OULUN AMMATTIKORKEAKOULU

Oskari Siipola

# USB-YHDISTETTÄVYYSTESTIN AUTOMATISOINTI

# USB-YHDISTETTÄVYYSTESTIN AUTOMATISOINTI

Oskari Siipola
Opinnäytetyö
Kevät 2022
Tietotekniikan tutkinto-ohjelma
Oulun ammattikorkeakoulu

**TIIVISTELMÄ**

Oulun ammattikorkeakoulu
Tietotekniikan tutkinto-ohjelma, Laite- ja tuotesuunnittelu suuntautumisvaihtoehto

---

Tekijä: Oskari Siipola
Opinnäytetyön nimi: USB-yhdistettävyystestin automatisointi
Työn ohjaaja: Kari Jyrkkä
Työn valmistumislukukausi ja -vuosi: Kevät 2022 Sivumäärä: 19

---

Opinnäytetyön aiheena oli USB-yhdistettävyystestin automatisointi ja tavoitteena oli suunnitella ja rakentaa laite, jolla tämän testin voi suorittaa automaattisesti. Laite suunniteltiin helpottamaan testaustyötä, sillä aiemmin testi vaati aina yhden testaajan pakollista läsnä olemista koko testin ajan. Aiemmin testi oli suoritettu manuaalisesti USB-johtoa kytkemällä, joten laitteen pitäisi pystyä simuloimaan USB-johdon irrotusta ja kytkemistä.

Laitteen suunnittelu aloitettiin varmistamalla laitteen vaatimukset. Kun vaatimukset olivat tiedossa, pystyttiin aloittamaan laitteen ja sen ohjelmistojen suunnittelu ja valittiin komponentit. Suunnitelmien perusteella rakennettiin laitteesta prototyyppiversio, jolla voitiin testata laitteen toiminnallisuus. Laitteen toiminnallisuus testattiin oikeassa ympäristössä. Testitulokset osoittavat, että laite pystyi simuloimaan USB-kytkemistä ja että testi pystyttiin nyt suorittamaan automaattisesti.

---

Asiasanat: USB, Automatisointi, Testaus

## ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Information Technology, Option of Device and Product Design

Author: Oskari Siipola
Title of thesis: Automation of USB connectivity test
Supervisor: Kari Jyrkkä
Term and year when the thesis was submitted: Spring 2022
Number of pages: 19

The goal of this thesis work was to create a device that could handle the USB connectivity test automatically. This device was made to ease testers workload as previously testers attention was needed for the whole duration of the test. In the manual version of USB connectivity test car infotainment system is tested to be able to handle 100 consecutive USB connections between mobile device and start projection from the mobile device in all connections. The device was made that it would have the same functionality what manual test has. The device disconnects and reconnects the USB connection 100 times and monitors that projection is started on the infotainment screen.

Keywords: USB, Automatisation, Testing

# SISÄLLYS

# 1 JOHDANTO

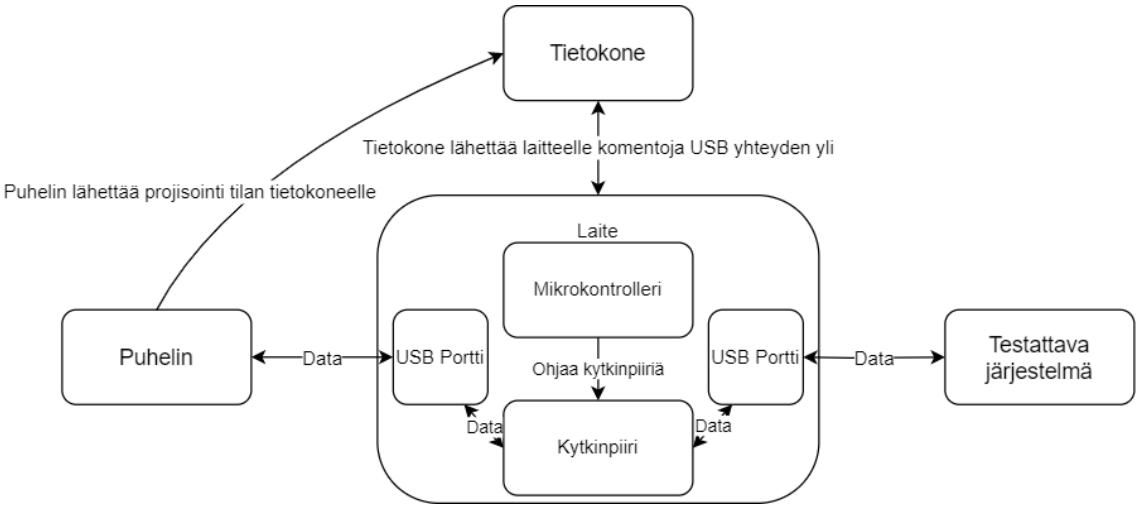
Nykyään testausta suoritetaan hyvin automaattisesti ja monet testit on pystytty automatisoimaan. Automaattisten testien yksi suurin etu on, että ne eivät vaadi fyysistä työtä vaan koneet ja ohjelmat hoitavat testin tekemisen. Koneet eivät myöskään tee inhimillisiä virheitä vaan suorittavat testit aina samalla tavalla. Automaattisen testin suorituksen jälkeen testaajan pitää vain analysoida tulokset tai lukea valmis analyysi.

