resonating amplitude.[1] This restriction implies a theoretical maximum of 4000 bits/s in the case of a 40-KHz transducer. Unfortunately, additional constraints exist that prohibit data transfer at 40 KHz. Dependent upon the environment, signal attenuation can severely limit the device's maximum bandwidth. An oscilloscope can provide the necessary insight to develop a rather inexpensive ultrasonic modem for reliably transferring data in an environment saturated with audible acoustic noise up to a distance of 25 m.

2. Previous Work

The vast majority of acoustic modem work has been for underwater autonomous vehicles. This is primarily due to poor EM wave propagation in water and the need for long-range communications in the ocean. Acoustic waves travel farther and faster in water than they do in air. The speed of sound in water is 4.4× faster than in air. The speed of sound is a function of material density ρ and stiffness C. As stiffness increases, so does the speed of sound.[2]

$$c = \sqrt{\frac{C}{\rho}} \,. \tag{1}$$

Acoustics in water, known as hydroacoustics, is a more desirable communications medium than in air. For these reasons, acoustic modems designed for transferring data in air have never become mainstream. As a result, there were no inexpensive commercial off-the-shelf ultrasonic modems that could have been purchased at that time. A great deal of work has been focused on data modulation and demodulation over twisted pair phone lines, but only a small fraction of that work is applicable for ultrasonic modems. This is mainly due to the piezoelectric element that generates and detects the audio at resonant frequency.

[1]Maxbotix, Inc. MaxSonar-UT High Performance Ultrasonic Transducer Datasheet, Tucson, AZ, July 2007.

[2]Wikipedia. http://en.wikipedia.org/wiki/Underwater_sound#Sound_waves_in_water (accessed October 2008).

3. The Transmitter

This ultrasonic modem was designed as a unidirectional communications device. Both the transmitter and receiver utilized a Microchip Technologies, Inc., 18F1320 programmable integrated circuit (PIC) microcontroller.[3] The transmitter device had four buttons that each mapped to a unique 2-bit message. Each message was transmitted using a modified RS-232 protocol. The most commonly used RS-232 protocol consists of 1 start bit, 8 data bits, and 1 stop bit. With only four possible messages, the modified RS-232 protocol consists of 1 start bit, 2 data bits, and 1 stop bit. Each bit consists of 50 square wave cycles at a frequency of 40 KHz to ensure peak resonating amplitude from the ultrasonic transducer. A value <50 but >10 may have worked equally well. After each bit is transmitted, a 7-ms delay is inserted to cope with attenuation in the environment. This was determined empirically using a Tektronix TDS 2002B oscilloscope (see figure 1). Further discussion on the delay is found in the Receiver section.

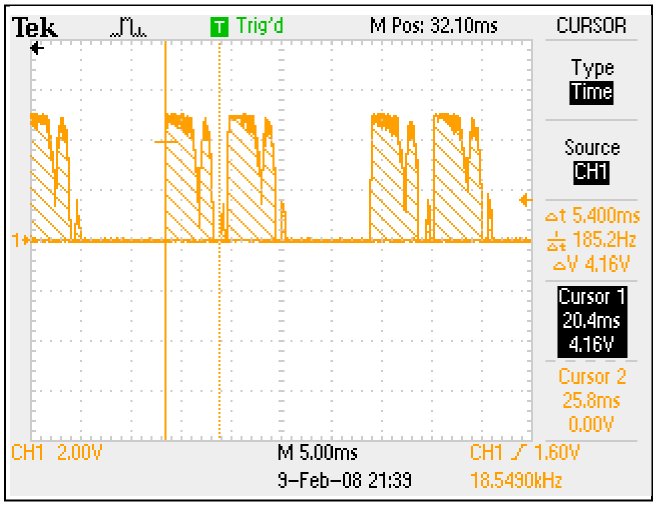

Figure 1. Time length of 1 message bit.

In order to deliver the peak amplitude from the transducer, both the frequency and voltage requirements must be satisfied. The transmitter was designed to operate at 5 V; however, the transducer is most powerful at 60 V. One could design a voltage multiplier such as a Buck-Boost or Walton-Cockroft circuit into the transmitter device or utilize some form of external amplifier. To minimize time and complexity, a Radio Shack Optimus brand of stereo amplifier was used. The amplifier had a peak output of 80 V. The output of the transmitter device was connected to the input of the amplifier. The output of the amplifier was connected to a transducer with 60-V output.

