ISTANBUL TECHNICAL UNIVERSITY

ELECTRICAL – ELECTRONICS ENGINEERING FACULTY

DESIGN AND IMPLEMENTATION OF AN EMERGENCY EVACUATION SYSTEM

BSc Thesis by

Gediz Morgil

Department: Electronics and Communication Engineering

Programme: Electronics and Communication Engineering

Supervisor: Assoc. Prof. Dr. Sıddıka Berna ÖRS YALÇIN

MAY 2016

ISTANBUL TECHNICAL UNIVERSITY

ELECTRICAL – ELECTRONICS ENGINEERING FACULTY

DESIGN AND IMPLEMENTATION OF AN EMERGENCY EVACUATION SYSTEM

BSc Thesis by

Gediz Morgil

040120022

Department: Electronics and Communication Engineering

Programme: Electronics and Communication Engineering

Supervisor: Assoc. Prof. Dr. Sıddıka Berna ÖRS YALÇIN

MAY 2016

ACKNOWLEDEGEMENT

First of all, I would like express my gratitude to my supervisor Assoc. Prof. Dr. Sıddıka Berna Örs Yalçın for sharing her knowledge and time with me to complete this thesis. I would not be able to finish this project without her help.

I also would like to thank to Alp Burak Pehlivan, Bahadır Gün and Ramazan Aktaş for giving their precious time to help me to solve some of the software problems. I have saved lots of time with their help.

Finally, I would like to thank to my family for all of their support and love for me. They have guided me with their experience throughout my life and helped me to come this point.

MAY 2016 Gediz Morgil

INDEX

ABBREVIATIONS

GPS : Global Positioning System

GLONASS : Global Navigation Satellite System

RFID : Radio Frequency Identification

ECDH : Elliptic Curve Diffie Hellman

AES : Advanced Encryption Standard

BLE : Bluetooth Low Energy

RSSI : Received Signal Strength Indication

XAML : Extensible Application Markup Language

BaaS : Building as a Service

USB : Universal Serial Bus

IDE : Integrated Development Environment

WPF : Windows Presentation Foundation

FSK : Frequency Shift Keying

GFSK : Gaussian Frequency Shift Keying

OOK : On-Off Keying

SPI : Serial Peripheral Interface

RF : Radio Frequency

IoT : Internet of Things

ARM : Acorn Risk Machine

ICSP : In Circuit Serial Programming

MOSI : Master Out Slave In

MISO : Master In Slave Out

SCK : Serial Clock

SS : Slave Select

EEPROM : Electrically Erasable Programmable Read-Only Memory

ID : Identification

SDK : Software Development Kit

FIGURE LIST

TABLE LIST

ÖZET

Acil bir durumda insanları kurtarmak ve konumlarını belirlemek çok önemlidir. Bir binayı tamamen tahliye etmek için binadaki insan sayısı ve konumları bilinmelidir. Bu projede acil durum tespiti ve kapalı alan konumlandırması yapan bir sistem tasarlanmaktır. Projenin amacı daha önce yapılmış olan alt parçaları birleştirmek, sensörler ekleyerek geliştirmek ve kurtarıcı ekibin kolaylıkla kullanabileceği bir ara yüz tasarlamaktır.

RF haberleşme kapalı alan konumlandırma çözümlerinden biridir. Bu projede mesafe ölçümü için RFID etiketler kullanılmıştır. RFID etiketler duvara takılır ve bir okuyucu tarafından dinlenirler. Bir RFID okuyucu sinyal aldığı zaman sinyalin gelme mesafesi, sinyalin RSSI değerinden anlaşılabilir. Konumlandırma için 2 boyutlu trilaterasyon algoritması kullanılmıştır ve algoritmanın çalışması için 3 tane etiketin mesafe bilgileri ana bilgisayara gönderilmelidir. Okuyucu bu bilgiyi Düşük Enerjili Bluetooth (Bluetooth Low Energy - BLE) kullanarak merkezi bilgisayara iletir. Sistemin güvenliği için bu haberleşmenin kriptolanması gerekir. Projede anahtar değişimi için Eliptik Eğrili Diffie-Hellman (Elliptic Curve Diffie-Hellman - ECDH) anahtar değişim protokolü kullanılmıştır. Üstün Şifreleme Standardı (Advanced Encryption Standard - AES) şifreleme yöntemi ise gönderilen verilerin şifrelenmesi için kullanılmıştır. Merkezi bilgisayar okuyucu ile bir Bluetooth dongle kullanarak haberleşir. Kullanıcı ara yüzü Windows Presentation Foundation kullanılarak yazılmıştır. Yazılan bu uygulama 3 etiketin mesafe bilgilerini alır, konumu hesaplar ve mekânın bir krokisi üzerinde konumu gösterir.

SUMMARY

It is very important to be able to save people in case of an emergency and the location of these people must be known in order to rescue them. In order to totally evacuate the building during the emergency, the total number of people and their locations are needed to be known. In this project, it is planned to design and implement an emergency detection and an indoor localization system. The purpose of the project is to combine the sub-blocks which were designed and implemented before, improve them by adding real sensors and design a good user interface which can be used by the rescue crew. Locating people can save time and resources.

Radio frequency communication is one of the solutions for indoor localization. In this project, RFID tags are used to measure distance. RFID tags are put on the walls and a reader listens to them. When an RFID module receives a signal, distance can be calculated by the signals RSSI value. 2D trilateration is used and the distance information of 3 tags must be sent to a central computer for the localization calculations. Reader sends this information using Bluetooth Low Energy. The system must be safe and should not be vulnerable to attacks. Therefore, cryptography is used in this communication. Elliptic Curve Diffie-Hellman (ECDH) is used for the key sharing and Advanced Encryption System (AES) is used for sending the information. Central computer communicates with the reader with a BLE dongle. User interface application is written in Windows Presentation Foundation (WPF). This application gets the information of 3 tags and calculates the location of the reader. This location is visualized on a map of the area.

1. INTRODUCTION

Locating every person in a building is very important to save people in case of an emergency. [1] An indoor localization system can solve the problem and help to understand if a building is totally evacuated or not. Outdoor localization systems are used for years. Global Positioning System (GPS) [2] and Global Navigation Satellite System (GLONASS) [3] are accurate and trusted systems for outdoor localization. However, they cannot be used in indoor because they use satellites and need a direct line of sight. Walls, trees and other objects prevent them to work efficiently. Because of these reasons, other solutions are tried to be used in indoor localization.

Radio Frequency Identification (RFID) [4] and Bluetooth [5] are two common solutions for indoor localization. RFID technology is used in this project. In an RFID system, there are tags and readers. Tags transmit a message wirelessly and the reader reads the message when it receives it. When a message is received, the strength of the distance can be calculated and this strength is used for estimating the distance. This method is called Received Signal Strength Indicator(RSSI) [6] and used in this project. All the tags and the reader in the project are active which means they use a battery.

Evacuating a building is a very serious task and the system must be very secure and reliable. Therefore, the important messages must be encrypted. A central computer calculates the locations and decides if there is an emergency situation so the communication between the central computer and the reader is encrypted in this project. A key is needed for the encryption algorithms and this key must be shared between the two devices. All the communication happens through an insecure wireless channel so this key exchange also must be encrypted. Elliptic Curve Diffie Hellman (ECDH) [7] is used for key exchange protocol. After the both parties agree on the same key, they can send the messages securely. Advanced Encryption Standard (AES) [8] is used for this data sharing process. Bluetooth Low Energy (BLE) [9] is chosen to be used in the secure communication because it consumes less power compared to other methods. The reader is the slave BLE device and the computer is the master BLE device.

Trilateration is used for the location algorithm [10]. Trilateration finds the location of an object using circle geometry. Trilateration is chosen because RSSI value gives the

distance data rather than the angle data. In the project, the position of all the tags are known. Reader calculates its distance to 3 of the closest tags. After that, 3 circle equations are established and the position is found solving the equations. Localization algorithm is realized on a C# project.

All the data must be visualized for a practical solution. A user interface is designed using Extensible Application Markup Language (XAML) [11] and C#. This interface has the map of the area and the tags are shown on the map. When an emergency situation occurs, the system shows the people in the area. A sensor is needed to understand the emergency situation. A temperature sensor is used in the project. When the temperature is above a certain degree, reader starts to send the location data to the central computer. All of the system is visualized in the Figure 1.1, Figure 1.2 shows the establishment of the system in a room.

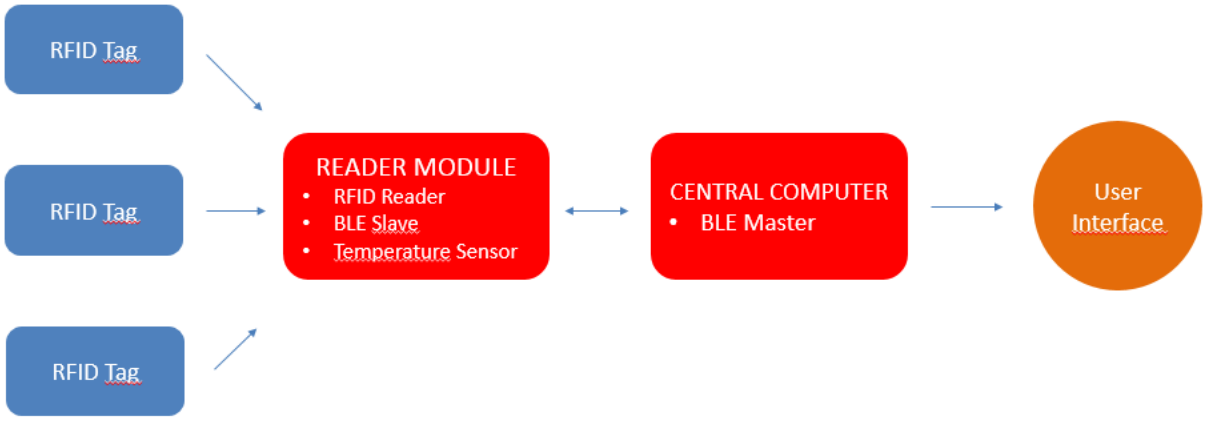

Figure 1.1: System Overview

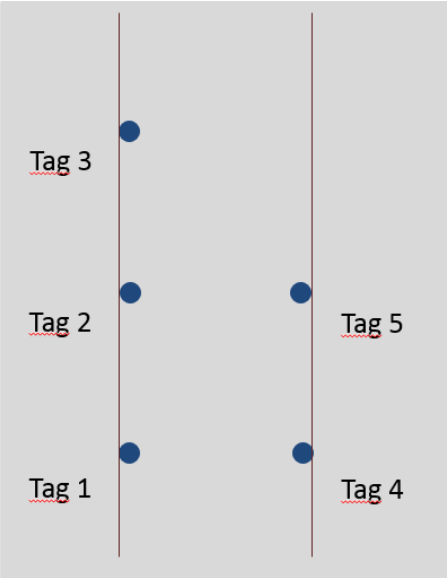

Figure 1.2: Visualization of RFID tags

This project is the continuation of the previous year's graduation projects. 5 people has been contributed to different sub-blocks to realize the project. This project uses all the previous sub-blocks and combines them into a one single working system with the addition of a temperature sensor and a user interface. Work flow of the sub-blocks is listed in the Figure 1.3. Driving RFID modules by using Arduino and processing RSSI values is realized by Ramazan Aktaş [12], secure Bluetooth communication is realized by Bahadır Gün [13]. Implementation of a localization algorithm is realized by Onur Azbar [14] and Alp Oran [15]. Driving RFID modules using Microblaze is realized by Oğuzhan Çik [16] and designing a user interface and combining all sub-blocks is realized in this project.

