

ZAVRŠNI RAD
PRVOG CIKLUSA STUDIJA

IMPLEMENTACIJA AUTOMATSKOG SISTEMA
RADIOGONIOMETRIJE

MENTOR

DR. SCI. ALJO MUJČIĆ, RED. PROF.

STUDENT

ADNAN LJESKOVICA

TUZLA, OKTOBAR 2024. GODINE

Sažetak

Razvoj i implementacija sistema za organizaciju i provođenje takmičenja iz radioorientacije predstavljeni su u ovom radu, s ciljem kreiranja efikasnog rješenja koje omogućava automatsko praćenje i bilježenje pronalaska radiopredajnika u realnom vremenu. Osnovna svrha sistema je podrška takmičarima u preciznom lociranju predajnika koristeći XBee komunikacione module u kombinaciji s Arduino platformom. Sistem automatski evidentira vrijeme potrebno za pronađak predajnika i omogućava korisniku da putem grafičkog korisničkog okruženja lako upravlja takmičenjem.

Metodologija rada uključuje konfiguraciju ZigBee mreže koristeći XBee module, implementaciju serijske komunikacije između računara i Arduino uređaja, te razvoj aplikacije za upravljanje takmičenjem. Razvijena aplikacija korisnicima nudi vizuelni prikaz rezultata. U radu su korišteni alati kao što su XCTU softver za konfiguraciju XBee modula i Visual Studio za razvoj aplikacije.

Glavni rezultati istraživanja pokazuju da je primjena XBee tehnologije u ovakvim sistemima vrlo efikasna, omogućavajući stabilnu i preciznu komunikaciju među uređajima. Sistem je dizajniran da bude jednostavan za korištenje, pouzdan i lako prilagodljiv za buduća proširenja, što ga čini pogodnim za organizaciju takmičenja iz radioorientacije.

Zaključak istraživanja je da ovakav sistem poboljšava regularnost i efikasnost takmičenja. Postavljene su i osnove za budući razvoj, uključujući pojednostavljenje upotrebe uređaja i proširenje na veće takmičarske mreže.

Ključne riječi: XBee, ZigBee mreža, Arduino, radioorientacija.

Abstract

The development and implementation of a system for organizing and conducting radio direction finding (ARDF) competitions are presented in this paper, with the aim of creating an efficient solution that enables automatic tracking and logging of the discovery of radio transmitters in real time. The main purpose of the system is to assist competitors in accurately locating transmitters using XBee communication modules in combination with the Arduino platform. The system automatically records the time required to find the transmitters and allows the user to easily manage the competition through a graphical user interface.

The methodology of the work includes the configuration of a ZigBee network using XBee modules, the implementation of serial communication between the computer and Arduino devices, and the development of an application to manage the competition. The developed application provides users with a visual display of the results. Tools such as the XCTU software for configuring XBee modules and Visual Studio for application development were used in the process.

The main research findings demonstrate that the application of XBee technology in such systems is highly efficient, providing stable and accurate communication between devices. The system is designed to be user-friendly, reliable, and easily adaptable for future expansions, making it suitable for organizing ARDF competitions.

The conclusion of the research is that this system improves the regularity and efficiency of competitions. Foundations have also been laid for future development, including simplifying device usage and expanding to larger competitive networks.

Keywords: *XBee, ZigBee network, Arduino, radio orienteering.*

Sadržaj

Sažetak	ii
Abstract	iii
Sadržaj	iv
Popis slika	v
Popis tablica	vii
Popis skraćenica	viii
1 Uvod	1
2 Komponente i alati za implementaciju sistema	3
2.1 Hardverske komponente	3
2.1.1 Arduino UNO razvojna ploča.....	4
2.1.2 XBee komunikacijski modul	13
2.1.3 Radiogoniometar	28
2.1.4 Radiopredajnik	32
2.1.5 Pasivni infracrveni senzor pokreta	34
2.1.6 Zvučni signalizator	39
2.2 Softverski alati.....	41
2.2.1 XCTU	41
2.2.2 Arduino IDE	44
2.2.3 Visual Studio 2022	48
3 Sistem za takmičenje iz radioorientacije	50
4 Implementacija sistema	53
Zaključak	74
Bibliografija	75
Dodatak	77

Popis slika

Slika 1. 1 Određivanje azimuta radio predajnika	1
Slika 2. 1 Arduino i mikrokontroler [1].....	4
Slika 2. 2 Arduino Uno R3 [3]	6
Slika 2. 3 Komponente Arduino Uno ploče [4].....	8
Slika 2. 4 ATmega328P raspodjela nožica [4]	9
Slika 2. 5 ATmega328P raspodjela portova [4]	9
Slika 2. 6 Struktura memorije [4]	10
Slika 2. 7 Raspored pinova [1]	13
Slika 2. 8 Evolucija XBee modula [7]	14
Slika 2. 9 XBee moduli u standardnoj i pro verziji [8]	15
Slika 2. 10 Tipovi antena [8]	17
Slika 2. 11 Fizičko numerisanje pinova a) prednji prikaz b) poleđinski prikaz [8]	18
Slika 2. 12 ZigBee protokol slojevita arhitektura.....	20
Slika 2. 13 Frekvencijski opseg ZigBee kanala.....	20
Slika 2. 14 Topologija ZigBee mreža [13]	22
Slika 2. 15 Vennov dijagram koji prikazuje kanal, PAN i adresiranje [8]	23
Slika 2. 16 PAN skeniranje.....	24
Slika 2. 17 Struktura nadzornog okvira [14]	26
Slika 2. 18 Data-flow dijagram UART interfejs [15]	26
Slika 2. 19 Unutrašnji tok podataka [15].....	27
Slika 2. 20 Feritni štapić sa namotajima [16]	28
Slika 2. 21 Radiogoniometar [17].....	29
Slika 2. 22 Dijagram feritne antene [18]	29
Slika 2. 23 Dijagram (vertikalne) štap antene [18].....	30
Slika 2. 24 Kardioida [18]	30
Slika 2. 25 Šematski dijagram radiogoniometra [19]	31
Slika 2. 26 Radiopredajnik	32
Slika 2. 27 Šematski dijagram radiopredajnika [20]	33
Slika 2. 28 Pasivni infracrveni senzor HC-SR501 [21].....	34
Slika 2. 29 a) Piroelektrični senzor b) Fresnelova leća [22].....	35
Slika 2. 30 Diferencijalno detektovanje infracrvenih promjena na PIR senzoru [22].....	35
Slika 2. 31 a) IR pokrivenost b) Prelamanje IR svjetlosti kroz Fresnel leću [22]	36
Slika 2. 32 Raspored komponenti na ploči HC-SR501 [22].....	37
Slika 2. 33 Jednokratni režim okidanja [22].....	38
Slika 2. 34 Višestruki režim okidanja [22].....	38
Slika 2. 35 Periodi zaključavanja detekcije na HC-SR501 [22].....	38
Slika 2. 36 Konfiguracija pinova zujalice [25].....	39
Slika 2. 37 Princip rada piezoelektričnog zvučnog signalizatora [26]	40
Slika 2. 38 XCTU [27]	42
Slika 2. 39 Meni traka [27].....	42
Slika 2. 40 Glavna alatna traka [27]	42

Slika 2. 41 a) Prva sekcija b) Druga sekcija c) Treća sekcija [27]	43
Slika 2. 42 Lista uređaja [27]	43
Slika 2. 43 Radno područje [27].....	43
Slika 2. 44 a) Konfiguracijski režim b) Konzolski režim c) Mrežni režim [27]	44
Slika 2. 45 Arduino IDE 2 [28]	45
Slika 2. 46 Arduino knjiga skica [28].....	46
Slika 2. 47 Arduino upravljač pločama [28]	46
Slika 2. 48 Arduino upravljač bibliotekama [28]	47
Slika 2. 49 Arduino serijski monitor [28].....	47
Slika 4. 1 a) USB dongle b) Pravilno postavljanje USB dongle-a [30].....	54
Slika 4. 2 Dodavanje XBee modula	54
Slika 4. 3 Konfiguracija XBee modula.....	55
Slika 4. 4 Ažuriranje modula	56
Slika 4. 5 Konfiguracija koordinatora	56
Slika 4. 6 Konfiguracija rutera	57
Slika 4. 7 Slanje poruke od koordinatora prema ruteru	58
Slika 4. 8 Slanje poruke od rutera prema koordinatoru	58
Slika 4. 9 a) XBee ekspanzijska ploča b) DLINE/UART prekidač [30]	59
Slika 4. 10 Povezivanje XBee modula sa Arduinom [30]	60
Slika 4. 11 Deklaracija pinova i inicijalizacija varijabli.....	60
Slika 4. 12 Setup() funkcija	61
Slika 4. 13 Loop() funkcija.....	62
Slika 4. 14 Uvod u kod	64
Slika 4. 15 Metod “button1_Click”	66
Slika 4. 16 While petlja	66
Slika 4. 17 Try blok unutar while petlje	68
Slika 4. 18 Try blok nastavak	69
Slika 4. 19 Catch blok	70
Slika 4. 20 Grafički korisnički interfejs takmičenja	71
Slika 4. 21 Prikaz sva tri Arduino uređaja u sistemu.....	72
Slika 4. 22 Simulacija takmičenja (predajnik B pronađen)	73
Slika 4. 23 Završena simulacija takmičenja	73

Popis tablica

Tabela 2. 1 Arduino Uno specifikacije	7
Tabela 2. 2 Karakteristike XBee modula.....	16
Tabela 2. 3 XBee opis pinova.....	18

Popis skraćenica

ARDF	Amateur Radio Direction Finding
ARG	Amateur Radio Goniometry
ARO	Amateur Radio Orienteering
PIR	Passive Infrared
IDE	Integrated Development Environment
RAM	Random Access Memory
GPIO	General Purpose Input/Output
PWM	Pulse Width Modulation
USB	Universal Serial Bus
ICSP	In-Circuit Serial Programming
FTDI	Future Technology Devices International
SDA	Serial Data
SCL	Serial Clock
AREF	Analog Reference
IOREF	Input/Output Reference
CPU	Central Processing Unit
AC	Alternating Current
DC	Direct Current
SD	Secure Digital
AVR	Alfred V. (Alf) R.
RISC	Reduced Instruction Set Computing
ADC	Analog-to-Digital Converter
SRAM	Static Random Access Memory
EEPROM	Electrically Erasable Programmable Read-Only Memory
I2C	Inter-Integrated Circuit
DAC	Digital-to-Analog Converter
USART	Universal Synchronous/Asynchronous Receiver/Transmitter

SPI	Serial Peripheral Interface
SS	Slave Select
MOSI	Master Out Slave In
MISO	Master In Slave Out
SCK	Serial Clock
TWI	Two Wire Interface
IIoT	Industrial Internet of Things
IEEE	Institute of Electrical and Electronics Engineers
P2P	Point to Point
IoT	Internet of Things
RF	Radio Frequency
PCB	Printed Circuit Board
RPSMA	Reverse Polarity SubMiniature version A
WPAN	Wireless Personal Area Network
AES	Advanced Encryption Standard
PHY	Physical
MAC	Media Access Control
NWK	Network
APS	Application
OSI	Open Systems Interconnection
OQPSK	Offset Quadrature Phase Shift Keying
BPSK	Binary Phase Shift Keying
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance
FFD	Full Function Device
RFD	Reduced Function Device
ZDO	ZigBee Device Object
ZC	ZigBee Coordinator
ZR	ZigBee Router
ZED	ZigBee End Device
APSDE	Application Support Data Entity

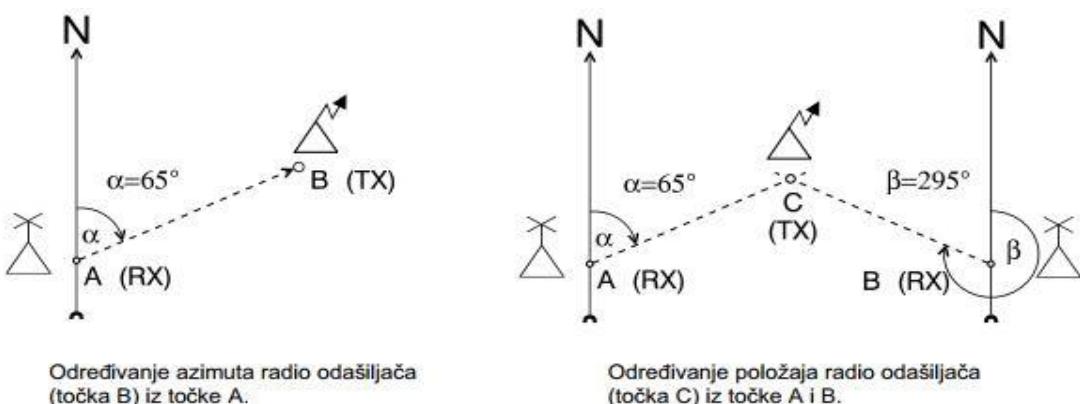
APSME	Application Support Management Entity
PAN	Personal Area Network
PAN ID	Personal Area Network Identifier
CAP	Contention Access Period
GTS	Guaranteed Time Slot
UART	Universal Asynchronous Receiver-Transmitter
CTS	Clear to Send
RTS	Request to Send
API	Application Programming Interface
PA	Power Amplifier
GUI	Graphical User Interface

1 Uvod

U ovom diplomskom radu obrađena je tema razvoja sistema za organizaciju i provođenje takmičenja iz radioorientacije koristeći moderne tehnologije, kao što su Arduino mikrokontroleri i ZigBee moduli za bežičnu komunikaciju. Fokus rada je na kreiranju automatizovanog sistema koji vjerno simulira stvarno takmičenje iz "Amaterske radiogoniometrije" – ARDF (engl. *Amateur Radio Direction Finding*). Cilj ovog rada bio je osmisliti i testirati sistem koji unaprjeđuje regularnost takmičenja uklanjanjem uticaja ljudskog faktora, automatizirajući ključne procese kao što su upravljanje predajnicima i mjerjenje vremena. Time se postiže veća preciznost i objektivnost u procjeni vještina takmičara, čineći takmičenje regularnijim i efikasnijim.

Riječ "goniometrija" potiče od grčkih termina "gonio" (ugao) i "metrein" (mjeriti), što ukazuje na mjerjenje uglova. Stoga, radiogoniometrija predstavlja disciplinu koja se fokusira na određivanje položaja izvora radiotalasa, odnosno predajnika, na karti putem mjerjenja azimuta. Ova mjerena se obavljaju s najmanje dva, ali su preciznija kada se koriste tri različite tačke. Naime, iz pozicije u kojoj se nalazi radiogoniometar (prijemnik - RX) povlače se pravci prema predajniku (TX), a presjek tih pravaca označava lokaciju traženog predajnika. (Slika 1. 1)

Radiogoniometrija igra ključnu ulogu u mnogim oblastima telekomunikacija. U osnovi, radiogoniometrija omogućava korisnicima da odrede pravac iz kojeg dolazi radio signal, čime se olakšava lociranje predajnika ili izvora signala. Ova tehnika se koristi u različitim sferama, kao što su praćenje brodova i aviona, potraga za izvorima ometanja u radio komunikacijama, te u potragama i spašavanjima, gdje je precizna detekcija signala od presudnog značaja.



Slika 1. 1 Određivanje azimuta radio predajnika

Pored svoje profesionalne primjene, radiogoniometrija se našla i u sferi amaterskih aktivnosti kroz takmičenja iz radiogoniometrije, poznata kao ARDF ili popularno nazvana "lov na lisice". ARDF je vrsta sportsko-tehničkog takmičenja u kojem radioamateri koriste radiogoniometre za pronalaženje skrivenih predajnika u određenom vremenskom okviru. Takmičenje ima cilj da unaprijedi sposobnosti učesnika u brzom i preciznom lociranju radio signala, ali i da popularizuje tehniku radiogoniometrije među mladim radioamatерima. Ova takmičenja se organizuju širom svijeta i zahtijevaju kombinaciju tehničkog znanja, vještine slušanja i detekcije signala, te snalažljivosti u terenskim uslovima.

U ovom radu je razvijen automatizovani sistem za simulaciju takmičenja iz radioorientacije, s ciljem poboljšanja regularnosti takmičenja i eliminacije ljudskog faktora. Tradicionalna takmičenja, koja zahtijevaju sudije i ručno uključivanje predajnika, zamjenjena su modernim pristupom u kojem Arduino mikrokontroleri, u kombinaciji sa ZigBee modulima, preuzimaju kontrolu nad predajnicima, senzorskim uređajima i komunikacijom. Sistem omogućava nasumično odabiranje predajnika, emitovanje signala na frekvenciji od 3.5 MHz, i automatsko registrovanje trenutka kada takmičar pronađe predajnik koristeći radiogoniometar.

U svrhu ovog takmičenja, sistem se sastoji od tri predajnika koja se nalaze na unaprijed određenim pozicijama. Takmičar, sa povezom na očima, oslanja se isključivo na svoj radiogoniometar kako bi locirao predajnike. Sistem koristi PIR senzore za detekciju prisustva takmičara, dok buzzer služi kao povratna zvučna informacija nakon pronalaska predajnika. Komunikacija između uređaja se ostvaruje putem ZigBee mreže, čime se osigurava pouzdana bežična veza između uređaja.

Ovaj rad se oslanja na tehnologije opisane u narednom poglavlju, uključujući Arduino platformu i njene različite senzore i module, kao i ZigBee tehnologiju koja omogućava formiranje mreža s više čvorova. Ove komponente su ključne za implementaciju bežičnih sistema u ovakvim aplikacijama. Sistem je razvijen uz primjenu softverskih alata poput Arduino IDE za programiranje mikrokontrolera i XCTU za konfiguraciju ZigBee modula. C# je korišten za izradu korisničkog interfejsa, kroz koji se upravlja takmičenjem.

Rad je podijeljen u nekoliko ključnih poglavlja. Nakon uvodnog dijela, slijedi poglavlje koje opisuje komponente korištene za implementaciju sistema, kako hardverske tako i softverske. Treće poglavlje detaljno objašnjava rad sistema za takmičenje iz radioorientacije, uključujući pravila i principe takmičenja. U četvrtom poglavlju, opisuje se konkretna implementacija sistema, od postavljanja opreme do programiranja Arduino mikrokontrolera i konfiguracije ZigBee mreže. Zaključak rada daje kratak pregled ključnih dostignuća i razmatra moguća unaprjeđenja sistema.

2 Komponente i alati za implementaciju sistema

Ovo poglavlje fokusira se na detaljan prikaz ključnih hardverskih komponenti i softverskih alata korištenih u implementaciji koji zajedno čine osnovu sistema za automatsku radioorientaciju. Poseban akcenat je stavljen na tehnički opis, karakteristike ovih komponenti i funkcionalne mogućnosti svakog elementa, s ciljem pružanja sveobuhvatnog pregleda kako hardverskih, tako i softverskih resursa korištenih u izradi sistema.

Dok će tehnički aspekti i individualne karakteristike hardverskih komponenti i softverskih alata biti temeljno razmotreni u ovom poglavlju, detaljna analiza njihove uloge i međusobne interakcije biće obrađene u poglavlju rada „Implementacija sistema“. Ova struktura omogućava čitaocu prvo razumijevanje osnovnih gradivnih elemenata sistema, uključujući i hardverske komponente i softverske alate, što će poslužiti kao osnova za dublje shvatanje njihove sinergije i funkcionalnosti u kasnijim fazama implementacije.

2.1 Hardverske komponente

U okviru sistema za automatsku radioorientaciju, hardverske komponente predstavljaju ključne gradivne elemente koji omogućavaju njegovu funkcionalnost. Ovaj dio rada fokusira se na tehničke karakteristike i uloge glavnih hardverskih elemenata, kao što su:

- Arduino UNO ploča
- XBee komunikacijski modul
- Radiogoniometar
- Radiopredajnik
- Pasivni infracrveni senzori pokreta – PIR (engl. *Passive Infrared Motion Sensors*)
- Zvučni signalizator (engl. *Buzzer*)

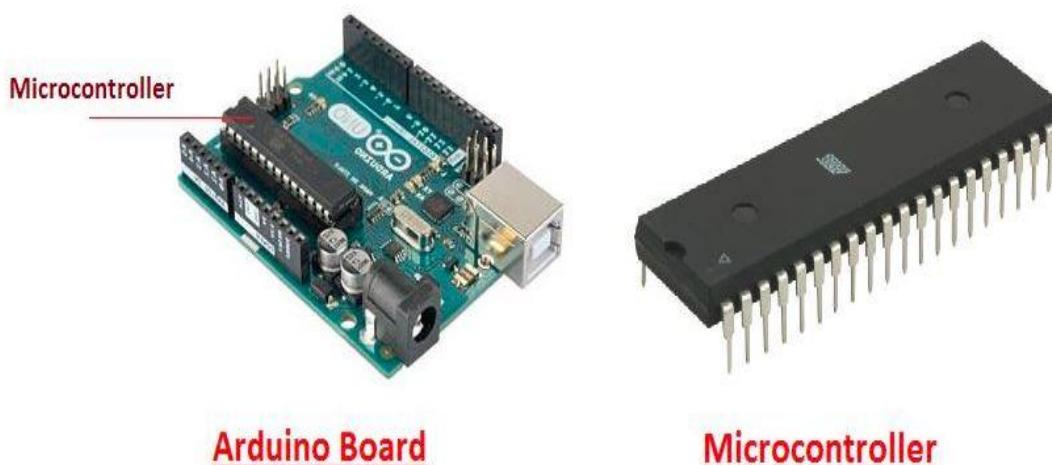
Svaka od ovih komponenti igra specifičnu ulogu u obezbjeđivanju pouzdanog i efikasnog rada sistema, a njihovo detaljno razmatranje omogućit će dublje razumijevanje tehnoloških osnova na kojima se sistem zasniva.

2.1.1 Arduino UNO razvojna ploča

Mikrokontroleri su postali neizostavni dio svakodnevnog života, upravljujući brojnim elektronskim uređajima i aparatima koje koristimo svakodnevno, od mobilnih telefona do raznih kućnih aparata i drugih elektronskih uređaja. Jedna od najpoznatijih razvojnih ploča koja koristi mikrokontroler je Arduino. Revolucionirao je industriju elektronike, omogućavajući ljudima svih uzrasta i nivoa znanja da kreiraju sopstvene projekte i automatizuju zadatke. Arduino je dizajniran sa ciljem da pojednostavi proces kreiranja elektronskih uređaja, pružajući jednostavan način za interakciju s fizičkim svijetom kroz kodiranje i upotrebu senzora, motora i drugih komponenti. Kroz svoj open-source pristup, Arduino je postao ključni alat u obrazovanju, hobijima i inovacijama, olakšavajući sve zadatke koji su povezani sa automatizacijom i kontrolom uređaja.

Kao što je navedeno u internet članku „What is Arduino Uno“ neki ljudi obično ne prepoznaju razliku između mikrokontrolera i Arduina. Mikrokontroler je samo čip sa 40 pinova koji dolazi sa ugrađenim mikroprocesorom, dok je Arduino ploča koja sadrži mikrokontroler na osnovi ploče, kao što je prikazano na slici 2. 1. [1]

Arduino ploče mogu izvoditi neke funkcije koje je sposoban izvesti jedan mikrokontroler. Međutim, hobisti i stručnjaci i dalje preferiraju Arduino ploče u odnosu na mikrokontrolere, zato što su Arduino ploče jednostavne za korištenje i ne zahtijeva se veliko znanje za rad s njima. Prilikom korištenja Arduino ploča, ne morate koristiti dodatne periferne uređaje i komponente za rad ploče. Arduino je kompletanica ploča koja dolazi sa GPIO pinovima, analognim pinovima i mikrokontrolerom kao srcem ploče. Mikrokontroler, s druge strane, je čip u kojem su svi neophodni dijelovi, poput mikroprocesora, RAM-a i flash memorije, integrirani u jedan čip. Možemo reći da je svaka Arduino ploča mikrokontroler, ali nije svaki mikrokontroler Arduino ploča. [2]



Slika 2. 1 Arduino i mikrokontroler [1]

Dakle, dok mikrokontroler sam po sebi može izvršavati različite funkcije, Arduino ploča pojednostavljuje i proširuje njegove mogućnosti dodavanjem dodatnih funkcionalnosti. U osnovi, Arduino ploča pruža sve u jednom paketu, dok mikrokontroler sam zahtijeva više rada i dodatnih komponenti za postizanje sličnih funkcionalnosti.

Arduino Uno je open-source mikrokontrolerska ploča zasnovana na mikrokontroleru „Microchip ATmega328P“. Ploča je opremljena nizom digitalnih i analognih ulazno/izlaznih pinova (engl. *I/O - Input/Output*), koji se mogu povezati sa raznim pločama za proširenje (šildovima) (engl. *shields*) i drugim elektronskim kolima. Arduino Uno ima 14 digitalnih ulazno/izlaznih pinova, šest njih može raditi kao izlazi sa modulisanom širinom impulsa – PWM (engl. *Pulse Width Modulation*), šest analognih ulaznih/izlaznih pinova (I/O), keramički rezonator od 16 MHz, univerzalnu serijsku magistralu kao priključak – USB (engl. *Universal Serial Bus*), naponski priključak, priključak za serijsko programiranje u krugu – ICSP (engl. *In-Circuit Serial Programming*) i dugme za restartovanje. [3]

Ploča sadrži sve što je potrebno za podršku mikrokontrolera i omogućava jednostavno programiranje i integraciju sa drugim uređajima. Uno se razlikuje od svih prethodnih ploča po tome što ne koristi FTDI (engl. *Future Technology Devices International*) čip za konverziju sa univerzalne serijske magistrale na serijski (RS232) protokol. Umjesto toga, koristi mikrokontroler Atmega16U2 (ili Atmega8U2 do verzije R2), koji je programiran kao konverter između univerzalne serijske magistrale i serijskog protokola. Često se uz naziv Uno modela može vidjeti oznaka R3 koja predstavlja treću reviziju pločice uređaja. Oznake R1 i R2 je danas vrlo teško negdje susresti pošto su te varijante davno prestale da se proizvode. [3]

Revizija 3 pločice sadrži sljedeće nove karakteristike [3]:

- Raspored pinova: Dodani su pinovi za serijske podatke – SDA (engl. *Serial Data*) i serijski sat – SCL (engl. *Serial Clock*) blizu pina za analognu referencu – AREF (engl. *Analog Reference*), kao i dva nova pina smještena blizu RESET pina. Pin ulazno/izlazne reference – IOREF (engl. *Input/Output Reference*) omogućava pločama za proširenje (šildovima) da se prilagode naponu koji pruža ploča. U budućnosti će ploče za proširenje biti kompatibilne kako s pločama koje koriste AVR mikrokontrolere (koji rade sa 5V), tako i s Arduino Due pločom koja radi sa 3.3V. Drugi pin je nepovezan i rezervisan je za buduće svrhe.
- Jače RESET kolo.
- Atmega16U2 je zamjenio Atmega8U2.

Arduino je razvijen od strane „Arduino.cc“ kompanije i objavljen je 2010. godine. Riječ "uno" znači "jedan" na italijanskom i odabrana je da označi početno izdanje Arduino softvera. Uno ploča je prva u seriji Arduino ploča baziranih na USB-u. Slika 2.2 prikazuje Arduino Uno R3 koja je treća i najnovija revizija Arduino Uno ploče. [3]



Slika 2. 2 Arduino Uno R3 [3]

Ova ploča dolazi sa svim funkcijama potrebnim za rad mikrokontrolera i može se direktno povezati s računarom putem USB kabla, koji se koristi za prenos koda na mikrokontroler koristeći softver za integrисано razvojno okruženje – IDE (engl. *Integrated Development Environment*), prvenstveno razvijen za programiranje Arduina. Funkcije Arduino ploče [1]:

- Viša frekvencija i broj instrukcija po ciklusu: Na ploči je postavljen Atmega328 mikrokontroler koji sadrži niz funkcija kao što su tajmeri, brojači, prekidi (engl. *interrupts*), PWM, centralna procesorska jedinica – CPU (engl. *Central Processing Unit*), I/O pinovi, i koji radi na 16 MHz taktu, što omogućava veću frekvenciju i veći broj instrukcija po ciklusu.
- Ugrađena regulacija: Ova ploča dolazi sa ugrađenom funkcijom regulacije koja održava napon pod kontrolom kada je uređaj povezan na vanjski uređaj.
- Fleksibilnost i jednostavnost upotrebe: Na ploči se nalazi 14 digitalnih I/O pinova i 6 analognih pinova koji omogućavaju spajanje s bilo kojim vanjskim krugom. Ovi pinovi pružaju fleksibilnost i jednostavnost upotrebe za vanjske uređaje koji se mogu povezati putem ovih pinova.
- Konfigurisani pinovi: 6 analognih pinova označeno je kao „A0“ do „A5“ i imaju rezoluciju od 10 bita. Ovi pinovi mijere napon od 0 do 5V, međutim, mogu se konfigurisati za viši raspon koristeći funkciju „analogReference()“ i „AREF“ pin.
- Brzi start: Na ploči je dostupan pin za restartovanje koji restartuje cijelu ploču i vraća pokrenuti program na početni nivo. Ovaj pin je koristan kada se ploča zaglavi tokom rada programa (pritiskom na ovaj pin sve će se obrisati u programu i program će se ponovo pokrenuti od početka).

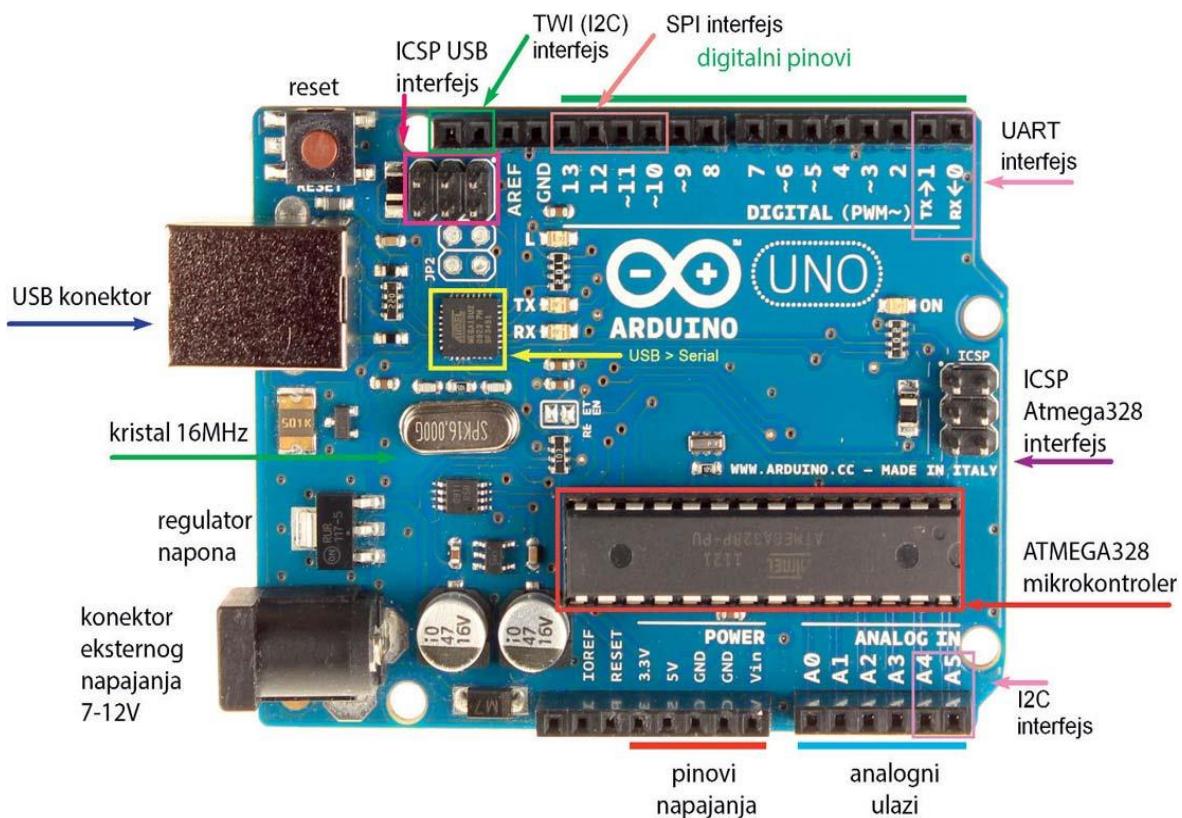
- Veća Flash memorija: 13 KB flash memorije se koristi za pohranjivanje broja instrukcija u obliku koda.
- Niska potreba za naponom: Za rad ploče potrebno je samo 5 V, što se može postići direktno putem USB porta ili vanjskog adaptera, međutim, ploča može podržati vanjski izvor napajanja do 12 V, koji se može regulisati i ograničiti na 5 V ili 3.3 V, zavisno o potrebama projekta.
- Plug & Play: Nema složenog interfejsa potreban za povezivanje uređaja s pločom. Jednostavno priključite vanjski uređaj u pinove ploče koji su raspoređeni na ploči u obliku zaglavlja.
- USB interfejs: Arduino Uno dolazi sa USB interfejsom, tj. USB port je dodan na ploču kako bi se razvila serijska komunikacija s računarom.
- Alternativni izvori napajanja: Osim USB-a, ploču se može napajati baterijom ili AC-DC adapterom.
- Više memorije: Postoji mogućnost korištenja Micro SD kartice na ploči za pohranu veće količine informacija.

Specifikacije Arduino Uno ploče su date u tabeli ispod.

Mikrokontroler	ATmega328P – mikrokontroler iz porodice AVR sa 8-bitnom arhitekturom
Radni napon	5 V
Preporučeni ulazni napon	7-12 V
Granice ulaznog napona	6-20 V
Analogni ulazni pinovi	6 (A0-A5)
Digitalni I/O pinovi	14 (od kojih 6 omogućava PWM izlaz)
DC struja na I/O pinovima	40 mA
DC struja na 3.3 V pinu	50 mA
Flash memorija	32 KB (0.5 KB se koristi za Bootloader)
SRAM	2 kB
EEPROM	1 kB
Frekvencija	16 MHz

Tabela 2. 1 Arduino Uno specifikacije

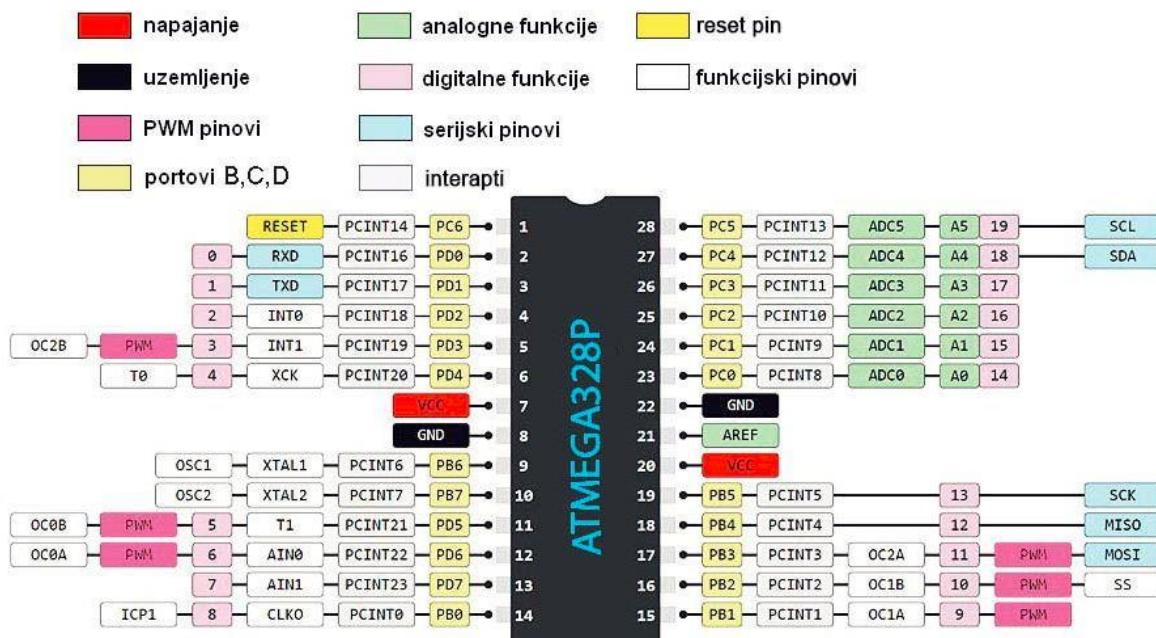
Arduino Uno ploča je osmišljena tako da omogući jednostavno programiranje i povezivanje sa raznim elektronskim uređajima. U srcu njene funkcionalnosti nalaze se komponente koje su pažljivo odabrane kako bi osigurale pouzdanost i svestranost u primjeni. Komponente Arudino Uno ploče uključuju nekoliko ključnih elemenata koji omogućavaju njenu funkcionisanje.



