OULUN YLIOPISTO

TIETO- JA SÄHKÖTEKNIIKAN TIEDEKUNTA

ELEKTRONIIKAN JA TIETOLIIKENNETEKNIIKAN TUTKINTO-OHJELMA

KANDIDAATINTYÖ

PC-KOTELON INSTRUMENTOINTI JA AUTOMATISOINTI

Tekijä Lassi Myöhänen

Ohjaaja Antti Mäntyniemi

Toukokuu 2023

Myöhänen L. (2023) PC-kotelon instrumentointi ja automatisointi. Oulun yliopisto, tieto- ja sähkötekniikan tiedekunta, elektroniikan ja tietoliikennetekniikan tutkinto-ohjelma. Kandidaatintyö, 25 s.

TIIVISTELMÄ

Tämän työn tarkoituksena oli kehittää PC-kotelon avausmekanismi ja ledien ohjaus Blynk-sovelluksella. Työssä keskitytään ratkaisemaan kuinka umpinaisen PC-kotelon avaaminen ulkoa käsin onnistuisi. Avausmekanismissa käytetään tunnistautumiseen RFID-lukijaa ja askelmoottoria oven mekaaniseen avaamiseen. Koko järjestelmää ohjataan ESP32-mikrokontrollerilla, jonka sisäänrakennettu WLAN-moduulin avulla toteutetaan myös ledien ohjaus Blynk-sovellusta käyttäen. Lopullisessa PC-kotelossa järjestelmän toimintaa ei tässä työssä demonstroida, vaan kotelon ominaisuuksia simuloidaan.

Avainsanat: RFID, ESP32, LED, askelmoottori, WLAN, IoT, Blynk.

Myöhänen L. (2023) Instrumentation and automation of a PC-case. University of Oulu, Degree Programme in Electronics and Communications Engineering. Bachelor's Thesis, 25 p.

ABSTRACT

In this thesis a proof of concept is created for a PC enclosure. Designed system includes electronics for opening the enclosure and LED's are controlled via Blynk app. Main challenge in the design is that the enclosure is fully sealed when closed. RFID and a stepper motor are used for the opening and closing sequence. The system uses an ESP32 micro controller to control the needed functions. The ESP32 micro controller has a built-in WiFi which allows the LED's to be controlled using the Blynk app. Functionality of the system is simulated since the actual enclosure does not exist yet.

Key words: RFID, ESP32, LED, stepper motor, WiFi, IoT, Blynk.

SISÄLLYSLUETTELO

ALKULAUSE

Sain idean täysin suljetusta kotelosta yli kymmenen vuotta sitten. Olen sitä pyöritellyt päässäni aika ajoin, mutta varsinainen toteutus alkaa tämän kandidaatintyön myötä. Idea täysin aukottomasta PC-kotelosta kuulostaa hullulta, ja juuri sen takia haluaisinkin sen toteuttaa. Fyysistä koteloa ei ole vielä olemassa, vaan se on vielä suunnitteluasteella. Lopullisen kotelon ollessa valmis, voin käyttää tässä työssä tekemiäni ohjausjärjestelmän osia. Ohjausjärjestelmä ei todennäköisesti tule pysymään täsmälleen samanlaisena, koska kotelo itsessään tulee määrittämään erilaisia kriteerejä ohjaukselle. Esimerkiksi askelmoottori voi olla liian tehoton, minkä myötä tarvitaan tehokkaampi moottori. Moottorin tehon kasvaessa virrankulutus nousee ja järjestelmää tulee muuttaa. Ajatuksena työssä on tehdä eräänlainen prototyyppi ja todentaa, että halutut toiminnallisuudet voidaan toteuttaa ylipäätään. Ohjausjärjestelmän toimintaperiaate ja perusominaisuudet pysynevät kuitenkin samoina lopulliseen toteutukseen asti ja työn aikana hankittu tieto ja taito vievät lopullista projektia eteenpäin.

Kiitokset työn ohjaajalle, Antti Mäntyniemelle.

Oulussa 13.5.2023

Lassi Myöhänen

LYHENTEIDEN JA MERKKIEN SELITYKSET

RFID	Radio Frequency Identification, radiotaajuinen etätunnistus
Hz	Hertsi
ESP32	Espressif Systemsin valmistama mikrokontrolleri
LED	Light-Emitting Diode, ledi
PWM	Pulse-Width Modulation, pulssinleveysmodulaatio
IoT	Internet of Things, asioiden internet
WLAN	langaton lähiverkko
GPIO	General-Purpose Input/Output, yleiskäyttöinen pinni

1 JOHDANTO

Työn tavoitteena oli toteuttaa todistus konseptista. Konseptina oli tietokonekotelon automatisointi ja instrumentointi käyttäen mikrokontrolleria ja oheislaitteita. Työtä ohjasi kriteeri siitä, että kotelon tulee olla täysin johdoton suljettuna. Tämä kriteeri pakotti järjestelmän olemaan langaton, huolimatta käytetystä teknologiasta.

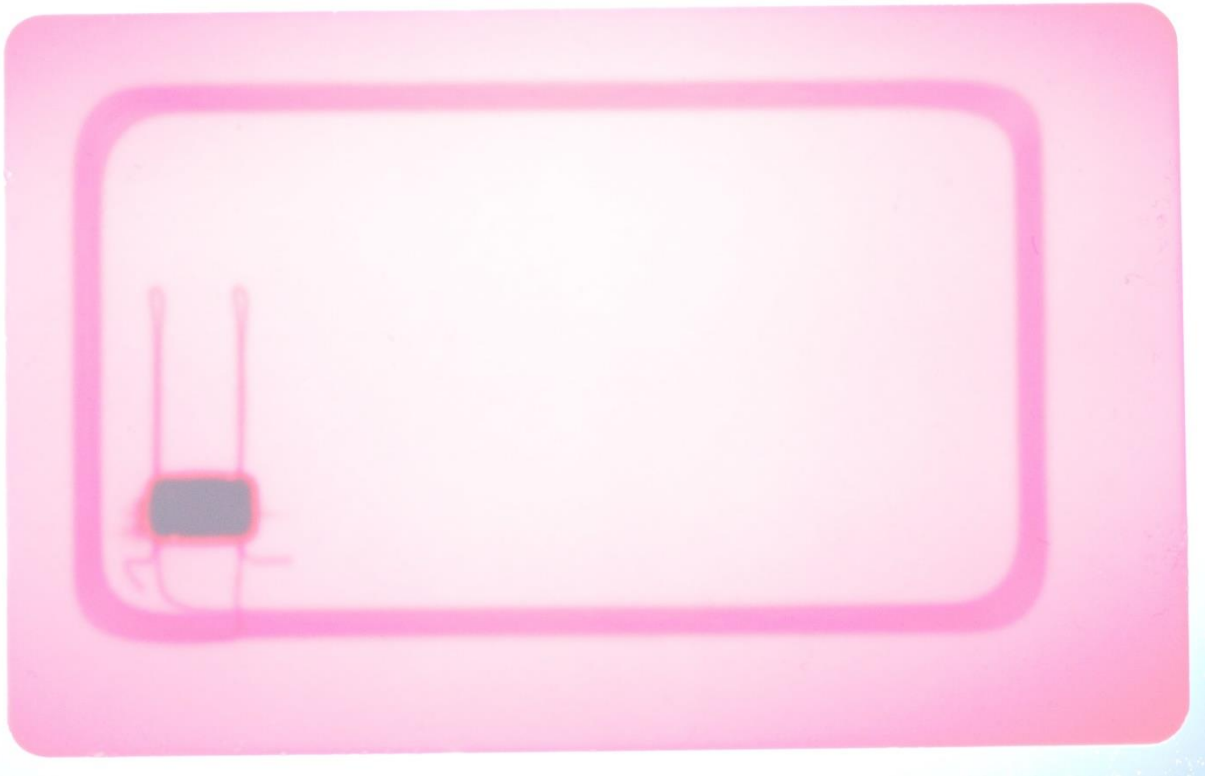
Työssä kotelon avaaminen ja sulkeminen liipaistiin käyttäen RFID-teknologiaa. LED-nauhan ohjaus toteutettiin myös langattomasti. Mikrokontrolleri oli yhteydessä internetiin WLAN:in välityksellä. Kotelon avauksen mekaanista osuutta simuloitiin askelmoottoria käyttäen. Työssä käytetty elektroniikka oli ESP32-mikrokontrolleri, WS2812B-piirillä toteutettu LED-nauha, MFRC522-piirillä toteutettu RFID-lukija sekä 28BYJ-48 askelmoottori. Mikrokontrollerin ohjelmointi tehtiin käyttäen Arduino IDE ohjelmointiympäristöä. Mobiililaitteen ja järjestelmän välinen yhteys toteutettiin käyttäen Blynk alustaa.