Symbiolla testaan autojen tietoviihdejärjestelmiä, joille suoritetaan erilaisia testitapauksia. Testitapaukset testaavat tietoviihdejärjestelmän näytölle puhelimelta projisoidun sovelluksen toimintoja. Yksi testitapaus on manuaalitesti, jossa testataan USB-yhdistettävyyttä. USB eli Universal Serial Bus on kaksisuuntainen sarjaliikennekäyttöliittymä, jonka avulla kaksi laitetta voi kommunikoida keskenään (1). Testissä kytketään USB:n välityksellä puhelin testattavaan järjestelmään. Kun projisointi näkyy järjestelmän näytöllä, tiedetään että USB-yhteys onnistuneesti luotu ja yhteys voidaan tämän jälkeen katkaista. Yhdistyksiä toistetaan 100 kertaa ja näiden aikana järjestelmän pitää toimia luotettavasti. Tämä testi vie paljon aikaa ja testissä testaajan pitää fyysisesti yhdistellä USB-johtoa testilaitteiden välillä.

Tämän opinnäytetyön tavoite on suunnitella ja toteuttaa laite, joka voisi suorittaa USB-yhdistettävyystestin automaattisesti. Laitteen pitäisi siis pystyä itse kytkemään USB-yhteyttä ja tarkastelemaan, toimiiko testattava järjestelmä luotettavasti.

# 2 LAITTEEN VAATIMUKSET

Jos tarkastellaan USB-yhdistettävyystestin vaiheita, voidaan niistä koostaa toteutettavalle laitteelle vaatimuksia. Koska laite tulee testattavan järjestelmän ja puhelimen väliin, tulee laitteessa olla kaksi USB-lähtöä kuvan 1 mukaisesti. USB-liittimien väliin pitää saada jokin piiri, joka pystyy katkomaan USB-porttien välistä yhteyttä. Tässä voidaan käyttää kytkinpiiriä, jolla voidaan katkaista USB-väylien väliset data yhteydet. Kun datayhteydet katkaistaan, testattava järjestelmä ja puhelin eivät enää voi kommunikoida keskenään. Viimeisenä laite vaatii vielä älyn, joka pystyy ohjaamaan kytkinpiiriä. Äly laitteelle saadaan mikrokontrollerilla, joka saa komentoja tietokoneohjelmalta. Jotta laite pystyy täysin automaattiseen yhteyden katkontaan, pitää tietokoneohjelman tietää, milloin projisointi on onnistuneesti alkanut. Tästä syystä puhelimelta pitää vielä saada viesti tietokoneelle.




*KUVA 1. Arkkitehtuurikuva laitteen toiminnasta*

## 2.1 USB-liitin

USB-liitintyyppejä on useanlaisia ja eri tyypit tukevat eri USB-standardeja. Yleisin USB-tyyppi on USB-A, joka tukee USB 2.0 -standardia. USB 2.0 tukee 480 Mbit/s:n siirtonopeutta (1). USB 2.0:n jälkeen on tullut jo USB 3.0-standardi, joka tukee jopa 4,8 Gbit/s:n tiedonsiirtonopeutta (1) ja suurin osa Symbiollakin testattavista järjestelmistä tukee tätä. Yleensä testattavista järjestelmistä löytyy USB-A- tai USB-C-tyyppinen portti, johon puhelin kytketään. USB 3.0-standardijärjestelmät tukevat myös USB 2.0 -standardia. Laitteeseen kannattaa valita USB-standardi ja portti, jotka tukevat mahdollisimman montaa testattavaa järjestelmää.

## 2.2 Mikrokontrolleri

Mikrokontrollerilla on tärkeä rooli laitteen toiminnassa. Mikrokontrolleri ohjaa laitteen kytkinpiiriä tietokoneohjelmalta saaduilla komennoilla. Yleensä kun mikrokontrollereita valitaan, tarkastellaan niiden energiatehokkuutta ja prosessointitehoa. Koska laitteen toiminta on melko yksinkertainen, mikrokontrolleri ei tarvitse paljon prosessointitehoa, jolloin voidaan myös valita energiatehokas mikroprosessori. Tähän laitteeseen tärkeimmät vaatimukset ovat, että mikrokontrolleri voi kommunikoida USB:n välityksellä ja että siitä löytyy ainakin yksi GPIO-datalähtö kytkinpiirin ohjaukseen. GPIO eli General-Purpose Input/Output on määrittämätön elektronisen piirin digitaalinen pinni, joka pystytään määrittämään lähdöksi, tuloksi tai molemmiksi (2). Mikrokontrolleri valittaessa tärkeitä huomioon otettavia asioita ovat myös saatavuus ja tuetut ohjelmistotyökalut.