Slika 2. 3 Komponente Arduino Uno ploče [4]

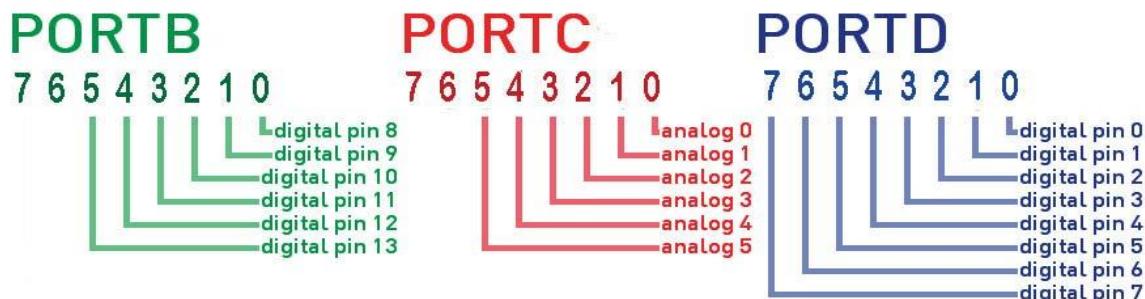
Glavna komponenta Arduino ploče je mikrokontroler, koji predstavlja "mozak" cijelog sistema. Mikrokontroler je zadužen za obradu podataka, izvršavanje programa i upravljanje svim povezanim uređajima. Na Arduino Uno ploči, ova ključna komponenta je ATmega328P (P je oznaka za "PicoPower" tehnologiju uštede energije), 8-bitni mikrokontroler koga proizvodi kompanija Atmel. Procesorsko jezgro Atmelovih kontrolera na bazi AVR tehnologije je izgrađeno po RISC dizajnu koji najčešće jednu mašinsku instrukciju obrađuje u toku jednog sistemskog takta. Mikrokontroler Atmega328P radi na maksimalnom taktu od 20 MHz, ali je zbog kompatibilnosti sa prethodnim modelima Arduina radni takt ograničen na 16 MHz. Osim toga, ovaj čip podržava različite brzine u zavisnosti od ulaznog napona. U slučaju da je napon na ulazu 1.8 V, procesor će raditi na maksimalnoj brzini od 4 MHz. Ako je pak u pitanju napon od 2.7 V brzina će biti 10 MHz. Za maksimalnu brzinu od 20 MHz je potreban napon od minimalno 4.5 V. Na slici 2. 4 je šematski prikazan mikrokontroler Atmega328P sa oznakama svih funkcija dodijeljenih pojedinim njegovim nožicama. [4]

Iz priloženog vidimo da, recimo, nožica označena brojem 5 može imati pet različitih funkcija. Ako malo pažljivije pogledamo ilustraciju, primjetićemo da su pinovima mikrokontrolera 23-28 dodijeljene funkcije analogno-digitalnog konvertora – ADC (engl. *Analog-to-Digital Converter*) i analognih ulaza (A0-A5). Međutim, odmah pored vidimo da isti pinovi obavljaju i funkciju digitalnih portova sa oznakama D14-D19, iako to nije obilježeno na samom uređaju (niti se o tome govori u specifikaciji). [4]



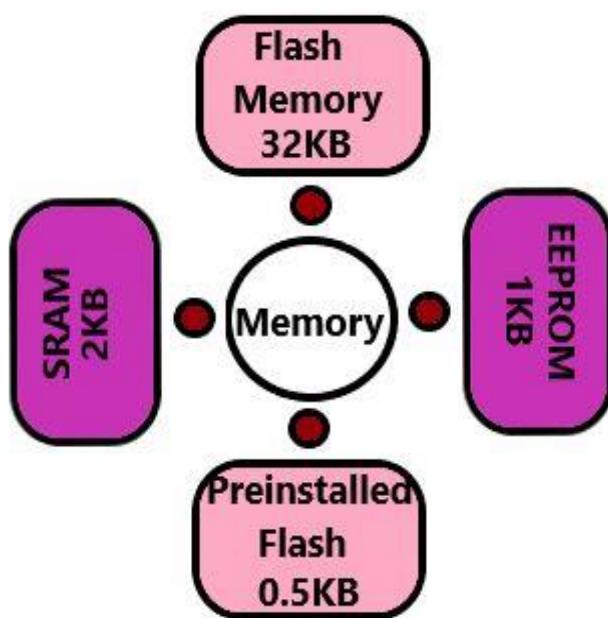
Slika 2. 4 ATmega328P raspodjela nožica [4]

Sve nožice mikrokontrolera osim onih čija je funkcija vezana za napajanje, imaju konektore sa početnim slovima PB, PC i PD iza kojih slijedi brojna oznaka. Ta tri porta se u Arduino žargonu nazivaju PORTB, PORTC i PORTD. PORTB se odnosi na digitalne pinove D13-D8, PORTC na analogne pinove A5-A0, dok se PORTD odnosi na digitalne pinove D7-D0. [4]



Slika 2. 5 ATmega328P raspodjela portova [4]

Što se tiče organizacije memorije, ATmega328P mikrokontroler organizuje memoriju u dva osnovna segmenta: programsku memoriju (engl. *flash memory*) i memoriju za podatke. U programskoj memoriji se pohranjuje kod programa i njena veličina iznosi 32 KB od kojih treba oduzeti 512 B namjenjenih smještanju tzv. "bootloader" programa koji omogućuje programiranje mikrokontrolera sa računara. Memorija za podatke se dalje dijeli na: staticku memoriju sa slučajnim pristupom – SRAM (engl. *Static Random Access Memory*) i električno izbrisivu programabilnu memoriju samo za čitanje – EEPROM (engl. *Electrically Erasable Programmable Read-Only Memory*). EEPROM je nevolatila memorija koja se koristi za trajno skladištenje podataka poput konfiguracijskih podataka ili stanja uređaja koji se ne smiju izgubiti kada se uređaj isključi. Ima kapacitet od 1 KB, što je dovoljno za čuvanje manjih količina podataka. SRAM je volatila memorija koja se koristi za privremeno skladištenje podataka tokom izvršavanja programa. Ovo uključuje varijable, stek i bafer memoriju. Ima kapacitet od 2 KB. Ovo je vrlo ograničen prostor, pa se programeri moraju pažljivo baviti korištenjem memorije da ne bi prepunili SRAM. [4]



Slika 2. 6 Struktura memorije [4]

Na pločici se nalazi USB-B konektor koji služi za programiranje uređaja, kao i njegovo napajanje dok je priključen na računar. Od ostalih uočljivijih elemenata na pločici možemo izdvojiti taster za restartovanje uređaja koji služi za restartovanje koda na ploči. Kada se ploča zaglavi, pritiskom na dugme vraća se u početno stanje. Regulator napona na Arduino ploči služi za stabilizaciju i regulaciju napona koji dolazi iz vanjskog izvora napajanja kako bi odgovarao naponu potrebnom za rad mikrokontrolera i drugih komponenti na ploči. Stvara stabilan napon od 5 V. Bez regulatora napona, promjene u naponu iz izvora napajanja mogle bi uzrokovati nestabilnost ili čak oštećenje elektronskih komponenti. [4]

Kristalni oscilator od 16 MHz na Arduino ploči služi za generisanje stabilnog takt-signalata koji kontrolira brzinu rada mikrokontrolera. Ovaj signal omogućava mikrokontroleru da izvršava instrukcije u preciznim vremenskim intervalima, što je ključno za pravilno funkcionisanje i sinhronizaciju svih operacija na ploči. U kontekstu Arduina, oscilator od 16 MHz postavlja brzinu "sata" mikrokontrolera, što direktno utiče na brzinu obrade podataka, izvršavanje programa i vremenske funkcije kao što su odgode i PWM. Ovaj precizan vremenski signal pomaže u osiguravanju da sve funkcije na ploči rade u skladu sa programiranim instrukcijama i vremenskim zahtjevima. [4]

Tu su i četiri minijaturna LED svjetla od kojih dva prikazuju slanje i prijem podataka preko serijskog porta (Rx, Tx), jedno prikazuje da je uređaj pod naponom, dok je četvrto fizički povezano sa pinom 13 i služi kao indikator njegove aktivnosti. [4]

Arduino Uno ploča može se napajati putem USB priključka ili vanjskog izvora napajanja. Izvor napajanja se automatski odabire. Vanjsko (ne-USB) napajanje može dolaziti iz AC-DC adaptera (adaptera za zidnu utičnicu) ili baterije. Adapter se može priključiti spajanjem utikača od 2.1 mm s pozitivnim centrom u naponski priključak ploče. Vodovi iz baterije mogu se povezati na "GND" i "Vin" pinove konektora za napajanje. Ploča može raditi na vanjskom napajanju u rasponu od 6 do 20 volti. Međutim, ako se napaja s manje od 7 V, pin za 5 V može davati manje od pet volti, što može uzrokovati nestabilnost ploče. Ako se koristi više od 12 V, regulator napona može se pregrijati i oštetići ploču. Preporučeni raspon napona je od 7 do 12 volti. [5]

Pinovi napajanja su sljedeći [5]:

- "Vin" pin na Arduino ploči služi kao ulazni napon kada se ploča napaja putem vanjskog izvora napajanja, kao što je AC-DC adapter. Kada koristite ovaj vanjski izvor, napon koji dolazi na "Vin" pin odgovara naponu adaptera, što obično može biti između 7 i 12 volti, ali može varirati u zavisnosti od napona adaptera koji koristite. Ovaj napon se zatim prosljeđuje kroz regulator napona na ploči, koji ga smanjuje na 5 V i 3.3 V za korištenje na ostalim pinovima. Kada napajate ploču putem USB priključka, "Vin" pin neće imati napon, jer ploča dobija stabilnih 5 volti direktno iz USB-a. "Vin" pin se najčešće koristi kada je potrebno napajanje ploče nezavisno od USB priključka, pružajući fleksibilnost u izboru izvora napajanja.
- "5V" pin na Arduino ploči isporučuje regulisani napon od 5 volti koji je rezultat obrade ulaznog napona kroz ugrađeni regulator napona. Kada ploča dobija napajanje putem USB priključka, 5 V pin pruža direktno 5 volti iz USB izvora. Ako se ploča napaja putem vanjskog adaptera koji je priključen na naponski priključak, napon na 5 V pin će i dalje biti regulisan na 5 volti zahvaljujući regulatoru napona koji pretvara viši ulazni napon na "Vin" pinu (koji može biti između 7 i 12 volti) u stabilnih 5 volti. Važno je napomenuti da direktno napajanje ploče preko 5 V ili 3.3 V pinova zaobilazi regulator napona, što može dovesti do oštećenja ploče ako napon nije pravilno regulisan ili je izvan preporučenih vrijednosti.
- "3V3" pin obezbjeđuje 3.3-voltno napajanje koje generiše regulator napona na ploči. Maksimalna struja koju ovaj pin može isporučiti je 50 mA.
- "Gnd" pinovi za uzemljenje.

- “IOREF” pin na Arduino ploči pruža važan podatak o naponu s kojim mikrokontroler radi. Ovaj pin omogućava da proširene ploče ili dodaci prepoznaju napon koji se koristi za rad mikrokontrolera i prema tome se prilagode.

Sljedeće polje od šest pinova pripada analognim ulazima i označeno je kao A0-A5, čija je namjena da prihvate informacije u analognom obliku koje su predstavljene različitim naponima. Te informacije se dalje proslijeđuju u tzv. ADC, gdje se pristigli napon pretvara u neku vrijednost izraženu 10-bitnim brojem. Pinovi A4 i A5 su interna povezani sa pinovima SDA i SCL, što omogućava uspostavljanje komunikacije sa drugim uređajima putem integriranog kruga magistrale – I2C (engl. *Inter-Integrated Circuit*). I2C je jedan od najjednostavnijih protokola za serijski prenos podataka i za to su mu potrebne svega dvije žice. Brzine prenosa ove magistrale se kreću od 1-5 Mb/s. [4]

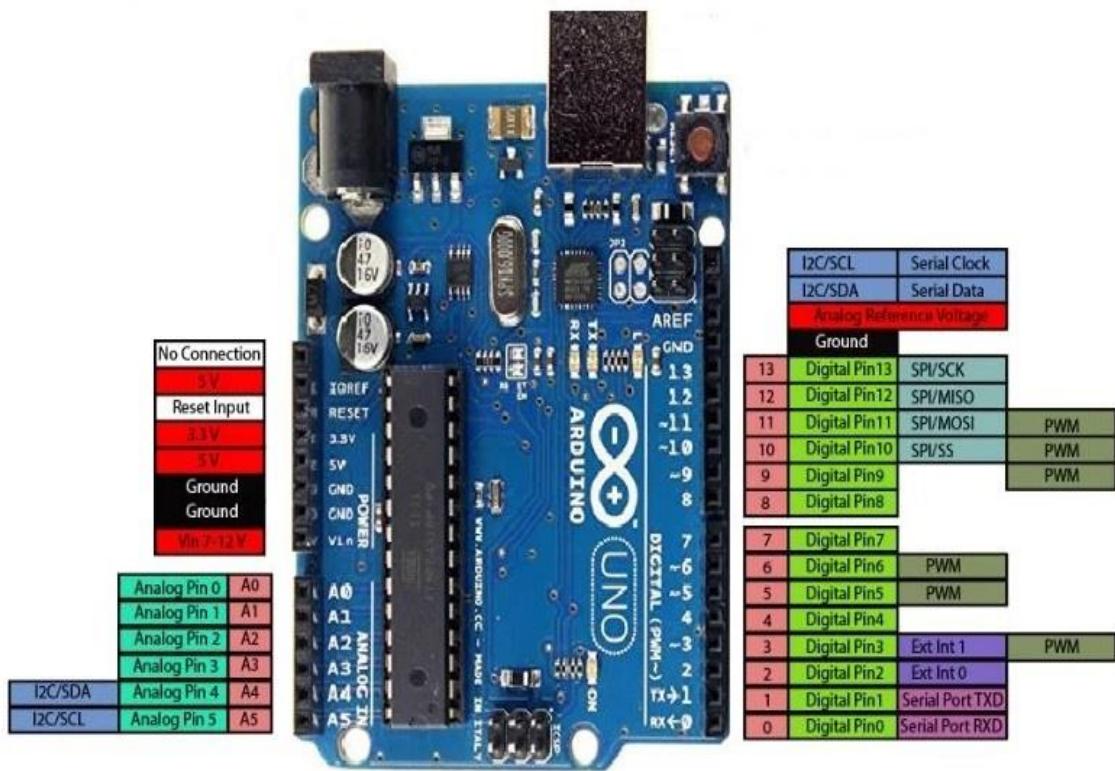
Sa gornje strane pločice se nalaze dva konektora (8 i 10 pinova) koji su najvećim svojim dijelom namijenjeni radu sa digitalnim signalima. Pinovi sa oznakama 0-13 po potrebi mogu da imaju funkciju kako ulaznih, tako i izlaznih konektora. Pored pinova 3, 5, 6, 9, 10 i 11 se nalazi znak „~“ koji napominje da se oni mogu koristiti kao pseudoanalogni izlazni portovi. Pošto Atmega328P nema logiku digitalno-analognog konvertora – DAC (engl. *Digital-to-Analog Converter*), koristi se tehnika PWM za generisanje analognih signala putem modulacije kvadratnog talasa. [4]

Na digitalnim pinovima D0 i D1 se nalaze oznake Rx i Tx koje sugeriraju da se radi o kontaktima koji mogu biti korišteni kao univerzalni sinhroni/asinhroni prijemnik/predajnik – USART (engl. *Universal Synchronous/Asynchronous Receiver/Transmitter*), odnosno serijski interfejs preko koga možemo komunicirati sa drugim računarima i uređajima, a u slučaju potrebe ga možemo koristiti za programiranje samog Arduina. [4]

Pinovi D10-D13 obavljaju i ulogu porta serijskog perifernog interfejsa – SPI (engl. *Serial Peripheral Interface*). Riječ je o serijskoj magistrali koja za razliku od ostalih serijskih protokola koje smo obradili omogućuje rad u dupleks režimu (istovremeno slanje i primanje podataka). Interfejs koristi četiri pina za svoj rad i to: D10 - izbor robota – SS (engl. *Slave Select*) pin koji koristi master za (de)aktivaciju uređaja; D11 - izlaz iz glavnog uređaja, ulaz u robota – MOSI (engl. *Master Out Slave In*) master linija za slanje podataka na slave; D12 - ulazni podaci za glavnu jedinicu – MISO (engl. *Master In Slave Out*) slave linija za slanje podataka masteru i D13 - serijski sat – SCK (engl. *Serial Clock*) sat za sinhronizaciju prenosa. [4]

Pin 15 pod nazivom AREF služi za određivanje gornje vrijednosti ulaznog opsega za analogno-digitalni konverter. Podrazumijevana vrijednost je pet volti za modele koji rade na naponu od 5 V i 3,3 V za uređaje koji nemaju napajanje od 5 volti. Dovođenje većeg napona na ovaj pin može da pokvari uređaj. [4]

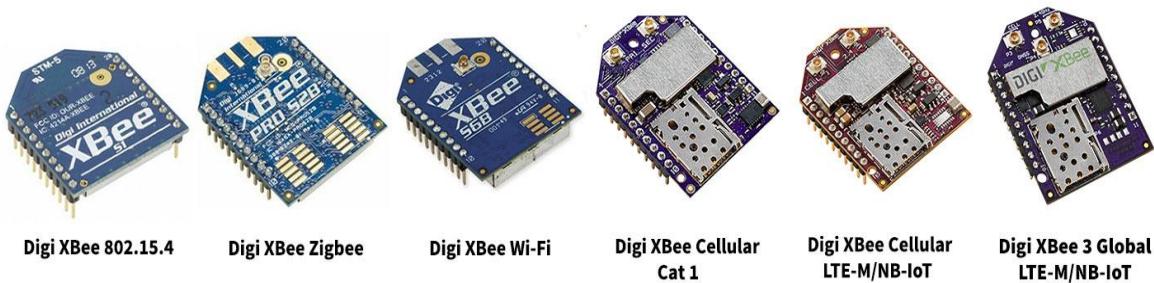
Preostala dva pina SCL i SDA (16 i 17) predstavljaju instancu serijskog interfejsa koji se naziva interfejs dvije žice – TWI (engl. *Two Wire Interface*), a koji je zapravo isto što i I2C, samo se koristi drugo ime kako bi se izbjegao sudski spor sa “Filipsom”, pošto je I2C registrovan kao njihov trgovачki znak. [4] Detaljan raspored pinova možemo pogledati na slici 2. 7.



Slika 2. 7 Raspored pinova [1]

2.1.2 XBee komunikacijski modul

Digi XBee je brend prepoznatljivih bežičnih komunikacijskih modula koji su standardizovani po obliku i potiču od američke kompanije “Digi International” koja je specijalizovana za tehnologije industrijskog interneta stvari – IIoT (engl. *Industrial Internet of Things*). Digi International nudi izuzetno širok spektar XBee radio uređaja. Ukupno, postoji najmanje 30 različitih kombinacija hardverskih komponenti, firmver (engl. *firmware*) protokola, snage prenosa i opcija antena. Prvi XBee moduli predstavljeni su pod brendom „MaxStream“ 2005. godine i bazirani su na standardu IEEE 802.15.4-2003, koji je razvijen od strane Instituta za električne i elektronske inženjere – IEEE (engl. *Institute of Electrical and Electronics Engineers*) za “tačka-do-tačka” – P2P (engl. *point-to-point*) i zvijezda (engl *star*) komunikacije. XBee modul je vrlo mali radio uređaj koji koristi različite komunikacijske protokole za slanje informacija, kao što su podaci sa senzora. Od početnog predstavljanja, XBee porodica se proširila, a nastao je kompletan ekosistem bežičnih modula, gateway-a, adaptera i softvera. Digi XBee moduli su izdržljivi, fleksibilni, sigurni i pouzdani, što ih čini veoma popularnim za integraciju bežične komunikacije u povezane proizvode. Digi XBee radio uređaji nalaze primjenu u širokom spektru aplikacija Internet stvari – IoT (engl. *Internet of Things*), uključujući industrijska senzorska rješenja, pametnu poljoprivredu i pametno osvjetljenje u urbanim sredinama. [6]



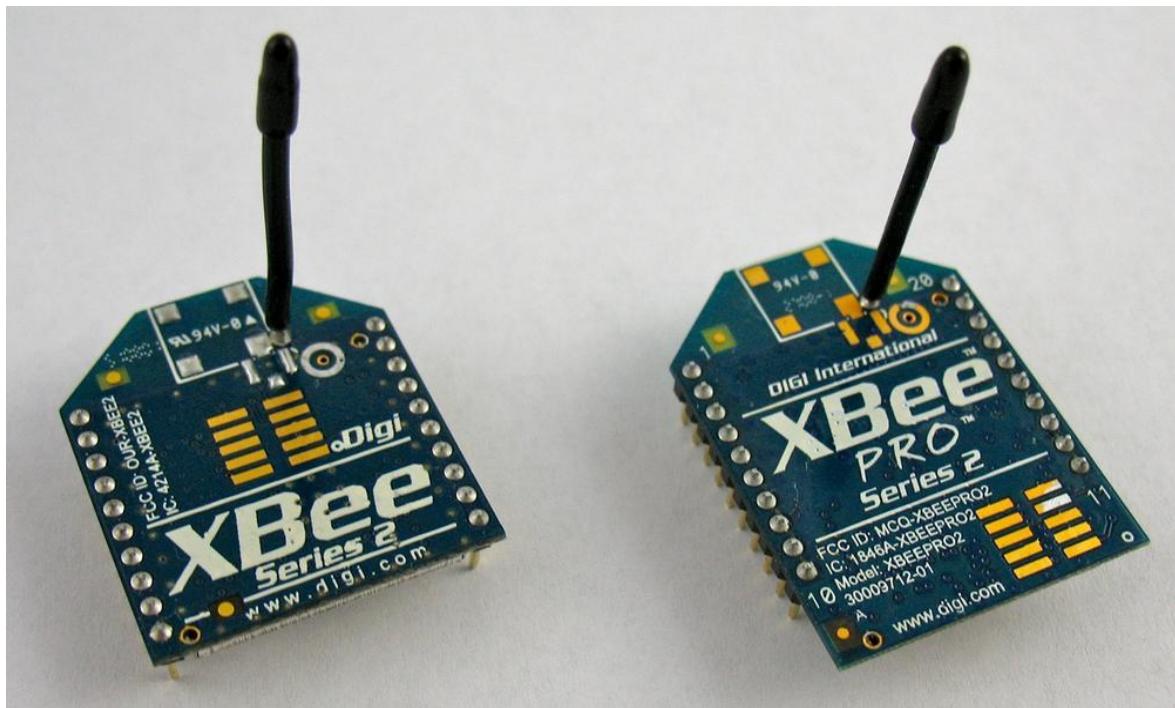
Slika 2.8 Evolucija XBee modula [7]

Kao što je prethodno navedeno, XBee modul je predstavljen 2005. godine od strane kompanije MaxStream. Autor u [7] dalje navodi da je Digi sljedeće godine kupio MaxStream i lansirao certificirani ZigBee XBee modul. Ovi proizvodi su postali toliko popularni da su nakon toga uslijedili Wi-Fi moduli. Treće strane su počele ugrađivati kompatibilne konektore za Digi XBee u svoje vlastite proizvode. Zatim je 2018. godine Digi predstavio Digi XBee 3 liniju programabilnih modula. Porodica je rasla sa dodatnim XBee modulima koji su povećali domet, kao i sa opcijama za mobilnu mrežu, te je razvijen ekosistem pratećeg softvera, biblioteka i alata koji pojednostavljaju razvoj XBee modula. Kratak pregled evolucije XBee modula:

- 802.15.4: Moduli XBee Serije 1, predstavljeni 2005. godine, predstavljaju pionire u XBee seriji sa svojim klasičnim 20-pinskim oblikom i podrškom za IEEE 802.15.4 standard. Njihov jednostavan AT komandni interfejs postao je standard u XBee liniji. Važno je napomenuti da moduli nisu kompatibilni sa Serijom 2, što je značajna stavka za razmatranje prilikom odabira modula za specifične primjene.
- ZigBee: Oslanjajući se na uspjeh XBee modula, Digi je predstavio "Seriju 2" XBee modul sa podrškom za ZigBee "mesh" umrežavanje. Serija 2 koristi mikročip iz "Ember Networks-a". ZigBee moduli automatski formiraju mreže. "Samo-izlječiva" mreža može da pronađe alternativne rute za slanje podataka. Ovo pomaže u očuvanju funkcionalnosti mreže i minimiziranju prekida komunikacije.
- Wi-Fi: Originalni XBee Wi-Fi moduli kompanije Digi predstavljeni su 2011. godine sa globalnom podrškom za Wi-Fi protokole. Druga verzija lansirana je naredne godine i uključivala je mogućnost ažuriranja putem bežične veze.
- Dugoročni razvoj – LTE (engl. *Long Term Evolution*) Cat 1: Digi XBee LTE Cat 1 označava ulazak kompanije Digi na tržište mobilnih mreža. Ovaj modul, koji je naslednik 3G tehnologije, je dizajniran za uređaje sa manjim potrebama za propusnošću i omogućava povezivanje u aplikacijama gde Wi-Fi nije dostupan.
- LTE Cat 4: Najnoviji modul u LTE liniji proširuje podršku za veće potrebe za propusnošću, kao što su stremovanje medijskih datoteka ili prenos podataka vrlo visokih frekvencija.

- LTE-M/NB-IoT: Nakon XBee LTE Cat 1, Digi je razvila i certificirala module koji podržavaju LTE-M i NB-IoT mobilne protokole. Ovi standardi omogućavaju uređajima sa baterijskim napajanjem da funkcionišu i budu povezani sa mobilnim mrežama tokom dugih vremenskih perioda.
- DigiMesh radio frekvencijski – RF (engl. *Radio Frequency*) moduli: Digi je razvila DigiMesh kako bi popunila praznine u funkcionalnosti ZigBee-a. U DigiMesh mreži, svi moduli su istog tipa uređaja i mogu biti napajani baterijama, što poboljšava postavljanje i stvara otpornu mrežu bez potrebe za stalno uključenim ruter modulima. DigiMesh može raditi na 2.4 GHz ili 868/900 MHz.
- 868/900MHz: Digi XBee-PRO 900HP i Digi XBee XR 868 pokrivaju domet od 9 milja (14.5 kilometara), uz opcije za mesh umrežavanje. Digi XBee SX 868/900 uključuje DigiMesh i može se konfigurirati za domet od 65 milja (104.6 kilometara).

XBee moduli dolaze u dvije verzije: običnoj i PRO. Obična verzija, jednostavno nazvana XBee, podržava aplikacije srednjeg dometa i koristi umjerenu količinu snage prenosa, što omogućava duži vijek trajanja baterije, manju proizvodnju toplote i niže troškove. Ovi moduli su pogodniji za primjene koje ne zahtijevaju velike domete i mogu biti certifikovani u regijama sa restrikcijama na snagu prenosa. S druge strane, XBee-PRO moduli su dizajnirani za veće domete i uključuju veće snage prenosa. Ovi PRO moduli, kao što su Digi XBee-PRO 900HP i Digi XBee-PRO SX, omogućavaju komunikaciju na udaljenostima od 14.5 kilometara do 104.6 kilometara. Zbog svoje veće snage, XBee-PRO moduli su veći, skuplji i proizvode više toplote, ali pružaju dug domet i veću pouzdanost u aplikacijama koje zahtijevaju veći domet komunikacije. [8]



Slika 2. 9 XBee moduli u standardnoj i pro verziji [8]

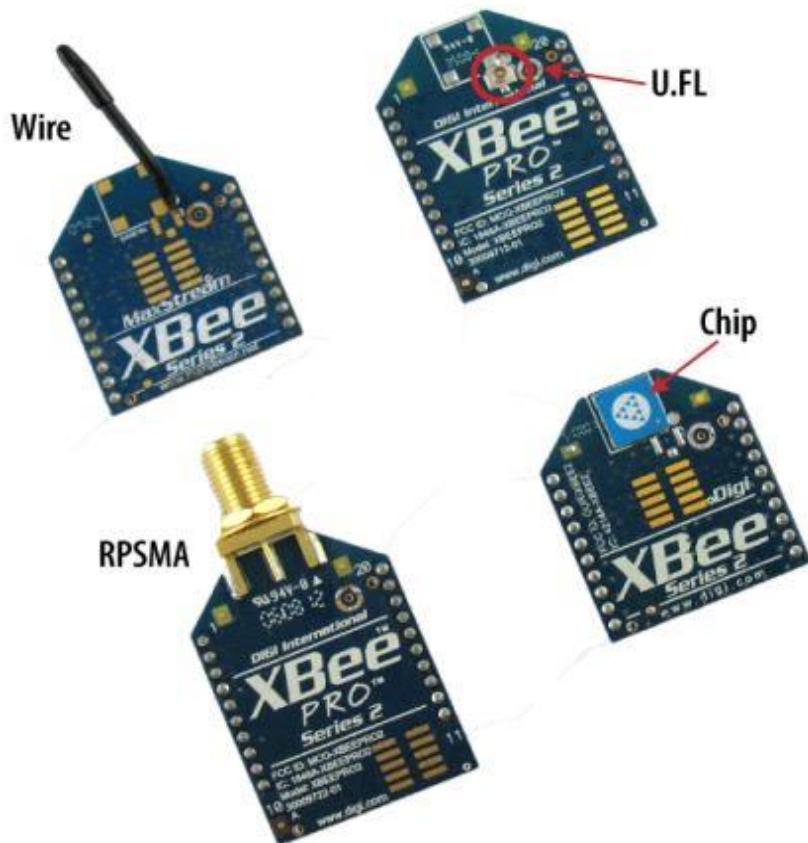
Karakteristika	Serijski 1	Serijski 2 (S2)	Serijski 2B (S2B)	PRO Serijski 2B (PRO S2B)
Protokol	IEEE 802.15.4	Zigbee	Zigbee	Zigbee
Vrsta mreže	Point-to-point, Point-to-multipoint	Mesh, Point-to-multipoint, Point-to-point	Mesh, Point-to-multipoint, Point-to-point	Mesh, Point-to-multipoint, Point-to-point
Frekvencija	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz
Domet (zatvoreni prostor)	30 m	40 m	40 m	90 m
Domet (otvoreni prostor)	100 m	120 m	120 m	3200 m
Snaga prenosa	1 mW (0 dBm)	2 mW (3 dBm)	2 mW (3 dBm)	63 mW (18 dBm)
Brzina prenosa	250 Kbps	250 Kbps	250 Kbps	250 Kbps
Struja prenosa	45 mA	40 mA	45 mA	215 mA
Struja prijema	50 mA	40 mA	40 mA	55 mA
Cijena	\$20-\$25	\$25-\$30	\$35-\$45	\$45
Kompatibilnost	Nije kompatibilan sa Serijom 2	Nije kompatibilan sa Serijom 1	Kompatibilan sa Serijom 2	Kompatibilan sa Serijom 2

Tabela 2. 2 Karakteristike XBee modula

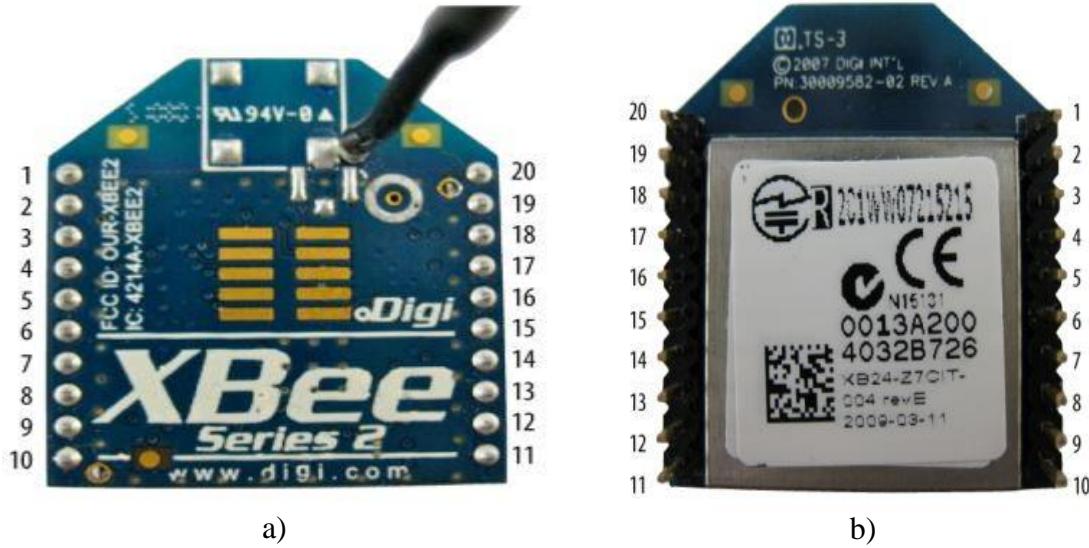
Radiji koriste antene za prenos i prijem signala, a postoji niz različitih konstrukcija antena, od kojih svaka ima svoje prednosti i nedostatke. Izbor prave antene zavisi od veličine uređaja, materijala, potreba za dometom i mehaničkih ograničenja. Kako bi zadovoljio različite zahtjeve za dometom i udaljenošću XBee uređaja, Digi nudi širok spektar antena. U nastavku su prikazane vrste antena koje su trenutno dostupne [8]:

- Žičana antena: Ovo je upravo ono što ime sugerije, jedan komad žice koja se izdiže iz tijela radija. U većini slučajeva, žičana antena je upravo ono što vam je potrebno. Ona je jednostavna i nudi omnidirekionalno zračenje, što znači da je maksimalna udaljenost prenosa otprilike ista u svim pravcima kada je žica ravna i okomita na modul.
- Čip antena: Kao što ime sugerije, čip antena je ravna keramička čip antena koja je u ravni sa tijelom XBee modula. Ova antena je manja i čvršća, ali te prednosti dolaze uz određene kompromise. Čip antene imaju kardioidni (srcoliki) obrazac zračenja, što znači da se signal slabi u mnogim pravcima. Međutim, ako pravite uređaj u kojem bi mehanički stres na žičanu antenu mogao uzrokovati njen prekid, ili ako treba da smjestite radio u veoma mali prostor, čip antena može biti najbolji izbor. Čip antene su često pravi izbor za bilo šta što se nosi.
- Antena za štampanu ploču – PCB (engl. *Printed Circuit Board*): Predstavljena sa XBee-PRO S2B modelom, PCB antena je štampana direktno na štampanoj ploči XBee modula. Sastoji se od niza provodnih tragova raspoređenih u fraktalnom uzorku. PCB antena nudi mnoge iste prednosti (i mane) kao čip antena, ali uz mnogo niže troškove proizvodnje.

- Ultra minijaturni koaksijalni – U. FL (engl. *Ultra Miniature Coaxial Connector*) konektor: Ovo je manji od dva tipa spoljnih konektora za antene. U većini slučajeva, spoljna antena nije neophodna, a dodatni trošak je opravdan samo ako jednostavna žičana antena ne može da ispunи zahtjeve. Međutim, kada vaš radio treba da se nalazi unutar metalne kutije, antena će morati da bude postavljena spolja. Na taj način signal neće biti oslabljen zbog kućišta. Takođe, ponekad je korisno usmjeriti spoljnu antenu drugačije od same XBee jedinice ili koristiti antenu specijalne namjene sa određenim obrascem zračenja, poput antene sa visokim dobitkom koja prenosi signale u jednom pravcu na većim udaljenostima. U. FL konektor je mali, pomalo krhak, pa je potrebno pažljivo rukovati prilikom povezivanja i odvajanja. U. FL konektori su poznati po svojoj maloj veličini i često se koriste u aplikacijama gde je prostor ograničen, poput u mobilnim telefonima, laptopovima i bežičnim uređajima.
- Konektor sa obrnutim polaritetom u verziji A – RPSMA (engl. *Reverse Polarity SubMiniature Version A*): RPSMA konektor je samo drugačija vrsta konektora u odnosu na U. FL konektor. Veći je i robustniji, ali se može koristiti sa spoljnjom antenom koja se montira direktno na XBee bez potrebe za povezivanjem kabla, čime se štedi prostor i troškovi, dok se istovremeno poboljšava performansa. Za većinu početničkih projekata, i dalje je najbolje koristiti jednostavnu žičanu antenu koja je manja, jeftinija i obično jednako dobra.