2 TEORIA

Teoriaosiossa käsitellään ohjausjärjestelmän eri osakokonaisuuksia ja niiden toiminallisuutta.

2.1 RFID

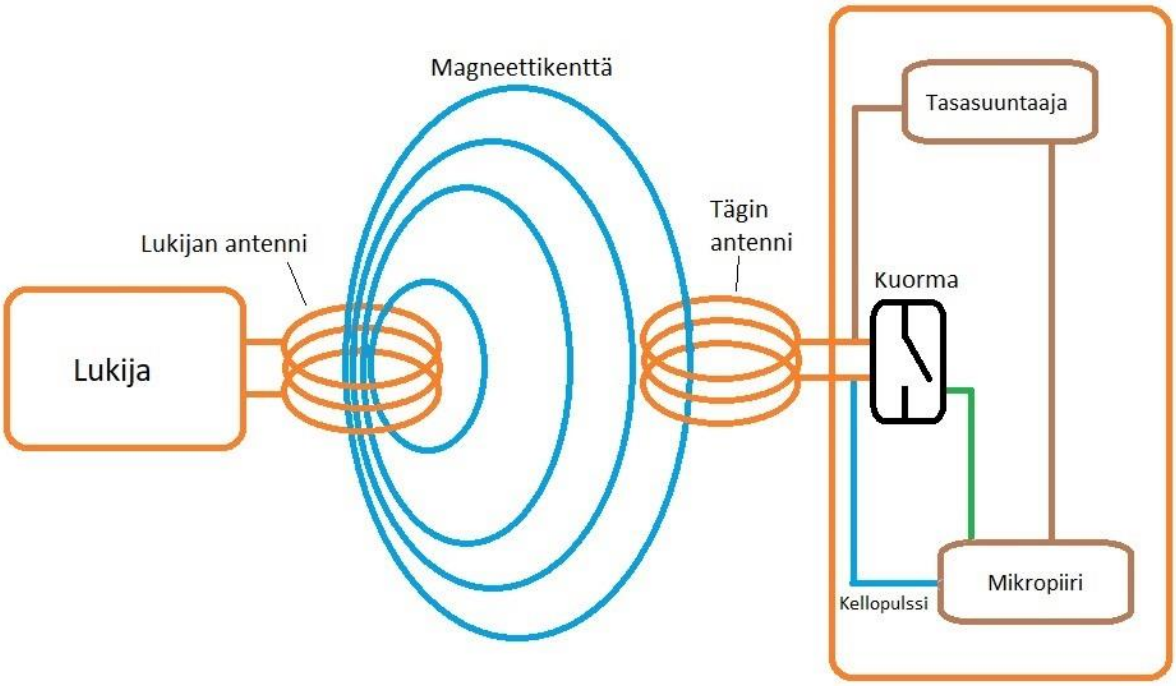
Radio Frequency Identification (RFID) on langaton tunnistautumistapa. Systeemi koostuu kahdesta osasta: lukijasta ja transponderista (tägi). Tägi voi olla esimerkiksi pieni litteä kiekko tai pankkikorttia muistuttava muovinen kortti. Tägissä on mikropiiri ja käämitty antenni. Kuvassa 1 on kortti, joka on valaistu kameran irtosalamalla vastapuolelta kameraa. Kuvassa näkyvä tumma osa on mikropiiri, joka ohjaa tägin toimintaa. Ympärillä kulkeva kehä on käämitty antenni.



Kuva 1. Läpivalaistu tägi.

Tiedonsiirto lukijan ja tägin välillä tapahtuu radioaalloilla. Tiedonsiirtotaajuudet voidaan jaotella kolmeen osaan: LF (Low Frequcency) 30-300 kHz, HF (High Frequency) 3-30 MHz ja UHF (Ultra-High Frequency) 300 MHz-3GHz. Tässä työssä käytettävä lähetin ja tägi toimivat 13.56 MHz taajuudella [1]. Työssä käytettävä tägi on virtalähteetön ja siten täysin passiivinen ollessaan lukijan kantaman ulkopuolella.

LF ja HF taajuuksilla operoivat passiiviset RFID-järjestelmät toimivat induktiivisen kytkeytymisen avulla.[2] Kuvassa 2 on havainnekuva lukijan ja tägin induktiivisesta kytkeytymisestä. Tägi saa lukijan magneettikentästä itselleen käyttövirran, kellopulssin ja tavan lähettää tietoa takaisin lukijalle.



Kuva 2. Induktiivinen kytkeytyminen.

Lukija lähettää sinimuotoista kantoaaltoa. Kantoaallon muuttuva magneettikenttä indusoi tägin kelaan jännitteen. Indusoitunut vaihtojännite tasasuunnataan tägin käyttöjännitteeksi. Tägin mikropiiri ohjaa antennin päiden välillä olevaa transistoria muistissa olevan datan perusteella. Transistorin tilanmuutokset moduloivat alikantoaaltoa, mikä näkyy lukijan päässä jännitehäviönä. Jännitehäviöstä lukija sekä tunnistaa tägin läsnäolon että demoduloi vastaanotetun datan. Lukijalta tägille kulkeva bittijono on Miller-koodattua ja amplitudinsiirtoavainnus moduloitua. Tägiltä lukijalle kulkeva bittijono on Manchester-koodattua kuormamodulaatiota.

