MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

INDOOR POSITIONING SYSTEM

DIPLOMA PROJECT

Author:**Raul Velcherean**

Scientifi coordinator: **asis.drd.ing. Gabriel Harja**

2019

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Vizat,

DECAN
Prof.dr.ing. Liviu MICLEA

DIRECTOR DEPARTAMENT AUTOMATICĂ
Prof.dr.ing. Honoriu VĂLEAN

Autor: **Raul Velcherean**

Indoor positioning system

1. **Enunţul temei:** *Provide the location of a TAG module in relation to the other ANCHOR modules, similar to the GPS system but on a smaller scale.*

2. **Conţinutul proiectului:** *(enumerarea părților componente) Pagina de prezentare, Declarație privind autenticitatea proiectului, Sinteza proiectului, Cuprins, Titlul capitolului 1, Titlul capitolului 2,… Titlul capitolului n, Bibliografie, Anexe.*

3. **Locul documentaţiei:** *Universitatea Tehnică din Cluj-Napoca, alte locuri dacă este cazul*

4. **Consultanţi:** *asis.drd.ing. Gabriel Harja*

5. **Data emiterii temei:**

6. **Data predării:**

Semnătura autorului ____________________________

Semnătura conducătorului științific ____________________________

2019

Declaraţie pe proprie răspundere privind
autenticitatea proiectului de diplomă

Subsemnatul(a) ______**Prenume NUME**__________________________________,
legitimat(ă) cu ______CI/BI__ seria______nr.______________, CNP__________________________,
autorul lucrării:

____

____

____

elaborată în vederea susţinerii examenului de finalizare a studiilor de licență la **Facultatea de Automatică şi Calculatoare**, specializarea **,** din cadrul Universităţii Tehnice din Cluj-Napoca, sesiunea a anului universitar 2017-2018, declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării, şi în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licenţă.

In cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, *anularea examenului de licenţă*.

Data	Prenume NUME
________	______________________
	(semnătura)

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

SINTEZA

proiectului de diplomă cu titlul:

Titlul lucrării

Autor: **Prenume NUME**

Conducător științific: **Titlu.ing. Prenume NUME**

1. Cerinţele temei: pinpoint a location in a room which will be shown on a computer.

2. Soluţii alese: use of Atmel's ATmega328P microcontroller as a host for Decawave's DWM1000 transceiver module, which has the capabilities to communicate wirelessly with another module of the same kind.

3. Rezultate obţinute: display of location data on a PC

4. Testări şi verificări: verify that the location data that was obtained is correct

5. Contribuţii personale: designing the hardware and software used for the project

6. Surse de documentare: online medium consisting of websites and scientific papers

Semnătura autorului ______________________

Semnătura conducătorului științific ______________________

1 Introduction 3
1.1 General context 3
1.2 Objectives 3
1.3 Specification 4
2 State of the art 5
2.1 Methods 5
2.1.1 Trilateration 5
2.1.2 Fingerprinting 6
2.1.3 Dead Reckoning 6
2.1.4 Proximity 7
2.2 Indoor positioning system technologies 7
2.2.1 Positioning using Wi-Fi 7
2.2.2 Radio-frequency identification 8
2.2.3 Ultrasonic 8
2.2.4 Bluetooth 9
2.2.5 Inertial 9
2.2.6 Ultra-wideband 9
2.3 Comparison of technologies 10
3 Analysis and implementation 12
3.1 Analysis 12
3.1.1 The Decawave DWM1000 module 12
3.1.2 The ATmega328P microcontroller 14
3.1.3 The SPI protocol 15
3.1.4 Serial communication 16
3.1.5 PCB 17
3.1.6 Trilateration 18
3.1.7 Two-way ranging 18
3.2 Design and Implementation 21
3.3 Testing and validation 42
4 Conclusions 44
4.1 Results 44
4.2 Further developments 45
5 Bibliography 46

1 Introduction

1.1 General context

There are plenty of reasons for the need of information regarding the location of certain things, as it can be clearly seen from our everyday use of the GPS. In its infancy, the GPS was used mostly by the military to coordinate troops, guide missiles and track targets, until it was cleared for public use.

Unfortunately the GPS doesn't have a very high accuracy. The location that can be found using this system usually has an error of aproximately 5 meters. This average error is also the one we get when we have a clear view of the satellites used for geolocation. When inside a building, for example, the error increases.

A solution to this accuracy problem is described in this paper. The main component of this solution is the DWM1000 module from Decawave, which boasts an error in the range of centimeters, more specifically, about 10 centimeters. The module uses radio waves, which do not have the requirement of a line of sight, the signal can pass through solid objects.

1.2 Objectives

The main objective of this paper is to obtain a set of coordinates that describe the precise location in relation with a set coordinate frame. The set of coordinates will be reduced to a two-dimensional form for simplicity and so, come in the (X, Y) form, Z being omitted for now.

In order to achieve the set objective, the following steps are required to be fulfilled:

1. Achieve communication between a chosen host microcontroller and the DWM1000 module. This step consists of:
   1.1. Choosing a microcontroller
   1.2. Understanding the particularities of the chosen microcontroller
   1.3. The design of a PCB to physically connect the microcontroller with the DWM1000 module
   1.4. The software for communication

   After this step is completed, the result will be a board which will henceforth be called a transceiver.
2. Wireless communication between 2 transceivers
   2.1. Having 2 transceivers completed

2.2. Necessary software written on each

3. Wireless communication between 4 transceivers
4. Communication between one of the transceivers and a PC
5. Trilateration – calculating the coordinates based on the distances from 1 transceiver to the other 3. This is done on the PC.
6. Plotting the coordinates. Also done on the PC

1.3 Specification

For the first step the ATmega328P microcontroller was chosen due to familiarity and previous experience. The DWM1000 module uses the SPI protocol to communicate with its host microcontroller. This protocol requires 4 pins to operate. Other notable connections are the RESET and INTERRUPT pins of the module.

The connection of the 2 components is done via a PCB which was designed based on previous specifications. Software is uploaded on the board via SPCI programming.

After 2 boards have been completed for the second step, they will be tested to be able to wirelessly communicate with each other. One will transmit a packet of data, while the other will receive said packet. When the data has been correctly received, this step can be considered done.

The third step is similar to the second, the difference being the number of boards involved in the communication.

The fourth step involves serial communication between one of the boards, which has information about the distance to each of the other boards. This information will be sent to the PC for further processing. The program used to process the data on the PC is MATLAB.

The fifth step involves the calculation of the coordinates of one of the transceivers in a program of our choosing (MATLAB for this paper).

The sixth and final step consists of plotting the data that was calculated in the previous step. Here, we use MATLAB as well because of its capabilities.

2 State of the art

Outdoor location services are currently used by a wide margin of people in everyday life. The most popular use is for the purpose of navigation, nowadays every car being capable of determining its location on a map, and also every smartphone is capable of using the Global Positioning System (GPS). It is a crucial tool for the emerging autonomous driving technology and most autonomous machines that move through space use it. The GPS has seen an exponential growth since its birth and has been adopted everywhere, now being the most popular system that serves for location related needs.

The need for accurate location is currently growing. This has been sparked by the increasing number of autonomous machines such as cars, drones, robots that are used inside depots or other environments, mainly indoors. Knowing the location of persons in an emergency situation is critical in them getting medical attention in due time. Old people in a nursing home or patients in a hospital must be monitored in order to determine their well-being. Children in daycares and playgrounds have to be supervised.

The main trend that arises in this topic is the need for very accurate location, the error must be of the order of centimeters. Also, the space where the location has to be determined is inside buildings.

The main drawback that the outdoor positioning systems suffer from is the accuracy. An accuracy of 1-3 meters can be achieved by using satellites. Unfortunately, this further decreases when the object we want to locate is found in a building. This is caused by the degradation of the signal that the GPS uses when it passes through walls and other obstacles. The signal will be attenuated and scattered.

This drawback makes the GPS and other similar systems unfit for the purposes of locating objects inside buildings. For this reason, in the last decades, a great deal of effort has been put in developing a system that is suitable for this task, an Indoor Positioning System.

2.1 Methods

When it comes to the method of calculating the location there are a couple of popular principles and algorithms which provide data of different kinds, that is processed and lastly providing the result.

2.1.1 Trilateration

Trilateration is what the GPS uses in order to determine a set of three coordinates, latitude, longitude and altitude. Trilateration uses the already known location of the satellites and the distance measured from the satellites to the object that the location will be determined.

This distance is measured using the properties of the electromagnetic signal that is transmitted and received between the satellite and object. Knowing the speed at which light travels, distance is easily calculated by using the time that passes between transmission and reception. In simple terms, the distance will be equal to the speed of the signal times the duration of the transmission.

Knowing the distance from 4 satellites to the object and the exact location of the 4 satellites, we can measure the exact location of the object. The 4 distances will be considered the radii of 4 four spheres of which the centers are determined by the location of each satellite. Two spheres will intersect and have common points represented by a circle. Three spheres narrow the intersection down to 2 possible locations and the fourth sphere will determine the actual location.

This method uses the algorithm described above which is not very computationally intensive but the process requires extreme time accuracy caused by the speed of light. In a picosecond, light travels about 0.3mm, making the requirement of clocks working in the tens of GHz a necessity. Such tech is rather expensive.

2.1.2 Fingerprinting

Fingerprinting is an empirical solution to the localization problem. It employs measurements that are done in advance in a predetermined area. The measured property is the received signal strength (RSS). The RSS in an environment varies due to the obstacles that are present between the signal transmitter and the signal receiver, which can be placed in a random variation. The distance from the transmitter and the receiver affects the RSS as well.

The RSS is measured and then stored in a database along with the coordinates where it was measured. When an object wants to find out its location, it will query the database with the RSS that it has measured at its current location. The database will then correlate the RSS with the ones it has stored and return the set of coordinates that match the RSS measurement.

This is easily done due to the wide availability of wi-fi access points, but there may be the need of upgrading the already present infrastructure due to the need of improved accuracy. One of the main drawbacks is the need to map the area in which the system works, and the environments could be rather volatile, sporadic changes have the possibility of appearing.

2.1.3 Dead Reckoning

Dead reckoning is a very rudimentary technique which relies on the laws of physics. The method used for localization in this case implies the measurement of speed and direction which are used to determine position.

If our object starts from a known location and it has the capability to discern its acceleration and orientation through an accelerometer and a gyroscope, the next location can be calculated as such:

1. Start from point A
2. Determine direction using the gyroscope

3. Determine acceleration using accelerometer
4. Determine speed by integrating the measured acceleration
5. Determine position of point B by integrating the speed obtained in the previous step

In simple terms, traveling with a speed of 1 m/s for 1 second means that we have traveled for a total of 1 meter from point A to point B.

Unfortunately, the double integration necessary for this method increases tremendously the error present in the acceleration measurement. This can be amended by the use of filters and implementing a dead zone that symbolizes a position in which the tracked object cannot be, such as inside a wall.

2.1.4 Proximity

Proximity is a very simple method of localization, providing the position of an object when it is in close proximity to another of known location.

2.2 Indoor positioning system technologies

Today, a number of indoor positioning systems have surface using the above mentioned methods in order to calculate location. Some of the most popular will be presented below.

2.2.1 Positioning using Wi-Fi

The exchange of data using wireless signals in computer networks has seen widespread adoption in the last couple of decades. Most modern homes contain at least one access point and buildings have several on a single floor. The range of the wireless signals rather limited in the 20 to 100 meters range. The operation of those devices is done in the 2.4GHz and 5GHz.

Indoor positioning systems using Wi-fi signals tend to use multiple access points in order the determine a location, increasing the number of beacons also increases the accuracy of the position.

The most popular method used by this technology is fingerprinting.

The first IPS using this technology was RADAR, which makes use of two techniques in order to determine location: fingerprinting and signal propagation modeling. It has managed to achieve an accuracy of around 2-3 meters and could track multiple devices at the same time.

The most popular positioning system that uses Wi-Fi is the commercially available Ekahau positioning system. This technology uses already existing infrastructure consisting of access points. Tags were developed in order to be worn by users or tracked objects. Those tags emit Wi-Fi signals periodically which are detected on the access points. Using the data that has been gathered in the mapping process, they can detect the location of the tag using the received signal strength indicator. It has an accuracy of about 1-3 meters. The system is low cost and very simple to use with multiple tags.

2.2.2 Radio-frequency identification

Radio-frequency identification (RFID) is a technology that uses radio frequencies between 125kHz and up to 960MHz. Data is transmitted and received wirelessly between a reader and a tag. There are two ways to implement a tag:

1. Passive tag – in this case the reader does most of the work. It will transmit data which the tag will receive and reflect back to the reader. Due to the lack of amplifying circuits in the tag, the range of this implementation is rather short, about 1-2 meters. But the lack of active components in the tag makes this implementation a very cheap one.
2. Active tag – in this case the tag is also capable of sending signals. It transmits data containing ID or something else to the readers. The range in this implementation is increased up to 100 meters, but he cost rises as well due to the increased complexity of the tag.

RFID usually uses the proximity method in finding the location.

One of the systems that used RFID technology to determine location was LANDMARC (Indoor Location Sensing Using Active RFID). Using active RFID tags for the tracked objects and tags that were used as reference combined with readers placed in fixed locations. The reference tags were used in order to improve accuracy. The accuracy of the system was of about 1 meter and the maximum error less than 2 meters. The battery can also last for a very long period, up to 5 years, but it took a long time to calculate the position, especially when multiple tags were present.

2.2.3 Ultrasonic

Ultrasonic positioning systems use sounds in order to determine location. Higher frequencies are desirable in the implementation since working in the 20Hz-20kHz range, the sounds are hearable by humans and can be quite vexing. Usually, frequencies over 22kHz are used.

Systems that implement the ultrasonic method, determine the target’s position by measuring the time it takes for a signal to propagate from the transmitter to the receiver, the speed of sound propagation through air being a known constant. While not exactly expensive, this technology still requires new infrastructure in the form of sensors and transmitters.

The Active Bat positioning system uses tags that broadcast signals received by microphones mounted on the ceiling of the room where position is calculated. It can calculate the position using at least 3 receivers and the trilateration method by measuring the signal’s time of arrival. In an area with 750 receivers and 1000 square meters it was able to achieve an accuracy of 3 centimeters. The battery of the tags used in this method was able to last for about 15 months.

The Lok8 system uses smartphone speakers and the time difference of arrival method in order to calculate location. In a room with 4 receivers and of approximately 50 squared meters, it achieved an accuracy of 10 centimeters.

2.2.4 Bluetooth

Bluetooth technology is very similar to Wi-Fi. It uses the same frequency, mainly 2.4GHz. It is integrated in most smartphones and other peripherals, due to its low price.