Slika 2. 10 Tipovi antena [8]



Slika 2. 11 Fizičko numerisanje pinova a) prednji prikaz b) poledinski prikaz [8]

Pin #	Ime	Opis
1	VCC	3.3 V napajanje
2	DOUT	Izlaz podataka (Tx)
3	DIN	Ulaz podataka (Rx)
4	DIO12	Digitalni I/O 12
5	RESET	Reset modula
6	PWM0/RSSI/DIO10	Izlaz analognog impulsa širine (PWM) 0, Indikator snage prijema signala (RSSI), Digitalni I/O 10
7	DIO11	Digitalni I/O 11
8	Reserved	Ne povezivati (rezervisan za buduće namjene)
9	DTR/SLEEP_RQ/DIO8	Pripremljen za rad (DTR) (signal za hardversko rukovanje), Kontrola sna, Digitalni I/O 8
10	GND	Masa
11	DIO4	Digitalni I/O 4
12	CTS/DIO7	Spreman za slanje (CTS), Digitalni I/O 7
13	ON/SLEEP	Indikator sna (ugašen kada je modul u režimu spavanja)
14	VREF	Referentni napon za ADC
15	ASSOC/DIO5	Indikator povezivanja (trepće ako je modul povezan sa mrežom, stalno svijetli ako nije), Digitalni I/O 5
16	RTS/DIO6	Zahtjev za slanje (RTS) (hardversko rukovanje), Digitalni I/O 6
17	AD3/DIO3	Analogni ulaz 3, Digitalni I/O 3
18	AD2/DIO2	Analogni ulaz 2, Digitalni I/O 2
19	AD1/DIO1	Analogni ulaz 1, Digitalni I/O 1
20	AD0/DIO0/COMMIS	Analogni ulaz 0, Digitalni I/O 0, Dugme za komisionisanje

Tabela 2. 3 XBee opis pinova

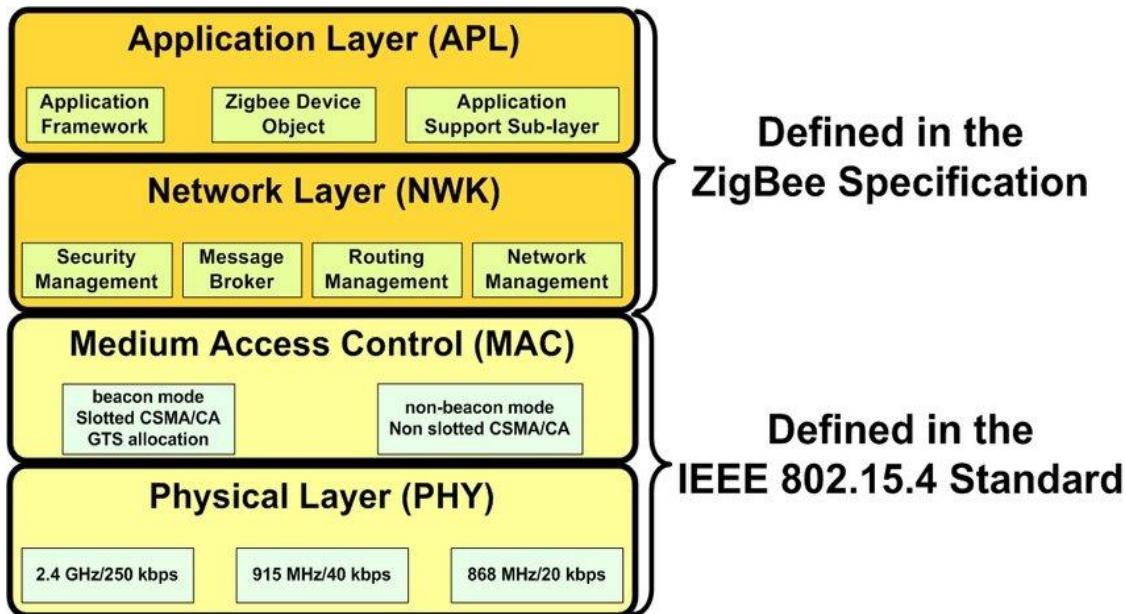
ZigBee je skup komunikacijskih protokola višeg sloja, definisan od strane grupe kompanija pod nazivom "ZigBee Alliance", temeljenih na IEEE 802.15.4 standardu za bežičnu mrežu kratkog dometa – WPAN (engl. Wireless Personal Area Network), baziranom na radio prenosu. ZigBee Alijansa je otvoreno, neprofitno udruženje većeg broja kompanija iz područja elektronike koji su se udružili sa ciljem razvoja ekoloških tzv. "green" standarda za bežične mreže, koje bi karakterisala: niska brzina prenosa podataka, mala složenost standarda, jednostavna implementacija standarda u uređaju, mala količina podataka koja bi se prenosila, malo vrijeme aktivnog stanja (engl. *low duty cycle*) uređaja te veoma niska potrošnja, koja bi omogućila dugotrajni rad uređaja u mreži bez potrebe za mijenjanjem baterija ili nadopunjavanjem baterija vanjskim izvorima energije (npr. solarne čelije). [9]

Naziv je dobio po pčelama koje lete zig-zag među cvjetovima tvoreći petljastu topologiju međusobno povezanih mreža (engl. *mesh*), koja je najfleksibilniji i najrobustniji vid umrežavanja. Ciljane primjene ZigBee mreža su aplikacije za nadzor i upravljanje koje zahtijevaju umrežavanje velikog broja uređaja, prenos male količine podataka, brzina (maksimalno 250 kb/s), jednostavnost, nisku cijenu, upotreba frekvencijskih opsega bez posebnih dozvola slobodnih (nelicenciranih), malu potrošnju energije te visoku sigurnost, pouzdanost prenosa i sposobnost samoprilagodbe mreže i preusmjeravanja poruka. ZigBee se temelji na IEEE 802.15.4, ali se često ova dva pojma poistovjećuju. ZigBee definiše programsko okruženje ZigBee framework i omogućuje raznim kompanijama jednostavnu realizaciju softvera. Primjenjuje se u kućnoj i poslovnoj automatizaciji, industrijskoj kontroli, sigurnosnim sistemima, medicinskim senzorima, igračkama, itd. ZigBee se nudi kao rješenje onim korisnicima koji ne žele postati stručnjaci na području bežičnih tehnologija, nego jednostavno imaju potrebu za bežičnom mrežom koja je pouzdana. Takva mreža je i [10]:

- Samo-organizirajuća (engl. *self-organizing*) čvorovi mreže imaju mogućnost detekcije novog čvora te obavljanja potrebne reorganizacije mreže u funkcionalnu mrežu s novim čvorovima bez ljudske intervencije.
- Samo-izlječiva (engl. *self-healing*) čvorovi imaju mogućnost otkrivanja greške do koje je došlo ili u nekom čvoru ili u međukomunikaciji bez ljudske intervencije.
- Sigurna sa ugrađenim sigurnosnim mehanizmima i zaštićena standardom naprednog šifriranja – AES (engl. *Advanced Encryption Standard*).

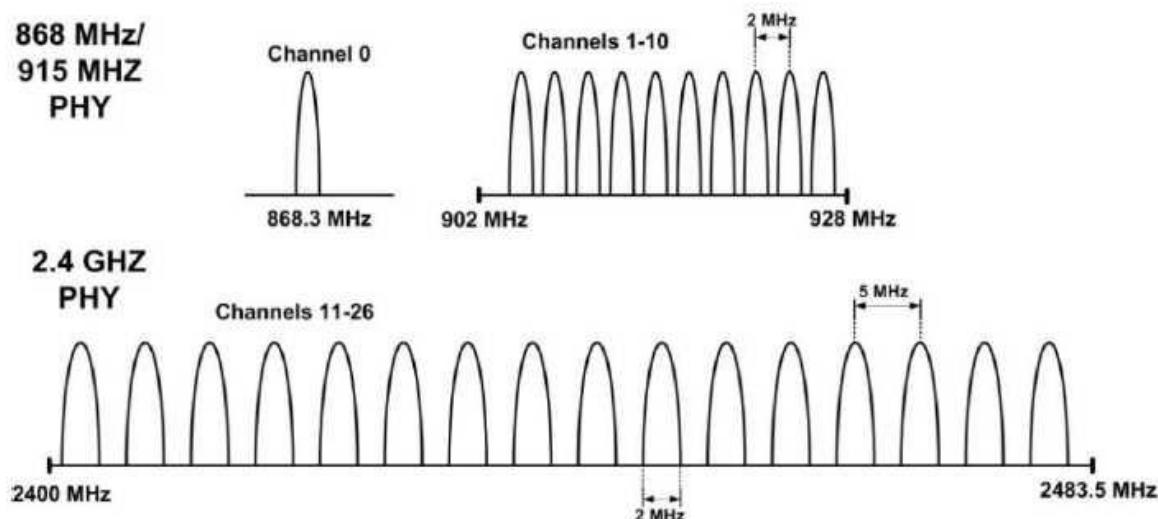
ZigBee protokol koristi IEEE 802.15.4 standard koji definiše dva niža sloja: fizički sloj – PHY (engl. *Physical*) i sloj kontrole pristupa mediju – MAC (engl. *Medium Access Control*), dok je ZigBee Alliance-a definisala i mrežni sloj – NWK (engl. *Network*) i aplikacijski sloj – APS (engl. *Application*). [11] ZigBee slojevita arhitektura je prikazana na slici 2. 12.

Fizički sloj u ZigBee protokolu predstavlja osnovni sloj u otvorenom sistemskom interfejsu – OSI (engl. *Open Systems Interconnection*) referentnog modela, koji omogućava prenos podataka i interfejs prema MAC podsloju i fizičkom radio kanalu. Fizički sloj upravlja fizičkim radio predajnikom, vrši odabir kanala, kao i funkcije upravljanja energijom i signalom. Odgovoran je za generisanje paketa, prijem paketa, transparentnost podataka i upravljanje energijom. Antena je ključna komponenta fizičkog sloja jer omogućava modulima da šalju i primaju radio signale koji su temelj za sve komunikacije. [12]



Slika 2. 12 ZigBee protokol slojevita arhitektura

ZigBee protokol operiše na tri glavna frekventna opsega: 868–868.6 MHz u Evropi, 902–928 MHz u Severnoj Americi i Australiji, i najčešće korišćeni 2400–2483.5 MHz opseg koji je dostupan globalno. Na frekvenciji od 2.4 GHz, ZigBee koristi 16 kanala sa razmakom od 5 MHz između njih, pri čemu svaki kanal koristi samo 2 MHz širine opsega, a za prenos koristi kvadratno fazno pomjeranje ključa – OQPSK (engl. *Offset Quadrature Phase Shift Keying*) modulaciju, koja prenosi 2 bita po simbolu i omogućava brzinu prenosa do 250 kbps. U opsegu od 868 MHz dostupan je jedan kanal sa brzinom prenosa od 20 kbps, dok opseg od 915 MHz nudi 10 kanala sa brzinom prenosa od 40 kbps. Oba opsega koriste binarno fazno pomjeranje ključa – BPSK (engl. *Binary Phase Shift Keying*). [12]



Slika 2. 13 Frekvencijski opseg ZigBee kanala

MAC sloj je odgovoran za kontrolu pristupa mediju u ZigBee protokolu. On upravlja pristupom radio kanalu koristeći višestruki pristup sa osluškivanjem nosioca i izbjegavanjem kolizija – CSMA-CA (engl. *Carrier Sense Multiple Access with Collision Avoidance*) mehanizam, koji omogućava uređaju da osluškuje kanal prije nego što pošalje poruku, čime se izbjegava kolizija podataka. Zadužen je za slanje signalnih ramova, sinhronizaciju između uređaja i pružanje pouzdane transmisije podataka. [11] Kao što je prikazano na slici 2. 12, sloj može raditi u dva različita režima: “beacon” režimu i “non-beacon” režimu, koji će biti detaljno objašnjeni u nastavku rada.

ZigBee Alijansa definiše mrežni sloj i aplikacijski sloj. Glavne funkcije mrežnog sloja su osigurati pravilnu upotrebu MAC podsloja i pružiti odgovarajući interfejs za upotrebu sljedećeg višeg sloja, odnosno aplikacijskog sloja. Mrežni sloj je zadužen za pravilno formiranje mrežne topologije, konfigurisanje uređaja, uključivanje i isključivanje čvora s mreže, dostavljanje poruke pravom odredištu, pravilno adresiranje, otkrivanje “komšija” i pravih puteva između dva čvora. Ovaj sloj koristi zvjezdastu, mrežnu i stablastu topologiju. Prema IEEE 802.15.4 standardu uređaji u mreži mogu se podijeliti na uređaje sa potpunom funkcionalnošću – FFD (engl. *Full Functional Device*) i uređaje sa redukovanim funkcionalnošću – RFD (engl. *Reduced Functional Device*). Koordinator i ruter su obavezno FFD uređaji, dok je krajnji uređaj najčešće RFD uređaj. [11]

Konfiguracija zvjezdaste topologije je također prilično jednostavna. Koordinator se nalazi u središtu zvjezdaste topologije i povezan je s krugom krajnjih uređaja. Svaka poruka u sistemu mora proći kroz koordinator, koji je odgovoran za usmjeravanje poruka između uređaja prema potrebi. Krajnji uređaji ne komuniciraju međusobno direktno. Mrežasta topologija slična je topologiji stabla, ali ima gušće veze i mreža je otpornija na smetnje. Implementacija ove topologije zahtijeva značajno više memoriskog prostora zbog složenosti umrežavanja i upravljanja podacima. U topologiji međusobno povezanih mreža svi uređaji mogu neposredno komunicirati ako su u dometu jedan drugog. Glavna odlika ove topologije je povezivanje čvorišta preko više putanja, što čini mrežu vrlo efikasnom i robusnom. Zbog standardizacije, omogućeno je korištenje komponenata različitih proizvođača, što rezultira međusobnom kompatibilnošću. Osim toga, mesh mreža je samoizlječiva, što znači da može automatski rekonfigurisati svoje rute i nastaviti s radom čak i ako dođe do kvara nekih čvorova, uključujući i koordinatora. U topologiji stabla, koordinator djeluje kao korijenski čvor odgovoran za uspostavljanje mreže i odabir određenih ključnih mrežnih parametara. Ruter može biti dijete koordinatora ili drugog rutera i odgovoran je za prenošenje podataka i kontrolnih poruka kroz mrežu koristeći hijerarhijsku strategiju rutiranja. Krajnji uređaj može biti dijete koordinatora ili rutera i može komunicirati s drugim krajnjim uređajem samo putem rutera ili koordinatora. [10] Topologija ZigBee mreža predstavljena je na slici 2. 14.

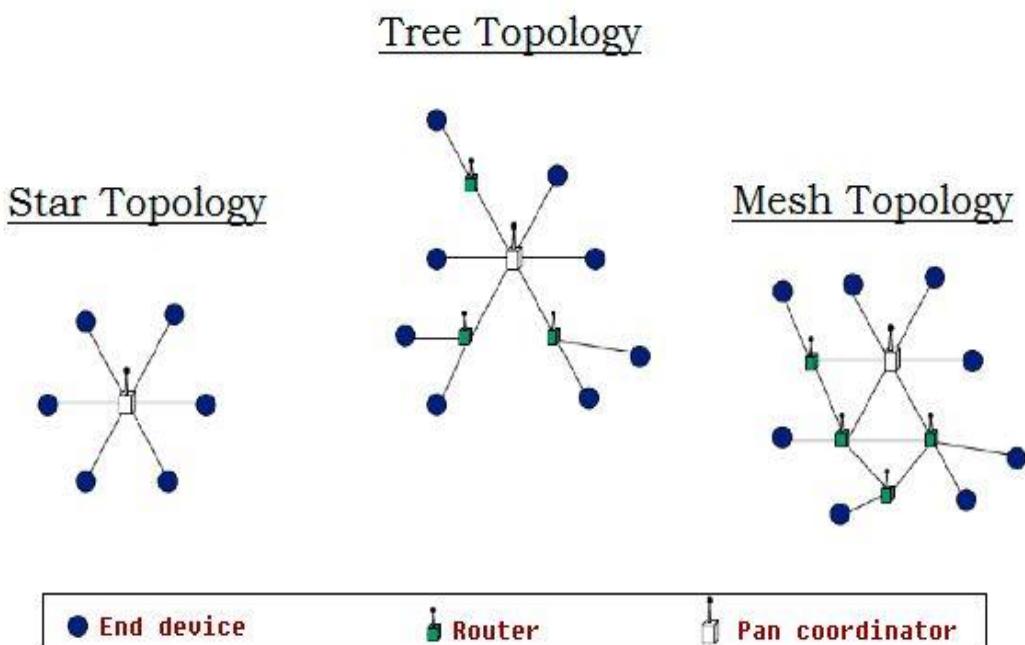
Aplikacijski sloj je zadužen za ispravnu komunikaciju između aplikacija koje koriste ZigBee mrežu kao sredstvo komunikacije među čvorovima. Sastoji se od tri podsloja [10]:

- Aplikacijski okvir (engl. *Application Framework*) je okruženje u kojem su hostovani aplikacijski objekti (može se definirati do 254 objekta). Ovi objekti su obično definirani od strane proizvođača. Aplikacijski okvir definira aplikacijske profile i klastere.

- ZigBee uređajni objekti – ZDO (engl. *ZigBee Device Objects*) podsloj otkriva uređaje u mreži, započinje i/ili odgovara na zahtjeve za povezivanje, uspostavlja sigurnosne veze među uređajima u mreži i ima: ZigBee koordinator – ZC (engl. *ZigBee Coordinator*), ZigBee ruter – ZR (engl. *ZigBee Router*) i ZigBee krajnji uredjaj – ZED (engl. *ZigBee End Device*).
- Aplikacijska podrška – APS (engl. *Application Support*) podsloj predstavlja interfejs između mrežnog i aplikacijskog sloja i omogućava uspostavljanje i održavanje sigurnosnih veza. Pruža usluge prenosa podataka između aplikacijskih entiteta – APSDE (engl. *APS Data Entity*) i upravlja sigurnosnim uslugama, povezivanjem uređaja i upravljanje grupama – APSME (eng. *APS Management Entity*). APS podsloj osigurava sigurnost poruka koristeći link ključeve ili mrežni ključ, te je zadužen za sigurno slanje, prijem okvira i upravljanje kriptografskim ključevima.

ZigBee mreže spadaju u personalne mreže – PAN (engl. *Personal Area Network*). Svaka mreža ima svoj vlastiti jedinstven identifikator – PAN ID (engl. *PAN Identifier*). Zigbee definiše tri različite vrste uređaja: koordinator, ruter i krajnji uredjaj. [9]

Koordinator je odgovoran za izbor kanala i PAN ID. ZigBee mreža mora imati samo jednog kordinatora, čiji je zadatak: uspostavljanje mreže, dodjeljivanje mrežnih adresa čvorovima, briga za sigurnost i ispravnost razmjene podataka između čvorova. Sav promet se odvija preko ovog čvora i na njemu je implementiran cijeli ZigBee skup protokola. Koordinator započinje kreiranje nove personalne mreže (kreira novi PAN). Nakon što je kreiran PAN, koordinator može dozvoliti ruterima i krajnjim uređajima da se priključe na mrežu. Koordinator može slati i primati RF podatke i može asistirati u rutiranju podataka kroz međusobno povezane, isprepletene (engl. *mesh*) mreže. [9]

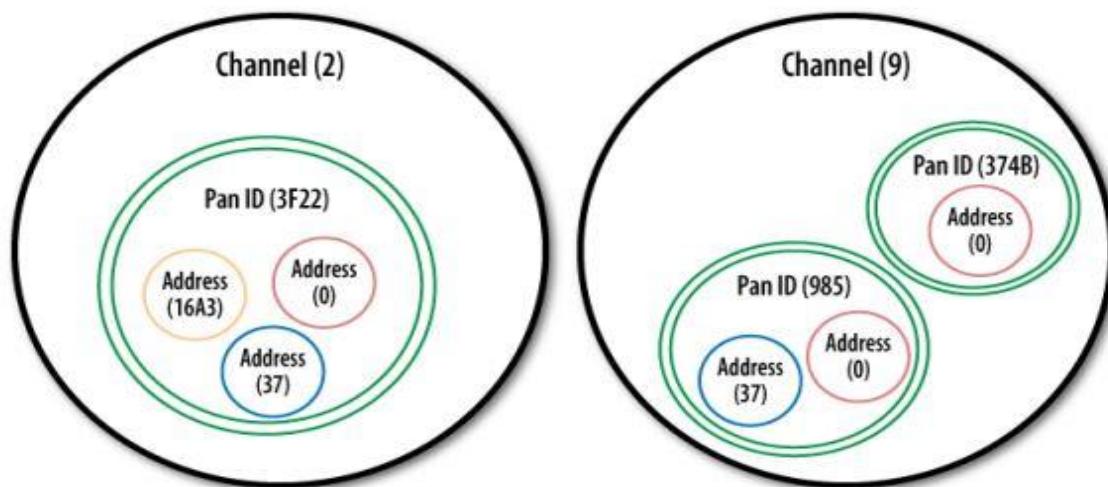


Slika 2. 14 Topologija ZigBee mreža [13]

Ruter nije obavezan uređaj u mreži, njegovo dodavanje u mrežu omogućava povezivanje većeg broja čvorov i samim tim povećavate domet mreže. On prosljeđujući podatke između drugih ruteru, krajnjih uređaja i koordinatora. Ruter se prvo mora priključiti tj. uvezati u ZigBee PAN da bi mogao da obavlja svoju funkciju. Nakon što se uveže on može dozvoliti drugim ruterima i krajnjim uređajima da se priključe u PAN. Ruter takođe može primati i slati RF podatke i može rutirati pakete podataka kroz mrežu. [9]

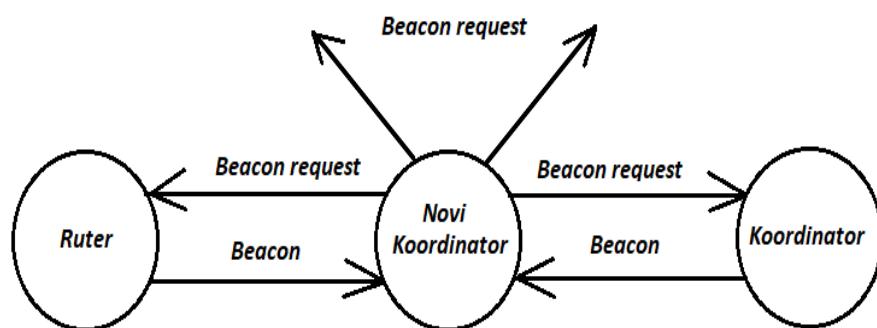
Krajnji uređaj je uređaj koji komunicira sa okolinom. Krajnji uređaj se mora priključiti u mrežu kao i ruter, ali krajnji uređaj ne omogućava drugim uređajima da se priključe na PAN i ne učestvuje u rutiranju podataka kroz mrežu. Ovi uređaji mogu slati i primati RF podatke. Obzirom da ovi uređaji mogu biti u stanju spavanja, ruter ili koordinator koji je dozvolio ovom uređaju da se priključi u mrežu mora sve pakete podataka namjenjenih ovom krajnjem uređaju baferovati i čuvati sve dok se ovaj ne probudi i ne bude u stanju da primi podatke koji su mu namjenjeni. Ruter ili koordinator koji dozvoljava krajnjem uređaju da se veže u mrežu i koji upravlja RF podacima namjenjenim tom krajnjem uređaju, naziva se roditelj krajnjeg uređaja, a krajnji uređaj je dijete. [9]

ZigBee mreža se kreira kada koordinator izabere kanal i PAN ID. Nakon što koordinator starta PAN, ruteri i krajnji uređaji se mogu priključiti u PAN. PAN ID je dakle selektovao koordinator kada je startao PAN a ruteri i krajnji uređaji mogu postati dio tog PAN-a i naslijediti PAN ID koordinatora kada se priključe u mrežu. Svaki ZigBee uređaj ima jedinstvenu 64-bitnu adresu. Korištenjem takve proširene adrese moguće je adresirati bilo koji uređaj u mreži. Nakon što se uređaj prijavlji koordinatoru, on ga upiše u tablicu u memoriji u kojoj se 64-bitnoj adresi pridružuje 16-bitna adresa (PAN ID). Na taj način, uređaji unutar mreže mogu komunicirati pomoću 16-bitnih adresa. Koordinator omogućava i komunikaciju uređaja iz mreža s različitim mrežnim identifikatorima, dakle komunikaciju između dvije neovisne ZigBee mreže. U tom je slučaju uređaj potrebno adresirati pomoću 64 bitne adresе mrežnog identifikatora. Koordinator ima mrežnu adresu "0" prema zadanim postavkama. [9]



Slika 2. 15 Vennov dijagram koji prikazuje kanal, PAN i adresiranje [8]

Obzirom da je koordinator odgovoran za startovanje, odnosno uspostavljanje ZigBee mreže, to svaka ZigBee mreža mora inicijalno imati koordinator. Da bi uspostavio PAN mrežu koordinator izvršava skeniranje da bi ustanovio nivo RF aktivnosti na različitim kanalima (engl. *energy scan*) i da bi otkrio neki eventualni obližnji PAN (engl. *PAN scan*). Na početku koordinator vrši "energy scan" na višestrukim kanalima (frekvencijama) da odredi energetski nivo svakog kanala. Kanali sa prekomjernom energijom se brišu sa njegove liste potencijalnih kanala. Ovo skeniranje, dakle, omogućava koordinatoru da izbjegne kanale sa visokim nivoom energije. Pored "energy scan" na samom početku koordinator vrši i "PAN scan". (Slika 2. 16). Naime, nakon što je kompletiran "energy scan" koordinator skenira preostale tihe kanale (engl. *quit channels*) koje je pronašao u "energy scan" u cilju pronalaženja postojanja susjednih PAN-ova. Da bi otkrio obližnje PAN-ove koordinator šalje broadcast poruku, "beacon request" susjednim koordinatorima ili ruterima. Ukoliko postoje, najbliži koordinator ili ruter će odgovoriti na ovaj beacon request slanjem fara-posebnog okvira, tzv. beacon okvira nazad koordinatoru. Beacon okvir sadrži informacije o njegovom pošiljaocu, uključujući i PAN identifikator i podatak da li uređaj dozvoljava ili ne dozvoljava umrežavanje. PAN skeniranje je poznato i kao aktivno skeniranje (engl. *active scan*) ili beacon skeniranje (engl. *beacon scan*). PAN skeniranje omogućava koordinatoru da detektuje obližnje PAN identifikatore i tako izbjegne eventualno duplicitanje postojećih PAN identifikatora. Kada koordinator kompletira ova skeniranja, on analizira sve pristigne beacon-e i starta mrežu sa neiskorištenim PAN ID i kanalom. Nakon što je mreža formirana, dozvoljeno je ruterima i/ili krajnjim uređajima da joj se priključe. Koordinator zadržava kanal i PAN ID atribut do nestanka napajanja ili reset aktivnosti. Ruteri i krajnji uređaji moraju biti u mogućnosti da otkriju i da se spoje u ZigBee PAN. Da bi to uradili oni moraju prvo, slično kao koordinator na početku formiranja ZigBee PAN mreže, izvršiti PAN skeniranje. Rezultat tog PAN skeniranja je lista beacon-a od obližnjih ZigBee uređaja. Ruter ili krajnji uređaj analiziraju tu listu da pronađu odgovarajući PAN kojem će se priključiti. Zavisno od njihove konfiguracije oni mogu da se priključe bilo kojem ZigBee PAN-u ili samo da se vežu u PAN sa određenim PAN ID. Bez obzira koja opcija je u pitanju oni uvjet moraju pronaći koordinator ili ruter koji će im omogućiti vezivanje. [9]



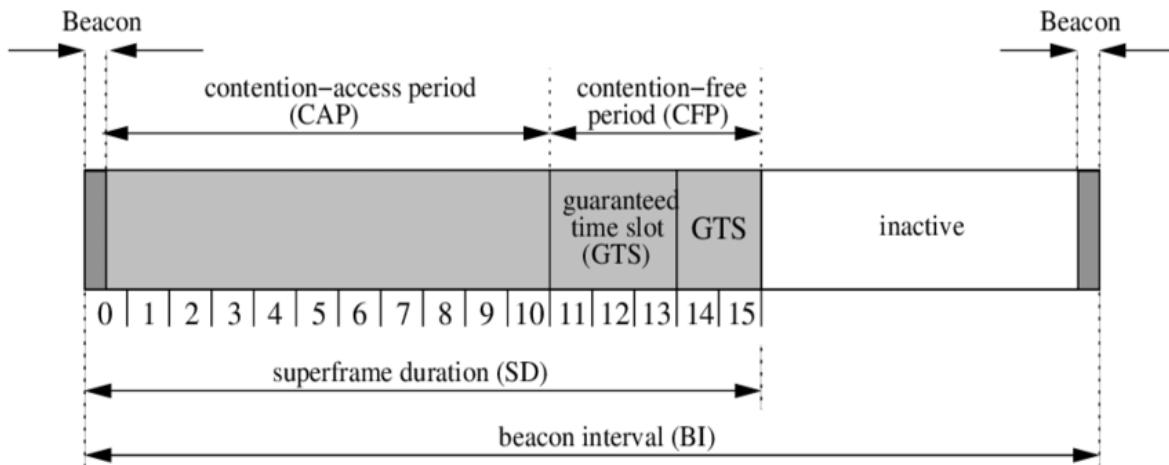
Slika 2. 16 PAN skeniranje

Nakon što uređaji, ruter ili krajnji uređaj, koji se žele spojiti na mrežu pronađu odgovarajući uređaj, koordinator ili ruter, koji im dozvoljava umrežavanje oni pokušavaju da se spoje na mrežu šaljući zahtjev za spajanjem (engl. *association request*) tom uređaju. Koordinator i svi ruteri mogu omogućiti novim ruterima i krajnjim uređajima da se vežu na njih. Da li će ili ne, pojedinačni koordinator ili ruter to dozvoliti zavisi od dvije stvari [9]:

- da li mu je to dozvoljeno (definisano „permit-joining“ atributima)
- broj „child“ krajnjih uređaja koje već ima

Koordinator i svi ruteri imaju tzv "permit-joining" attribute. Ovi atributi na koordinatoru i svim spojenim ruterima mogu biti konfigurisani da: dozvole spajanja, dozvole spajanja na kratko vrijeme i da ne dozvoljavaju dodatna spajanja. Dakle da bi se novi uređaj spojio na mrežu mora pronaći uređaj, koordinator ili ruter, koji ima vrijednost "permit-joining" atributa koja to dozvoljava. Kako se krajnji uređaj oslanja, na to da će njegov roditelj (engl. *parent*) ruter ili koordinator, prenijeti njemu namjenjene pristigne RF pakete, jasno je da koordinator ili ruter mogu podržati određen, konačan broj podređenih (engl. *child*) krajnjih uređaja. Nakon što postigne taj broj krajnjih uređaja, određeni ruter ili koordinator ne dozvoljava spajanje novih uređaja. Ako je mreža zaštićena odnosno opcija "security" omogućena, koordinator u startu koristi 128-bitni AES ključ za šifriranje. Samo uređaji koji imaju isti sigurnosni ključ (engl. *security key*) mogu komunicirati u mreži. Ruteri i krajnji uređaji koji žele da se priključe u takvu zaštićenu mrežu moraju posjedovati važeći sigurnosni ključ. Sigurnosni ključ može biti postavljen u toku same konfiguracije i/ili u samom procesu spajanja na mrežu (engl. *over-the-air*). [9]

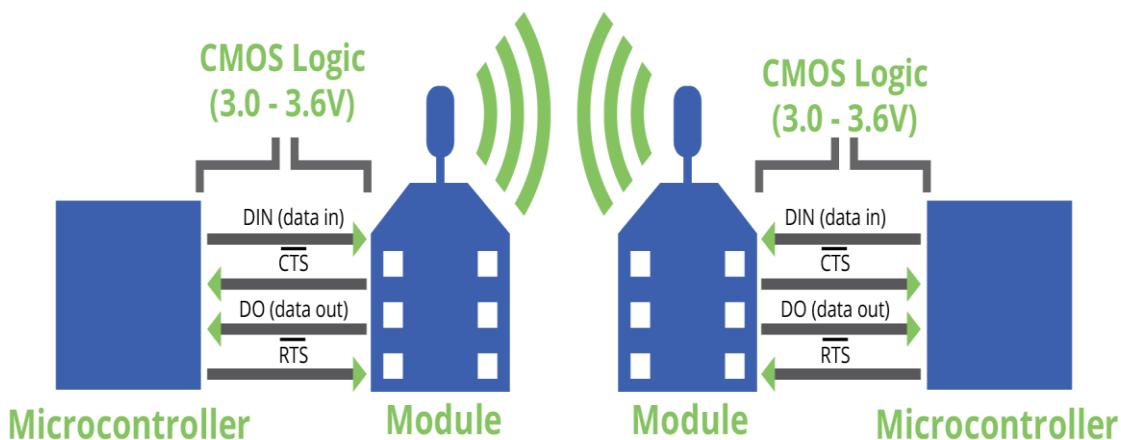
Nakon što se krajnji uređaji prijave PAN koordinatoru i na taj se način uspostavi mreža, oni se "natječu" za korištenje prenosnog medija prema protokolu CSMA/CA. Takav način rada naziva se "non-beacon" način rada. Drugim riječima, PAN koordinator čitavo vrijeme osluškuje zahtjeve krajnjih uređaja. Očigledno je da mora biti čitavo vrijeme aktivan te se zbog toga ušteda energije svodi na uštedu u krajnjim čvorovima. Ovaj problem rješava "beacon" način rada. Za pristup mediju upotrebljava se naziv "unslotted" CSMA/CA. Ukoliko uređaji rade u "beacon" načinu rada moraju imati uključenu opciju korištenja nadzornog-okvira (engl. *superframe*). Nadzorni okvir (Slika 2. 17) podijeljen je na 16 jednakih vremenskih priključaka. Na prvom vremenskom priključku šalje se "beacon" okvir koji služi za dojavu upravljačkih postavki do krajnjih uređaja. Kao što nadzorni okvir propisuje vrijeme kada su krajnji uređaji pridruženi PAN koordinatoru aktivni ili neaktivni (šaljući im "beacon" okvir u kojem se nalaze sve informacije) jednako tako nadzorni okvir definiše vrijeme kada je PAN koordinator aktivan ili neaktivan. Dakle, unutar vremena trajanja 16 vremenskih priključaka, koordinator može imati neaktivne dijelove tokom kojih ne komunicira sa svojom mrežom i tada se nalazi u stanju "spavanja" te mu je potrošnja svedena na minimum. Vremenski raspon od početka nadzornog okvira pa do početka sljedećeg nadzornog okvira je vrijeme tokom kojeg krajnji uređaji mogu pristupati prenosnom mediju. Pristup prenosnom mediju od strane krajnjih čvorova odvija se putem CSMA/CA protokola, a period u kojem uređaji imaju mogućnost konkurentnog pristupa kanalu naziva se period konkurentnog pristupa – CAP (engl. *Contention Access Period*). Svakom uređaju može biti dodijeljen poseban vremenski priključak, poznat kao garantovani vremenski priključak – GTS (engl. *Guaranteed Time Slot*). [11]



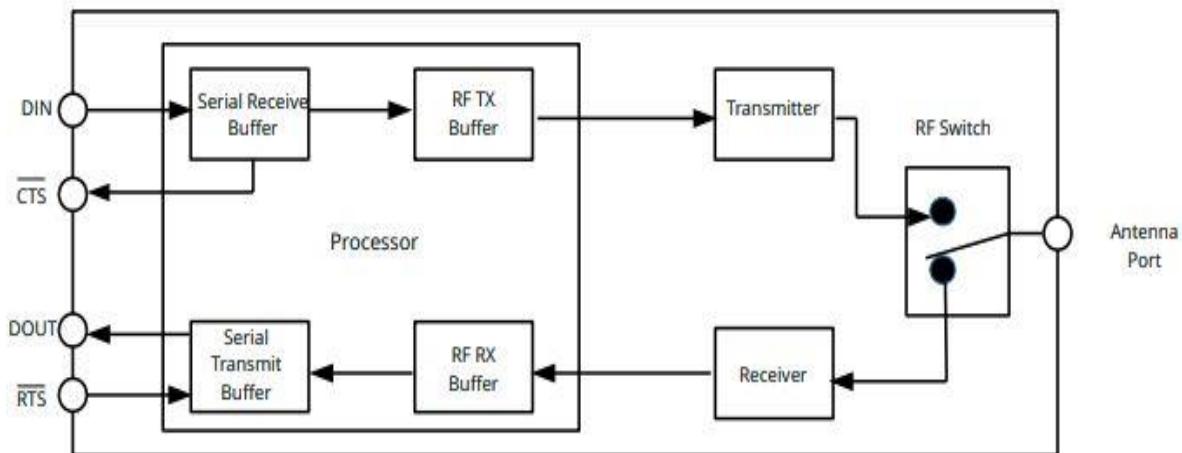
Slika 2. 17 Struktura nadzornog okvira [14]

Za komunikaciju sa host uređajem XBee/XBee PRO RF modul koristi asinhroni serijski interfejs. Koristeći svoj serijski port, modul može komunicirati sa bilo kojim kompatibilnim univerzalnim asinhronim prijemnikom i predajnikom – UART (engl. *Universal Asynchronous Receiver-Transmitter*). Uređaj koji ima UART interfejs može se spojiti direktno na odgovarajuće pinove RF modula, kao što je prikazano na slici 2. 18. Serijska komunikacija dakle, zavisi od dva UART-a (UART mikrokontrolera, računara ili host uređaja i UART-a RF modula) i mora biti konfigurisana sa odgovarajućim kompatibilnim vrijednostima (brzina prenosa, paritet, start bit, stop i data bit) kako bi mogla uspješno komunicirati. [9]

XBee moduli imaju male bafer-e za skladištenje primljenih serijskih i RF podataka, kako je prikazano na slici 2. 19. Bafer za prijem serijskih podataka (engl. *serial receive buffer*) zadržava dolazne serijske informacije dok ne budu spremne za obradu. S druge strane, bafer za slanje serijskih podataka (engl. *serial transmit buffer*) prikuplja podatke primljene putem RF veze, koje će potom poslati kroz UART interfejs. [9]