2.2 LED-nauha

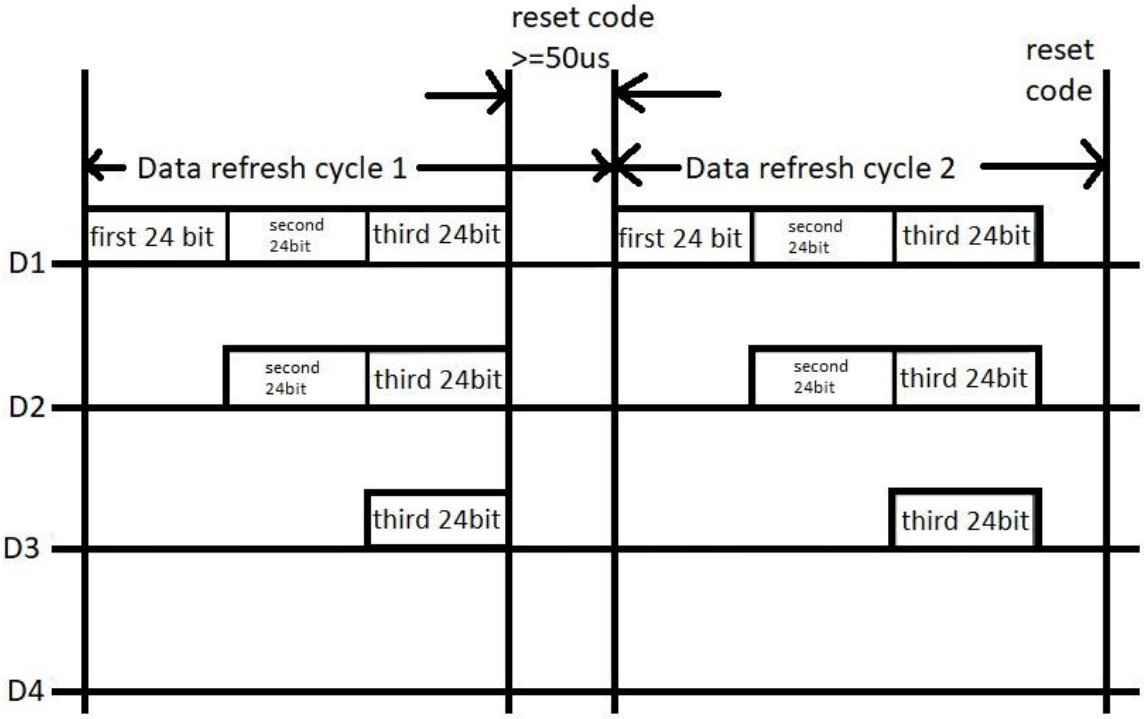
Työssä käytetään WS2812B-piirillä toteutettua RGB (Red, Green, Blue) LED-nauhaa. WS2812B on mikropiiri, johon on integroitu ohjaus ja LED samaan 5050-koon koteloon. LED-nauhassa piirit ovat kytkettyinä peräkkäin ja ohjaussignaali etenee piiriltä piirille. Ohjainpiirin avulla LED-nauhassa voidaan ohjata yksittäisiä ledejä, eikä jokaiselle ledille tarvita erillistä johdinta ohjaamista varten. Piiriin on integroitu ledin lisäksi kytkentäsuojaus, signaalin uudelleenmuotoilu, data kiikku, reset ja oskillaattori. LED-nauhan ohjaaminen tapahtuu syöttämällä ohjausdata DIN-pinniin (Data In). Ohjausdata on NRZ-koodattua ja yhden ledin ohjausdata on 24 bittiä.[3] 24-bittinen ohjausdata on kuvattu kuvassa 3. Ohjausdata sisältää kahdeksan bittiä jokaista väriä kohti. On hyvä huomioida, että värit ovat eri järjestyksessä totutusta RGB:stä.

G7	G6	G5	G4	G3	G2	G1	G0	R7	R6	R5	R4	R3	R2	R1	R0	B7	B6	B5	B4	B3	B2	B1	B0

Kuva 3. Ohjausdatassa värien järjestys on vihreä, punainen ja sininen.

Jokaiselle värille on käytössä kahdeksan bittiä, joilla määritellään kyseessä olevan värin kirkkaus. Yksittäisen värin kirkkautta voi siis säätää välillä 0-255. Kuvassa 4 on kuvattu, kuinka

data siirtyy nauhassa. Kuvassa D1 on nauhan ensimmäinen ledi, D2 on toinen ledi ja niin edelleen.

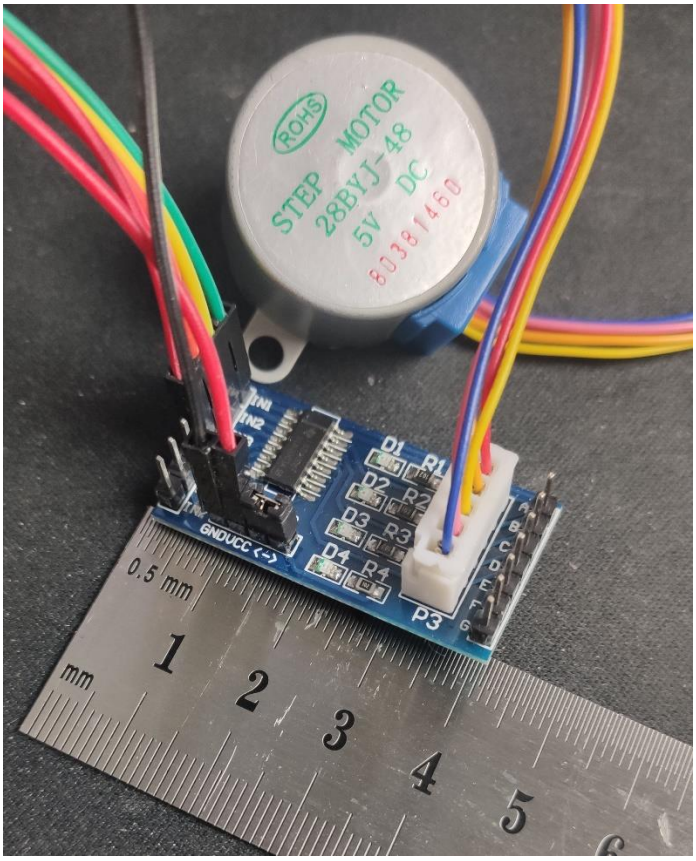


Kuva 4. Havainnekuva tiedonsiirrosta LED-nauhassa.