## 2.3 Kytkinpiiri

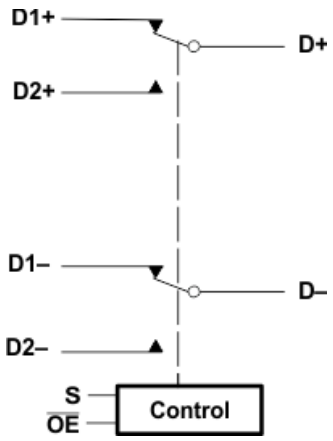
Kytkinpiirin toteuttamiseen voi käyttää erilaisia vaihtoehtoja. Kuvassa 2 näkyy USB-kaapelissa kulkevat signaalit, joita tarvitaan USB 2.0 -standardiyhteyden toimintaan. Yksinkertaisin vaihtoehto kytkinpiirin toteuttamiseen on käyttää relepiiriä, joka katkoo VBUS-linjaa puhelimen ja testattavan järjestelmän välillä. Kun VBUS-linja katkaistaan puhelimen ja testattavan järjestelmän väliltä, USB-dataväylät lakkaavat myös siirtämästä dataa. Toinen vaihtoehto on käyttää TS3USB30 -väyläkytkintä. Tällä piirillä voidaan katkaista kuvan 3 mukaisesti USB-dataväylät, joita on USB 2.0 -standardissa kaksi.

**USB Pinout, Cable Assembly**

| Pin | Signal name | Description |
|---|---|---|
| 1 | VBUS | Red |
| 2 | D- | White |
| 3 | D+ | Green |
| 4 | GND | Black |
| Shell | Shield | Drain |


**USB Pin Location, Type A Connector**

*KUVA 2. USB type A liitin (1)*




*KUVA 3. TS3USB30 toimintalohko (3)*

## 2.4 Ohjelmistot

Jotta laite saadaan toiminaan USB-yhdistettävyystestin mukaisesti, pitää ohjelmistot suunnitella vastaamaan manuaalitestin vaiheita. Vähimmäismäärä ohjelmistoja, mitä laite tarvitsee toimiakseen, on kaksi. Ensimmäinen ohjelma huolehtii tietokoneelta testin suorittamisesta ja testin tuloksien raportoinnista. Tämä ohjelma antaa myös mikrokontrollerille käskyjä ja monitoroi aina tarvittaessa puhelimen projisointitilaa. Tietokoneohjelman tulee siis pystyä kommunikoimaan myös puhelimen kanssa. Toinen ohjelma on mikrokontrollerilla suoritettava ohjelma, joka vastaanottaa komentoja ja huolehtii kytkinpiirin ohjaamisesta. Näiden ohjelmien välille pitää sisällyttää USB:n kautta kommunikointi eli pitää valita tähän sopiva protokolla ja suunnitella protokollan kautta kulkevat viestit. Näillä kahdella ohjelmistolla voidaan jo saavuttaa täysin automaattinen testiympäristö.

# 3 LAITTEEN SUUNNITTELU JA TOTEUTUS

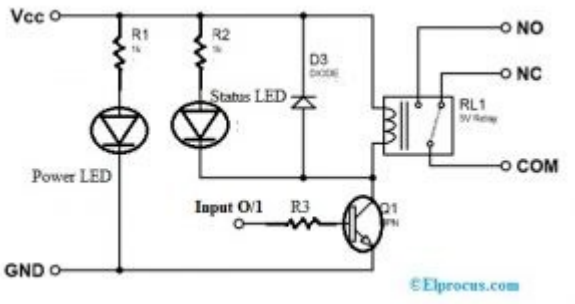
Kun tiedetään laitteen ja sen eri komponenttien vaatimukset, voidaan alkaa suunnittelemaan laitetta. Suunnittelu voidaan aloittaa valitsemalla komponentit, joita laitteessa käytetään. Komponenttien valitsemisen jälkeen suunnitellaan, millaiset kytkennät niiden välille tehdään. Laitteen suunnittelun jälkeen aloitetaan rakentamaan prototyyppiversio laitteesta.

Kun prototyyppiversio laitteesta on valmis, kannattaa mikrokontrollerin ja releen yhteistoiminta tarkistaa testikoodilla. Tämä koodi voi esimerkiksi katkaista ja yhdistää USB-yhteyttä tiettyjen ajastimien avulla. Tämä kannattaa tehdä, jotta voidaan varmistua toiminnasta myös oikealla koodilla.

Laiteen toiminnallisuuden tarkistuksen jälkeen voidaan aloittaa suunnittelemaan laitteen mikrokontrollerin ja tietokoneohjelman koodia ja myös rajapintaa näiden välille. Kun koodit ja rajapinnat ovat suunniteltu valmiiksi, aloitetaan niiden rakentaminen. Rakennusvaiheessa voidaan myös testata vielä laitteen ja tietokoneen välistä kommunikointia testikoodien avulla. Testikoodit sisältävät oikeat protokollaviestit ja toiminnot. Toimintoaja testikoodissa ei välttämättä tarkisteta oikeasti, vaan toiminnot suoritetaan ajastimilla.