Slika 2. 18 Data-flow dijagram UART interfejs [15]



Slika 2. 19 Unutrašnji tok podataka [15]

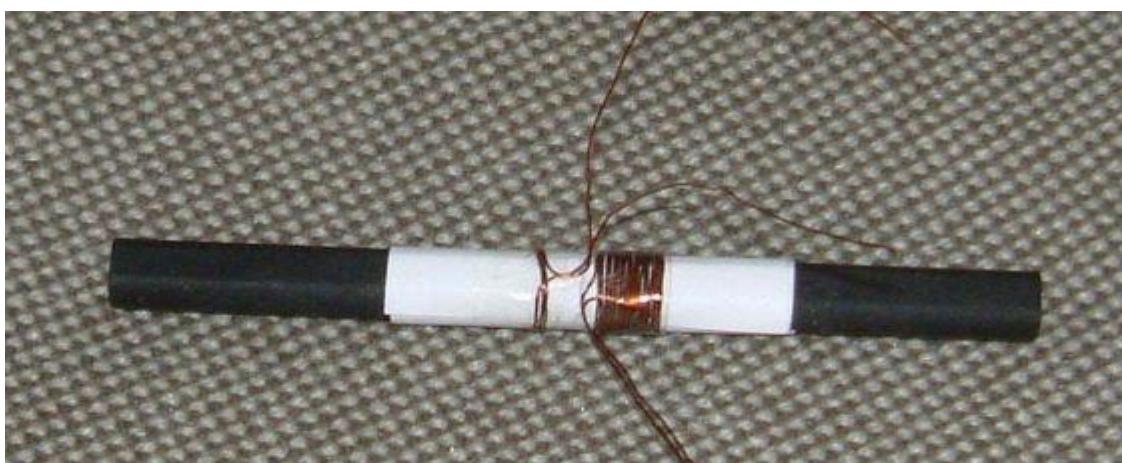
Kada serijski podaci uđu u RF modul putem "DIN" pina (pin 3), podaci se pohranjuju u bafer za prijem serijskih podataka dok ne budu obrađeni. U određenim uslovima, modul možda neće moći odmah obraditi podatke u baferu za prijem serijskih podataka. Ako se modulu pošalju velike količine serijskih podataka koje bi mogle preopteretiti bafer za prijem, novi podaci će biti odbačeni. Ako se koristi UART, ovo se može izbjegići tako što će host uređaj poštovati CTS (engl. *Clear To Send*) protokol za upravljanje protokom. Hardverska kontrola toka podataka omogućena je CTS signalom. Kada baferu za prijem serijskih podataka preostane 17 bajtova da postane pun, po defaultu se signalizira hostu (CTS signal u neaktivnom stanju - 1) da prestane sa slanjem podataka. Ponovno slanje podataka je moguće kada je dostupno 34 bajta memorije baferu za serijski prijem (CTS signal aktivno stanje - 0). Da bi se izbjegle ovakve situacije, potrebno je ili slati poruke manje od veličine bafera za serijski prijem (202 bajta) ili smanjiti brzinu prenosa definisanu BD (engl. *Interface Data Rate*) parametrom. Kada se RF podaci prime, oni se prebacuju u bafer za slanje serijskih podataka i šalju preko UART-a. Ako bafer za slanje serijskih podataka postane pun tada se cijeli RF paket podataka odbacuje. Kada podaci stignu brže nego što mogu biti obrađeni i poslani preko serijskog porta, postoji mogućnost da se podaci izgube. Ako je kontrola toka podataka preko RTS signala omogućena, podaci u serijskom predajnom baferu neće biti poslati preko "DOUT" pina sve dok je RTS neaktivan (postavljen na visok nivo). Moguća situacija kada može doći do toga da bafer za serijski prenos podataka postane pun i da se desi prekoračenje kapaciteta (engl. *overflow*) je kada je brzina RF prenosa veća od brzine prenosa interfejsa modula, tj. kada modul prima podatke od modula koji vrši prenos većom brzinom od one kojom modul šalje podatke na host uređaj. Po deafultu XBee/XBee PRO RF moduli rade u transparentnom modu (engl. *transparent mode*). Kada rade u ovom modu svi UART podaci pristigli "DIN" pinom su u redu čekanja za RF prenos. S druge strane, svi RF podaci, podaci pristigli RF prenosom, se šalju van koristeći "DOUT" pin. Alternativa, defaultnom transparentnom modu rada je aplikativni programski interfejs – API (engl. *Application Programming Interface*) mod. Kada modul radi u API modu svi podaci koji ulaze ili napuštaju modul sadrže okvire (engl. *frames*) koji definišu operacije ili događaje (engl. *events*). [15]

2.1.3 Radiogoniometar

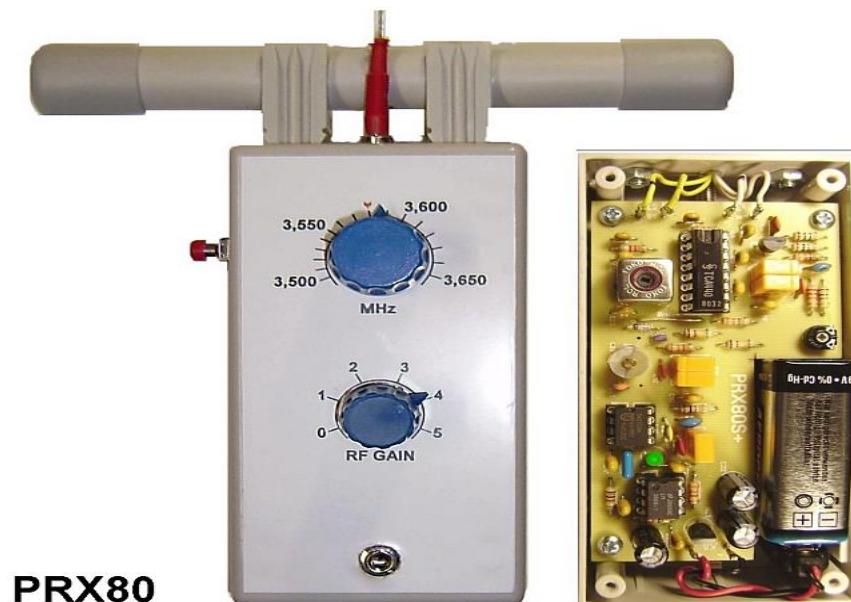
Radiogoniometar je uređaj koji koristimo za određivanje pravca dolaska radio signala. Ključni je alat u radioamaterskim aktivnostima poput amaterske radiogoniometrije i oslanja se na precizno otkrivanje pravca dolaznog signala, omogućavajući korisniku da odredi lokaciju predajnika. Radiogoniometri su posebno dizajnirani da odgovore na izazove praćenja signala na određenim frekvencijama, kao što je 3.5 MHz, koja se često koristi u takmičarskom i tehničkom kontekstu. Ovaj uređaj omogućava precizno praćenje radiopredajnika i igra značajnu ulogu kako u amaterskoj radiogoniometriji tako i u različitim poljima, uključujući komunikacijske sisteme i potragu i spašavanje.

Radiogoniometar temelji se na prijemniku amplitudne modulacije u integrisanom kolu TCA 440, koji je ujedno i glavni dio sklopa. On podržava prijem frekvencija do 50 MHz. Radiogoniometar je podešen da radi u rasponu u kojem rade i predajnici od 3.5 MHz do 3.6 MHz. Izlazni stepen čini audio pojačalo LM386. Na njega se spajaju slušalice otpora od 30 do 100 oma. Sklop se napaja baterijom od 9 V koja zahvaljujući maloj potrošnji sklopa, relativno dugo traje. [16]

Antena je neophodni element svakog radiogoniometra od koje zavisi tačnost goniometrisanja. Antena prima energiju radio talasa i pretvara je u struju visoke frekvencije. Na opsegu od 80 m (3.5 MHz) obično se koriste okvirne i feritne antene. Feritna antena se sastoji od male zavojnice namotane na fertitni štapić prečnika 1 cm i dužine oko 14 cm. Zavojnica je izrađena od lakirane bakrene žice prečnika 0.3 mm. Na slici 2. 20 su prikazane zavojnice L1 i L2. Zavojnica L1, koja predstavlja glavnu antenu, sastoji se od 20 zavoja, dok zavojnica L2 ima 3 zavoja od lakirane bakrene žice. Između njih je razmak od 5 mm. Feritni štapić na kojem su namotane ove zavojnice postavljen u plastičnu cijev koja ga štiti od mehaničkih oštećenja, na sredini feritne antene dodaje se aluminijski "L" profil koji služi kao pomoćna antena za određivanje smjera. Ova namotana zavojnica treba biti zaštićena od kontakta sa goniometristom (kako bi se sprječilo oštećenje) i od vibracija (kako bi se očuvala stabilnost frekvencije). [16]



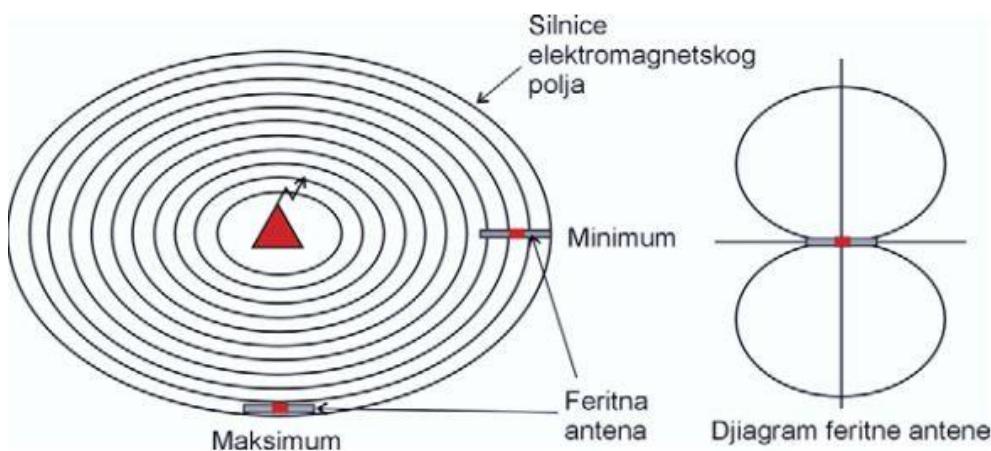
Slika 2. 20 Feritni štapić sa namotajima [16]



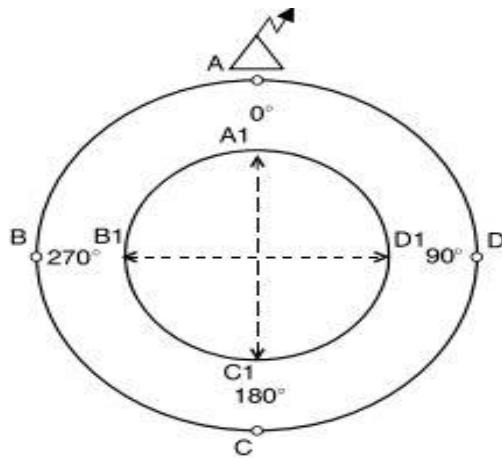
Slika 2. 21 Radiogoniometar [17]

U radiogoniometru, feritni štapić je smješten unutar plastične cijevi promjera 13 mm, kakva se koristi za odvod kondenza iz klima uređaja. Na krajevima se cijev začepi plastičnim čepovima. Za sprječavanje vibracija štapića unutar plastične cijevi možemo koristiti pjenu, silikon, spužvu ili neku drugu metodu. U radiogoniometru se nalaze dva potenciometra. Jedan služi za promjenu frekvencije, a drugi za promjenu RF pojačanja. [16]

Jačina signala ovisi o položaju antene u odnosu na predajnik. Okrenemo li feritni štap bočno prema predajniku, napon je najveći i čujnost najbolja (maksimum). Okrene li se os feritnog štapa prema predajniku, napon je najmanji i čujnost najslabija (minimum). I ovdje se pojavljuju dva minimuma i dva maksimuma u krugu od 360 stepeni. [18]

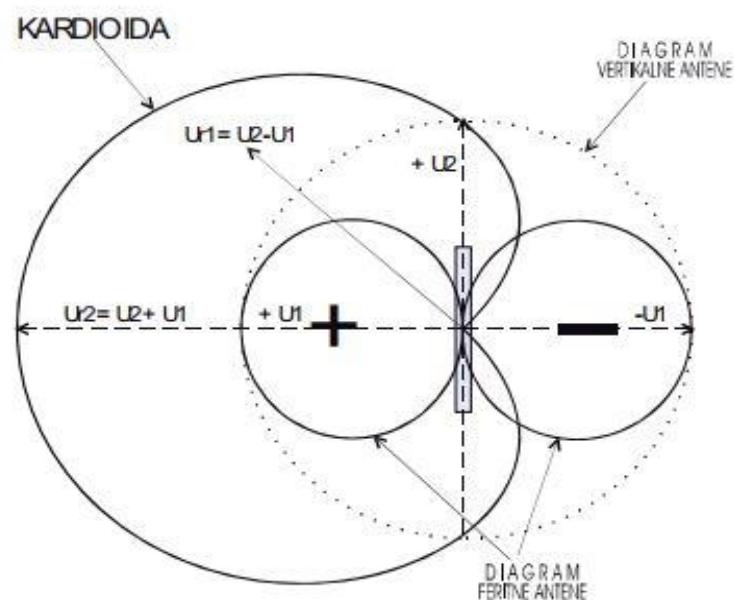


Slika 2. 22 Dijagram feritne antene [18]



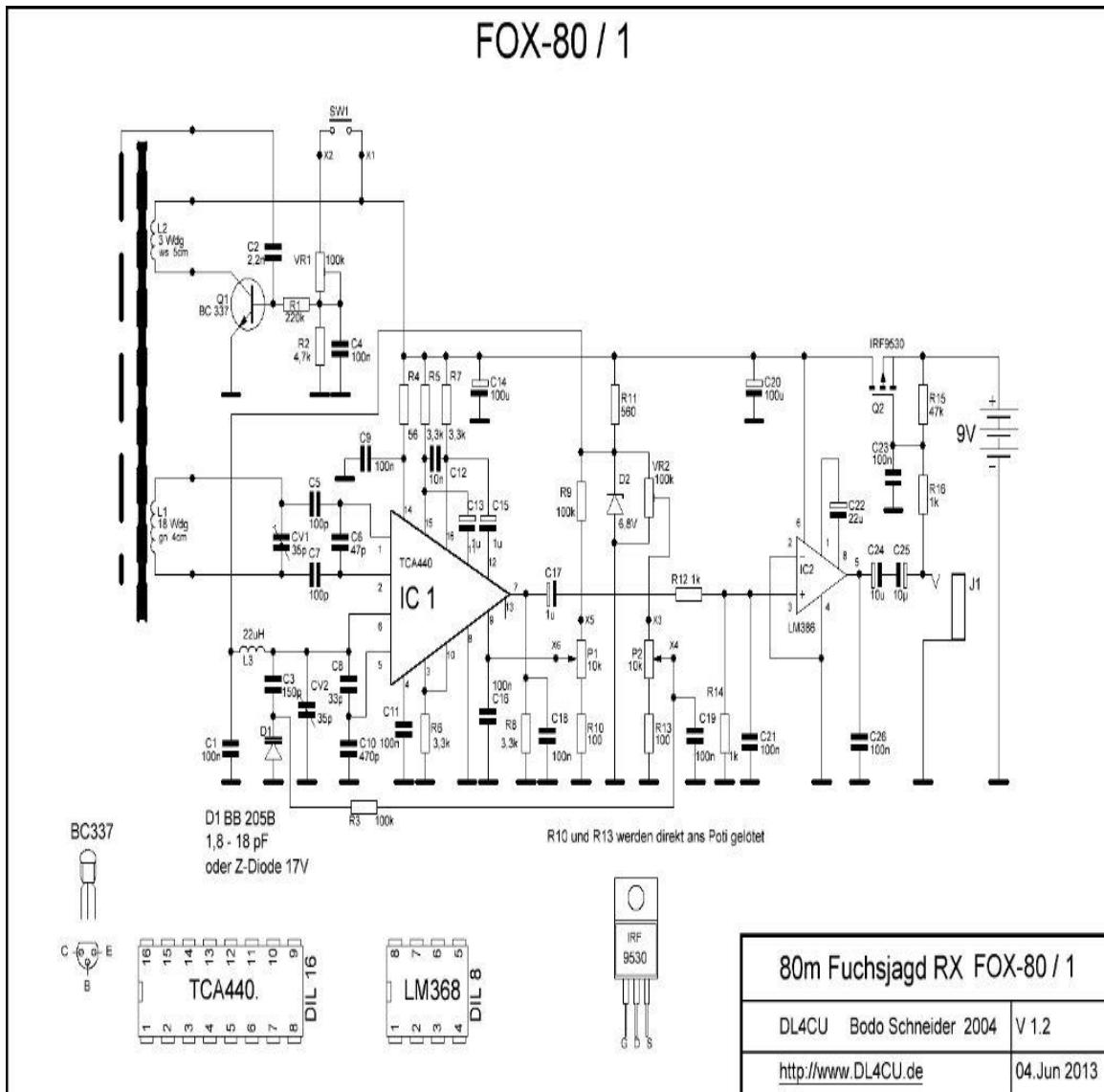
Slika 2. 23 Dijagram (vertikalne) štap antene [18]

Da bi odredili položaj predajnika potrebno je izvršiti goniometrisanje iz dvije tačke međusobno što udaljenije, a ugao koji zatvaraju pravci između tih tačaka i predajnika ne smije biti ni preveliki ni premalen (optimalno 90 stepeni). Na taj način se dobije presjek pravaca, kojim je određen položaj predajnika. Budući da ta metoda zahtijeva mnogo vremena (mnogo trčanja i precizan rad s kartom), u praksi se smjer predajnika određuje jednim mjeranjem, ali kombinacijom dvije antene: usmjerene (okvirne ili feritne) i neusmjerene (štap). Feritna antena je stalno spojena na prijemnik radiogoniometra. Štap antena je pomoćna i uključuje se samo prilikom određivanja smjera. Dijagram štap antene je kružnica. [18]



Slika 2. 24 Kardioda [18]

Na pomoćnoj (štap) anteni inducira se napon jednake veličine kao i na usmjerenoj anteni na maksimumu prijema. Ako se obje antene priključe istovremeno na prijemnik, njihovi dijagrami se pretvaraju u novi, sročili dijagram (kardiodu) pomoću kojeg se može odrediti smjer iz kojeg emitira predajnik. Određivanje smjera na opsegu 80 m (3.5 MHz) vrši se prema minimumu signala, budući da je na minimumu dovoljan mali zaokret antene da bi se uočila razlika, dok je kod maksimuma potreban veći zaokret. Osim toga, ljudsko uho lakše raspoznaće promjene slabih signala nego jakih. [18]



Slika 2. 25 Šematski dijagram radiogoniometra [19]

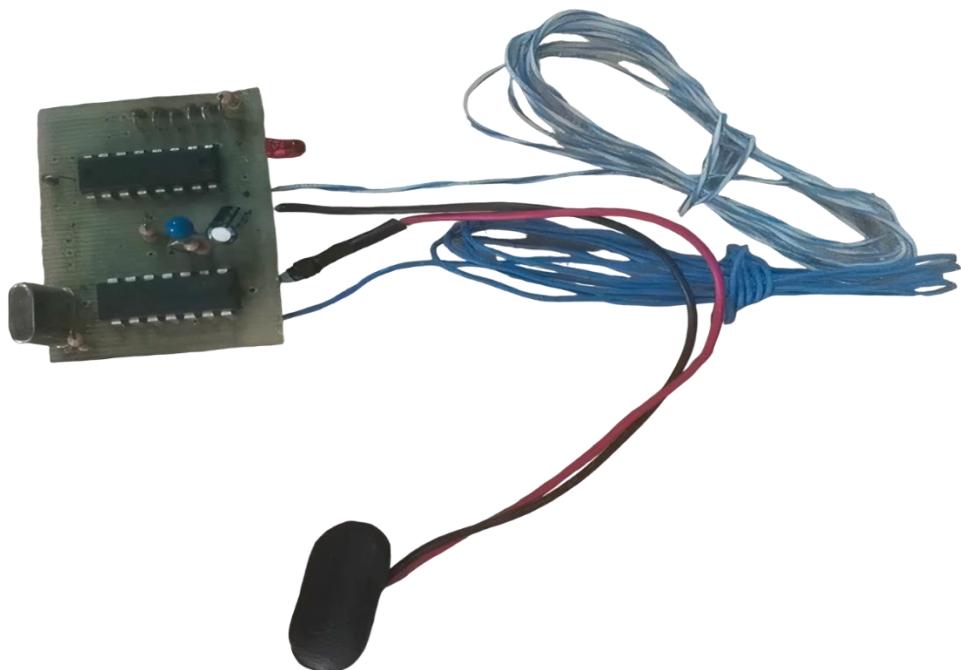
2.1.4 Radiopredajnik

Radiopredajnik je uređaj koji emituje radio talase kako bi prenosio informacije na određenoj frekvenciji. Njegova osnovna funkcija je pretvaranje električnih signala u radio talase, koji se zatim šalju kroz antenu i prenose na udaljene prijemnike. U amaterskoj radiogoniometriji, radiopredajnici igraju ključnu ulogu kao izvori signala koje takmičari pokušavaju locirati pomoću radiogoniometara. Amaterska radiogoniometrija se često oslanja na predajnike koji rade na specifičnim frekvencijama, poput 3.5 MHz, kako bi omogućili izazovno, ali precizno praćenje signala. Dakle, radiopredajnici u amaterskoj radiogoniometriji služe kao "meta" za takmičare, omogućavajući im da koriste svoje vještine u određivanju pravca i lokacije izvora radio talasa.

Za ARDF takmičenja na opsegu od 80 metara potreban je predajnik koji ima sljedeće karakteristike [20]:

- Snagu od 1 do 5 W
- Visoku efikasnost (napajanje baterijama)
- Jednostavnost i pouzdanost
- Dovoljno potiskivanje harmonika
- Konstantnu snagu za impedanse antene između 20 i 100 oma

Sa naponom napajanja od 12 do 13 volti, ovaj predajnik ima izlaznu snagu od oko 3 W. Povećanjem napona napajanja pojačavača – PA (engl. *Power Amplifier*) na 30 V, može se postići izlazna snaga od 20 W. [20]

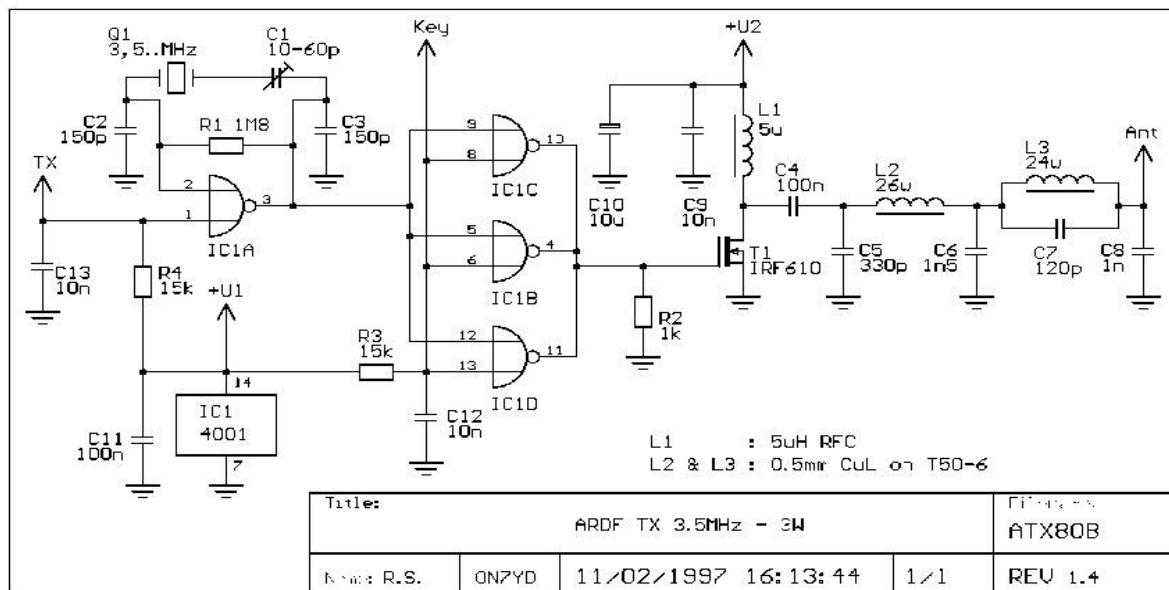


Slika 2. 26 Radiopredajnik

Radiopredajnik funkcioniše tako što uzima signal koji generiše oscilator, pojačava ga, i zatim ga šalje kroz antenu u obliku radio talasa. Predajnik se sastoji od tri glavna dijela [20]:

- Oscilator je ključni dio koji stvara osnovni signal na određenoj frekvenciji, a frekvencija ovog signala se može malo prilagoditi pomoću kondenzatora C1. Oscilator se aktivira povlačenjem signala "TX" na uzemljenje, a frekvencijski opseg može biti od 3.58 MHz do 3.63 MHz u zavisnosti od komponenti. Stabilnost frekvencije zavisi od kvaliteta upotrijebljenih kondenzatora (C2 i C3), dok zamjena kristala Q1 keramičkim rezonatorom omogućava širi opseg podešavanja, ali uz smanjenu stabilnost. Da bi se frekvencija dodatno precizno podešila ispod 3.6 MHz, koristi se mali induktor u seriji sa C1. Takođe, kako bi se obezbijedila stabilnost u različitim uslovima, preporučuje se adekvatno hlađenje oscilatora.
- Drajver je dio predajnika koji služi da pojača signal koji dolazi iz oscilatora i pripremi ga za pojačavač snage. Tri preostale kapije (engl. gate) IC1 su povezane paralelno i djeluju kao međuskladište između oscilatora i pojačavača snage, obezbjeđujući dovoljno struje za njegov rad. Aktiviranje predajnika se vrši povlačenjem veze "KEY" na uzemljenje. Važno je napomenuti da se predajnik ne smije aktivirati ako oscilator nije uključen, jer to može izazvati kratki spoj koji može oštetiti napajanje i druge komponente.
- Pojačavač snage koristi FET tranzistor koji je veoma izdržljiv i brzo prelazi u klase C, pružajući efikasnost od 60% do 70%. On osigurava stabilnu izlaznu snagu za impedanse antene između 20 i 100 omu.

Nakon što signal prođe kroz ove faze, predajnik ga šalje kroz antenu u obliku radio talasa. Antena je ključna jer pretvara električni signal u elektromagnetski talas koji može putovati kroz vazduh. Predajnik može raditi na različitim naponskim nivoima, pri čemu se sa višim naponom postiže veća snaga signala. [20]



Slika 2. 27 Šematski dijagram radiopredajnika [20]

2.1.5 Pasivni infracrveni senzor pokreta

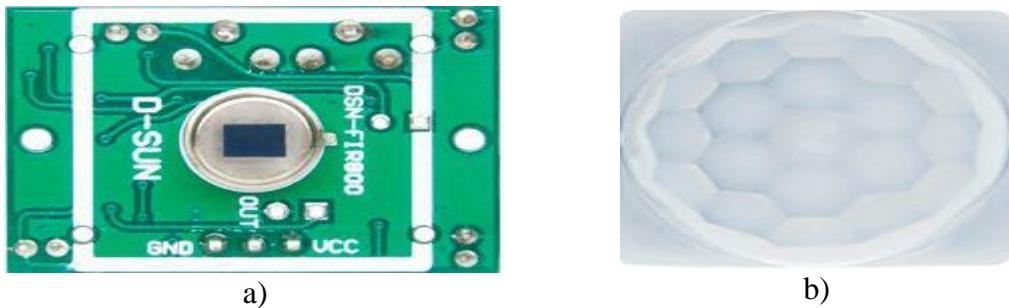
Pasivni infracrveni – PIR (engl. *Passive Infrared*) senzor je uređaj koji detektuje promjene u infracrvenom zračenju, što ga čini veoma korisnim za prepoznavanje prisustva toplokrvnih organizama, poput ljudi i životinja. Ljudi zrače toplotu koja mijenja temperaturni nivo prostorije kada u nju uđu. Ova promjena temperature može se iskoristiti za pokretanje različitih uređaja, poput alarma za provalnike ili sistema za štednju energije. [21]

Za razliku od drugih senzora koji emituju infracrveno svjetlo i detektuju njegovu refleksiju, PIR senzor ne emituje ništa. On samo mjeri količinu infracrvene energije u prostoriji i reaguje na nagle promjene u njoj, kao što su one koje se dešavaju kada toplokrvno biće, poput takmičara u ARDF takmičenju, uđe u zonu detekcije. Infracrvena energija je oblik toplotnog zračenja koje nije vidljivo golim okom, ali ga PIR senzor može prepoznati. U našem projektu koristimo PIR senzor, konkretno model HC-SR501, kako bismo detektovali prisustvo takmičara koji se približava predajniku.

Bilo koji objekat s temperaturom iznad apsolutne nule emituje toplotnu energiju u obliku zračenja. To zračenje nije vidljivo ljudskom oku jer se zrači na infracrvenim talasnim dužinama, ispod spektra koji ljudi mogu da vide. Mjerenje infracrvene energije nije isto što i mjerenje temperature. Mjerenja temperature zavise od toplotne provodljivosti, tako da kada osoba uđe u prostoriju, ona ne povećava odmah temperaturu prostorije (osim ako nije u plamenu). Međutim, ona ima jedinstven IR „potpis“ (engl. *footprint*) ili „trag“ zbog svoje tjelesne temperature, i upravo taj potpis PIR senzor traži. [21]



Slika 2. 28 Pasivni infracrveni senzor HC-SR501 [21]

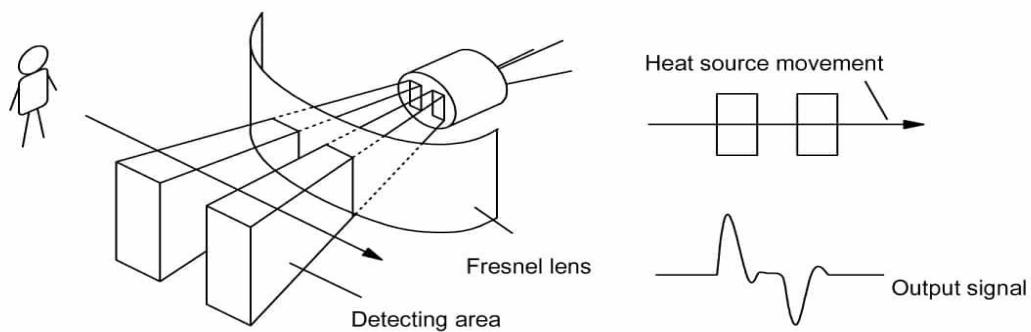


Slika 2. 29 a) Piroelektrični senzor b) Fresnelova leća [22]

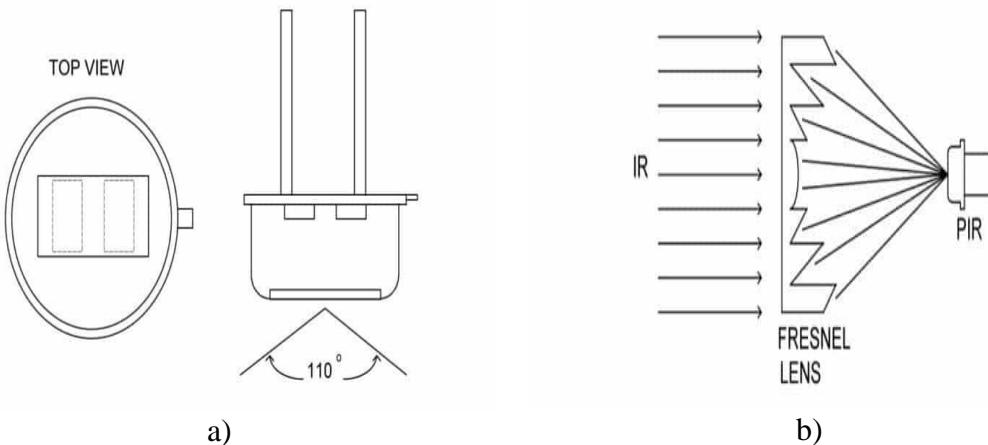
Pasivni infracrveni senzor HC-SR501 se sastoji od dva glavna dijela koja su prikazana iznad na slici 2. 29 [22]:

- Piroelektričnog senzora (engl. *pyroelectric sensor*), okrugli metalni dio sa pravougaonim kristalom u sredini.
- Posebne leće koja se zove Fresnelova leća (engl. *Fresnel lenses*), a koja fokusira infracrvene signale na piroelektrični senzor.