Mikrokontrollerilta lähetetään ohjausdata ensimmäisen ledin DIN-pinniin. Ensimmäinen ledi siirtää ensimmäiset 24 bittiä datakiikkuunsa ja siirtää kaikki muut bitit korjausvahvistimen kautta DO-pinniin (Data Out). Ensimmäisen ledin DO-pinni on kytketty toisen ledin DIN-pinniin. Ohjausdata kulkee tällä tavoin ledien läpi nauhan loppuun saakka. Ohjatessa LED-nauhaa, tulee ledien määrä määrittää, jotta mikrokontrolleri osaa lähettää oikean määrän 24:n bitin tietueita LED-nauhalle. Signaalin korjaavan vahvistimen ansiosta ohjausdata ei vaimene ja siten se ei rajoita ledien määrää nauhassa.

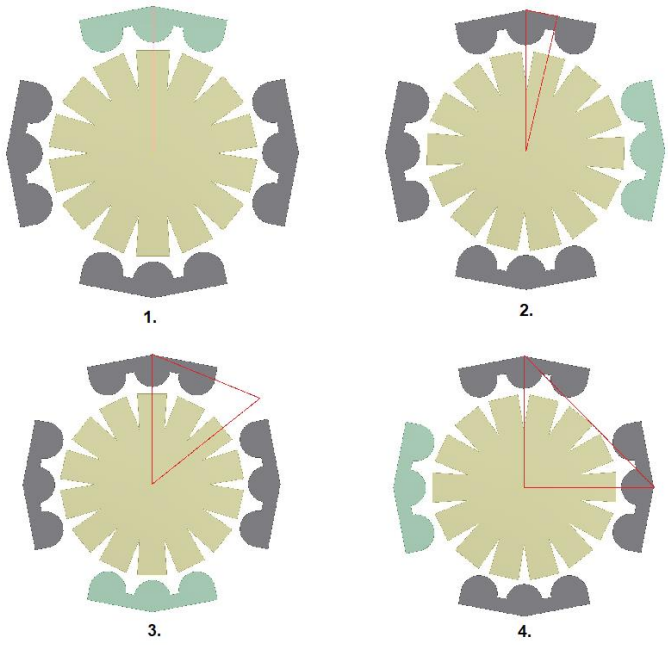
2.3 Askelmoottori ja virtavahvistin

Työssä käytettävä askelmoottori on tyypiltään 28BYJ-48. Moottori on pieni ja sen käyttöjännite on viisi volttia tasajännitettä. Askelmoottori on yksinapainen ja nelivaiheinen. Kuvassa 5 on työssä käytetty moottori ja virtavahvistin.



Kuva 5. 28BYJ-48 sekä virtavahvistin.

Yksinapaisissa askelmoottoreissa virta kulkee kelassa aina samaan suuntaan, minkä takia moottorinohjaus on yksinkertaisempaa toteuttaa [4]. Askelmoottorin pyörivä liike saadaan, kun virtaa johdetaan keloihin vuorotellen. Kelat toimivat sähkömagneetteina ja keskellä sijaitseva roottori liikahtaa hampaiden määrittelemän askeleen verran, kun kelaan johdetaan virtaa. Kuvassa 6 on yksinkertaistettu malli roottorin liikkeestä.



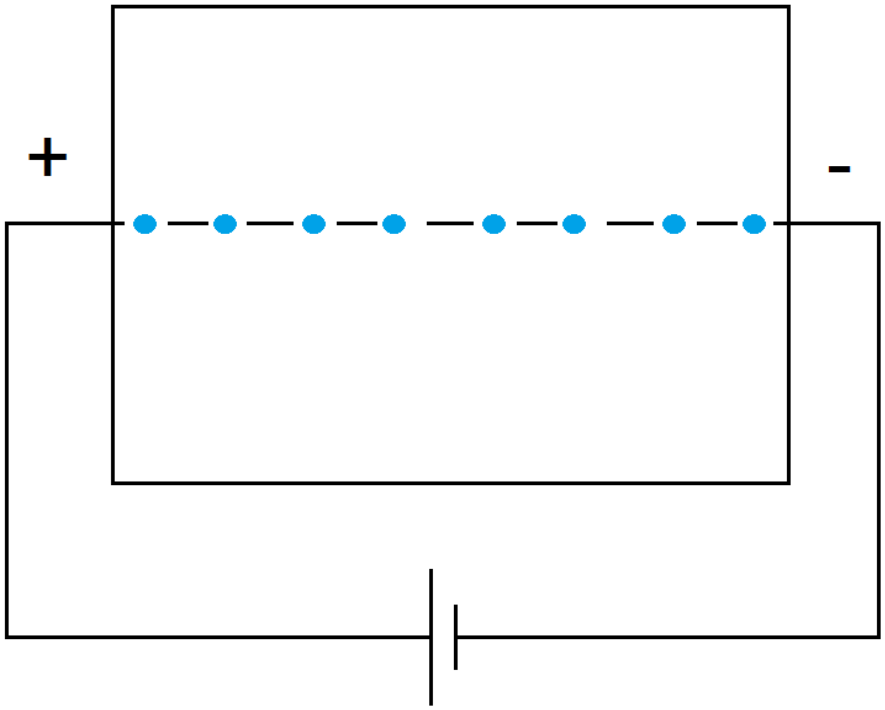
Kuva 6. Moottorin neljä askelta. Vihreällä kuvattuina kelat, jossa kulkee virta.

Mikrokontrollerin virranantokyky on pienempi kuin askelmoottorin toiminta vaatii. Työssä käytetään virtavahvistinpiiriä, joka on toteutettu ULN2003A mikropiirillä. Mikropiirin sisällä on Darlington-kytkentä, joka koostuu transistoripareista.

Askelmoottorin valinta ei perustunut lopullisen kotelon mekaanisiin vaatimuksiin. Tarkoituksena on sisällyttää työssä jonkinlainen askelmoottori ja todentaa konseptin toimivuus ohjelmallisesti ja elektronisesti.

