**JELGAVAS TEHNIKUMS**

**Aleks Markuss Balodis**

**Datu ieguve un apmaiņa izmantojot mikrokontrolierus**

**Kvalifikācijas darbs**

**kvalifikācijas ieguvei**

**programmēšanas tehniķis**

**Darba izpildītājs:  410.gr.izgl. Aleks Markuss Balodis**

***paraksts, datums***

**Jelgava**

**2022**

ANOTĀCIJA

Aleks Markuss Balodis – kvalifikācijas darbs.

Mikrokontrolieri plaši tiek izmantoti mūsdienās dažādās ierīcēs, lai veiktu vienkāršas funkcijas, kur nav nepieciešama liela veiktspēja. Darba mērķis ir ievākt datus par laikapstākļiem ar mikrokontrolieriem un izmantojot bezvadu komunikāciju nosūtīt tos uz serveri, kas tos saglabās datubāzē un attēlos lietotājam viegli pārredzamā veidā mājaslapā.

Mikrokontrolieri paši nespēj izmērīt temperatūru, gaismas daudzumu vai nosūtīt informāciju bez vadiem, bet vien apstrādāt loģiku, tāpēc tiek pievienoti dažādi moduļi vai komponentes. Komponentes kā *LED* tiek pievienotas izmantojot *GPIO* kontaktus, bet moduļi kā, piemēram nRF24L01 bezvadu modulis, izmanto komunikācijas protokolu SPI.

Darba veikšanai nepieciešams būs izmantot dažādas programmēšanas valodas kā *C*. Koda rediģēšanai tiks izmantots *Visual Studio Code* un *Arduino IDE*. Versiju kontroles sistēmai tiks izmantots *GIT*.

ANNOTATION

Aleks Markuss Balodis – qualification paper.

Microcontrollers are widely used today in various devices to perform basic functions which do not require high performance. The purpose of my final year project is to harvest weather data using microcontrollers and then using wireless communication send the harvested data to a server, which would store the data in a database.

Microcontrollers by themselves cannot measure temperature, light or send information wirelessly. They can only process logic, that is why extra modules or components are added to them. Components like LEDs are added using GPIO pins, but modules like the nRF24L01 wireless module use a communication protocol, in this case SPI.

For this project I will need to use a programming language like C. To edit the code, I will be using Visual Studio Code and the Arduino IDE. For version control I will be using GIT.

Saturs

[Ievads 5](#_Toc105021006)

[1. PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA 6](#_Toc105021007)

[1.1 Ievads 6](#_Toc105021008)

[1.1.1 Dokumentācijas mērķis 6](#_Toc105021009)

[1.1.2 Produkta sfēra 6](#_Toc105021010)

[1.1.3 Dokumentācijas pārskats 6](#_Toc105021011)

[1.2 Vispārīgs apraksts 6](#_Toc105021012)

[1.2.1 Produkta funkcijas 6](#_Toc105021013)

[1.2.2 Produkta pielietojums 6](#_Toc105021014)

[1.2.3 Programmatūras gaita 6](#_Toc105021015)

[1.2.4 Mērķauditorija 6](#_Toc105021016)

[1.2.5 Pienākumu sadale 6](#_Toc105021017)

[1.3 Funkcionālās prasības 6](#_Toc105021018)

[2. IZSTRĀDEI IZMANTOTĀS TEHNOLOĢIJAS UN PROGRAMMATŪRA 9](#_Toc105021019)

[2.1 Programmatūras izstrādes vide 9](#_Toc105021020)

[2.2 Visual Studio Code izmantotie paplašinājumi 9](#_Toc105021021)

[2.2.1 PlatformIO 9](#_Toc105021022)

[2.2.2 C / C++ Syntax 9](#_Toc105021023)

[3. PRODUKTA KOMPONENŠU APRAKSTS 10](#_Toc105021024)