Piroelektrični senzor se sastoji od prozora sa dva pravougaona otvora i izrađen je od materijala (obično obloženog silicijuma) koji omogućava prolazak infracrvenog zračenja. Iza prozora se nalaze dvije odvojene elektrode za detekciju infracrvenog zračenja, jedna zadužena za stvaranje pozitivnog izlaza, a druga za stvaranje negativnog izlaza. Elektrode su povezane tako da se međusobno poništavaju. To je zato što tražimo promjene u nivou infracrvenog zračenja, a ne ambijentalne nivoe. Dakle, kada jedna polovina vidi više ili manje infracrvenog zračenja od druge, dobijamo izlazni signal. Kada nema kretanja oko senzora, oba otvora detektuju istu količinu infracrvenog zračenja, što rezultira nulom izlaznog signala. Međutim, kada toplo tijelo poput čovjeka ili životinje prođe pored senzora, prvo pokriva jednu polovinu senzora. Ovo uzrokuje pozitivnu diferencijalnu promjenu između dvije polovice. Kada toplo tijelo prekrije drugu polovinu senzora (napusti područje detekcije), događa se suprotno i senzor proizvodi negativnu diferencijalnu promjenu. Čitanjem ove promjene napona, detektuje se kretanje. [22]



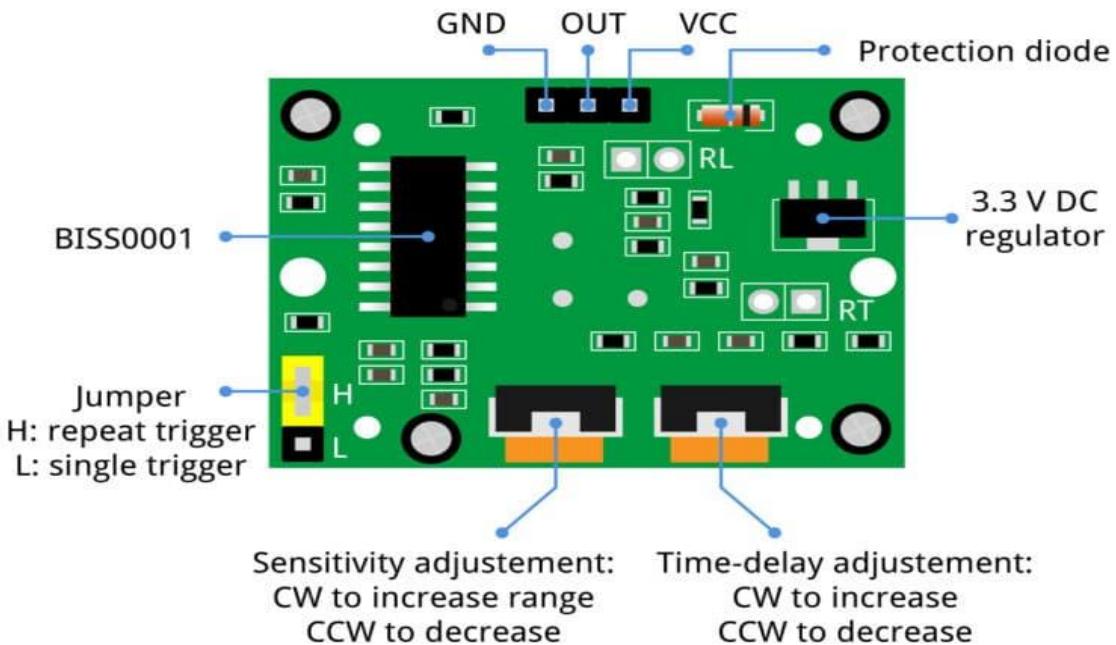
Slika 2. 30 Diferencijalno detektovanje infracrvenih promjena na PIR senzoru [22]



Slika 2. 31 a) IR pokrivenost b) Prelamanje IR svjetlosti kroz Fresnel leću [22]

Fresnelova leća značajno doprinosi povećanju dometa i vidnog polja PIR senzora. Iako se može činiti da leća ne igra značajnu ulogu, njen dizajn je ključan za efikasnost senzora. Leća se sastoji od serije koncentričnih utora u plastičnom materijalu, koji djeluju kao pojedinačne površine za refrakciju svjetlosti, usmeravajući paralelne svjetlosne zrake ka piroelektričnom senzoru. Da bi se dodatno povećao domet i vidno polje PIR senzora, leća je podijeljena na nekoliko fasetnih sekcija, pri čemu je svaka sekcija zasebna Fresnelova leća. HC-SR501 ima podešavanje osjetljivosti koje mu omogućava da detektuje pokret na udaljenosti od 3 do 7 metara, što je dovoljno da pokrije prostor prosječne veličine. Njegov izlaz se može podešiti da ostane u visokom stanju u trajanju od tri sekunde do pet minuta. Senzor ima ugrađeni regulator napona, tako da se može napajati bilo kojim jednosmjernim naponom u rasponu od 4.5 V do 20 V i troši minimalnu količinu struje od 2 mA. U srcu modula nalazi se pasivni infracrveni čip IC – BISS0001. Zbog imuniteta na šumove koji pruža, BISS0001 je jedan od najstabilnijih dostupnih PIR čipova. Ovaj čip uzima izlazni signal sa piroelektričnog senzora i vrši manju obradu tog signala kako bi emitovao digitalni izlazni impuls. Modul dolazi sa zaštitnom diodom (poznatom i kao sigurnosna dioda) koja štiti modul od inverzne struje i napona. PIR senzor ima potenciometar na zadnjoj strani za podešavanje osjetljivosti. Ovaj potenciometar postavlja maksimalni domet detekcije. Osjetljivost se može podešiti u rasponu od otprilike 3 metra do 7 metara. Okretanjem potenciometra u smjeru kazaljke na satu povećavate osjetljivost, a samim tim i domet, dok se suprotnim smjerom smanjuje. Na zadnjoj strani PIR senzora nalazi se još jedan potenciometar za podešavanje vremenskog kašnjenja. Ovaj potenciometar postavlja koliko dugo će izlaz ostati u visokom stanju nakon što se detektuje kretanje. Može se podešavati u rasponu od 1 sekunde do otprilike 3 minuta. Okretanjem potenciometra u smjeru kazaljke na satu povećava se kašnjenje, dok okretanje u suprotnom smjeru smanjuje kašnjenje. Postoje dva režima okidanja koji određuju kako će senzor reagovati kada se detektuje kretanje [22]:

- Jednokratni režim okidanja (engl *Single Trigger Mode*)
- Višekratni režim okidanja (engl. *Multiple Trigger Mode*)



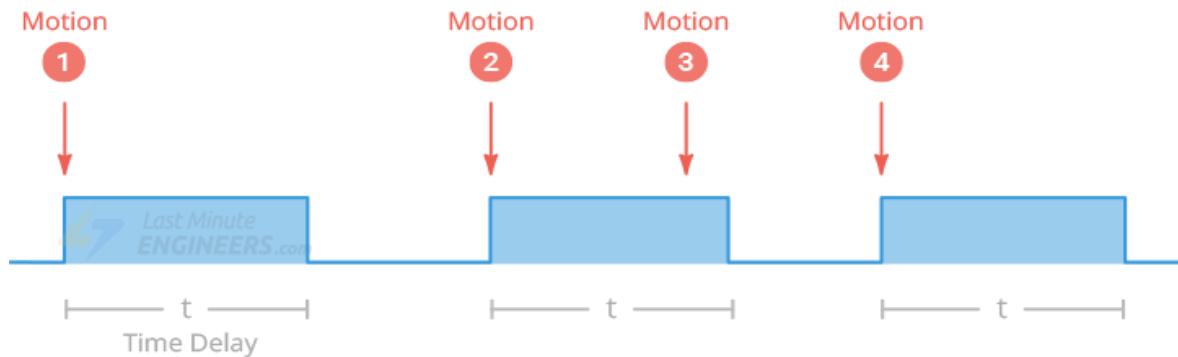
Slika 2. 32 Raspored komponenti na ploči HC-SR501 [22]

Ploča dolazi sa “berg jumperom” (specifična vrsta prespojka koja se koristi za mijenjanje postavki na ploči) koji vam omogućava da izaberete jedan od dva režima [22]:

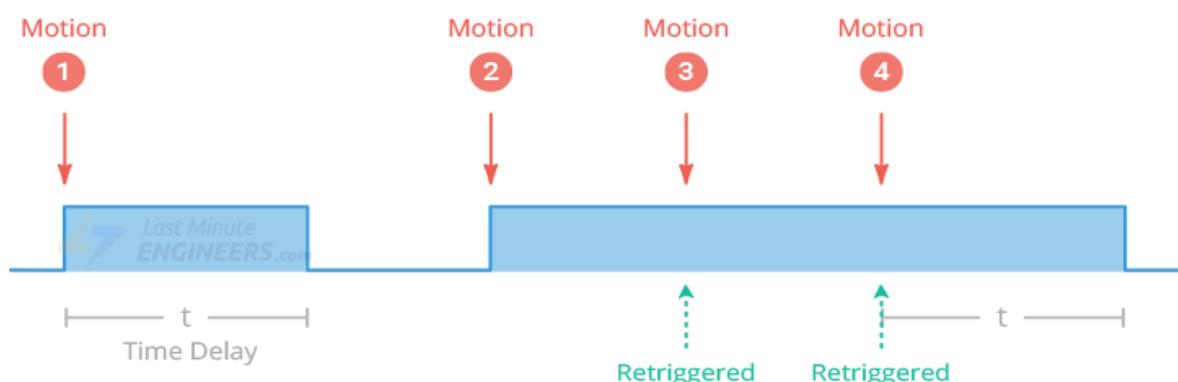
- L – Odabir ove opcije postavlja režim pojedinačnog okidanja. U ovom režimu, izlaz ide na “HIGH” čim se detektuje kretanje i ostaje “HIGH” tokom perioda koji određuje potenciometar za podešavanje kašnjenja. Dalje detekcije su blokirane dok izlaz ne pređe na “LOW” na kraju kašnjenja. Ako se kretanje i dalje detektuje, izlaz će ponovo preći na “HIGH”. Kao što se može vidjeti na slici 2. 33, kretanje 3 je potpuno ignorisano.
- H – Izborom ove opcije postavlja se režim višestrukog okidanja. U ovom režimu, izlaz ide u “HIGH” stanje čim se detektuje pokret i ostaje u “HIGH” stanju tokom perioda koji je određen potenciometrom za vremensko kašnjenje. Za razliku od režima sa jednim okidanjem, dalja detekcija nije blokirana, pa se vremensko odlaganje restartuje svaki put kada se detektuje pokret. Kada se pokret zaustavi, izlaz se vraća u “LOW” stanje tek nakon što prođe vrijeme kašnjenja. Zbog toga se ovaj režim naziva režim višestrukog okidanja, prikazan na slici 2. 34.

HC-SR501 ima 3-pinski konektor za povezivanje sa spoljnim svijetom. Povezivanja su sljedeća [22]:

- “VCC” – Ovo je pozitivni DC naponski ulaz od 4.5 do 20 VDC.
- “OUT” – Logički izlaz od 3.3 V. “LOW” označava da detekcija nije izvršena, dok “HIGH” znači da je neko detektovan.
- “GND” – Priključak za uzemljenje.

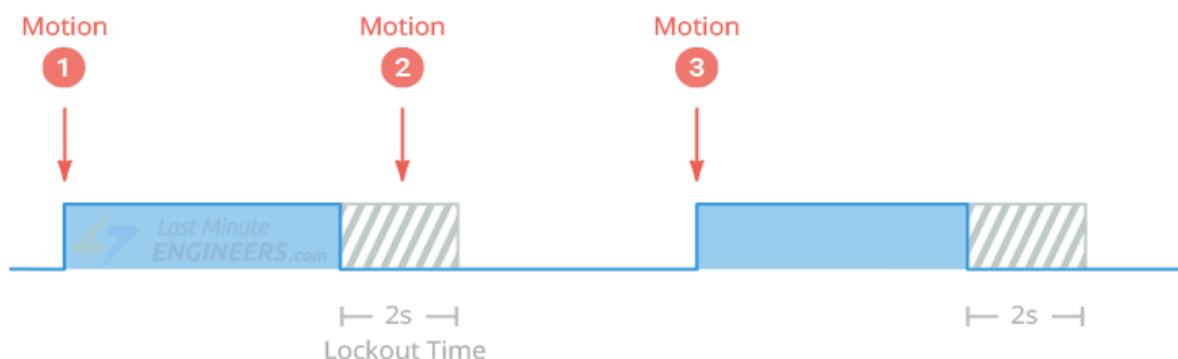


Slika 2. 33 Jednokratni režim okidanja [22]



Slika 2. 34 Višestruki režim okidanja [22]

Pri dizajniranju sistema zasnovanog na HC-SR501, treba imati na umu sljedeće vremenske intervale. Kada izlaz senzora postane "LOW", on će ostati "LOW" otprilike dvije sekunde. Tokom ovog perioda, detekcija kretanja je onemogućena. Na primjer, recimo da je senzor podešen na vremenski interval kašnjenja od 4 sekunde i preklopnik na "L". Tako da, prilikom detekcije senzora, izlaz će biti "HIGH" tokom 4 sekunde i zatim će preći u "LOW" na oko dvije sekunde. Svako kretanje tokom ovog perioda biće potpuno ignorisano kao što je prikazano na slici 2. 35, kretanje 2 je ovdje ignorisano. [22]



Slika 2. 35 Periodi zaključavanja detekcije na HC-SR501 [22]

2.1.6 Zvučni signalizator

Piezoelektrični zvučni signalizatori (zujalice), poznati i kao piezoelektrične zujalice, su jednostavni uređaji koji proizvode zvuk na osnovu unosa električne energije. Izraz "piezo" dolazi iz grčkog jezika i znači "pritisnuti" ili "stisnuti". U svijetu elektronike, ovaj izraz se odnosi na princip piezoelektriciteta. Piezoelektricitet je električni naboј koji nastaje u određenim materijalima kao odgovor na primjenjeni mehanički stres. Ovaj fenomen je osnovni princip koji omogućava rad piezoelektričnih zujalica. [23]

Piezoelektrični efekt predstavlja je veliki naučni probaj 1880. godine, kada su braća Curie, Jacques i Pierre, otkrili da određeni materijali proizvode električni naboј pod mehaničkim opterećenjem. Materijali koji pokazuju ovo svojstvo nazivaju se piezoelektrični materijali. Piezoelektrične zujalice koriste ovu pojavu za proizvodnju zvuka. Umjesto da proizvode električni naboј primjenom sile, kod zujalica je suprotno – na materijal se primjenjuje napon kako bi se izazvala deformacija. Kada se primjeni izmjenični napon, materijal se konstantno deformeše i vraća u prvobitni oblik, stvarajući zvučne talase. Ova vrsta zujalice sadrži keramički disk sa elektrodama pričvršćenim na obje strane, unutar zaštitnog kućišta. Struktura piezoelektrične zujalice je relativno jednostavna, ali efikasna, i sastoji se od ključnih komponenti [24]:

- Piezoelektrični disk: Obično je napravljen od specijalizovanog keramičkog materijala kao što je olovo-cirkonat-titanat. Ovaj disk generiše zvučne talase kada je podvrgnut električnom polju, što uzrokuje mehaničke vibracije.
- Metalne ploče: Ploče, koje su često napravljene od mesinga ili nerđajućeg čelika, služe kao elektrode za prenos električnog signala do piezoelektričnog materijala. Njihova veličina i oblik mogu varirati u zavisnosti od veličine zujalice i njenog izlaza.
- Kućište: Neke piezo zujalice imaju kućište, koje je najčešće plastično ili metalno, kako bi zaštitovalo komponente od oštećenja. Dizajn kućišta zavisi od namjene zujalice.



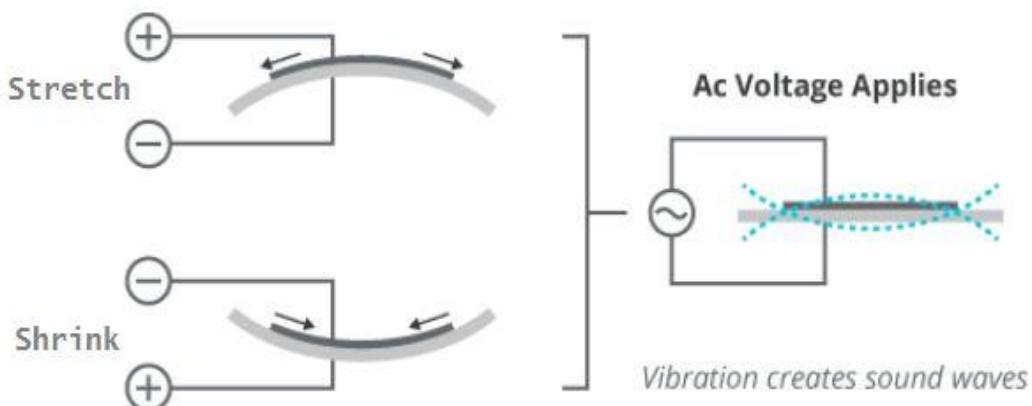
Slika 2. 36 Konfiguracija pinova zujalice [25]

Zujalica ima dva pina, i to pozitivni i negativni. Pozitivni terminal je označen simbolom "+" ili dužim pinom. Ovaj terminal se napaja sa 6 V, dok je negativni terminal označen simbolom "-" ili kraćim pinom i povezan je sa uzemljenjem. [25]

Zujalice mogu raditi ili kao pretvarači ili indikatori u elektronским sklopovima. Pretvarači nemaju integrisano upravljačko kolo, pa im je potreban poseban signal za pravilno funkcionisanje, u obliku izmjeničnog napona. Uređaji koji koriste ove zujalice trebaju vanjsko upravljačko kolo koje će osigurati taj signal. Prednost ovih zujalica je što omogućavaju generisanje zvuka na različitim frekvencijama, prema potrebama aplikacije, ali to povećava složenost dizajna jer je potrebno kreirati dodatno upravljačko kolo. Zujalice dizajnirane kao indikatori imaju ugrađeno upravljačko kolo koje generiše fiksnu frekvenciju ili ton. Za njihov rad, potrebno je samo primijeniti izvor napajanja. Indikatorske zujalice mogu se jednostavno aktivirati putem istosmjernog napona, često koristeći digitalne izlaze mikrokontrolera ili računara, s minimalnom ili nikakvom dodatnom elektronском podrškom. Ova vrsta zujalica je veoma popularna zbog svoje jednostavnosti i praktičnosti, jer ne zahtijevaju složene vanjske sklopove, što ih čini veoma pogodnim za široku primjenu. [26]

Piezoelektrični zvučni signalizator radi na principu piezoelektričnog efekta. Glavna komponenta piezoelektričnog zvučnog signalizatora je piezoelektrični element, koji se sastoji od piezoelektrične keramike i metalne ploče. Ove komponente su povezane ljepljivim slojem. Elektrode su pričvršćene na piezokeramički disk. Kada se na disk primjeni izmjenična struja, piezoelektrični disk se širi i skuplja dijametralno, što uzrokuje vibracije unutar piezoelektričnog elementa. Ove vibracije generiraju zvuk određene frekvencije ili raspona frekvencija. Piezoelektrični element se napaja izmjeničnom strujom iz osculatorskog kola. [26]

Piezo zujalice imaju širok radni napon koji se kreće od 3 V do 250 V. Većina piezo zujalica koje se koriste u elektronskim krugovima ima radni napon između 3 V i 12 V. Ove zujalice imaju visok nivo zvučnog pritiska i veoma nisku potrošnju struje. Što je viša frekvencija ili ton piezo zujalice, to je niža potrošnja struje. Zujalice koje se koriste u elektronskim aplikacijama troše struju od samo 30 mA. [26]



Slika 2. 37 Princip rada piezoelektričnog zvučnog signalizatora [26]

2.2 Softverski alati

U ovom dijelu rada biće detaljno opisani softverski alati koji su korišteni u implementaciji sistema za automatsku radiogoniometriju. Dok su hardverske komponente predstavljene u prethodnom odjeljku, ovdje se fokus prebacuje na softverske alate koji su korišteni za implementaciju sistema, čija je uloga bila integracija različitih hardverskih komponenti u jednu funkcionalnu cjelinu. Softver igra ključnu ulogu u omogućavanju pravilne komunikacije između komponenti, preciznog prikupljanja podataka i efikasnog upravljanja funkcijama sistema. U radu su korištena sljedeća tri alata:

- XCTU
- Arduino IDE
- C#

Svaki alat ima specifičnu ulogu u procesu razvoja, od programiranja mikrokontrolera do konfiguracije komunikacijskih modula i upravljanja glavnim dijelom sistema putem korisničkog interfejsa. Ovaj pregled će pružiti osnovne informacije o alatima, njihovim funkcionalnostima i ključnim karakteristikama, čime se stvara osnova za bolje razumijevanje njihove uloge u kasnijim fazama rada.

2.2.1 XCTU

XCTU je softver razvijen od strane kompanije Digi International za konfiguraciju, testiranje i upravljanje XBee komunikacijskim modulima. Ovaj alat omogućava korisnicima da lako interaguju s XBee uređajima, čime se pojednostavljuje rad s bežičnim mrežama i uređajima. XCTU je razvijen kao odgovor na potrebu za jednostavnim i efikasnim alatom koji omogućava korisnicima da maksimalno iskoriste potencijal XBee modula. S obzirom na kompleksnost bežičnih komunikacija, XCTU je osmišljen da bude intuitivan i lako upotrebljiv, što je olakšalo pristup širokom spektru korisnika, od početnika do iskusnih inženjera. [27]

XCTU je posebno koristan za inženjere i hobiste koji rade s XBee modulima, jer nudi različite funkcionalnosti [27]:

- Konfiguracija modula: Korisnici mogu lako postaviti i konfigurisati parametre XBee modula, kao što su brzina prenosa podataka, identifikatori mreže i sigurnosne postavke.
- Testiranje komunikacije: XCTU omogućava korisnicima da testiraju i provjeravaju komunikaciju između XBee modula, olakšavajući otkrivanje problema i optimizaciju mrežne performanse.
- Upravljanje mrežom: Pruža mogućnosti za upravljanje i vizualizaciju XBee mreža, što pomaže u identifikaciji i rješavanju problema sa povezivanjem.

2 Komponente i alati za implementaciju sistema



Slika 2. 38 XCTU [27]

XCTU je podijeljen na pet glavnih sekcija: meni traku (engl. *the menu bar*), glavnu alatnu traku (engl. *main toolbar*), listu uređaja (engl. *devices list*), radno područje (engl. *working area*) i statusnu traku (engl. *status bar*). Meni traka se nalazi na vrhu aplikacije. Možete koristiti meni traku za pristup svim funkcijama, alatima i radnim režimima XCTU-a. Glavna traka s alatima se nalazi na vrhu aplikacije i podijeljena je u tri sekcije. Prva sekcija uključuje dvije ikone koje služe za dodavanje radio modula u listu. (Slika 2. 41 a)). Druga sekcija obuhvata statičke funkcionalnosti XCTU-a koje ne zahtjevaju prisustvo radio modula. Ova sekcija sadrži alate XCTU, opcije za konfiguraciju, obrazac za povratne informacije, kao i funkcije pomoći i ažuriranja. (Slika 2. 41 b)). Treća sekcija se sastoji od kartica koje predstavljaju tri različita radna režima XCTU-a. Da biste iskoristili ovu funkcionalnost, potrebno je dodati jedan ili više radio modula u listu. (Slika 2. 41 c)). [27]



Slika 2. 39 Meni traka [27]

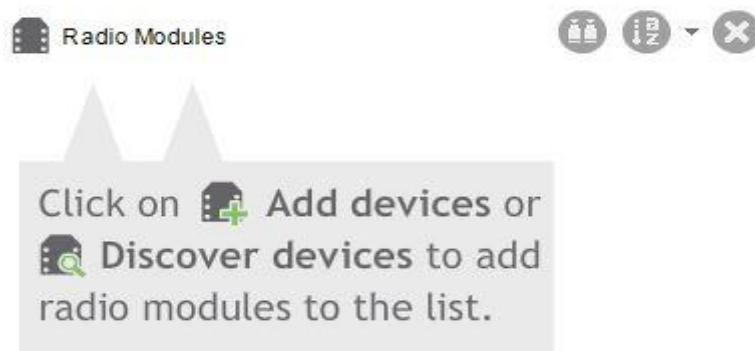


Slika 2. 40 Glavna alatna traka [27]

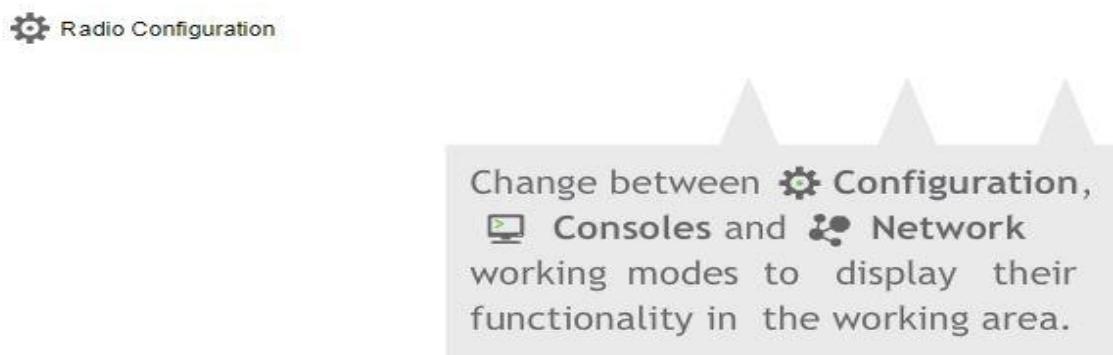


Slika 2. 41 a) Prva sekcija b) Druga sekcija c) Treća sekcija [27]

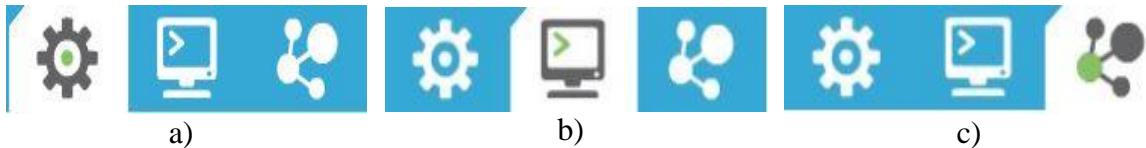
Lista radio modula, ili lista uređaja, nalazi se na lijevoj strani alata i prikazuje radio module koji su povezani s vašim računarom. Ako poznajete konfiguraciju serijskog porta radio modula, možete ga direktno dodati na listu. Takođe možete koristiti funkciju otkrivanja u XCTU-u kako biste pronašli radio module povezane s vašim računaram i dodali ih na listu. (Slika 2. 42). Radni prostor je najveći dio i nalazi se na desnoj strani aplikacije. Sadržaj radnog prostora zavisi od izabranog radnog režima u alatnoj traci. Da biste mogli interagovati sa kontrolama prikazanim u radnom prostoru, morate dodati jedan ili više radio modula na listu, a jedan od modula mora biti izabran. (Slika 2. 43). [27]



Slika 2. 42 Lista uređaja [27]



Slika 2. 43 Radno područje [27]



Slika 2. 44 a) Konfiguracijski režim b) Konzolski režim c) Mrežni režim [27]

U režimu konfiguracije možete konfigurisati radio modul odabran iz vašeg spiska uređaja. Režim konzola omogućava interakciju ili komunikaciju s odabranim radio modulom. Kada pređete na režim mreže, možete otkriti i vizualizirati topologiju i međusobne veze vaše mreže. [27]

2.2.2 Arduino IDE

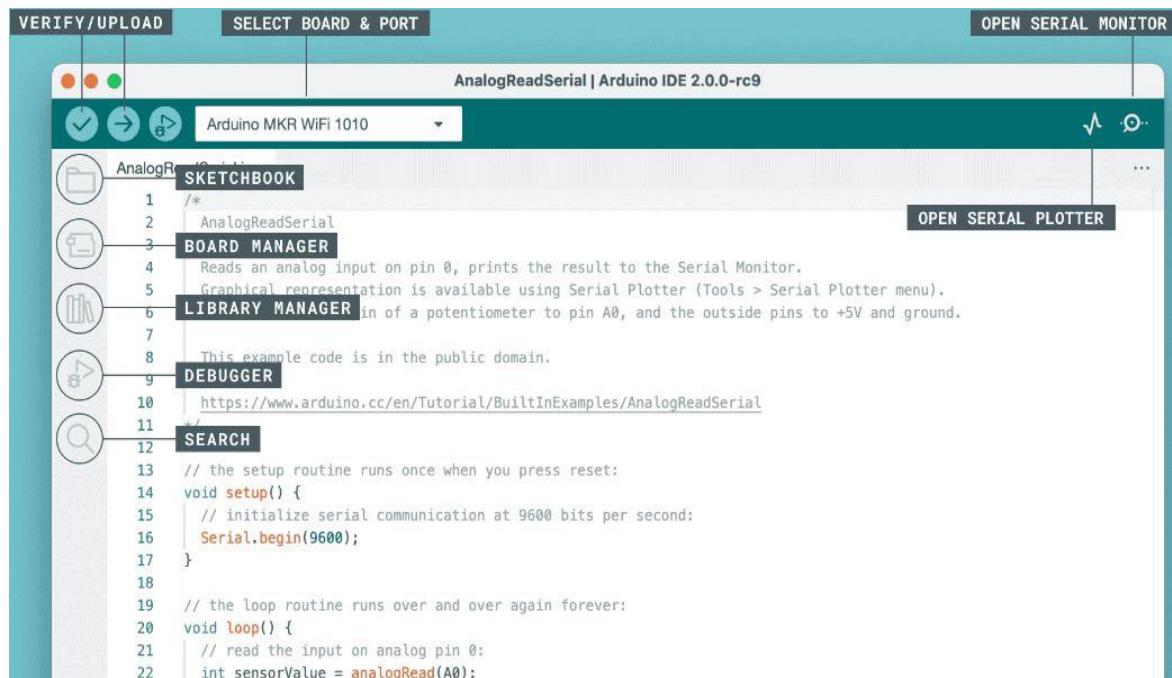
Arduino IDE je softverski alat dizajniran za jednostavno programiranje i razvoj aplikacija na Arduino platformi. Ovo integrисano razvojno okruženje – IDE (engl. *Integrated Development Environment*) korisnicima omogućava pisanje, uređivanje i slanje koda na Arduino mikrokontrolere. Kroz intuitivan interfejs, korisnici mogu brzo razvijati i testirati svoje projekte, što čini ovaj alat izuzetno korisnim za početnike, ali i za iskusne inženjere koji žele efikasno raditi na svojim aplikacijama. Za pisanje programa za Arduino koristi se Embedded C. Ova vrsta programiranja se koristi u slučajevima kada je kod veoma blizak hardveru, što znači da jezik direktno komunicira sa hardverom. Arduino programski jezik je sličan C++, ali sadrži dodatne funkcije i metode. Arduino podržava oba jezika, C i C++. Uvođenje Arduino IDE-a je značajno pojednostavilo proces rada s mikrokontrolerima, jer nudi jednostavno okruženje za razvoj i debagovanje aplikacija. Takođe dolazi s unaprijed instaliranim bibliotekama koje olakšavaju upravljanje različitim hardverskim komponentama, kao što su senzori, aktuatori i drugi periferni uređaji. Ključne karakteristike Arduino IDE-a [28]:

- Pisanje koda: Korisnici mogu unositi i uređivati kod u glavnom prozoru aplikacije, koji podržava prepoznavanje sintakse, olakšavajući rad s kodom i omogućavajući lakšu navigaciju kroz funkcije i komande.
- Kompajliranje i učitavanje: Nakon što je kod napisan, moguće ga je kompajlirati i poslati direktno na Arduino ploču putem serijske komunikacije, omogućavajući brzo testiranje.
- Serijski monitor: IDE nudi serijski monitor koji omogućava prikazivanje povratnih informacija sa Arduino ploče, čime korisnici mogu testirati kako sistem funkcioniše u stvarnom vremenu.

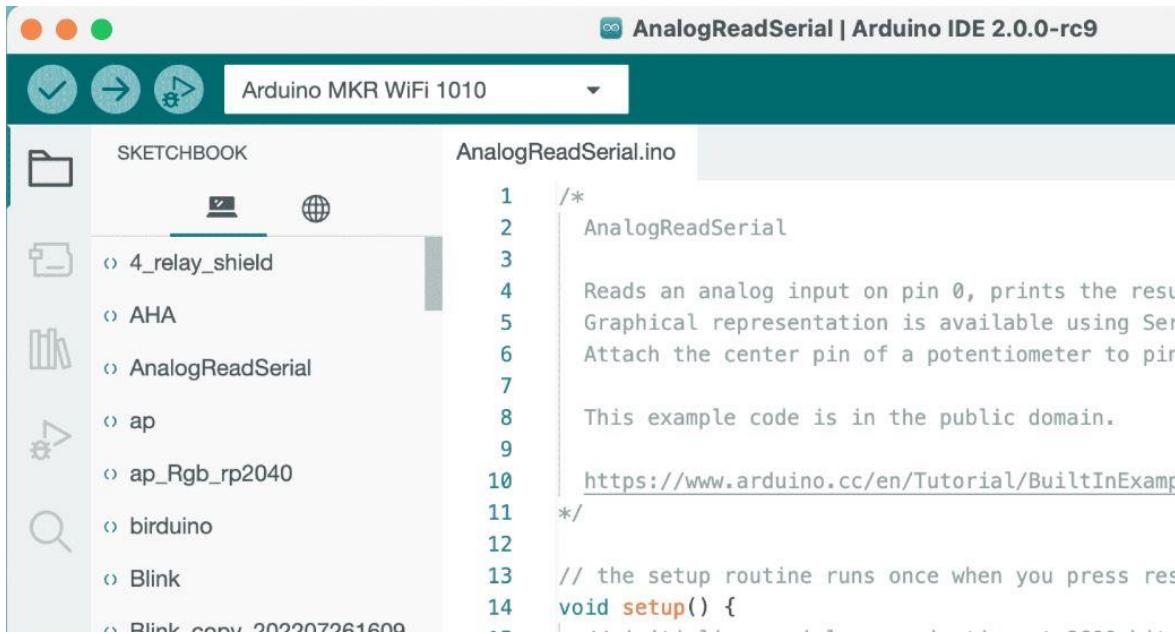
Arduino IDE je razvijen da bude kompatibilan s većinom Arduino razvojnih ploča, uključujući Arduino Uno, Nano, Mega, i druge. Njegova jednostavnost omogućava korisnicima, bez obzira na nivo stručnosti, da lako pišu i primjenjuju kod u različitim projektima. [28]

Arduino IDE 2 je unaprjeđena verzija klasičnog IDE-a, koja uvodi nove funkcije i unaprjeđuje korisničko iskustvo. Njegova redizajnirana struktura i novi alati omogućavaju lakši pristup najčešće korištenim opcijama, kao što je bočna traka, čineći rad sa Arduino pločama još efikasnijim i intuitivnijim. Osnovne funkcionalnosti ovog okruženja su [28]:

- Verifikacija/Učitavanje (engl. *Verify/Upload*) – kompajlira i učitava vaš kod na Arduino ploču.
- Odabir ploče i porta (engl. *Select Board & Port*) – automatski prikazuje otkrivene Arduino ploče, zajedno sa brojem porta.
- Otvorite serijski monitor (engl. *Open Serial Monitor*) – otvara alat za serijski monitor kao novi tab u konzoli.
- Otvorite serijski ploter (engl. *Open Serial Plotter*) – otvara alat za vizualizaciju podataka pomoću grafikona, koji omogućava praćenje različitih varijabli, poput naponskih skokova, u stvarnom vremenu.
- Knjiga skica (engl. *Sketchbook*) – ovdje se nalaze sve vaše skice lokalno pohranjene na računaru. Takođe, možete sinhronizovati sa Arduino Cloud-om i preuzeti skice iz online okruženja.
- Upravljač pločama (engl. *Boards Manager*) – pregledajte Arduino i pakete trećih strana koje možete instalirati.
- Upravljač bibliotekama (engl. *Library Manager*) – pregledajte hiljade Arduino biblioteka kreiranih od strane Arduina i njegove zajednice.
- Alat za otklanjanje grešaka (engl. *Debugger*) – testira i otklanja greške u programima u stvarnom vremenu.
- Pretraga (engl. *Search*) – pretražuje ključne riječi u vašem kodu.

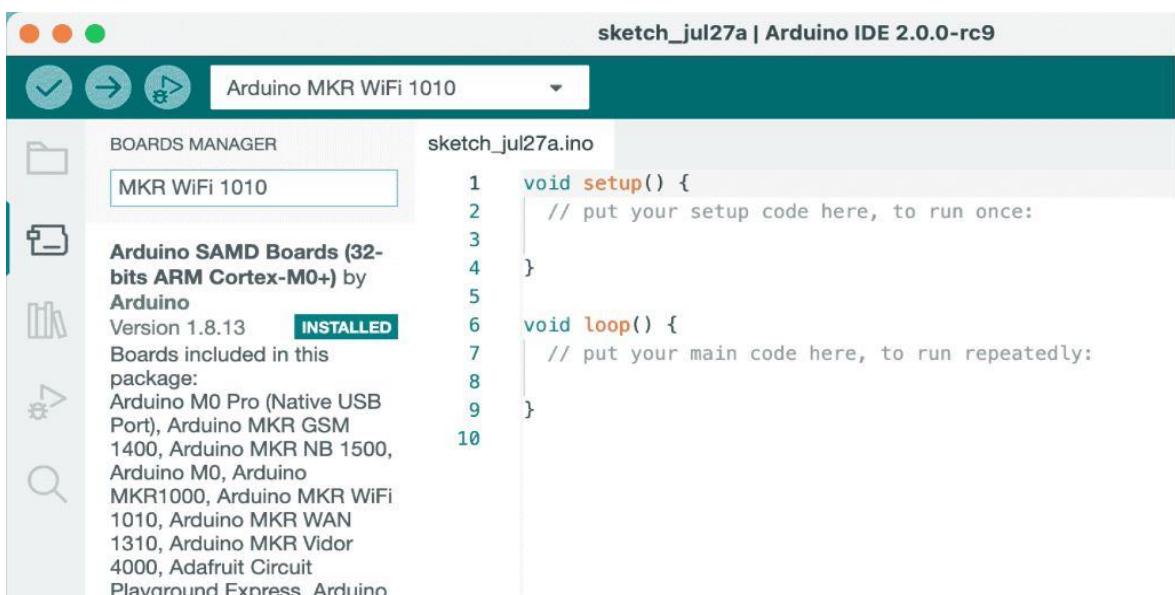


Slika 2. 45 Arduino IDE 2 [28]

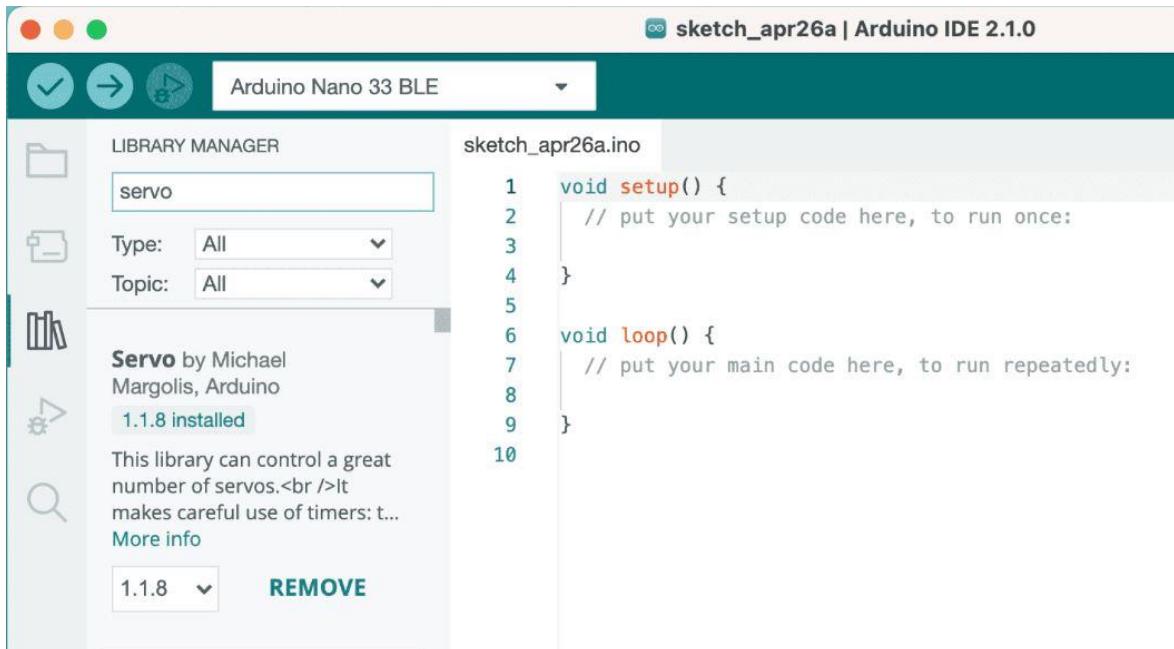


Slika 2. 46 Arduino knjiga skica [28]

Mapa knjiga skica je mjesto gdje se pohranjuju datoteke s kodom. Arduino skice spremaju se kao “.ino” datoteke i moraju biti sačuvane u fascikli s istim imenom. Na primjer, skica nazvana “my_sketch.ino” mora biti pohranjena u fascikli pod nazivom “my_sketch”. Obično se skice spremaju u fasciklu nazvanu “Arduino” unutar fascikle “Documents”. Uz upravljač ploča, možete pregledavati i instalirati pakete ploča. Paket ploča sadrži upute za kompilaciju vašeg koda za ploče koje su uključene u taj paket. Postoji nekoliko paketa ploča za Arduino, kao što su “avr”, “samd”, “megaavr” i drugi. [28]



Slika 2. 47 Arduino upravljač pločama [28]



Slika 2. 48 Arduino upravljač bibliotekama [28]

Upravljač biblioteka omogućava da pretražujete i instalirate brojne biblioteke. Ove biblioteke proširuju funkcionalnosti Arduino API-ja i olakšavaju zadatke kao što su upravljanje servo motorima, očitavanje određenih senzora ili korištenje Wi-Fi modula. Serijski monitor je alat koji omogućava da pregledate podatke koji dolaze s ploče, putem, na primjer, komande "Serial.print()". Historijski gledano, ovaj alat je bio smješten u odvojenom prozoru, ali je sada integriran s editorom. To olakšava pokretanje više instanci istovremeno na računaru. [28]



Slika 2. 49 Arduino serijski monitor [28]

2.2.3 Visual Studio 2022

Visual Studio 2022 je moćno integrirano razvojno okruženje koje je razvila kompanija “Microsoft”. Pruža podršku za širok spektar programskih jezika, među kojima je i C#, koji je u ovom projektu korišten za programiranje skripte i grafičkog korisničkog interfejsa – GUI (engl. *Graphical User Interface*). Visual Studio je idealan za kreiranje aplikacija različite složenosti, od jednostavnih konzolnih programa do naprednih, kompleksnih sistema. [29]

Ključne karakteristike Visual Studija 2022 [29]:

1. Kreiranje i postavljanje projekta:

- Prilikom pokretanja novog projekta, Visual Studio omogućava odabir tipa projekta (u ovom slučaju, Windows Forms aplikacije) za razvoj s GUI-jem. Možete imenovati projekat i odrediti njegovu lokaciju. Nakon postavljanja, IDE se otvara, pružajući organizovano okruženje za pisanje, dizajniranje i testiranje koda.