## 3.1 Komponenttien valitseminen ja niiden kytkennät

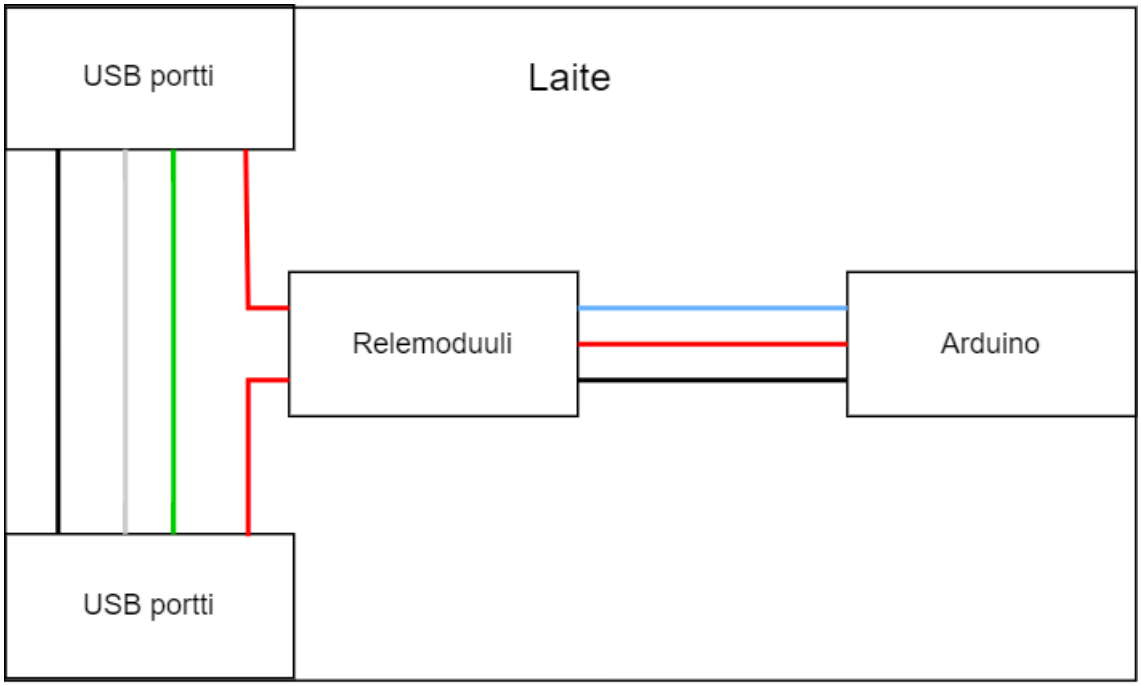
Komponentteja, jotka laitteeseen pitää valita ovat mikrokontrolleri, USB-portit ja kytkinpiiri. Mikrokontrolleriksi valikoitui ATmega328-mikrokontrolleri. Tämä mikrokontrolleri täyttää kaikki sille vaaditut vaatimukset. Koska tämä mikrokontrolleri löytyy myös Arduino Nano -kehityskortista, voitiin kehityskorttia hyödyntää myös laitteen prototyyppiversiossa. Kehityskortin etu on, että sen ympärille on helppo ja nopea rakentaa kytkennät. USB-porteissa kannattaa käyttää USB-A-tyyppiä ja USB 2.0 -standardia, sillä tällä yhdistelmällä voidaan testata myös USB 3.0 -standardia tukevia järjestelmiä. Kytkinpiirissä tullaan käyttämään 5VDC-relettä, koska pelkän VBUS-väylän katkominen on helpompi toteuttaa kuin kahden dataväylän katkominen. Myös TS3USB30-väyläkytkimien saatavuus on heikompi kuin 5VDC-releen. Relettä voidaan ohjata kuvan 4 mukaisella kytkennällä, johon mikrokontrolleri voidaan kytkeä. Mikrokontrolleri kytketään kuvassa 4 näkyvään Input I/O-tuloon.




*KUVA 4. Kytkinpiiri, joka sisältää releen (5)*

Laitteen prototyyppi versiossa hyödynnettiin KY-019-piirimoduulia, sillä siihen on sisällytetty tarvittavat kytkennät ja 5VDC-rele (4). Laitteessa rele kytkettiin USB-porttien VBUS-linjojen väliin kuvan 5 mukaisesti. Linjan toinen pää kytkettiin releen COM-porttiin ja toinen NC-porttiin. Tällaisella kytkennällä VBUS-linja on yhdistettynä USB-porttien välillä, kun rele ei ole aktiivinen. Vastaavasti kun rele on aktiivinen, VBUS-linjat ovat katkaistuna USB-porttien välillä. Tämä pitää ottaa huomioon, kun suunnitellaan laitteen ja tietokoneen ohjelmistoja.

Mikrokontrollerilta releelle tulee kolme kytkentää. Kytkennät ovat 5V:n jännite, maa eli GND ja releen tilan vaihdolle tulosignaali. Mikrokontrolleri kytketään tietokoneeseen Arduino Nanon USB-porttia hyödyntäen. Näin mikrokontrolleri ja tietokone voivat kommunikoida sarjaportin yli.



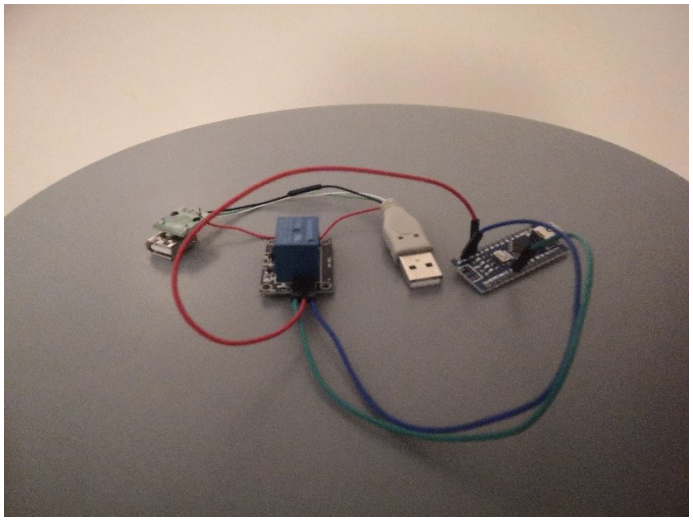

*KUVA 5. Laitteen sisäiset kytkennät.*

## 3.2 Laitteen prototyyppiversio ja releen testaus

### 3.2.1 Laitteen prototyyppiversion rakentaminen

Kun tiedetään laitteen prototyyppiversion rakentamiseen tarvittavat osat ja kytkennät, voidaan aloittaa laitteen rakentaminen. Kytkentöjen tunnistuksen helpottamiseksi käytetään kytkennöissä värikoodattuja johtoja. USB-portteina toimivat yksi USB-naaras- ja yksi USB-uros-portti. Laite on nyt helppo kytkeä puhelimen ja testattavan laitteen välille, kun käytetään kahta eri USB-porttia laitteessa.