[3]Microchip, Inc. PIC18F1220/1320 Datasheet, 18/20/28-Pin High-Performance, Enhanced Flash MCUs With 10-bit A/D and nanoWatt Technology, Chandler, AZ, 2007.

The transmitted data generates harmonics that can be heard in the form of a "ticking" sound. This "ticking" sound is a function of the rate at which messages were being transmitted. The audible sound is not the 40-KHz carrier wave but the starting and stopping of the transducer for each transmitted bit. The circuit schematic, board layout, and assembled transmitter can be seen in figures 2–4, respectively. The source code is available in appendix A.

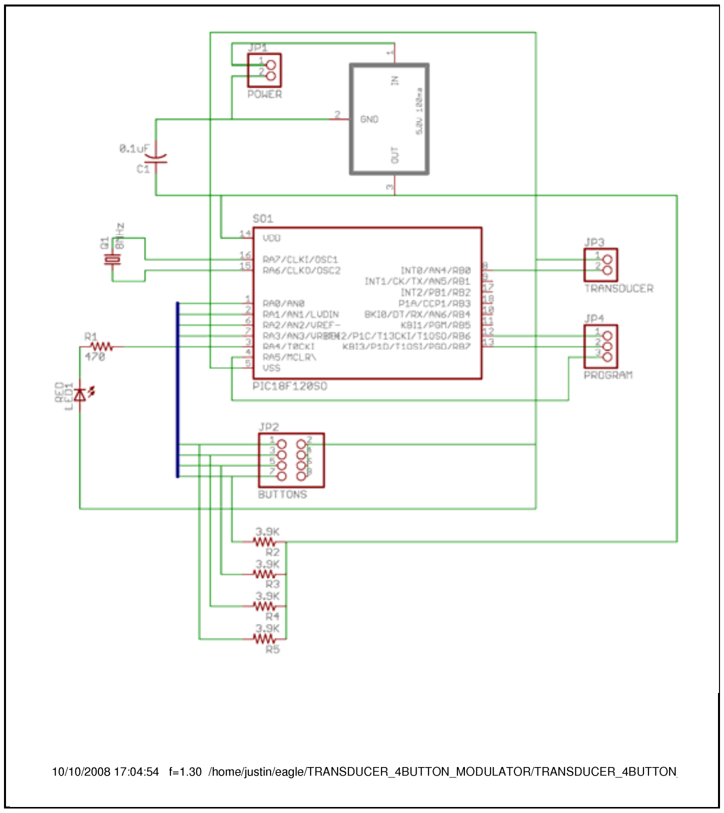

Figure 2. Transmitter schematic.

4. The Receiver

The receiver or demodulator is responsible for transforming potential differences generated by the receiver's transducer into binary 1's and 0's. The most laborious parts of designing the receiver were the demodulation algorithm and operational amplifier (op-amp) circuit. The demodulation algorithm had to take into account much of the acoustic phenomena so as to prevent wave cancellation due to acoustic attenuation. It was very important to sample the incoming signals in a manner that did not lend to the detection of false positives. Additionally, without a properly designed op-amp circuit, the signal may be overly attenuated or contain undesirable noise (see figure 5).

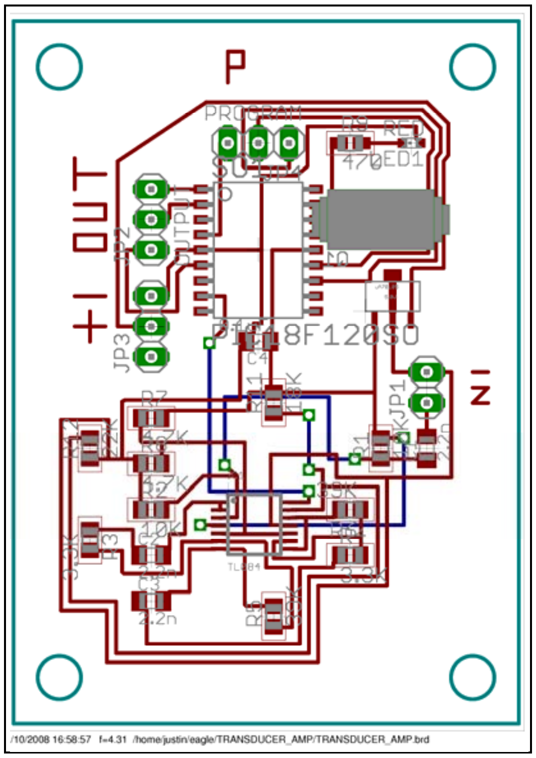
P
PROGRAM
OUT
+|
JP2
JP3
OUTPUT
PIC18F1220SO
470
LED1
RED
JP1
IN
C4
TL084
2.2n
10K
3.3K
/10/2008 16:58:57 f=4.31 /home/justin/eagle/TRANSDUCER_AMP/TRANSDUCER_AMP.brd

Figure 3. Transmitter board layout.

