BEE HIVE TEMPERATURE AND SOUND MONITOR

A design project report presented to the Engineering division of the Graduate school of Cornell University in partial fulfillment of the requirements for the degree of Master of Engineering (ECE)

By

Madhuri Kandepi (mk2335)

Project Advisor: Dr. Bruce Land

Degree Date: May 2015

Abstract

Master of Electrical Engineering Program
Cornell University
Design Project Report

Project title: Bee Hive Temperature and Sound Monitor

Author: Madhuri Kandepi

Abstract:

The goal of the project is to provide a solution for remote monitoring of a beehive to monitor the temperature and acoustics. A transmitter module is installed at the beehive which measures the temperature and audio levels (using temperature sensor and microphone) and transmits the information through a radio. A receiver module is installed inside the house (or any target location) which receives the input and displays the same on a graphic LCD. The communication between the transmitter and the receiver happen using XBEE Pro modules (Radio communication). The hive data collection unit is solar powered and weather-proof.

Executive summary

The project involves the design and implementation of a telemetry system to monitor the temperature and sound intensity in a Beehive. The primary requirements for the system are that the hive data collection unit should be solar powered, weather-proof, have temperature probes and microphones compatible with hive conditions, cost to be less than $500, and ease of use.

The system design consists of a temperature sensor to detect the temperature in a hive, a microphone and necessary circuitry to measure the sound inside the beehive, two Microcontroller units each at the source (Beehive) and destination (Beehive owner's house), two XBEE Pro radios for the wireless data transfer between the source and destination, a graphic LCD to display the data received from the hive, a solar panel and a 12V 7AH battery to make the hive data collection unit solar-powered.

In order to achieve the objective of the project, necessary hardware and software components are designed. To measure the temperature in the hive, a temperature sensor, which is calibrated directly in º F, is used. An electret microphone and an op-amp are used to measure the sound. Each microcontroller unit handles its own dedicated task. The microcontroller unit at the source handles data acquisition and transmission of the acquired data through a RF radio. The microcontroller unit at the destination handles data reception and display of the data. A graphic LCD is used to display the data. Log of the beehive data is also displayed on the LCD, in order to facilitate the user to know and compare the data at different time instants i.e., to facilitate the user to know about both the absolute as well as relative data.

The planning and implementation of the project has gone well and smoothly. Work was equally split between the two semesters. The system designed was able to achieve satisfactory results. The final deliverable is fairly compact and solid. The cost of the project is also well within the limits.

Contents

1. Introduction

Beekeeping is a production branch of the agriculture. Honeybees are very important economical insects not only for pollination of crops, but also for their valuable products. These products are used either directly as human food (honey) or as a raw material for an impressive number of medicinal, cosmetic, pastry produces, etc.

In apiculture, one of the major problems is to monitor honeybee colonies for their health, population, and other biological activities. Beside honeybee diseases, one of the most common problems in apiculture is colony losses because of unusual alteration in environmental conditions such as temperature, humidity etc.

In the previous 20th century, when information technologies were not widely available, measurements of the bee colony temperature were done manually using thermocouples and later special temperature loggers or iButtons. With the rapid development of IT, it has become possible to develop whole sensor networks for temperature measurements.
This project aims to monitor the hive conditions; temperature and sound and gives the opportunity for beekeepers to monitor their own bee colonies in real time at a low cost. With the help of a display device which will be installed in their house, the bee keepers could know what is going on in the bee-hive from the comfort of their home.
One of the main goals of this project is simplicity. The system should be reliable, easy to understand, easy to use, low cost, solar powered, and the results should be accurate. The system designed meets all of these requirements.

A user manual is provided in Appendix to show how to get a graphical interpretation of collected data.

2. Design problem & System Requirements

2.1 Design Problem

The goal of this project is to design a low cost, compact and reliable temperature and sound monitor for monitoring the hive conditions. The hive data collection unit should be solar powered and placed in weather proof boxes. A simple graphical display should be available to display the collected data at the destination. The system is composed of two sub-systems and they are transmitter and receiver. Transmitter is responsible in collecting data (temperature and sound) from the beehive, arranging data into packet, and sending the packet to the receiver. Receiver is responsible in receiving packet and displaying the data on the graphical LCD. Finally, data is analyzed in some graphs.

2.2 System-level Requirements

2.2.1 Bee Hive Temperature and Sound Monitoring System (BHTSM)

This section lists the requirements to be met by the Beehive temperature and sound monitoring system (BHTSM).
[SYS-01] The BHTSM system shall include Transmitter module to relay temperature and sound
Note: The requirements for the Beehive Transmitter are listed in Section 2.2
[SYS-02] The BHTSM system shall include receiver module that receives and displays hive temperature and acoustics.
Note: The requirements for the receiver module are listed in Section 2.3
[SYS-03] The BHTSM system shall read the beehive temperature.
[SYS-04] The BHTSM system shall read the beehive acoustics details.
[SYS-05] The BHTSM system shall wirelessly transmit information from hive to the destination.
[SYS-06] The BHTSM system shall indicate if the data is out of range.
[SYS-07] The bill-of-materials for the BHTSM system shall not exceed $500.
[SYS-08] The BHTSM system shall be designed to be weather-proof (outdoor beehive).
[SYS-09] The BHTSM system shall be designed considering power saving options.
[SYS-10] The BHTSM system shall work with a range of 1500 meters.
[SYS-11] The BHTSM system shall read temperatures in °F (Degrees Fahrenheit).
[SYS-12] The BHTSM system shall measure temp with (+/-)1 degree accuracy.
[SYS-13] The BHTSM system shall read accurate sound intensity levels in the hive.

2.2.2 Beehive Transmitter

This section lists the requirements to be met by the Beehive Transmitter (BHT).

[BHT-01] The BHT shall read temperature and sound inside the beehive in periodic intervals.
[BHT-02] The BHT shall consume less power.

[BHT-03] The BHT shall wirelessly transmit temperature details.
[BHT-04] The BHT shall wirelessly transmit acoustics details.
[BHT-05] The BHT shall be solar powered.

2.2.3 Beehive Receiver

This section lists the requirements to be met by the Beehive Receiver (BHR).

[BHR-01] The Beehive Receiver (BHR) shall receive wireless/radio inputs.
[BHR -02] The BHR shall understand and parse the received beehive temperature information
[BHR -03] The BHR shall understand and parse the received beehive acoustics information
[BHR -04] The BHR shall display the received temperature in °F.
[BHR -05] The BHR shall graphically display the hourly temperature history.
[BHR -06] The BHR shall display the received sound intensity information in dB (Decibels).
[BHR -07] The BHR shall graphically display the hourly sound level history.

3 RANGE OF SOLUTIONS

The core requirement can be achieved by using multiple options including wired and Wi-Fi based communication. But the following solution had been chosen for its simplicity, reliability and cost effectiveness.

The chosen solution at a high level includes two major components, Beehive transmitter and Receiver. Transmitter is responsible for reading hive conditions and transmitting necessary data. Receiver is responsible for receiving the transmitted data and display the analysis. XBee Pro radios are responsible for the communication between these 2 components. Radio communication is easy to build, reliable and cost effective.

The solution also uses various hardware components including microcontrollers, temperature sensor, acoustics measurement circuitry, a solar panel, a 12V 7 AH battery, a graphical LCD to display the information and a bunch of other components like resistors and capacitors.
This section these majors components of the system are introduced.

3.1 Microcontroller

Atmel AVR microcontroller (ATmega 1284p) is chosen for this project because I am relatively familiar with it. It has been used in ECE4760-Microcontroller class which I had taken during fall semester and general technical support is available from Dr. Land or various websites. It is a good fit for the project as it has 8 built-in ADC (Analog to Digital Conversion) channels, Built-in USART, Interrupts & Timers, Sufficient Program and Data memory, Various Sleep Modes, Suitable frequency etc. and is low in cost.

3.2 Temperature Sensor

For measuring the temperature, LM34DZ sensor was used because it is calibrated directly in degrees Fahrenheit, linear +10.0 mV/˚F scale factor, suitable for remote applications, Low cost due to wafer-level trimming, operates from 5 to 30 volts, low self-heating, 0.18˚F in still air etc.

3.3 Sound Sensor

For measuring the sound inside the beehive an electret microphone and LM358 op-amp are used along with a bunch of other resistors and capacitors. The circuitry is explained in detail in the hardware section of the report.

3.4 XBEE Pro Radio

For the communication between Transmitter and Receiver, XBE Pro Radios are used. This is because it has the radios have an outdoor/RF Line-of-Sight Range Int’l 5000 ft (1500 m), serial Data Rate 1200 bps - 1 Mbps, fast 250 kbps RF data rate to the end node, 2.4 GHz for worldwide deployment, sleep modes supported for extended battery life.

3.5 Graphic LCD

For displaying the data at the receiver end, graphical LCD KS0108 (128X64) is used. This is because, it allows full graphical control compared to a regular 16x2 LCD. Also, it suits the project requirement of showing the data graphically at the receiver. It is the most commonly used graphical LCD and has good online support for header files and user manuals on how to use it. It also has a low power white LED back-light to help the user view the screen clearly.

3.6 Solar Panel

The Solar panel used in the project is a 2 Watt, 12V battery maintainer panel. As the battery being used in the project is a 12V battery, it serves the purpose. It can easily be connected and disconnected to the battery. No direct sunlight is needed and the panel works even in cloudy conditions. It is overcharge and discharge protected. Also, it is lightweight and compact in design. It provides a voltage between 16 and 23 Volts DC in full sun.