The Topaz positioning system uses the Bluetooth technology in order to locate targets indoors. The system consists of a positioning server, access points and the tags that are placed on the tracked objects or persons. It can achieve accuracy of around 2 meters but the delay is rather big one, up to 30 seconds.

The iBeacon technology developed by Apple also uses Bluetooth technology, and was developed in order to help customers locate products in a store, but its capabilities can be extended to locating targets.

2.2.5 Inertial

Inertial positioning technology uses sensors such as accelerometers and gyroscopes to determine speed and direction. When those are obtained and having previous knowledge of the targets initial position, the location can be obtained – the dead reckoning method.

A system that uses an improved implementation of this method was proposed by Beauregard et al. The improvement comes in the form of a backtracking particle filter. The difference between the simple particle filter, which uses walls and other obstacles as locations that the target cannot possibly be in, and the backtracking particle filter is the recalculation of the trajectory when an impossible one is met. If we arrive in a wall it means that the trajectory was wrong from the start.

The system was quite successful when a detailed map of the room was available, the accuracy being 0.75 meters and 2.5 meters when a map containing just the walls was available.

2.2.6 Ultra-wideband

Ultra-wideband is a technology similar to Wi-Fi in that it uses the same frequency range, starting at 2.4 GHz, but the bandwidth of the emitted frequencies are wide, about 500 MHz in this case, for a radio to be considered ultra wide band, the bandwidth must be at least 20% of the carrier frequency. Because it uses such a wide portion of the frequency spectrum, the interferences with other narrowband technologies are avoided. It uses the time of arrival or time difference time of arrival method in order to calculate distance and coupled with a method like trilateration, it can successfully and accurately calculate location. It also has quite a low power requirement, the cost being its main disadvantage.

The ultra wide technology was successfully used by the Ubisense company. The system developed by them consists of sensors distributed across a mapped area and tags, which are attached to the objects that are being tracked. The tags have the ability to transmit and receive data within the network, as well as the sensors mounted in a known location.

The system uses time difference of arrival in order to calculate distance and location. The tag's signal need to be received by at least two sensors in order to calculate the 3D location of a tag. The accuracy is of about 15 centimeters. The location of the tags is known at any time, the delay being very small. The main disadvantage of this system is the cost.

2.3 Comparison of technologies

Table 2.1

IPS	Technology	Accuracy	Cost	Advantages	Disadvantages	Complexity
RADAR	Wi-Fi	2-3m	L/L	Price, infrastructure	Accuracy, complexity	M
Ekahau	Wi-Fi	1-3m	H/L	Infrastructure	Mapping	L
LANDMARC	RFID	2m	H/L	Cheap	Delay	M
Active Bat	Ultrasonic	3cm	H/L	Cheap, precise	Requires many beacons, battery	H
Lok8	Ultrasonic	10cm	L/L	Smartphones, precise	Infrastructure	M
Topaz	Bluetooth	2m	L/L	Price	Delay	M
iBeacon	Bluetooth	0.5-3m	H/L	Smartphones	Req. beacons	L
Beauregard et al.	Inertial	0.74-2.5m	L/L	Cheap	Detailed map	M
Ubisense	UWB	15cm	H/H	Precise, robust	Expensive	H

Taking into consideration the factors that were presented in the table above, mainly accuracy and response time, a technology was chosen in order to suit our needs. Since technologies such as Wi-Fi, Bluetooth, RFID and inertial don't have a very good accuracy and some of them have a locating delay which can be quite a burden, a choice had to be made between the ultrasonic and ultra wide band technology.

Both of these technologies offer an acceptable accuracy and real time location, the difference was made by the other factors such complexity, cost and other miscellaneous factors. The ultrasonic solution requires a lot of beacons in order to be effective compared to the ultra wide band alternative. Also, the battery life in the case of the ultrasonic solution is worse while the ultra wide band has low power requirements. The medium in which those technologies work efficiently also differs. The ultrasonic solution that uses sounds can become useless in environments that absorb sounds such as spongy materials. The ultra wide band solution doesn't require a line of sight to the tracked object and can penetrate a number of materials due to the properties of the carrier radio wave.

Due to those reasons, the ultrasonic solution was not chosen, going in the favor of the ultra wide band one. After some further research, the DWM1000 module from Decawave was chosen in the implementation of this solution. The module is a transceiver which makes it perfect for implementing an indoor positioning system with just four devices, the number increases with 1 for each new target that needs to be tracked. The devices will measure distance between the tracked tag and at least 3 other devices which are placed in locations whose coordinates are known. The company boasts an accuracy of 10 cm when calculating distance.

Once to distance from the tag to the 3 anchors is known, a location can be calculated using the trilateration method.

3 Analysis and implementation

3.1 Analysis

3.1.1 The Decawave DWM1000 module

Transceiver, data rate, radio, spi, form factor, time

The main component that this paper makes use of, and around which the whole implementation revolves around is the Decawave DWM1000 transceiver module. As the name implies, the transceiver is able to transmit and receive data. This is done wirelessly via radio waves. It can use up to 4 radio frequency bands bounded by the 3.5 GHz and 6.5 GHz. This method of data transmission is a must because of the well known properties of radio waves.

Radio waves are characterized by the frequency range which they occupy in the electromagnetic radiation spectrum (radio being a type of electromagnetic radiation), this being lower bounded by 30Hz and upper bounded by 300GHz. A radio wave is basically an oscillation which occurs at a frequency in the afore mentioned range, of electric and magnetic fields that constantly create one another. The speed at which this happens is the speed of light, a well known constant, having a value of 299,792,458 metres per second.

Radio waves can occur naturally, being emitted by various weather phenomena, most notably lightning, astronomical objects, which are the main source of radio waves through stars. But, radio waves can also be generated artificially. By having a certain signal of electric current that is amplified and passed through an antenna, electromagnetic radiation will be generated with the antenna as a source and it will start propagating in space, usually omnidirectionally, but that can also be changed by the design of the antenna.

An antenna is also used on the receiving end of the signal. The antenna captures the signal which is passed through a circuit for further processing.

Using the process described above, we can send data from point A to point B wirelessly. The main properties of the signal sent are its amplitude and frequency. And thus we can send bits of data by changing those properties. This is done by modulating the signal that is sent along the air.

There are 2 main methods for modulating a signal based on the 2 properties mentioned above, but there are also other possibilities.

Amplitude modulation is the process of encoding information using the amplitude of the radio wave. For binary data, we can impose that an amplitude denotes the value 0 and another amplitude the value 1. And so, choosing 2 Volts for 0 and 5 Volts for 1, when the amplitude of the wave in the current period hits a maximum of 5V, it is interpreted as a 1, the process being similar for 0.

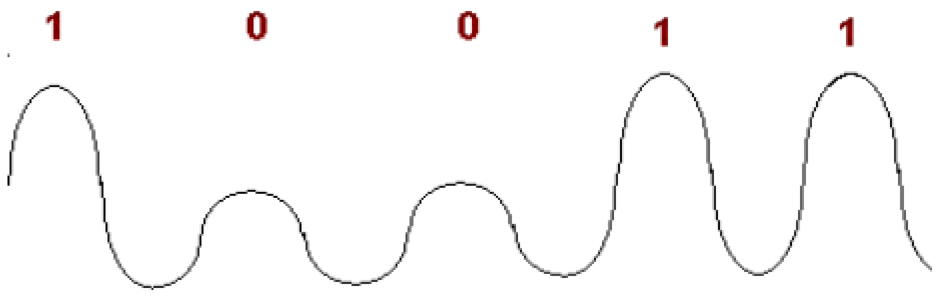


Figure 3.1

Frequency modulation is the process of encoding information using the frequency of the radio wave. The process for modulation is very similar to the previous case, but instead of amplitude we modify the value of the frequency.

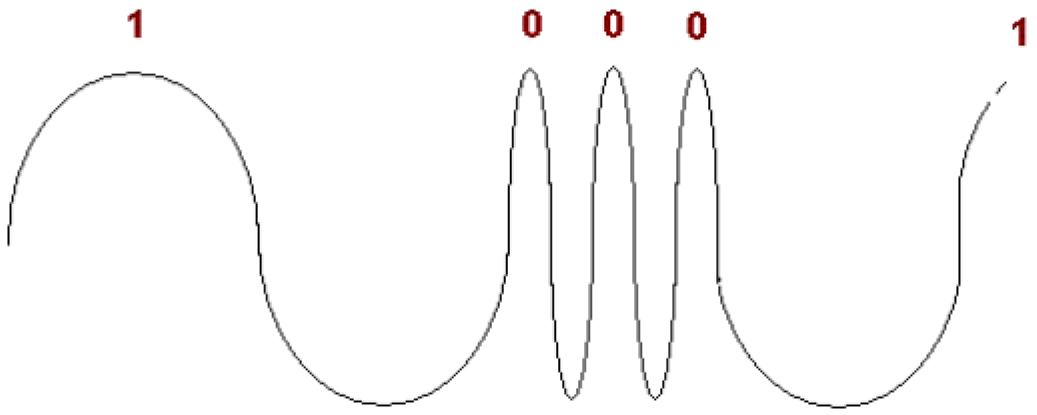


Figure 3.2

Another type of modulation is the phase modulation, which is the process of encoding information using the phase of the radio wave. For example, a 0 signal will be represented by a 0° change in phase, while a 1 will be represented by a 180° change in phase.

Those 3 types of modulation are the basic ones when it comes to this process. Modern wi-fi signals use something a bit more complex, mainly, Quadrature Phase Shift Keying. It is similar to the Phase Shift Modulation method, the difference being the number of states the wave can be in. If we increase the number of states, we increase the number of symbols sent. In this case we send 4 symbols, which are 00, 01, 10 and 11. Those are represented by a difference of 90° change. So, starting with an offset of 45°, we have 45° to represent 00, 135° to represent 01, 225° to represent 10 and 315° to represent 11.

The DWM1000 uses Binary Phase-Shift Keying.

The DWM1000 module transmits data with frequencies between 3.5Ghz and 6.5Ghz on 4 different radio frequency bands. Those 4 channels are 1, 2, 3 and 5 and they have a bandwidth of 500MHz. The centre frequencies for those channels are the following: 3494.4MHz for channel 1, 3993.6 MHz for channel 2, 4492.8 for channel 3 and 6489.6 for channel 5.

The data rate is the amount of data that is sent per unit of time, and in our case is equivalent with the number of bits that are sent in a second since we only have 2 states, hence 1 symbol equals 1 bit. The DWM1000 module supports 3 different data rates. Those are: 110 kbps(kilo-bits per second), 850 kbps and 6.8 Mbps(mega-bits per second).

One of the main features of the DWM1000 module that allows it to be used for the purpose of our paper is the sampling clock that is used for ranging. This clock works at 63.8976 GHz which means it has a time period of 15.65 picoseconds. This clock is what allows us to calculate the distance between 2 transceivers with such a high accuracy. Since we know the speed at which light travels and of course, radio waves, we know that in a picosecond, the distance that has been traveled is about 0.3 millimeters. Multiplying that with our value of 15.65 we get 4.695. This number represents in millimeters the minimum error in millimeters that we can have in our system when we measure distance. The error is further increased to the range of centimeters because of other factors in the circuit, such as the antenna delay and the processing of the microcontroller.

3.1.2 The ATmega328P microcontroller

Even though our module is quite powerful, it cannot do the job we want it to alone. It is capable of sending messages through radio waves, but it can’t do so by itself, it needs a brain to control it. This problem is solved with the help of a microcontroller.

A microcontroller is very similar to a computer: it has a processor, memory (ROM and RAM), Input/Output pins that let it interact with various peripherals like our module, and all of this comes on a single chip of very small size which makes it perfect for our embedded application.

To make use of a microntroller, first we must upload code unto it. The code is stored in its ROM memory (usually Flash). When the microcontroller boots, the CPU, which comes in options as simple as 4-bit to more complex such as 32-bit or 64-bit, fetches the instructions stored in memory and starts executing them. This behaviour makes microntrollers suitable for a certain task that it does over and over again. Tasks such as awaiting data from sensors and doing something based on the received data, periodic tasks, simple robots, and of course controlling peripherals.

The microcontroller that we choose has the task telling our DWM1000 module to send data and listening for the data that the module sends back. The data involved in this process will then be processed and sent further to a PC.

The microcontroller that was chosen for this project is Atmel’s ATmega328P. This is a high performance 8-bit from the AVR family of microntrollers. It is based on the RISC (Reduced Instruction Set Computing) architecture, which as the name implies uses a reduced number of instructions that total 131 and the execution time is of a single clock cycle for most of them. This makes it capable of achieving 16 MIPS (million instructions per second) at 16 MHz. It has 32 x 8 general purpose registers.

It has 32 Kbytes of flash memory that is used to store the program data, an additional 1Kbyte of EEPROM memory for data and 2 Kbytes of internal SRAM. The flash memory supports up to 10,000 writing and erasing cycles, while the EEPROM up to 100,000 cycles.

The locking of a certain section of the memory for the purpose of a boot sector is also possible.

Regarding peripherals, it has two 8-bit Timer/Counters and one 16-bit Timer/Counter. Six PWM channels and an 8 channel 10-bit ADC that can be used to interpret analogue sensors.

The operating voltage for the ATmega328P is between the 2.7V and 5.5V range. This is very important since the DWM1000 module operates at the 3.3V level, which is included in the above mentioned range. The small physical size of the microntroller (34.5mm x 8mm) is also useful for the final implementation of the project.

Regarding communication peripherals, the ATmega328P has a couple to choose from. The I2C which stands for inter integrated circuit, also pronounced as I squared C, is a serial communication protocol which uses a master/slave implementation, where the master sends data to a slave on a common bus using an addressing format.

The UART, which stands for Universal Asynchronous Receiver-Transmitter, is a simple serial communication protocol, having only two pins, the Rx and Tx, used to transmit and receive serial data. This is the protocol that we use with our PC with the help of a simple USB to UART converter.

The most important peripheral feature that this microntroller has and we need for our project is the Master/Slave SPI serial interface that is used to communicate with the DWM1000 module. The microntroller acts as a master while the module is a slave in this exchange of data.

3.1.3 The SPI protocol

Communication involves messages sent in two direction with the involved parties playing the role of receiver and transmitter, one sending data and the other receiving said data and vice-versa. The SPI (Serial Peripheral Interface) is a synchronous serial data protocol that facilitates the communication between the two involved parties. The relationship established between the 2 devices that are communicating is a master/slave one, meaning that the master is the device that sends instructions to the slave, while the slave executes said instructions. Also one master is able to control more than one slave, and can choose which slave to control.