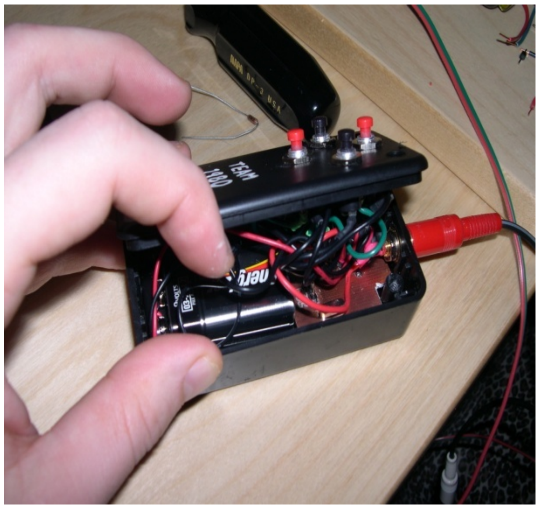

Figure 4. Picture of assembled transmitter.

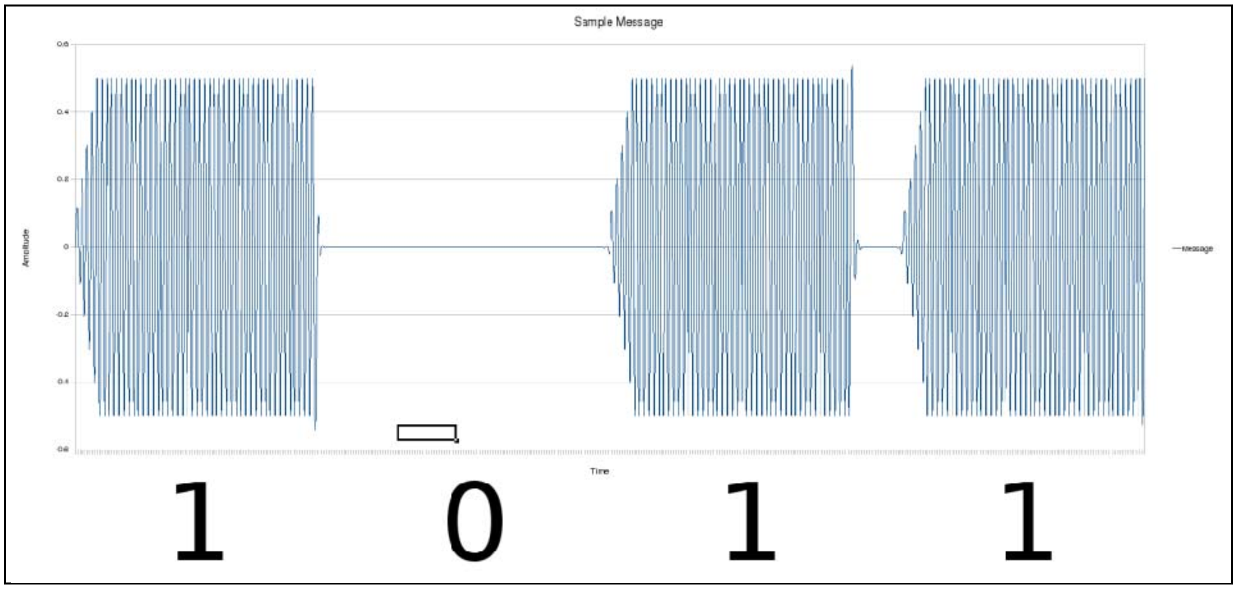

Figure 5. Ideal message waveform.

A Simulation Program for Integrated Circuit Emulation (SPICE) is often a necessity when designing a robust operational amplifier circuit. The SPICE package used was MINDI, which is available for free from Microchip Technology, Inc. From a high-level perspective, this circuit was required to amplify the potential difference from the ultrasonic transducer and produce a binary high or low value. A circuit was designed using four operational amplifiers with a first-stage unity gain, a dual-stage high-pass filter with gain, and a final stage with voltage clamping. The MCP6004, which is a quad input rail-to-rail operational amplifier, was used to handle the aforementioned tasks.