3.7 Battery

The battery used in the project is a 12V 7AH battery. It is sealed and maintenance-free. The battery has a long service life; under normal operating conditions it works five to six years in standby applications or between 200 and 1000 charge/discharge cycles depending upon depth of discharge. It is resistant to shock, vibration, chemicals, and heat. It is compact. It operates over a temperature range of -76°F to +140°F. All these features make it very suitable for the project.

4 DESIGN

4.1 Approach

The project is divided into two parts: The Transmitter Circuit and The Receiver Circuit.

The data flow of the project is linear. First, the temperature and sound intensity information is read from the sensors. The data is then fed to two separate ADC (Analog to Digital Converter) channels in the transmitter MCU which samples the signals at 8KHz and converts them into digital signals. Then, the digital data is converted back to meaningful information and is transmitted to the Transmitter XBEE Pro Radio through USART. The Transmitter radio transmits the information to the Receiver XBEE Pro Radio in the form of packets using wireless communication. The Receiver radio receives the packets and then sends this information to the Receiver MCU using USART. The Receiver MCU analyses the data and displays it on the graphical LCD interfaced to it.

The flow is shown in the below block diagram.

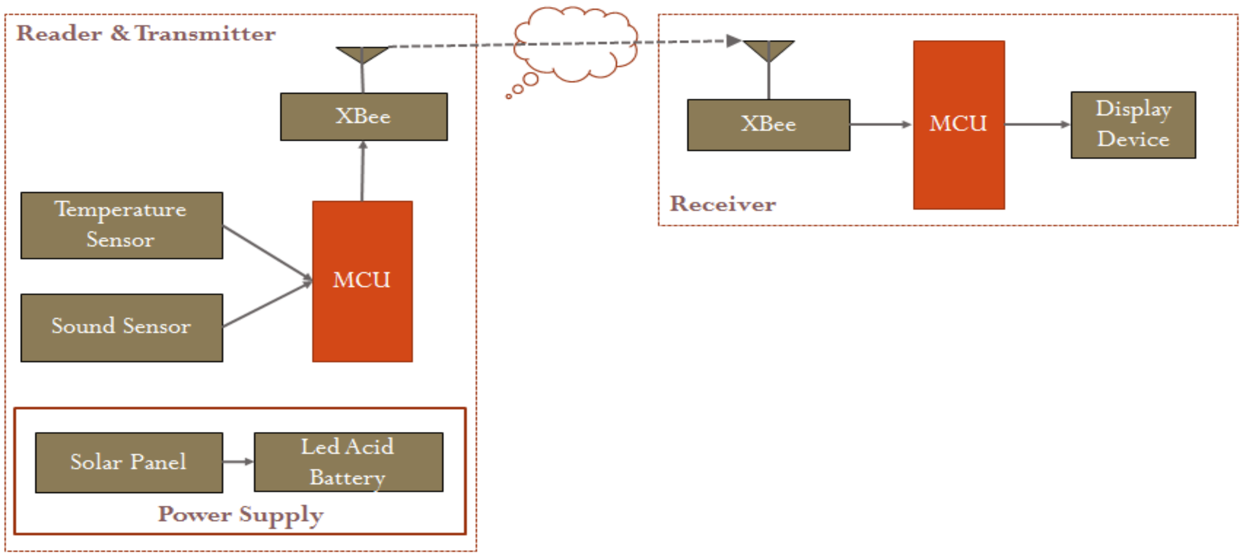


Figure 1 Block Diagram

4.2 Hardware

The hardware is split into two main parts: The Transmitter Circuit and the Receiver Circuit. Both these circuits are explained in detail in this section.

4.2.1 Beehive Transmitter

Beehive transmitter circuit is responsible for reading and transmitting hive conditions. Its schematic diagram is below.

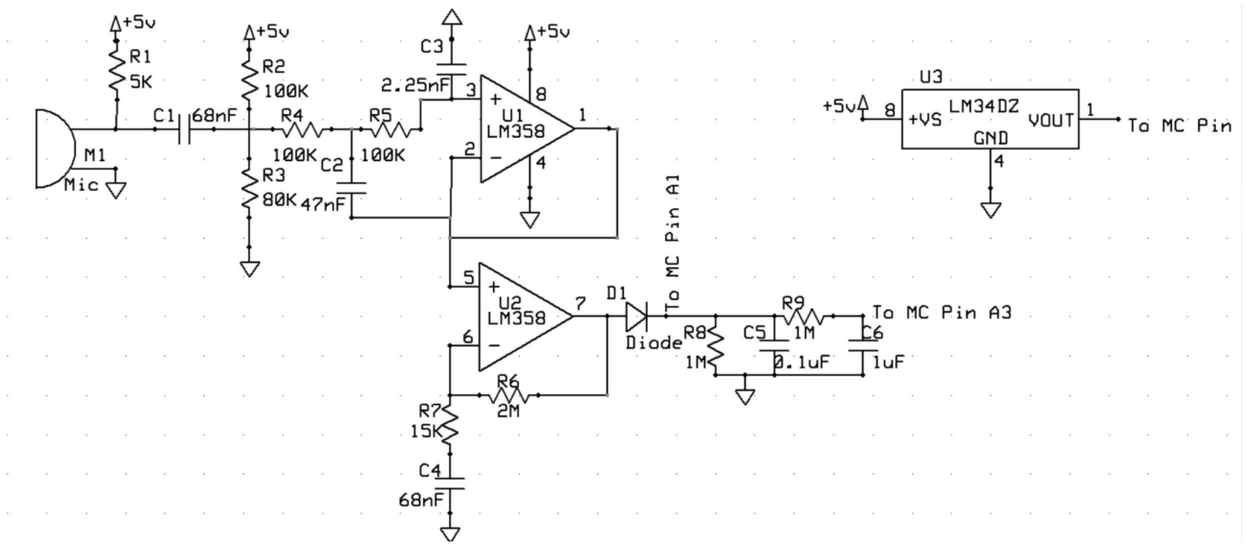


Fig 2: Schematic Diagram for Beehive Transmitter Circuit

The Transmitter circuit mainly consists of Transmitter MCU, Temperature sensor, Acoustics measuring circuitry, Transmitter radio.

Once built and tested, I built a PCB for the transmitter using EAGLE software. Then I ordered the printing of PCB online at Advanced Circuits (**4pcb.com)** by uploading necessary gerber files. Refer to "Appendix E" for detailed information on PCB Transmitter board details.

4.2.1.1 Transmitter MCU (ATmega 1284P)

The transmitter MCU is responsible for reading the information from the sensors, operating on the information read, transform the same to user understandable information and transmitting it through USART to Transmitter XBEE radio. This MCU has a temperature sensor, electret microphone, operational amplifier, and a XBEE Radio (Transmitter Radio) interfaced to it. The MCU uses specific algorithm to transmit the information. This is explained in detail in the software section below.

4.2.1.2 Temperature Sensor (LM34DZ)

The temperature sensor should be capable of measuring a wide range of temperatures. In the project, LM34DZ is used to measure the temperature in the beehive. It is calibrated directly in ˚F and measures from -50 to +300 ˚F.

Below is the diagram of LM34DZ. It has 3 pins: Ground, Power supply and Vout. The Vout pin is connected to ADC channel 7. The temperature sensor reads the temperature and passes it on to the MCU (A7). MCU translates the value to corresponding Fahrenheit value and transmits the same.

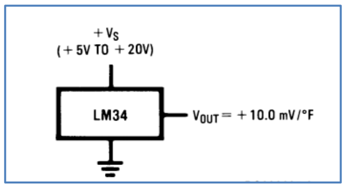


Figure 2 LM34DZ

Beehive Temperature Requirement

As honey bees are social insects, a honeybee colony can survive when the temperature of the external environment is between -4 and +118 ºF. However, they show the best performance at the temperatures between +70 and +95ºF. When the temperature falls below +57ºF, the honeybee starts not leaving the hive and forming a ball (winter cluster). When the temperature falls below +43 ºF, the hive has the appearance of an exact ball. It is known that each bee can produce a heat of 0.1 calorie per minute at 50ºF. Also, depending on various activities going on inside a beehive, the temperature varies widely. For example, when the temperature drops to 57° F, the bees begin to form a tight cluster. Within this cluster the brood (consisting of eggs, larvae, and pupae) is kept warm-about 93° F – with heat generated by the bees. However, most of the times, the beehive temperature will be in the range of 60° F to 100° F.

4.2.1.3 Acoustics Measuring Circuitry

In a beehive, the typical frequency range of the sound will be between 180Hz and 260Hz. So, the acoustic measuring circuitry i.e., the sound sensor built should be able to operate in this band of frequency. The circuitry is shown in the Schematic diagram above.

The audio amplifier circuit accepts a microphone audio input signal. The signal is fed into LM358 op-amp. A 2nd order low-pass filter with a cutoff frequency at 500Hz and a high pass filter with a cutoff frequency at 50Hz are built using the op-amp. The filtered signal is then fed to the ADC through a 330Ω resistor.

