

Project 2: Metal Detector

Course: ELEC 291/292

Instructor: Dr. Jesús Calviño-Fraga

Members: Alex Lassoij , Christophorus Hansen, Scott Jiang

Date: April 14th, 2021




Project Contributions			
Member	Student Number	Percentage Contributed	Signature
Alex Lassoij	38574752	33.3%	
Christophorus Hansen	10984375	33.3%	
Scott Jiang	78882545	33.3%	

Table of Contents

1.	Introduction	
1.1.	Objective.....	2
1.2.	Specification.....	3
2.	Investigation	
2.1.	Idea Generation.....	4
2.2.	Investigation Design.....	5
2.3.	Data Collection.....	5
2.4.	Data Synthesis.....	5
2.5.	Analysis of Results.....	6
3.	Design	
3.1.	Use of Process.....	6
3.2.	Need and Constraint Identification.....	7
3.3.	Problem Specification.....	7
3.4.	Solution Generation.....	8
3.5.	Solution Evaluation	9
3.6.	Safety/Professionalism.....	9
3.7.	Detailed Design	9
3.7.1.	Hardware Detailed Design	9
3.7.2.	Software Detailed Design.....	11
3.8.	Solution Assessment.....	13
4.	Life-Long Learning.....	14
5.	Conclusion	15
6.	References.....	16
7.	Appendices.....	17
8.	Bibliography.....	17

1. Introduction

1.1 Objective

For this project, our group's objective was to develop a metal detector capable of detecting both non-ferrous and ferrous metals. Additionally, it needed to detect the size of the metal (between small and large) which was determined by a frequency threshold. The metal detector uses voice feedback to indicate the following types of objects:

1. Small non-ferrous metal
2. Large non-ferrous metal
3. Small ferrous metal
4. Large ferrous metal

We also added additional functionalities with one being an LCD that also displayed the type of metal, but also indicated when no metal was near the metal detector to which it would indicate, "Place a metal". Another additional functionality that we added is adding the red LED as an indicator if the metal detector is still searching for reference frequency.