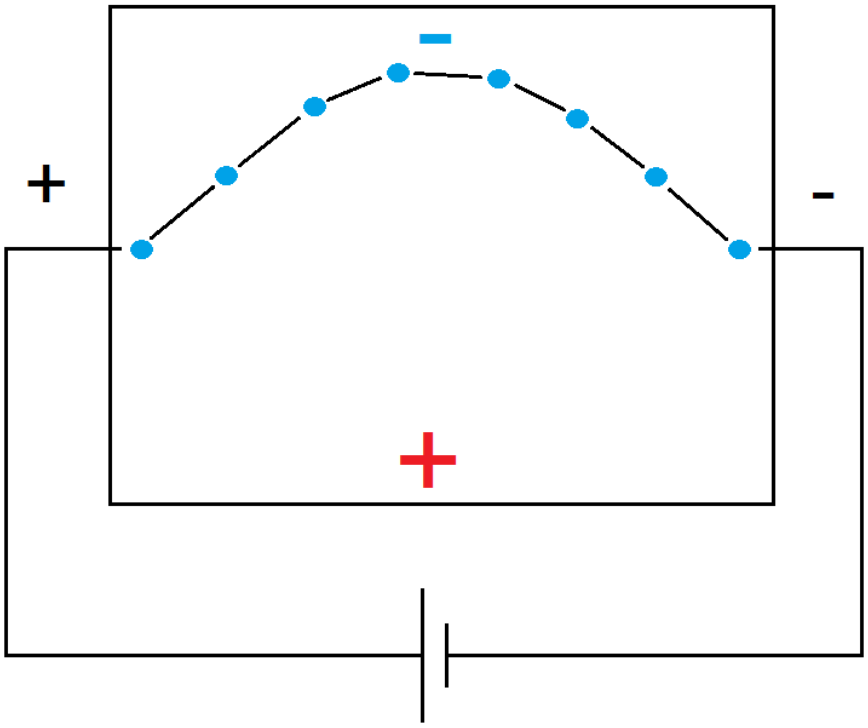
2.4 Hall-anturi

Hall-anturin toiminta perustuu hall-ilmiöön. Hall-ilmiö ilmenee, kun virtajohdin on kohtisuorassa magneettikenttään nähden. Ilman magneettikenttää johtimen varauksenkuljettajat kulkevat johtimen sisällä suorassa linjassa, kuten kuvassa 7 on esitetty.



Kuva 7. Sinisellä merkityt varauksenkuljettajat kulkevat suorassa linjassa levymäisen johtimen läpi.

Kun johdin tuodaan magneettikenttään kohtisuorassa, varauksenkuljettajien kulkusuunta muuttuu Lorenzin voiman vuoksi. Koska varauksenkuljettajat eivät enää kulje johtimen keskellä, vaan kasaantuvat johtimen reunaan, johtimen reunojen välille syntyy potentiaaliero. Kuvassa 8 on kuvattuna varauksenkuljettajien muuttunut kulkusuunta ja sen aiheuttama potentiaaliero johtimen ylä- ja alareunan välillä. Kuvassa 8 magneettikenttä on katsojasta poispäin.



Kuva 8. Varauksenkuljettajien muuttunut kulkureitti ja syntynyt potentiaaliero.

Hall-anturissa johtimen ylä- ja alareunan välille muodostuvaa jännitettä kutsutaan hall-jännitteeksi. Hall-jännitteen avulla voidaan havaita magneettikentän läsnäolo ja sen voimakkuus.[6] Hall-jännite on suoraan verrannollinen magneettikentän voimakkuuteen. Koska hall-jännite on usein pieni, täytyy jännitettä vahvistaa anturin sisäisellä vahvistimella.

2.5 Blynk

Blynk on alusta, jolla voidaan luoda IoT-laitteita hyödyntäviä ympäristöjä. Blynk koostuu neljästä pääosasta. Blynk.Edgent on kirjasto, joka sisällytetään mikrokontrollerin koodiin. Blynk.Edgentillä hallitaan tiedonsiirtoyhteyksiä, kuten WLAN:ia, Ethernet:iä ja matkapuhelinyhteyksiä. Datan siirto mikrokontrollerin ja Blynk.Cloudin välillä tapahtuu myös kyseisen kirjaston avulla. Blynk.Console on verkkosovellus, jossa voidaan hallita liitettyjä laitteita. Hallintaominaisuuksiin kuuluu muun muassa konfigurointi, yhdistäminen, sensorin lähettämän datan analysointi ja laiteohjelmiston päivitys ilmarajapinnan yli. Kolmas osa Blynkiä on Blynk.Apps. Blynk.Apps on mobiilisovellus, jolal voidaan luoda käyttöliittymä Blynk-projektille. Käyttöliittymä rakennetaan raahaamalla valmiita elementtejä ruudulle. Elementit konfiguroidaan vastaamaan mikrokontrollerille ohjelmoituja toimintoja. Projektikokonaisuus nitoutuu yhteen Blynk.Cloudin avulla. Blynk.Cloudin kautta dataa saadaan siirrettyä mobiilisovelluksesta mikrokontrollerille ja sille ohjelmoiduille oheislaitteille.

Tässä työssä ledien ohjaus toteutettiin Blynkin avulla sen muokattavuuden ja helppokäyttöisyyden vuoksi. Blynk mahdollistaa ledien kirkkauden säätämisen ja värien vaihtamisen WLAN-yhteyden kautta. Ledien värejä ja kirkkautta voidaan säätää joko

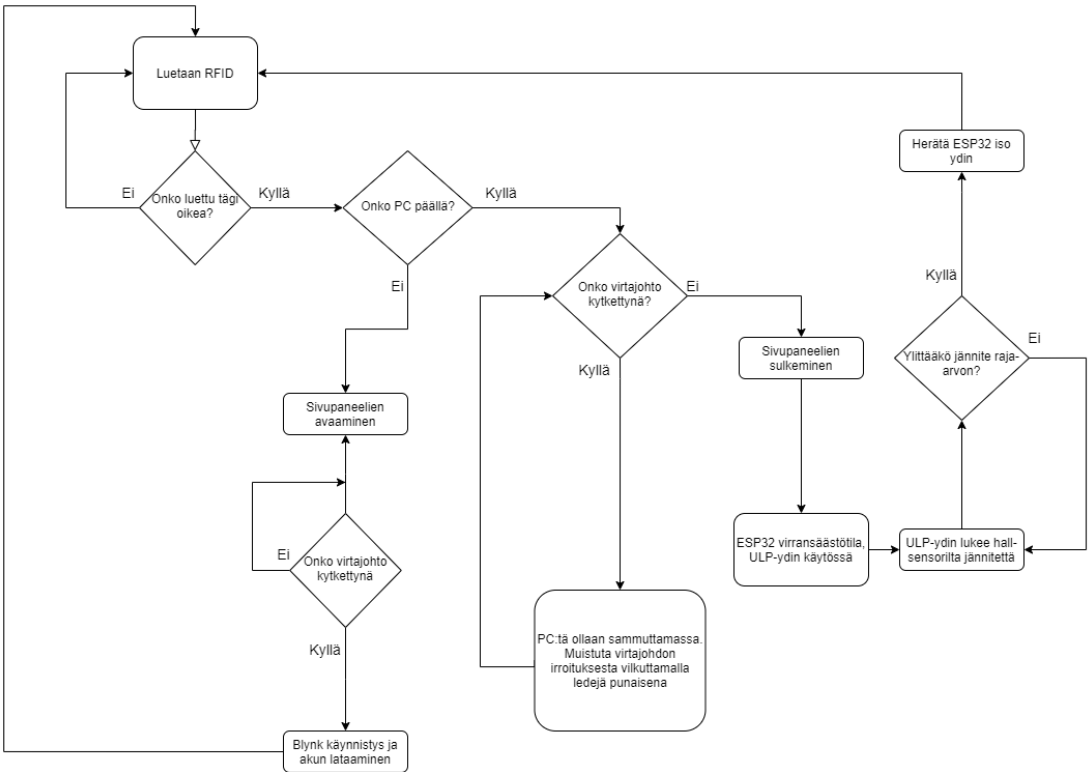
liukusäätimillä tai valitsemalla mikrokontrollerille ohjelmoituja toimintatiloja. Jotta mikrokontrolleriin kytkettyjä ledejä voidaan ohjata, täytyy mikrokontrollerilla saada yhteys Blynkin servereihin internetin yli. Blynk käyttää autentikointiavainta, joka saadaan Blynkin web-sivulta. Autentikointiavain määritetään mikrokontrollerin koodissa muuttujalla ”auth”. Jotta mikrokontrollerilla on yhteys internetiin, täytyy sen yhteyden asetukset määrittää koodissa. ESP32 käyttäessä voidaan käyttää WLAN:ia. Koodissa langattoman lähiverkon verkkotunnus määritetään muuttujalla ”ssid”. Mahdollinen verkon salasana määritetään muuttujalla ”pass”.