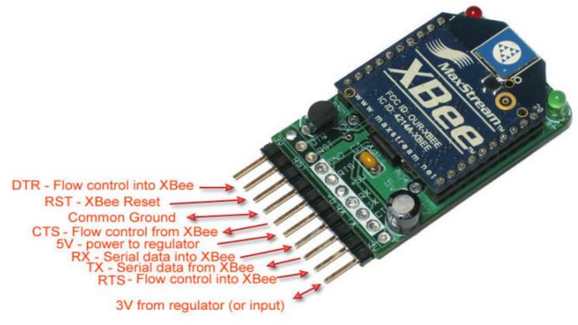
Since the ADC cannot read negative voltages, a bias voltage of 2.5V is chosen so that it sits between 0V and Vcc. To achieve this bias, the input from the microphone is DC-biased to 2.5V with two 100kΩ resistors and a 68nF capacitor. The resulting biased signal is then amplified through a non-inverting amplifier circuit with high-pass cutoff frequency at 50Hz. The high-pass cutoff frequency was chosen such that the DC bias voltage would not be amplified by the non-inverting amplifier. This is also the reason for choosing the specific value for the capacitor at the biasing stage, since the high-pass cutoff frequency of the level-shifter should match with that of the amplifier circuit.

The mic-in circuit runs the input from the microphone through the biasing circuit. This signal is then passed through a non-inverting amplifier with a 2MΩ resistor on the negative feedback loop. This value was chosen after experimentally determining the maximum voltage swing of the mic-in signal to be around 20mV peak-to-peak. Thus, to be able to detect sounds of reasonable strength and distance away from the microphone, the above resistor value is chosen to obtain a gain of 133.33V/V, or +42.4 dB. This ensures that the maximum output of the microphone results in a full voltage swing across the ADC range.

Since the audio signal is sampled with the ADC at 8 KHz sampling rate, it must be low-pass filtered before the ADC conversion to remove aliasing. A 2nd-order Sallen-Key low-pass filter is built using the op-amp with a cutoff frequency of 500Hz.

4.1.1.1 XBEE Pro Radio & Adaptor

An XBEE Pro radio is installed at the transmitter which is used for transmitting the temperature and sound information read and transformed by the MCU. XBEE Pro has a UART interface and can connect directly to a microcontroller. Using UART, the radio can be used to communicate between microcontroller to microcontroller or between PC to microcontroller. XBee Pro radio is designed for 3.3V system. So, in order to interface the radio with ATmega 1284p MCU, an adaptor kit is used.


DTR - Flow control into XBee
RST - XBee Reset
Common Ground
CTS - Flow control from XBee
5V - power to regulator
RX - Serial data into XBee
TX - Serial data from XBee
RTS - Flow control into XBee
3V from regulator (or input)
MaxStream
XBee

XBEE	MCU
Rx	Tx
Tx	Rx
5V	5V
Gnd	Gnd

Fig 4. XBEE Radio with adaptor

Table1: XBEE and MCU Connections

The adaptor can be used to interface for 5V microcontroller board. The XBEE radio can be connected to either a PC or MCU using this adaptor. The radio is mounted on top of the adaptor kit as shown in figure 8. This connection is used to configure the radio which is explained in detail in the software section. Once the radios are configured, then they are connected to the MCU using headers. The connections between the radio and the USART of the MCU are shown in the above table.

Once the radios are configured and the hardware is set up as explained above, one can transmit and receive data using the radios. A XBEE Pro radio could be configured in different configuration depending on the network topology. Various network topologies are explained in detail in the Appendix section of this report

4.2.2 Beehive Receiver

Beehive receiver circuit is responsible for receiving the transmitted values from the transmitter using XBEE, analyze the received values and display them on an LCD.

The schematic diagram of the receiver circuit is shown below.

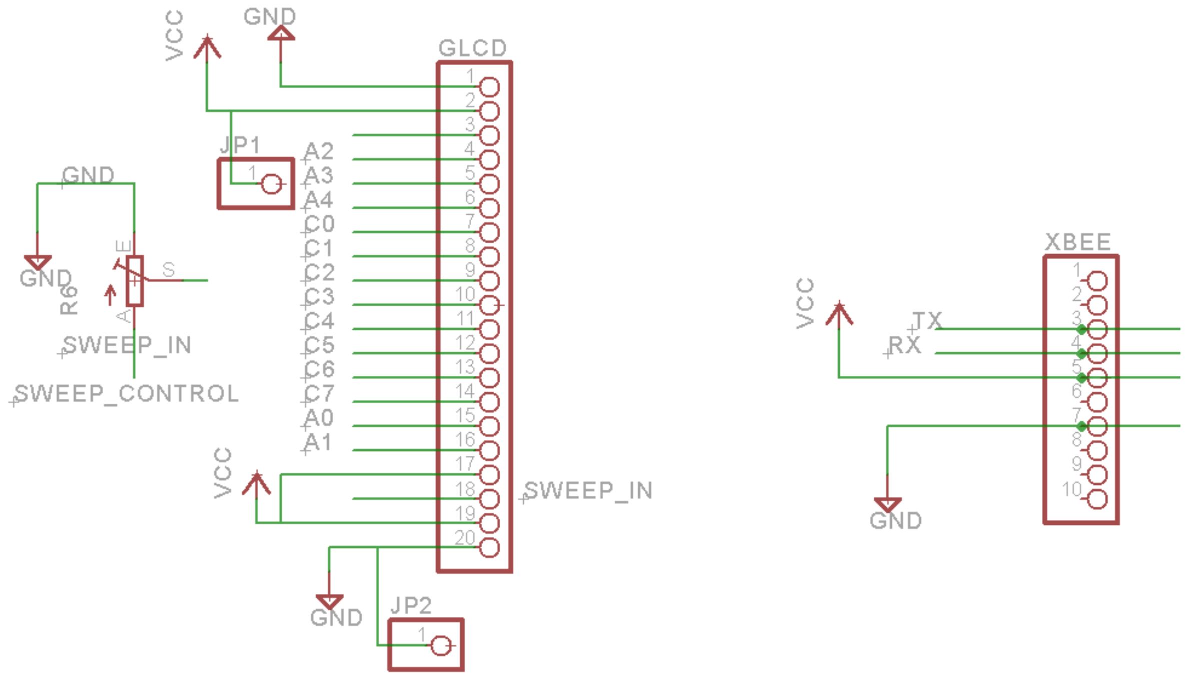


Fig 5. Schematic Diagram of Beehive Receiver Circuit

The A0, C0 etc pins represent the port pins on the MCU.

Once built and tested, I built a PCB for the receiver using EAGLE software. Refer to “Appendix E” for detailed information on PCB Receiver board details.

4.2.2.1 Receiver MCU (ATMega 1284P)

The Receiver MCU is responsible for the reception of the data and data display. This MCU has an XBEE Radio (Receiver Radio) and a graphic LCD interfaced to it.

4.2.2.2 XBEE Pro Radio

An XBEE Pro radio is installed at the Receiver which receives the temperature and sound information transmitted by the transmitter through XigBee protocol.

4.2.2.3 Graphic LCD

KS0105 graphic LCD is used for displaying the temperature and sound information. It has a blue background with 128 x 64 'monochrome' white pixels. For the contrast adjustment, a 10K contrast pot is used. The pin description and MCU connections are explained in detail in the table below. The connections are shown in the picture below.

Figure 3 Receiver MCU Connected to Graphic LCD

PIN	SYMBOL	FUNCTION	MCU PIN

NO.			
1	VSS	Ground	GND
2	VDD	Power supply (+ 5 V)	VCC
3	V0	Contrast adjustment	-
4	D/I	Data/instruction	PA2
5	R/W	Data read/write	PA3
6	E	H → L enable signal	PA4
7	DB0	Data bus line	PC0
8	DB1	Data bus line	PC1
9	DB2	Data bus line	PC2
10	DB3	Data bus line	PC3
11	DB4	Data bus line	PC4
12	DB5	Data bus line	PC5
13	DB6	Data bus line	PC6
14	DB7	Data bus line	PC7
15	CS1	Chip select for IC1	PA0
16	CS2	Chip select for IC2	PA1
17	RST	Reset	VCC
18	V_{EE}	Negative voltage output	-
19	A	Power supply for LED (+ 4.2 V), RA = 0 Ω	VCC
20	K	Power supply for LED (0 V)	GND

Table 1 KS0108 Pin Description and MCU Connections

4.2.3 PCB boards

The resulting circuits (both transmitter and receiver) are then

4.3 SOFTWARE

The software portion is split into two independent project modules i.e. one for the Transmitter MCU and the other for the Receiver MCU.

4.3.1 Beehive Transmitter

The transmitter MCU is responsible for the data acquisition, converting the analog data signal into digital signal by ADC sampling and transmitting the processed data to transmitter Radio through USART.

4.3.1.1 ADC Sampling

The temperature data and modified audio signal from the audio analog circuit was fed into the ADC port A.7 and A.3 respectively. The DC bias voltage was fed into A.1. The ADC voltage reference was set to AVCC. By default, the successive approximation circuitry requires an input clock frequency between 50kHz and 200kHz to get maximum resolution (according to the Atmel Mega1284 specifications). The ADC is set to run at 125 kHz and it takes 13 ADC cycles to produce a new 10-bit ADC value (ranging from 0-1023), or about 104us. Since 10-bit ADC result is used, the ADLAR in ADC was set to zero.

4.3.1.2 The Algorithm:

Temperature Reading:

The output of the temperature sensor is fed into the ADC port A.3. The sensitivity of LM34DZ is 10 millivolts per °F. So if the temperature of the surrounding is 1°F then it's output will be 10mv or if the temperature is 31°F then the output will be 310mv. These small voltages can easily be detected by the AVR after converting them into ADC and by doing a certain amount of calculation; we can get back the temperature. Since the ADC is configured in 10 bit mode with a reference voltage of AVCC, the maximum ADC value can be 1023 and minimum can be 0. The resolution in terms of voltage can be:-

$$5/1023 = 4.88 \text{ mV}$$

If the ADC reading is 10, it means the input voltage is 48.8mV. With a sensitivity of 10mv °F, this accounts to 4.8°F temperature of the surrounding. So for a particular ADC value we will just have to divide by 2.04 (10/4.88) to get the temperature. The temperature value thus obtained is processed into a packet and transmitted. This is explained in the USART section.