1.2 Specifications

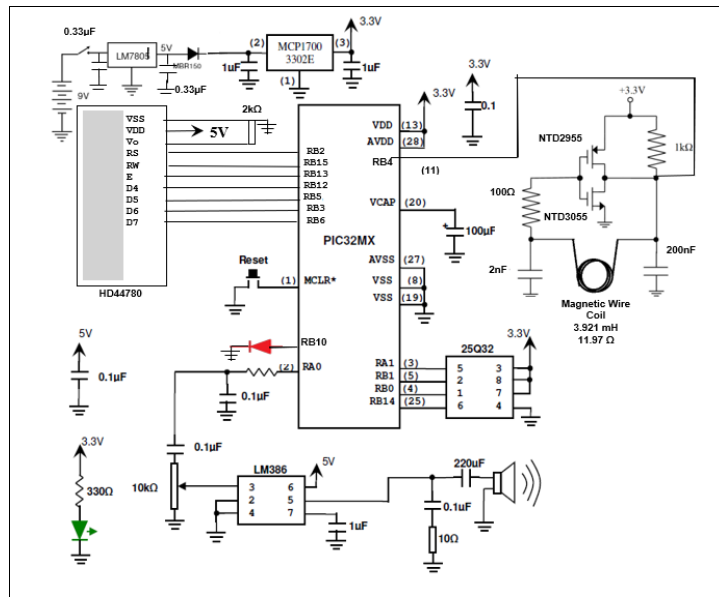


Figure 1: Hardware block diagram

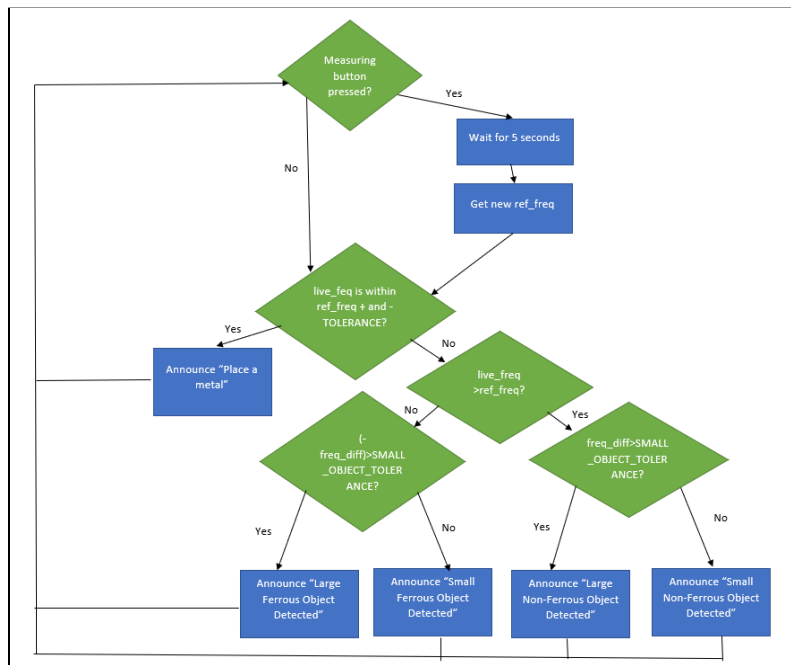


Figure 2: Software Block Diagram

Below is a list of all hardware parts used for this metal detector:

- PIC32MX130F064B Microcontroller
- HD44780 LCD controller
- LM386 Low Voltage Audio Power Amplifier
- W25Q32 Serial Flash Memory
- MCP1700 3302E Voltage Regulator
- BO230XS Board
- NTD2955 and NTD3055 MOSFETs
- Bourns 10k Variable Resistor
- 220 uF, 100 uF Electrolytic Capacitors
- 0.1 uF, 47 nF, 10nF Ceramic Capacitors
- Green and Red LEDs

2. Investigation

2.1 Idea Generation

Our group brainstormed ideas by following the engineering design process. Before looking at adding any additional functionality, we first tried to understand how to implement the basics - metal detection and voice feedback. We researched each of these features together as a group and it took our group around 5 days to fully understand how to implement the basic features of this project before working on it. We first tested if our inductor coil worked by using the pre-written program posted by our instructor. We wanted our metal detector to react quickly to different metals so that meant operating at a higher frequency, which also meant that it had higher sensitivity to surrounding metals. Our group also wanted to know when a new metal could be placed in the coil so we decided to implement an LCD display that not only indicated the type and size of the metal but also when no metals were detected within the coil.

2.2 Investigation Design

To ensure that we are aware of any relevant information that may help us execute our ideas and avoid making mistakes, we referred to an article, some online resources and performed some of our own preliminary tests with our hardware.

2.3 Data Collection

It was recommended that we gave the Circuit Cellar article posted on Canvas a read. This article helped us understand the different problems that may arise when building a metal detector. Furthermore, it helped paint a clearer picture of what steps we would needed to follow to achieve our goal. While extensive hardware and code examples were given, it solidified our understanding of the project as a whole, and made debugging and problem solving easier along the way. A crucial part of our detector was the Colpitts Oscillator. To build one properly, we referred to datasheets to pick appropriate values for the capacitors [1]. Later on, we connected the output of the oscillator to our microcontroller of choice. This enabled us to view the oscillator's frequency in a PuTTY terminal session. Furthermore, we looked at datasheets concerning the PIC32 microcontroller family and the LCD in our kit to incorporate an LCD screen into our detector to give the user visual feedback.

2.4 Data Synthesis

Our main source of data was the output in the PuTTY terminal. We were able to get an idea of what frequency our detector would operate at. This way, we could also find out how our detector would react to several different metallic objects. We also connected our LCD to the circuit to

make sure it functions properly. By looking at the datasheet, we knew which pins of the PIC32 we were able to use to wire up the LCD and have it work properly.

2.5 Analysis of Results

Through thorough testing, we noticed that our oscillator was less susceptible to unwanted inconsistencies if the operating, or searching frequency was higher. Therefore, we opted to go with different capacitors to alter the frequency. By moving many different objects towards the detector and observing the changes in frequency, we were able to determine threshold values. These values would be crucial in differentiating between an actual object and a simple fluctuation, as well as defining the line between a small and large metallic object. We also observed that the frequency was stable most of the time, which was a result of the frequency being measured many times over a certain time interval.