Arduino Nano -kehityskortti ja erillinen rele-piirimoduuli olivat helppo yhdistää hyppylankoja käyttäen. Hyppylankojen käyttö on myös hyvä, jos myöhemmässä vaiheessa pitää muuttaa kytkentöjä. Valmis prototyyppiversio näkyy kuvassa 6, josta näkee myös kytkentöjen yksinkertaisuuden.



*KUVA 6. Laitteen prototyyppiversio*

### 3.2.2 Releen toiminnan testaaminen prototyyppiversiossa

Laitteen prototyyppiversion testaus voitiin nyt suorittaa, kun laite oli kasattu valmiiksi. Tällä testauksella haluttiin varmistaa, että laite pystyy katkomaan onnistuneesti puhelimen ja testattavan laitteen välistä yhteyttä. Jotta tämä toiminnallisuus voitiin testata, piti mikrokontrollerille tehdä testiohjelma, joka simuloi laitteiden välisen yhteyden katkomista. Yksinkertaisin tapa toteuttaa simulointi on tehdä ajastimilla katkaisu ja takaisinkytkentä. Tässä tapauksessa rakennettiin ohjelma, joka silmu-

kassa pitää yhteyttä yllä 15 sekuntia ja tämän jälkeen katkaisee yhteyden 15 sekunniksi. Toiminnallisuus testattiin oikealla järjestelmällä ja yhteydenkatkomistoiminnallisuus toimi juuri halutulla tavalla. Kun yhteys katkaistiin, projisointi loppui järjestelmän näytöltä. Vastaavasti kun yhteys palautetaan, myös projisointi palautui.

### 3.3 Laitteen ja tietokoneohjelman koodin suunnittelu ja rakentaminen

Koska laiteessa käytettiin Arduino Nano -kehityskorttia, voitiin hyödyntää Arduinon omaa kehitysympäristöä Arduino IDE -ohjelmaa koodin suunnitteluun ja rakentamiseen. Tietokoneen ohjelmaksi rakennetaan Python-koodi, jonka voi ajaa esimerkiksi Windowsin komentorivillä tai Macin terminaalilla. Python-koodia voi rakentaa esimerkiksi Thonny-kehitysympäristöllä. Arduino-koodi ja Python-koodi kommunikoivat keskenään sarjaportin avulla. Sarjaportin yli kulkevat protokollaviestit pitää suunnitella, jotta Arduino ja Python-koodi ymmärtävät toisiaan.

Jotta tietokone ja puhelin voivat kommunikoida keskenään, pitää Pythonissa hyödyntää Adb-komentoja. Adb eli Android debug bridge on komentorivityökalu, jolla pystytään kommunikoimaan Android-käyttöjärjestelmän sisältävien laitteiden kanssa (6). Adb-työkalua hyödynnettiin, koska testipuhelimet käyttävät Android-käyttöjärjestelmää. Adb-komentoja suoritetaan komentorivillä, joten Python-koodiin tarvitaan lisäosa, jolla pystytään avaamaan koodin sisällä komentorivi. Python-koodiin sisällytettiin os-kirjasto, joka sisältää tämän toiminnallisuuden.

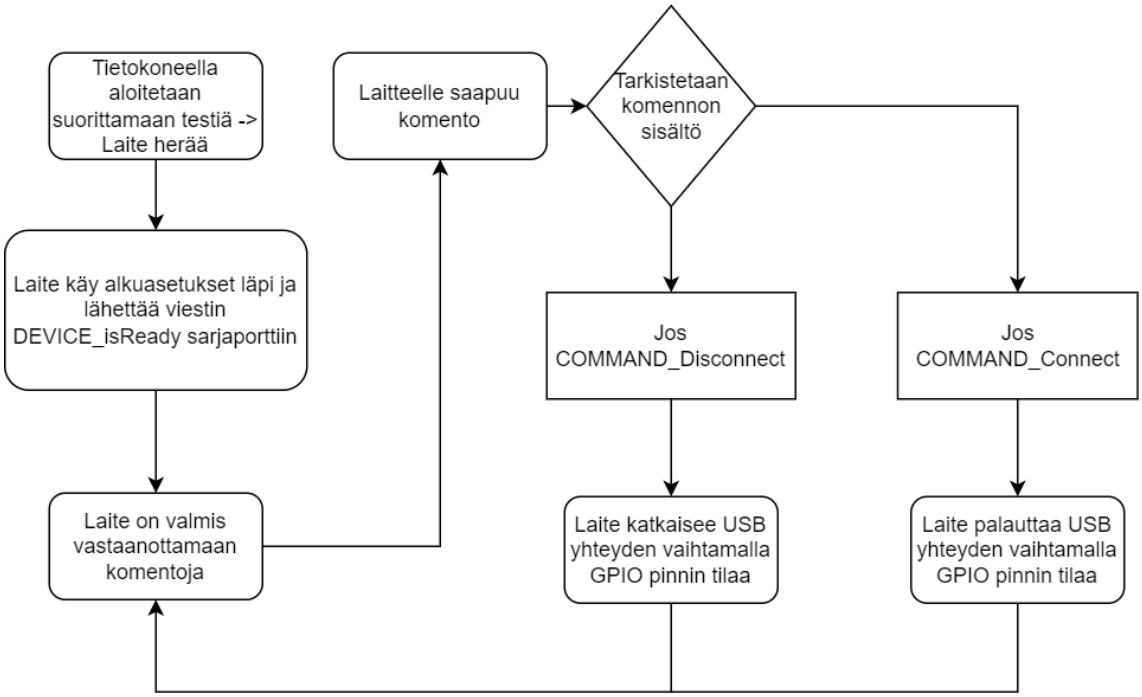
#### 3.3.1 Laitteen ja tietokoneen välinen kommunikointi sarjaportin yli