Acoustics Reading:

The modified audio signal from audio analog circuit was fed into the ADC port A.3 and the DC bias into A.1 respectively. The digital DC bias output read from A.1 is subtracted from the digital audio output at A.3, so that the actual audio signal value is obtained. Then, this value is converted into decibels using the below formula.

$$\text{Sound} = 20*\log(A.3 - A.1)$$

The sound value thus obtained is processed into a packet and transmitted. This is explained in the USART section.

The Transmission:

The ADC reading happens every 100 milliseconds and ADC transmits the data every minute. This frequency can easily be configured in the code.

Before transmitting the data, the measured readings (temperature and sound) are converted and concatenated. A protocol has been developed for managing the radio communication between the two MCUs. A pipe symbol is used to separate these 2 readings. To indicate the start of a transmission, a special character (#) is appended to the start of the string and to indicate the termination of a transmission a (*) symbol is appended at the end of the string.

The transmission happens one character at a time. So, a "Transmit()" method is developed which uses USART transmission protocol and sets the USART Buffer to the passed in character. This buffer value is transmitted using the XBEE module. Another method

"TransmitString()" is also developed which accepts a string of characters and calls the Transmit function in a loop to deliver the complete string.

How the USART Data Transmission & Reception Work

The USART protocol uses two wires for transmit and receive, and an additional wire when using synchronous mode. I opted to use USART in Asynchronous normal mode since synchronous mode requires an additional clock to check the integrity of the communication whereas the asynchronous mode requires no such clock, instead transmits and receives on an agreed upon format. USART channel 0 is used on the transmitter MCU to transmit the data to the transmitter radio, and the same channel is used on the receiver MCU to receive the data from the receiver radio.

Both USART are set up to transmit 8-bit size characters, with a start bit and a stop bit to signal the start and end of a character and an even parity bit for error detection of 1-bit. Thus, a frame consists of a total of 11 bits. Since both USART are operating in asynchronous mode, the Baud rate of the transmission is set by the following equation:

$$BAUD = \frac{f_{OSC}}{16(\mathbf{UBRR}n + 1)}$$

where fosc is the CPU clock frequency. Since the external clock frequency has to be less than a quarter of the CPU frequency - which is at 16MHz - the maximum BAUD rate that can be achieved from the above equation is 0.25MHz. However, since 9600 is a standard baud rate and a stable one as well, it is used in the project, which is achieved by setting the corresponding UBRROH and UBRROL registers. We can keep sending data at every clock edge as long as we keep the USART Transmit Data Buffer register full at all times.

4.3.2 Beehive Receiver

The Receiver is responsible for receiving the data from the transmitter through receiver radio, analyze and display the data on the graphic LCD. AVR Studio 4 version 4.15 with the WinAVR GCC Compiler version 20080610 is used to write and build the code, and to program the microcontroller. The crystal frequency is set to 16 MHz and compiler optimization was set to -Os to optimize the speed.

The LCD display toggles between temperature and sound information every 5 seconds (configurable in code). The temperature is displayed in Fahrenheit and corresponding indicator showing whether the temperature is Normal, High or Low is displayed on the screen. Similarly the sound information with corresponding high/low indicator is also displayed when toggled to show sound information.

Historic Data: There is a graphical representation of the history displayed below in the screen which shows temperature information for last 80 minutes. This is represented in 8 sets for every 10 minutes and each set denotes the minimum, average and maximum temperature recorded for the period. This period and the number of representations on the screen can be configured easily in the program. When toggled to show sound information, the sound history for last 128 records received from the transmitter is displayed on the screen.

4.3.2.1 The Algorithm:

Reading the received values:

As defined in the developed protocol, the receiver starts the reception as soon as a "#" character is encountered in the received buffer. Until then, any other data transmitted will be ignored. The data is then read until a "*" character is received. The string is built using the characters received between these 2 special characters. This string now contains both sound and temperature values. Using the split function (strtok()) the sound and temperature values are extracted from this string.

Temperature Values:

If the ShowTemp flag is set to true, the received temperature value is displayed on the LCD with corresponding high/low indicator. This value is then passed to a local function for graphical analysis. Along with the 10 minutes average, the minimum and maximum temperature values are also recorded and displayed on the screen graphically.

Sound Values:

If the ShowTemp flag is set to true, the received sound value is displayed on the LCD with corresponding High/Low indicator. This value is then passed to a local function for storing in an array. This array maintains the most recent 128 sound values received by the receiver. These values are displayed on the graphic LCD as vertical lines with the length of each line proportional to the corresponding sound value.

Below is a flow chart describing the functional flow of the receipt and processing of the values.

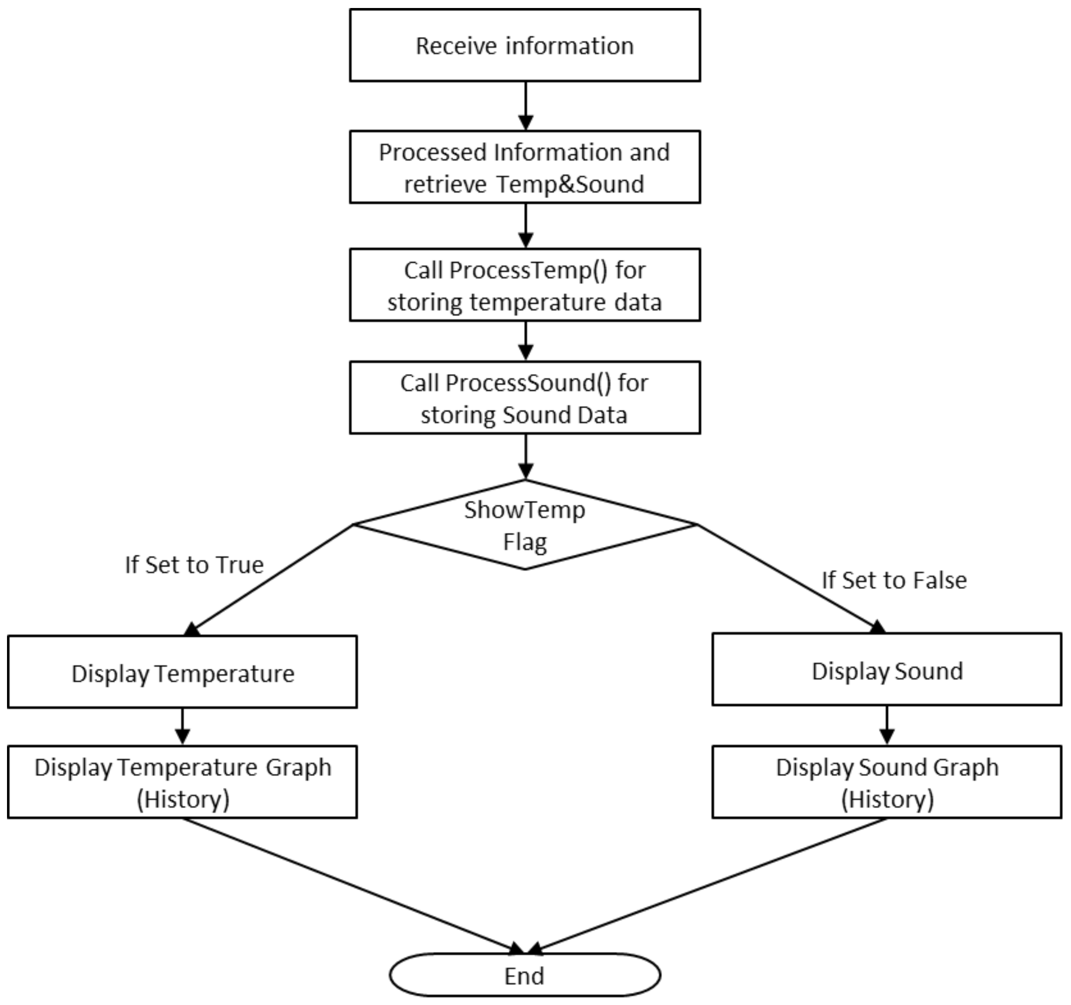

Receive information
Processed Information and retrieve Temp&Sound
Call ProcessTemp() for storing temperature data
Call ProcessSound() for storing Sound Data
ShowTemp Flag
If Set to True
If Set to False
Display Temperature
Display Sound
Display Temperature Graph (History)
Display Sound Graph (History)
End

Fig 6: Flow Chart for Receiver

5 TEST METHODOLOGY & DIFFICULTIES

The test strategy mainly focused on building and testing individual components and promoting the end system in phases. Each unit is developed and tested independently. The approach of building and testing different units separately helped me in facing very few problems in the integrated test environment.

Some of the key components (build and test)

Temperature Sensor:
I interfaced the temperature sensor and a 16X2 LCD (that I used in other labs) on a single MCU. This was mainly to test the temperature sensor and see if the readings are rightly converted into corresponding ADC values. After interfacing and developing the necessary code, I tested the

circuit extensively by inputting various temperature values and observing the outputs. To simulate different temperatures, I used room heaters and windows mostly. The circuit was not hard to build, but getting the LCD connections properly without any loose contacts was a bit time consuming task. But once the LCD got working, the output was easy to check and found to be consistent.