[2.3 Mikrokontrolieri 10](#_Toc105021025)

[2.3.1 Mikrokontroliera koda funkcijas 10](#_Toc105021026)

[Nobeigums 11](#_Toc105021027)

[4.1 Secinājumi 11](#_Toc105021028)

# Ievads

Darba tēma “Datu ieguve un apmaiņa izmantojot mikrokontrolierus” tika izvēlēta, lai uzlabotu savas prasmes zema līmeņa programmēšanā. Mūsdienās programmētāji arvien vairāk izvēlas pilnveidot savas zināšanas WEB vai aplikāciju programmēšanas, bet arvien retāk zema līmeņa programmēšanā – BIOS programmēšanā, mikrokontrolieru programmēšanā un cita veida embedded programmēšanā.

Darba laikā tika izmantota C++ programmēšanas valoda un netika izmantotas gatavas bibliotēkas, lai izprastu kā ir strādāt bez abstrakcijas.

**Darba mērķis:** Izveidot *no nulles mikrokontroliera programmatūru, kas var ievākt datus par laika apstākļiem un tos nosūtīt uz citu mikrokontrolieri.*

***Darba uzdevumi:***

1. Izveidot bibliotēkas nRF24L01 bezvadu modulim, DHT11 temperatūras/mitruma sensoram un gaismas sensoram.
2. Izveidot pārbaudi vai saņemtie dati ir neskarti.

## 1. PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA

## Ievads

### Dokumentācijas mērķis

Programmatūras prasību specifikācijas nolūks ir aprakstīt prasības uzdevumu plānošanas informācijas sistēmā.

Dokuments paredzēts izstrādātājiem un programmatūras lietotājiem, lai varētu izprast programmatūras funkcijas, darbību un specifiku. Dokumentācija palīdzēs tās lietotājiem integrēt programmatūru veiksmīgi savā projektā.

### Produkta sfēra

Programmas produkta nosaukums ir “MCU Weather” – C++ pirmkods, ko var izmantot savos projektos, lai ievāktu datus par laikapstākļiem un nosūtīt tos izmantojot bezvadu savienojumu.

Programmu var pielietot dažādos veidos, dažas aplikācijas ir termometrs, apkures sistēmas un gaisa mitrinātājs.

### Dokumentācijas pārskats

Fygyu

## 1.2 Vispārīgs apraksts

### 1.2.1 Produkta perspektīva

Izstrādātā programma (pirmkods) ir pieejams vietnē GitHub.

Šo pirmkodu var izmantot jebkurš, piemēram, hobijisti, lai integrētu, pielāgotu un izmantotu savām vajadzībām. Apskatot šo kodu arī rodas priekštats par to kā programmēt zemā līmenī.

### 1.2.2 Produkta funkcijas

Izstrādātai programmatūrai ir vairākas funkcijas. Pirmā pamata funkcija ir iespēja vienkāršoti strādāt ar nRF24L01 bezvadu moduli – to konfigurēt savām vajadzībām, sūtīt un saņemt datus. Otrā funkcija ir iespēja veikt temperatūras un gaisa mitruma mērījumus ar DHT11 sensoru. Trešā funkcija ir veikt apgaismojuma mērījumus ar foto-pretestību. Ceturtā funkcija ir nosūtīt datus uz citu mikrokontrolieri kurš tos apstrādā un nosūta datoram/serverim.

### 1.2.3 Programmatūras gaita

Mikro kontrolierim uzsākot darbu tiks inicializēti visi nepieciešamie moduļi – nRF24L01, DHT-11, Gaismas sensors. Tālāk tiks periodiski ievākti dati – gaismas līmenis, gaisa mitrums, temperatūra un tiks nosūtīti centrālajam kontrolierim. Centrālais kontrolieris savukārt būs pievienots datoram un nosūtīs visus saņemtos datus tam. Pēc tam dators saņemtos datus ievietos datubāzē.