Sarjaportin yli kommunikointiin käytetään UART-protokollaa eli universaalia asynkronista lähetin-vastaanotinta. UART-protokollan avulla voidaan lähettää ja vastaanottaa tavu kerrallaan dataa sarjaportin yli (7). UART-protokolla valittiin, koska se oli helppo toteuttaa Arduinon ja tietokoneen välille. Arduinosta löytyy jo valmiiksi USB-Sarjaväylä muunnin, joka hyödyntää UART-protokollaa (8). Arduinolla voidaan myös hyödyntää Arduinon sisäänrakennettua Serial kirjastoa, jonka avulla voidaan käsitellä sarjaportin dataa. (8) Python-koodissa sarjaportin dataa päästään käsiksi pySerial moduulilla (9).

Koska UART:in avulla lähetetään viestit tavu kerrallaan, kannattaa viesteillä olla ainakin loppumerkki. Tähän toteutukseen valittiin alku- ja loppumerkit, jotka ovat ”<” alkumerkiksi ja ”>” loppumerkiksi. Näiden merkkien väliin siis voidaan kirjoittaa kaikki komennot ja viestit. Koska tässä toteutuksessa Arduino-koodin ei tarvitse lähettää ollenkaan komentoja, kannattaa komentoviestit lähettää vain Python-koodista, johon Arduino-koodi vastaa aina statusviestillä. Python-koodin komentoviestit laitteelle ovat ”COMMAND_Disconnect” ja ”COMMAND_Connect ”. Näiden avulla Arduino tietää, pitääkö rele kytkeä vai katkaista. Arduinon tilaviestejä ovat ”DEVICE_isReady” ja ”DEVICE_completedTask”. ”DEVICE_isReady” statusviestillä voidaan varmistaa, että Python-koodi tietää laitteen olevan valmis ja kytketty. ”DEVICE_completedTask” statusviesti kertoo Python-koodille, että laite on saanut komennon suoritettua onnistuneesti.

### 3.3.2 Laitteen koodin suunnittelu ja rakentaminen

Arduino-koodin alkuun tulee osio, jossa laite käy alkuasetukset ja kovakoodatut arvot läpi, kuten kuvassa 7 on esitetty. Tässä osiossa laite lähettää myös sarjaporttiin viestin DEVICE_isReady, joka kertoo, että laite on valmis vastaanottamaan komentoja. Seuraava koodin osio on ikisilmukka. Ikisilmukassa laite tarkistaa viestejä, joita se lukee sarjaportista ja reagoi sitten viestin perusteella. Koodissa on omat funktiot tietyille viesteille, mitä se vastaanottaa. Kun koodi vastaanottaa viestin COMMAND_Disconnect, se vaihtaa releen tilan, joka katkaisee USB-yhteyden testattavan järjestelmän ja puhelimen väliltä. vastaavasti COMMAND_Connect viestin vastaanotettuaan laite yhdistää USB-yhteyden takaisin. Jokaisen toiminnon suoritettuaan laite kuittaa onnistuneesta suorituksesta Python-koodille lähettämällä DEVICE_completedTask viestin.