The physical layer of the SPI protocol consists of the 4 wires that are used to connect the devices and looks something like this:

- MOSI (Master Ouptup/Slave Input) – the wire on which the master sends data to the slave
- MISO (Master Input/Slave Output) – the wire on which the slave sends data to the master
- SCLK (Clock) – the wire on which the clock signal used for synchronizing is sent
- SS (Slave Select) – the wire on which the signal for selecting the slave is sent

SPI works by first selecting the slave chip that the master wants to communicate with. After this, the clock signal is started by the master which means that data is about to be sent. This signal also synchronizes the data since bits will be sampled according to the clock signal. The speed of the transfer is also dictated by the frequency of the clock. Higher frequency meaning higher speeds and lower frequencies lower speeds.

After the clock signal, data starts to stream from the master through the MOSI line one bit at a time and the slave reads those bits as they are received. Data sampling starts with the first edge of the clock signal. Then, if a response is needed from the slave, it will be received on the MISO line in a similar way to the process of sending data from master to slave.

After the transfer is complete the clock signal is stopped, meaning that the data transfer is over for now.

3.1.4 Serial communication

The data that is gathered on the microcontroller has to be processed further in order to make actual use of it. This means that the microcontroller has to send it to the PC where the calculations happen. We have already talked about communication using the SPI protocol, but the PC doesn't use it as the DWM1000 module does.

The most common communication protocol used by PCs is the USB protocol. It is so popular because of its data rates up to 480 Mbits/sec, the distance it can transmit the data through the cable, the standard 5V power with up to 500mA current. Those specifications are for the USB 2.0, there being improved versions, mainly USB 3.0 and other which have raised the capabilities of this extremely popular standard.

The ATmega328P, in terms of serial communication has an UART serial port. When it comes to microcontrollers, the UART port is one of the most common found incorporated. It is quite simple since it requires a small number of pins, three to be exact, those being the Rx pin, the Tx pin and the GND pin which the previous two levels reference.

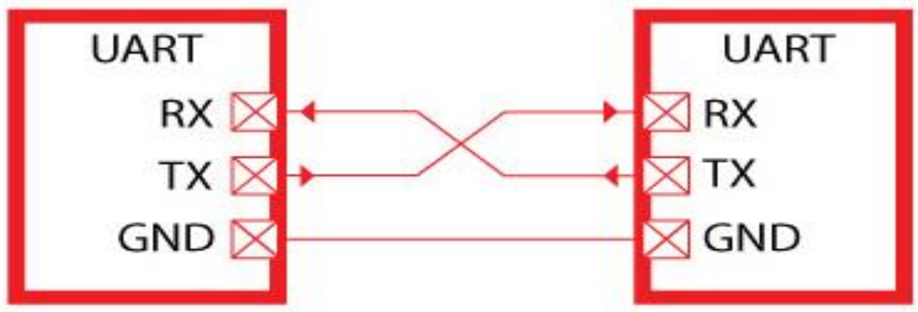


Figure 3.3

One of the features that makes the UART protocol different from the SPI protocol is the asynchronous type of data transmission. In this case, the clock will not be transmitted on a separate wire, but a fixed baud rate is determined in advance. This baud rate represents the speed of the transmission and in this case represents bits per second. Also, it can have any value, but usually a standard one is chosen, the most common being 9600. The transmitter will send the data with a certain speed while the receiver will sample the data at the same speed.

The communication goes through the two Rx and Tx lines mentioned above. This can also be:

- Simplex – only the transmitter sends data to the receiver
- Half Duplex – one device sends data while the other receives, then the roles change
- Full Duplex – both devices can send and receive data at the same time

The conversion necessary to be able to communicate between these two protocols is rather easy, requiring just another integrated component. This is one of the reason that this solution was chosen. The UART data will be sent to the component which will then convert it to the USB format and send it further to the PC which will then interpret it.

3.1.5 PCB

The components need to be physically attached to one another in order for the electronic components to be electrically connected to one another. This can be done with the help of some wires and a breadboard, but this implementation is only good for prototyping and testing reasons since the structure is rather feeble, with wires hanging in the air, and worse still loose connections can appear which can cause short-circuits in the system, destroying electrically sensible components.

For this reason, the final implementation of the project was chosen to be on a Printed Circuit Board. A PCB is a board that harbors the necessary connections between the components. Copper paths run from one hole or pad to another and even through the PCB itself, allowing multiple layers to be included in a single board. This allows the total number of connections, which can span for a sizeable amount of meters, to be realized in a very compact space. Also, the type of path can also be specified, allowing larger paths for current intensive components and smaller ones for signal and command paths. The connections will also be covered in a non-conductive layer to prevent unwanted short circuits, leaving only some small areas unprotected where the components will be soldered.

After the PCB is designed, components will then be soldered unto it either directly on pads or through the PCB itself. The method depends on the components used, the latter being the most common for bigger components like resistors, capacitors, certain ICs, and the former being the most common for Surface Mounted Devices (SMD) which are very small.

Our project will use both methods since the DWM1000 module is an SMD and the most common components available are through hole.

3.1.6 Trilateration

Trilateration is the same algorithm that the GPS uses. It is based on the mathematical principle of the same name. In two-dimensional geometry, this principle states that if a point lies on three circles of which the radii and centers are known, we can calculate the location of said point in relation to the coordinate frame that the circles use. If only two circles are used, the possibilities increase to two, hence a minimum of three circles must be used to deduce the true location.

This algorithm is quite effective since we only need to know the distance from our measured point to the other three points that are already known, being chosen by us.

3.1.7 Two-way ranging

Knowing the speed at which light travels it is possible to calculate distance based on the time a message took to get from one device to the other. The formula is quite simple, involving just a multiplication:

Distamce = ToF x c,

Where ToF is Time of Flight, how much it took to get from point A to B, and c is the speed of light constant. The hard part is obtaining the ToF variable.

The method used for calculating the distance is called two-way ranging. This method involves three messages that are sent between two devices in order to calculate the distance between them. The process is the following:

1) First, one device sends a Poll message to the other one. This message contains the timestamp of when the message was sent.
2) The receiving device records its own timestamp of when the message was received. Then, a response is issued back to the first device. Again the time is recorded.
3) The initiating device records the time when it receives this message and a final message is composed and send back to the second device where the distance will be calculated.

We note the timestamps the following way:

- Tsp – time when the poll message was sent (Device1)
- Trp – time when the poll message was received (Device2)
- Tsr – time when the response was sent (Device2)
- Trr – time when the response was received (Device1)
- Tsf– time when the final message is sent (Device1) – this message contains all the previous timestamp data
- Trf – time when the final message was received (Device2)

Using those notations, we can calculate the Time of Flight with the following formula:

$$ToF = \frac{(Trr - Tsp) - (Tsr - Trp) + (Trf - Tsr) - (Tsf - Trr)}{4}$$

There is also a simpler method that can be used to calculate the ToF, involving only a transmission and a reply. This is called Single-sided Two-way Ranging. It uses the simple measurement of the round trip delay of a single message from one device and a response to the original device from the second one. The graphical representation is presented below:

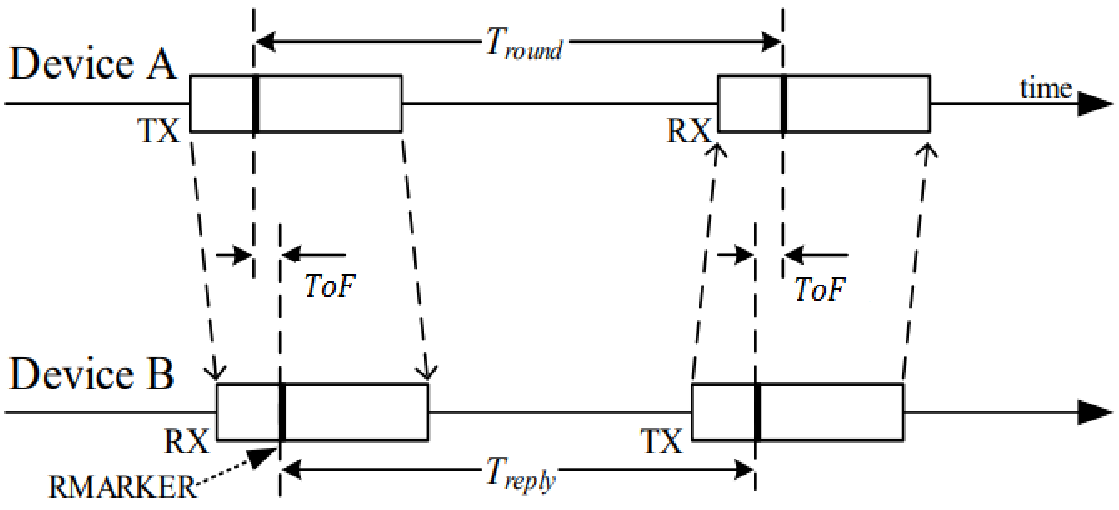


Figure 3.4

Device A initiates the exchange and Device B responds to complete it. Each device timestamps the transmission and reception times. The sum of the 2 $ToFs$ can be calculated by simple subtraction of T_{round} and T_{reply}. The answer is then divided by 2 in order to obtain the final ToF.

Now, we have all the variables to calculate the distance.

In the following section of this paper, the steps necessary to be taken in order to bring this project to completion will be presented, and also the reasoning behind them. The objective is to plot location data on a PC, data that will be gathered with the help of other 4 main components. These components will be henceforth called ANCHORS and TAGS. The ANCHORS and TAGS have no physical differences, they are made of the same components connected in the same way, the only difference being the software written on them. Also notable, the ANCHORS will be physically stationary, fixed in a place where the spatial coordinates are known or set by us. The TAGS will move through space and it is necessary to calculate the spatial coordinates of those boards.

The list of necessary components to make the realization of one board feasible is the following:

- DWM1000 module
- ATmega328P microcontroller
- Custom made PCB

- 8 MHz oscillator
- 2 x 22 pF capacitors
- 3.3V voltage regulator
- 100 nF capacitor
- 2.2 μF capacitor
- Mini-USB cable
- Mini-USB mother connector

The bulk of the process consists of designing the PCB on which the components will be connected and most of the process takes place. The data will then be sent through the UART serial interface to a converter which translates it so it can be understood by the USB protocol used by the PC.

The data that is sent consists of a set of the distance measured from the board, which is moving, to one of the other boards which are stationary. This must be processed further in order to obtain a set of spatial coordinates in the XY form.

After the set of coordinates has been calculated, a visual plot will be made using MATLAB in order to place the obtained coordinates in a physical space.

The whole project revolves around the DWM1000 module so most of the specifications of the project were chosen based on its capabilities and limitations. The module requires a host microcontroller that has a couple of features, mainly:

1. SPI communication to exchange messages between the module and the microcontroller
2. Working voltage level of 3.3V

Since the module can consume up to 160 mA of current, which can't be supplied by most microcontrollers, the limit being around 40 mA, it must have an external power supply.

We also need to send serial data to he PC, meaning the microcontroller has to have or another type of serial communication that can be easily converted.

As mentioned above, the power levels of the microcontroller must be of 3.3V so that we do not harm the module that has a max rating of 3.6 volts. This means that we need a regulated power supply of 3.3V.

A microcontroller needs a clock source and even though most of them have an integrated RC oscillator, an external source is desirable since it performs better.

And lastly, the PCB on which the module will be placed has some restrictions coming from the use of radio waves. The material, copper, in the PCB can interact with those radio waves creating undesirable results, and because of this we must be careful with the physical placement of the module on the board.

3.2 Design and Implementation

Having all the components available, we must decide how the to electrically connect them together. Again, we start this step with the DWM100 module. Below a picture of the diagram of the module is presented:

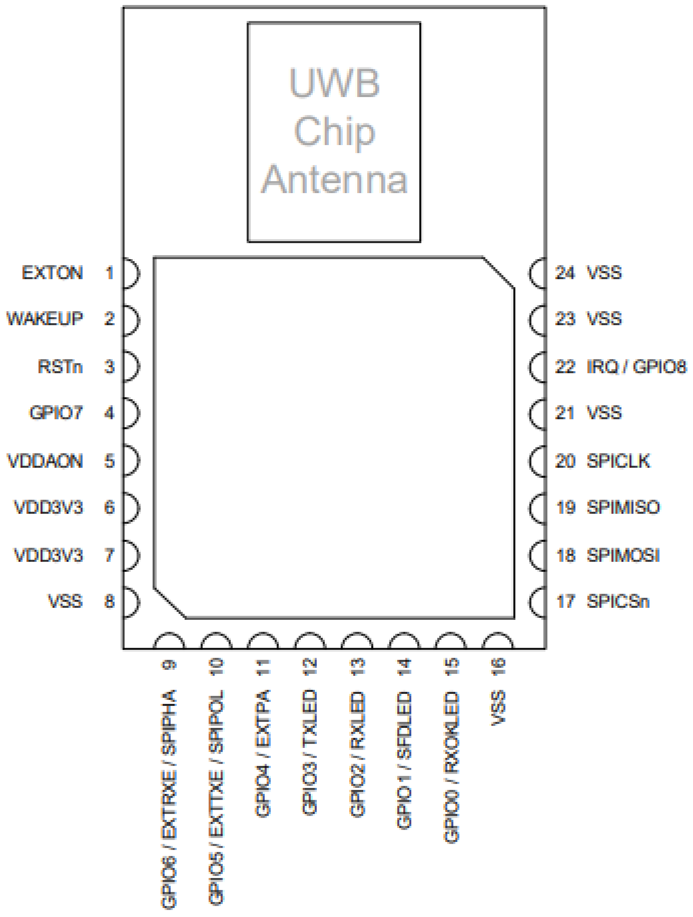


Figure 3.5: DWM1000 Pin Diagram

The module has 24 pins that we can make use of, but for this project only a couple of them will be used, those for supplying power and those regarding communication. Those are the following:

Table 3.1: Pin Descriptions

SIGNAL NAME	PIN	I/O (Default)	DESCRIPTION
SPICLK	20	DI	SPI clock.
SPIMISO	19	DO (O-L)	SPI data output.
SPIMOSI	18	DI	SPI data input.
SPICSn	17	DI	SPI chip select. Active low enable input.

IRQ/GPIO8	22	DIO (O-L)	Interrupt Request output from the DWM1000 to the host processor. Default active high.
RSTn	3	DIO (O-H)	Reset pin. Active Low Output.
VDDAON	5	P	External supply for the Always-on (AON) portion of the chip.
VDD3V3	6,7	P	3.3V supply pins.
VSS	8,16,21,23,24	G	Common ground.

Table 3.2: Explanation of Abbreviations

ABBREVIATION	**EXPLANATION**
D	Digital
I	Input
O	Output
IO	Input/Output
O-L	Defaults to output, low level after reset
O-H	Defaults to output, high level after reset
P	Power Supply
G	Ground

The next step after determining which pins of the module we will use is to connect them to the microcontroller. Since we have already determined that most of the microcontrollers available are not capable of the output the necessary current that the module uses in its most intensive states, the connections of the power supply do not concern any connections between the module and microcontroller even if they use the same power supply. For now we focus on the SPI communication lines and leave the power supply connections for later explaining in this paper.