### 1.2.4 Mērķauditorija

### 1.2.5 Pienākumu sadale

## 1.3 Funkcionālās prasības

* Ievākt sekojošos datus par apkārtni: temperatūru, gaisa mitrumu, gaismas spilgtumu.
* Periodiski nosūtīt ievāktos datus izmantojot bezvadu komunikāciju uz centrālo mikro kontrolieri.
* Centrālajam mikro kontrolierim nosūtīt iegūtos datus uz datoru.

**1.2.4 Pienākumu sadale**

Aleks Markuss Balodis:

* Mikrokontrolieru (*Embedded*) programmēšana *C/C++*
* Datu ievākšana un nosūtīšana datoram
* Dokumentācijas sagatavošana

# **IZSTRĀDEI IZMANTOTĀS TEHNOLOĢIJAS UN PROGRAMMATŪRA**

## 2.1 Programmatūras izstrādes vide

Lai izstrādātu programmatūru *ATmega328P* mikro kontrolierim esmu izvēlējies izmantot *Visual Studio Code (VSC)* ar paplašinājumiem *PlatformIO* un *C / C++ Syntax*. *VSC* izvēlējos, jo šis *IDE* pataisa kodu pārskatāmu un tam arī ir *intellisense*, kas atvieglo koda rakstīšanu. Paplašinājums *PlatformIO* pielāgo *VSC* darbam ar mikro kontrolieriem ļaujot augšupielādēt kodu uz tiem un izmantot dažas pamata bibliotēkas, kas atvieglo darbu.

Par cik rakstot kodu uz mikro kontrolieriem tiek izmantota *C / C++* valoda paplašinājums C / *C++ Syntax* ļauj *VSC* saprast *C / C++* sintaksi un izcelt to.

*Visual Studio Code (VSC)* ir *Microsoft* veidota teksta rediģēšanas programma, kurai ir iespējams uzstādīt dažādus paplašinājumus, kas nodrošina lietotājam nepieciešamo funkcionalitāti. VSC pamatā nodrošina šīs un citas funkcijas:

* integrētu komandas rindu;
* sintakses pārbaudītāju;
* *intelliSense*, kas ļauj automātiski pabeigt rakstīto, dod informāciju par funkciju, tās parametriem un tipu un daudz ko citu.

## 2.2 Visual Studio Code izmantotie paplašinājumi

### PlatformIO

### C / C++ Syntax

# **PRODUKTA KOMPONENŠU APRAKSTS**

## Mikrokontrolieri

### Mikrokontroliera koda funkcijas

void nRF24\_RegisterWrite(uint8\_t reg, uint8\_t regbit, uint8\_t val);

Funkcija ļauj viegli ierakstīt nRF24L01 reģistrā vērtību. Šai funkcijai nepieciešami 3 parametri – reģistrs, konkrētais reģistra bits un vērtība ko vēlies ierakstīt šajā reģistra bitā.

void nRF24\_Init();

Inicializācijas funkcija, kas sagatavo nRF24L01 bezvadu moduli darbam. Neprasa parametrus

void nRF24\_WriteTXData(uint8\_t\* pTXdata, uint8\_t asize);

Ieraksta nosūtāmos datus nRF24L01 nosūtīšanas reģistrā. Pieprasa 2 parametrus – rādītāju uz datu masīvu un masīva izmēru.

void nRF24\_ENTXMode();

Funkcija iestata nRF24L01 sūtīšanas režīmā. Nepieprasa parametrus.

void nRF24\_ENRXMode();

Funkcija iestata nRF24L01 saņemšanas režīmā. Nepieprasa parametrus.

uint8\_t \* nRF24\_ReadRXBuffer();

Funkcija nolasa datus no ienākušo datu reģistra un atgriež rādītāju uz saņemto datu masīvu. Parametrus nepieprasa.

# **Nobeigums**

## 4.1 Secinājumi