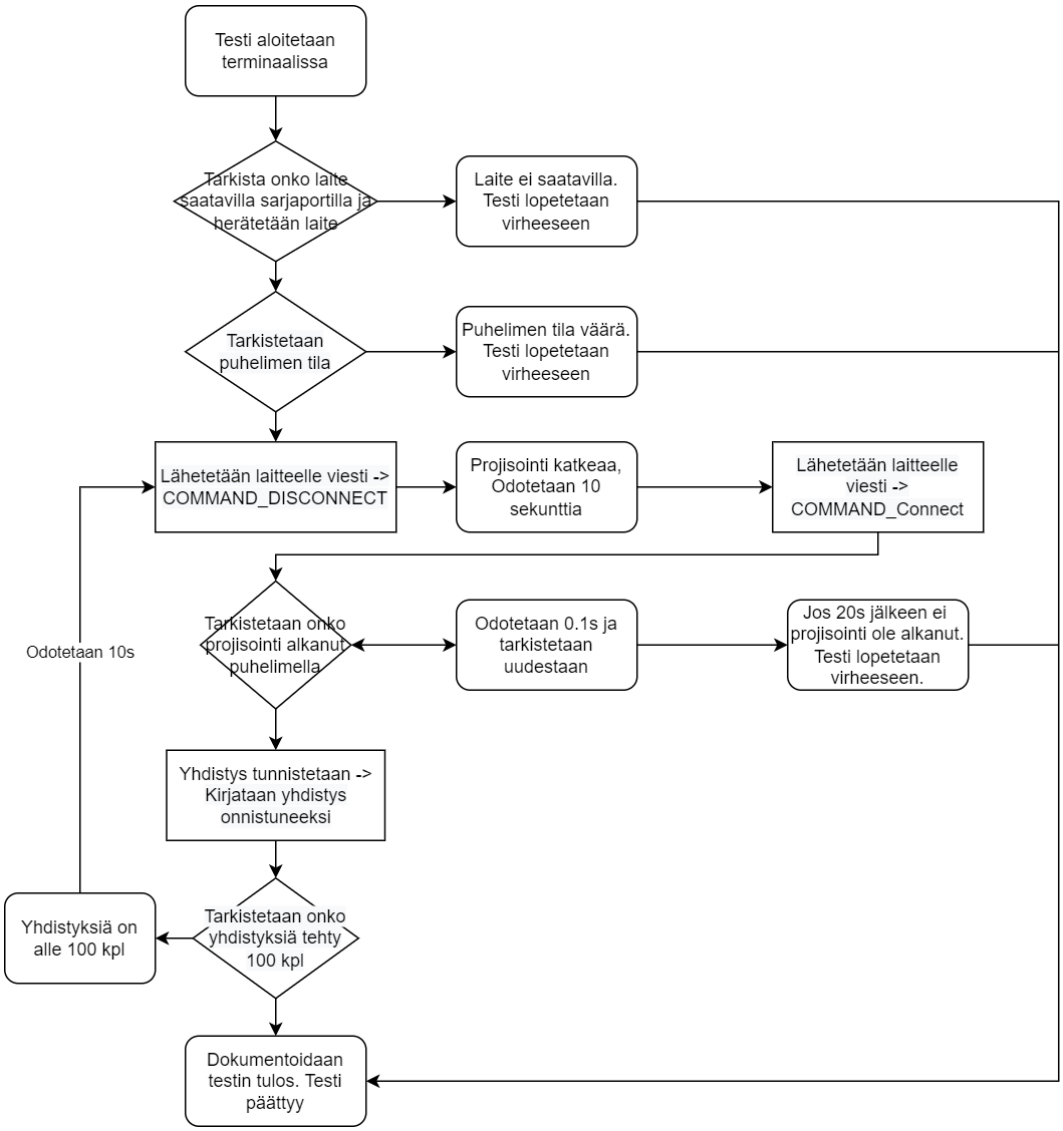

*KUVA 7. Laitteen mikrokontrollerikoodin vuokaavio.*

### 3.3.3 Python koodin suunnittelu ja rakentaminen

Tietokoneella ajettavan Python-koodin alussa pitää kuvan 8 mukaisesti tehdä kaksi tilan tarkistusta, ennen kuin varsinainen testi voi alkaa. Ensimmäinen tarkistettava tila on tarkistaa, että laite kytketty tietokoneeseen. Tietokone avaa sarjaportin ja laitteen pitäisi vastata tähän DEVICE_isReady viestillä. Jos koodi ei löydä laitteelta saatua tilaviestiä, testi päättyy virheeseen. Toinen tarkistettava tila on tarkistaa, että puhelin on oikeassa tilassa. Testin alussa puhelimen on oltava projisointi tilassa. Jos puhelin ei ole testin alussa projisointi tilassa, testi päättyy virheeseen.

Kuvan 8 mukaisesti tilatarkistuksien jälkeen päästään koodin silmukkaan, joka katkoo yhteyttä. Ensin koodi lähettää laitteelle viestin yhteyden katkaisemisesta. Viestin lähettämisen jälkeen odotetaan 10 sekuntia, jotta annetaan testattavalle järjestelmälle hetki aikaa prosessoida yhteyden katkaisua. Kun on odotettu 10 sekuntia, tietokone lähettää viestin laitteelle yhteyden palautuksesta. Kun yhteys on palautettu, aloittaa tietokone silmukan, jossa tarkistetaan sekunnin kymmenesosan välein puhelimen tilaa. Jos puhelin ei saa 20 sekunnin aikana oikeaa puhelimen tilaa, testi päättyy virheeseen.

Kun koodi saa oikean tilan puhelimelta, lasketaan yhdistys onnistuneeksi. Onnistuneen yhdistyksen jälkeen tarkistetaan, onko 100 yhdistystä vielä suoritettu. Jos yhdistyksiä on alle 100, koodi odottaa 10 sekuntia, jonka jälkeen palaa silmukan alkuun ja lähettää yhteyden katkaisuviestin. Kun kaikki 100 yhdistystä on suoritettu, koodi ottaa lopuksi puhelimelta ongelmaraportin. Ongelmaraportista voidaan tarkastella mahdollisia ongelmia, mitä puhelimeen on tallentunut testin aikana.