From the pins that we have already highlighted in the previous table, we will now use the 4 pins regarding SPI communication – SPICLK, SPIMISO, SPIMOSI, SPICSn – and the interrupt (IRQ) pin and reset (RSTn) pin in a connection that is presented below:

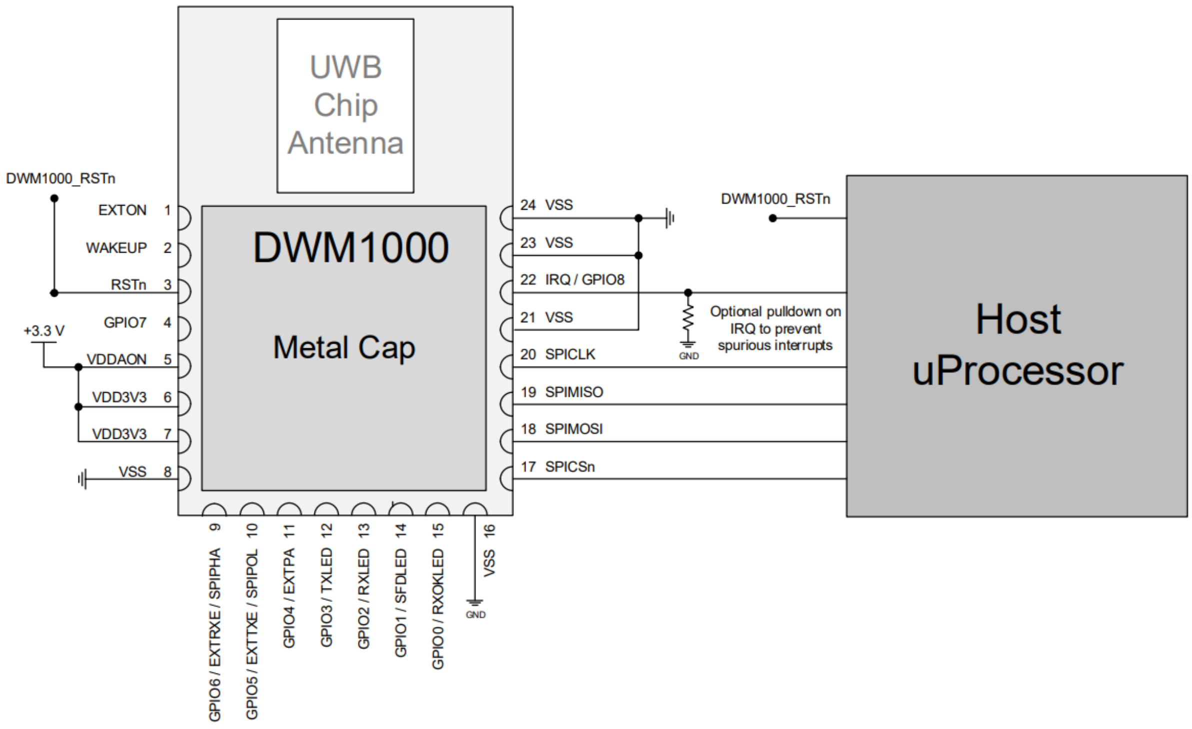


Figure 3.6: Connection between the DWM1000 and Microcontroller

The type of SPI communication that happens between the two components is, of course, reading and writing data. The DWM1000 module is capable of sending data that the microcontroller tells it to and also, it is capable of receiving data when the microcontroller tells it to expect it.

This is done mainly by the use of the receiving and transmitting data buffers. Those buffers store the data that is meant to be sent and collect the data that was received. Note that the receiving buffer is double the size of the transmitter buffer. The reason for this is the need to receive a new data frame while the microcontroller still processes the previous buffer.

The data frame that is written in the buffers has the following structure:

1. The preamble – this part is used as an introductory part of the transmission that lets the receiver know that someone is about to send data
2. SFD – Start of Data Frame – indicates the completion of the preamble section of the frame and the start of the next section
3. PHR – the Physical layer Header – this is the section in the frame that comes right after the SFD and before the actual message payload and defines the characteristics of the whole frame that the receiver must know
4. Data – the payload, the data that actually interests us, everything else just facilitates the communication

In case of a transmission, the microcontroller uses the SPI interface to write a certain data frame on the transmit buffer of the DWM1000 module. Then, the parameters of the transmission will be configured, those being the data rate, the channel used for transmission, the power of the transmission, etc. After the writing has been successfully done, a new message is sent to the module, telling it to send the data that it has written on its buffer. This begins the process of data transmission through radio waves involving modulation. After this has been successfully done, the module will notify the microcontroller that a successful transmission has been made. This is the last step in a transmission procedure and the module will go into another state or begin the transmission process anew.

In case of reception, the host controller must first set the parameters of the reception, similar to the transmission step. Then, the reception process truly starts. The module will hunt for a certain frame of data that signifies the start of a reception. The data frame in question is the preamble. It is possible to abort the reception if a timeout in preamble detection is implemented. This means that if a preamble is not detected for the set amount of time, the process will restart from the first step.

After a preamble is detected, the detection of the SFD frame commences. This is a key event in the reception of a frame since it marks the start of the PHY header where timestamp is, and a change from preamble demodulation to BPM/BPSK demodulation of the PHR and data. It is also possible to implement a timeout for this step as well.

The PHR has the role to convey the length of the data portion of the frame and to indicate the data rate employed for data demodulation. The data octets are then received and stored in the reception buffer. After a successful reception of a frame, the module signals the host microcontroller which then proceeds to read the data that has been received from the buffer. New data can still be received because of the presence of a second buffer which can store new incoming data while the host microcontroller processes the data from the first buffer.

After we have defined the needs of our module, we have to connect it with the microcontroller of our choosing which is the ATmega328P-PU model. Below a picture of the diagram of the microcontroller is presented:

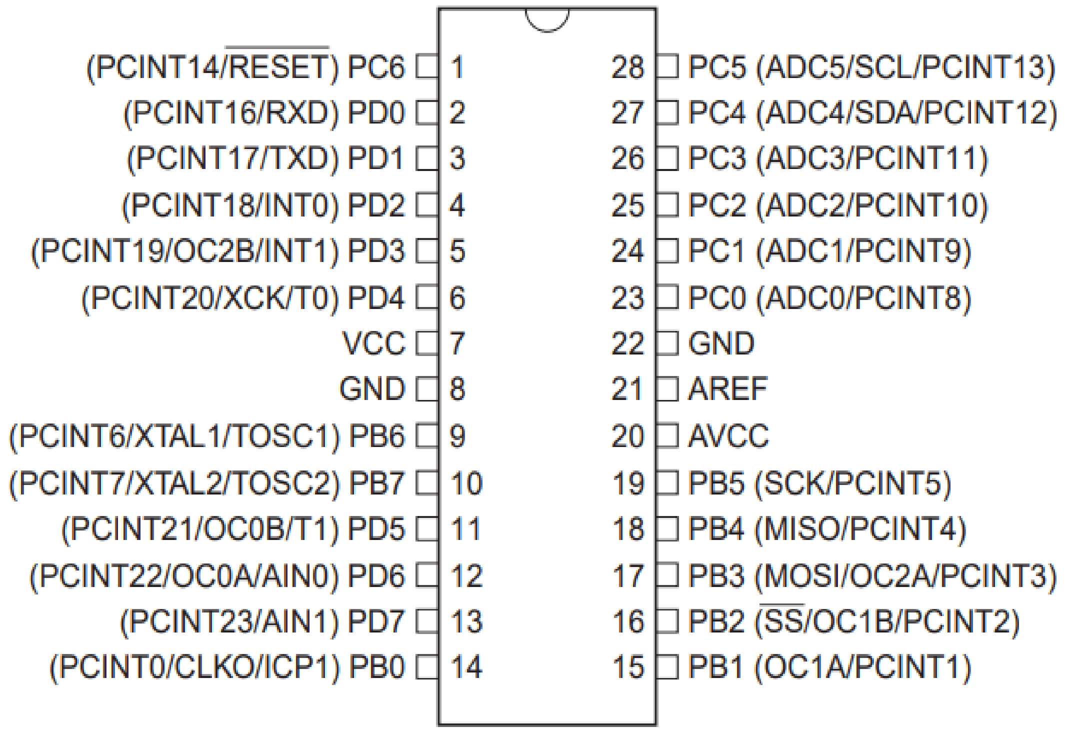


Figure 3.7

The microcontroller has a number of 28 pins that we can make use of, but for the purposes of this project, we will use only the pins for communication with the DWM1000 module and the PC, and the required power pins. The short general pin descriptions is as such:

- **Port B (PB7:0) –** Port B is an 8-bit bi-directional Input/Output port with internal pull-up resistors.
- **Port C (PC5:0)** – Port C is a 7-bit bi-directional Input/Output port with internal pull-up resistors
- **Port D (PD7:0) –** Port D is an 8-bit bi-directional Input/Output port with internal pull-up resistors

Those pins and their purpose is presented in the table below:

TABLE OF PINS

The ATmega328P microcontroller comes without any type of program written unto it, and so, it must be programmed with the software that it will run. There are 2 ways to write software on the microcontroller:

1. **In-Circuit Serial Programming (ICSP) –** this method allows the microcontroller to be programmed by use of the SPI pins, MISO, MOSI and SCK along with the reset pin and the power pins which are required for the microcontroller to work. This method also requires a dedicated programmer tool.

2. **Serial programming** - this method requires just 2 pins for data – the Tx and Rx, and of course power. A bootloader is also necessary to be written on the microcontroller in advance which means the chip must be programmed first with ICSP method, but this needs to be done only once. Another disadvantage of this method is the space required by the bootloader, which could be used for the application data. This method requires just a data cable and no additional hardware.

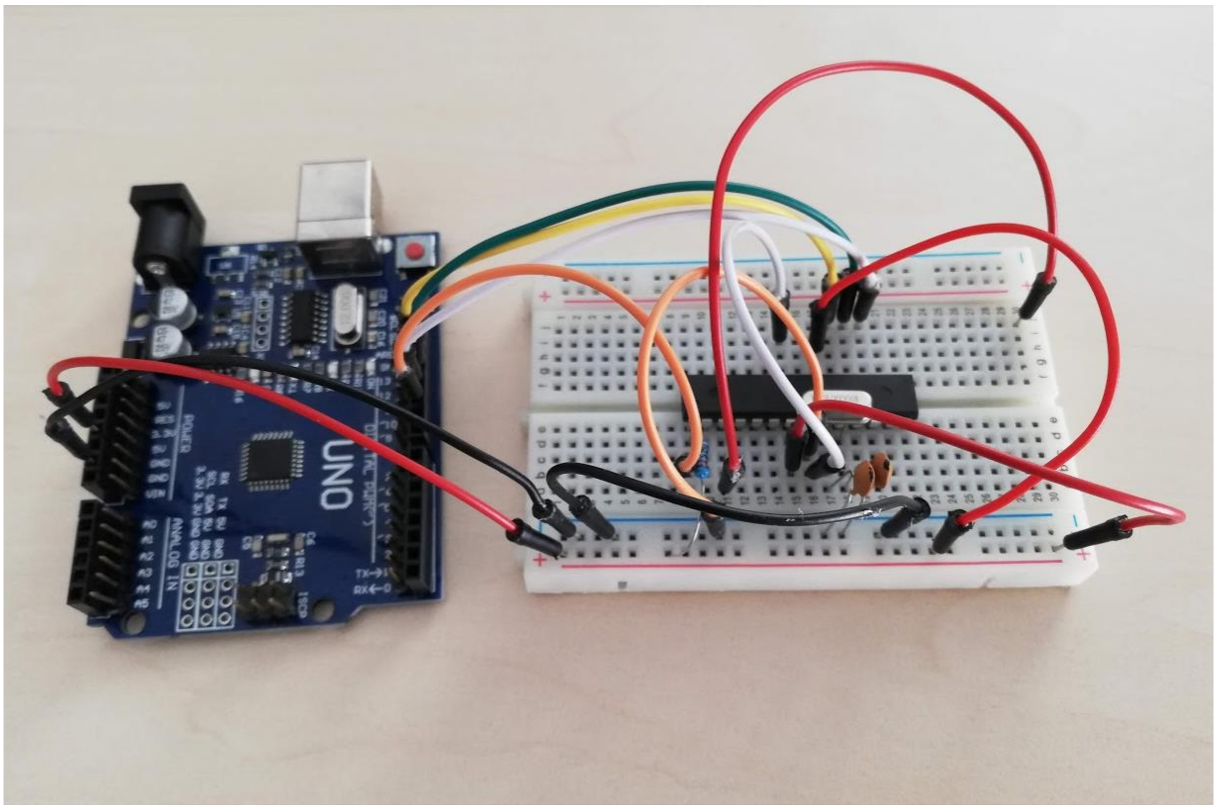
Before we integrate the circuits on the final implementation on the PCB, testing will be done off-board. As mentioned above, the microcontrollers come wiped out of the factory so the ICSP method is a must in order to program them. The programming can be done using a specialized programmer tool or an Arduino board which is programmed with the code of a programmer tool.

Arduino boards are the product of an open source project which intends to develop easy to use hardware and software. The board revolves around the microcontroller which is usually of the ATmega family, facilitating the use of it through the additional hardware it needs to operate successfully such as: an oscillator, a serial to UART converter for PC communication (sending and receiving data from the PC and programming the board), multiple powering options such as directly from the USB connection or with a 9V battery, indicator LEDs, easy access to the pins for input and output and various other electronic components (capacitors, resistors, diodes etc.).

The Arduino boards also have software which was specifically coded for them and an IDE used for writing the software on the board. This IDE contains a lot of helpful resources like tutorials, examples and libraries used for various purposes like communication (SPI, I2C), defining pins as Input/Output, changing the state of pins from High to Low and vice-versa, and other useful features. The program is written in C/C++ code which then the IDE compiles in flashable hex files which are written on the board itself.

As the board is similar in hardware to the actual programmer tool, it can be converted into one with the appropriate software. The board will be connected to the PC through an USB cable which is used for sending the programming data to the board. The board then converts the data and sends it through the SPI pins to the ATmega328P microcontroller, writing data to its flash memory.