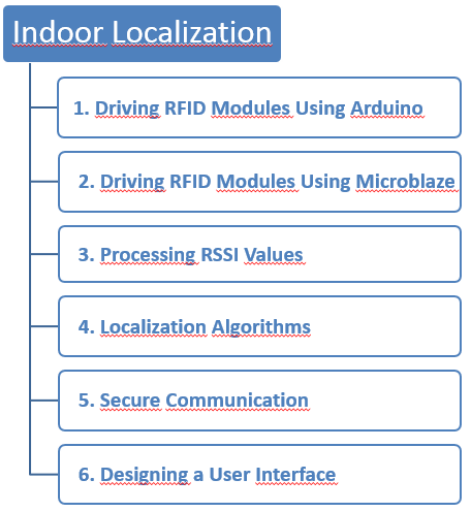

Figure 1.3: Workflow of the Project

There is a European Union project called Building as a Service (BaaS) [17]. Buildings are managed, controlled and monitored through different systems. Baas aims to maintain the intelligence of these systems in one place and create a software platform to provide a reference system for the future buildings. BaaS project covers all the intelligence technologies like secure access to a building, light control and indoor air quality services. Building evacuation in case of an emergency is also a part of the project. Indoor localization offers a solution to this problem. Detecting the number of people in a building and locating them in a secure way is suitable for the aims of the project. This graduation project is designed to be a part of the BaaS project.

2. BACKGROUND INFORMATION

2.1. Arduino

Arduino is an open-source platform for electronic projects [18]. It is designed for interacting with the physical world and creating projects easily. It consists of hardware and software elements. Arduino boards has a microcontroller on them for the main processes and a universal serial bus (USB) [19] to serial converter in order to communicate with the computer. Arduino's programmes are written in C/C++ language. Arduino has its own Integrated Development Environment (IDE) and the code is written in this environment.

In this project Arduino is used to drive the Bluetooth and RFID modules. Arduino is chosen for this task because it has libraries for both the Bluetooth and RFID modules which makes it very easy to use.

For the localization algorithm, Arduino Nano [20] is used to drive 5 slave RFID modules because it is one of the smallest Arduino's and easy to hang on the walls. A picture of Arduino Nano is in Figure 2.1.

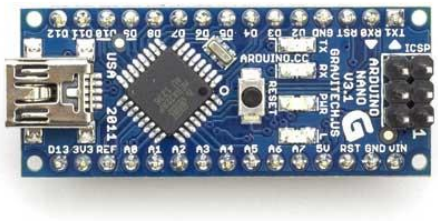

Figure 2.1: Arduino Nano [20]

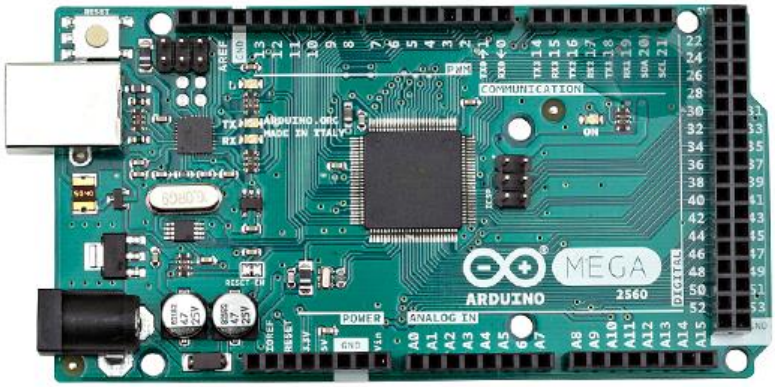

Figure 2.2: Arduino Mega [22]

Reader device's code size is larger than the slave device's and Arduino Nano's flash memory of 32 KB is not enough for that. Arduino Mega [21] has a flash memory of 256 KB which is large enough for the reader. Therefore, Arduino Mega is used as the reader. A picture of Arduino Mega is in Figure 2.2.

2.2 Radio Frequency Identification (RFID)

In RFID technology, radio waves are used to identify objects or people. [4] Tags have unique identifiers and they are assigned to an object in order to track it. An RFID system consists of three components: tags, tag readers and a host system. Tags emit signals and readers reads those signals. Then a host system processes the information.

RFID tags can be active or passive. Active tags use batteries to work. On the other hand, passive tags do not use batteries. Passive tags produce power with the coming RFID signals. RFID technology is used in many parts of our life. It can be used to track inventories, pay money, use public transport and access to a building.

2.2.1 RFM23

In this project HopeRF's RFM23 modules are used [23]. RFM23 is a low cost 433/470/868/915 MHz RF module. It has +20dB which ensures extended range. It also has a built in antenna for frequency hopping. It can use Frequency Shift Keying (FSK), Gaussian Frequency Shift Keying (GFSK) or On-Off Keying (OOK) modulation to transmit the data and input supply voltage changes between 1.8-3.6 Volts.

RFM23 uses Serial Peripheral Interface (SPI) [24] to communicate with the Arduino. It uses SPI0 mode, most significant bit first and 1Mhz default speed. Arduino has a special library for this module and this library is used in the project. Module can be used as a transceiver or a receiver. Because of this property it is used as tag and a reader. In order to use the module, it was printed on a circuit board and added an external antenna housing. The module is in Figure 2.3. Also an antenna is used for better results.

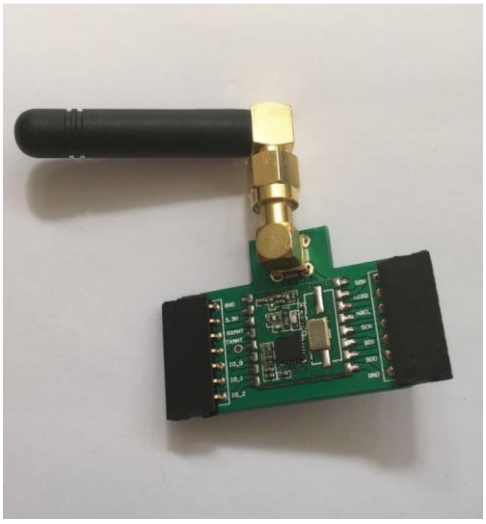

Figure 2.3: RFM23 Module

2.2.2 Received Signal Strength Indicator (RSSI)

RSSI is a standard to measure the power of a signal on a Radio Frequency antenna [6]. When a receiver module receives a message in its antenna, this message's power can be measured. Generally, RSSI indicates the quality of the signal and if the devices get closer, the value becomes higher. RSSI is an integer between 0-255 and it does not have a unit. It is a relative number. Signal strength can also be measured in decibels as dBm. dBm can be used in calculations so the RSSI values must be converted to dBm. Every manufacturer gives their own RSSI tables. There is a graph in RFM23's datasheet which shows the how to convert the RSSI value to power in dBm. The graph is in the Figure 2.4. This graph is used in the project.

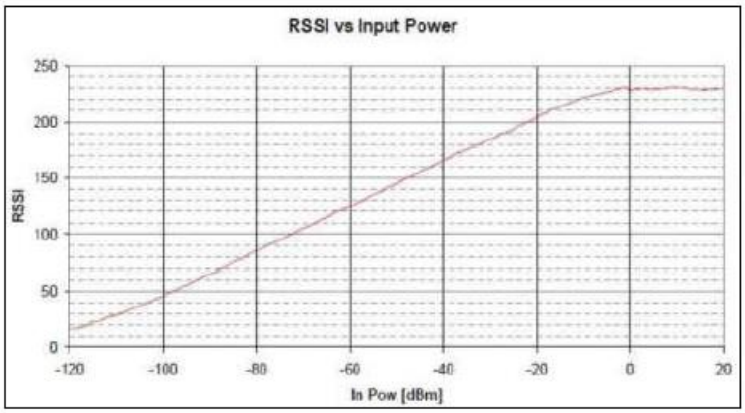

Figure 2.4: RSSI vs Input Power [23]

2.2.3 Distance Measurement

It is possible to measure the distance between two devices using RSSI values. However, this is not an easy process because some calculations have to be made. Electromagnetic waves have attenuation and it changes in every medium. This lost is called the path loss of electromagnetic waves. Different objects and different materials affect the path loss of a wave. This effect cannot be calculated beforehand but can be estimated using the Formula 2.1.

$$PL = 10 \times \alpha \times \log_{10}\left(\frac{d}{d_0}\right), \tag{2.1}$$

In this formula, d_0 is the reference distance which is generally 1 meter, d is the measured distance and the α is the path loss exponent [6].

α can be found like Equation 2.2. d_{min} is the minimum measured distance and d_{max} is the maximum measured distance. $P_r(d_{min})$ is the measured received signal at d_{min} and $P_r(d_{max})$ is the measured received signal at d_{max} [6].

$$\alpha = \frac{P_r(d_{min}) - P_r(d_{max})}{10 \times \left(log_{10}\frac{d_{max}}{d_0} - log_{10}\frac{d_{min}}{d_0}\right)} \tag{2.2}$$

Path loss is equal to the signal strength at desired point minus the signal strength at the reference point. In Equation 2.3, Pr (d) is the signal strength at distance d and Pr (d0) is the signal strength at the reference point [6].

$$P_r(d) = P_r(d_0) - 10 \times \alpha \times log_{10}\left(\frac{d}{d_0}\right) \tag{2.3}$$

After all of the arrangements distance d can be found as the Equation 2.4 [6].

$$d = 10^{\frac{-(P_r(d)-P_r(d_0))}{10 \times \alpha}} \times d_0 \tag{2.4}$$

2.3 Bluetooth Low Energy (BLE)

Bluetooth was created as a wireless alternative to data cables by exchanging data using radio transmissions in 1994. [25] Until 2010, 3 major versions are announced and every version had better capabilities. In 2010, Bluetooth version 4.0 was announced. Bluetooth low energy (BLE) (also called Bluetooth Smart or Version 4.0+ of the Bluetooth specification) is the power and application friendly version of Bluetooth that was built for the Internet of Things (IoT) [26].

