University of Khartoum – Faculty of Engineering

Department of Electrical and Electronic Engineering

Microprocessor Systems design

Three-Dimensional Indoor Localization System

Prepared by:

Ahmed Mansour Mohammed	134009
Amr Muhammad-Alameen Khalifa	135058
Eltayed Khalid Eltayeb	134022
Montaser Fathelrhman Hussen	134098
Mosab Ahmed Diab	134097

Three-Dimensional Indoor Localizing System

Objectives

To build a system capable of determining its location within a building.

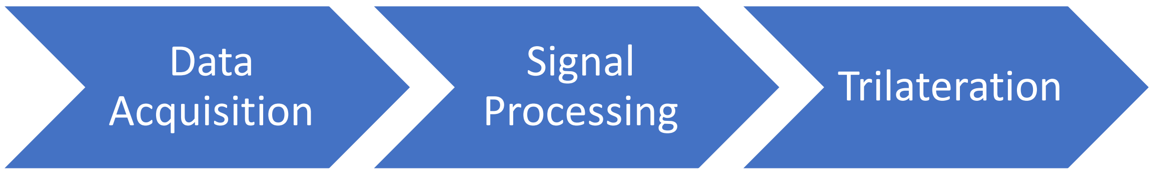
Architectural Design

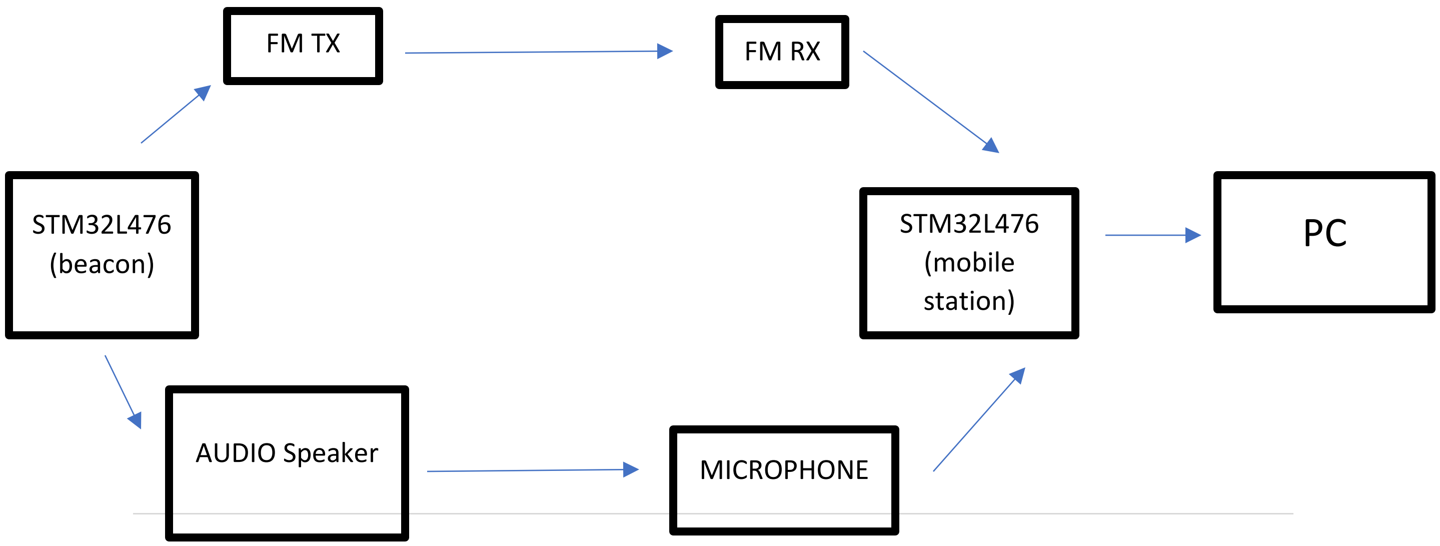
Overview

The system subsists of 3 or more beacons at known positions and a mobile station. To determine the position of the mobile station the mobile station calculates its distance from each of the beacons, with this information in addition to the knowledge of the positions of the beacons the mobile station is able to calculate its position by a procedure called trilateration.