The programming will be done on a breadboard for the first part with later implementation on the PCB. The setup looks like this:

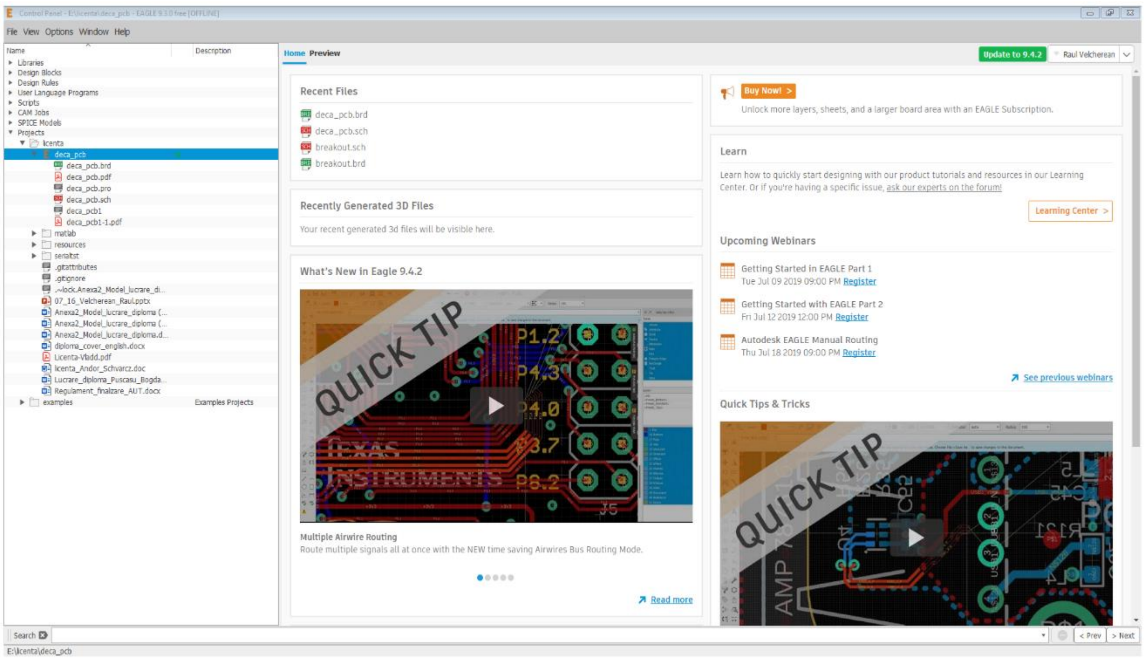


After the initial testing and satisfactory results, the project is to be ported on a PCB. This implies the design of the PCB itself and other circuits that are implemented unto it. We have already established the connections between the microcontroller and DWM1000 module, and what remains is the circuit that supplies power to the whole board and additional electronic components.

There are a couple of possibilities of approach when we design a PCB. It can be done on paper and then later ported on a board that is covered in copper and treated with ultraviolet light and chemical substances that remove the layer etched on the paper.

The method chosen for this project was one that employs computer aided design (CAD). Computer aided design represents the use of computers in order to create, modify, analyze, optimization and even simulation of a design. Its capabilities allow users to design faster and more efficiently since flaws in the initial design can be identified in an easier manner than by simple human observation and interaction.

The program chosen was Autodesk's EAGLE which is electronic design automation, a type of CAD software specific for electronic systems design. The software allows users to create schematics, place components on the board, route the copper traces and other features. It contains a number of libraries for the most common components used nowadays but new ones can be easily added and created by users. It has a friendly user interface and comes in a free version.


Control Panel - E:\licenta\deca_pcb - EAGLE 9.3.0 free [OFFLINE]
File View Options Window Help
Name
Description
Libraries
Design Blocks
Design Rules
User Language Programs
Scripts
CAM Jobs
SPICE Models
Projects
licenta
deca_pcb
deca_pcb.brd
deca_pcb.pdf
deca_pcb.pro
deca_pcb.sch
deca_pcb1
deca_pcb1-1.pdf
matlab
resources
serialtst
.gitattributes
.gitignore
.~lock.Anexa2_Model_lucrare_di...
07_16_Velcherean_Raul.pptx
Anexa2_Model_lucrare_diploma (...
Anexa2_Model_lucrare_diploma (...
Anexa2_Model_lucrare_diploma.d...
diploma_cover_english.docx
Licenta-Vladd.pdf
licenta_Andor_Schvarcz.doc
Lucrare_diploma_Puscasu_Bogda...
Regulament_finalizare_AUT.docx
examples
Examples Projects
Home
Preview
Update to 9.4.2
Raul Velcherean
Recent Files
deca_pcb.brd
deca_pcb.sch
breakout.sch
breakout.brd
Recently Generated 3D Files
What's New in Eagle 9.4.2
QUICK TIP
Multiple Airwire Routing
Read more
Buy Now! >
Learn
Learning Center >
Upcoming Webinars
Getting Started in EAGLE Part 1
Tue Jul 09 2019 09:00 PM Register
Getting Started with EAGLE Part 2
Fri Jul 12 2019 12:00 PM Register
Autodesk EAGLE Manual Routing
Thu Jul 18 2019 09:00 PM Register
See previous webinars
Quick Tips & Tricks
QUICK TIP
Search
< Prev
> Next
E:\licenta\deca_pcb

Your recent generated 3d files will be visible here.

Route multiple signals all at once with the NEW time saving Airwires Bus Routing Mode.

Unlock more layers, sheets, and a larger board area with an EAGLE Subscription.

Learn how to quickly start designing with our product tutorials and resources in our Learning Center. Or if you're having a specific issue, ask our experts on the forum!

Figure 3.9

The first step in a circuit design in EAGLE is creating the schematic. The schematic contains all the components of the circuit and the connections between them. The components each have a number of pins that will help to physically connect them together. We have already decided on the connections between the microcontroller and the DWM1000 module.

The next step is to choose the appropriate power supply for our circuit. We currently have only two components in our circuit that require power: the microcontroller and the DWM1000 module. The module requires a voltage of 3.3V from the power supply and it can draw up to a maximum of 200 mA of current in certain transmission modes, current which the microcontroller cannot provide. On the other hand, the microcontroller has a larger interval of operating voltages. It can function with a supply of a minimum of 1.8V and a maximum of 5.5V.

There are 2 possible ways to implement the power supply:

1. Supplying the board with power and regulating the voltage for each one of the components.
2. Supplying the board with power and using a single voltage level in order to power both of the components.

The first options allows us to use a higher voltage level for out microcontroller which has one main advantage: the frequency at which the microcontroller works. The specific ranges are as follows:

- 1.8V-5.5V – it can work at 0-4MHz
- 2.7V-5.5V – it can work at 0-10MHz
- 4.5V-5.5V – it can work at 0-20MHz

The higher the supply voltage, the higher the frequency and of course higher speed. Some of the standards for oscillators are 8MHz and 16MHz, and even though we could run the microcontroller at 3.3V and 16 MHz it will be unstable.

Also, this implementation requires 2 regulating steps, one for the microcontroller and one for the DWM1000 module. Another problem encountered in this case is the SPI communication. The microcontroller uses the voltage levels it is supplied with in order to determine logic levels, logic level '1' being the supply voltage and logic level '0' being 0V or close to it. Since the DWM1000 module can't take levels over 3.3V, it is necessary to convert the voltage levels from let's say 5.5V to 3.3V. This can be done with a specialized logic level converter or with a simple voltage divider. If we know the input and output voltage, we can calculate the resistances need with the following formula:

$$V_{out} = \frac{R_2}{R_1+R_2} V_{in} \tag{3.1}$$

The pins coming from the module to the microcontroller do not require level shifting since it still perceives the 3.3V as high. This leaves us with the RST, MOSI, SCLK, and SS pins requiring shifting, and each level divider consists of 2 resistors making a total of 8 extra resistors needed. This problem can be avoided in the second configuration.

The second option has the disadvantage of running the microcontroller at a lower speed, but it drastically reduces the number of components required and the complexity of the whole circuit since we do not need to perform logic level shifting and we can supply both components from a common regulated source.

The second option was chosen for this project. Power will be supplied from an outside source and regulated to the decided 3.3V voltage level with the help of a voltage regulator.

The regulator chosen for this project is the LF33CV voltage regulator. The most notable features of this component are its output voltage of 3.3V, the rather low dropout voltage of 0.45V and the output current of up to 500mA, enough to supply both the DWM1000 module and the microcontroller. The dropout voltage implies the use of voltage levels at the input of the regulator that are higher with at least 0.45V from the output level of 3.3V. With this in mind and the capabilities of the regulator, the circuit can be supplied from the outside with voltages in the range of 4V up to 40V.

Another property of the regulator is the need for filtering capacitors. While the output of the regulator is the proper DC voltage required for the whole circuit to work in no load state, the addition of a load introduces a ripple effect to our output. This makes it impossible for our microcontroller and DWM1000 module to properly function being in a state of constant brownouts. To filter out this effect, a pair of capacitors was added in order to filter the input and output of the regulator. A 2.2μF electrolytic capacitor was used for filtering the output voltage of the regulator while a smaller 0.1pF ceramic capacitor was used for filtering the input voltage. This results in a steady DC voltage at the output of the regulator which can be safely used by our circuit.

The outside supply voltage was chosen to be 5V which meets the requirements of our regulator. This level was chosen since it is readily available from USB power supplies which are very common. This is done through an addition of a mini-USB port on the board that connects to an outside supply through an USB cable. The mini-USB has 5 pins that are used for the following purposes:

1. VCC – +5V DC
2. D- - data -
3. D+ - data +
4. ID – usually not connected
5. GND – ground

We currently have no need for the 2 data lines, nor the ID one. The only lines used are the VCC and GND in order to supply power. The VCC line goes to the input of the regulator and the GND line goes to the GND of the regulator.

This concludes the design of the power supply circuit for the board. To summarize the process, the circuit uses a 5V outside power supplied through an USB cable and mini-USB port that connects to a 3.3V voltage regulator in parallel with a 0.1pF capacitor. The output is 3.3V filtered by a 1.1μF capacitor. This output supplies the whole board with current.

The imposed 3.3V supply on all the components has the frequency range requirement of 0-10MHz. The ATmega328P has an 8MHz internal RC oscillating circuit that can be used to generate the clock signal or alternatively, an outside clock source can be used. The internal oscillator, even though it is calibrated properly, it still has an error rate of 10%. This is not acceptable for asynchronous data transfer where timing is very important. For this reason, an external clock source will be used.

A standard value for crystal oscillators in the 0-10MHz range is 8MHz which was chosen for this project. The crystal oscillator also requires two additional capacitors with the same value at its terminals. The recommended value of the capacitors which have to have the same value for 8MHz is 12-22 pF. The 22pF value was chosen.

The connections of those components with the microcontroller is the following:

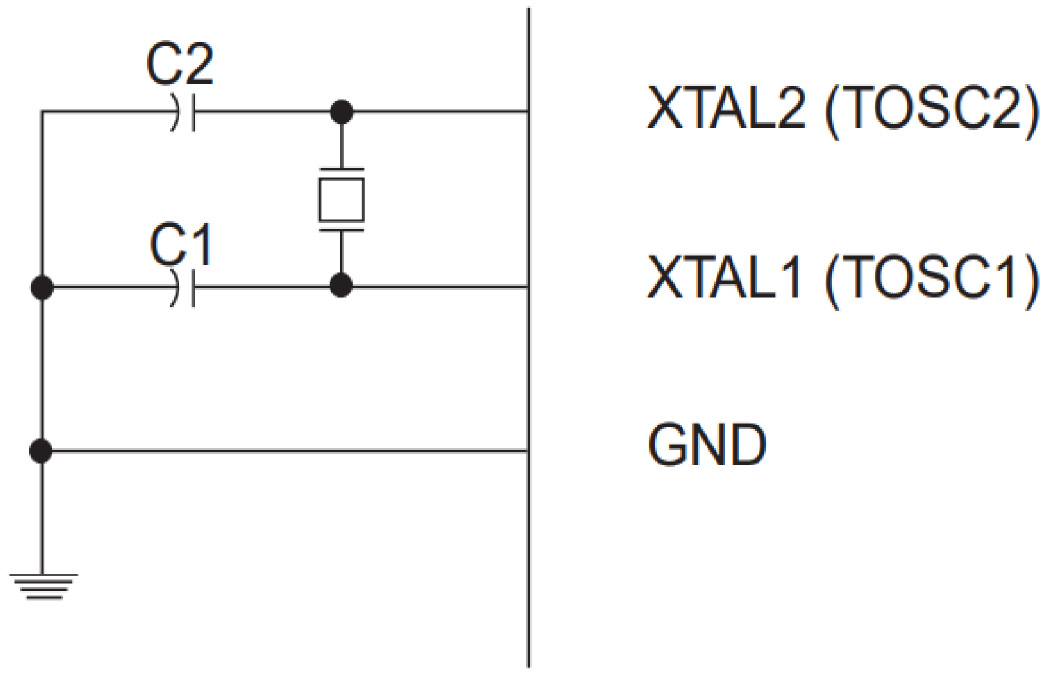


Figure 10

In order to be able to program the microcontroller while it is on the board, we must have access to the pins needed to do so. Those are the SPI pins without the MOSI pin, the reset pin and the power pins. Various jumpers were added in order to increase the safety of the circuit, most notably the VCC jumper used to disconnect the DWM1000 module from the power supply, and also jumpers for the SPI connection to the module. Those are necessary when the microcontroller is being programmed with the ICSP method, we want to isolate those pins from the module.