2. Pregled rasporeda: Visual Studio 2022 se sastoji od više sekcija koje pružaju sve alate potrebne za razvoj:

- Istraživač rješenja (engl. *Solution Explorer*): Smješten na desnoj strani, ova sekcija prikazuje hijerarhijski prikaz svih datoteka i resursa u projektu. Ključna je za navigaciju i upravljanje strukturom koda.
- Prozor uređivača (engl. *Editor Window*): Ovo je centralni dio rasporeda za pisanje i uređivanje koda. Pruža isticanje sintakse, IntelliSense za prijedloge koda i podršku za otklanjanje grešaka. U ovom dijelu napisan je C# kod za aplikaciju.
- Alatni panel (engl. *Toolbox*): Prilikom rada s aplikacijama zasnovanim na GUI-ju (poput Windows Forms), alatni panel nudi široku paletu kontrola koje se mogu prevući i ispustiti (engl. *drag-and-drop*), kao što su dugmad, oznake i tekstualna polja, koje se mogu dodati na formu radi vizuelnog dizajniranja interfejsa.
- Prozor svojstava (engl. *Properties Window*): Nakon dizajniranja korisničkog interfejsa, prozor vam omogućava da modifikujete svojstva svake kontrole, kao što su veličina, boja i ponašanje.
- Prozor izlaza i lista grešaka (engl. *Output and Error List*): Donji dio prikazuje povratne informacije u realnom vremenu iz IDE-a, kao što su izlaz pri gradnji i lista svih grešaka ili upozorenja koja se javljaju tokom kompajliranja ili izvršavanja.

3. Gradnja, pokretanje i otklanjanje grešaka (engl. *Build, Run and Debugging*):

- Gradnja (engl. *Build*): Nakon pisanja koda, aplikacija se gradi kompajliranjem putem opcije “Build”. Visual Studio provjerava kod na greške i generiše izvršnu datoteku (.exe).

- Pokretanje (engl. *Run*): Aplikacija se može pokrenuti direktno iz IDE-a, što omogućava njeno testiranje u realnom vremenu.
 - Otklanjanje grešaka (engl. *Debugging*): Visual Studio nudi napredne alate za otklanjanje grešaka, omogućavajući prolazak kroz kod liniju po liniju, provjeravanje promjenljivih i postavljanje tačaka prekida za pauziranje izvršavanja. Ovi alati pomažu u osiguravanju da aplikacija funkcioniše kako se očekuje.
4. Integracija Gita i GitHub-a: Visual Studio 2022 omogućava besprijeckornu integraciju sa Git-om za upravljanje verzijama. Korisnici mogu lako:
 - Klonirati repozitorije sa GitHub-a direktno u Visual Studio.
 - Izvršavati izmjene, kreirati grane i slati ažuriranja u repozitorij bez napuštanja IDE-a. Ova integracija olakšava upravljanje verzijama koda, saradnju sa drugim programerima i efikasno praćenje promjena.
 5. Proširivost: Visual Studio nudi širok spektar ekstenzija koje poboljšavaju radne tokove. Na primer, moguće je dodati nove jezike ili alate koji pomažu u testiranju, dokumentaciji i još mnogo toga.

3 Sistem za takmičenje iz radioorientacije

U ovom poglavlju biće predstavljen sistem takmičenja iz radioorientacije, koji je dizajniran s ciljem unaprjeđenja sposobnosti učesnika u preciznom lociranju radio signala koristeći radiogoniometar. Takmičenje simulira stvarne situacije u kojima se takmičari suočavaju sa izazovima "lova na lisice," što podrazumijeva brzo pronalaženje izvora radio signala. U nastavku će biti detaljno opisan sistem koji se koristi tokom takmičenja, kako bi čitatelji sa manje tehničkog predznanja lakše razumjeli funkcionisanje sistema. Takođe, biće pojašnjeni principi i pravila takmičenja, omogućavajući takmičarima da shvate ključne aspekte ovog zanimljivog i dinamičnog sporta. Tradicionalno, ova vrsta takmičenja organizuje se uz značajan uticaj ljudskog faktora. Takmičenje zahtijeva prisustvo sudije, koji prati učesnike i potvrđuje da li su pronašli predajnik, kao i tri osobe koje ručno uključuju predajnike na osnovu gestikulacije sudije. Sudija odlučuje koji će predajnik biti uključen, dok se vrijeme takmičenja mjeri od trenutka početka takmičenja korištenjem štoperice. Ova vrsta takmičenja u amaterskoj radioorientaciji, koja se koristi za razvijanje vještina mlađih radioamatera, zahtijevala je automatizaciju kako bi se poboljšala regularnost i eliminisao ljudski faktor. Kroz automatizaciju, kao što je implementirano u ovom radu, proces takmičenja postaje znatno pouzdaniji jer se svi ključni koraci, uključujući uključivanje predajnika, praćenje takmičara i registraciju vremena, izvršavaju bez potrebe za ručnom intervencijom. Na ovaj način se osigurava nepristrasnost i preciznost, što omogućava pravičnije i efikasnije takmičenje za sve učesnike.

Rad sistema takmičenja iz radioorientacije započinje tako što organizator postavlja tri Arduino mikrokontrolera, koji kontrolisu predajnike, PIR senzore, buzzere i XBee module. Organizator prvo povezuje mikrokontrolere na napajanje putem baterije, osigurava uzemljenje i postavlja antene kako bi se pripremili za rad. Nakon toga, XBee modul, koji ima ulogu koordinatora u ZigBee mreži, se povezuje sa laptopom. Sistem je spreman kada se na laptopu pokrene aplikacija pod nazivom "SignalSeekComp", koja je kreirana u C# jeziku pomoću Visual Studio okruženja. Korisnik vidi GUI sa porukom dobrodošlice i dugmetom "Start", koji započinje takmičenje.

Nakon pritiska na dugme "Start", skripta nasumično odabira jedan od tri XBee modula koji su povezani sa Arduino mikrokontrolerima u ZigBee mreži. Odabrani Arduino mikrokontroler prima signal preko svog XBee modula, koji je bežičnom komunikacijom poslat preko ZigBee koordinatora. Nakon prijema signala, mikrokontroler aktivira predajnik na frekvenciji od 3.5 MHz, koji počinje emitovati signal. Takmičar zatim koristi radiogoniometar da pronađe lokaciju predajnika.

Radiogoniometar, koji takmičar koristi, je prijemnik signala sa antenom koji omogućava lociranje izvora signala na frekvenciji od 3.5 MHz. Kada predajnik počne emitovati signal, takmičar putem slušalica priključenih na radiogoniometar osluškuje promjene u intenzitetu signala. Kako se takmičar približava predajniku, jačina signala postaje sve izraženija, što omogućava preciznije navođenje prema izvoru. Uredaj obično koristi feritnu antenu koja pomaže u određivanju pravca odakle dolazi signal, a takmičar koristi ovu informaciju kako bi suzio potragu i fizički pronašao lokaciju predajnika. Kada se signal čuje najglasnije u slušalicama, to znači da se takmičar nalazi u neposrednoj blizini predajnika, a aktivacija PIR senzora potvrđuje njegovu prisutnost.

Kada se takmičar dovoljno približi predajniku, PIR senzor detektuje njegovo prisustvo, što označava da je predajnik pronađen. Arduino zatim šalje povratni signal putem istog XBee modula kojim je primio signal, koristeći ponovo bežičnu komunikaciju. Ovaj signal se potom prosljeđuje skripti preko ZigBee koordinatora, nakon čega se aktivira buzzer, pružajući zvučnu povratnu informaciju da je predajnik pronađen. Skripta zatim nasumično odabire naredni Arduino mikrokontroler, koji započinje isti proces – aktivira predajnik na novoj lokaciji, a takmičar ponavlja potragu za drugim predajnikom. Nakon pronalaska drugog predajnika, treći Arduino se aktivira na isti način, sa mogućnošću da se uključi i predajnik koji je već ranije bio aktiviran. Takmičenje se završava kada takmičar pronađe sva tri predajnika ili diskvalifikacijom, a sistem prestaje sa radom.

Takmičenje iz radioorijentacije odvija se prema nekoliko ključnih pravila:

- Takmičenje se odvija na frekvenciji 3.5 MHz, sa predajnicima snage do 50 mW koji koriste vrstu emisije A1A. Antene su vertikalno polarizovane i duge 1/2 do 1 metar.
- Takmičenje je potpuno automatizovano i ne zahtijeva prisustvo sudijskog žirija. Sistem koristi centralni računar povezan putem XBee modula sa svakim predajnikom. Računar kontroliše uključivanje i isključivanje predajnika.
- Organizator određuje mjesto na kojem će se organizirati takmičenje. Takmičenje se u pravilu organizira na školskom igralištu koje može biti asfaltirano ili travnato, odnosno na nekom drugom prikladnom terenu veličine 20 x 20 m ili većem.
- Pola sata prije samog takmičenja ždrijebom se izvlače startni brojevi takmičara (redoslijed startanja).
- Na terenu su postavljena tri Arduino uređaja, koji kontrolišu predajnike, PIR senzore i buzzere, postavljeni su na rubovima jednakostaničnog trougla. Takmičar započinje potragu iz centra trougla, kako bi mu sva tri predajnika bila jednako udaljena.
- Oko svakog predajnika nalazi se označeni krug poluprečnika 1.5 metara. Kada takmičar uđe u krug, automatski se registruje njegov dolazak pomoću PIR senzora, a predajnik se isključuje, dok sledeći predajnik automatski počinje emitovati.
- Start ima oblik kruga promjera 1 metar i označen je plastičnom trakom ili kredom u prahu (gipsom). Kada je takmičar na startnom mjestu, na oči stavlja posebne neprozirne naočale (može i povez za oči) koje osigurava organizator i na znak sudca na stazi okreće se nekoliko puta oko vlastite ose kako bi izgubio osjećaj za svoj položaj u prostoru. Nakon toga stavlja slušalice na uši i uključuje radiogoniometar.

- Pokretanjem takmičenja na računaru, istovremeno se aktivira startni sat i prvi predajnik. U tom trenutku, računar generiše zvučni signal za start, čime signalizira takmičaru da je takmičenje počelo, a mjerjenje vremena započinje od tog trenutka.
- Vrijeme se mjeri neprekidno od trenutka starta do (znak startnog sata) do ulaska u krug oko zadnjeg (ciljnog) predajnika. Program mora mjeriti i prolazna vremena takmičara kod svakog predajnika u svrhu određivanja rezultata.
- U slučaju kvara radiogoniometra, takmičar završava takmičenje. Plasman se u tom slučaju određuje prema broju do tada pronađenih predajnika. U koliko takmičar nije stigao do prvog predajnika glavni sudac mu može omogućiti novi start s drugim uređajem, ako se utvrdi da je radiogoniometar neispravan.
- Svaki takmičar ima 5 minuta da pronađe sva tri predajnika. Ukoliko ne uspije locirati sve predajnike u tom vremenskom roku, biće diskvalifikovan.
- Rang lista takmičara radi se prema postignutom vremenu, tako da je najbolji onaj takmičar koji je za pronalazak predajnika trebao najmanje vremena.

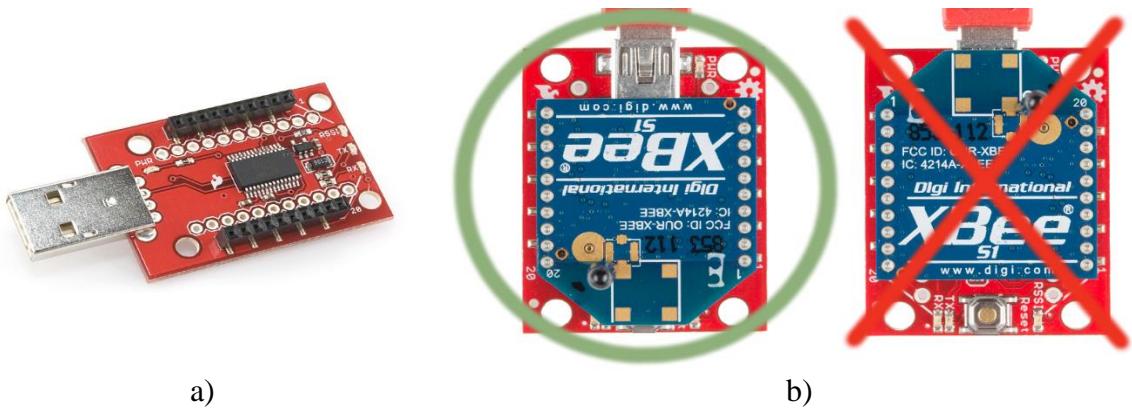
4 Implementacija sistema

U ovom poglavlju detaljno ćemo opisati proces implementacije sistema za automatsko upravljanje takmičenjem iz radioorientacije. Sistem koristi Arduino platformu, XBee komunikacione module, kao i softverske alate za bežičnu komunikaciju i upravljanje takmičenjem. Proces implementacije uključuje nekoliko ključnih koraka: konfiguraciju hardverskih komponenti, razvoj softvera, integraciju sistema, kao i testiranje i evaluaciju.

Da bi se omogućila bežična komunikacija između različitih komponenti sistema, prvo je bilo potrebno konfigurisati XBee module. Korišten je alat XCTU, softver koji omogućava konfiguraciju i testiranje XBee uređaja. Prvi korak bio je postavljanje jednog modula kao koordinatora, dok su ostali moduli konfigurisani kao ruteri ili krajnji uređaji. Koordinator je zadužen za formiranje mreže i upravljanje komunikacijom između modula, dok ruteri i krajnji uređaji komuniciraju s njim putem bežičnog ZigBee protokola. Kako bi se omogućila bežična komunikacija između različitih komponenti sistema, u nastavku će biti prikazan proces konfiguracije XBee modula. Na primjeru dva modula, jedan će biti podešen kao koordinator, dok će drugi biti podešen kao ruter. Ovaj proces važi i za preostala dva modula u sistemu, pri čemu koordinator i prvi ruter služe samo kao primjer. Nakon konfiguracije prvog rutera, taj modul se može zamijeniti sa drugim nepodešenim XBee modulom, te se postupak konfiguracije rutera ponovi.

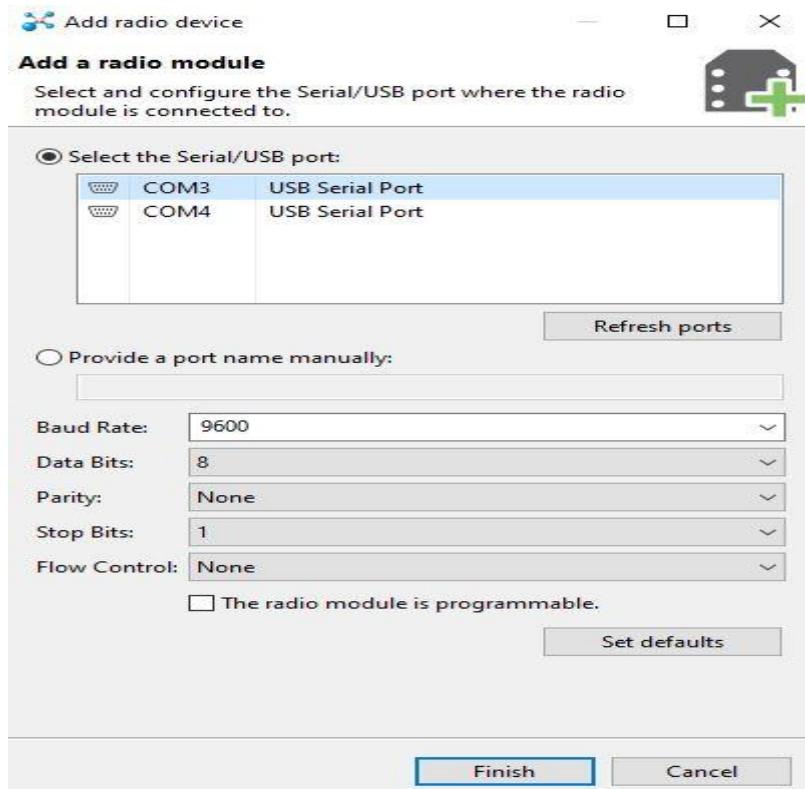
Konfiguracija se vrši pomoću USB adaptera za povezivanje XBee modula, uređaja koji se povezuje sa USB portom laptopa, prikazan na slici 4. 1 a). Ovaj adapter u engleskom jeziku poznat i kao “dongle” omogućava XCTU softveru da prepozna XBee module i omogući njihovo podešavanje. XCTU prepoznaće module automatski kada su povezani preko dongle-a, što znatno olakšava rad sa XBee uređajima. USB donglovi koji se koriste za povezivanje XBee modula sa računarom koriste FTDI FT231X čip, koji omogućava konverziju između serijskog signala (UART) i USB signala. Ovaj čip je popularan zbog svoje kompatibilnosti sa svim platformama, kao i jednostavnosti korištenja. Prilikom prvog povezivanja ovog čipa sa računatom, potrebno je instalirati odgovarajuće drajvere kako bi uređaj dobio jedinstven “COM” port broj, koji se koristi za komunikaciju. Nakon što je drajver instaliran, XBee modul se pažljivo postavi na USB dongle, vodeći računa da se pinovi ne saviju prilikom povezivanja i samog postavljanja u pravilnu poziciju. (Slika 4. 1 b)).

Nakon postavljanja XBee modula na USB adapttere za XBee module, ovi adapteri se povezuju na USB portove laptopa. Time se omogućava da računar prepozna module i omogući njihovu dalju konfiguraciju putem softverskog alata XCTU. Sljedeći korak je pokretanje XCTU softvera, u kojem započinjemo postupak konfiguracije XBee modula. Nakon pokretanja XCTU softvera, prikazuje se njegovo radno okruženje, kao što je prikazano na slici 2.38.

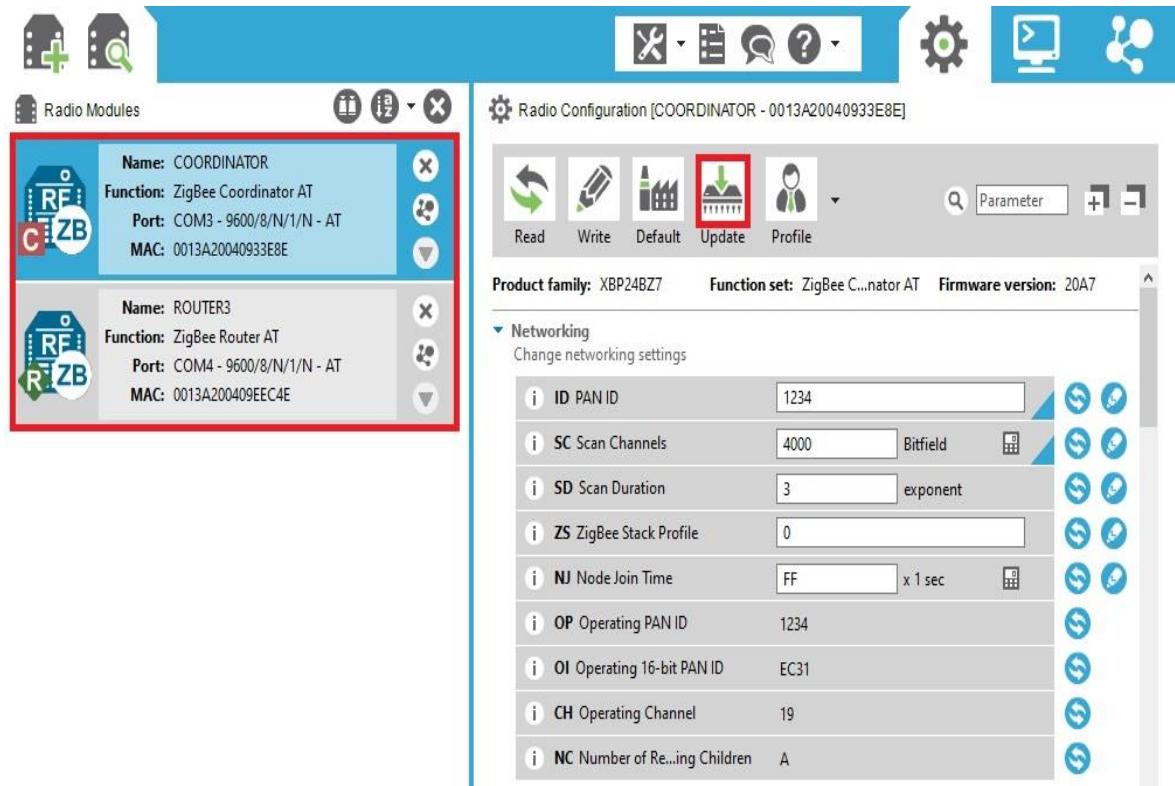


Slika 4. 1 a) USB dongle b) Pravilno postavljanje USB dongle-a [30]

Da biste dodali svoj XBee, kliknite na "Add a radio module" u gornjem lijevom dijelu prozora. Kada se otvorи prozor za dodavanje radio uređaja, korisnici trebaju tražiti "USB Serial Port" u našem slučaju na portu "COM3", kao što je prikazano na slici 4. 2. U tom prozoru postavite brzinu prenosa na 9600 bps. Također, potrebno je podesiti vrijednosti za broj podataka, paritet i bitove za zaustavljanje. Za jednostavnost, preporučuje se da se zadrže svi podrazumijevani parametri, prije nego što kliknete na "Finish". Ovaj postupak će dodati odabrani XBee uređaj. Nakon toga, ponovite istu proceduru za dodavanje drugog preostalog XBee uređaja na portu "COM4".



Slika 4. 2 Dodavanje XBee modula

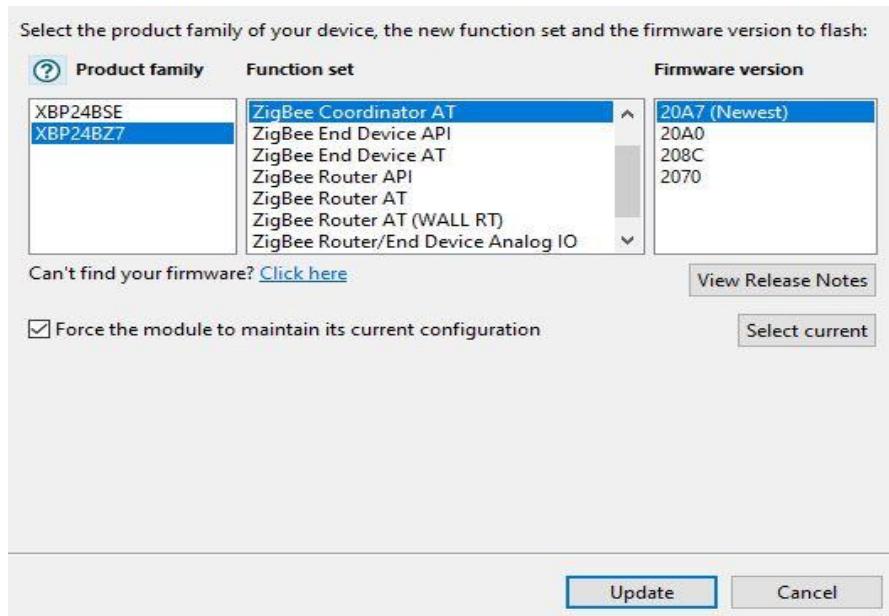


Slika 4. 3 Konfiguracija XBee modula

Nakon što se oba uređaja prikažu u gornjem lijevom uglu XCTU-a, klikom na novi modul inicirate čitanje njegovih trenutnih konfiguracijskih postavki. Kada se konfiguracija XBee modula prikaže kao na slici 4. 3, sljedeći korak je ažuriranje firmvera na željeni set funkcija koji izvršavamo odabirom simbola "Update" u gornjem desnom uglu označenog na slici 4. 3. U ovom prozoru (slika 4. 4) potrebno je odabrati porodicu proizvoda (engl. *Product Family*), set funkcija (engl. *Function Set*) i verziju firmvera (engl. *Firmware Version*). Za realizaciju projekta koristimo "ZigBee Coordinator AT" kao funkcionalni set jer modul podešavamo kao koordinator i verziju "20A7". Nakon što su sve opcije postavljenje, klikom na "Update" ažuriramo firmver odabranog modula. Nakon ažuriranja, modul će biti spremjan za dalju konfiguraciju u okviru ZigBee mreže. Isti postupak ponavljamo i za drugi modul kako bi oba bila podešena za rad u željenom režimu, s tom razlikom što će njegov funkcionalni set biti "ZigBee Router AT" jer modul u ovom slučaju ima ulogu rutera u mreži. Obavezno ažurirajte firmver na svim XBee uređajima u vašoj mreži koristeći isti protokol. U suprotnom, možete imati problema s prenošenjem podataka kroz mrežu.

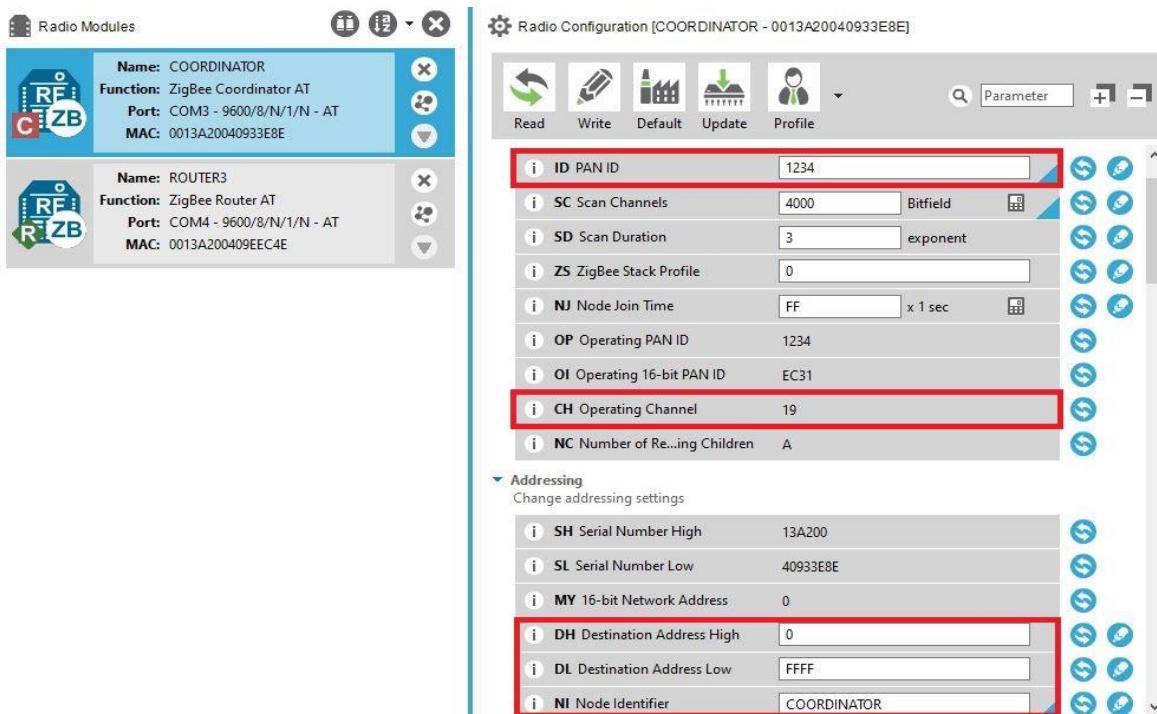
Nakon što su XBee uređaji ažurirani odgovarajućim firmverom, ključno je osigurati da su tri osnovna parametra ispravno postavljena na svim uređajima unutar mreže:

- Brzina prenosa (engl. *baud rate*): Ovaj parametar mora biti identičan za svaki uređaj u mreži.
- PAN ID: Svaki XBee uređaj treba imati jedinstven PAN ID unutar iste mreže.
- Radni kanal: Svi uređaji trebaju raditi na istom kanalu za uspješnu komunikaciju.



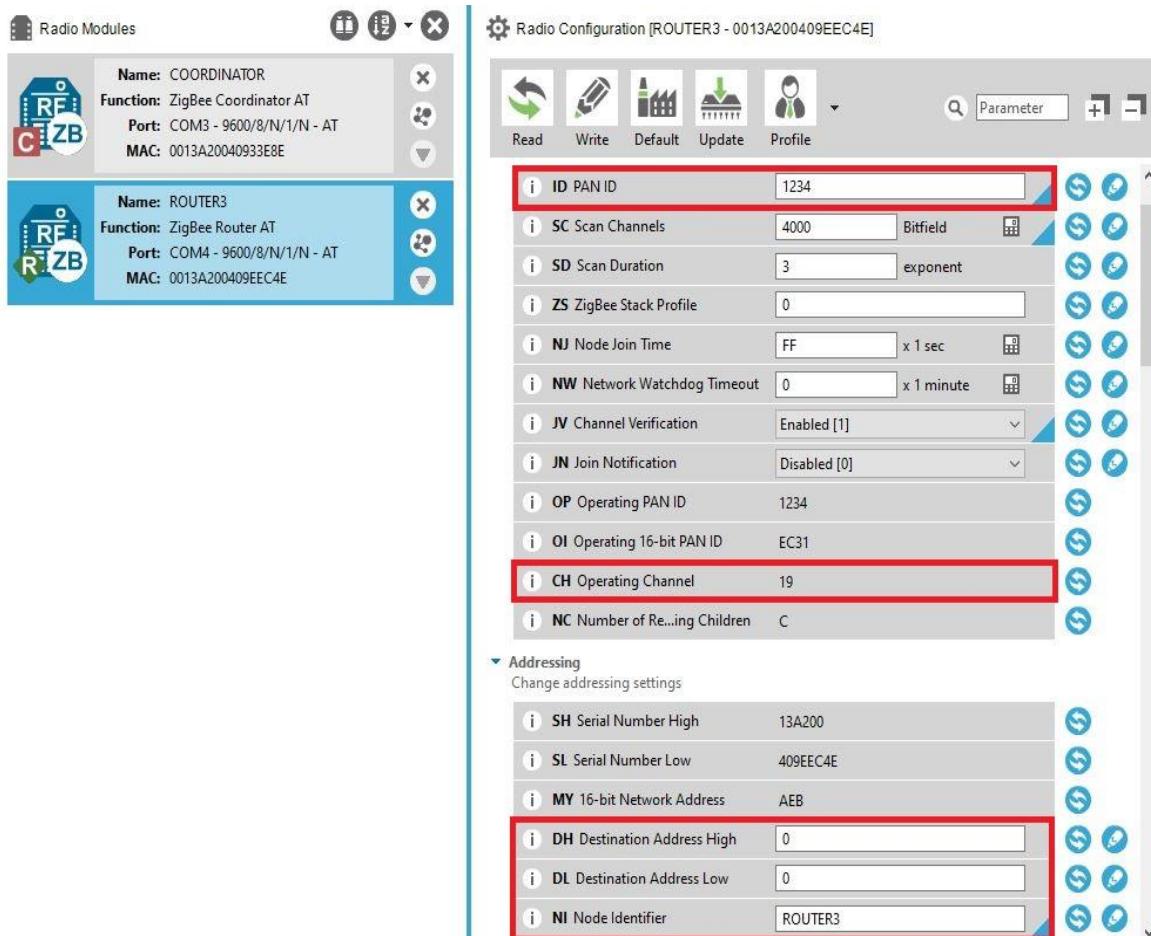
Slika 4. 4 Ažuriranje modula

U ovoj fazi ćemo podesiti parametre XBee uređaja, uključujući PAN ID, kanal i omogućiti komunikaciju u broadcast režimu. U ovom konkretnom slučaju, PAN ID je postavljen na 1234, a izabrani kanal je 19. Koordinator će raditi u broadcast režimu, što znači da su parametri DH (engl. *Destination High*) i DL (engl. *Destination Low*) postavljeni na 0 i FFFF.