To determine the distance of the mobile station from a beacon the beacon releases at the same time both an FM signal and an audio signal. The FM signal travelling at the speed of light arrives instantaneously while the audio signal travelling at a much slower speed takes a measurable amount of time. By calculating the delay between the arrival of the FM signal and the audio signal at the base station the mobile station is able to calculate its distance from the beacon.

The following flow chart describes the operation of the mobile station





Hardware Components:

1. STM32L476G Discovery board.
2. FM transmitter – FM receiver.
3. Speaker – Microphone.

Hardware tools

1. Signal generator.
2. Oscilloscope.

Software tools:

I. STM32 CubeMX.
II. STM32 ST-LINK Utility.
III. Keil µVision5.
IV. MATLAB & Filter Builder toolbox.
V. Python tool kit for scientific computing (scipy, numpy , and matplotlip).

Data Acquisition process:

Sending FM and Audio signals:

For the message wave, we used a sinusoid signal because of its convenience for FM transmission and its ease of generation.

Zadoff-Chu sequence

The Zadoff-Chu signal was selected and it is suitable for this application because it has the following properties:

1. Can be generated easily using the formula.

2. Any shifted version of the signal is orthogonal to the original signal. i.e. gives zero correlation.

We have used a Zadoff-Chu signal with a base of 5 and a length of 1023 sample.

$$x_u(n) = \exp\left(-j\frac{\pi u n(n+1+2q)}{N_{\text{ZC}}}\right),$$

where

$$0 \le n < N_{\text{ZC}},$$

$$0 < u < N_{\text{ZC}} \text{ and } \gcd(N_{\text{ZC}}, u) = 1,$$

$$q \in \mathbb{Z},$$

$$N_{\text{ZC}} = \text{length of sequence}.$$

In our Case:

n= 5

Nzc = 1023

Before using the board, we needed to test it and test our build chain (STM CubeMX code generator with keil µvision IDE). With the guide of the reference manual and *HAL* library we successfully loaded and tested a blinking LED program using a *HAL* GPIO library function:

HAL_GPIO_TogglePin (GPIOB, GPIO_pin_2);

STM32L4 has a LED internally connected to port B pin 2.

<u>Experiment: generating a sine wave using the STM32L476VG internal Digital to Analogue Convertor:</u>

<u>STM32L476VG DAC:</u>

The L4 DAC module is a 12-bit, voltage output digital-to-analog converter. It can be configured in 8 or 12-bit mode, the data could be left or right aligned. It has two output channels each with its own converter with internal or external reference voltage options. The equation used for an analogue voltage output:

Analog output = Digital input * (V_{REF} /4095) V_{REF} = 3.3 volts

With the guidance of the reference manual and the *HAL* library manual, we started by testing the DAC functionality by first setting a constant voltage at PA5 (DAC1_OUT2 pin) using a *HAL* library function:

HAL_DAC_SetValue (&hdac1, DAC_CHANNEL_2 , DAC_ALIGN_12B_R, 1200);

Then using a sine wave look up table with predefined resolution and frequency we output it using a *HAL* library delay function to separate between each sample and the next one. We ran into the problem that the minimum delay possible with the delay function is one millisecond which put constraints on the frequency of the output leaving us with a maximum frequency of one hundred Hz for ten samples.

We overcame this problem using timers to separate between each sample thus allowing us to use higher frequencies.

$$\textbf{Timer frequency} = \frac{\textbf{Internal frequency}}{\textbf{prescaler}} \qquad \boldsymbol{Time\ to\ interrupt} = \frac{\boldsymbol{Counter\ period}}{\boldsymbol{Timer\ frequency}}$$