This concludes the components needed for the board, and to summarize, we have the following:

1. DWM1000 module
2. ATmega328P microcontroller
3. LF33CV voltage regulator
4. 0.1µF filtering capacitor
5. 2.2µF filtering capacitor
6. Mini-USB mother port
7. 8MHz crystal oscillator
8. 2x22pF capacitors
9. Various pins and jumpers

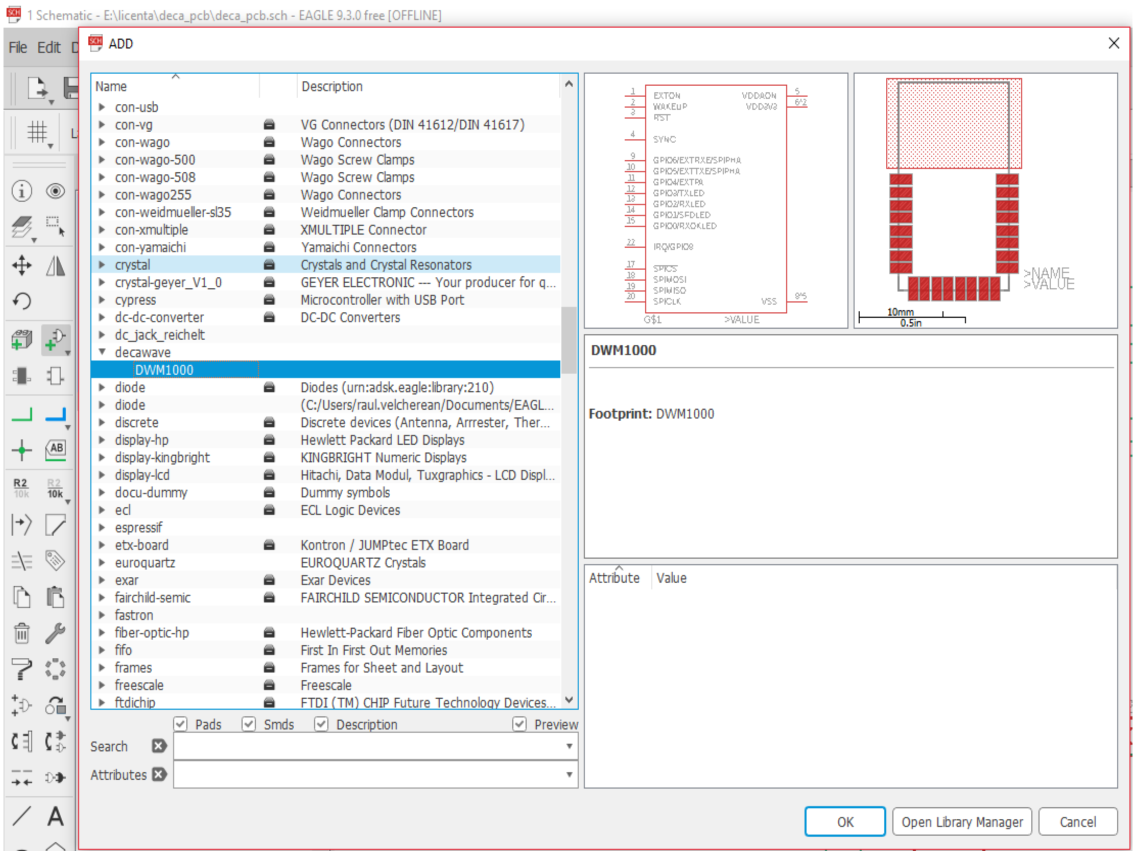
After those particularities have been set, we can continue with the design of the schematic of the board in the EAGLE software. To do this, we must find each of the components in the EAGLE libraries and place them in our schematic. Most basic components are already implemented in the basic software, but for example the

DWM1000 module doesn't have the part implemented. We can design the part on our own with the corresponding pin designations and physical features such as size and electrical footprint, but there already exists a model online and we choose to use that.

In order to add a new library to EAGLE, the following process is used:

1. Download the wanted library from the internet.
2. Copy the library to the path that EAGLE uses for library storage.
3. In the main windows of EAGLE go to Libraries and find the added library, right click it and press Use all.

Once this has been done the components from the library will appear alongside the other components. Now, since we have all the needed components available, we can start to add them to our project. This is done by firstly pressing the *Add Part* button. A new window will open containing all the available components. We navigate to the part that we want to add, select it and add it to our schematic. The window looks like this:


1 Schematic - E:\licenta\deca_pcb\deca_pcb.sch - EAGLE 9.3.0 free [OFFLINE]

ADD

Name	Description
con-usb	
con-vg	VG Connectors (DIN 41612/DIN 41617)
con-wago	Wago Connectors
con-wago-500	Wago Screw Clamps
con-wago-508	Wago Screw Clamps
con-wago255	Wago Connectors
con-weidmueller-sl35	Weidmueller Clamp Connectors
con-xmultiple	XMULTIPLE Connector
con-yamaichi	Yamaichi Connectors
crystal	Crystals and Crystal Resonators
crystal-geyer_V1_0	GEYER ELECTRONIC --- Your producer for q...
cypress	Microcontroller with USB Port
dc-dc-converter	DC-DC Converters
dc_jack_reichelt	
decawave	
DWM1000	
diode	Diodes (urn:adsk.eagle:library:210)
diode	(C:/Users/raul.velcherean/Documents/EAGL...
discrete	Discrete devices (Antenna, Arrrester, Ther...
display-hp	Hewlett Packard LED Displays
display-kingbright	KINGBRIGHT Numeric Displays
display-lcd	Hitachi, Data Modul, Tuxgraphics - LCD Displ...
docu-dummy	Dummy symbols
ecl	ECL Logic Devices
espressif	
etx-board	Kontron / JUMPtec ETX Board
euroquartz	EUROQUARTZ Crystals
exar	Exar Devices
fairchild-semic	FAIRCHILD SEMICONDUCTOR Integrated Cir...
fastron	
fiber-optic-hp	Hewlett-Packard Fiber Optic Components
fifo	First In First Out Memories
frames	Frames for Sheet and Layout
freescale	Freescale
ftdichip	FTDI (TM) CHIP Future Technology Devices...

☑ Pads ☑ Smds ☑ Description ☑ Preview

Search

Attributes

DWM1000

Footprint: DWM1000

Attribute	Value

Figure 3.11

After adding all the parts to the schematic, we must start with the connections between them. They are as such:

- The mother mini-USB port connects to the voltage regulator, the input filtering capacitor and 2 extra pins used for testing.

- The voltage regulator is connected to the output filtering capacitor and both the microcontroller and DWM1000 module. There are 2 additional pins used for testing.
- The 8MHz oscillator is connected to the 2 capacitors and the XTAL pins of the microcontroller.
- The SPI pins between the microcontroller and the DWM1000 module are connected with an interrupted connection using jumpers. There is also the interrupt and reset pins.
- There are additional pins used for testing, communication and programming: 1 for microcontroller reset, 1 for Tx pin, 1 for Rx pin, 1 for power supply of the board that is placed after the voltage regulator. Only 3.3V outside regulated voltage can be placed here.

The schematic implemented in EAGLE is presented below:

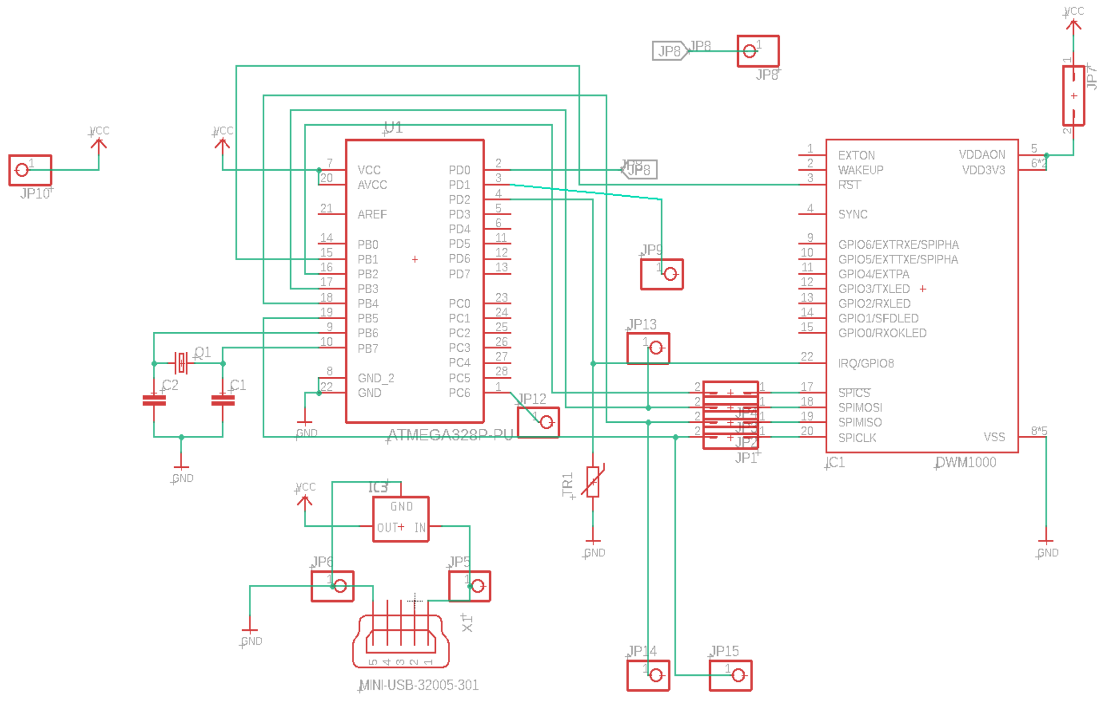


Figure 3.12

Now that the schematic is complete, we can move on to the design of the board itself. In this stage, we decide the physical space that the components will occupy on the board. The most important characteristics that must be considered carefully is the size of the components, the holes in which the components will be placed and soldered and other properties of the components that can interact with the board.

Due to the nature of the project, a board that will be placed on a tracked object or person, we want to make the implementation as small as possible. Old and sickly persons are not to be obstructed or even mildly annoyed with the extra accessory, so it must be as discrete as possible. This means reducing the space between components to the minimum.

The placement of the components is also affected by the production possibilities of the board. Currently, the only possibility in the design of the PCB regarding its copper traces was on only one layer. This limits the paths that are possible to connect the components between them. If placed in a certain way, a connection can become impossible to achieve. This must be avoided and the only way around it would be an implementation on multiple layers, which would help the design greatly and ease the process.

With those factors in mind we can begin the component placement process. When we switch from the schematic window to the board window, all the components that were already added in the schematic are now present in this window as well. They are placed in the bottom left corner of the window, ready to be placed. They are connected between them with yellow lines that represent the connection that was made in the schematic design process. The board itself is also present on this window.

The components are represented in actual size and have their footprint visible. This is helpful when they are placed on the board. The placement is easily done, requiring just a drag and drop of the component by clicking and holding unto it. The components can also be rotated in case it is needed. Whenever we place a component on the board, it can be seen if it is a valid placement, the program throws a warning whenever an invalid placement is made e.g. a footprint is placed outside the board.

One particularity of this PCB, is the presence of the antenna on the DWM1000 module. The antenna is the one that does all the radio transmission and reception of the project. Radio waves are quite sensitive to metal, it can interfere and harm the signal that is being transmitted or received. This means that we cannot have any path going under the area covered by the antenna. This results in 2 implementations:

1. There will be no paths or any metal parts under the antenna or at its sides.
2. The area covered by the antenna will be placed outside of the board.

A visual representation of the 2 possibilities is presented below:

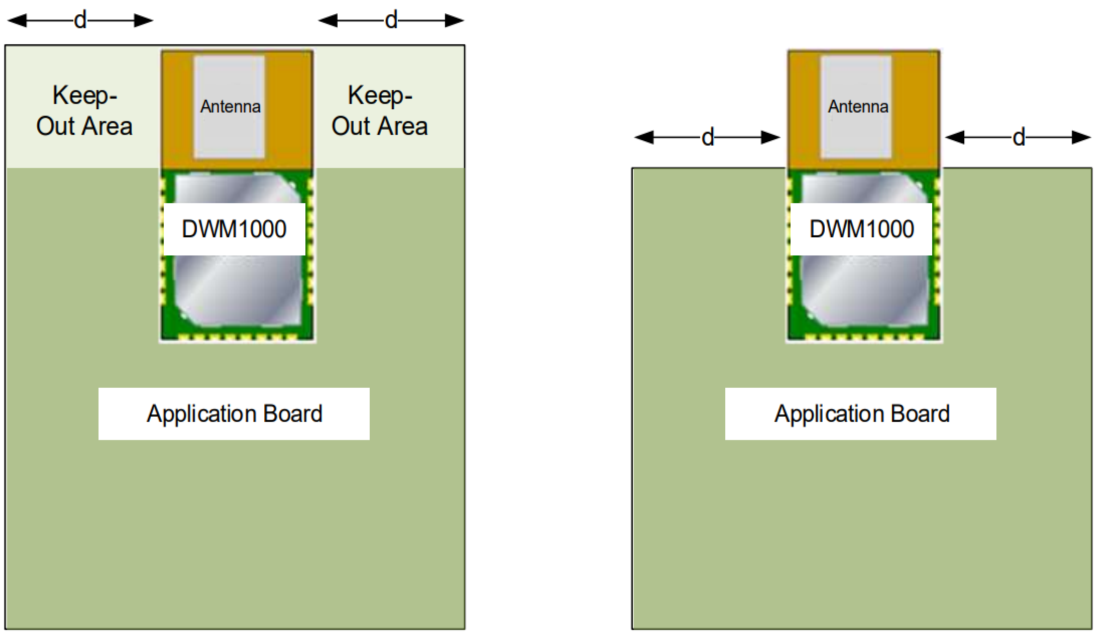


Figure 3.13

After the placement of all the components is finished, the process of routing paths will begin. The advantage of using CAD software is that EAGLE has the capability to route the paths by itself. This is done using the *Autorouter* tool. This tool uses the connections configured in the schematic and finds the best paths the copper traces can take. After the tool has finished the job, we can further tweak the design where it is felt that it is still not enough, or a different approach is desirable.

With all those factors in mind, the following PCB has resulted:

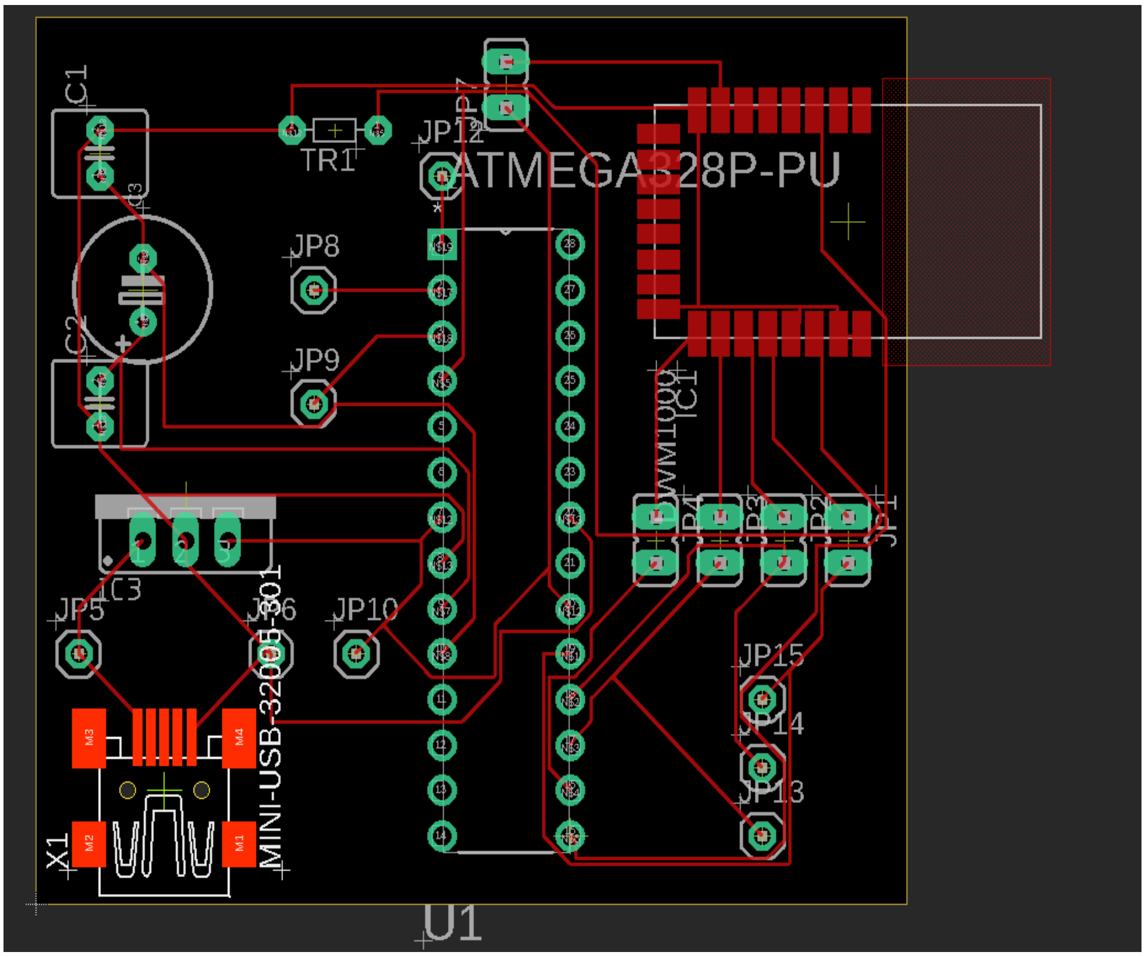


Figure 3.14

As it can be seen, the antenna of the DWM1000 module is placed outside of the board, mitigating any radio interference from the board itself. Four of these boards have been made, 3 to serve as anchors and 1 to serve as a tag.