Bluetooth uses 2.4 GHz band for radio communications [27]. It also has frequency hopping ability which enable many Bluetooth devices to work without problem. BLE works with the advertising principle. A slave device advertises its data and the master device can read this data with or without connecting to the device. A BLE slave can transmit 31 bytes of data [28]. The detail of this data can be seen in Figure 2.5. First 3 bytes of this data must be used for the flags. These flags show the type of the device and the advertisement. After the flags, 2 bytes are used as a header. After these bytes, 26 are left for the user and the user can transmit the desired data in this 26 bytes.

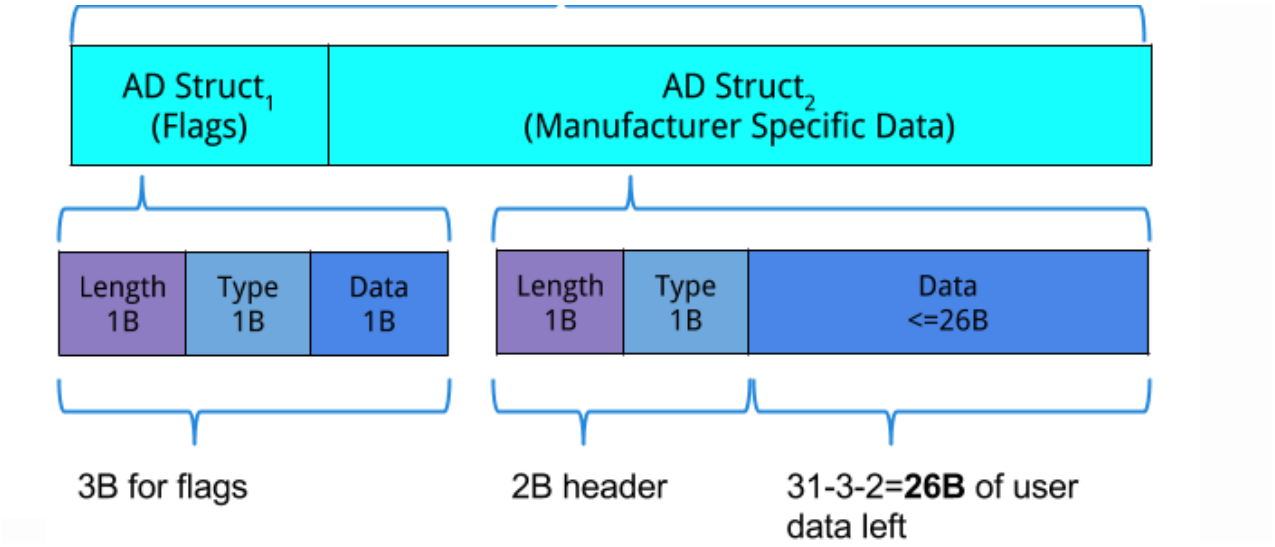

Figure 2.5: BLE Advertising Packet Structure [28]

2.3.1 Bluetooth Realization

In this project, Nordic Semiconductor's PCA10000 and PCA64105 modules are used which are part of nRF8001 development kit [29]. The modules are shown in the Figure 2.6. PCA10000 dongle is used for the communication with the computer and PCA64105 Arduino shield is used for the master device. PCA64105 has nRF8001 module on it. nRF8001 chip is only a Bluetooth radio and it does not have a microcontroller with it, therefore it must be used with a microcontroller and Arduino is chosen to be the microcontroller. On the other hand, PCA10000 has nRF51822 on it. nRF51822 has its own Acorn Risk Machine (ARM) [30] microcontroller so it does not require any other microcontrollers. Nordic Semiconductor provides its own Software Development Kit (SDK) [31] to use these modules with Arduino and a computer.

Figure 2.6: PCA64105 and PCA10000 Modules

2.4 Temperature Sensor

Understanding the case of an emergency is very essential in this project. Projects is designed to be activated when there is a fire. A temperature sensor is integrated to the system in order to achieve this. Maxim Integrated's DS18B20 [32] temperature sensor is chosen because of its operating voltage range high accuracy. Sensor is in the Figure 2.7.

DS18B20 is a digital temperature sensor [32]. It can measure the temperature from 9 to 12-bit resolution which corresponds the increments of 0.5°C, 0.25°C, 0.125°C, and 0.0625°C. It can work from -55°C to +125°C with 0.5°C accuracy. Supply voltage is between 3 to 5 V. The sensor communicates over 1-Wire protocol which uses only one wire. A pull up resistor is needed for the data line. Every produced DS18B20 has their own unique 64-bit serial code. This allows multiple sensors to work in the same data line.

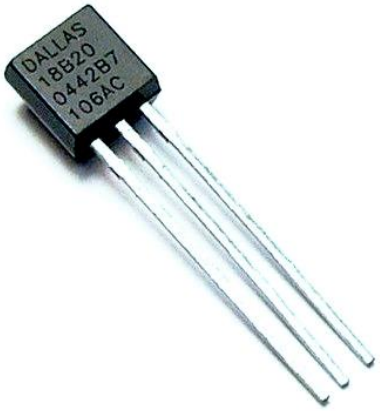

Figure 2.7: DS18B20 Temperature Sensor[33]

2.5 Windows Presentation Foundation (WPF)

Nordic Semiconductor's SDK has an example code called nRFUart. This is the basic code that communicates with the PCA10000 Bluetooth dongle. In the project this example is used as a basis for the Bluetooth communication because it has all the required classes and functions. nRFUart is written in WPF so the user interface of the project is also designed in WPF for the compatibility. The picture of the interface is in Figure 2.8. WPF is used for a rich presentation system for building Windows desktop applications [34]. It consists of two languages: XAML and C#.

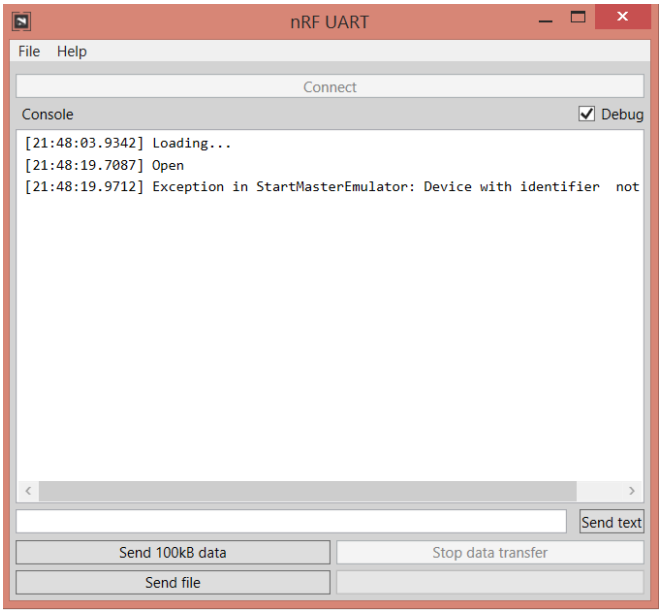

Figure 2.8: nRFUart Interface

XAML is Windows's language for creating graphical user interfaces. Rather than a windows forms application which the components are put by drag and drop method, all the components in the user interface are put together by using XAML code. Behind the user interface, all the processes are done using C#. C# is also a language created by Windows in order to create windows applications. In the project, Bluetooth communication and localization algorithms are written in C#. C# code communicates with the user interface which is written in XAML. Microsoft Visual Studio is used as the programming environment.

3. ALGORITHMS AND PROTOCOLS

3.1 Elliptic Curve Diffie-Hellman (ECDH)

Elliptic Curve Diffie-Hellman is a key sharing protocol over an insecure public channel [7]. It is a version of Diffie-Hellman key exchange protocol. In this method, parties agree on a shared secret key without exposing their private keys. ECDH uses elliptic curves rather than modulo operation. Addition operation in elliptic curves is a one-way function therefore, it is used in the key sharing.

Before the key exchange begins, parties should agree on the elliptic curve domain parameters which are (q, FR, a, b{, $SEED$}, G, n, h) [13]. In order to start the communication, parties A and B first generate random private keys d_A and d_B. After that they generate public keys $Q_A=d_AG$ and $Q_B=d_BG$. Than they send these public keys over the channel to each other. When parties receive each other's public key, they generate secret key as d_AQ_B and d_BQ_A. It is seen that $d_AQ_B = d_Ad_BG = d_Bd_AG = d_BQ$. Both of the parties have calculated the same value. The calculated value is a point on the curve and the x coordinate is used as the key. Diagram of this protocol can be seen in Figure 3.1

In this project ECDH is used in the communication between the master module and the computer. The protocol is realized over Bluetooth Low Energy.

3.2 Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) is an algorithm which is used in electronic data protection [8]. It consists of a symmetric block cipher which enables the encryption and the decryption of the data. It is also known as the Rjindael algorithm. AES is a strong and one of the most trusted algorithms to encrypt classified data.

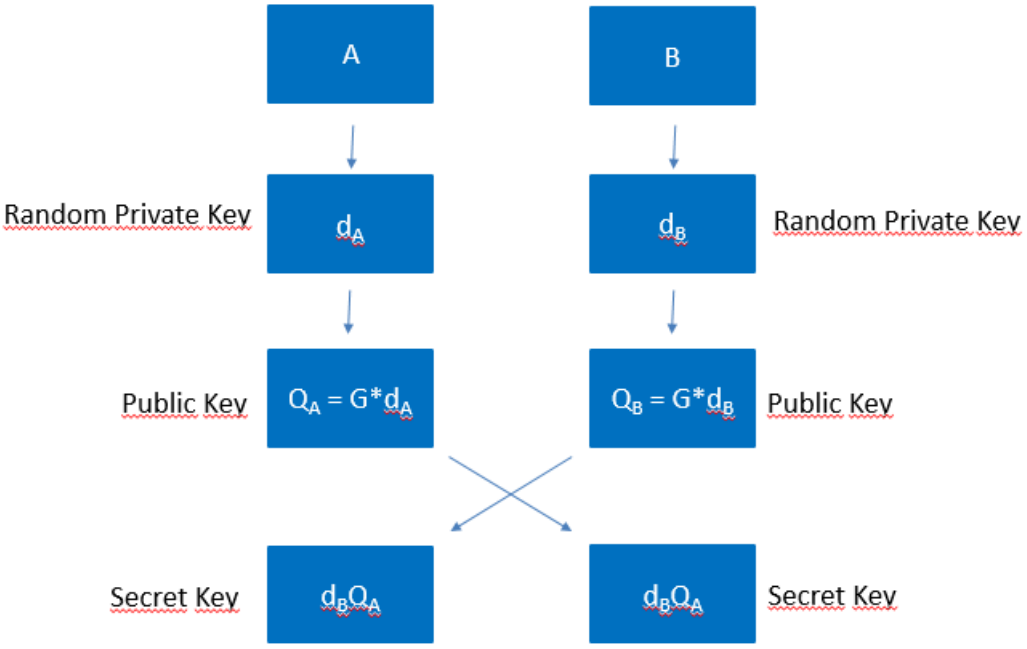

Figure 3.1: ECDH Key Sharing

In the project, AES is used to encrypt the information between the parties after the key exchange. The implementation of the AES algorithm is not a part of this project so it will not be explained in detailed. A library is used to implement the algorithm to Arduino and C#.