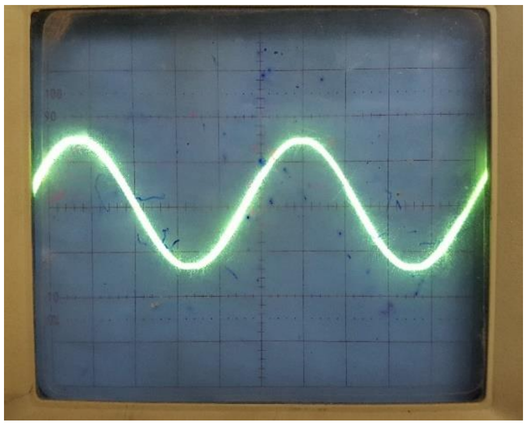
Using the above equations and with no prescaler and internal frequency 48 MHz with look up table with 100 sample and 480 as a counter period we successfully generated a one KHz sine wave for FM modulation and transmission and used the oscilloscope to test it.

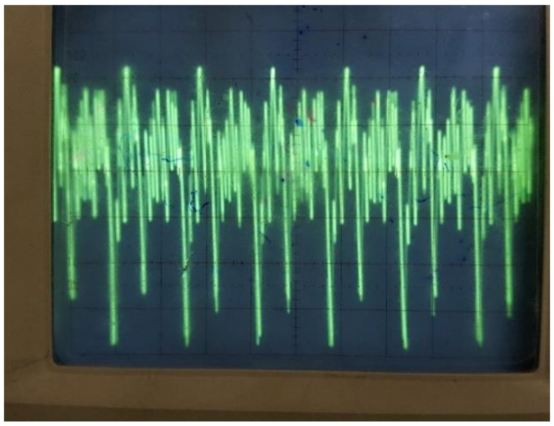
Using a *HAL* library function: **HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)**

Experiment: generating an audio signal simultaneously with the FM signal using the STM32L476 internal Digital to Analogue Convertor:

A Zad-off Chu sequence of 1023 sample was generated and the samples were output using the DAC and the time between each sample was set using the L4 timer.

We ran into the problem that PA4 (DAC1_OUT1 pin) was not extended out of the board thus we couldn't generate the two signals simultaneously. We overcome this problem by using the option of connecting the DAC channel_1 output to the internal OpAmp and operated it in follower mode then output the signal at PA3 (OPAMP1_OUT pin).





Experiment: generating an audio signal simultaneously with the FM signal after a signal to warm up the microphone and the speaker:

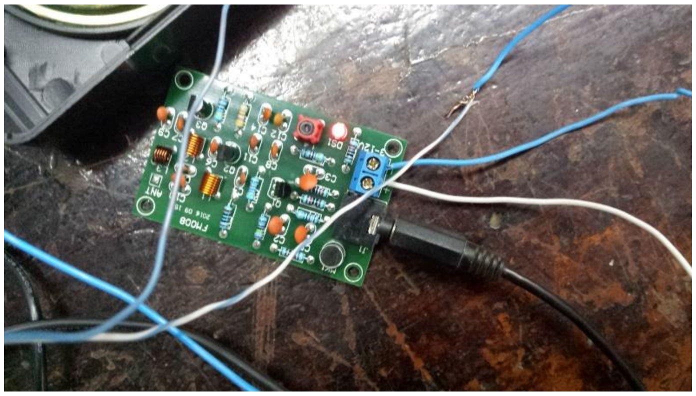
We ran into the problem that the speaker and microphone need time to warm up before operating correctly, so we used two timers one for generating the warm up signal which is a sinusoid with sufficient frequency and the other one for generating the zad-off Chu sequence and the FM signal.

FM

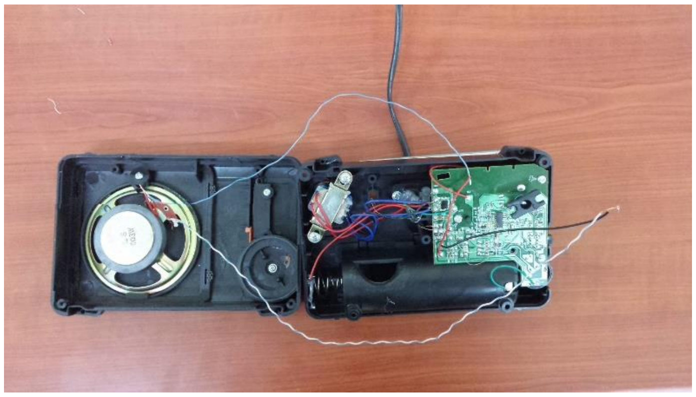
The goal is to transmit and receive a sinusoidal signal and detect its arrival by the microcontroller.