Now that the boards have been designed and produced, work can be started on the next step of the project. The network made by the four boards will be defined. The network contains each of the 4 boards and a PC. Most of the communication that takes place in the network goes through the board designated as a tag. Each anchor module will communicate with the tag in order to calculate the distance between them and the tag will communicate with the PC in order to transmit the distance data. This is done through serial communication involving an UART to USB converter. The device network is presented below:

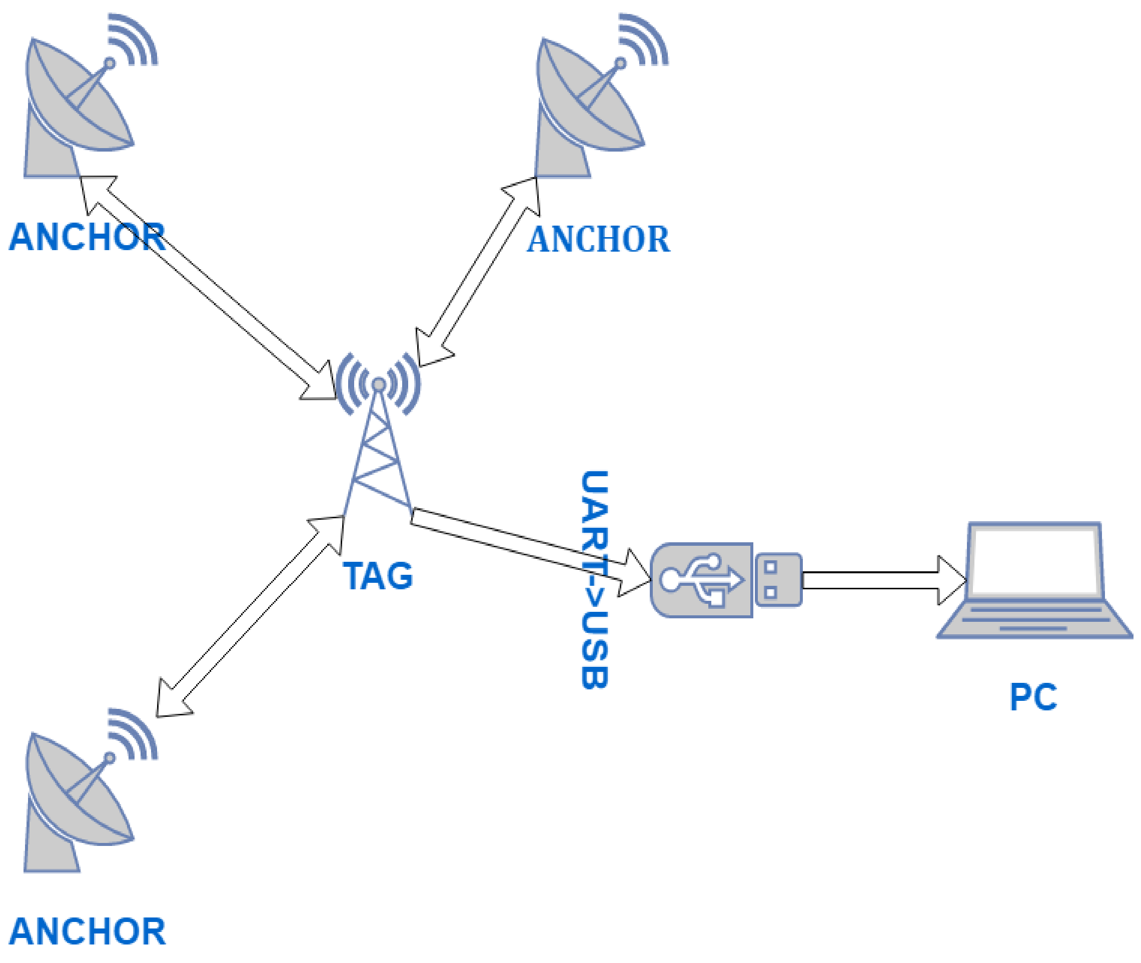


Figure 3.15

In order to obtain actual communication, the devices must be flashed with the appropriate software. There are two versions of software that go on the boards: the anchor specific software and the tag specific software. The 2 versions are very similar, the code being almost identical, but it has subtle differences related to the two-way ranging process.

When flashing the microcontroller, there are so particularities that have to be kept in mind. Microcontrollers have a set of options that are valid throughout the execution of the program. For the ATmega328P microcontroller, those are called the fuse bits. There are a total of 19 fuse bits which make up 3 fuse bytes. Each byte has a name that identifies it, and those are: the *Fuse High Byte,* the *Fuse Low Byte* and the *Extended Fuse Byte* which contains just the remaining 3 bits, the rest of the byte being unused. It is very important to modify those bits with the utmost care. The configuration done here can make the microcontroller unusable by making flashing impossible and also some other hardware related changes. Bits that have the level '1' are considered as unprogrammed, while bits that have the level '0' are considered as being programmed.

The extended fuse byte deals mostly with brownout detection. A brownout is similar to a blackout which is a total supply loss in the system. A brownout represents a drop in the voltage of the power supply of the system. A brownout in a microcontroller can result in a malfunction of the program which is always undesirable. The extended fuse byte can configure the behavior of the microcontroller in case of a brownout, usually it prompts the microcontroller to reset, the level at which it happens being set in the fuse bits.

The high fuse byte deals with more options that are also different in nature to the extended fuse byte. In this byte the boot size can be selected, which is the allocated size of the memory space in case it is desirable to use a bootloader for the microcontroller. There are other miscellaneous options but the most important 2 bits are here. They represent the External Reset Disable and Serial Program and Data Downloading. If programmed incorrectly, the microcontroller will not behave in expected ways.

The low fuse byte deals mostly with clock related settings. Options like clock division and start-up time, which is important for allowing the microcontroller to start when the appropriate voltage levels have been met. The most important options here are the selection of the clock source. The clock source can be provided by 2 different sources: internal and external. The microcontroller has 2 internal clocks: a 128kHz RC Oscillator and a Calibrated 8MHz RC Oscillator, but even though it is calibrated, the error for this clock is too high. There are also more options for the external clock source, specifying the type of oscillator used. Most notable types are: the Low Power Crystal Oscillator, the Full Swing Crystal Oscillator, the Low Frequency Crystal Oscillator.

Since we have chosen to use an external oscillator, the appropriate Option Bytes must be configured accordingly. This needs to be done only once, the preferences persist through all the flashing cycles unless they are rewritten. The process followed in order to program the Option Bytes was a peculiar but very simple one. The option chosen, since our microcontroller is the same one that the Arduino boards use, is to write a bootloader specific to a board configuration that was similar to our implementation. The Arduino Pro Mini works with a supply voltage of 3.3V and at a frequency of 8Mhz. Those are the same settings that we use for our project. Now, to problem of flashing the board must be tackled.

In order to flash the ATmega328P microcontroller a programmer is necessary although there is an alternative, emulating a programmer. The hardware that a programmer uses and the hardware on an Arduino board is very similar. There is the possibility to program the Arduino board with software that allows flashing *through* it. This solution was chosen since it was easily available and simple. The Arduino is flashed with said software and the appropriate connections are made between the Arduino and our microcontroller. The main connections are the power and SPI pins, but it is also important to add the external oscillator to the circuit since that is what we will be using. After the setup is done, the first step that must be is writing the fuse bytes with the method chosen which is through a bootloader. This is done through the Arduino IDE interface. After the connections are properly made, in the Arduino IDE we select the *Tools* tab and then the *Burn Bootloader* option. After this operation is done, we can flash the software we need unto the microcontroller. This is done in a similar manner, since we choose to use the Arduino as programmer flashing option. The code will be written in the Arduino IDE because of the simplicity of it. After the code is written, the *Sketch* tab is selected and the *Upload Using Programmer* option is pressed. This will start the flashing operation that once is over, the microcontroller is ready to be used.

The software written for the purpose of range detection works in the following way. It comes in 2 variants, 1 for the tag and 1 for the anchors. Those 2 variants come together in order to successfully calculate distance. The decisions of the software will be presented in parallel since the boards are dependent on one another.

When the board is first powered, it initializes itself with the values programmed into it, the most notable being its address. After the initial setup, the program starts to run its course.

The process begins with the TAG which broadcasts a POLL signal. A POLL signal has the purpose of identifying other devices on the network, in this case the ANCHORS. The transmission is omnidirectional and should be received by all ANCHORS. After the POLL message has been sent, the TAG now expects a POLL_ACK (Poll Acknowledge) message in order to confirm that the POLL message has been received.

The ANCHOR, after it has started, starts listening for a POLL message coming from the TAG. After the POLL message has been successfully received, the ANCHOR replies with a POLL_ACK message in order to confirm the reception of the POLL message. This message contains the address of the ANCHOR that has replied.

After the TAG has received a POLL_ACK message, it reads the data contained within and adds the address of the device list stored in the TAG. This list is kept in order to determine the distance to with each specific ANCHOR, specified by its address. Now the ANCHOR awaits a RANGE message in order to start the ranging process.

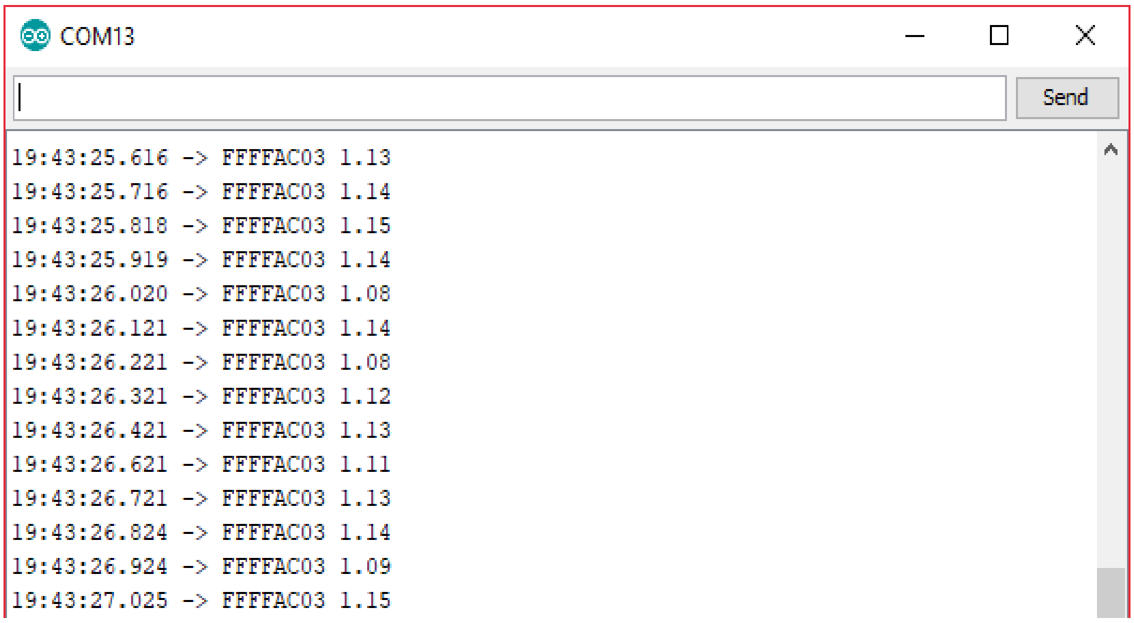
After at least 1 device has been added to the network, the actual ranging process can begin. The TAG now sends a RANGE message containing the address of the TAG, the address of the ANCHOR for which the message is meant and the timestamp. It is now awaiting a RANGE_REPORT message. Now the TAG awaits a RANGE_REPORT message.

The ANCHOR receives the RANGE message and timestamps the moment of reception. The ANCHOR now responds back with a RANGE_REPORT message that contains the timestamp of the transmission, the address of the current ANCHOR and other info of the message that it has previously received, the RANGE message. The ranging should now be successfully done on the part of the ANCHOR and it now awaits a RANGE message yet again.

The TAG receives the final RANGE_REPORT message from the ANCHOR and timestamps the moment of arrival. It has now all the data necessary to calculate the range between it and the ANCHOR. The time of the first transmission, the time of the first transmission arrival at the ANCHOR, the time of the transmission from the reply of the ANCHOR and the time that said reply has arrived at the TAG. It now uses this data to calculate the range.

The ranging process repeats for each ANCHOR stored in the device list of the TAG, but in-between the ranging process, the TAG makes more POLL calls in order to find out if other ANCHOR devices have been added to the network.

The TAG now reports the distance calculated and the address of the ANCHOR it came from. The data looks like this:



```
19:43:25.616 -> FFFFAC03 1.13
19:43:25.716 -> FFFFAC03 1.14
19:43:25.818 -> FFFFAC03 1.15
19:43:25.919 -> FFFFAC03 1.14
19:43:26.020 -> FFFFAC03 1.08
19:43:26.121 -> FFFFAC03 1.14
19:43:26.221 -> FFFFAC03 1.08
19:43:26.321 -> FFFFAC03 1.12
19:43:26.421 -> FFFFAC03 1.13
19:43:26.621 -> FFFFAC03 1.11
19:43:26.721 -> FFFFAC03 1.13
19:43:26.824 -> FFFFAC03 1.14
19:43:26.924 -> FFFFAC03 1.09
19:43:27.025 -> FFFFAC03 1.15
```

Figure 3.16

The report of the distance data is done through the serial interface of the microcontroller which connects to the UART to USB converter and arrives at the PC. The distance data will be further process on the PC in order to obtain actual coordinates of the TAG.