3.3 2D Trilateration Algorithm

Localization algorithms usually use geometry and mathematical formulas to calculate the desired location. Trilateration measures the distance to a several known points to calculate the location. In this project RFID modules are used for localization and these modules have RSSI values which can be converted into distances.

Trilateration depends on circle geometry. Minimum of 3 circles and distances to these circles must be known for trilateration. Figure 3.2 shows a circle in Cartesian coordinates. Equation 3.1 is the equation of this circle and trilateration uses this formula to find the location.

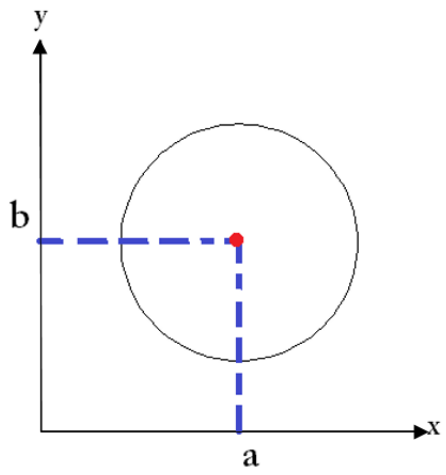

Figure 3.2: Cartesian Coordinates of a circle

$$(x-a)^2 + (y-b)^2 = r^2 \tag{3.1}$$

3 Circles can intersect at one point like in the Figure 3.3. This is the unique solution for these circles. In Figure 3.3, red point's points location can be calculated by measuring the distances r1, r2 and r3 and knowing the initial centers of the circles x1,x2,x3,y1,y2,y3. In the project red point is the reader module and the center of the circles represent the tags.

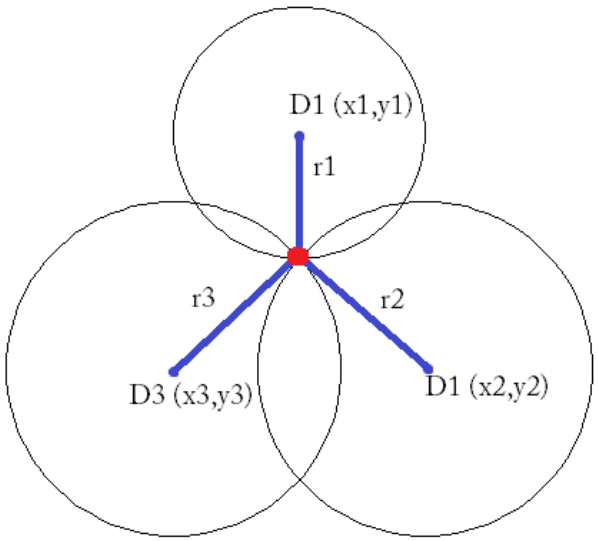

Figure 3.3: Intersection of 3 circles

Equations for the three of the circles can be written like equation 3.2. “x” and “y” is the coordinates of the reader module in this situation. 3 equations can be solved for x and y and the exact coordinates can be found.

$$(x - x1)^2 + (y - y1)^2 = r1^2$$
$$(x - x2)^2 + (y - y2)^2 = r2^2$$
$$(x - x3)^2 + (y - y3)_2 = r3^2$$

(3.2)

3.4 RF Communication Protocol

Frequency Shift Keying(FSK) is a frequency modulation type where the binary data is sent using different frequencies in digital communication [35]. 1 are 0 are marked with different frequencies. Gaussian Frequency Shift Keying(GFSK) is a type of FSK. GFSK filters the pulses with Gaussian filter to have a smoother transition. This have an advantage in reducing the interference with neighbouring channels.

In this project, GFSK is used with the RFM23 modules. Implementation of this modulation is not a part of this project so it is not going to be explained in detail. There is an Arduino library for driving the modules and the modules generates the signals.

3.5 One Wire Communication Protocol

1-Wire is a communication bus system which was created by Dallas Semiconductor Corp [32]. In this protocol parties communicate on one data line by pulling the line low or high. The line is pulled high using a resistor. Communication starts with the master by pulling down the line followed by the pulse from the slave. After the slave sends the presence pulse master understands that there is a device on the line and starts the communication.

In order to start the communication, master has to pull the bus low for a minimum of 480µs [32]. Slave device shows the presence by pulling the bus for 240µs. After these two steps, communication starts. Each device communicates in read or write time slots and only one signal is sent in a moment. Master device decides the read and write time slots. All of the read and write time slots must be 60µs duration. In order to write 1 to the bus, master the must pull the line and release it before 15µs. For a write 0 time slot,

master must continue pulling the line down for at least 60 seconds. All the communication is controlled by the master. Therefore, master tells the slave when it needs to send a data. All the read timeslots must be minimum of 60µs like the write time slots. Read time slot starts when the bus is pulled low for a duration of 1µs and released. After that DS18B20 transmits 0 or 1. It transmits a 1 by leaving the bus high and transmits 0 by pulling the bus low. This process lasts for 15µs, therefore master has to read the line after 15µs.

4. HARDWARE IMPLEMENTATION

4.1 BLE Implementation

4.1.1 Arduino and nRF8001 Shield Setup

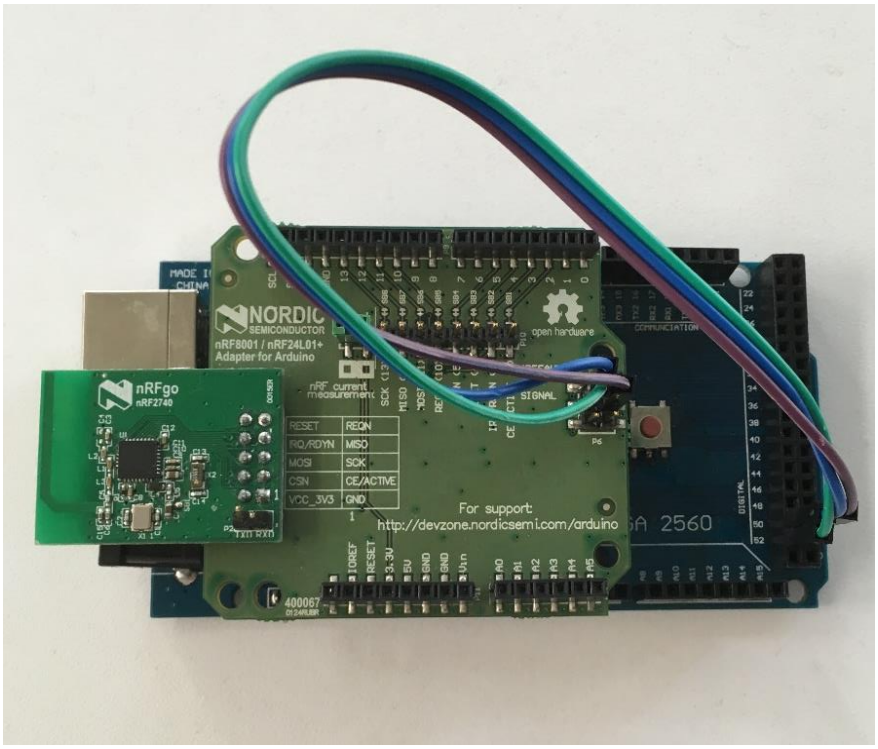

Figure 4.1: Arduino Mega and nRF8001 Shield

Arduino Mega and nRF8001 shield are used like in the Figure 4.1. nRF8001 shield communicates over SPI and it connects to Arduino through 6 In Circuit Serial Programming (ICSP) pins. Normally there are headers below these pins which makes connection to the Arduino's ICSP pins. These header pins were broken while testing the shield so the connection was made by 3 jumper cables. For the SPI communication, Master Out Slave In (MOSI), Master In Slave Out (MISO) and Serial Clock (SCK) pins are needed. Arduino pins related to SPI is seen in the Figure 4.2. Figure 4.3 shows the ICSP pins of the nRF8001 shield. As a result, Arduino pin 51 is connected to ICSP-4, pin 50 is connected to ICSP-1 and pin 52 is connected to ICSP-3.

Arduino / Genuino Board	MOSI	MISO	SCK	SS (slave)	SS (master)	Level
Uno or Duemilanove	11 or ICSP-4	12 or ICSP-1	13 or ICSP-3	10	-	5V
Mega1280 or Mega2560	51 or ICSP-4	50 or ICSP-1	52 or ICSP-3	53	-	5V

Figure 4.2: Arduino SPI Pins [36]

Figure 4.3: Arduino ICSP Pins [36]

On the Arduino side, ble_uart_reference which is written by Bahadır Gün is used for the Bluetooth communication [13]. In this code, communication is encrypted using ECDH and AES. In order the test the code, some changes have made. For the SPI communication, slave select (SS) pin is needed. nRF8001 is designed for Arduino Uno. It iş seen from the Figure 4.4 that Arduino Uno has SS pin of 10 and Arduino Mega has a SS pin of 53. This change can be done in the code. 10 must be written instead of SS.

After the key sharing, Arduino writes the shared key to its Electrically Erasable Programmable Read-Only Memory (EEPROM) in order to remember it for the next time. When the Arduino is powered on, it checks the EEPROM address 0x00. If it is 0, Arduino starts the ECDH, if it is not 0 it uses the saved key. In order to observe the key sharing in every power on, 0 is written to the EEPROM address 0 by the code: EEPROM.write(0, 0x00). After all of these changes, the code is compiled and uploaded successfully to Arduino.

A variable called connectedToComputer is added to code to send the desired data. The function aci_loop is used for handling the Bluetooth communication. When the state is equal to ACI_EVT_TIMING, it means that key sharing is successful and the device

is ready to send data. connectedToComputer is set to 1 in this state. In the void loop() code, an if statement is added for sending the desired data and checks the value of connectedToComputer.

4.1.2 nRFUart_reference Setup

ble_uart_reference communicates with the nRFUart_reference which is also written by Bahadır Gün[13]. In order to test the functionality of ble_uart_reference, it had to be used by the nRFUart_reference example. nRFUart_reference is code written in C#. ECDH and AES also implemented in the code for secure communication. Visual Studio is used to open and run the project. At the beginning some errors occurred and the code did not compile.

The code must be opened under administrator credentials. Therefore it is copy and pasted to folder C:\Program Files (x86)\Nordic Semiconductor\Master Emulator\2.1.13.14 \Example code which is under C folder. When the code is opened, LibraryBG comes as a startup project. This prevents code to be compiled. In order to solve this, nRFUart is set as a startup project. The code uses dll's and they were compiled in release mode so the code must be always compiled in release mode instead of debug mode. After these changes, the compiled successfully.