Blynk alustetaan void setup:in sisällä komennolla: ”Blynk.begin(auth, ssid, pass);”. Mikrokontrollerin koodissa Blynkia suoritetaan void loopin sisällä käyttäen funktiota Blynk.run(). Blynk.Apps käyttää virtuaalisia yleiskäyttöisiä pinnejä, eli General-Purpose Input/Output (GPIO). Virtuaalipinneihin voidaan kirjoittaa dataa käyttämällä BLYNK_WRITE() funktiota.

3 TYÖN TOTEUTUS

Tarkoituksena on, että PC-kotelo voisi olla pari viikkoa suljettuna ilman ulkoista virtalähdettä ja kotelon voisi edelleen avata ohjausjärjestelmää käyttäen. Ohjausjärjestelmä toimii akun varassa PC-virtalähteen ollessa virrattomana. Virrankulutuksen minimoimiseksi mikrokontrolleri toimii lepotilassa akun varassa ollessaan. Mikrokontrollerin lisäksi ainoa virtaa kuluttava komponentti on hall-anturi.

Kuvassa 10 on ohjausjärjestelmän toiminta kuvattuna vuokaaviona.



Kuva 10. Ohjausjärjestelmän toiminta vuokaaviona.

Ohjausjärjestelmä voidaan ajatella tilakoneena, jossa on neljä tilaa: lepotila, kotelon avaaminen, aktiivitila ja kotelon sulkeminen.

Mikrokontrolleria valitessa tärkeitä kriteerejä olivat sen ominaisuuksien monipuolisuus ja virrankulutuksen minimointi. ESP32 oli tähän hyvä valinta, koska siinä on kattavat ominaisuudet, kuten sisäänrakennettu WLAN-piiri ja kaksi erillistä prosessoriydintä. Toinen prosessoriytimistä on erittäin vähän virtaa kuluttava, mikä mahdollistaa laitteen pitkäaikaisen käytön ilman verkkovirtaa.

3.1 Lepotila

Järjestelmän ollessa lepotilassa kuvitteellinen kotelo on suljettuna. Askelmoottori kuluttaa paljon virtaa, myös kun se ei pyöri, joten askelmoottori on tehty virrattomaksi lepotilan ajaksi. Kotelon avautuminen askelmoottorin virrattomuuden takia voidaan estää käyttämällä kierreruuvia eli matoruuvia. Mikrokontrolleri käyttää lepotilassa pääprosessorin sijaan ULP-ydintä (Ultra Low Power, erittäin vähävirtainen) virrankulutuksen minimoimiseksi. Lepotilassa ULP-ydin suorittaa Assembly-koodia tavallisen C-koodin sijaan. ULP-ytimen lisäksi ainoa virtaa kuluttava komponentti on Hall-anturi. Hall-anturi on kytketty mikrokontrollerin

analogiseen sisääntuloon. Assembly-koodi seuraa Hall-anturin yli olevaa jännitettä. Jos jännite ylittää ohjelmassa määritetyn raja-arvon, herättää ULP-ydin mikrokontrollerin pääprosessorin. Pääprosessin käynnistyttyä järjestelmä siirtyy suorittamaan C-koodia ja järjestelmä siirtyy kotelon avaustilaan.

3.2 Kotelon avaus

Ohjelman suorituksen siirryttyä pääprosessorille, alkaa RFID-lukija lähettämään. Kotelon avaus aloitetaan tuomalla tägi lukijan kantaman sisälle. Lukijan tunnistettua oikea tägi, askelmoottori avaa kotelon. Avatessa koteloa askelmoottori saa virtansa kotelossa olevasta akusta. Ajallisesti avaaminen ei kestä kovinkaan kauaa, eikä avauksia tehdä kuin yksi, mutta askelmoottorin suuri virrankulutus on hyvä ottaa huomioon akkua valitessa. Kotelon auettua ohjelma jää odottamaan PC:n virtalähteen kytkentää verkkovirtaan. Mikrokontrollerin analoginen sisääntulo on kytketty PC:n virtalähteeseen, joten mikrokontrolleri tietää, milloin virtalähde on kytketty verkkovirtaan. Kun virtalähde on kytketty verkkovirtaan, ohjelma siirtyy ledien ohjaustilaan ja kotelon sisäinen akku latautuu latauspiirin kautta.

3.3 Aktiivitila

Mikrokontrollerissa on sisäänrakennettu WLAN-ominaisuus. Ohjelmassa määritetään mihin WLAN-verkkoon halutaan kytkeytyä ja mahdollinen salasana verkkoon. On hyvä huomioida, että sekä verkon nimi että salasana voivat olla selkokielisenä ohjelmassa. Jakaessa koodia tulee varmistua siitä, että arkaluontoiset tiedot poistetaan. Vaihtoehtoisesti koodin voi toteuttaa niin, että arkaluontoiset tiedot luetaan tiedostosta, jota ei jaeta.

Mikrokontrollerin yhdistettyä WLAN-verkkoon ohjelma siirtyy suorittamaan Blynk:ltä tulevia ledien ohjauskäskyjä. Suoritus pysyy aktiivitilassa, kunnes tietokone sammutetaan.

3.4 Kotelon sulkeminen

Kun tietokone sammutetaan, voidaan kotelo sulkea käyttämällä tägiä. Kotelon sulkeminen aloitetaan asettamalla tägi kotelon ulkopinnalle, RFID-lukijan kantaman sisäpuolelle. Virtajohdon poistamisesta muistuttaa punaisena vilkkuvat ledit ja muistutus on päällä niin kauan kuin virtajohto on kytkettynä tietokoneeseen. Kun virtajohto irrotetaan tietokoneesta, alkaa askelmoottori pienen viiveen jälkeen sulkemaan koteloa. Kun kotelo on suljettu, siirtyy mikrokontrolleri lepotilaan. Lepotilassa ESP32 käyttää jälleen ULP-ydintä ja suorittaa sen Assembly-koodia.