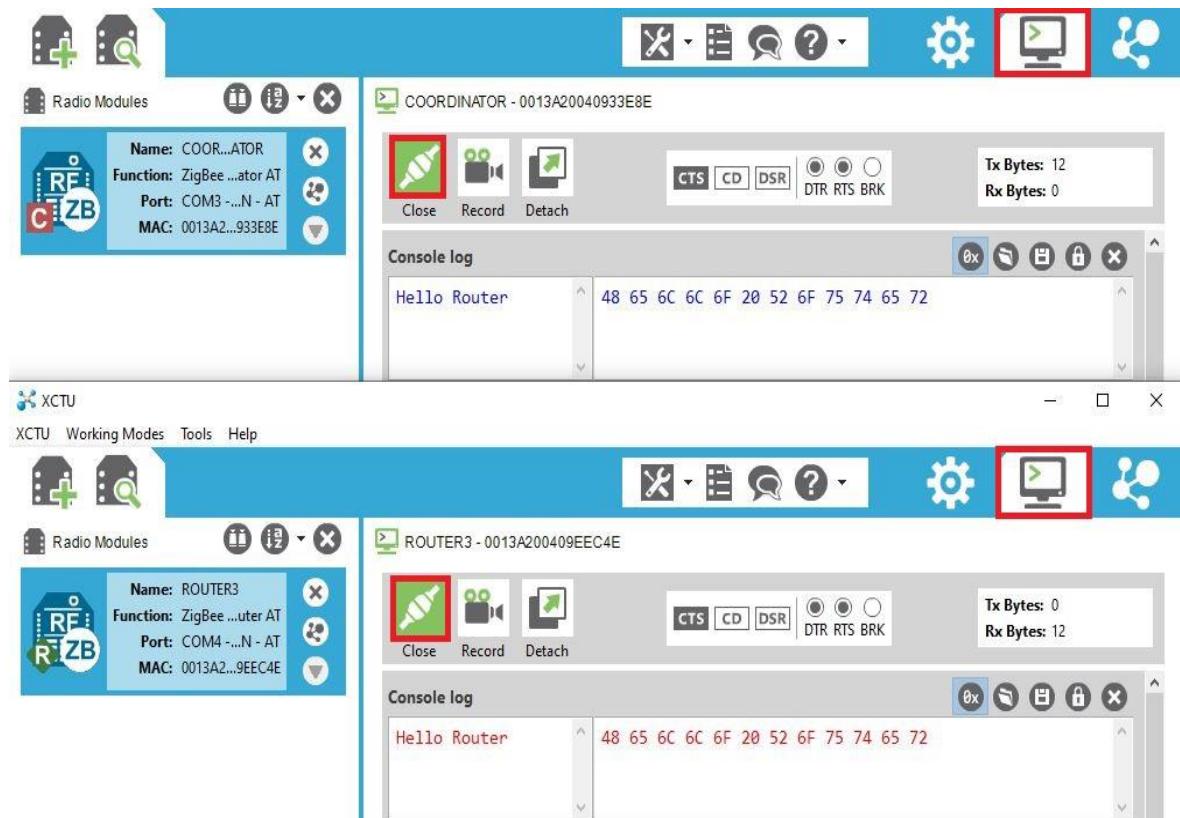


Slika 4. 5 Konfiguracija koordinatora

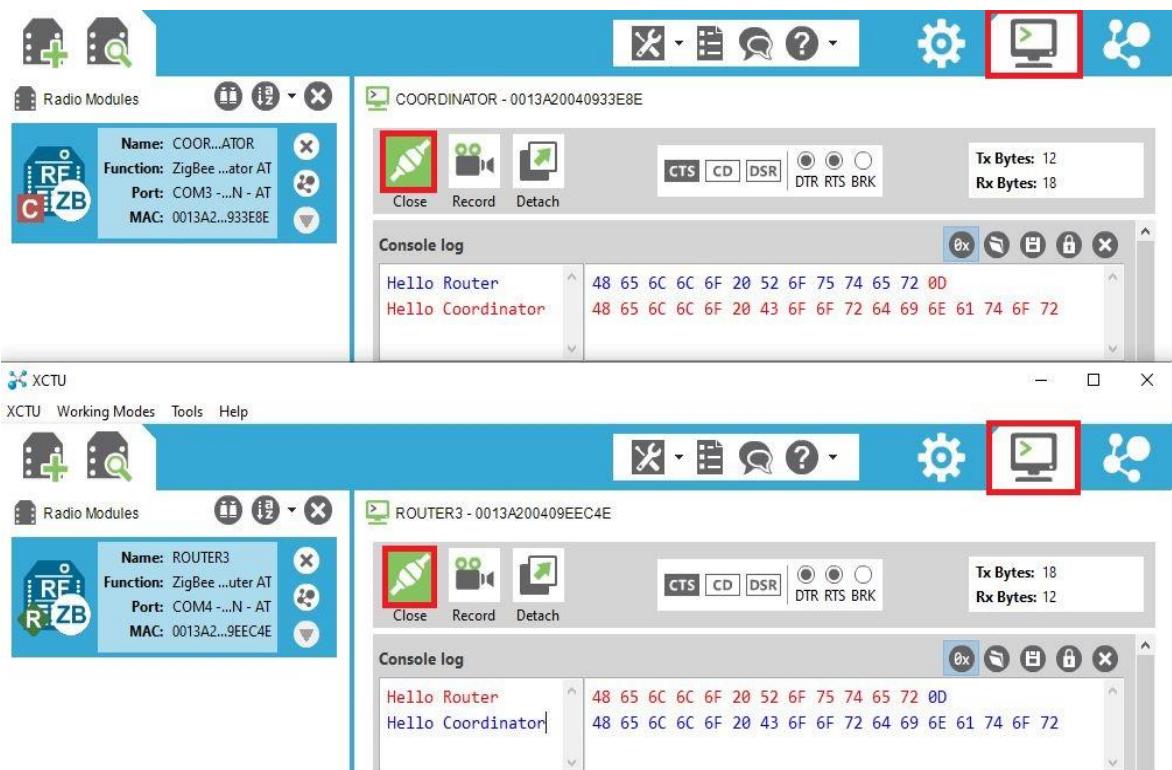
Za drugi modul, koji je konfigurisan kao ruter, PAN ID ostaje isti (1234), dok su DH i DL postavljeni na 0, što je prikazano na slici 4. 6. Takođe, važno je naglasiti da brzina prenosa mora biti jedinstvena za sve uređaje u mreži i iznosi 9600. Sva ostala polja u konfiguraciji XBee uređaja ostavljena su na podrazumijevane vrijednosti. Ovaj pristup omogućava jednostavnu i brzu postavku, uz zadržavanje osnovne funkcionalnosti modula. Na taj način se minimizuje mogućnost grešaka tokom konfiguracije, a uređaji su spremni za rad u okviru definisanih parametara mreže (za detaljnije informacije o ostalim konfiguracijskim poljima, moguće je istražiti navedenu referencu [8]). Ovu konfiguraciju ćemo ilustrovati u konzolnom radnom modu, što će omogućiti da se prate komunikacijski tokovi između koordinatora i rutera. U konzolnom radnom modu, redoslijed komunikacije između XBee modula može se opisati kao proces slanja i primanja poruka između koordinatora i rutera. Na slici 4. 7, prikazana je komunikacija od strane koordinatora prema ruteru, gdje koordinator, označen plavim slovima, šalje tekst "Hello Router". Ova poruka se zatim pojavljuje na strani rutera, prikazana crvenim slovima. Nakon toga (slika 4. 8), ruter odgovara koordinatoru s porukom "Hello Coordinator", koja je označena plavom bojom. Ova poruka se na strani koordinatora prikazuje crvenom bojom. Ovaj primjer uspješne bežične komunikacije između dva XBee modula ilustruje efikasnost konfiguracije i ispravnost postavki u komunikaciji.



Slika 4. 6 Konfiguracija rutera



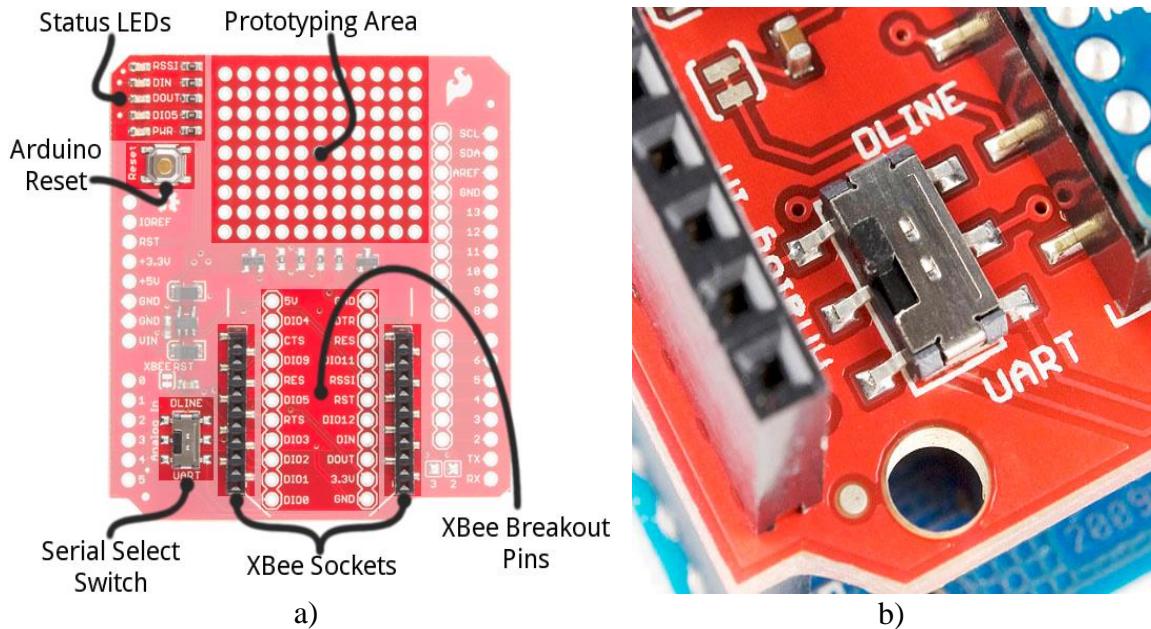
Slika 4. 7 Slanje poruke od koordinatora prema ruteru



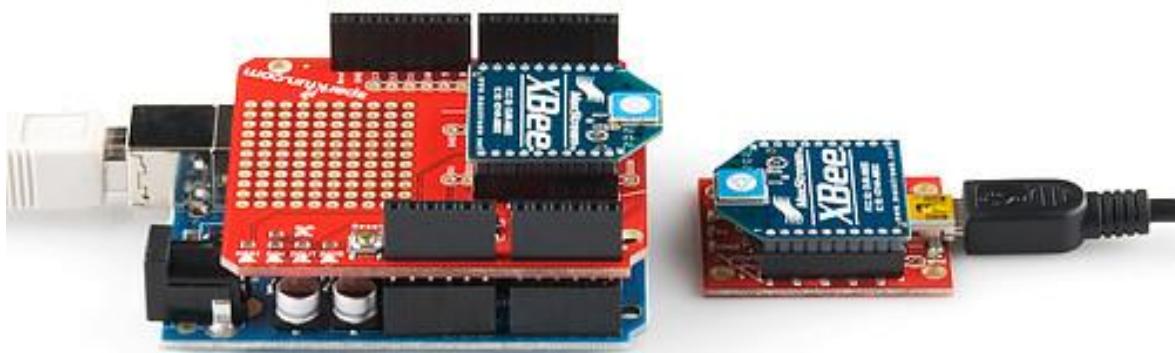
Slika 4. 8 Slanje poruke od rutera prema koordinatoru

Nakon što je objašnjena konfiguracija XBee modula u XCTU softveru, sljedeći korak je povezivanje XBee modula s Arduino platformom putem odgovarajuće ekspanzijske ploče i programiranje samog Arduina. XBee ekspanzijska ploča omogućava jednostavno povezivanje XBee modula sa Arduino pločom, olakšavajući implementaciju bežične komunikacije u sistemu. Ovdje ćemo opisati proces fizičkog povezivanja XBee modula s Arduinom koristeći ovu ekspanzijsku ploču, nakon čega ćemo detaljno objasniti programiranje Arduina za omogućavanje komunikacije između uređaja. Pored XBee modula, na Arduino će se spojiti i druge komponente korištene u projektu, kao što su PIR senzor, predajnik i piezo zvučnik, čime ćemo stvoriti kompletan sistem za detekciju pokreta, slanje i prijem signala, te emitovanje zvučnih upozorenja. Sve ove komponente bit će integrirane kako bi Arduino mogao upravljati interakcijom između senzora i XBee modula te izvršavati predviđene funkcije unutar mreže.

XBee modul se postavlja na XBee ekspanzijsku ploču, koji je dizajniran da omogućava jednostavno povezivanje modula s Arduino pločom. Nakon što je XBee pravilno postavljen na ekspanzijsku ploču, cijela ploča se montira na Arduino Uno. Na samoj XBee ekspanzijskoj ploči nalazi se klizni prekidač koji je potrebno postaviti u "UART" poziciju. Ovaj prekidač omogućava da se XBee modul poveže sa hardverskim serijskim portom Arduina, što je ključno za stabilnu i direktnu komunikaciju. Korištenjem hardverskog serijskog porta, komunikacija između XBee modula i Arduina postaje pouzdanija, što je posebno važno zbog povezanih komponenti poput PIR senzora, predajnika i buzzera u projektu. Nakon spajanja, prelazimo na programiranje samog koda za Arduino kako bismo omogućili izvršavanje svih funkcija na strani Arduina. Važno je napomenuti da je potrebno odspojiti ekspanzijsku ploču sa Arduinom pri svakom učitavanju koda na Arduino, a nakon što se kod učita, ponovo se montira ploča. Na slici 4.10 je prikazana povezanost XBee modula sa ekspanzijskom pločom i Arduinom, bez prikazivanja ostalih komponenti sistema.



Slika 4.9 a) XBee ekspanzijska ploča b) DLINE/UART prekidač [30]



Slika 4. 10 Povezivanje XBee modula sa Arduinom [30]

Kako bismo olakšali objašnjavanje koda i jasnije prikazali njegove funkcionalnosti, podijelit ćemo kod na logičke blokove. Ovaj pristup omogućava detaljnije objašnjenje svakog segmenta koda, korak po korak. Cijeli kod će biti priložen u poglavlju "Dodatak" na kraju rada, gdje će se nalaziti u potpunosti kako bi čitatelji imali pregledniji uvid u sve linije koda.

Prvi logički blok odnosi se na deklaraciju pinova i inicijalizaciju varijabli. U ovom dijelu koda definišemo koji će se pinovi na Arduinu koristiti za povezivanje s različitim komponentama kao što su PIR senzor, LED dioda i buzzer. Također, inicijalizujemo nekoliko varijabli koje ćemo kasnije koristiti za praćenje stanja buzzera, primanja naredbi i upravljanje vremenom trajanja signala. Ova inicijalizacija je ključna jer omogućava Arduinu da prepozna koji pinovi će biti korišteni za unos (engl. *input*) i izlaz (engl. *output*) te kako će se upravljati različitim funkcijama u toku rada. Korištenjem ključne riječi "const int", definišu se brojevi pinova za svaku komponentu. Na primjer, "pirPin = 2" povezuje PIR senzor na digitalni pin 2, dok su pinovi "vPin = 11", "redPin = 12", i "bluePin = 13" postavljeni za upravljanje RGB LED diodom. "buzzerPin = 6" i "buzzerGnd = 7" kontrolišu buzzer, pri čemu jedan pin šalje signal za zvuk, a drugi se koristi za uzemljenje.

```

1 const int pirPin = 2; // Pin za PIR senzor
2 const int vPin = 11; // Pin za LED RGB
3 const int redPin = 12; // Pin za crvenu diodu
4 const int bluePin = 13; // Pin za plavu diodu
5 const int buzzerPin = 6; // Pin za buzzer
6 const int buzzerGnd = 7; // Pin za GND buzzera
7 bool buzzerActive = false; // Stanje buzzera
8 bool commandReceived = false; // Da li je primljena naredba
9 unsigned long buzzerStartTime = 0; // Vrijeme početka trajanja buzzera

```

Slika 4. 11 Deklaracija pinova i inicijalizacija varijabli

Važno je napomenuti da se u ovoj implementaciji koristi RGB LED dioda kao komponenta koja olakšava demonstraciju rada sistema. Iako je implementirana u kodu, funkcija RGB diode je prvenstveno vizualna, za prikaz rada sistema na jednostavan način. Kod je osmišljen tako da se dioda može uvijek isključiti sa Arduino ploče, a na njeno mjesto povezati predajnik. Na ovaj način, sistem može funkcionirati u pravoj demonstraciji sa predajnikom i radiogoniometrom, dok je RGB dioda prisutna isključivo radi lakše demonstracije i vizualizacije funkcionalnosti. Tada se predajnik spaja tako da se pozitivni pin predajnika poveže na “vPin” (digitalni pin 11), dok se drugi pin povezuje na “GND”. Ova konfiguracija omogućava pravilno napajanje predajnika, čime se omogućava njegova funkcionalnost u sistemu.

Drugi logički blok odnosi se na funkciju “setup()”, koja se koristi za inicijalizaciju raznih postavki u Arduino kodu. Ova funkcija se izvršava samo jednom, kada se Arduino uključi ili restartuje. Njena primarna svrha je da postavi sve potrebne konfiguracije koje će omogućiti pravilno funkcionisanje programa u toku dalnjeg izvođenja “loop()” funkcije, gdje se odvija glavni dio logike programa. U funkciji “setup()” svaki pin se postavlja prema njegovoj namjeni. Na primjer, “pinMode(pirPin, INPUT)” linija definiše pin povezan s PIR senzorom kao ulazni pin. To znači da će Arduino pratiti signal sa ovog pina, koji se koristi za detekciju pokreta. “pinMode(vPin, OUTPUT)” linija postavlja “vPin”, koji je povezan s RGB LED diodom, kao izlazni pin. To omogućava Arduinu da šalje signal za uključivanje ili isključivanje LED diode. “pinMode(redPin, OUTPUT)” i “pinMode(bluePin, OUTPUT)” linije definišu pin za crvenu i plavu diodu kao izlazni pin, omogućavajući upravljanje ovim bojama diode, dok se linijama “digitalWrite(redPin, LOW)” i “digitalWrite(bluePin, LOW)” isključuju crvena i plava boja diode jer neće biti korištene u radu. “pinMode(buzzerPin, OUTPUT)” linija definiše pin za buzzer kao izlazni pin, što omogućava Arduinu da šalje signal za aktivaciju buzzera. “pinMode(buzzerGnd, OUTPUT)” linija postavlja pin za “GND” buzzera kao izlazni pin, što je važno za pravilno napajanje i rad buzzera, dok “digitalWrite(buzzerGnd, LOW)” linija postavlja pin za uzemljenje buzzera na “LOW”, čime se osigurava da je buzzer isključen pri inicijalizaciji. “Serial.begin(9600)” linija pokreće serijsku komunikaciju sa brzinom prenosa od 9600 bita u sekundi.

```

11 void setup() {
12     pinMode(pirPin, INPUT);
13     pinMode(vPin, OUTPUT);
14     pinMode(redPin, OUTPUT);
15     pinMode(bluePin, OUTPUT);
16     pinMode(buzzerPin, OUTPUT);
17     pinMode(buzzerGnd, OUTPUT);
18     digitalWrite(buzzerGnd, LOW);
19     digitalWrite(redPin, LOW); // Isključimo crvenu diodu
20     digitalWrite(bluePin, LOW); // Isključimo plavu diodu
21
22     Serial.begin(9600); // Pokretanje serijske komunikacije
23 }
```

Slika 4. 12 Setup() funkcija

```

25 void loop() {
26     // Provjera da li je detektiran pokret i da li je primljena naredba
27     if (digitalRead(pirPin) == HIGH && commandReceived) {
28         // Ako je detektiran pokret, isključujemo zelenu diodu i uključujemo buzzer
29         digitalWrite(vPin, LOW);
30         tone(buzzerPin, 3500);
31         buzzerActive = true;
32         buzzerStartTime = millis(); // Postavljanje vremena početka trajanja buzzera
33         commandReceived = false;
34
35         // Slanje potvrde nazad skripti da je predajnik pronađen
36         Serial.println("Predajnik A pronađen");
37     } else {
38         // Ako nije detektiran pokret, provjeravamo naredbu sa računara
39         if (Serial.available() > 0) {
40             char receivedChar = Serial.read();
41
42             if (receivedChar == 'A') {
43                 // Uključujemo zelenu diodu i isključujemo buzzer
44                 digitalWrite(vPin, HIGH);
45                 noTone(buzzerPin);
46                 buzzerActive = true;
47                 commandReceived = true;
48             } else if (receivedChar == 'X') {
49                 // Ako je primljena naredba 'X', isključujemo zelenu diodu i buzzer
50                 digitalWrite(vPin, LOW);
51                 noTone(buzzerPin);
52                 buzzerActive = false;
53                 commandReceived = false;
54             }
55         }
56     }
57
58     // Provjeravamo vrijeme proteklo od početka trajanja buzzera
59     if (buzzerActive && millis() - buzzerStartTime >= 3000) {
60         // Ako je prošlo više od 3 sekunde, isključujemo buzzer
61         noTone(buzzerPin);
62         buzzerActive = false;
63     }
64 }

```

Slika 4. 13 Loop() funkcija

Treći logički blok u kodu odnosi se na funkciju "loop()", koja se izvršava neprekidno nakon što se funkcija "setup()" završi. Ova funkcija omogućava Arduinu da kontinuirano prati stanje senzora i reaguje na promjene u okruženju, što je ključno za interaktivne aplikacije. U funkciji "loop()", glavni dio koda je organizovan tako da prati dva ključna uslova: detekciju pokreta preko PIR senzora i prijem naredbi putem serijske komunikacije.

Prvo, unutar “if” uslova, ispituje se stanje PIR senzora. Ako PIR senzor detektuje pokret (njegovo stanje je “HIGH”) i ako je prethodno primljena odgovarajuća naredba za uključivanje predajnika (u slučaju demonstracije RGB LED dioda), tada se aktivira reakcija Arduina. Ova reakcija uključuje isključivanje predajnika (ili zelene LED diode koja indicira da je sistem aktivan) i aktivaciju buzzera koji emituje zvučni signal određene frekvencije pomoću funkcije “tone()”. Pored toga, bilježi se vrijeme kada je buzzer aktiviran koristeći funkciju “millis()”, što omogućava kontrolu trajanja zvuka buzzer od tri sekunde. Varijabla “commandReceived” služi kao indikator da li je primljena naredba, a ako jeste, ona se postavlja na “false” nakon što je buzzer aktiviran. Da bi ovaj blok bio izvršen, dva uslova moraju biti ispunjena: PIR senzor mora detektovati pokret, a varijabla “commandReceived” mora biti “true”. Ovo se dešava samo kada je Arduino već primio odgovarajuću naredbu, u ovom slučaju, karakter “A” od glavne skripte napisane u C# aplikaciji. Ova naredba uključuje zelenu LED diodu, koja simulira da je predajnik aktivan.

Ako uslovi iz prvog “if” bloka nisu ispunjeni (tj. pokret nije detektovan ili naredba nije primljena), program prelazi na “else” blok. U ovom dijelu se provjerava da li je na serijskoj komunikaciji primljen neki podatak, koristeći funkciju “Serial.available()”. Ako je taj podatak veći od 0, karakter koji je primljen smješta se u varijablu “receivedChar”, a zatim se ispituje da li je taj karakter “A”. Ako jeste, to znači da je skripta izabrala ovaj Arduino i tada se uključuje zelena dioda putem funkcije “digitalWrite()”. Takođe, funkcija “noTone()” osigurava da buzzer nije aktivan, dok se varijable “buzzerActive” i “commandReceived” postavljaju na “true”, što omogućava ulazak u prvi “if” blok kada se detektuje pokret. Kada je predajnik pronađen, preko serijske komunikacije Arduino šalje povratnu informaciju C# skripti, potvrđujući da je predajnik “A” lociran. Nakon toga, skripta šalje karakter “X”, koji Arduino interpretira unutar novog “if” bloka. Ako je primljena naredba “X”, zelena dioda i buzzer se isključuju, a sve relevantne varijable se postavljaju na “false”, čime se sistem vraća u početno stanje.

Prethodni postupak programiranja Arduina treba ponoviti za ostale Arduino ploče koje se koriste u sistemu. U ovom slučaju, jedina razlika u kodu bit će promjena vrijednosti varijable “receivedChar”. Umjesto karaktera “A”, za drugi Arduino treba postaviti karakter “B”, a za treći karakter “C”. Takođe, u dijelu gdje se putem serijske komunikacije šalje poruka o pronalasku predajnika, umjesto “Predajnik A pronađen”, treba promijeniti tekst u “Predajnik B pronađen” i “Predajnik C pronađen” kako bi se jasno razlikovali različiti predajnici. Na taj način, svaki Arduino prepoznaje specifične naredbe i šalje odgovarajuće potvrde nazad skripti na računaru.

Sada prelazimo na objašnjenje glavne skripte u C# koja je napisana u Visual Studio okruženju. Ova skripta ima ključnu ulogu u upravljanju sistemom, komunicirajući sa Arduinom putem serijske komunikacije. Ona šalje specifične naredbe prema Arduinu (poput slanja karaktera “A”, “B” ili “C” kako bi aktivirala određeni predajnik), prima povratne informacije o pronalasku predajnika. Za detaljnije objašnjenje, podijelit ćemo ovu skriptu u logičke blokove, kako bismo pojedinačno opisali sve njene funkcionalnosti i način na koji komunicira sa sistemom implementiranim na Arduinu.

Ova C# skripta počinje sa “using” direktivama koje uključuju potrebne biblioteke za funkcionalnosti koje će se koristiti u programu. Osnovna biblioteka “System” pruža pristup osnovnim funkcionalnostima C# jezika poput datuma, vremena i tipova podataka. Biblioteka “System.Windows.Forms” se koristi za kreiranje grafičkog korisničkog interfejsa aplikacija, uključujući prozore, dugmad i druge kontrole. “System.IO.Ports” omogućava rad sa serijskim portovima (neophodno za komunikaciju sa Arduino uređajem). Biblioteka “System.Threading” omogućava pauziranje i kreiranje dodatnih niti za paralelno izvršavanje. Biblioteka “System.Collections.Generic” omogućava korištenje generičkih kolekcija kao što su liste i mape. “System.Diagnostics.Eventing.Reader” koristi se za čitanje događaja iz sistemskih logova, iako ovdje nema direktnu primjenu i može biti uklonjeno ako nije potrebno. Nakon ovih direktiva, kod se organizuje unutar “namespace comp”, koji sadrži cijelu aplikaciju. Ovim se definiše imenovani prostor “comp”, koji grupiše sve klase i funkcije unutar tog prostora. Ovaj blok započinje deklaraciju glavne klase ‘Form1’, koja nasljeđuje ‘Form’ i predstavlja glavni prozor aplikacije. Sve dalje metode i varijable koje se odnose na GUI će biti definisane unutar ove klase.

```
1 using System;
2 using System.Windows.Forms;
3 using System.IO.Ports;
4 using System.Threading;
5 using System.Collections.Generic;
6 using System.Diagnostics.Eventing.Reader;
7
8 namespace comp
9 {
10    public partial class Form1 : Form
11    {
12        private SerialPort serialPort;
13        private List<string> targets;
14        private DateTime startTime;
15        private const int timeLimitSeconds = 300;
16        private string firstTransmitter = null;
17        private List<string> foundTargets = new List<string>(); // Lista za praćenje pronađenih predajnika
18        private bool competitionFinished = false; // Da omogući novo takmičenje
19
20        public Form1()
21        {
22            InitializeComponent();
23            // Inicijalizacija serijskog porta
24            serialPort = new SerialPort("COM7", 9600); // Promijenite 'COM7' u odgovarajući serijski port
25            // Povezivanje događaja klika na dugme s odgovarajućom metodom
26            button1.Click += button1_Click;
27        }
28
29        private void Form1_Load(object sender, EventArgs e)
30        {
31            // Vaša logika za inicijalizaciju forme
32            panel1.BackColor = Color.FromArgb(100, 0, 0, 0);
33        }
34
35        private void button1_Click(object sender, EventArgs e)
```

Slika 4. 14 Uvod u kod

U nastavku su navedene varijable koje su deklarisane unutar klase "Form1". Ove varijable služe za čuvanje ključnih informacija tokom rada programa, poput serijske komunikacije sa Arduino uređajem, praćenja vremena takmičenja, i stanja pronađenih predajnika. Svaka od njih ima specifičnu funkciju, koja će biti detaljnije objašnjena u sljedećem dijelu. Objekat "serialPort" predstavlja serijski port za komunikaciju sa Arduino uređajem. Lista "targets" sadrži ciljeve (predajnike) koje takmičar treba da pronađe. U ovom slučaju, to su "A", "B", i "C". Varijabla "startTime" bilježi početno vrijeme kada takmičenje počne. Koristi se za mjerjenje vremena trajanja takmičenja. "timeLimitSeconds" definiše maksimalno trajanje takmičenja (300 sekundi, to jest 5 minuta). Varijabla "firstTransmitter" bilježi koji je prvi predajnik pronađen tokom takmičenja. U početku je postavljena na "null". Kreira se prazna lista "foundTargets" koja će se koristiti za praćenje predajnika koji su pronađeni tokom takmičenja. Varijabla "competitionFinished" prati da li je takmičenje završeno ili ne. Na početku je postavljena na "false" kako bi se omogućilo pokretanje takmičenja.

Konstruktor klase "Form1" koji se poziva prilikom kreiranja objekta forme. U ovom dijelu se inicijalizuju GUI elementi i otvara serijski port za komunikaciju. "InitializeComponent()" je metoda generisana od strane Visual Studija koja inicijalizuje GUI elemente. "serialPort = new SerialPort("COM7", 9600)" postavlja serijski port (koristi se port "COM7" sa baud rate-om od 9600). "button1.Click += button1_Click" povezuje klik na dugme sa metodom "button1_Click", što znači da kada korisnik pritisne dugme, pokreće se određeni kod. "Form1_Load" metoda se poziva kada se forma učita. U njoj se postavljaju osnovne karakteristike GUI-a, kao što je boja pozadine.

Metoda "button1_Click" je zadužena za upravljanje logikom takmičenja kada korisnik pritisne dugme za početak takmičenja. Prvo ćemo naglasiti da se metoda "button1_Click" sastoji od dva glavna dijela: prvog dijela koji obuhvata inicijalizaciju i pripremu za takmičenje, i drugog dijela, koji uključuje "while" petlju koja upravlja glavnim tokom takmičenja. Ovdje ćemo se fokusirati na objašnjenje prvog dijela. (Slika 4. 15). "targets = new List<string> { "A", "B", "C" }" linija koda inicijalizuje listu ciljeva, koja sadrži tri predajnika označena karakterima "A", "B", i "C". Ovi predajnici predstavljaju ciljeve koje takmičari treba da pronađu. "Thread.Sleep(2000)" linija pauzira izvršavanje koda na dvije sekunde kako bi se serijski port stabilizovao prije nego što se nastavi sa slanjem ili primanjem podataka. Uslov "!serialPort.IsOpen" u "if" bloku provjerava da li je serijski port zatvoren. Ako je zatvoren, otvara se za komunikaciju. Serijski port omogućava komunikaciju između Arduina i računara. Linija "startTime = DateTime.Now" postavlja trenutni datum i vrijeme u varijablu "startTime", kako bi se zabilježilo kada je takmičenje započelo. U narednom "if" uslovu se provjerava da li je prethodno takmičenje završeno preko varijable "competitionFinished". Ako jeste, restartuju se varijable koje su korištene tokom tog takmičenja, a ako prethodno takmičenje nije završeno tekst u "label1" se takođe briše kako bi se osiguralo da nema starog sadržaja kao što je prikazano na slici 4. 15. Ovo je prvi dio metode "button1_Click", koji postavlja sve potrebne varijable i priprema sistem za početak takmičenja. Nakon ovoga slijedi "while" petlja, koja upravlja procesom pronalaženja predajnika prikazana na slici 4. 16.

```

33     private void button1_Click(object sender, EventArgs e)
34     {
35         targets = new List<string> { "A", "B", "C" }; // Inicijalizacija liste ciljeva
36         Thread.Sleep(2000); // Čekanje nekoliko sekundi kako bi se serijski port stabilizirao
37         if (!serialPort.IsOpen)
38         {
39             serialPort.Open(); // Otvorite serijski port
40         }
41         startTime = DateTime.Now; // Pokretanje vremena od početka takmičenja
42
43         if (competitionFinished)
44         {
45             // Resetovanje promenljivih za novo takmičenje
46             label1.Text = ""; // Brisemo sadrzaj labele
47             foundTargets.Clear(); // Resetujemo listu pronađenih predajnika
48             firstTransmitter = null; // Postavljamo varijablu za prvi predajnik
49             competitionFinished = false; // Postavljamo da takmičenje nije završeno
50         }
51         else
52         {
53             label1.Text = ""; // Brisemo sadrzaj labele
54         }
55
56         // Glavna petlja takmičenja
57         Random random = new Random();
58         while (foundTargets.Count < 3)

```

Slika 4. 15 Metod “button1_Click”

```

56         // Glavna petlja takmičenja
57         Random random = new Random();
58         while (foundTargets.Count < 3)
59         {
60             // Provjera vremena takmičenja
61             TimeSpan elapsedTime = DateTime.Now - startTime;
62             int remainingTimeSeconds = timeLimitSeconds - (int)elapsedTime.TotalSeconds;
63
64             if (remainingTimeSeconds <= 0)
65             {
66                 remainingTimeSeconds = 0; // Postavite na minimalnu vrijednost
67             }
68
69             // Postavljanje timeout-a na serijskom portu
70             serialPort.ReadTimeout = remainingTimeSeconds * 1000; // Pretvaranje vremena u milisekunde
71
72             // Odabir nasumičnog cilja iz preostalih ciljeva
73             string target = targets[random.Next(targets.Count)];
74
75             if (serialPort.IsOpen)
76             {
77                 // Slanje odabranog cilja na Arduina
78                 serialPort.WriteLine(target);
79                 Thread.Sleep(1);
80             }
81
82             try
83             {

```

Slika 4. 16 While petlja

Sada prelazimo na objašnjenje “while” petlje, koja predstavlja drugi glavni dio metode “button1_Click”. Ova petlja upravlja tokom takmičenja i ponavlja se sve dok nisu pronađena sva tri cilja. Kako je “while” petlja opširna, podijelićemo je na dva dijela: kod “while” petlje do “try” bloka i “try” i “catch” blokovi, koji upravljaju greškama i posebnim situacijama. U ovom objašnjenju ćemo se fokusirati na prvi dio, tj. kod unutar “while” petlje do “try” bloka. Uslov “foundTargets.Count < 3” označava početak glavne petlje koja se izvršava sve dok se ne pronađu sva tri cilja. Uslov petlje je da broj pronađenih predajnika bude manji od 3. Kada se pronađu sva tri cilja, petlja prestaje sa izvršavanjem. Unutar “while” petlje se izračunava koliko je vremena prošlo od početka takmičenja koristeći varijablu “elapsedTime”. Na osnovu toga se računa koliko vremena je ostalo do kraja takmičenja (remainingTimeSeconds). Ako preostalo vreme padne ispod nule, postavlja se na nulu kako bi se osigurala korektna logika dalje u programu. “serialPort.ReadTimeout = remainingTimeSeconds * 1000” linija koda postavlja “timeout” za serijski port na preostalo vrijeme (u milisekundama). Ovo znači da, ako takmičar ne pronađe sve predajnike i Arduino ne pošalje odgovor unutar tog vremenskog okvira (5 minuta), generisće se greška tipa “TimeoutException”. Preciznije rečeno, ako se takmičenje ne završi u roku od 5 minuta, takmičar će biti diskvalifikovan, što će se kasnije obraditi u “try-catch” bloku. “string target = targets[random.Next(targets.Count)]” linija bira nasumično jedan od preostalih ciljeva iz liste “targets”. Funkcija “random.Next(targets.Count)” generiše nasumični indeks koji se koristi za izbor jednog cilja (predajnika). Nakon toga se provjerava da li je serijski port otvoren. Ako jeste, odabrani cilj (karakter “target”) se šalje Arduino uređaju putem serijskog porta. Nakon toga, dolazi kratka pauza od 1 milisekunde pomoću “Thread.Sleep(1)” kako bi se obezbjedilo stabilno slanje podataka. Ovo je prvi dio logike unutar “while” petlje, koji se bavi izračunavanjem preostalog vremena, izborom nasumičnog cilja i slanjem tog cilja na Arduino. Nakon ovoga, slijede “try i catch” blokovi, koji obrađuju komunikaciju sa Arduinom i eventualne greške.