After the compilation of nRFUart_reference and ble_uart_reference, the secure communication wanted to be observed. At the first try nRFUart_refernece could not find nRF8001. In order to solve the problem, ServicesCompleteListUuid128 variable which is in the function isEligibleForConnection, is changed to ServicesMoreAvailableUuid128. After the change, both of the codes worked perfectly. Secure Communication with AES algorithm can be seen in Figure 4.4 and Figure 4.5. The data is encrypted before it was sent in Figure 4.5 and decrypted when it is received in Figure 4.4.

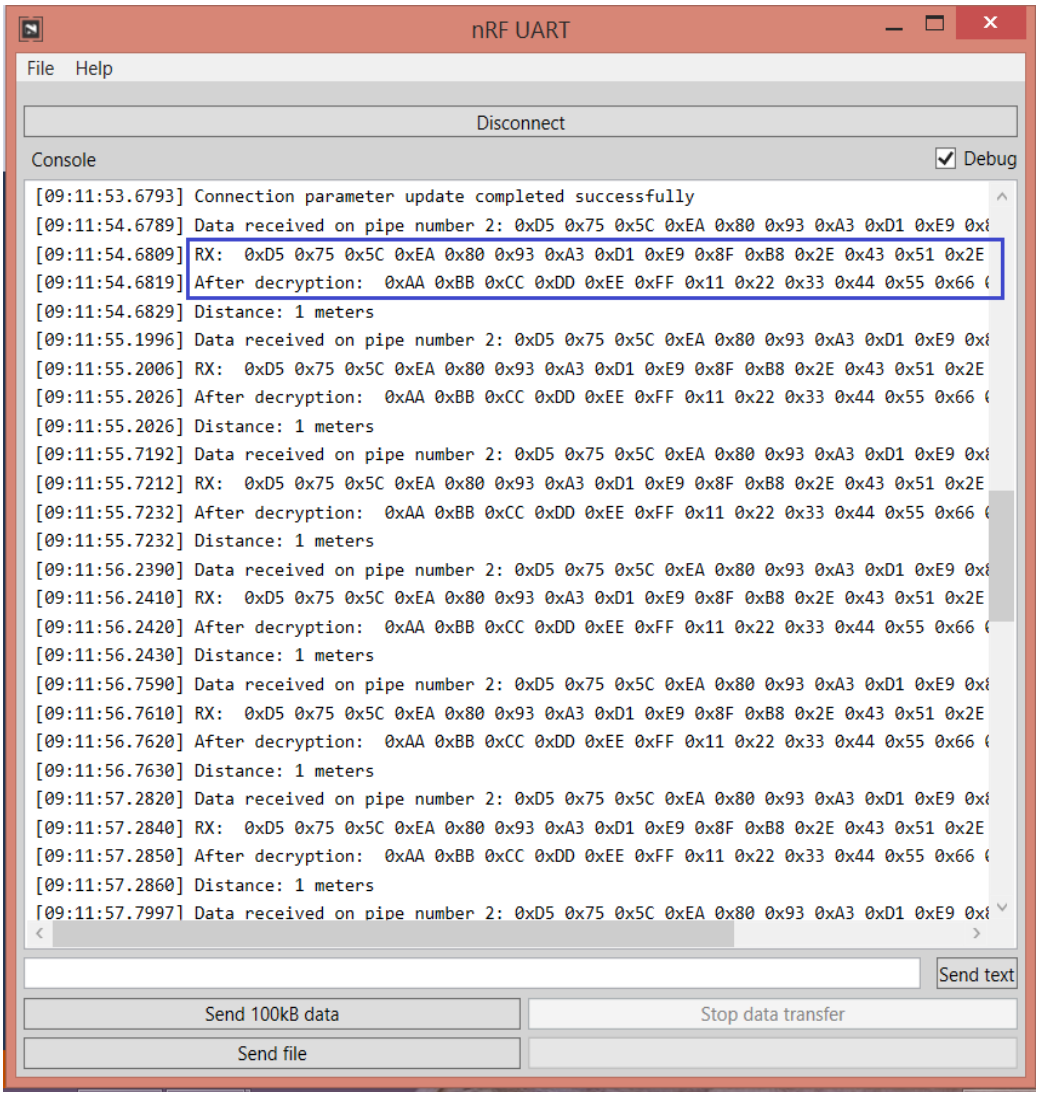

Figure 4.4: nRFUart Reference Output Window

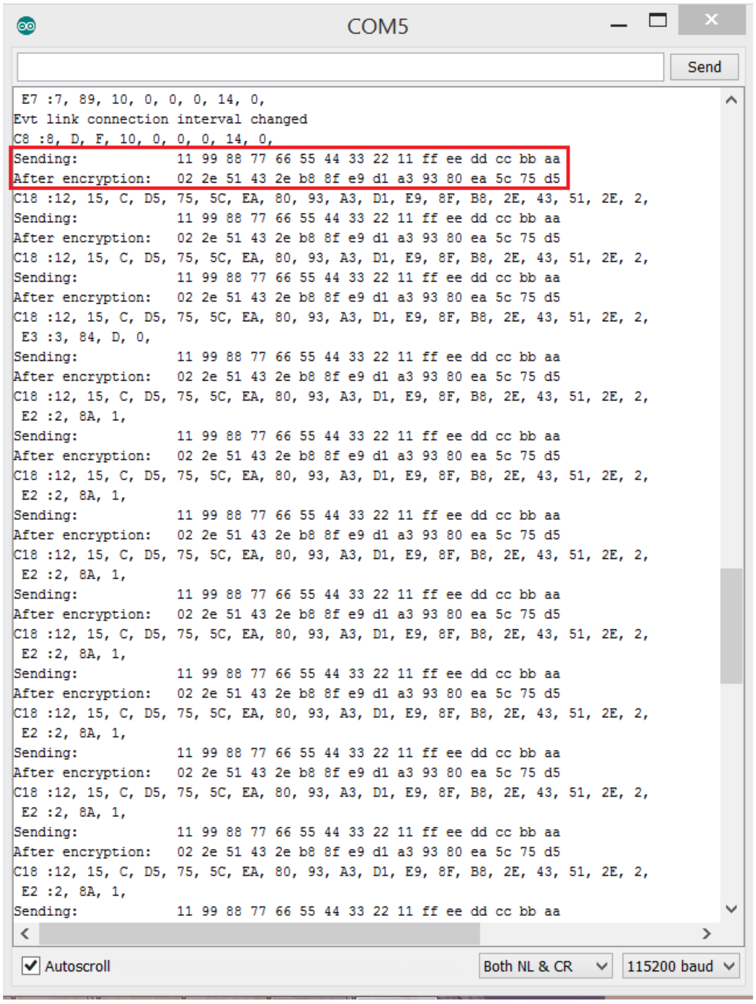

Figure 4.5: ble_uart_reference Output Window

4.2 RFM23 Implementation

RFM23 modules are connected to Arduino Mega and Arduino Nano's. Figure 4.6 shows the pin connections to Arduino Mega and Figure 4.7 shows the pin connections to Arduino Nano.

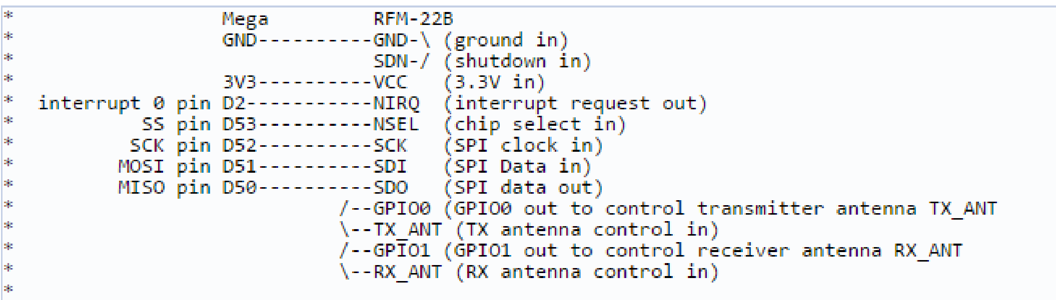

Figure 4.6: Arduino Mega and RFM23 Connections[43]

In the project there are 5 tags and 1 reader. First the codes for the tags are uploaded. RFM23 modules are initialized by rf22.init(1,0) function. They are initialized by a header ID which will be used to identify the tags. 5 tags are initialized by header ID's 1,2,3,4 and 5. RF modules communicates at 434 MHz so the frequency is set by the function rf22.setFrequency(434.0). Modules communicate using GFSK protocol.

```
*                 Arduino      RFM-22B
*                 GND----------GND-\ (ground in)
*                              SDN-/ (shutdown in)
*                 3V3----------VCC   (3.3V in)
*  interrupt 0 pin D2-----------NIRQ  (interrupt request out)
*           SS pin D10----------NSEL  (chip select in)
*          SCK pin D13----------SCK   (SPI clock in)
*         MOSI pin D11----------SDI   (SPI Data in)
*         MISO pin D12----------SDO   (SPI data out)
*                            /--GPIO0 (GPIO0 out to control transmitter antenna TX_ANT
*                            \--TX_ANT (TX antenna control in)
*                            /--GPIO1 (GPIO1 out to control receiver antenna RX_ANT
*                            \--RX_ANT (RX antenna control in)
*
```

Figure 4.7: Arduino Nano and RFM23 Connections[43]

This is arranged by the function f22.setModemConfig (RF22::GFSK_Rb2Fd5). Finally, message that they are going to be send is decided. Every tag sends their header ID as a message. All the tags transmit the data in 100ms intervals.

After the tags, reader module is initialized in the same way with the header ID of 6. The aim of the reader is to read the tags and decide the closest 3 of them. When a message is received, its headerID, RSSI value and the message can be understood.

For the ease of the process, a C++ class called Tag is written. Every class object has, tagID, tagRSSI and tagMessage variables. A code called RFIDListenAndOrder is written to test the functions. At the beginning of the code, reader initializes an array of Tag objects. Reader listens the tags for 2 seconds and if it receives a message with different header ID, it adds this as a new Tag object. After the two seconds, reader has an array of Tag objects. Reader compares every TAG's RSSI value to each other and puts the smallest ones at the beginning of the array. After the process. Array's elements of index numbers 0,1 and 2 are the closest tags to the reader. The output of the RFIDListenAndOrder is in Figure 4.5. 4 tags with ID's 1,2,3 and 4 are used. The tag module system can be seen in Figure 4.9.