Before any simulation was performed, the ultrasonic transmitter was activated while an ultrasonic transducer had been wired up to an oscilloscope probe to measure the voltage profile. It was realized that the voltage ranged from 100 mV at a distance of 3 m to <5 mV at a distance of 25 m. The amplification from the op-amp circuit had to be on the order of 100 in order to detect a signal at a distance of 25 m. Because it was undesirable to have a gain of more than 40 on this particular op-amp,[4] two stages were necessary to achieve a desired gain of 144 (see figure 6). The transient graph in figure 7 shows the output from each stage of the op-amp circuit over a period of 400 µS. The input to the op-amp was a 10-mV sinusoidal voltage source, and the order of stages according to the legend is X1-out, X2-out, X3-out, and X-4 out, respectively.

[4] Microchip, Inc. Microchip MCP6001/1R/1U2/4 Datasheet - DS21733G, Chandler, AZ, 2007.

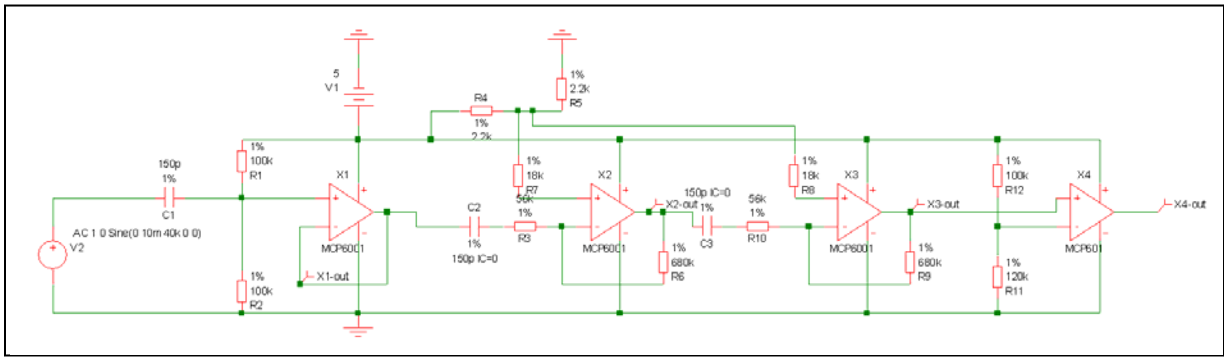

Figure 6. MINDI schematic.