The data is sent as a string containing 2 important parts separated by a space. The first part of the string represents a device address and the second part is distance. The address is of the ANCHOR in relation to which the distance was measured. At this point we have a list containing the addresses and the distances corresponding to those addresses stored on the PC. The list updates as fast as the TAG sends data to the PC. This list can now be used to calculate a set of coordinates for the TAG.

Unfortunately, some errors are present in the distance data, the accuracy varies for each ANCHOR. The errors are still considerable after applying a filter in which the last couple of values of the range are averaged. This makes it impossible to calculate a location through classical trilateration methods. This has caused the splitting of the trilateration algorithm in 2 phases:

1. We use the radii of 2 circles and their centers in order to calculate their intersection. The 2 circles can intersect either in 1 or 2 points. If they intersect in only 1 point, the coordinates of that point represent the coordinates we are searching for. If the circles intersect in 2 points, there are 2 possibilities in which our TAG is.

2. We use the radius and center of the third circle in order to determine which of the previous 2 points is the correct one. The final point chosen between the 2 will be the point that is closer to the circumference of the third circle.

After applying this algorithm, a set of coordinates results which can be used to plot a trace of locations. Note that the locations of the ANCHORS will be determined in this program as well. This part of the project was developed in MATLAB.

MATLAB is a desktop environment that is tuned for analysis and design processes and accompanied by a programming language that can handle most math operations, it has support for matrices and arrays, the capability to simulate systems through Simulink and many other features. It also has useful tools for plotting data that are easy to use, this being one of the main reason this environment was chosen.

In the first part of the MATLAB program, a serial connection is opened on the port that the UART to USB converter is connected. After the connection is open, we can start to read the data that is sent, periodically reading the data buffer. A string is received and then split into 2 parts. The first part of the string is the address of the ANCHOR and the second part of the string, separated by a space, is the distance data. The distance data is then attributed to its corresponding address in a list, bu first it goes into a filter. The filter is a simple mean of the last 5 values of the distance, the result being a distance that does not have very big variations which are usually due to errors. After data for each of the ANCHORS has been obtained, it is passed on to the next part of the program, and the first step of the trilateration algorithm takes place.

The first step in the algorithm is to calculate the distance between the 2 centers of the circles. We will denote this distance with d. The next step is to calculate an auxiliary point that is placed on the line that is perpendicular to the line defined by the 2 centers, and said line also passes through both circle intersection points. The coordinates of this point, P, will be:

$$P = \frac{A1 + a(A2 - A1)}{\mathrm{d}} \tag{3.2}$$

,where A1 is the center of one of the circles, A2 is the center of the other circle and a is the distance from A1 to P.

After we have obtained P, we can calculate the 2 intersection points with the following formula:

$$x = \frac{x_P \pm h(y_{A2} - y_{A1})}{\mathrm{d}} \tag{3.3}$$

$$y = \frac{y_P \mp h(x_{A2} - x_{A1})}{\mathrm{d}} \tag{3.4}$$

The 2 intersection points have been obtained. The next step is to calculate which of those 2 points is closer to the circumference of the third circle. This is done by calculating the shortest distance from each point to the circumference of the circle, which will result in 2 distances. The distance that is shorter corresponds to the point that is closer to the circle. The formula for calculating the distance is the following:

$$D = \left|\sqrt{(x - x_{A3})^2 + (y - y_{A3})^2} - r\right| \tag{3.5}$$

At this point, the coordinates of the TAG in relation to the ANCHORS are known and all is left is to plot them. The following plot has been obtained:

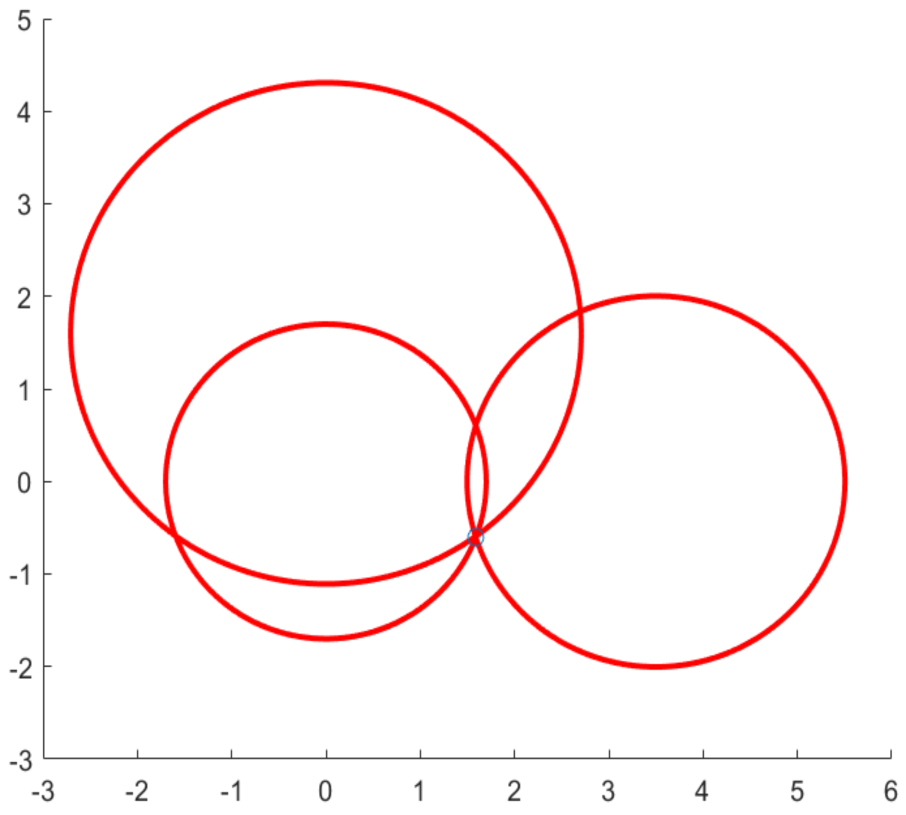


Figure 3.17

This concludes the implementation part of this paper.

3.3 Testing and validation

For the testing purposes of this project, the following conditions were created: the ANCHORs were place inside a room in a right triangle configuration and the TAG was moved inside this room. The configuration was chosen due to the coordinates that result for the ANCHORs, those being (0,0), (X,0) and (Y,0).

The actual number that were used for the ANCHOR coordinates are: (0,0) for the first ANCHOR, A1. (3.5,0) for the second ANCHOR, A2. This means that the second ANCHOR was placed at a distance of 3.5 meters from the first ANCHOR, and the X axis is defined by the straight line that connects these 2 ANCHORs. The coordinates of the third and final ANCHOR are (0,1.6) which place the third ANCHOR at a distance of 1.6 meters from the first ANCHOR and the straight line that connects these 2 points and is perpendicular on the axis defined as X, represents the Y axis of the system.

After placing the ANCHORs in place and starting the program, the TAG was carried on an almost circular motion through the room, the starting point being also the stopping point for this measurement. The trace obtained by the mapping is the following:

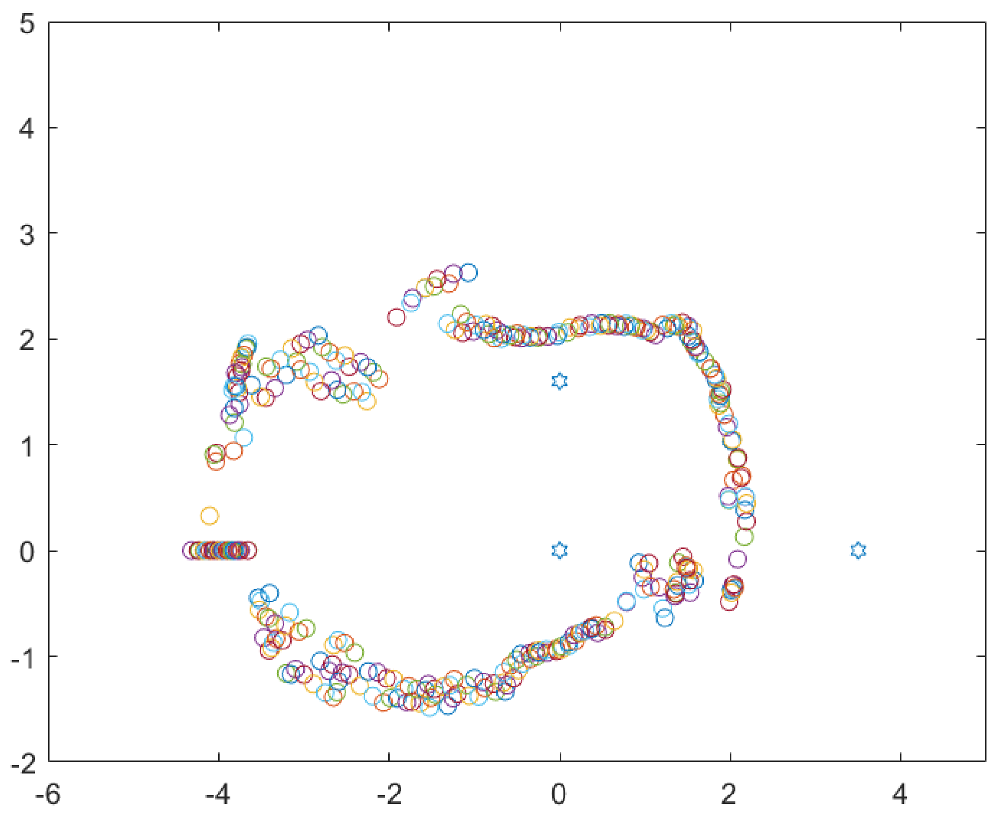


Figure 3.18

The ANCHORs are also represented on this plot. The plotting starts at approximately the point (2,0), between the ANCHORs A1 and A2, and starts going towards the third ANCHOR. As it can be seen, the path is rather smooth until approximately point (-1,2). Until this point, the speed at which the TAG was moved was relatively low. After this point, the speed increases and also, a number of obstacles appear between the TAG and ANCHORs. The trace seems to stabilize, but when it reaches the location around the point (-4,0), the distance measuring is erroneous and the 2 circles used in the first phase of the trilateration algorithm do not meet each other, and the calculated point gives complex results.

After this point, there is a clearer line of sight of the ANCHORs and the signal starts to stabilize once again, providing accurate distance data. The path is rather stable once it reaches the (-1,-1) point.

The actual physical distance measured corresponds quite well with the plotted data. When the path is in the stable region, the error of the location is in the order of centimeters, which is what we have proposed for this project.

4 Conclusions

4.1 Results

The purpose of this project was the accurate localization of an object or person that is located indoors. This is a harder task than the outdoor alternative since walls and other obstacles such as furniture and electronics, pose a threat to the signals used for the classical outdoor localization which is represented by the GPS.

A similar solution that is based on radio waves was used due to its advantages over other technologies such as ultrasonic, inertial and Wi-Fi. The solution in question is using the ultra-wide band in order to reduce the interference of most common appliances which use the narrowband.

For the implementation, the DWM1000 transceiver module from Decawave was used. The module is capable of sending and receiving data at speeds up to 6.8Mbps, and it is able to provide a very accurate timestamp, which is mandatory for the successful ranging process.

The module needs a host microcontroller in order to operate, and for serving this purpose the ATmega328P microcontroller was used. The component was chosen due to familiarity and the simple implementation process.

After the main components were chosen, a PCB was designed in order to connect all the parts of the project together, those being the main 2 components and various other electronic components like capacitors, oscillators and regulators.

Four boards were created, the minimum number for successful localization. The boards were designated as ANCHORs for the stationary parts of the project, those number 3, and a TAG that represents the moving part of the project.

Using the two-way ranging process, the TAG calculates the distance to each of the 3 ANCHORs, and using serial data communication, it sends the distance data to a PC. After the data has been received on the PC, a trilateration algorithm is employed in order to calculate a location based on distance and already known ANCHOR positions.

In the end, locations were plotted on a graph that resulted in a trace of the path taken by the TAG object. The location data was mostly accurate with errors resulting due to the obstacles in the environment and other interferences. Overall, the data resulted is satisfactory.

There was also quite a bit of knowledge that was gained in the making of this process, most notably the design of a PCB which proved very valuable to this project. The microcontroller based knowledge was also increased due to the extended work with them, both physical and software related. Knowledge was also gained in the communication field, due to the SPI protocol used between the 2 main components of this project, and also radio communication involved.

4.2 Further developments

The project can be improved upon by adding a number of features such as:

1. Wireless communication between the TAG and the PC. This way, the TAG transmits data to the PC, without the need to carry a laptop around. Furthermore, the distance data could be collected on a stationary ANCHOR that can be connected to the PC.
2. PCB improvements – the PCB can be reduced in size by using different components and also improving the PCB design.
3. The addition of more TAGs in the network in order to be able to track more objects.
4. The addition of more ANCHORs in the network in order to improve range and accuracy of location. This also means using a different algorithm than trilateration, a multilateration one that can take advantage of the multiple ANCHORs in the system. This also allows the detection of the Z axis.
5. The possibility to map a room, which is done with the help of the TAG that goes around a room and detects obstacles.

5 Bibliography

[1] L. Batistic and M. Tomic, *Overview of Indoor Positioning Sytem Technologies,* Opatija, 2018.

[2] S. de Miguel-Bilbao, J. Roldan, J. Garcia, F. Lopez, P. Garcia-Sagredo, V. Ramos, *Comparative Analysis of Indoor Location Technologies for Monitoring of Elderly,* Madrid, 2013.

[3] J. Karnik, J. Streit, *SUmmary of available indoor location techniques,* Brno.

[4] *Shortest distance between a Point and a Circle*

https://www.varsitytutors.com/hotmath/hotmath_help/topics/shortest-distance-between-a-point-and-a-circle

[5] *What are radio waves?, Thuy May,* 3 Dec. 2018

https://www.nasa.gov/directorates/heo/scan/communications/outreach/funfacts/txt_radio_spectrum.html

[6] *How GPS Receivers Work,* Marshall Brain, Tom Harris

https://electronics.howstuffworks.com/gadgets/travel/gps1.htm

[7] *ATmega328P Datasheet,8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash*

http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

[8] *DWM1000 Datasheet*

https://www.decawave.com/sites/default/files/resources/DWM1000-Datasheet-V1.6.pdf