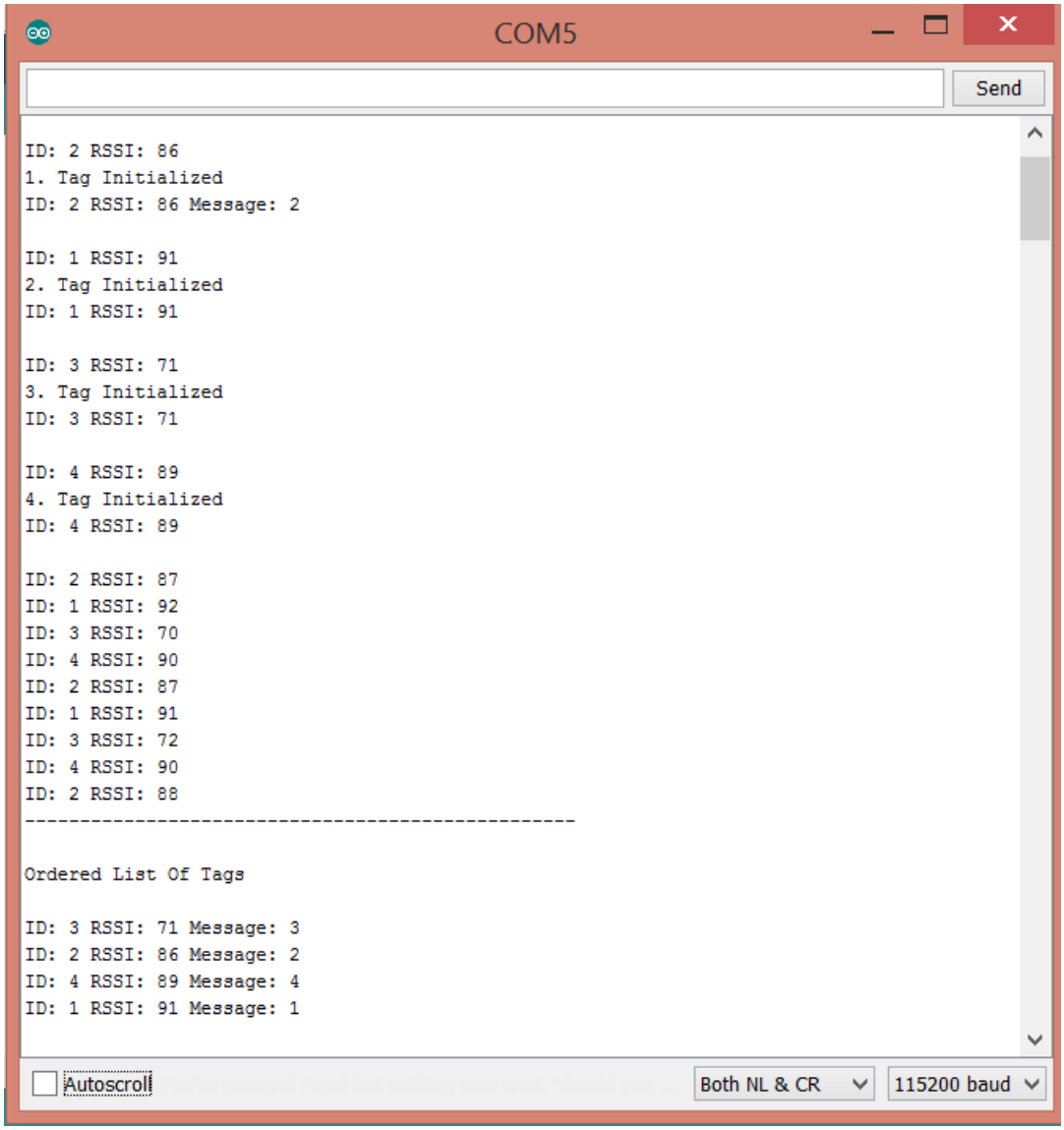

Figure 4.8: Output of RFIDListenAndOrder Code

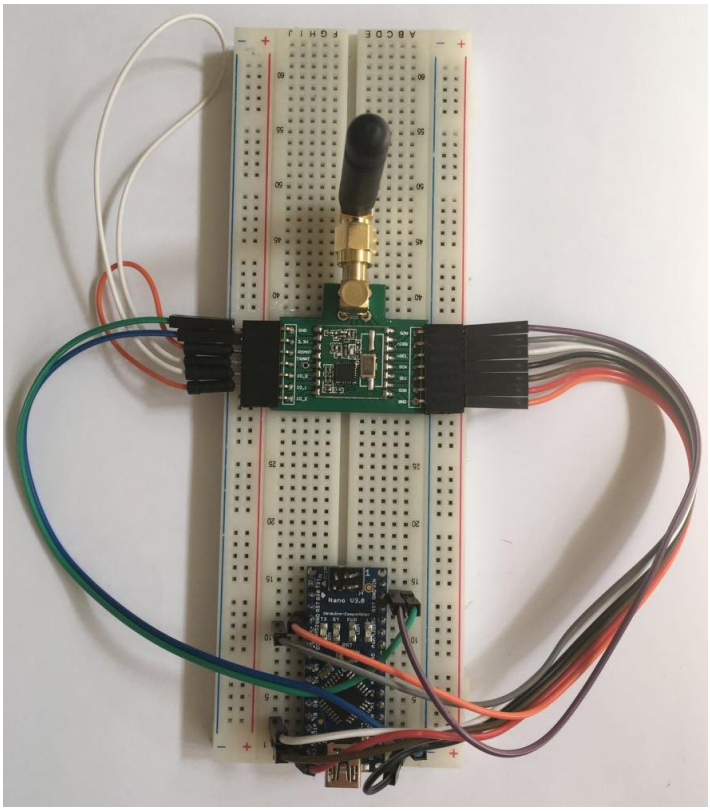

Figure 4.9: Tag Module with Arduino Nano

4.3 Temperature Sensor Implementation

Arduino has a library called OneWire.h. This library has all the necessary functions to read DS18B20 temperature sensor. A temperature threshold is chosen for the system to work. This sensor is going to decide if there is fire or not in the room. Threshold is chosen as 60 °C. If the temperature is above 60 °C, Reader modules starts to send data to the computer.

5. SOFTWARE IMPLEMENTATION

5.1 BLE Communication Protocol

AES function works with 128-bit data blocks only so all the sent data must be 128-bit in length. Sent data is chosen to be an array of 16 elements which is also equal to sending 16 bytes. First byte can be set as a header and the last byte can be set as a footer. This enables both parties to agree on the sent data. The distance is a double so it can have a decimal and an integer part. The data is sent in bytes so it is restricted with integers from 0-255. Therefore, the distance has to be separated into two. Integer part can be sent in one byte and the decimal part can be sent in another byte. Only 1 digit can be sent from the decimal part from these reasons.

In this project, only information of 3 tags will be send. The header is chosen to be 255. Next 3 elements are the tag ID's of the closest 3 tags. 4^{th} element is the integer part of the distance 1, 5^{th} element is the decimal part of the distance 1, 6^{th} element is the integer part of the distance 2, 7^{th} element is the decimal part of the distance 2, 8^{th} element is the integer part of the distance 3, 9^{th} element is the decimal part of the distance 3. The sent data looks like "255,1,2,3,1,5,2,0,3,3". This data means that tag 1 is 1.5 meters, tag 2 is 2.0 meters and tag 3 is 3.3 meters away from the reader.

5.2 Receiving Data from BLE Dongle

Event driven programming is one of the common techniques used in user interface programming. When an action occurs, this triggers another method. There are 2 members in event driven programming: a publisher and a subscriber. Publishers raises an event and the subscribers listen to them. Many subscribers can subscribe to one publisher and the publisher is unaware of them. It just raises an event. nRFUart_reference example is written in C# and event handlers are used a lot. The user interface of the project is added on to the nRFUart_reference so event handlers are used in the project.

nRFUartController.cs is the class which is responsible for the Bluetooth communication. This class has all the methods to communicate with the BLE dongle and sends feedback to the main loop. There is method called OnDataReceived() in this class. This is where the incoming data is handled. When a data is received, it is copied

to and array called inputHex. After that the data is processed, if the first byte is 0xFF in hexadecimal, this means the distance data is received and must be processed. Event handers are used to notify the responsible methods. A custom event is created called OutputCoordinateEventArgs. This is event is capable of sending a message. In the method OnDataReceived(), if the first byte is 0xFF, this event is raised to send the tag distance to the desired methods.

MainWindow.xaml.cs is the main class. This class controls the communication between the user interface and backend. In the InitializeNrfUartController() method, OnOutputCoordinate method is the event handler and it is subscribed to the the OutputCoordinateEventArgs event. In the method OnOutputCoordinate, the coming data is parsed and the coordinate of the reader is calculated.

5.3 Initialization of Tags

In the project, it is wanted to cover an area with enough tags so the localization can be accurate. Therefore, the map of the area and coordinates of the tags must be known before the system is established. Each tag has a tag ID, x coordinate and a y coordinate. The best way to identify each tag is to have a Tag class. Therefore, a Tag class is written in C# called Tag.cs and added to the project. The localization algorithm works according to the coordinates of the tags so all the tags must be initialized at the beginning of the code. 5 tags are used in this project. These tags are initialized according to the table 5.1. These are not real values, they are just used for the test purposes.

Table 5.1: Initial Coordinates of the Tags

Tag ID	X Coordinate (m)	Y Coordinate (m)
1	10	10
2	10	15
3	10	20
4	20	10
5	20	15

5.4 Implementation of Localization Algorithm

Trilateration is used in the localization algorithm as stated is section 2. The code that calculates the exact location is taken from Onur Azbar's graduation project [14]. The

code is written in C++ but implemented to C#. First of all, the code is written as a function. This function is called calcCoordinates in the project. calcCoordinates function takes 6 parameters. First three of them are the tag ID's and the last three of them is the distances of these tags to the reader module. The function compares the tag ID's with the previously initialized tags which are initialized according to the section 5.2. After the comparison, function calculates the exact location using the coordinates of the three tags and the distances to these tags.

In order to test the calcCoordinates function, 3 sets of data are sent from Arduino to BLE dongle. Firstly, 3 target coordinates are chosen. These are (15,15), (18,18), (12.5,12.5). Than the distance from this coordinates to the 3 closest tags have been calculated manually. Finally, the transferred data is initialized and sent. These 3 coordinates and tag ID's are shown in the Table 5.2. When the data is transferred to the computer, it is expected that the calcCoordinates function to calculate the estimated coordinates of the readers. Coordinate data is sent according to the protocol in section 5.1. A new code called nRFUart_calcCoordinates is written with the implementation in sections 5.1, 5.2 and 5.3. ble_uart_reference is also modified to send the 3 coordinate data. The new code is called ble_uart_send3Location. Output window of nRFUart_calcCoordinates is in Figure 5.1 and the output window of ble_uart_reference is in Figure 5.2.