XBEEs

Testing the XBEE was the trickiest part. Since testing both the radios at the same time was a bit difficult in order to analyze and narrow down where exactly the problem is, I performed staged testing of receiver radio and transmitter radio.

XBEE (Transmitter)

After configuring one of the XBEEs for transmission, and connecting it to the Transmitter MCU through USART, the other XBEE (receiver) was then connected to PC using X-CTU software for testing. Then the transmitter is tested with various temperatures and the corresponding results were observed on the X-CTU's receiver terminal. This also partly completes the integrated system testing for the transmitter part.

The next step is to ensure that the receiver MCU is receiving the information correctly and the same is displayed on the graphic LCD.

XBEE (Transmitter)

The other XBEE is configured for reception, and connected to the Receiver MCU through USART. Then the transmitter XBEE was connected to PC using X-CTU software for testing. Then the input is passed from the X-CTU software (from transmitter terminal) and successfully checked for the display of the same on the graphic LCD. This also partly completes the integrated system testing for the receiver part.

Once both these units are found to be functioning, I performed a complete system integration test using both receiver and transmitter circuits independently functioning. Till this point, only 16X2 character LCD is used for its simplicity.

GraphicLCD:

Once the integrated system is functioning correctly, I replaced the 16X2 LCD with the graphic LCD and wrote the software necessary for the graphic LCD. Initial target was to just ensure proper display of the temperature values. It was bit challenging to interface graphic LCD as it's more complicated than character LCD and I was using it for the first time.

Once this was found working as expected, I built the necessary software to display graphs and to enhance the visualization of the data being displayed on the LCD.

Microphone & Audio Information:

Testing the transmission and display of audio information was relatively easy as most of the system was ready by then. Lot of the software from the initially built system is reused.

Below are some of the difficulties/challenges faced during the course of the project.

XBEE Usage:

As I was using XBEEs for the first time, there was a big learning curve to learn and understand the XBEE functionality and how to use them.

Bricked XBEEs:

During the testing of communication using XBEEs, I suddenly found one of the XBEE radio non-functional. I initially thought that the XBEE failed and cannot be used anymore. As I had a backup XBEE I proceeded with my tests with that. After sometime, it got the same problem and I had no clue on how to proceed with the project for some time. Then after lot of googling I learnt about Bricked XBEEs and how to make them functional again.

Graphic LCD related Issues

Initially, I also faced difficulty connecting the graphic LCD. I didn't have the correct documentation and there were different connections described in different web sites. Finally I was able to get this connection done and since then there were no issues with the LCD connectivity. But, while displaying the graphs on the LCD, I faced issues like graphs interfering with the text. It required lot of manual calculations and add lot of code validations to ensure the software does not allow this overlap.

6 RESULTS AND OBSERVATIONS

6.1 Requirements Match

Tag	Status	Comments
[SYS-01]	Met	A transmitter module with a temperature & sound sensor is designe
[SYS-02]	Met	A receiver module with a display device is designed
[SYS-03]	Met	A transmitter module with a temperature sensor is designed
[SYS-04]	Met	A transmitter module with a sound sensor is designed
[SYS-05]	Met	A transmitter XBEE radio is interfaced with the transmitter MCU
[SYS-06]	Met	The data was classified in to low, normal, high ranges
[SYS-07]	Met	The system designed is well within the budget
[SYS-08]	Met	The system is placed in weather proof box
[SYS-09]	Met	All the necessary optimizations are done
[SYS-10]	Met	Tested
[SYS-11]	Met	Tested
[SYS-12]	Met	Tested
[SYS-13]	Met	Tested
[BHT-01]	Met	Temperature and sound are read once every 100ms and averaged once every 100 samples
[BHT-02]	Met	All the necessary optimizations are done
[BHT-03]	Met	A transmitter XBEE radio is interfaced with the transmitter MCU
[BHT-04]	Met	A transmitter XBEE radio is interfaced with the transmitter MCU
[BHT-05]	Met	The transmitter is solar powered
[BHR-01]	Met	A receiver XBEE radio is interfaced with the receiver MCU
[BHR-02]	Met	Data received is displayed in a user understandable format
[BHR-03]	Met	Data received is displayed in a user understandable format
[BHR-04]	Met	Temperature is displayed in °F
[BHR-05]	Met	A history of 1.33 hours data is displayed graphically
[BHR-06]	Met	Sound intensity information is displayed in dB (Decibels
[BHR-07]	Met	A history of 128 samples data is displayed

Table 4: Requirements Match

6.2 Results at a glance

The final results of the project were extremely satisfying. Majority of the expectations set in the project proposal were met. A fully functional telemetry system to monitor the temperature and sound in a beehive with a graphical display (bar graphs and line graphs) of the data in real time is designed. The project accurately displayed the data received. The corresponding PCBs are designed for Receiver and Transmitter and ordered online at Advanced Circuits. Refer to Appendix E for the details of PCBs received. The material costs were kept down and the system designed is also user friendly. By switching on the display device, the user will be able to know the temperature and sound intensity level inside a beehive at that time instant. The added display options like displaying the minimum, maximum and average values in bar graph representation over a specific period helps the user to know about the data in a more user-friendly way. On the display device, the log data is also shown, which helps the user to know and compare not only the absolute data values but also the relative data values.

Integrated system with both receiver and transmitter is found functioning as expected. Both XBEEs are connected and the communication has been established and noticed to be consistence for long periods of time. The display of temperature and sound information are periodically toggled and the graphical information is displayed consistently.

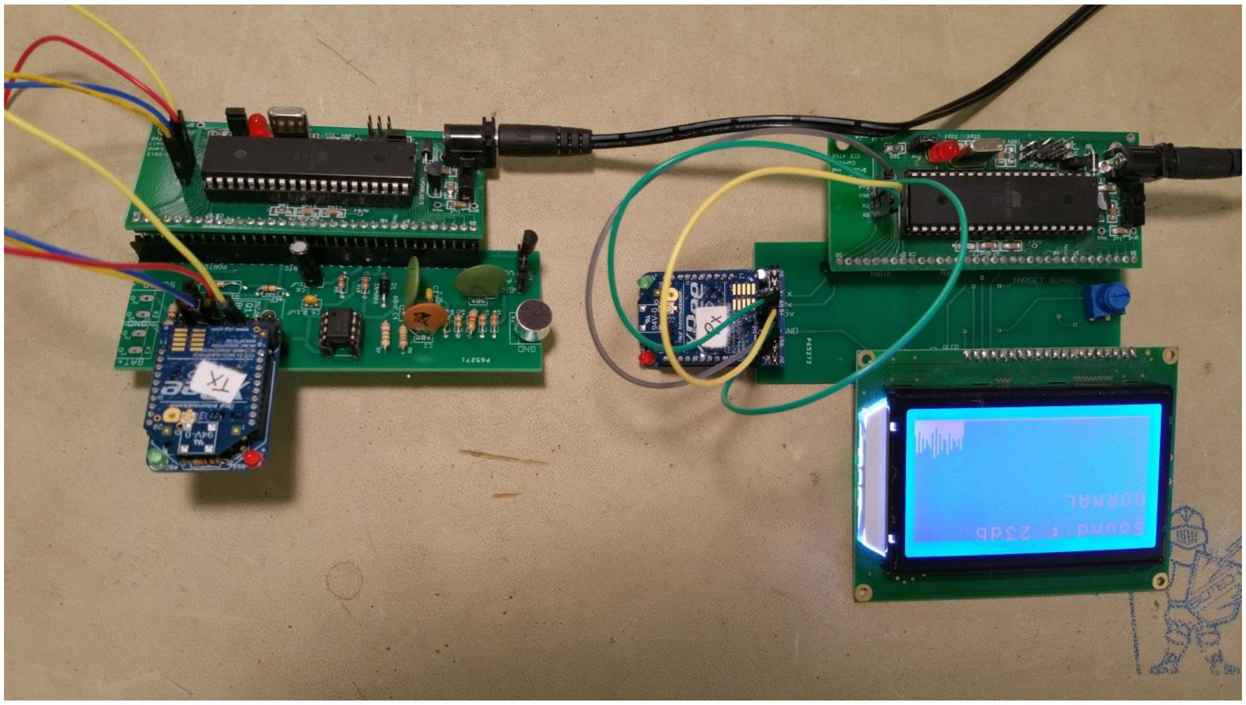


Fig 7. Receiver and Transmitter integrated and functioning correctly

Temperature Display

The temperature transmission is consistent and the values are displayed and refreshed every 10 seconds. History of temperature information (average, peak and low temperatures) is displayed as a graph at the bottom of the screen. The history is indicated by 8 towers, each containing low, average and peak for the given time slot and scrolled whenever a new record is added.

For simulating high temperature, I have used a room heater and noticed the peaks and lows displayed correctly on the screen.