4 YHTEENVETO

Työn lopputuloksena oli toimiva konsepti. Mikrokontrollerin valinta oli helppo. ESP32 oli tuttu entuudestaan ja ESP32:ssa on monipuoliset sisäänrakennetut ominaisuudet. Lisäominaisuuksien varalta ESP32 tarjoaa myös paljon lisää laajennusvaraa. Vähävirtainen ydin oli hyvä tapa hillitä virrankulutusta, mutta sen toteuttaminen vaati Assembly-koodin osaamista. Yksinkertaisemmin asian voisi toteuttaa käyttäen lukkiutuvaa magneettikytkintä. Tällöin kotelon ollessa suljettuna, ei järjestelmä kuluttaisi lainkaan virtaa.

Kotelon läpäiseväksi tekniikaksi RFID oli hyvä valinta. RFID on yksinkertainen ja robusti teknologia, mikä helpotti järjestelmän toteutusta. LED-nauha on helppo keino ohjata yksittäistenkin ledien väriä ja kirkkautta. Huonona puolena LED-nauhoissa on niiden korkea virrankulutus. Ledeihin sisäänrakennettu operaatiovahvistin lisää virrankulutusta ja nauhassa voi olla jopa 144 lediä metriä kohti. Tehonkulutus voi siis olla jopa 35 wattia, jolloin virtalähteestä pitäisi saada seitsemän ampeeria, kun käyttöjännite on viisi volttia [7].

Askelmoottorit ovat suhteellisen yksinkertaisia laitteita, mutta niiden ohjaaminen voi olla monimutkaista. Askelmoottorityyppejä on useita ja niitä ohjataan eri tavoilla. Lisäksi askeleita voi olla useanlaisia, esimerkiksi mikroaskellus (micro-stepping). Suunnittelussa kannattaa ottaa huomioon askelkulma, moottorin vääntö ja se, että moottori kuluttaa paljon virtaa myös paikallaan ollessaan.

Hall-anturin lisääminen projektiin helpotti suuresti virrankulutuksen hallintaa. Magneettikentän voimakkuutta mittaamalla pystyttiin sammuttamaan RFID-lukija sekä enemmän virtaa kuluttava prosessoriydin ESP32:ssa. Kotelon avaaminen muuttui kuitenkin samalla kompleksisemmaksi. Avaus ei enää toimi pelkällä RFID-tunnisteella, vaan aluksi tarvitaan magneetti, johon hall-anturi reagoi. Hall-anturin jännitettä seuraava assembly-koodi oli mukava lisähaaste projektiin.

Blynk tarjosi helpohkon tavan ohjata ledejä mikrokontrollerin kautta. Tätä työtä tehdessä Blynk muuttui paljon. Palvelut on selkeästi suunnattu yritysmaailmaan ja isompiin kokonaisuuksiin. Harrastelijan näkökulmasta Blynk ei välttämättä ole enää järkevin tapa toteuttaa IoT-projekteja. Tietoturvan näkökulmasta Blynk on ongelmallinen, kuten IoT-maailmassa usein onkin. Jotta laitteet ovat etäkäytettävissä, joutuu mikrokontrolleri olemaan internetiin yhteydessä. Kaikki data kulkee Blynkin kautta, jolloin ei voida olla varmoja mitä datalle tapahtuu. Samalla hyökkäyspinta-ala omiin järjestelmiin kasvaa. IoT-laitteita ja palveluita käyttöönottaessa tulisikin ottaa huomioon mahdolliset tietoturvariskit. Esimerkiksi viime vuonna paljastui isoja aukkoja älykotilaitteisiin keskittyvän yrityksen, Eufyn, tietoturvassa [8]. Yritys väitti, että asiakkaiden kameroiden kuva ei poistuisi laitteista. Kyseessä oli kuitenkin törkeä valhe. Kameroista data valui Eufyn servereille. Videostriimit olivat myös salaamattomia, joten kuka tahansa pystyi katsomaan niitä VLC-mediatoistimella. Mahdollisten tietoturvariskien lisäksi hinnoittelu on yksi Blynkin huonoista puolista. Nykyään ilmaisversioon saa vain kaksi laitetta. Jos laitteita halutaan kymmenen, on tämänhetkinen kuukausimaksu 6,99 dollaria.

Koodin tekeminen oli suhteellisen virtaviivaista. Vuokaavio helpotti huomattavasti kokonaisuuden hahmottamista ja tarvittavien silmukoiden toteuttamista. Jokainen komponentti saatettiin haluttuun toimintakuntoon ensin erikseen, mikä helpotti vianmääritystä huomattavasti. Valmiit osakokonaisuudet olivat helppo koota isommaksi kokonaisuudeksi.

Kokonaisuudessaan projekti onnistui kohtuullisesti. Lopputuloksena on toimiva kokonaisuus, joskin joitakin ominaisuuksia jäi toteutuksesta pois. Aikaansaadun järjestelmän tulisi toimia akun varassa kotelon ollessa suljettu. Akun lisäksi projektista jäi pois akun tarvitsema latauspiiri. Alkuperäinen suunnitelma oli, että akku ladattaisiin automaattisesti, kun

tietokone on kytkettynä sähköverkkoon. Paranneltua versiota tehdessä tulisikin ennen kaikkea ottaa huomioon avausmekanismin toteutus. RFID-tekniikka on toimiva, mutta ominaisuus voisi olla toteutettavissa yksinkertaisemmin jotain muuta tapaa käyttäen. Transistorikytkentä magneettikytkimellä varustettuna tai vähävirtainen Bluetooth voisivat toimia toteutuksessa paremmin.

5 LÄHDELUETTELO

[1] MFRC522 Standard performance MIFARE and NTAG frontend (2016) datalehtinen.

[2] Sorrells, P. (1998) Passive RFID Basics. Microchip Technology Inc.

[3] WS2812B Intelligent control LED integrated light source. Worldsemi. URL: http://www.world-semi.com

[4] Earl B. (2020) All About Stepper Motors. Adafruit Industries. URL: https://learn.adafruit.com/all-about-stepper-motors

[5] ULN2003A datasheet, Diodes Incorporated. URL: https://www.diodes.com/assets/Datasheets/ULN200xA.pdf

[6] Nave R. Hall Effect. URL: http://hyperphysics.phy-astr.gsu.edu/hbase/magnetic/Hall.html

[7] Adafruit NeoPixel. URL: https://www.adafruit.com/product/1506

[8] Verge (2020) URL: https://www.theverge.com/2022/11/30/23486753/anker-eufy-security-camera-cloud-private-encryption-authentication-storage