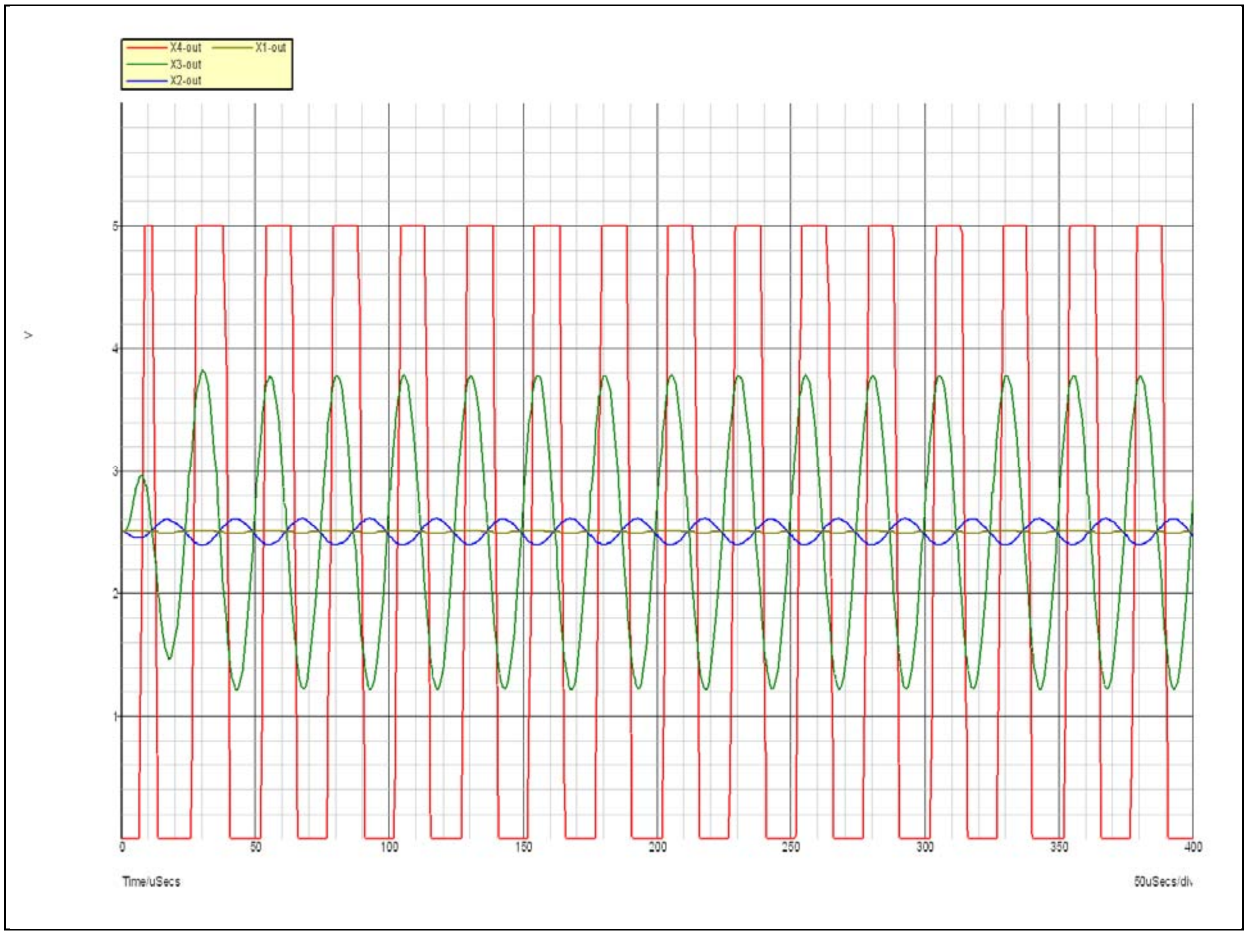

Figure 7. MINDI simulation output.

The receiver circuit board incorporated the op-amp circuit and an 8-bit 18F1320 PIC chip. The output of the op-amp circuit was wired up to an external interrupt on the PIC microcontroller. When the op-amp would output a high signal, the interrupt service routine (ISR), a function callback, would execute and some timing measurements would occur. The purpose of the ISR was to analyze the signal timing for valid messages. It was during the first several communications tests that the signal attenuation was quite problematic. The oscilloscope measured ~5.4 ms of transient signal attenuation. This measurement lead to the decision to insert a 7-ms delay between bits to allow the signal to fully attenuate. Once this adjustment was made, the oscilloscope was configured to overlay the binary output of the op-amp and the raw signal from the transducer (see figure 8).

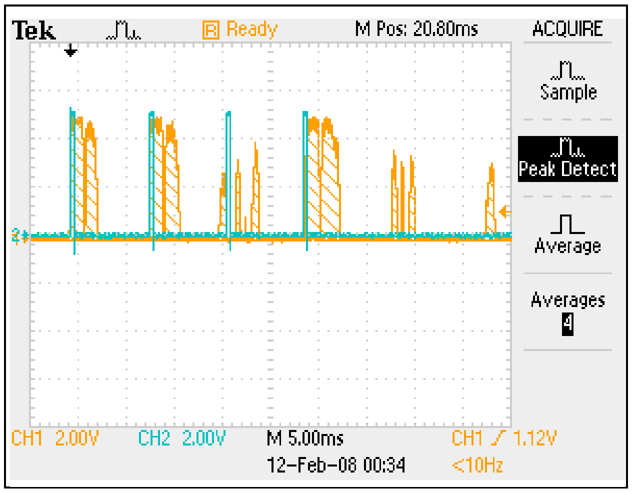

Figure 8. Overlay of binary interpretation to analog data captured.

The receiver software involves two software timers and one external interrupt. The first timer was used to measure the length of time between bits. The second timer was used pull the output pins low because as long as the output pins were held high, the robot would interpret those pins as an active message. For example, if the appropriate output pins were held high to indicate a "move forward" message was received, then the robot would move forward. As one can see, it is desirable to have those pins expire and go low in case the transmitter could no longer communicate with the robot. The external interrupt would measure the number of pulses received during each bit. Once the first interrupt of a new message occurred, timer1 would become activated. Each subsequent interrupt would increment a counter. After the timer1 expired, the counter would be processed. During each transmitted bit, a maximum of 16 pulses can be detected. If 14 of those 16 pulses were detected, an 87.5% acceptance rate, then the bit is marked as valid, otherwise the bit was discarded. The core of the message processing logic occurs in the timer1_isr function. This process would occur 4 times for a complete message. This accounts for the start bit, 2 data bits, and the stop bit. Upon receiving the stop bit, the counters would be reset and standby for the start bit of the next message to occur.