3. Design

3.1 Use of Process

Our group was guided by our ELEC 291 instructor, Dr. Jesus Calvino Fraga, to resolve hardware and software constraints during the process of assessing and debugging the metal detector. We were given the Computer_Sender.c and PIC32_Receiver.c to upload our .wav sound file. This helped us figure out how the SPI communication between the PIC32 microcontroller and W25Q32 SPI flash memory played the .wav file in our project 2 code. Most of the time that was dedicated to this project was used to develop and debug our software algorithms. In order to minimize code errors during the process of developing algorithms, we commented out some functions that could possibly ruin the whole workings of the metal detector. If we did not find the

bug, we repeated that process once or twice more and eventually figured out where the bug came from.

3.2 Need and Constraint Identification

The needs and constraints that we identified were largely based on the project requirements, but we also added some additional functionalities to emulate real industry metal detectors. Below are some things we considered:

- Quick and easy identification of metals (immediate response to when a new metal is detected)
- Indication whether or not a metal is near the coil
- Adding visual feedback for those who are hearing impaired
- Indicate when the metal detector is searching for a proper reference frequency

3.3 Problem Specification

Before the metal detector could start measuring, we implemented a blinking red LED to indicate that the microcontroller was still searching for a reference frequency within the time window of 5 seconds. This allows the user to move away any metals nearby to get proper reference frequency before measuring. If there is no metal nearby or no metallic object, the LCD is going to print “Place a metal”. For indicating whether a metal is small, large, ferrous, or non-ferrous, we coded output sounds using W25Q32 SPI flash memory and the speaker as well as writings through LCD. For example, if a soda aluminium pop can is placed within the coil, “Large N Ferrous Object Detected” would be printed on the LCD and “Large Non Ferrous Object Detected” would be announced.

3.4 Solution Generation

Firstly, we wanted to ensure that our metal detector could respond quickly when different metals were detected in or near the coil. This was in response to making the metal detector more “realistic” in situations such as trying to find metals on a beach or elsewhere. Originally, the metal detector was quite slow at detecting when a metal was swapped out so we increased the frequency from 23kHz to around 47kHz. Additionally, since we used an LCD to display details about the detected metal, we wanted to ensure that the user would understand when no metal was detected. Before our metal detector can search for metal, it needs to get the reference frequency for comparing the frequency when it is trying to detect metal. For searching the reference frequency, we coded our PIC32 to wait for 5 seconds to ensure the reference frequency is obtained when there is no metal nearby. After getting the reference frequency, the LCD would indicate ‘Place a metal’ when nothing was detected near or within the coil. This would only be indicated when the measuring frequency is between the value of reference frequency - TOLERANCE and reference frequency + TOLERANCE (TOLERANCE is defined as 30 Hz). If the measuring frequency is out of these bounds, we coded the microcontroller to subtract the measuring frequency and reference frequency. If the difference is below or above SMALL_OBJECT_TOLERANCE (SMALL_OBJECT_TOLERANCE is defined as 100 Hz), the detector recognizes the object as a small or large metal. If the frequency difference($\text{freq_diff} = \text{live_freq} - \text{ref_freq}$) is positive or negative, the detector will recognize the object as a non-ferrous or ferrous metal. We added an LCD as a bonus that would allow the hearing impaired to see what form of metal was being detected. We were also able to implement an additional feature with the LED that would flash when indicating that the metal detector was searching for reference frequency.

3.5 Solution Evaluation

We also thought about adding a buzzing sound that would indicate how close the coil was to a metal (much like a real life beach metal detector). However, we couldn't implement this feature in concurrence with the required voice feedback feature. We also considered adding a series of LEDs that could indicate the strength of the signal and proximity to a metal by the number of LEDs that were lit. Unfortunately, after adding the required features and LCD, we didn't have enough ports to add the LEDs.

3.6 Safety/Professionalism

Throughout the course of this project we ensured that everyone communicated effectively with each other. This meant properly setting a schedule for when to work on the project and having different responsibilities for everyone. Additionally, Since one of our members was in a different city, we still wanted to ensure that everyone followed proper safety measures while working on the project (i.e wearing safety goggles and not having water near any electronics).