6 LIITTELUETTELO

Liite 1 Työn lähdekoodi.

```
#include "esp_sleep.h"
#include "driver/rtc_io.h"
#include "driver/adc.h"
#include "esp32/ulp.h"
#include "ulp_main.h"
#include "ulptool.h"

#include <SPI.h>
#include <MFRC522.h>
#include <stdio.h>
#include <FastLED.h>
#include <Stepper.h>
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

#define BLYNK_PRINT Serial

//Hall sensor data is being read from GPIO34

extern const uint8_t ulp_main_bin_start[] asm("_binary_ulp_main_bin_start");
extern const uint8_t ulp_main_bin_end[]   asm("_binary_ulp_main_bin_end");

//#define RST_PIN 9 // Configurable, see typical pin layout above
//#define SS_PIN 10 // Configurable, see typical pin layout above
const int RST_PIN = 22; // Reset pin
const int SS_PIN = 21; // Slave select pin

const int stepsPerRevolution = 32;  // change this to fit the number of steps per revolution
const int GearedStepsPerRev = 32*64;  // change this to fit the number of steps per revolution
// for your motor

// initialize the stepper library on pins 8 through 11:
Stepper myStepper(stepsPerRevolution, 26, 25, 33, 32);

char auth[] = "<INSERT API KEY>";
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "<WIFI SSID>";
char pass[] = "<WIFI PASSWORD>";

#define NUM_LEDS 60
#define DATA_PIN 27
CRGB leds[NUM_LEDS];
```

```
//variables
boolean correct_card = false;
int n=0;
int i=0;
int k=0;

int R = 0;
int G = 0;
int B = 0;

void setup() {
 init_ulp_program();
 Serial.begin(115200); // Initialize serial communications with the PC
 while (!Serial); // Do nothing if no serial port is opened (added for Arduinos based on
ATMEGA32U4)
 SPI.begin(); // Init SPI bus
 pinMode(12, INPUT_PULLDOWN);
 FastLED.addLeds<NEOPIXEL, DATA_PIN>(leds, NUM_LEDS);
 Blynk.begin(auth, ssid, pass);
}

static void start_ulp_program()
{
 /* Start the program */
 esp_err_t err = ulp_run((&ulp_entry - RTC_SLOW_MEM) / sizeof(uint32_t));
 ESP_ERROR_CHECK(err);
}

static void init_ulp_program()
{
 esp_err_t err = ulp_load_binary(0, ulp_main_bin_start,
                    (ulp_main_bin_end - ulp_main_bin_start) / sizeof(uint32_t));
 ESP_ERROR_CHECK(err);

 /* Configure ADC channel */
 /* Note: when changing channel here, also change 'adc_channel' constant
    in adc.S */
 adc1_config_channel_atten(ADC1_CHANNEL_6, ADC_ATTEN_DB_11);
 adc1_config_width(ADC_WIDTH_BIT_12);
 adc1_ulp_enable();
 ulp_low_threshold = 2.3 * (4095 / 3.3); //2 volt

 /* Set ULP wake up period to 100ms */
 ulp_set_wakeup_period(0, 100 * 1000);

 /* Disable pullup on GPIO15, in case it is connected to ground to suppress
    boot messages.
```

```
 */
 rtc_gpio_pullup_dis(GPIO_NUM_15);
 rtc_gpio_hold_en(GPIO_NUM_15);
}

boolean BLYNK_ready(){

}
BLYNK_WRITE(V2)
{
 R = param.asInt();

 for(int i=0;i<NUM_LEDS;i++){
  leds[i] = CRGB(R,G,B);
 }
 FastLED.show();
}

BLYNK_WRITE(V3)
{
int G = param.asInt();

for(int i=0;i<NUM_LEDS;i++){
  leds[i] = CRGB(R,G,B);
 }

FastLED.show();
}

BLYNK_WRITE(V4)
{
 B = param.asInt();
 for(int i=0;i<NUM_LEDS;i++){
  leds[i] = CRGB(R,G,B);

}
FastLED.show();
}

boolean RFID_read(){
// Look for new cards
printf("Looking for new cards..\n");
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance
mfrc522.PCD_Init(); // Init MFRC522
if ( ! mfrc522.PICC_IsNewCardPresent()) {
 printf("No card found..\n");
 //delay(2000);
 return false;
```

```
}

// Select one of the cards
if ( ! mfrc522.PICC_ReadCardSerial())  {
  printf("Selecting  a card to read..\n");
  boolean  tag_read = true;
  return  false;
}
if (mfrc522.uid.uidByte[0]  == 0x09 &&
    mfrc522.uid.uidByte[1]  == 0x18 &&
    mfrc522.uid.uidByte[2]  == 0xa7 &&
    mfrc522.uid.uidByte[3]  == 0x63){
    correct_card = true;
    printf("Correct  card read, access granted..\n");
    return  true;
    }
}


boolean  power_state(){
  if (digitalRead(12)  == HIGH){
    printf("PSU  is connected..\n");
    return  true;
  }
  else{
    printf("PSU  is NOT connected..\n");
    return  false;
  }
}


void  LED_alarm(){
  for(n=0;  n<3;  n++){
    for(i=0;  i<NUM_LEDS;  i++)
      leds[i]  = CRGB::Red;  // CRGB::{255,255,255,255}  // color [4] = {r, g, b, brightness}
    FastLED.show();
    delay(250);
    for(k=0;  k<NUM_LEDS;  k++)
      leds[k] = CRGB::Black;
    FastLED.show();
    delay(250);
  }
  return;
}

void  shut_down(){
  printf("Shutdown  initiated..  Remove  PSU cord!\n");
  while(digitalRead(12)  == HIGH){
```

```
    printf("WARNING!  PSU still  connected!\n");
    LED_alarm();
    delay(1000);
  }
  //summeri();
  printf("Hatch  closing  initiated..\n");
  delay(5000);
  stepmotor_close();
  //ESP_sleep();
  return;
}

void startup(){
  printf("Hatch  opening  initiated..\n");
  stepmotor_open();
  while(digitalRead(12)  == LOW){
    printf("Hatch  opened, waiting  for PSU cord..\n");
    delay(1000);
  //battery_charge();
  }
  LED_alarm();
  return;
}

void stepmotor_close(){
  myStepper.setSpeed(150);
  // step one revolution  in one direction:
  printf("Closing  the hatch..\n");
  myStepper.step(GearedStepsPerRev);
}

void stepmotor_open(){
  myStepper.setSpeed(150);
  // step one revolution  in the other direction:
  printf("Opening  the hatch..\n");
  myStepper.step(-GearedStepsPerRev);
}

void loop()  {
  printf("Loop  starting..\n");
  while(RFID_read()  == 0 && digitalRead(12)  == HIGH){
    RFID_read();
    Blynk.run();
  }
  while(RFID_read()  == 0 && digitalRead(12)  == LOW){
    RFID_read();
  }
  printf("Right  card found! Continuing..\n");
```

```
  if(correct_card  == true && power_state()){
    printf("Correct  card found  and power is on, lets shutdown!\n");
    shut_down();
    delay(100);
    start_ulp_program();
    ESP_ERROR_CHECK(  esp_sleep_enable_ulp_wakeup()  );
    esp_deep_sleep_start();
  }
  else if(correct_card  == true && power_state() == false){
    printf("Correct  card found  and power is NOT on, lets startup!!\n");
    startup();
  }
  else{
    printf("Didn't  find  the right  card. Lets try again..\n");
  }
}
```