*KUVA 8. Python-koodin vuokaavio*

# 4 LAITTEEN TESTAUS

## 4.1 Valmistelut ennen laitteen käyttöönottoa

Kun laite oli rakennettu ja ohjelmisto oli valmis, voitiin aloittaa laitteen testaus. Laite testattiin oikealla testattavalla järjestelmällä Symbion tiloissa, jolloin voidaan varmistua laiteen ja ohjelmistojen toimivuudesta. Testaaminen aloitettiin kytkemällä laite kiinni testattavaan järjestelmään, joka onnistui kiinnittämällä uros USB-portti järjestelmän USB-porttiin. Laitteen naaras USB-porttiin kytkettiin USB-johto, jonka toinen pää kytkettiin puhelimen USB-porttiin. Näillä kytkennöillä projisointi pystyttiin aloittamaan järjestelmällä. Seuraavaksi puhelin ja tietokone piti yhdistää yhteiseen WiFi-tukiasemaan, jotta pystyttiin aloittamaan langaton Adb-yhteys niiden välille. Kun Adb-yhteys on muodostettu, voitiin kytkeä Arduino USB-johdolla tietokoneeseen.

## 4.2 Laitteen testaaminen

Kun kaikki tarvittavat valmistelut oli tehty, voitiin aloittaa tietokoneohjelman suorittaminen terminaalissa. Testin aikana laite toimi, kuten oli odotettu. Testattavan järjestelmän projisoinnin katkaisu ja yhdistäminen toimi ja testin aikana saatiin suoritettua haluttu 100 yhdistystä, joka vastaa hyväksyttyä testi tulosta manuaali testistä. Kun ensimmäinen testi oli suoritettu, testattiin laite vielä kolmella eri järjestelmällä ja puhelimella. Tulokset näistä testeistä olivat samanlaisia. Laite siis toimii oikeassa ympäristössä.

# 5 YHTEENVETO

Opinnäytetyön tavoitteena oli suunnitella ja tehdä manuaalisesta USB-yhdistettävyystestistä automaattisesti suoritettava. Tässä tavoitteessa onnistuttiin ja nyt testi pystytään suorittaa automaattisesti. Automaattisesti suoritettava testi säästää nyt testaajien resursseja ja aikaa.

Kun testin voi nyt suorittaa erilliseen laitteen avulla automaattisesti, voidaan alkaa miettimään, voisiko testiin lisätä uusia ominaisuuksia. Esimerkiksi voitaisiin tutkia, miten kauan puhelimella kestää saada yhteys uudelleen kytkennän jälkeen. Laite antaa siis hyvät kehitysmahdollisuudet testin suoritukseen ja sen tarjoamaan tietoon.

Laitetta voisi vielä kehittää enemmän käyttäjäystävällisemmäksi. Laitetta voisi olla helpompi käyttää, jos laiteella olisi kompakti kotelo. Kotelo voisi auttaa laitteen kytkentävaiheessa, koska USB-johdot olisi helpompi kytkeä. Laitteen kytkentöjä voisi parantaa suunnittelemalla piirilevyn, jossa olisi kaikki laitteen komponentit kytkettynä. Piirilevyn suurimpia hyötyjä laitteelle ovat elektronisen häiriön väheneminen ja komponenttien mahduttaminen pienempään tilaan (10). USB-porttien liitäntätyyppiä voisi myös miettiä uudelleen. USB-C tyyppinen portti voisi olla parempi vaihtoehto laitteelle, sillä USB-C-liitäntätyyppi on yleistymässä kaikissa järjestelmissä. Testi ei tällä hetkellä myöskään tue kuin Android-käyttöjärjestelmällä toimivia puhelimia, joten tulevaisuudessa laitetta voisi myös kehittää toimimaan myös muilla puhelinkäyttöjärjestelmillä. Myös Python-koodi on rakennettu tukemaan Windows-käyttöjärjestelmää, joten toimivuus muilla käyttöjärjestelmillä ei ole varma.

# LÄHTEET


1. Universal Serial Bus 2012. USB interface description. Hakupäivä 26.1.2022. http://www.interfacebus.com/USB-Interface-Description.html.
2. Wikipedia 2022. General purpose input/output. Hakupäivä 4.5.2022. https://en.wikipedia.org/wiki/General-purpose_input/output.
3. Texas Instruments 2022. TS3USB30. Hakupäivä 24.2.2022. https://www.ti.com/product/TS3USB30#tech-docs.
4. ArduinoModules 2021. KY-019 5V RELAY MODULE. Hakupäivä 10.4.2022. https://arduinomodules.info/ky-019-5v-relay-module/.
5. Elprocus 2022. What is a 5V Relay Module:Working & Its Applications. Hakupäivä 10.2.2022. https://www.elprocus.com/5v-relay-module/.
6. Android studio 2022. Android debug bridge. Hakupäivä 02.05.2022. https://developer.android.com/studio/command-line/adb.
7. Peňa, Eric & Legaspi, Mary Grace 2020. UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter. AnalogDialogue Vol 58. Hakupäivä 29.3.2022. https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html.
8. Arduino 2022. Serial. Hakupäivä 7.4.2022. https://www.arduino.cc/en/reference/serial.
9. Pythonhosted 2022. pySerial – overview. Hakupäivä 7.4.2022. https://pythonhosted.org/pyserial/pyserial.html#overview.
10. Lee, H.L.M 2019. Advantages of a PCB board. Sciencing. Hakupäivä 8.4.2022. https://sciencing.com/advantages-pcb-board-8261204.html.