A proper frequency of transmission was chosen were there was no interference from any other transmission station. A diode was placed between the receiver's output and the microcontroller's input to clip off reverse (negative) voltages, this will not cause any delay issues because the sinusoidal transmission starts at the positive half cycle.

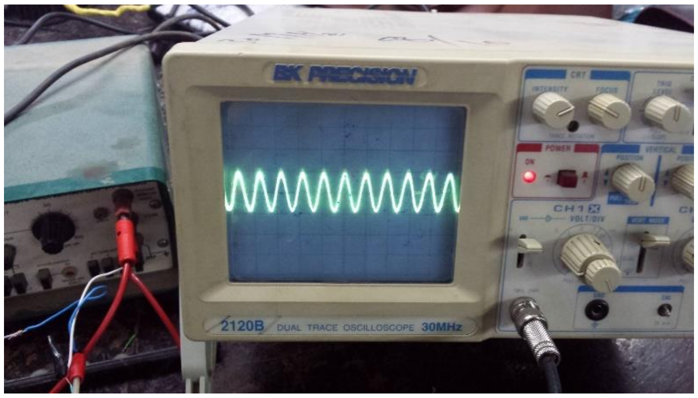
FM transmitter:



FM receiver:

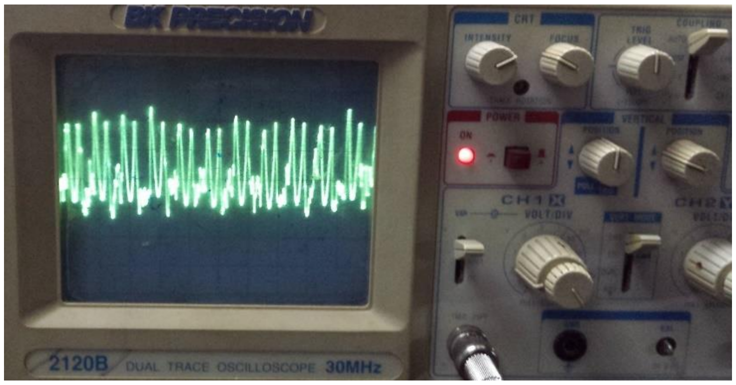


Signal to be transmitted:



F = 1 KHz V_{pp} = 4V

Demodulated and received signal:



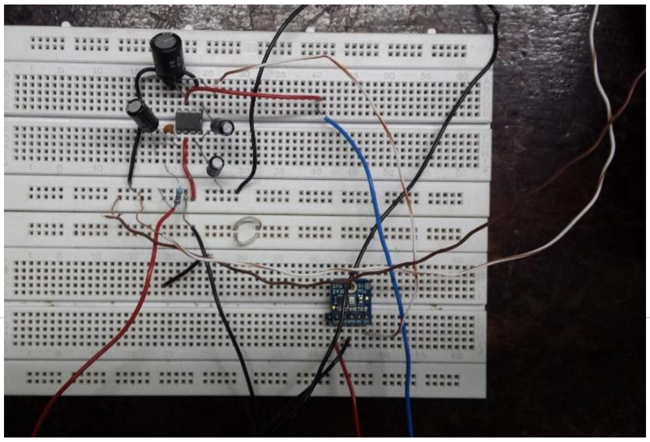
F = 1 KHz V_{pp} = 4V

Audio

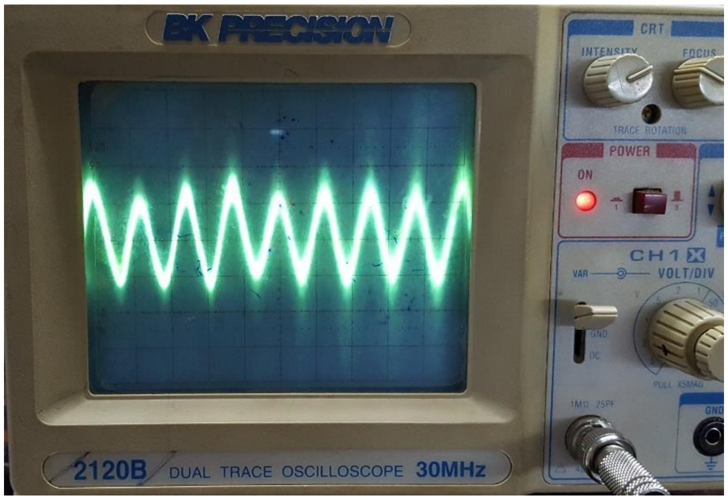
The maximum frequency that the microphone can receive is 10 KHz so operation is in the audible range. Also the maximum voltage on the output of the microphone is 200mV$_{pp}$ so an amplifier was used. The chosen amplifier is an LM386 audio power amplifier which has a bandwidth of 300KHz.

The chosen gain is 200.

Microphone and amplifier configuration:

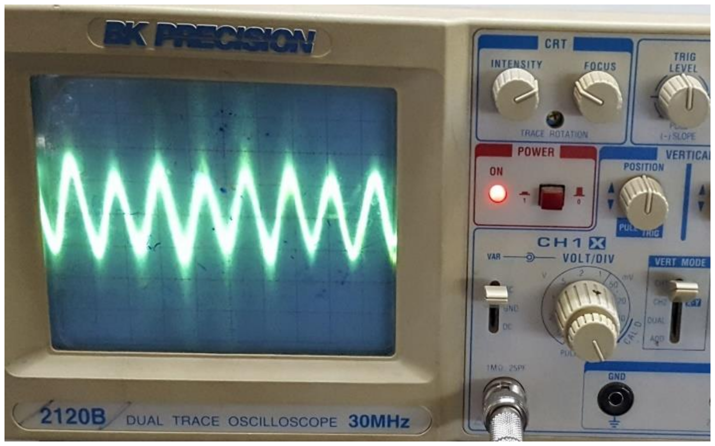


Microphone output to a sinusoidal input:



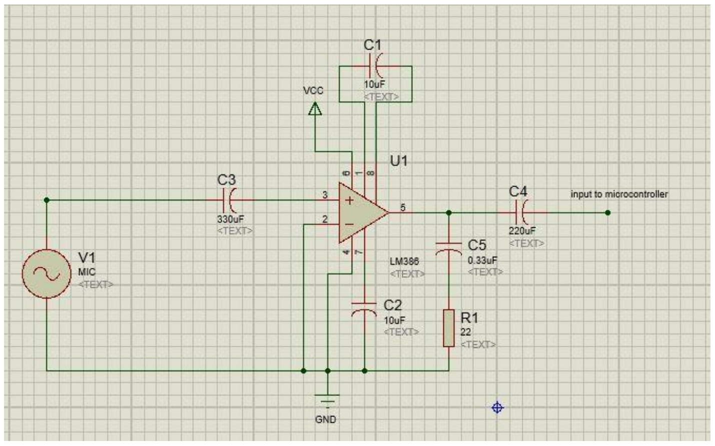
F = 1KHz V_{pp} = 10mV

Amplifier output with the microphone output as input:

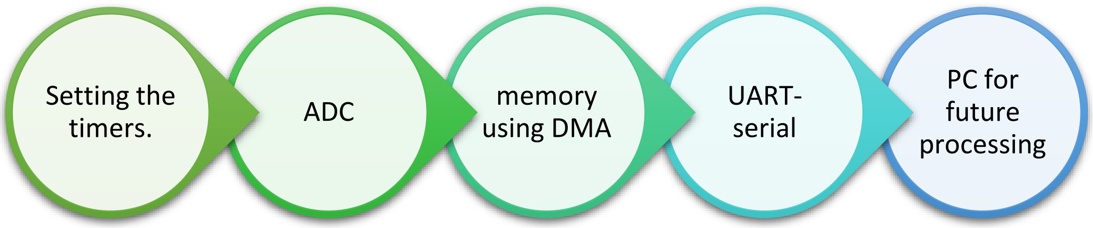


F = 1KHz V_{pp} = 2V

Circuit Schematic:



Receiving Audio signal:

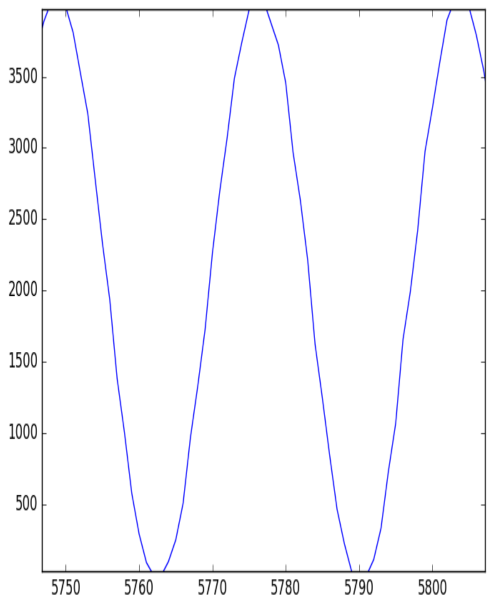
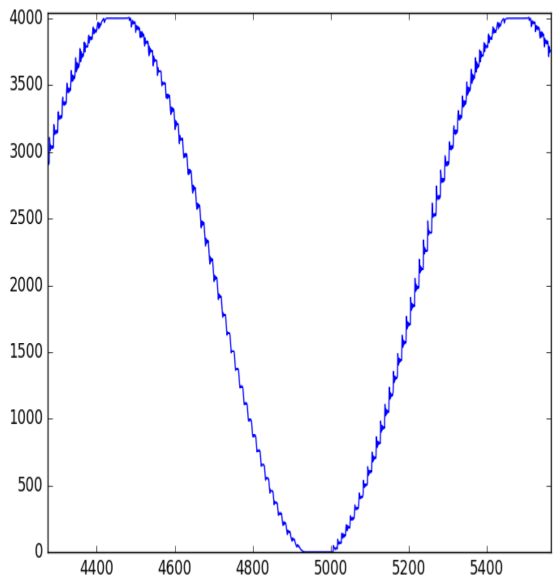
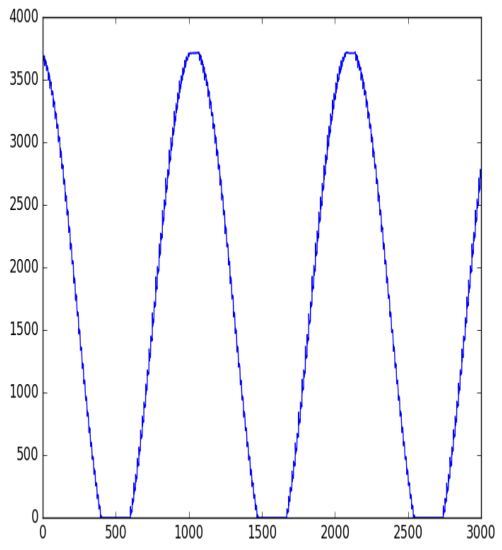
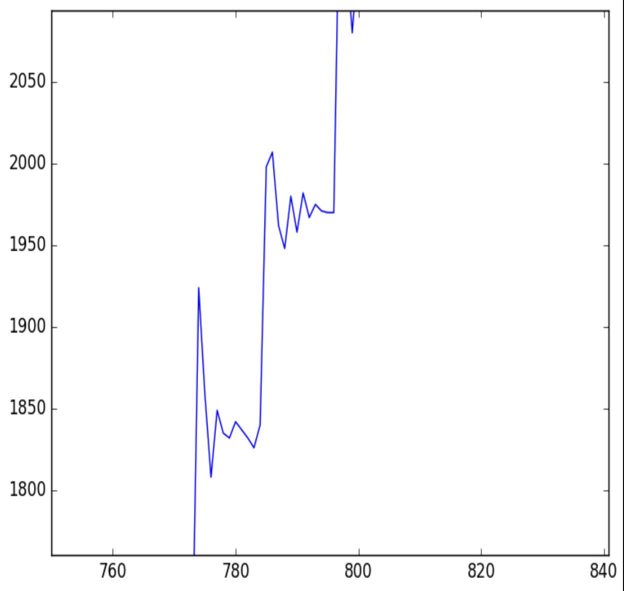