Table 5.2: Test Data for the Localization Algorithm

Desired Coordinates (x,y)	Tag ID1	Tag ID2	Tag ID3	Distance 1 (m)	Distance 2 (m)	Distance 3 (m)
15,15	2	5	1	5	5	7
18,18	3	2	5	8.2	8.5	3.6
12.5,12.5	1	2	4	3.5	3.5	7.9

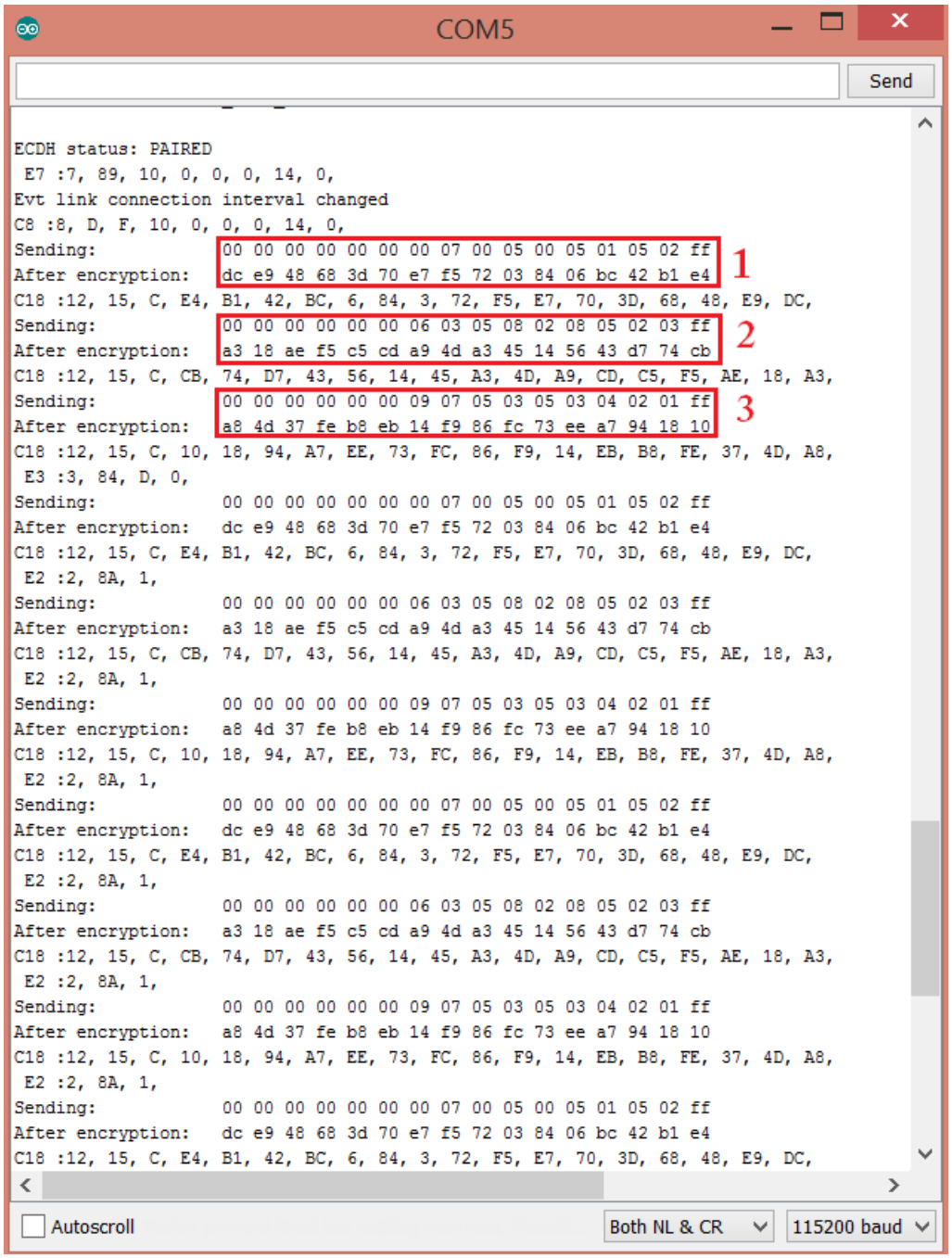

Figure 5.1: Output of the ble_uart_send3Location

First red rectangle in Figure 5.1 shows the first sent data. It is “FF,2,5,1,5,05,0,7,0”. When nRFUart_calcCoordinates takes this string, it calculates the coordinates. The can be seen in Figure 5.2, in the first blue rectangle. The desired coordinates were 15,15 and the estimated coordinates are 15,14.9.

Second red rectangle in Figure 5.1 shows the second sent data which is “FF,3,2,5,8,2,8,5,3,6”. The estimated coordinates can be seen in the second blue rectangle in the Figure 5.2. The desired coordinates were 18,18 and the estimated coordinates are 17.9645,18.001 and the de

Third red rectangle in Figure 5.1 shows the third sent data which is “FF,1,2,4,3,5,3,5,7,9”. The estimated coordinates can be seen in the third blue rectangle in the Figure 5.2. The desired coordinates were 12.5,12.5 and the estimated coordinates are 12.492,12,5.

Three of the estimated coordinates are very close to the desired coordinates. Location estimation was successful.

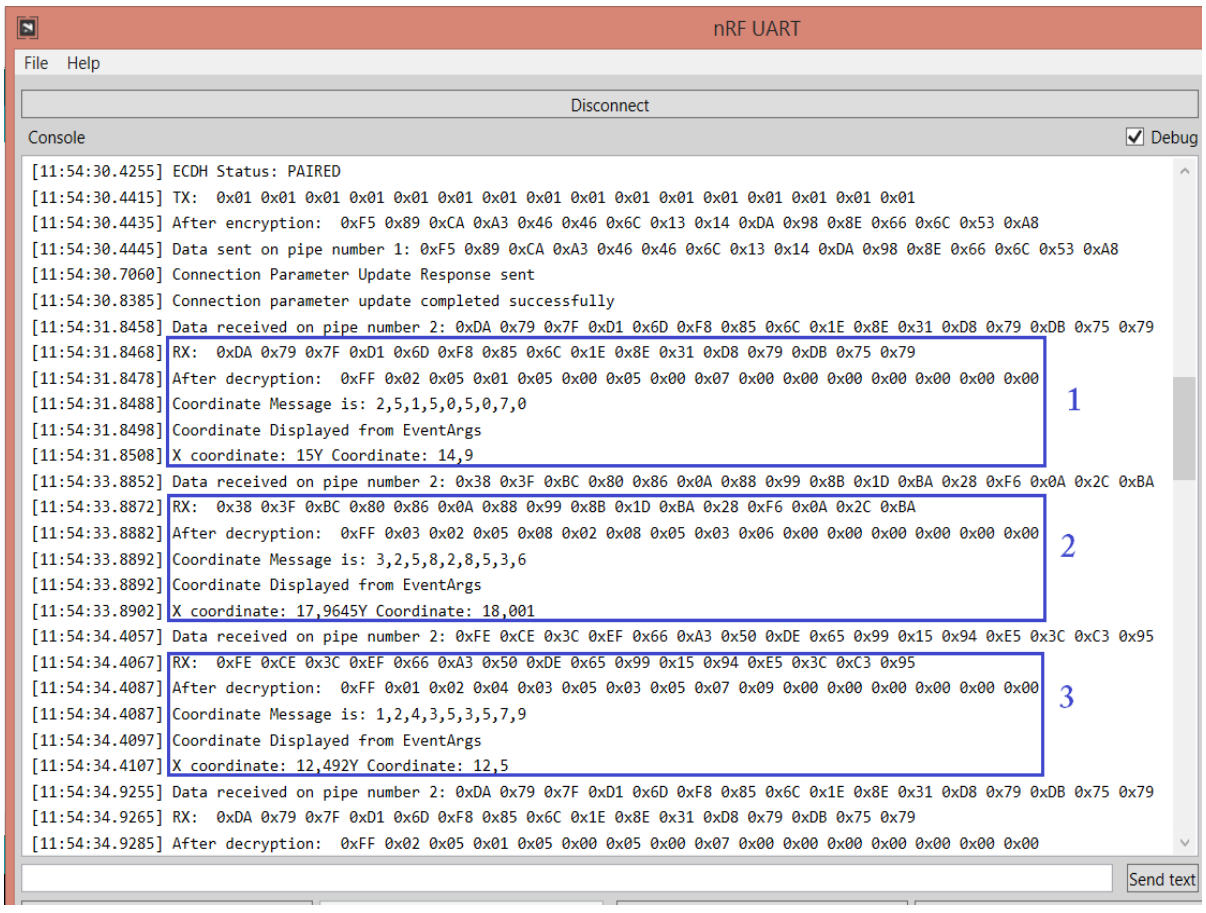

Figure 5.2: Output Window of nRFUart_calcCoordinates Example

5.5 Design of the User Interface

5.5.1 XAML

This project will be used in case of an emergency so a good user interface is very important in this project. The user must understand everything clearly. The user interface is designed base on nRF_Uart_Reference project because this project has ECDH and AES implemented already. User Interface side is written in XAML in the previous project so it is used in this project.

User interface consist of a big DockPanel and everything is put in this DockPanel. Map interface is added without changing the previous interface. It is added on the right of the previous system. Old interface is kept as it was because it is good for debugging. There is a text box and everything about the communication and the localization is written on this text box. Everything that is added is added to right by the function DockPanel.Dock=”Right”. There are 2 different objects to be displayed on the map. One is the tags and they are going to be blue circles. The other one is the reader module and it is going to be red.

The map of the area is imported as an image to the interface. The size of the image is chosen by pixel numbers. Image area is chosen to be 350 pixels in height and 500 pixels in width. This size can be changed according the map of the area. Each 10 pixel corresponds to 1 meter in the interface. This is done to make it easier to visualize the content.

3 buttons are added to the bottom right of the interface. These buttons are “Show Tags”, “Clear Map” and “Calc Coordinate”. When “Show Tags” is pressed, Tags are visualized as blue boxes on their coordinates. When “Clear Map” is pressed, red circle is erased from the Map. When “Calc Coordinate” is pressed, the reader is shown on the map. These 3 buttons are related to 3 functions in the MainWindow.xaml.cs. When a button is pressed, it raises an event and this event is handled in MainWindow.xaml.cs.

5.5.2 Canvas

Canvas is a panel where the drawing can be done in WPF. Different geometric shapes can be drawn in the user interface with on panel. Background of a canvas can be transparent. This feature is used in the project. A canvas is created in 350 pixels

height and 500 pixels width which is same as the imported map picture. After this creation, a change in the canvas will look like a change in the map picture. In order to use the drawing functions, two classes must be imported to the project. System.Windows.Shapes and System.Windows.Media is added to the MainWindow.xaml.cs file. All the tags and the reader will be shown as ellipse objects. The height and the width of the ellipse object is set as 10 pixels in the creation of the objects. This interface is added on to nRFUart_calcCoordinates and the same data in Table 5.2 is sent to the computer.

The final output window looks like Figure 5.3. Blue dots are the tags which are initialized according to the Table 5.1 and red dots are the estimated coordinates. Normally the system is real time and only 1 red dot is seen at a time. However, 3 is shown here just for demonstration purposes.