3.7 Detailed Design

3.7.1 Hardware Detailed Design

Colpitt Oscillator Circuit (Refer to Appendix A for Image)

The colpitt oscillator circuit is a part of the circuit that generates frequency based on the value of capacitors on the left (C1) and right (C2) in parallel ($C_T = C1 || C2$), and inductance of magnetic wire coil L using the equation below:

$$f = \frac{1}{2\pi\sqrt{LC_T}}$$

For optimizing the metal detecting function, we designed the reference frequency to be around 47kHz. Hence, we chose C1 to be 2nF and C2 200 nF since C2 needs to be 10 times larger and the inductance of the coil is around 3.9mH. We also used the NTD3055 and NTD2955 to make a NOT gate so that the output is an oscillating square wave. The resistors are used to buffer the signals. Placing a metal inside the magnetic wire loop will either increase or decrease the oscillator frequency as the metal affects the overall inductance of the coil, making this part one of the most essential parts to implement metal detection.

Audio Amplifier Circuit and Red LED (Refer to Appendix B for Image)

After the .wav file is stored as a digital memory at 25Q32, we used the built-in pulse width modulation DAC(digital to analog converter) from the PIC32 microcontroller to convert digital to analog signal in the form of sound. The sound is amplified by LM386 which is an operational amplifier. The red LED was added to indicate that the metal detector was still searching for a reference frequency within the 5 second time window. More details about how the red LED blinks are provided in the Software Detailed Design section.

LCD and Voltage Regulator (Refer to Appendix C for Image)

The HD44780 LCD is used as a visual indicator to display “Place a metal” if there is no object that could increase or decrease the oscillator frequency out of the TOLERANCE bounds.

“Small/Large Ferrous/N Ferrous Object Detected” would be displayed if the frequency is outside of the TOLERANCE bounds depending on how large the frequency decreases or increases.

There are two voltage converters that are used for this project. The LM7805 converts 9 volts from the battery to 5 volts, which then goes to the input of the next converter, MCP17003320E.

The MCP17003320E then converts the 5 volts into 3.3 volts to power the PIC32 as it can only accept a range of 2.3-3.6 volts [3].

PIC32MX Microcontroller and BO230XS Board (Refer to Appendix D for Image)

The PIC32MX130F0648-I/SP is a 32-bit microcontroller that can be programmed using C. The BO230XS is a custom-made board built by UBC that was used to flash programs into a microcontroller, including the PIC32. After flashing the program into the microcontroller, we can use the battery to power up the microcontroller so that our metal detector is portable.

3.6.2 Software Detailed Design

Acquiring Reference Frequency (Get_Ref_Freq) Function:

```
while (measure_count < MEASURING_COUNT)
{
    count = GetPeriod(100);
    if (count > 0)
    {
        LATBbits.LATB10 = !LATBbits.LATB10; // Blink led on RB6
        T = (count * 2.0) / (SYSCLK * 100.0);
        new_freq = 1 / T;
        sum_freq += new_freq;
        measure_count++;
    }
    waitms(100);
}
```

Figure 3: The while loop inside our Get_Ref_Freq Function

To calculate the searching frequency whenever the detector starts up, this function is called before other tasks are carried out. To get the frequency, it calls the given GetPeriod function which accurately calculates the period of the colpitts oscillator. To ensure even more consistent readings, we called this function 50 times over a span of 5 seconds and took the average of all measurements, as shown above. While the frequency is being measured, an LED is blinked to

indicate that it is still in progress. Finally, the period was inverted and then returned to the main function.