Data Acquisition pipeline

Tests and experiments:

1. Read Potentiometer value using ADC (software triggering)
2. Send character from microcontroller to PC using USART and show it in serial monitor
3. Blinking LED by timer
4. Read Potentiometer value using ADC triggering by timer and send it to the PC by USART
5. Generate square wave signal (10 Hz), convert it using ADC software interrupt (sampling rate 40 Hz) triggering by timer and send it to PC
6. Generate square wave signal (100 Hz), convert it using ADC software interrupt (sampling rate 400 Hz) triggering by timer and send it to PC
7. Generate square wave signal (10 KHz), convert it using ADC software interrupt (sampling rate 40 KHz) triggering by timer and send it to PC
8. Convert sinusoidal wave (1 KHz), convert it using ADC software interrupt (sampling rate 4 KHz) triggering by timer and send it to PC
9. Convert sinusoidal wave (10 KHz), convert it using ADC software interrupt (sampling rate 40 KHz) triggering by timer and send it to PC
10. Convert sinusoidal wave (10 KHz), convert it using ADC software interrupt (sampling rate 40 KHz) triggering by timer and send it to PC to plot output.
11. Convert sinusoidal wave (10 KHz), convert it using ADC, Direct Memory Access (DMA) (sampling rate 40 KHz) triggering by timer and send it to PC to plot output.
12. Convert ZADOFF-CHU) wave, convert it using ADC, Direct Memory Access (DMA) (sampling rate 40 KHz) triggering by timer and send it to PC to plot output.

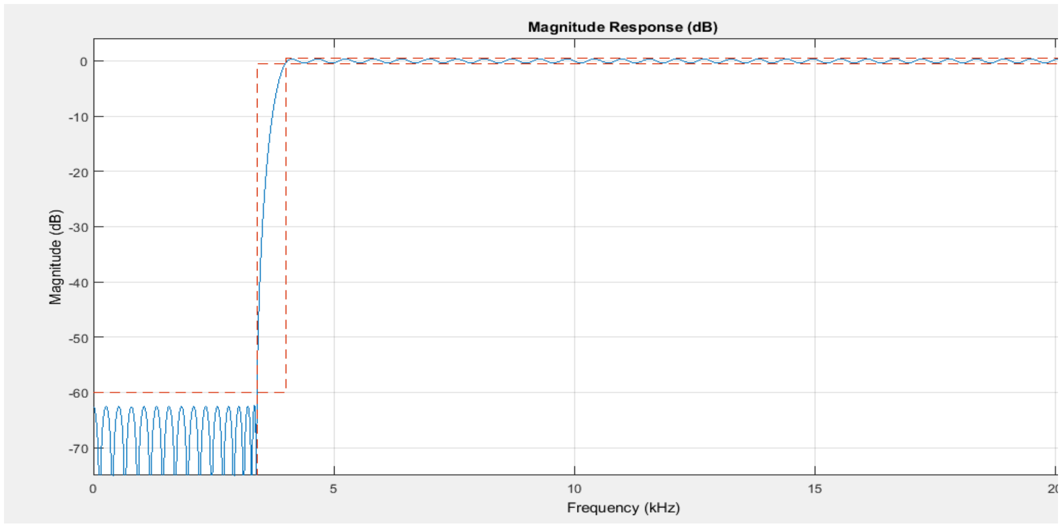
Results

 sinusoidal wave (10 KHz)ADC output (sampling frequency 10 KHz), triggering by software	 ZADOFF-CHU (1 KHz) ADC (sampling frequency 10 KHz) output , triggering by software
 Zadoff- Chu ADC output (sampling frequency 50 KHz), using DMA	 Zoomed in ADC (sampling frequency 50 KHz) using DMA