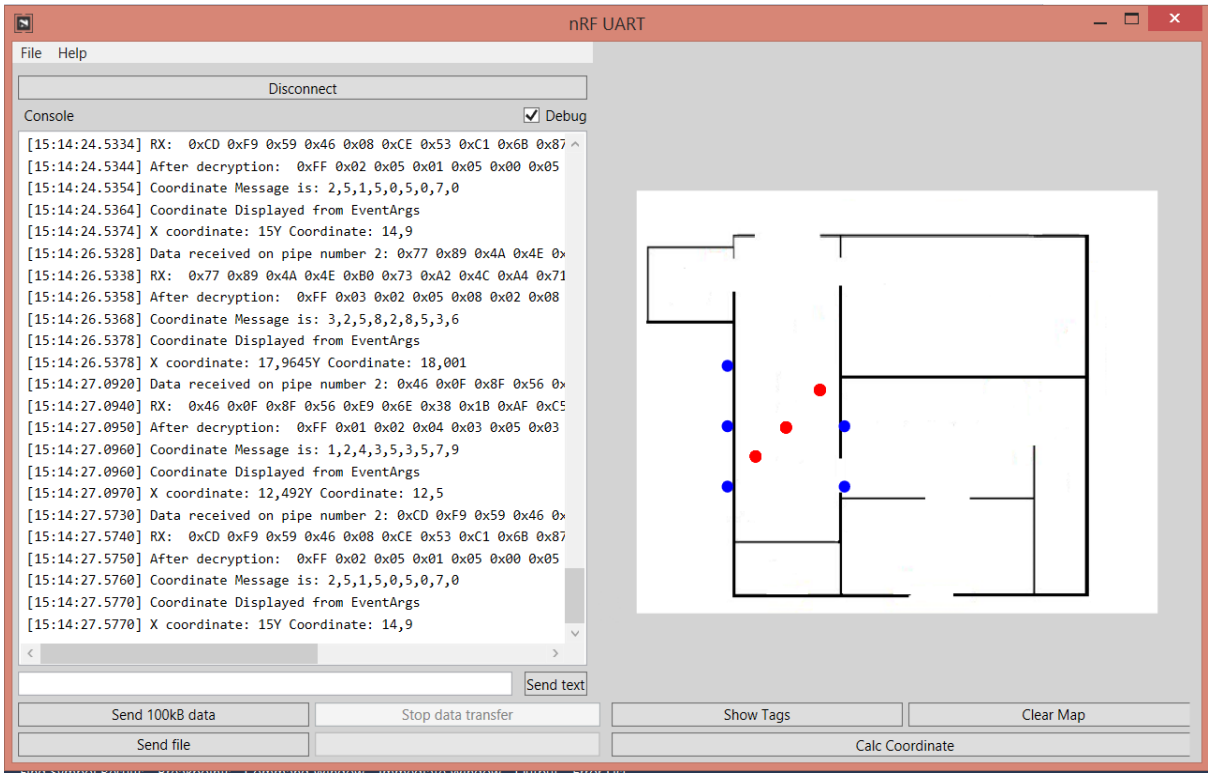

Figure 5.3: Final Output of the User Interface

6. CONCLUSION

In order to totally evacuate a building in case of an emergency, all the locations of the people must be known. An indoor localization system is designed and implemented in this project to solve this problem. This project combined the sub-blocks which are previous year's graduation projects and added a graphical user interface to it.

The final design aims to cover a building with RFID tags and the people have a reader RFID module. This reader module reads the messages which come from the tags and calculates its distance to these tags. After that, it sends all the data to a central computer through BLE. This communication is encrypted in order to main the security of the system. When the computer receives the data, it calculates the location of the reader using trilateration and displays is on a map of the area.

At first secure BLE communication is realized. This part of the system works very accurate. No errors were detected during any of the encryption processes. At the beginning of a communication, two of devices agree on a key using ECDH and then encrypt the real data using AES. When one of the devices resets, ECDH needs to be start from the beginning. Two of the protocols are fast, reliable and easy to implement to many devices.

Secondly, localization algorithm was tested. This algorithm is implemented on the user interface using C#. Three sets of data are sent to the system and the algorithm calculated the estimated locations which are very close to the real coordinates. Because of the communication protocol, only one digit after the decimal point of the coordinates can be sent to the central computer. Therefore, the resolution is 10 cm in the system. However, this is not a problem for locating a person in a building.

Thirdly, RFID tags are used for localization. A code is written for the reader module. This code listens to the surrounding RFID tags and decides the closest 3 of them by comparing their RSSI values. This 3 coordinates are used in the localization algorithm. RSSI values are very unstable and set of calculations must be done before the system is established in a building.

Finally, a graphical user interface is designed in WPF. This user interface has a map of the area. The locations of the tags and the reader module can be seen on the map. The location of the reader module is updated in real time.

The system is composed of many elements like RFID and Bluetooth modules, WPF, localization algorithm, secure communication and a temperature sensor. In this project, these elements are combined to work together in a one complete system. Even though every element works as expected, the system is not tested in a real environment in real time.

REFERENCES

[1] **Sanson, H., Mitsuji, M.,** (2005), Localization for Emergency Sensor Networks. *The 7th International Conference on Advanced Communication Technology,* pp. 982-987

[2] **Global Positioning System (GPS),** Retrieved May 20, 2016, from www.gps.gov

[3] **Global Navigation Satellite System (GLONASS),** Retrieved May 20, 2016, from www.glonass-iac.ru/en

[4] **Roberts, C.M.,** (2006), Radio Frequency Identification (RFID), *Electronic Article Surveillance; Uniform Code Council; EAN,* pp. 19-26

[5] **Bluetooth SIG**, Retrieved May 20, 2016, from https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth

[6] **Pu, C.C., Lim, S.Y. and Ooi, P.C.** (2012), Measurement Arrangement for the Estimation of Path Loss Exponent in Wireless Sensor Network. *7th International Conference on Computing and Convergence Technology (ICCCT),* pp. 807 – 812.

[7] **Koblitz N., Menezes A.,Vanstone S.,** (2000), The State of Elliptic Curve Cryptography, *Towards a Quarter-Century of Public Key Cryptography*, pp 103-123

[8] **Federal Information Processing Standards Publication 197,** (2001), Announcing the ADVANCED ENCRYPTION STANDARD (AES)

[9] **Bluetooth SIG**, Retrieved May 20, 2016, from https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth-technology-basics/low-energy

[10] **Oguejiofor O.S, Aniedu A.N, Ejiofor H.C, Okolibe A.U**., (2013) Trilateration Based Localization Algorithm for Wireless Sensor Network, *International Journal of Science and Modern Engineering (IJISME),* pp.21-27

[11**] What is XAML?,** Retrieved May 20, 2016, from https://msdn.microsoft.com/en-us/library/cc295302.aspx

[12] **Aktaş, R.,** (2015), Distance Estimation Using Received Signal Strength for RFID Tracking System, *BSc Thesis*, I.T.U. Faculty of Electrics and Electronics, Istanbul.

[13] **Bahadır, G.,** (2015), Design and Implementation of A Secure Bluetooth Low Energy Communication, *BSc Thesis*, I.T.U. Faculty of Electrics and Electronics, Istanbul.

[14] **Azbar, O.,** (2015), Hardware/Software Codesign of an RFID Based Indoor Localization System, *BSc Thesis*, İ.T.Ü. Electrical and Electronics Faculty, İstanbul.

[15] **Oran, A.,** (2015), Implementation of A RFID Based Indoor Localization System on FPGA, *BSc Thesis*, I.T.U. Faculty of Electrics and Electronics, Istanbul.

[16] **Çık O.,** (2015), Bir RFID Takip Etme Sistemi için Okuyucuları Sürecek Bir Alt Sistemin FPGA Üzerinde Gerçeklenmesi, *BSc Thesis*, I.T.U. Faculty of Electrics and Electronics, Istanbul.

[17] **Baas,** Building as a Service, Retrieved May 5, 2016, from http://baas-itea2.eu/cms/project-menu-item.

[18] **arduino.cc**, What is Arduino, Retrieved May 20, 2016, from https://www.arduino.cc/en/Guide/Introduction

[19] **USB Implementers Forum, Inc**. Instant, No Hassle Connections, Retrieved May 20, 2016, from http://www.usb.org/home

[20] **arduino.cc**, Arduino Nano, Retrieved May 20, 2016, from https://www.arduino.cc/en/Main/ArduinoBoardNano

[21] **arduino.cc**, Arduino Mega, Retrieved May 20, 2016, from https://www.arduino.cc/en/Main/arduinoBoardMega2560

[22] **arduino.org**, Arduino Mega *2560*, Retrieved May 20, 2016, from http://www.arduino.org/products/boards/arduino-mega-2560

[23] **HopeRF Electronics,** RFM22B/23B ISM Transciever Module, V1.1.

[24] **Texas Instruments**, (2012), KeyStone Architecture Literature Number: Serial Peripheral Interface (SPI) User Guide

[25] **Bluetooth SIG**, (2016), our History, Retrieved May 20, 2016, from https://www.bluetooth.com/media/our-history

[26] **theguardian.com**, (2015), What is The Internet of Things, Retrieved May 20, 2016, from https://www.theguardian.com/technology/2015/may/06/what-is-the-internet-of-things-google

[27] **Bluetooth SIG** (2010). Bluetooth Specification Version 4.0

[28] **mbed.com**, BLE Modes and Profiles, Retrieved May 20, 2016, from https://docs.mbed.com//docs/ble-intros/en/latest/Introduction/BLEInDepth/

[29] **Nordic Semiconductor** (2014). nRF8001 Development Kit User Guide v2.0

[30] **ARM**, Retrieved May 20, 2016, from https://www.arm.com/

[31] **Nordic Semiconductor**, nRF51 SDK, Retrieved May 20, 2016, from https://www.nordicsemi.com/eng/Products/Bluetooth-Smart-Bluetooth-low-energy/nRF5-SDK

[32] **Maxim Integrated**, (2015), DS18B20 datasheet

[33] **Tushev S.,** (2013), How it works: DS18B20 and Arduino, Retrieved May 20, 2016, from https://tushev.org/images/electronics/arduino/ds18x20/DS18B20.jpg

[34] **Microsoft**, Windows Presentation Foundation, Retrieved May 20, 2016, from https://msdn.microsoft.com/tr-tr/library/ms754130(v=vs.110).aspx

[35] **Northern Illınois University**, Binary Frequency Shift Keying, Retrieved May 20, 2016, from http://www.niu.edu/remotelab/samplegui/binshiftkey.shtml

[36] **arduino.cc,** SPI Library, Retrieved May 20, 2016, from https://www.arduino.cc/en/Reference/SPI

[37] **airspayce.com,** RF22 library for Arduino, Retrieved May 20, 2016, from http://www.airspayce.com/mikem/arduino/RF22/

RESUME

Name: Gediz Morgil

Date and Place of Birth: 9 May 1993, Istanbul

Address: Emirhan Cad. Barış Sok. Fulya Sit. E Blok No:21 Beşiktaş/İstanbul

University: Istanbul Technical University