Upon receiving a valid message, one of four output pins would be set to high. This made it straightforward to wire up the receiver to the microcontroller on the robot to detect which of four possible messages occurred and respond accordingly. The four messages were programmed for moving forward, stopping, rotating left, and rotating right. These four commands made it possible to drive the robot around a circular track and score points. The circuit schematic and board layout can be seen in figures 9 and 10, respectively. The source code is available in appendix B.

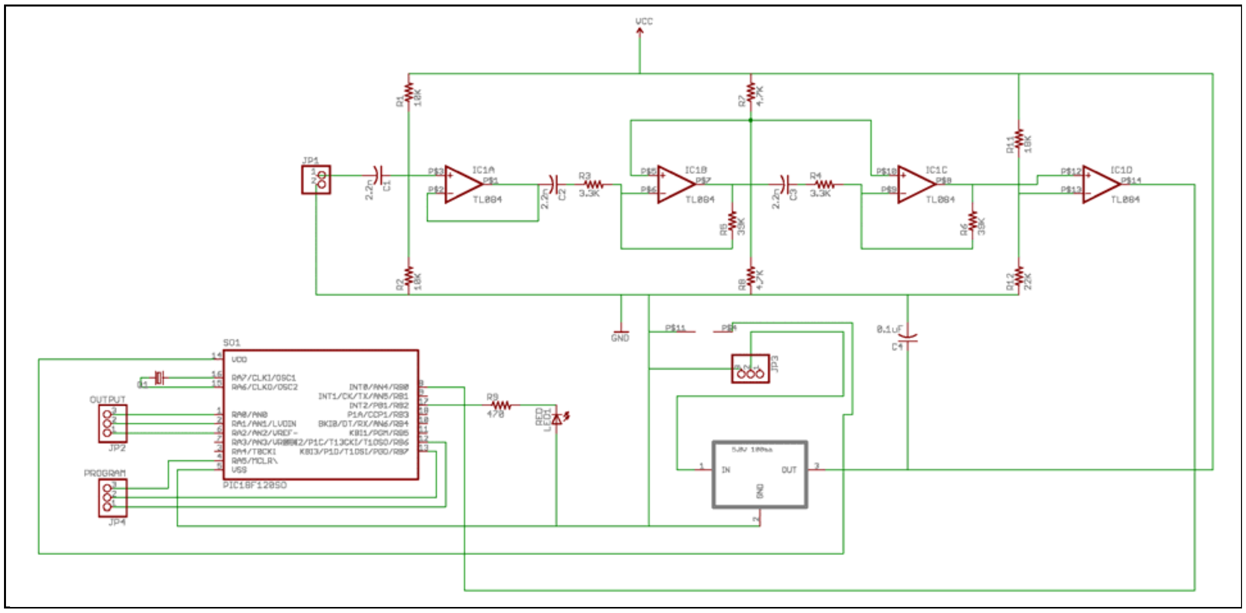

Figure 9. Receiver schematic.

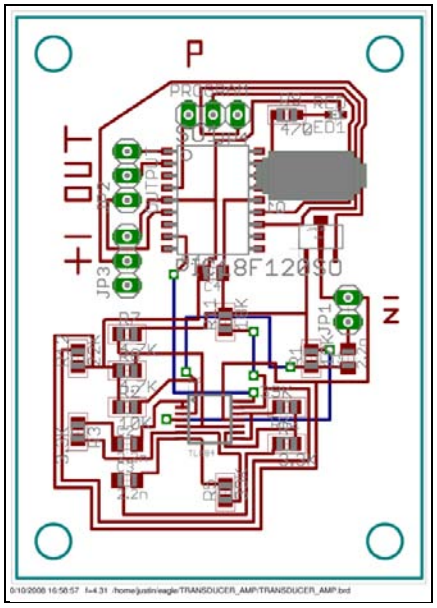

Figure 10. Receiver board layout.

5. Conclusion

The acoustic modem was tested on the Aberdeen High Schools 2008 FRL robot with positive results. The overall cost in parts for both the transmitter and receiver was <$15. Benchmarking the modem revealed a communications accuracy of 100% at a distance of 25 m with a direct line of sight. Even without a direct line of sight, the receiver would receive messages because the sound would bounce off of walls as well as the floor. While this form of communications device is rather crude when compared to modern EM communications devices, it still has useful applications. An EM communications device of similar size and mass can range from 1 to several orders of magnitude higher throughput. If a communications device is needed for use with equipment that is susceptible to EM radiation of any kind, it may be apropos to use such a device. Applications may include communications for EM-sensitive laboratory equipment, security systems, and equipment requiring stealth from EM detection.