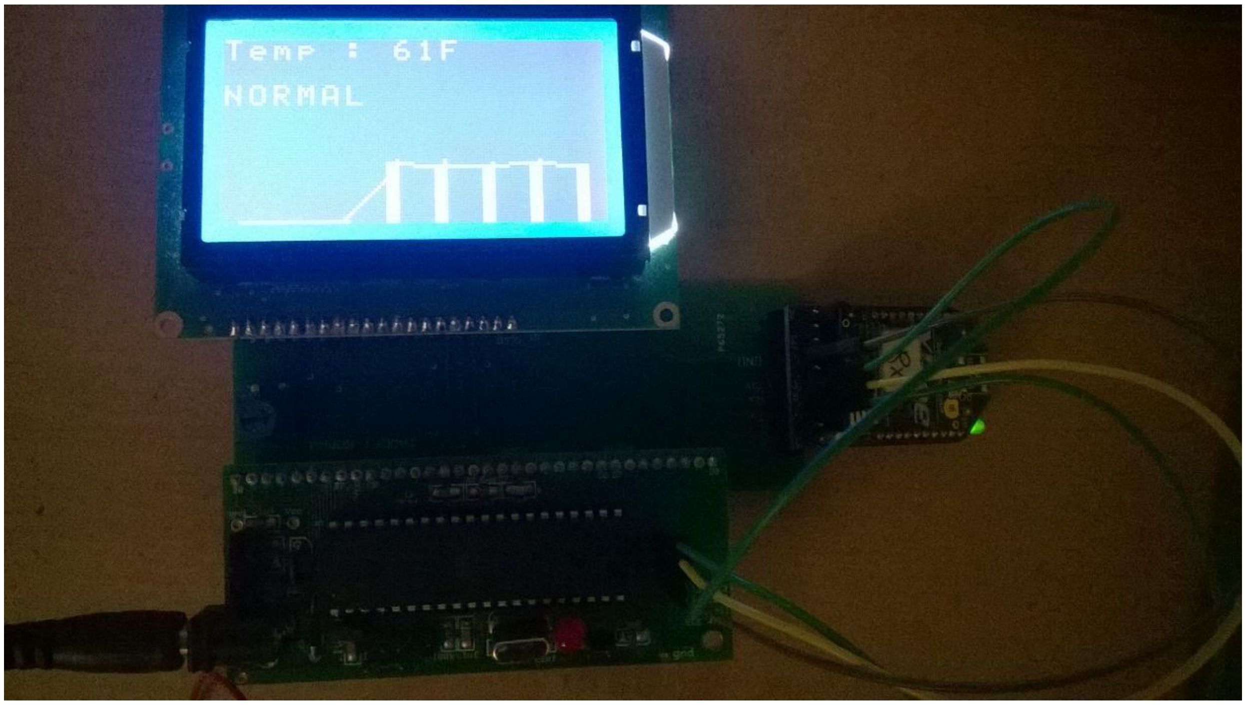


Fig 8. Temperature Values accumulated over different time instants

Sound Display:

The sound information is displayed with the decibel value and a history for past 128 receptions (received every 10 seconds) is displayed as a scrolling graph at the bottom of the screen. Simulated different sound levels and noticed the peaks and lows consistently recorded and displayed on the screen.

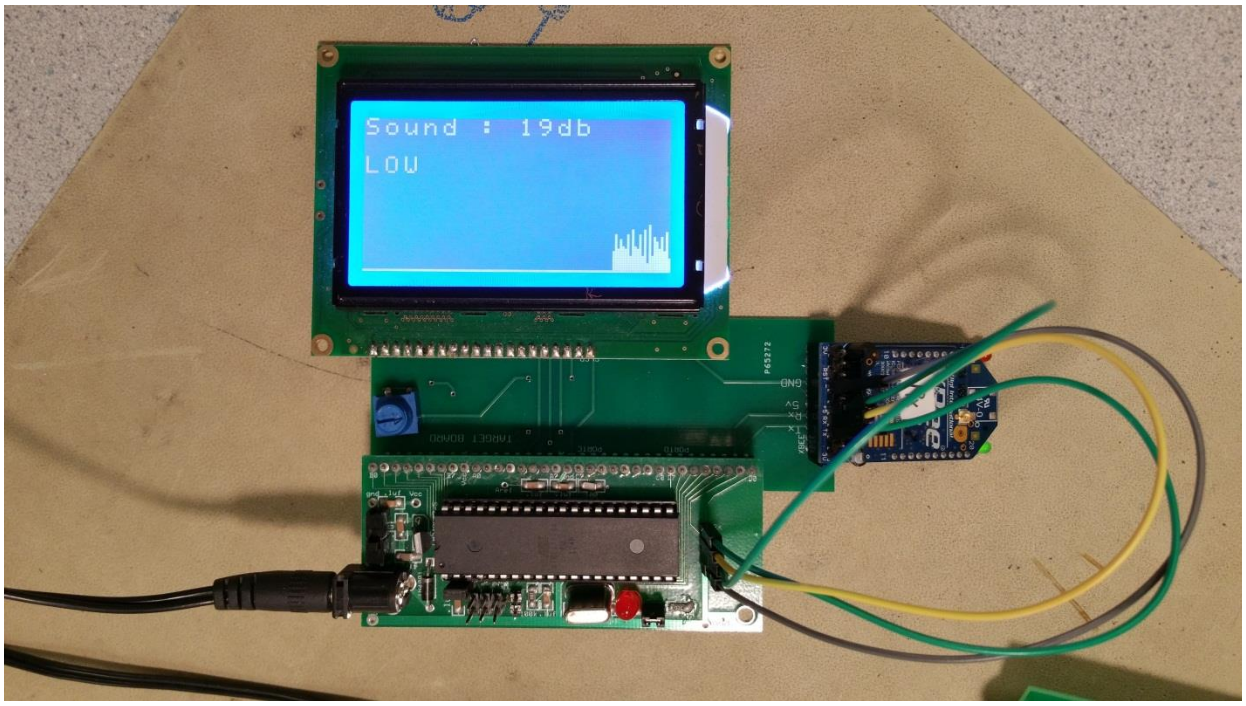


Fig 9. Sound Values accumulated over different time instants

Overall, the results are consistent and both the modules are found to be stable.

6.3 Cost: bill of materials

An important challenge was to keep the costs as minimal as possible; the target was to keep the cost under $500. This has been met. The bill of materials is given below.

Part Name	Digi-Key/Adafruit Part Number	Unit Price ($)	Quantity	Total Price ($)
MCU	ATMEGA1284P-PU-ND	8.6	2	17.2
Temperature Sensor	LM34DZNOPB-ND	2.52	1	2.52
XBEE Pro	602-1101-ND	34	2	68
XBEE Adaptor	ADA/126	10	2	20
Graphic LCD	ADA/188	24	1	24
Electret Microphone	COM-08635	0.95	1	0.95
OP-Amp	LM358p	0.05	1	0.05
Solar Panel	2 WATT 12 Volt Model # 58012	19.77	1	19.77
Battery	BP7 - 12	34.95	1	34.95
Total				**187.44**

Table 3: Bill of Materials

7 CONCLUSION

7.1 Future Work

In the future, the project could be extended to control the temperature inside the hive. Also, an alarm could be added to the system to notify the owner in case of any sudden changes in the hive conditions. Also the log data could be stored in some sort of storage device for the user to view it at a later point of time.

7.2 More Compliance Considerations

All the standards relevant to the project were followed and conformed to.

7.3 Ethical Considerations

During the course of working on this project, the IEEE Code of Ethics was followed with strict accordance. I ensured that neither mine nor the user's safety is at stake while working on the project. There are no high voltage/current sources that could hurt a person using our project. So, there are no safety measures to be taken while operating the project. I kept the lab space clean and minimal. I began the task after making sure that I have a good and attainable design that was thoroughly reviewed by my project advisor. When stating the lists of cost estimates, I was accurate and honest, not concealing any potential costs that I know of. I was also quite self-critical of the design, accepting honest criticism of my work to help me improve. I sought the help of the project advisor whenever I couldn't figure out how to tackle an issue. I was not biased towards anyone based on any factors, such as race, religion, and national origins. I treated all of my peers with respect and did not say or publish anything offensive while in lab. I was respectful to others wishes and made sure that I was not hindering their work. I cited all contributions made to the project including any of my own previous work. Overall, I honestly tried to create the best project that I could, given time constraints, and I was honest with others and myself throughout the process.

7.4 Legal Considerations

There are no major legal considerations to be concerned as I did not transgress upon any intellectual property. The project does not pose any known danger to any people or property. I used standard 9-12V AC adapters to power the system, all of which should conform to legal standards.

8 ACKNOWLEDGEMENTS

I would like to thank my MEng project advisor, Dr. Bruce Land for supporting me throughout the entire project. Bruce is someone you will instantly love and never forget once you meet him. He’s the coolest advisor and one of the smartest people I know. I hope that I could be as lively, enthusiastic, and energetic as Bruce. I am also very grateful to him for his advice and knowledge and many insightful discussions and suggestions. He is my primary resource for getting my questions answered and was instrumental in helping me finish this project.

This journey would not have been possible without the support of my family. Thank you for encouraging me in all of my pursuits and inspiring me to follow my dreams. I am especially grateful to my mother, who supported me emotionally and financially.

I would also like to specially thank my Uncle, for helping me through the entire project. I always knew that you believed in me and wanted the best for me. Without your support, I wouldn’t have been able to finish the project on time.

I would like to thank my cousin, Vamsi, for helping me with the layouts.

9 REFERENCES

[1] http://www.ti.com/lit/ds/symlink/lm34.pdf
[2] http://people.ece.cornell.edu/land/courses/ece4760/AtmelStuff/mega1284full.pdf
[3]http://tutorial.cytron.com.my/2011/07/30/xbeexbee-pro-wireless-communicate-with-microcontroller/
[4] http://ftp1.digi.com/support/documentation/90001003_a.pdf
[5] http://www.instructables.com/id/XBee-adapter/
[6] http://www.vishay.com/docs/37329/37329.pdf
[7]http://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/f2014/hl529_vh84_mk2335/vh84_hl529_mk2335/vh84_hl529_mk2335/index.html
[8]http://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/s2006/ajk28/ajk28/Images/schematic.bmp

APPENDICES

Appendix A: Installation and User Manual

Installation of the beehive monitoring system is pretty simple. It consists of 2 major components. I.e. Transmitter and receiver.