Indicating Size and Type of Metal

```
if (live_freq > ref_freq) {    //non-ferrous
    if (freq_diff > SMALL_OBJECT_TOLERANCE) {
        printf("Large Non Ferrous: Live = %.0f Ref = %.0f\r",live_freq,ref_freq);
        Start_Playback(LARGE, LARGE_LEN);
        while (play_flag);
        Start_Playback(NON, NON_LEN);
        while (play_flag);
        Start_Playback(FERROUS, FERROUS_LEN);
        while (play_flag);
        Start_Playback(OBJECT, OBJECT_LEN);
        while (play_flag);
        Start_Playback(DETECTED, DETECTED_LEN);
        while (play_flag);
        LCDprint("Large N Ferrous", 1, 1);
        LCDprint("Object Detected", 2, 1);
        T1CONbits.ON = 1;
    }
}
```

Figure 4: An if statement from our main function showing what voicelines to play if its non-ferrous

After getting the reference frequency, the sound and LCD response can be programmed based on comparing the reference frequency(ref_freq), live frequency(live_freq), and the difference between them(freq_diff). If the live_freq is in between ref_freq-TOLERANCE and ref_freq+TOLERANCE, the metal detector will not recognize the object as metals usually increase or decrease until exceeding the bounds. If the live_freq exceeds those bounds, the code checks if live_freq is greater than ref_freq. If Live_freq is greater than ref_freq, that means a non-ferrous object is detected!; therefore, the code only needs to check if freq_diff is greater than SMALL_OBJECT_TOLERANCE or not to detect the metal's size. For detecting ferrous metals, live_freq has to be less than ref_freq as ferrous metals decrease the frequency of the oscillator. The same method for indicating the size of non-ferrous metals also applies to indicating the size of ferrous metals. After recognizing the size and the type of the metal, the code plays sound

using `Start_Playback(defined sound index, defined length)` and prints to the LCD using `LCDprint("string",line 1 / 2, 1)` as shown in the example above.

3.7 Solution Assessment

Considering the design specifications, our metal detector performed well. It was able to detect various metals correctly and quickly. The feedback was of satisfactory quality as well. The LCD and speaker both tell the user what metal is present under the detector. The volume of the speaker was also sufficiently loud. We did encounter some difficulty with detecting small objects far away from the detector and some fluctuations in the default searching frequency. We noticed that large metallic objects that appeared to be far from the coil did affect the frequency. We increased the oscillator's frequency to get rid of most of the fluctuation, and thoroughly tested frequency changes for different materials in order to accurately set thresholds for detection. Below are some of the numbers from our tests:

Frequency Change (Hz):

Ferrous:

Small: Quarter: - 70 Nail: -90

Large: Ferrite Rod: -810

Non-Ferrous:

Small: Looney: +110 Pen: +150

Large: Coke Can: +240 Speaker: +400

The strength of this design is that it can consistently report what type of metal is present near the detector with high sensitivity. Its weaknesses are that it is susceptible to false readings from other large objects in the proximity of the detector. It also isn't able to indicate if multiple types of metals are present, and can only differentiate between four different cases.

4. Life-Long Learning

For this project, we found that ELEC 211 was extremely beneficial in terms of giving us background knowledge regarding how magnetic fields and current within the coil interact with each other. Additionally, CPSC 259 from our first semester gave us a much needed refresher on C programming for ELEC 291. A knowledge gap that we encountered was trying to understand how to properly measure frequency and how it responds to ferrous and non-ferrous metals. The concept of a metal detector was quite foreign to us prior to starting this project. From reading online sources and a report that was posted on Canvas, we were able to understand different methods for implementing a frequency meter. From doing so, we also found out how to properly set a threshold frequency for metal sizing and what changes in frequency would indicate a ferrous or non-ferrous metal.

5. Conclusion

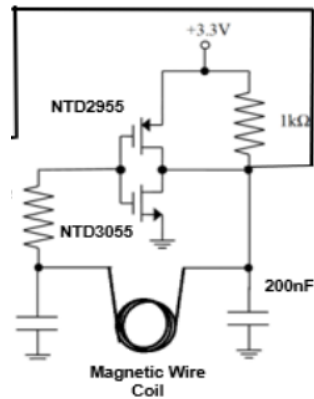
Overall, our group designed a metal detector that could be optimized for individuals who are either hearing or visually impaired. The basic requirements of this project forced us to implement active voice feedback to identify the detected metals. On the other hand, for the hearing impaired, we added an LCD display that would indicate the type of metal and when no metal was detected near the coil. We also added a red LED that would blink when indicating that it was calculating a reference frequency. We encountered challenges such as properly implementing both LCD and voice feedback concurrently, debugging radical frequency changes, and trying to understand which ports to connect for the PIC32. In its entirety, considering the time it took us to brainstorm ideas, build hardware, code, and debug, we dedicated around 50-60 hours.