INTENTIONALLY LEFT BLANK.

Appendix A. Transmitter Source Code

This appendix appears in its original form, without editorial change.

```
#include <18F1320.h>
#DEVICE ADC=10
#fuses HS,NOWDT,NOLVP,PUT,NOMCLR
#use delay (clock=8000000)

#define  PIN_LED  PIN_A4


void
transmit (int bit1, int bit2)
{
  int i;

  for (i = 0; i < 50; i++)
  {
    output_high (PIN_B0);
    delay_us (10);
    output_low (PIN_B0);
    delay_us (10);
  }
  delay_ms (7); /* This is done to minimize attenuation */


  if (bit1)
  {
    for (i = 0; i < 50; i++)
    {
      output_high (PIN_B0);
      delay_us (10);
      output_low (PIN_B0);
      delay_us (10);
    }
    delay_ms (7); /* This is done to minimize attenuation */
  }
  else
  {
    delay_ms (8);
  }


  if (bit2)
  {
    for (i = 0; i < 50; i++)
    {
      output_high (PIN_B0);
      delay_us (10);
      output_low (PIN_B0);
      delay_us (10);
    }
    delay_ms (7); /* This is done to minimize attenuation */
  }
  else
  {
    delay_ms (8);
  }
  for (i = 0; i < 50; i++)
  {
    output_high (PIN_B0);
    delay_us (10);
    output_low (PIN_B0);
    delay_us (10);
  }
  delay_ms (7); /* This is done to minimize attenuation */
}


void
main ()
{
  output_low (PIN_LED);

  while (1)
  {
    if (!input (PIN_A0))
    {
      transmit (0, 0);
    }
    else if (!input (PIN_A1))
    {
      transmit (0, 1);
    }
    else if (!input (PIN_A2))
    {
      transmit (1, 0);
    }
    else if (!input (PIN_A3))
    {
      transmit (1, 1);
    }
    else
    {
      output_low (PIN_B0);
    }
    /*
    * It's best to have a down time equal to the transit time since
    * the demodulator could possibly pick up the 4th bit as the first bit
    * which means that if the next start bit occurs in under 24 ms (length of 3 bits)
    * then that next start bit would be seen as a stop bit thus producing (1001).
    */
    delay_ms (32); /* (4) 8ms bits + 32ms rest = 64ms per mesg = ~15.5625 msgs/sec */
  }
}
```

INTENTIONALLY LEFT BLANK.