The integrated transmitter comes in a box with a battery and solar panel attached to it. The solar panel needs to be positioned in a way that it is exposed correctly to the sun light and is connected to the battery which is connected to the transmitter circuit. The box (main transmitter circuit) is placed inside the beehive as close as possible to the bees. Once it is switched on, the transmitter is ready and starts transmitting the temperature and audio information.

The receiver can be placed at a convenient location inside the house at a maximum of 1500 mt distance from the hive/transmitter circuit. As this is placed inside the house, this can be connected to a power supply. Wall mounting the receiver is suggested as it gives the best view of the graphic LCD.

Appendix B:XBEE Pro Radio

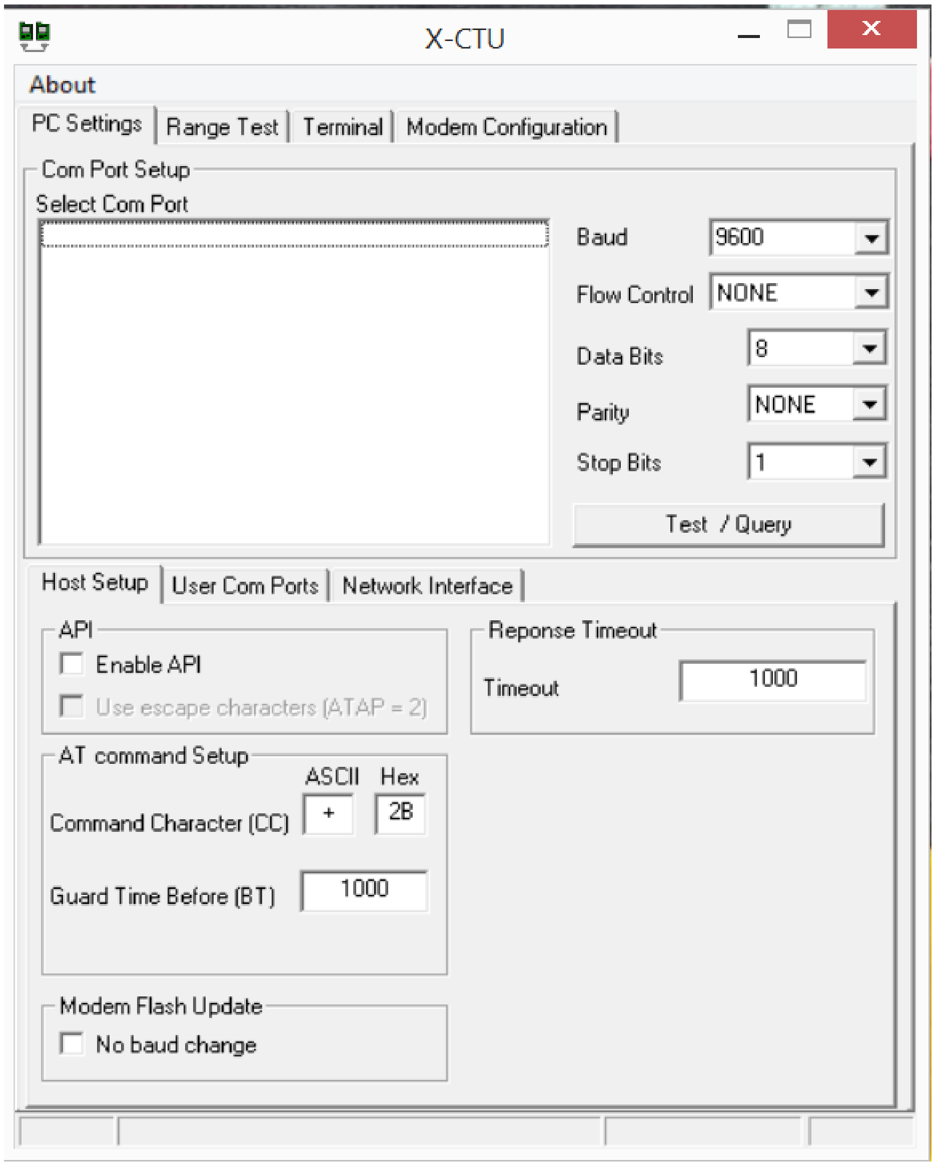
- Outdoor/RF Line-of-Sight Range Int’l 5000 ft (1500 m)
- Serial Data Rate 1200 bps - 1 Mbps
- Fast 250 kbps RF data rate to the end node
- 2.4 GHz for worldwide deployment
- Sleep modes supported for extended battery life

XBEE Pro radio can act in different configurations which are described in the table below.

Term	Definition
PAN	Personal Area Network - A data communication network that includes one or more End Devices and optionally a Coordinator.
Coordinator	A Full-function device (FFD) that provides network synchronization by polling nodes [NonBeacon(w/ Coordinator) networks only]
End Device	When in the same network as a Coordinator - RF modules that rely on a Coordinator for synchronization and can be put into states of sleep for low-power applications.
Association	The establishment of membership between End Devices and a Coordinator. Association is only applicable in NonBeacon (w/Coordinator) networks.

Table:5 XBEE Radio Network Configurations

Appendix C: X-CTU Software



Com Port Setup
Select Com Port

Baud: 9600
Flow Control: NONE
Data Bits: 8
Parity: NONE
Stop Bits: 1

API
- [ ] Enable API
- [ ] Use escape characters (ATAP = 2)

Reponse Timeout
Timeout: 1000

AT command Setup

	ASCII	Hex
Command Character (CC)	+	2B
Guard Time Before (BT)	1000	

Modem Flash Update
- [ ] No baud change

Fig.9 X-CTU PC Settings Tab

PC Settings: Allows a user to select the desired COM port and configure that port to fit the radio settings.

Range Test: Allows a user to perform a range test between two radios.

Terminal: Allows access to the computers COM port with a terminal emulation program. This tab also allows the ability to access the radios' firmware using AT commands.

Modem Configuration: Allows the ability to program the radios' firmware settings via a graphical user interface. This tab also allows users the ability to change firmware versions.