References

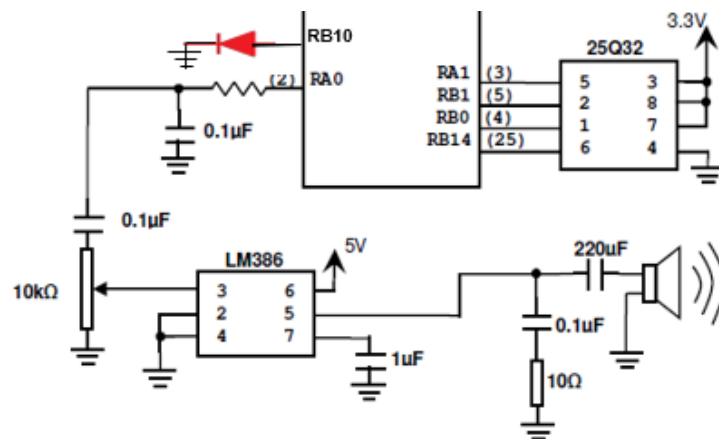
- [1] Chtchedrine, A., & Kolokolov, Y. (2001). Frequency Meter Metal Detector (Rep. No. 130). Circuitcellar.
- [2] (2015) . 32-bit Microcontrollers (up to 256 KB Flash and 64 KB SRAM) with Audio and Graphics Interfaces, USB, and Advanced Analog. Microchip.

Appendices

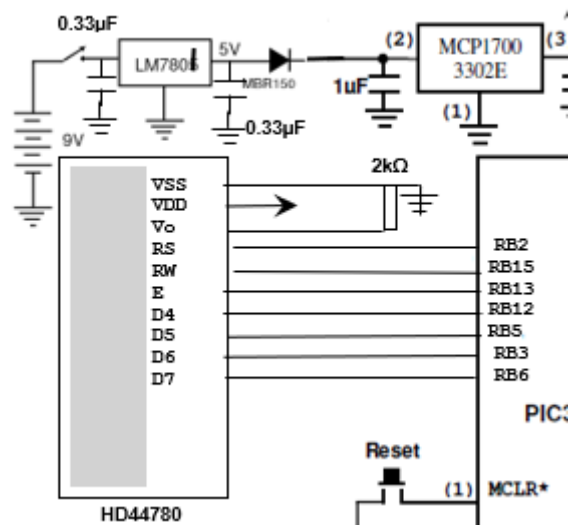
Appendix A



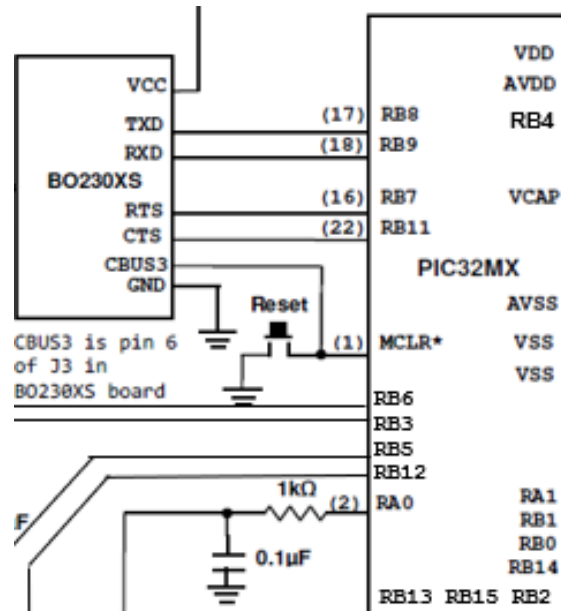
Appendix B



Appendix C



Appendix D



Bibliography

Ellingson, Steven W. (2018) Electromagnetics, Vol. 1. Blacksburg, VA: VT Publishing.

<https://doi.org/10.21061/electromagnetics-vol-1> CC BY-SA 4.0