Appendix B. Demodulator Source Code

This appendix appears in its original form, without editorial change.

```
#include <18F1320.h>
#DEVICE ADC=10
#fuses H4,NOWDT,NOLVP,PUT,NOMCLR
#use delay (clock=8000000)

#define  PIN_LED          PIN_B2

#define MODE_LISTEN       0
#define MODE_CAPTURE      1
#define MODE_SLEEP        2

int8 mode;
int16 bits[4];
int8 bits_ind;


#INT_EXT
void
ext_isr ()
{
  if (mode == MODE_LISTEN)
  {
    mode = MODE_LISTEN;
    if (!bits[0])
    {
      /*
      * Possible beginning of new message.
      * Set Timer1 to listen for 10 pules (25us per pulse).
      * So, listen for 400us and see if 14 or more pulses come in (400/25 = 16 max pulses).
      * Each tick of timer1 accounts for (8 / 8,000,000) = 1us.
      * So, set timer1 to 65,535 - 400) = 65,135.
      */
      set_timer1 (65135);
output_high (PIN_B6);

      /*
      * Clear TMR1IF (Timer1 overflow Interrupt Flag) so that
      * the timer1_isr () is not called immediately after enabling the interrupt.
      */
      bit_clear (*0xF9E, 0);
      enable_interrupts (INT_TIMER1);
    }
    bits[0]++;
  }
  else if (mode == MODE_CAPTURE)
  {
    bits[bits_ind]++;
  }

// output_low (PIN_LED);
}


#INT_TIMER0
void
timer0_isr ()
{
  output_low (PIN_A0);
  output_low (PIN_A1);
  output_low (PIN_A2);
  disable_interrupts (INT_TIMER0);
}


#INT_TIMER1
void
timer1_isr ()
{
  if (mode == MODE_LISTEN)
  {
    /* TEST_XXX - Set timer1 to just wait for 9ms to wait out the rest of the bits and go into listen mode again. */

    /* Now that the 400us time limit as expired, check to see that 7 or more pulses were detected in bit[0]. */
    if (bits[0] >= 14)
    {
output_low (PIN_B6);
      /* GOOD START BIT, PROCEED */
      bits_ind = 1;
      mode = MODE_SLEEP; /* Sleep for 8ms and capture next bit (first data bit), alignment is now at the center of
the bit */
      set_timer1 (57735); /* Sleep 7.8ms - because there is an additional 200us lag between start bit and first data
bit */
    }
    else
    {
      /* BAD, MUST BE NOISE */
      mode = MODE_LISTEN; /* Go back into MODE_LISTEN for a start bit. */
      disable_interrupts (INT_TIMER1);
      bits[0] = 0;
    }
  }
  else if (mode == MODE_CAPTURE) /* End of Capture Cycle */
  {
output_low (PIN_B6);
    mode = MODE_SLEEP;
    bits_ind++; /* increment for bits[d0..d1..stop] */
    if (bits_ind == 4)
    {
      /* Done Capturing data, wait 24ms for good measure to ensure MODE_LISTEN resumes during the 32ms low period. */
      set_timer1 (41535);
    }
    else
    {
      /* For both data bits */
      set_timer1 (57735); /* Because sampling occurred over 400us only wait 7600us (400us + 7600us = 8ms) to stay on
8ms intervals */
    }
  }
  else if (mode == MODE_SLEEP) /* End of Sleep Cycle */
  {
    if (bits_ind >= 1 && bits_ind <= 3)
    {
      mode = MODE_CAPTURE;
output_high (PIN_B6);
      set_timer1 (65135); /* 400us to capture 14 or more pulses */
    }
    else
    {
      /* END OF 4-BIT CAPTURE - TIME TO EVALUATE THE BITS */
      /* If start bits are ok then output high on appropriate pins */
      if (bits[3] >= 12) /* bits[0] (start bit) is already known to be good */
      {
        if (bits[1] >= 12 && bits[2] >= 12)
        {
          /* MESG: 11 - OUTPUT: 100 */
          output_low  (PIN_A0);
          output_low  (PIN_A1);
          output_high (PIN_A2);
        }
        else if (bits[2] >= 12)
        {
          /* MESG: 01 - OUTPUT: 010 */
          output_low  (PIN_A0);
          output_high (PIN_A1);
          output_low  (PIN_A2);
        }
        else if (bits[1] >= 12)
        {
          /* MESG: 10 - OUTPUT: 011 */
          output_high (PIN_A0);
          output_high (PIN_A1);
          output_low  (PIN_A2);
        }
        else
        {
          /* MESG: 00 - OUTPUT: 001 */
          output_high (PIN_A0);
          output_low  (PIN_A1);
          output_low  (PIN_A2);
        }
      }

      output_toggle (PIN_LED);

      set_timer0 (0);
      bit_clear (*0xFF2, 2); /* Clear Timer0 interrupt flag bit */
      enable_interrupts (INT_TIMER0); /* Begin 32.768ms countdown */


      bits[0] = 0;
      bits[1] = 0;
      bits[2] = 0;
      bits[3] = 0;

      mode = MODE_LISTEN;
      disable_interrupts (INT_TIMER1);
    }
  }
}


void
main ()
{
  int8 i;

  for (i = 0; i < 10; i++)
  {
    output_high (PIN_LED);
    delay_ms (100);
    output_low (PIN_LED);
    delay_ms (100);
  }

  bits[0] = 0;
  bits[1] = 0;
  bits[2] = 0;
  bits[3] = 0;
  mode = MODE_LISTEN;

  ext_int_edge (0, H_TO_L);
  enable_interrupts (INT_EXT);

  setup_timer_0 (RTCC_INTERNAL | RTCC_DIV_8); /* Used to maintain most recent output for up to 65.535 ms */
  setup_timer_1 (T1_INTERNAL | T1_DIV_BY_8);
  enable_interrupts (GLOBAL);

  while (1)
  {
  }
}
```

NO. OF COPIES	ORGANIZATION
1 (PDF only)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA 8725 JOHN J KINGMAN RD STE 0944 FORT BELVOIR VA 22060-6218
1	DIRECTOR US ARMY RESEARCH LAB IMNE ALC HR 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB AMSRD ARL CI OK TL 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB AMSRD ARL CI OK PE 2800 POWDER MILL RD ADELPHI MD 20783-1197

ABERDEEN PROVING GROUND

1	DIR USARL AMSRD ARL CI OK TP (BLDG 4600)