Appendix D: Schematics

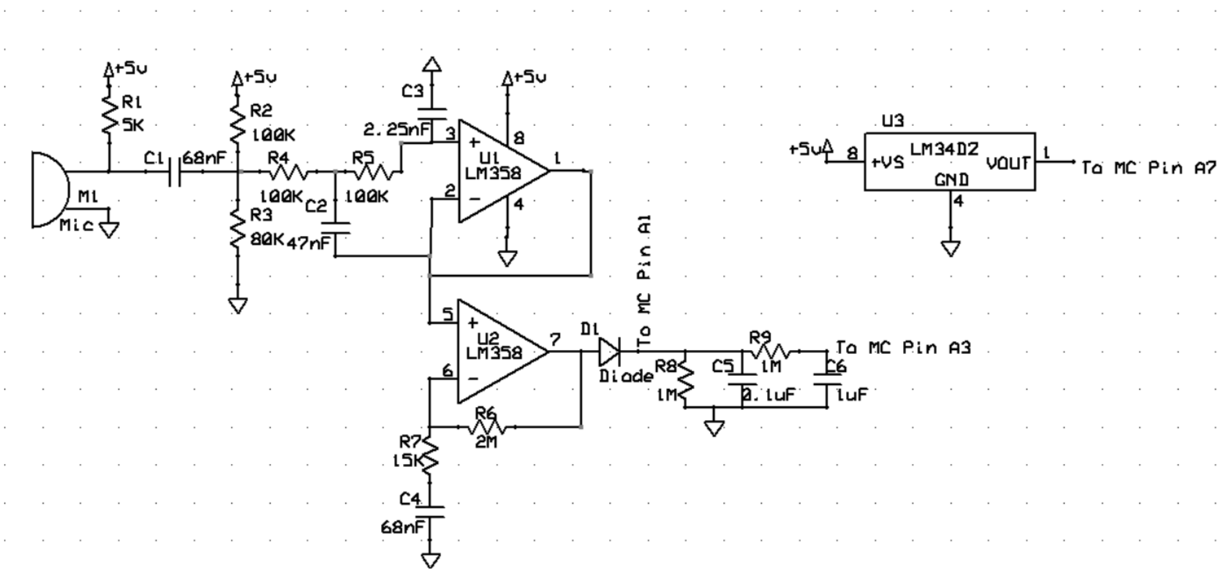


Fig 10: Beehive Transmitter Schematics

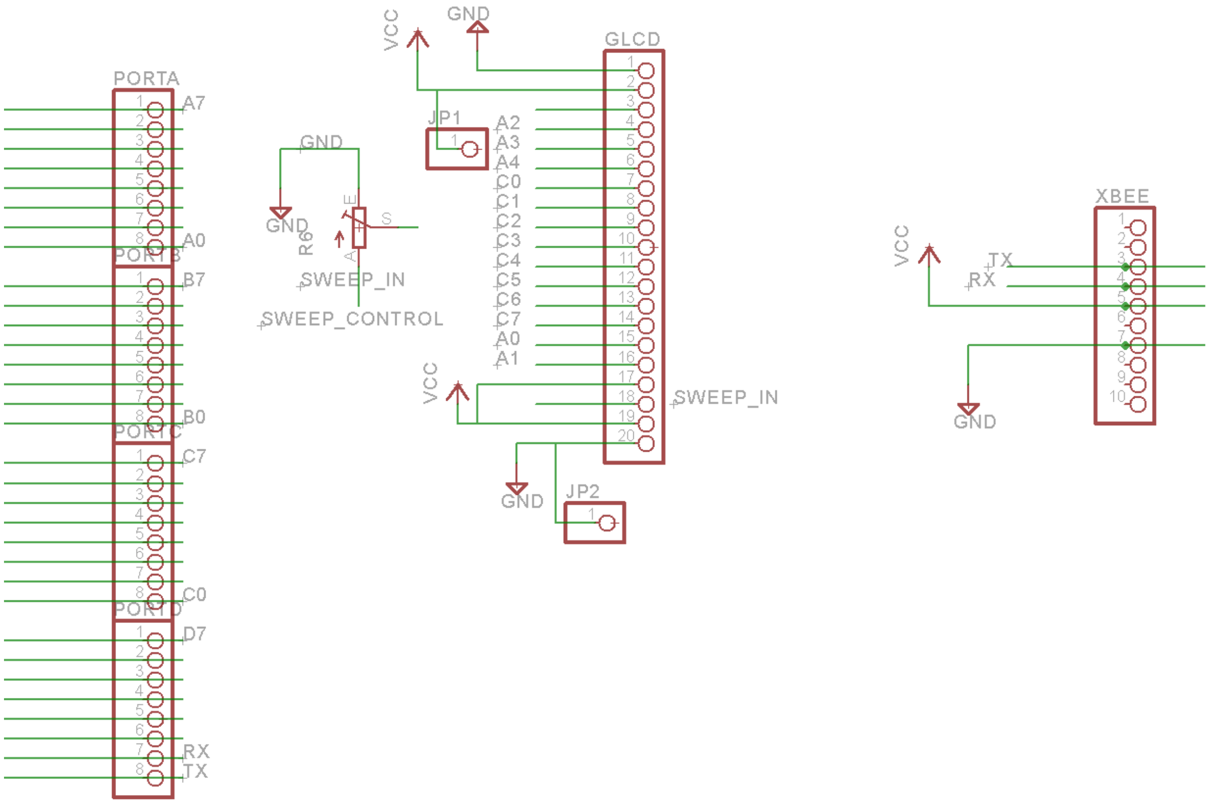


Fig 11: Beehive Receiver Schematics

Appendix E: Layouts

Find below the transmitter and receiver layouts built using EAGLE software. Though there are multiple software options available for PCB layout designing I chose EAGLE as it has plenty of features that make it very easy to build PCB designs. Layout Auto Routing is one of such features that is very helpful in the design.

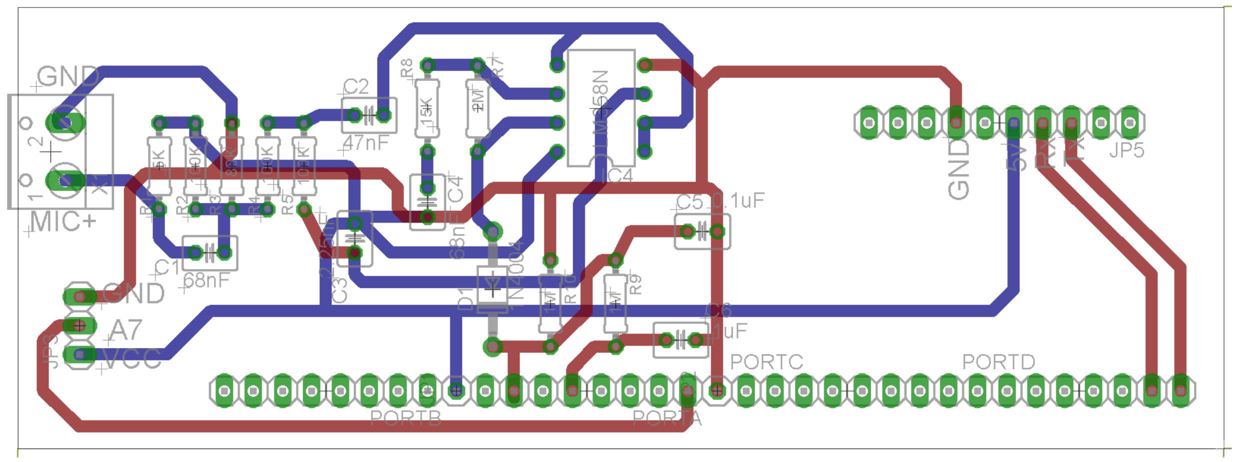


Fig 12: Beehive Transmitter Layouts

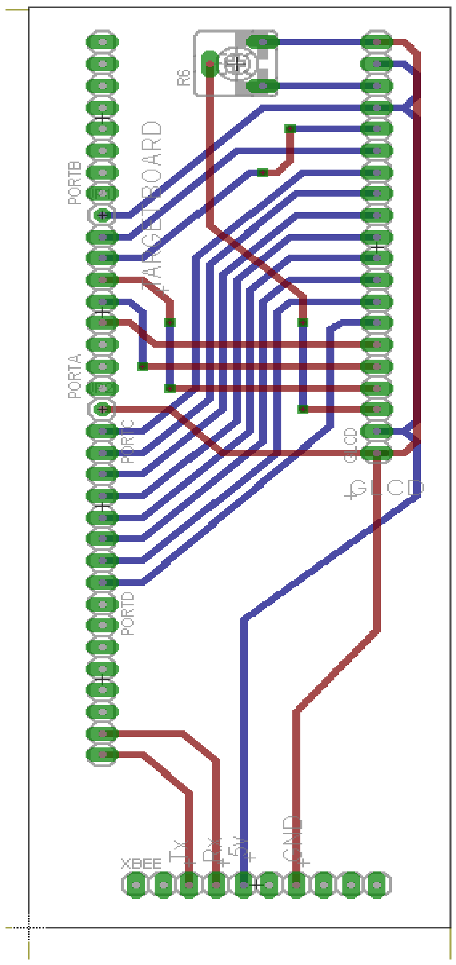
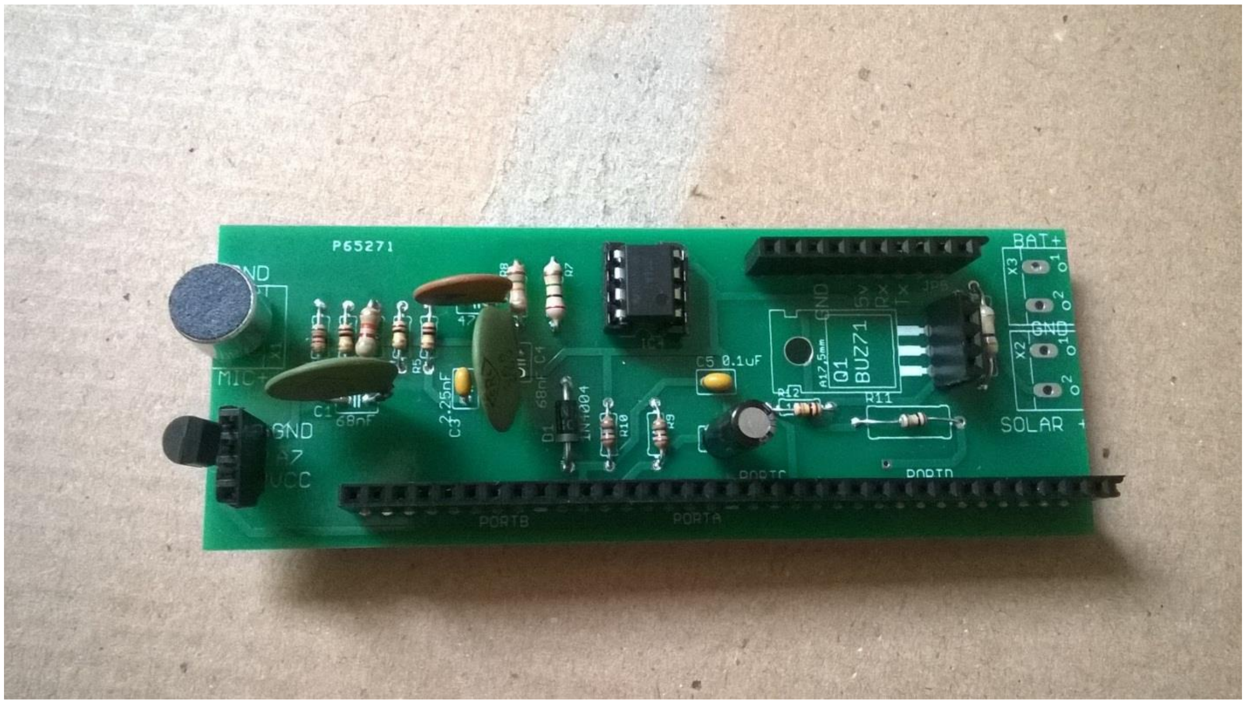


Fig 13: Beehive Receiver Layout

Once built using EAGLE software, I have placed the order on Advanced Circuits (4pcb.com). 4pcb.com has multiple options to get the quote and place the order. It also has a special discount option for the students. You need to upload corresponding gerber files for placing the order. The boards costed a total of $90 for both PCBs including shipping.

Here are the received PCBs.

Transmitter:



Receiver (with MCU target board plugged in):