Filtering and Correlation

After the signal has been received and converted to digital values it had been filtered from noise. We have used FIR high pass filter that has a stop band frequency of 3.4 KHz and a pass band frequency of 4 KHz. The filter was designed with the aid of MATLAB filter builder tool.

The filter coefficients generated with the tool were taken and convolved to filter out the signal, then the filtered signal was correlated with the original signal and the peaks were found.



The equation used to find the distance is:

$$(i.of.p\ in\ c\ between\ recieved\ signal\ - i.of.p\ in\ c\ w\ original\ signal\ ) * \frac{speed\ of\ sound}{sampling\ rate}$$

i.of.p in c.w = index of peak in correlation

Testing and performance evaluation

In STM32 board, the correlation used directed convolution (not FFT).

For 1023 short int array of Zadoff-Chu sequence correlation took about 1 second.

For 1023 short int array of Zadoff-Chu sequence correlation took about 34 second.

Notice that DSP instructions were not used.

Trilateration

Trilateration is the locating of a point with knowledge of its distances from 3 or more known points. This boils down to solving M equations with 3 unknowns. If M>=3 then the problem is known as the non-linear least squares problem (NLS). There are many methods of solving these problems. In this design, we used the Newton gauss method. The Newton-Gauss method is an iterative method where

$$x_{n+1} = x_n - [J^t * J]^{-1} * J^t * F(x_n)$$

Where J is the Jacobian Matrix and F is function vector.

In the case of trilateration

$$F = \begin{bmatrix} (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 - {d_1}^2 \\ \vdots \\ (x - x_m)^2 + (y - y_m)^2 + (z - z_m)^2 - {d_1}^2 \end{bmatrix}$$

$$J = 2 * \begin{bmatrix} x - x_1 & y - y_1 & z - z_1 \\ \vdots & \vdots & \vdots \\ x - x_m & y - y_m & z - z_m \end{bmatrix}$$

The algorithm was executed until convergence which was reached when the step taken became less than one centimeter.

Experiment:

Arbitrary values were set for the locations of the beacons and their distances from a specified point were calculated. This information was fed to the trilateration algorithm on the STM32L4 board and the algorithm successfully returned the correct specified point within 5ms. Low amplitude random noise was added to the distances and algorithm still successfully converged to a point close to the original specified point.

Results

We have done ranging for one beacon only.

Same method will be used for the three or more beacons.

peak at sample	calculated distance	Actual distance	Error in cm
425	.8	.77	3
750	1.16	1.19	3
870	1.43	1.41	2
1090	1.74	1.69	5
1570	2.46	2.38	8
3358	5.52	5.59	7

This error is mainly due to ambient noise and analog to digital conversion accuracy.

Also, the state of the ambient air affects the results. We get Slightly different results when we turn on the fans in the lab. We hypothesize that it changes the sonic speed in the air.

Conclusion

1. In order for the localization system to work efficiently, high quality equipment must be used. (In terms of frequency response and response time).
2. Speakers with High gain are needed as for the system to work properly in long distances.
3. Zadoff-Chu orthogonality property is tolerant to the filtering we have used. And both the real and imaginary parts of the signal have this property.
4. ADC using Direct Access Memory (DMA) is more efficient than regular (polling and interrupt) ADC, because it takes less time to store data.

Bill of Materials

Part	Quantity
STM32L476G Discovery board.	2
FM transmitter (FM008)	1
FM receiver	1
Speaker	2
Microphone (SPW2430)	2
Resistors	1
Capacitors	4
Diode (1N4007)	1
LM386 Audio Power Amplifier	1