Dio koda unutar “try” bloka na slici 4. 17 obavlja ključne zadatke u vezi s primanjem i obradom podataka sa serijskog porta. Blok započinje provjerom da li je serijski port otvoren. Ovaj korak je neophodan jer podaci mogu biti pročitani samo ako je serijska komunikacija aktivna. Linija “string response = serialPort.ReadLine().Trim()” čita cijeli red podataka koji je stigao sa serijskog porta. Funkcija “ReadLine()” čita sve do kraja reda, dok “Trim()” uklanja eventualne praznine ili nevidljive znakove na početku ili kraju. Nakon toga se provjerava da li odgovor (string) koji je stigao počinje sa “Predajnik” i završava sa “pronadjen”, što označava da je predajnik pronađen. Također, “response.Split()[1]” izdvojiti će drugi dio odgovora koji bi trebao biti slovo (npr. “A”, “B” ili “C”) i provjeriti da li je jednak trenutnom cilju (target). Zatim se preko linije koda “double elapsedTimeSeconds = (DateTime.Now - startTime).TotalSeconds” izračunava koliko je vremena proteklo od početka takmičenja (startTime) do trenutka kada je predajnik pronađen. Vrijeme se mjeri u sekundama. Preko stringa “logMessage” se kreira poruka koja sadrži informaciju o pronađenom predajniku i vremenu potrebnom za njegovo lociranje. Format “{elapsedTimeSeconds:F2}” znači da će vrijeme biti prikazano sa dvije decimale. “label1.Text += logMessage” linija koda dodaje poruku u “label1”, tj. kontrolu koja prikazuje informacije korisniku na ekranu. Nakon dodavanja teksta u “label1”, potrebno je osvežiti prikaz kako bi se nova informacija prikazala odmah preko “label1.Refresh()”.

```

82     try
83     {
84         if (serialPort.IsOpen)
85         {
86             // Čitanje odgovora sa serijskog porta
87             string response = serialPort.ReadLine().Trim();
88
89             // Provjera je li pronađen novi cilj
90             if (response.StartsWith("Predajnik") && response.EndsWith("pronađen") && response.Split()[1] == target)
91             {
92                 // Izračunaj vrijeme proteklo od početka takmičenja do pronađaska trenutnog predajnika
93                 double elapsedTimeSeconds = (DateTime.Now - startTime).TotalSeconds;
94
95                 // Ispis informacija o pronađenom predajniku
96                 string logMessage = $"Pronaden predajnik {target} za {elapsedTimeSeconds:F2} sekundi.\n";
97
98                 // Dodaj informaciju u tekst Label kontrole
99                 label1.Text += logMessage;
100
101                // Osveži labelu da bi se prikazala ažuriranja
102                label1.Refresh();
103
104                // Ako je ovo prvi pronađeni predajnik, spremi ga kao prvi predajnik
105                if (foundTargets.Count == 0)
106                {
107                    firstTransmitter = target;
108                }
109
110                // Dodaj pronađeni predajnik u listu pronađenih ciljeva
111                foundTargets.Add(target);
112
113                // Ukloni pronađeni cilj iz liste ciljeva
114                targets.Remove(target);
115            }
}

```

Slika 4. 17 Try blok unutar while petlje

Nakon toga provjeravamo da li je ovo prvi pronađeni predajnik. Ako je lista "foundTargets" prazna, što znači da još nije pronađen nijedan predajnik, trenutni cilj (target) se postavlja kao "firstTransmitter", odnosno prvi pronađeni predajnik i dodaje se u listu "foundTargets" koja prati sve uspješno locirane predajnike preko komande "foundTargets.Add(target)". Nakon što je predajnik pronađen, uklanja se iz liste ciljeva (targets), kako bi se spriječilo njegovo ponovno biranje preko linije "targets.Remove(target)". Na slici 4. 18 je prikazan nastavak "try" bloka. Ovaj nastavak bloka obuhvata logiku koja se odnosi na završetak takmičenja i restartovanje sistema ako su svi predajnici pronađeni. U uslovu se provjerava da li su već pronađena dva predajnika, a u listi ciljeva ostao je još samo jedan. Ukoliko je taj uslov zadovoljen prvi predajnik (firstTransmitter) se ponovo dodaje u listu ciljeva kako bi takmičar ponovo mogao dobiti prvi predajnik ili preostali zadnji predajnik kao naredni cilj. Drugi uslov provjerava da li su pronađena sva tri predajnika. Ukoliko su pronađena, serijski port se zatvara ukoliko je bio otvoren čime se osigurava da se serijska komunikacija prekine nakon što je takmičenje završeno. Zatim se računa i prikazuje ukupno vrijeme koje je proteklo od početka takmičenja do završetka.

```
// Ako su pronadjeni svi ciljevi osim zadnjeg
if (foundTargets.Count == 2 && targets.Count == 1)
{
    // Dodaj prvog izabranog predajnika nazad u listu ciljeva prije biranja posljednjeg preostalog predajnika
    targets.Add(firstTransmitter);
}

if (foundTargets.Count == 3)
{
    // Zatvaranje serijskog porta nakon završetka takmičenja
    if (serialPort.IsOpen)
    {
        serialPort.Close();
    }

    // Ispis ukupnog vremena trajanja takmičenja
    double totalTime = (DateTime.Now - startTime).TotalSeconds - 2;
    label1.Text += $"Ukupno vrijeme takmičenja: {totalTime:F2} sekundi\n";

    // Obavijest o završetku takmičenja
    var result = MessageBox.Show("Takmičenje je završeno! Želite li pokrenuti novo takmičenje?", "Kraj takmičenja", MessageBoxButtons.YesNo);
    if (result == DialogResult.Yes)
    {
        competitionFinished = true; // Postavljamo da takmičenje nije završeno
        button1_Click(sender, e); // Ponovno pokretanje takmičenja
        return; // Izlaz iz metode kako bi se sprijećilo izvršavanje ostatka koda
    }
    else if (result == DialogResult.No)
    {
        label1.Text = ""; // Brisemo sadržaj labela
        Environment.Exit(0); // Pravilno zatvaranje aplikacije
    }
}
```

Slika 4. 18 Try blok nastavak

Vrijeme se računa kao razlika između trenutnog vremena (DateTime.Now) i vremena kada je takmičenje počelo (startTime), a zatim se od te vrijednosti oduzima dvije sekunde, što je urađeno zbog kašnjenja na početku takmičenja. Rezultat se ispisuje na "label1" u formatu sa dvije decimale ({totalTime:F2}). Kada se takmičenje završi, korisniku se prikazuje dijalog putem "MessageBox.Show()" sa pitanjem da li želi pokrenuti novo takmičenje. Dijalog ima dvije opcije: "Yes" (Da) i "No" (Ne). Ako korisnik odabere opciju "Yes", postavlja se da takmičenje nije završeno (competitionFinished = true), što omogućava restartovanje varijabli za novo takmičenje. Metoda "button1_Click()" se poziva ponovo, što simulira pritisak na dugme i pokreće novo takmičenje. "return" zaustavlja daljnje izvršavanje trenutne metode kako bi se spriječilo izvršavanje koda ispod. Ako korisnik odabere "No", aplikacija se zatvara. Prvo se briše sadržaj "label1", a zatim se koristi "Environment.Exit(0)" da bi se aplikacija pravilno zatvorila. Ova naredba osigurava da se svi resursi aplikacije oslobole i aplikacija prestane sa radom.

```

    catch (TimeoutException)
    {
        // Poruka o diskvalifikaciji
        label1.Text += "Diskvalifikacija: Vrijeme je isteklo, niste pronašli sve predajnike u roku.\n";
        label1.Refresh();

        if (serialPort.IsOpen)
        {
            serialPort.Write("X"); // Slanje naredbe Arduino-u da isključi predajnik
            serialPort.Close();
        }

        //Prikazivanje dialoga za novo takmičenje
        var result = MessageBox.Show("Takmičenje je završeno! Želite li pokrenuti novo takmičenje?", "Kraj takmičenja");
        if (result == DialogResult.Yes)
        {
            competitionFinished = true; // Postavljamo da takmičenje nije završeno
            button1_Click(sender, e); // Ponovno pokretanje takmičenja
        }
        else if (result == DialogResult.No)
        {
            if (serialPort.IsOpen)
            {
                serialPort.Close();
            }
            Environment.Exit(0);
            // Pravilno zatvaranje aplikacije
        }
    }
}

```

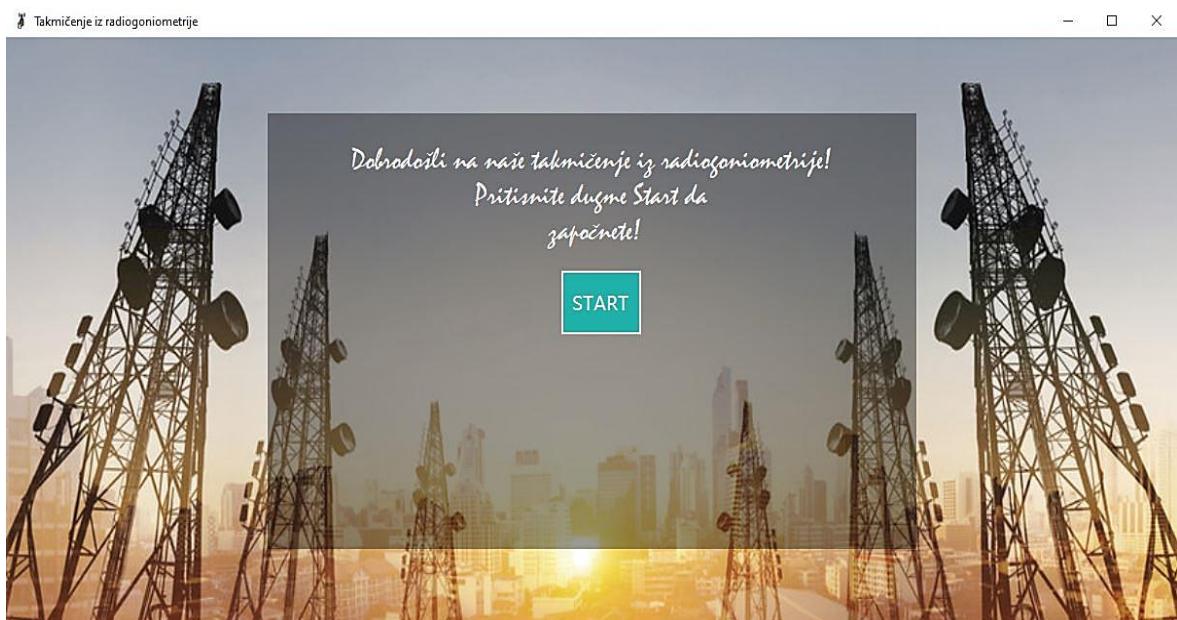
Slika 4. 19 Catch blok

Ovaj “catch” blok je vrlo sličan dijelu koda u “try” bloku, ali služi isključivo za slučaj diskvalifikacije zbog isteka vremena (kada prođe 5 minuta, a takmičar nije pronašao sve predajnike). Kada dođe do “TimeoutException”, prikazuje se poruka na “label1” koja informiše takmičara da je diskvalifikovan jer nije pronašao sve predajnike u zadatom roku. Nakon diskvalifikacije, aplikacija šalje komandu “X” Arduinu kako bi isključila predajnike preko linije “serialPort.Write("X")”. Sve ostalo je gotovo identično kao i kod u “try” bloku (korisniku se nudi opcija da započne novo takmičenje ili zatvori aplikaciju). Na kraju ovog “catch” bloka završava se glavna skripta koja se izvršava na računaru. Ova skripta upravlja komunikacijom sa Arduino uređajem putem serijskog porta, prati napredak takmičenja, upravlja vremenom i pruža korisnički interfejs koji omogućava pokretanje novog takmičenja ili zatvaranje aplikacije. Nakon što smo detaljno objasnili sve aspekte glavne skripte, konačno možemo preći na demonstraciju takmičenja. S obzirom na to da je sve ispravno konfigurisano i programirano, možemo očekivati da će sistem funkcionisati kako je predviđeno, omogućavajući takmičarima da efikasno lociraju predajnike i učestvuju u takmičenju.

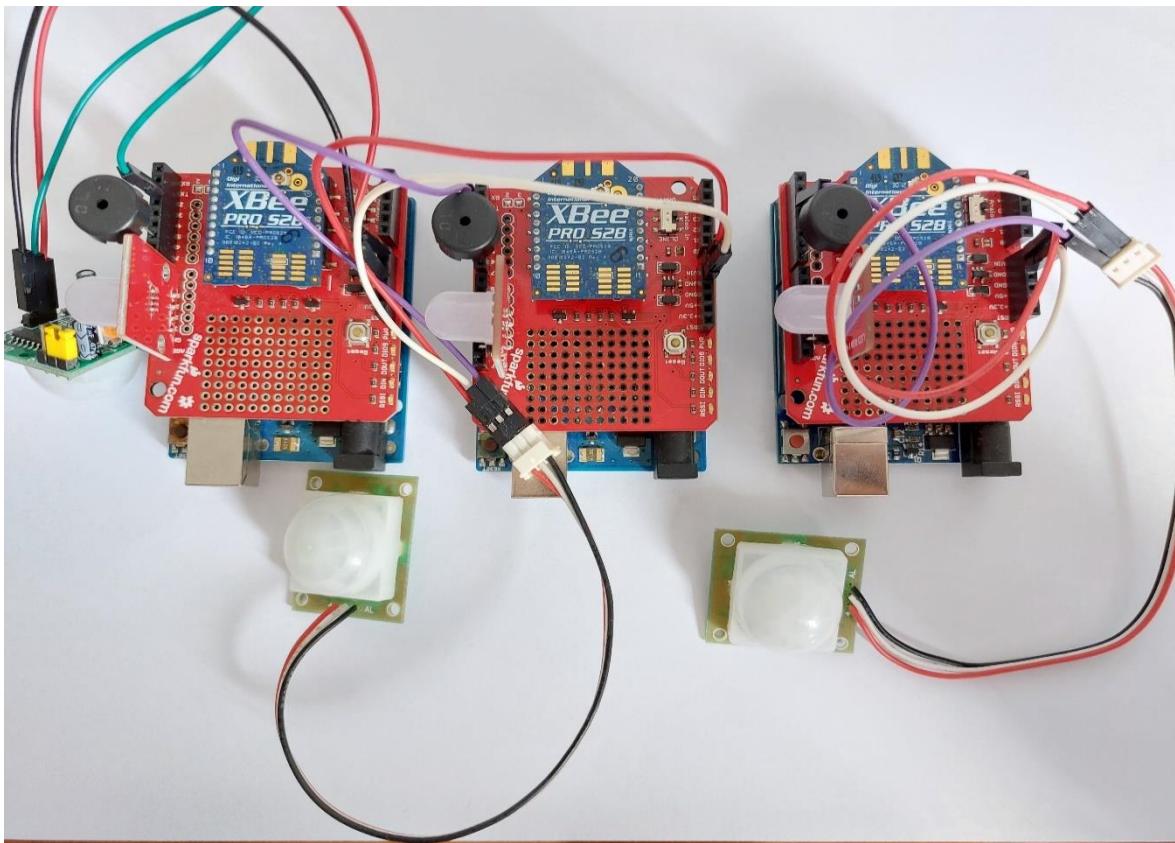
Takmičenje započinje otvaranjem aplikacije pod nazivom "SignalSeekComp". Kada korisnik pokrene aplikaciju, na ekranu se prikazuje grafički korisnički interfejs koji pozdravlja korisnika i uvodi ga u svijet takmičenja iz radioorientacije. Interfejs sadrži dugme "Start", koje omogućava jednostavno pokretanje takmičenja. (Slika 4. 20). Ova intuitivna platforma omogućava korisniku brz pristup svim funkcionalnostima aplikacije i olakšava početak takmičenja.

Prvi korak nakon pokretanja sistema je uključivanje svih Arduino uređaja, tj. dovesti napajanje na njih kako bi sistem bio spreman za funkcionisanje. Kada se napajanje doveđe na Arduino, na XBee ekspanzijskoj ploči će svijetliti "PWR" dioda, koja pokazuje da je uređaj uključen i napajan. Također, svijetlit će i "DIO5" dioda. "PWR" dioda signalizira da je XBee modul pravilno napajan, dok "DIO5" dioda korisniku pokazuje da taj Arduino sa XBee modulom trenutno nije spojen na ZigBee mrežu. Ova dioda svijetli neprestano sve dok se modul ne poveže na mrežu, čime pruža vizualnu povratnu informaciju o statusu veze.

Svi Arduino uređaji povezuju se na ZigBee mrežu tek kada se dongle s koordinatorom spoji na USB port računara. Tada će na dongleu zasvijetliti "PWR" dioda, što signalizira da je dongle pod naponom. Nakon što je koordinator napajan, započinje proces uspostavljanja mreže, kako je prethodno objašnjeno u poglavljju o XBee komunikacionim modulima. Potrebno je sačekati do 5 sekundi kako bi se svi uređaji sinhronizovali i povezali na ZigBee mrežu. To ćemo znati po tome što će tada "DIO5" dioda na XBee ekspanzijskoj ploči početi svijetliti s intervalom od jedne sekunde, tj. dioda će se uključivati i isključivati, što signalizira da je taj Arduino sa XBee modulom uspješno povezan na mrežu i spreman za upotrebu. Nakon što se uvjerimo da su svi uređaji uspješno povezani na mrežu i da "DIO5" diode na svim uređajima trepere u pravilnim intervalima, možemo započeti takmičenje. Spremnost uređaja signalizira da je ZigBee mreža uspostavljena, i sada je sistem u potpunosti funkcionalan za takmičenje.



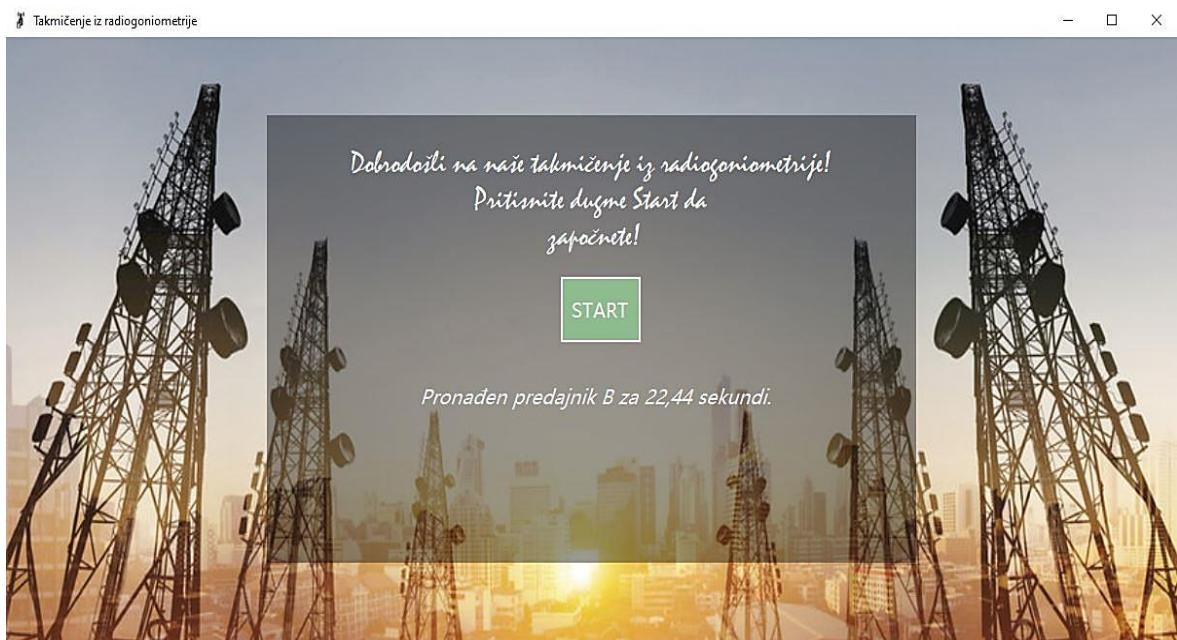
Slika 4. 20 Grafički korisnički interfejs takmičenja



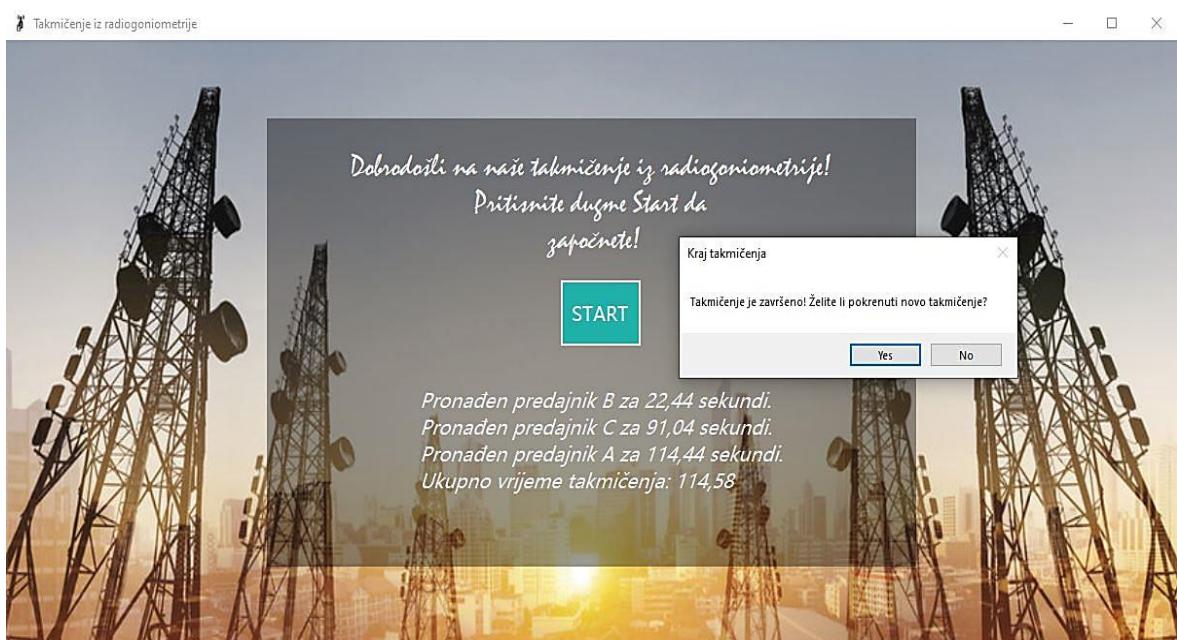
Slika 4. 21 Prikaz sva tri Arduino uređaja u sistemu

Važno je napomenuti da prije nego što korisnik pokrene takmičenje, treba provjeriti na kojem portu je spojen dongle sa koordinatorom na računaru, jer aplikacija će raditi samo ako je port postavljen na "COM7". To se može provjeriti tako što se ode na "My Computer", zatim na "Properties", pa se izabere "Device Manager" i odabere "Ports". U dijelu "Ports" potrebno je provjeriti na kojem portu se nalazi koordinator. Ako nije postavljen na "COM7", treba otvoriti "USB Serial Port", odabratи "Port Settings", a zatim izabrati "Advanced opciju". U toj sekciji može se promijeniti broj COM porta na "COM7", a nakon toga treba sačuvati promjene.

U prilogu su postavljene slike 4. 22 i 4. 23, simulacija jednog takmičenja, koje prikazuju rezultate nakon pronalaska svih predajnika. Ova slika demonstrira kako sistem funkcioniše u realnom vremenu, uključujući prikaz vremena potrebnog za pronalazak svakog predajnika i ukupnog vremena takmičenja. Ovaj primjer služi kao vizuelni prikaz uspješne izvedbe sistema i završetka takmičenja. Ovom simulacijom završava se prikaz implementacije sistema za takmičenje iz radiogoniometrije. Kroz prikazane korake, uključujući konfiguraciju uređaja, uspostavu ZigBee mreže, te uspješnu simulaciju takmičenja, demonstrirano je kako sistem funkcioniše u praksi. Cjelokupan proces pokazuje spremnost sistema za korištenje u stvarnim takmičenjima, a dodatne nadogradnje i prilagodbe mogu biti izvedene prema potrebama budućih korisnika ili specifičnim zahtjevima.



Slika 4. 22 Simulacija takmičenja (predajnik B pronađen)



Slika 4. 23 Završena simulacija takmičenja

Zaključak

Zaključak ovog rada ističe važnost razvoja sistema za organizaciju i provođenje takmičenja iz radiogoniometrije, koji omogućava automatizaciju ključnih procesa, čime se povećava preciznost i objektivnost samog takmičenja. Implementirani sistem zasnovan na XBee komunikacionim modulima i Arduino platformi pokazao se kao efikasno i pouzdano rješenje za bežičnu komunikaciju između uređaja, omogućavajući stabilan rad u stvarnom vremenu.

Tokom rada na sistemu uočeni su određeni izazovi, prvenstveno u konfiguraciji ZigBee mreže i serijske komunikacije, ali su ti problemi uspješno prevaziđeni kroz primjenu odgovarajućih softverskih i hardverskih alata. Na kraju, testiranje sistema je pokazalo da razvijeno rješenje omogućava jednostavno upravljanje takmičenjem, precizno bilježenje rezultata i pouzdano funkcionisanje uređaja.

Kao buduća unaprjeđenja sistema, ostavljena za naredne generacije, mogla bi se implementirati dodatna funkcionalnost koja bi omogućila automatsko slanje rezultata u Excel dokument. Na taj način, ime i prezime takmičara zajedno s njegovim rezultatom bilo bi direktno evidentirano, a podaci automatski sortirani prema ostvarenom vremenu. Time bi organizatori odmah po završetku takmičenja imali spremne, sortirane rezultate, što bi dodatno ubrzalo i olakšalo cijeli proces.

Još jedna moguća nadogradnja je uvođenje naprednijih senzora kako bi se smanjila mogućnost lažnih detekcija (što je ujedno i nedostatak ovog sistema), kao što su situacije kada PIR senzor, koji detektuje pokret, reaguje na prolazak životinja poput ptica, što bi moglo narušiti regularnost takmičenja. Ova i slična unaprjeđenja mogla bi dodatno povećati preciznost sistema i njegovu upotrebljivost u različitim uslovima.

U konačnici, ovaj rad predstavlja značajan doprinos automatizaciji radiogoniometrijskih takmičenja, otvarajući nove mogućnosti za dalji razvoj i unaprjeđenje sistema u ovoj oblasti. Ova implementacija je posebno značajna jer reflektuje savremene tendencije u organizaciji takmičenja, gdje se sve više teži ka automatizaciji procesa. U skladu sa vremenom u kojem živimo, ovakav pristup osigurava da radiogoniometrija prati tehničke trendove, čime se povećava efikasnost i preciznost, a samim tim i atraktivnost takmičenja.

Pored toga, ovakav sistem može pomoći radioamaterima, koji ulažu veliki trud i zalaganje u ovu oblast, da privuku veći broj mladih zainteresovanih za radiogoniometriju i tehničke nauke. Automatizovan i modernizovan način provođenja takmičenja može doprinijeti popularizaciji ovog hobija, a samim tim i osigurati njegovu budućnost kroz nove generacije takmičara i radioamatera.

Bibliografija

- [1] "What is Arduino Uno? Robu.in Your Ideas, Our Parts," 12 Avgust 2024. [Online]. Available: <https://robu.in/what-is-arduino-uno/>.
- [2] "What is Arduino UNO? RS DESIGNSPARK," RS DESIGNSPARK, 12 Avgust 2024. [Online]. Available: <https://www.rs-online.com/designspark/what-is-arduino-uno-a-getting-started-guide>.
- [3] "Farnell," 14 Avgust 2024. [Online]. Available: <https://www.farnell.com/datasheets/1682209.pdf>.
- [4] "Svet kompjutera," 25. Avgust 2024. [Online]. Available: <https://www.sk.rs/arhiva/clanak/13678/arduino-uno>.
- [5] "Sigmaelectronica," 28 Avgust 2024. [Online]. Available: https://www.sigmaelectronica.net/wp-content/uploads/2012/10/A000066_Web.pdf.
- [6] "Wikipedia," 4 Septembar 2024. [Online]. Available: <https://en.wikipedia.org/wiki/XBee>.
- [7] "Digi," 4 Septembar 2024. [Online]. Available: <https://www.digi.com/blog/post/xbee-buying-guide>.
- [8] R. Faludi, Building Wireless Sensor Networks, O'Reilly Media.
- [9] O. Janković, "infoteh," 7 Septembar 2024. [Online]. Available: <https://infoteh.etf.ues.rs.ba/zbornik/2012/radovi/STS/STS-9.pdf>.
- [10] V. V. Rava Filipović, "infoteh," 7 Septembar 2024. [Online]. Available: <https://infoteh.etf.ues.rs.ba/zbornik/2010/radovi/B-II/B-II-11.pdf>.
- [11] O. Janković, "infoteh," 7 Septembar 2024. [Online]. Available: <https://infoteh.etf.ues.rs.ba/zbornik/2010/radovi/F/F-10.pdf>.
- [12] "Wikipedia," 7 Septembar 2024. [Online]. Available: https://en.wikipedia.org/wiki/IEEE_802.15.4.
- [13] "electricalfundablog," 09 Septembar 2024. [Online]. Available: <https://electricalfundablog.com/zigbee-technology-architecture/>.
- [14] "ReasearchGate," 13 Septembar 2024. [Online]. Available: https://www.researchgate.net/figure/Structure-of-the-superframe-in-a-beacon-enabled-ZigBee-network_fig1_221628346.

- [15] "Digi," 14 Septembar 2024. [Online]. Available:
<https://www.digi.com/resources/documentation/digidocs/pdfs/90000976.pdf>.
- [16] "Radioklub Pazin," 14 Septembar 2024. [Online]. Available:
<https://www.rkp.hr/active/>.
- [17] "OVSV," 14 Septembar 2024. [Online]. Available:
<https://www.oevsv.at/funkbetrieb/ardf/technik/>.
- [18] "QRZ.com.hr," 14 Septembar 2024. [Online]. Available: https://www.qrz.com.hr/wp-content/uploads/2011/05/uvod_u_radiogoniometriju.pdf.
- [19] B. Schneider, "DL4CU," 15 Septembar 2024. [Online]. Available:
<http://dl4cu.de/ardf.html>.
- [20] "qsl," 15 Septembar 2024. [Online]. Available: <https://www.qsl.net/on7yd/atx80.htm>.
- [21] "DroneBot Workshop," 16 Septembar 2024. [Online]. Available:
<https://dronebotworkshop.com/using-pir-sensors-with-arduino-raspberry-pi/>.
- [22] "Last Minute Engineers," 16 Septembar 2024. [Online]. Available:
<https://lastminuteengineers.com/pir-sensor-arduino-tutorial/>.
- [23] "electricity-magnetism," 20 Septembar 2024. [Online]. Available:
<https://www.elprocus.com/buzzer-working-applications/>.
- [24] "FHD ELECTRONICS CORPORATION," 20 Septembar 2024. [Online]. Available:
https://fhdmfg.com/news/news_detail/piezo-buzzers.
- [25] "elprocus," 20 Septembar 2024. [Online]. Available:
<https://www.elprocus.com/buzzer-working-applications/>.
- [26] "ENGINEERSGARAGE," 20 Septembar 2024. [Online]. Available:
<https://www.engineersgarage.com/buzzers-types-transducer-indicator-piezo-magnetic/>.
- [27] "Digi.com," 24 Septembar 2024. [Online]. Available:
<https://www.digi.com/resources/documentation/digidocs/pdfs/90001458-13.pdf>.
- [28] "arduino.cc," 24 Septembar 2024. [Online]. Available:
<https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2/>.
- [29] "AI Skills Challange," 25 Septembar 2024. [Online]. Available:
<https://learn.microsoft.com/en-us/visualstudio/windows/?view=vs-2022>.
- [30] "sparkfun," 07 Oktobar 2024. [Online]. Available:
<https://learn.sparkfun.com/tutorials/xbee-shield-hookup-guide>.

Dodatak

Kod za Arduino:

```
const int pirPin = 2;           // Pin za PIR senzor
const int vPin = 11;            // Pin za LED RGB
const int redPin = 12;          // Pin za crvenu diodu
const int bluePin = 13;         // Pin za plavu diodu
const int buzzerPin = 6;        // Pin za buzzer
const int buzzerGnd = 7;        // Pin za GND buzzera
bool buzzerActive = false;      // Stanje buzzera
bool commandReceived = false;    // Da li je primljena naredba
unsigned long buzzerStartTime = 0; // Vrijeme početka trajanja buzzera

void setup() {
pinMode(pirPin, INPUT);
pinMode(vPin, OUTPUT);
pinMode(redPin, OUTPUT);
pinMode(bluePin, OUTPUT);
pinMode(buzzerPin, OUTPUT);
pinMode(buzzerGnd, OUTPUT);
digitalWrite(buzzerGnd, LOW);
digitalWrite(redPin, LOW);       // Isključimo crvenu diodu
digitalWrite(bluePin, LOW);      // Isključimo plavu diodu
Serial.begin(9600);             // Pokretanje serijske komunikacije
}

void loop() {
// Provjera da li je detektiran pokret i da li je primljena naredba
if (digitalRead(pirPin) == HIGH && commandReceived) {
// Ako je detektiran pokret, isključujemo zelenu diodu i uključujemo buzzer
digitalWrite(vPin, LOW);
tone(buzzerPin, 3500);
buzzerActive = true;
buzzerStartTime = millis();      // Postavljanje vremena početka trajanja buzzera
commandReceived = false;
// Slanje potvrde nazad skripti da je predajnik pronađen
Serial.println("Predajnik A pronađen");
} else {
```

```
// Ako nije detektiran pokret, provjeravamo naredbu sa računara
if (Serial.available() > 0) {
    char receivedChar = Serial.read();
    if (receivedChar == 'A') {
        // Uključujemo zelenu diodu i isključujemo buzzer
        digitalWrite(vPin, HIGH);
        noTone(buzzerPin);
        buzzerActive = true;
        commandReceived = true;
    } else if (receivedChar == 'X') {
        // Ako je primljena naredba 'X', isključujemo zelenu diodu i buzzer
        digitalWrite(vPin, LOW);
        noTone(buzzerPin);
        buzzerActive = false;
        commandReceived = false;
    }
}
}

// Provjeravamo vrijeme proteklo od početka trajanja buzzera
if (buzzerActive && millis() - buzzerStartTime >= 3000) {
    // Ako je prošlo više od 3 sekunde, isključujemo buzzer
    noTone(buzzerPin);
    buzzerActive = false;
}
```

Kod za C#:

```
using System;
using System.Windows.Forms;
using System.IO.Ports;
using System.Threading;
using System.Collections.Generic;
using System.Diagnostics.Eventing.Reader;

namespace comp
{
    public partial class Form1 : Form
    {
        private SerialPort serialPort;
```

```
private List<string> targets;
private DateTime startTime;
private const int timeLimitSeconds = 300;
private string firstTransmitter = null;
// Lista za praćenje pronađenih predajnika
private List<string> foundTargets = new List<string>();
private bool competitionFinished = false; // Da omogući novo takmičenje

public Form1()
{
    // Inicijalizacija serijskog porta
    InitializeComponent();
    // Promijenite 'COM7' u odgovarajući serijski port
    serialPort = new SerialPort("COM7", 9600);
    // Povezivanje događaja klika na dugme s odgovarajućom metodom
    button1.Click += button1_Click;
}
private void Form1_Load(object sender, EventArgs e)
{
    // Vaša logika za inicijalizaciju forme
    panel1.BackColor = Color.FromArgb(100, 0, 0, 0);
}
private void button1_Click(object sender, EventArgs e)
{
    targets = new List<string> { "A", "B", "C" }; // Inicijalizacija liste ciljeva
    // Čekanje nekoliko sekundi kako bi se serijski port stabilizirao
    Thread.Sleep(2000);
    if (!serialPort.IsOpen)
    {
        serialPort.Open(); // Otvorite serijski port
    }
    startTime = DateTime.Now; // Pokretanje vremena od početka takmičenja

    if (competitionFinished)
    {
        // Resetovanje promenljivih za novo takmičenje
        label1.Text = ""; // Brisemo sadrzaj labele
        foundTargets.Clear(); // Resetujemo listu pronađenih predajnika
        firstTransmitter = null; // Postavljamo varijablu za prvi predajnik
        competitionFinished = false; // Postavljamo da takmičenje nije završeno
    }
}
```

```
else
{
    label1.Text = ""; // Brisemo sadrzaj labele
}

// Glavna petlja takmičenja
Random random = new Random();
while (foundTargets.Count < 3)
{
    // Provjera vremena takmičenja
    TimeSpan elapsedTime = DateTime.Now - startTime;
    int remainingTimeSeconds = timeLimitSeconds -
(int)elapsedTime.TotalSeconds;

    if (remainingTimeSeconds <= 0)
    {
        remainingTimeSeconds = 0; // Postavite na minimalnu vrijednost
    }

    // Postavljanje timeout-a na serijskom portu
    // Pretvaranje vremena u milisekunde
    serialPort.ReadTimeout = remainingTimeSeconds * 1000;

    // Odabir nasumičnog cilja iz preostalih ciljeva
    string target = targets[random.Next(targets.Count)];

    if (serialPort.IsOpen)
    {
        // Slanje odabranog cilja na Arduina
        serialPort.Write(target);
        Thread.Sleep(1);
    }

    try
    {
        if (serialPort.IsOpen)
        {
            // Čitanje odgovora sa serijskog porta
            string response = serialPort.ReadLine().Trim();

            // Provjera je li pronadjen novi cilj
        }
    }
}
```

```
if (response.StartsWith("Predajnik") && response.EndsWith("pronadjen")
&& response.Split()[1] == target)
{
    // Izračunaj vrijeme proteklo od početka takmičenja do pronađaska
    trenutnog predajnika
    double elapsedTimeSeconds = (DateTime.Now -
startTime).TotalSeconds;

    // Ispis informacija o pronađenom predajniku
    string logMessage = $"Pronađen predajnik {target} za
{elapsedTimeSeconds:F2} sekundi.\n";

    // Dodaj informaciju u tekst Label kontrole
    label1.Text += logMessage;

    // Osveži labelu da bi se prikazala ažuriranja
    label1.Refresh();

    // Ako je ovo prvi pronađeni predajnik, spremi ga kao prvi predajnik
    if (foundTargets.Count == 0)
    {
        firstTransmitter = target;
    }

    // Dodaj pronađeni predajnik u listu pronađenih ciljeva
    foundTargets.Add(target);

    // Ukloni pronađeni cilj iz liste ciljeva
    targets.Remove(target);
}

// Pauza između svake iteracije
Thread.Sleep(2000);

// Ako su pronađeni svi ciljevi osim zadnjeg
if (foundTargets.Count == 2 && targets.Count == 1)
{
    // Dodaj prvog izabranog predajnika nazad u listu ciljeva prije biranja
    posljednjeg preostalog predajnika
    targets.Add(firstTransmitter);
}
```

```

if (foundTargets.Count == 3)
{
    // Zatvaranje serijskog porta nakon završetka takmičenja
    if (serialPort.IsOpen)
    {
        serialPort.Close();
    }

    // Ispis ukupnog vremena trajanja takmičenja
    double totalTime = (DateTime.Now - startTime).TotalSeconds - 2;
    label1.Text += $"Ukupno vrijeme takmičenja: {totalTime:F2}"
    sekundi\n";

    // Obavijest o završetku takmičenja
    var result = MessageBox.Show("Takmičenje je završeno! Želite li
pokrenuti novo takmičenje?", "Kraj takmičenja", MessageBoxButtons.YesNo);
    if (result == DialogResult.Yes)
    {
        competitionFinished = true; // Postavljamo da takmičenje nije završeno
        button1_Click(sender, e); // Ponovno pokretanje takmičenja
        return; // Izlaz iz metode kako bi se spriječilo izvršavanje ostatka koda
    }
    else if (result == DialogResult.No)
    {
        label1.Text = ""; // Brisemo sadrzaj labele
        Environment.Exit(0); // Pravilno zatvaranje aplikacije
    }
}
catch (TimeoutException)
{
    // Poruka o diskvalifikaciji
    label1.Text += "Diskvalifikacija: Vrijeme je isteklo, niste pronašli sve
predajnike u roku.\n";
    label1.Refresh();

    if (serialPort.IsOpen)
    {
        serialPort.Write("X"); // Slanje naredbe Arduino-u da isključi predajnik
        serialPort.Close();
    }
}

```

```
    }
    //Prikazivanje dialoga za novo takmičenje
    var result = MessageBox.Show("Takmičenje je završeno! Želite li pokrenuti
novo takmičenje?", "Kraj takmičenja", MessageBoxButtons.YesNo);
    if (result == DialogResult.Yes)
    {
        competitionFinished = true; // Postavljamo da takmičenje nije završeno
        button1_Click(sender, e); // Ponovno pokretanje takmičenja
    }
    else if (result == DialogResult.No)
    {
        if (serialPort.IsOpen)
        {
            serialPort.Close();
        }
        Environment.Exit(0);
        // Pravilno zatvaranje aplikacije
    }
}
}
}
}
}
```