TAMPEREEN
AMMATTIKORKEAKOULU

TULOSTUSYMPÄRISTÖ TIETOLIIKENNE-LABORATORIOON

Johannes Pitkänen

Opinnäytetyö
Toukokuu 2018
Tieto- ja viestintätekniikka
Tietoliikennetekniikka ja tietoverkot

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikan koulutusohjelma
Tietoliikennetekniikka ja tietoverkot

PITKÄNEN JOHANNES:
Tulostusympäristö tietoliikennelaboratorioon

Opinnäytetyö 46 sivua, joista liitteitä 3 sivua
Toukokuu 2018

Jo pitkään yleismaailmallisena kehityssuuntana on ollut paperin käytön vähentäminen organisaatioissa. Kouluissa tämä näkyy esimerkiksi kurssimateriaalien siirtymisenä verkkoon. Kulutuksen pienentämisessä motivaattorina on usein ekologiset periaatteet, mutta toisaalta kyse on myös kustannuksista. Tampereen Ammattikorkeakoulun tietotekniikan osaston tietoliikennelaboratoriossa paperille tulostaminen on tähän asti ollut varsin kiinteä osa siellä työskentelyä. Tulostaminen on usein ainoa tapa ottaa mittaustulokset talteen laboratorion mittalaitteilta.

Tämän opinnäytetyön tavoitteena oli suunnitella ja toteuttaa tulostimet laboratoriossa korvaava järjestelmä. Motiiveina toteuttamiselle oli pyrkimykset helpottaa ja nopeuttaa raportointityötä laboratoriokursseilla, jo aiemmin mainittujen ekologisten ja kustannuksellisten syiden lisäksi. Ratkaisu tulostinten korvaamiseksi oli siirtää mittalaitteiden tuottama tulostusdata keskitetylle palvelimelle, jonka kautta tulosteet voitiin välittää käyttäjälle suoraan sähköisessä muodossa. Työ painottui tulostimen korvaavan laitteen kehittämiseen, sekä palvelinohjelmiston toteuttamiseen.

Työ vaati jonkin verran taustatutkimusta laitteiden käyttämistä tiedonsiirtoväylistä ja tulostuksen dataformaateista. Suurimman osan työstä kuitenkin muodosti käytännön tekeminen elektroniikan ja ohjelmoinnin saralla. Tuloksena syntyi järjestelmä, joka ratkaisee useita tähän asti laboratoriotyöskentelyä ja raportointia hankaloittaneita ongelmia. Uusi järjestelmä on myös tulevaisuutta ajatellen kestävämpi ratkaisu, kun verrataan aiempaan tulostimiin perustuvaan toimintamalliin.

Järjestelmän laajamittaista käyttöönottoa ei kuitenkaan toteutettu tämän opinnäytetyön tiimoilta, vaan työssä keskityttiin pääasiassa suunnitteluun ja laiteprototyypin rakenteluun. Tulostimet korvaavien laitteiden laajamuotoisempi valmistaminen sovittiin jäävän myöhemmin tietotekniikan osaston tehtäväksi.

Asiasanat: tulostaminen, tietoliikennelaboratorio, ESP8266, rinnakkaisväylä

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
ICT Engineering
Telecommunications and Networks

PITKÄNEN JOHANNES:
Printing Environment for Telecommunications Laboratory

Bachelor's thesis 46 pages, appendices 3 pages
May 2018

For a long time, it has been a global trend to reduce paper usage in organizations, for ecological and financial reasons. However, in the telecommunication laboratory of Tampere University of Applied Sciences, printing to paper has been an integral part of working there. Printing is the main method for saving measurements from the measurement devices. Using printers is rather cumbersome as many of them are broken in some way or another, and scanning printed papers is extra work.

The aim of this thesis was to develop a new system that would replace printers in the laboratory. Instead of printing to paper, user can get the measurements in all digital form via browser-based user interface. This way reporting process becomes much easier, therefore making laboratory work more convenient altogether. The solution was to capture printing data from the measurement devices and send the data for a dedicated server. A data capture device was designed for this purpose. The capture device replaces a printer and it looks like any other printer for a measurement device.

This thesis mostly focused on the development of the capture device and the server software, although some basic background theory was also covered. The development resulted in a new system, which solves many of the problems there used to be with the usage of printers. The new printing environment benefits new students and the school, as the new system is easier to manage and more future-proof solution.

Key words: printing, telecommunication laboratory, ESP8266, parallel port

SISÄLLYS

1 JOHDANTO 7
2 MOTIIVIT JÄRJESTELMÄN TOTEUTTAMISELLE 8
2.1 Lähtötilanne 8
2.2 Ekologisuus 9
2.3 Huoltovarmuus 9
2.4 Raportointityön sujuvoittaminen 10
3 JÄRJESTELMÄKOKONAISUUDEN KUVAUS 11
3.1 Mittalaite 11
3.2 Lukija 12
3.3 Langaton lähiverkko 13
3.4 Palvelin 14
3.4.1 Käyttöliittymä 14
4 TIEDONSIIRTOVÄYLÄT JA SIVUNKUVAUS 17
4.1 Rinnakkaisväylä 17
4.1.1 Centronics-kättely 18
4.2 Sarjaväylä 19
4.3 Sivunkuvauskielet 20
5 LUKIJALAITTEEN TOTEUTUS 21
5.1 ESP8266 21
5.1.1 Spesifikaatiot 21
5.1.2 GPIO 22
5.1.3 Arduino-ohjelmointiympäristö 24
5.2 Kättelyn elektroninen toteutus 24
5.2.1 SR-salpa 25
5.3 Multiplekseri 27
5.4 Puskurirekisteri 27
5.5 Sarjaväyläadapteri 28
5.5.1 Tasomuutos 28
5.6 Regulaattorikytkentä 29
5.7 Ohjelmallinen toteutus 30
5.7.1 Wifi-kirjastot 31
5.7.2 Setup() –alustusfunktio 31
5.7.3 Loop() –pääsilmukka 32
6 PALVELINOHJELMOINTI 34
6.1 Virtuaaliserveri 34
6.2 Taustaosa 34

6.2.1 Node.js 35
6.2.2 MongoDB 36
6.2.3 GhostPCL 37
6.3 Käyttöliittymä 38
6.3.1 Angular 38
6.3.2 Bootstrap 39
7 POHDINTA 40
LÄHTEET 41
LIITTEET 44
Liite 1. Lukijalaitteen kytkentäkaavio. 44
Liite 2. Sarjaväyläadapterin kytkentäkaavio. 45
Liite 3. Lukijalaitteen toiminnan vuokaavio. 46

LYHENTEET JA TERMIT

ack	signaali rinnakkaisväylässä, kertoo kun tulostin on vastaanottanut datan onnistuneesti
Angular	sovelluskehys selainpohjaisten sovellusten kehittämiseen
Bootstrap	kirjasto responsiivisten web-sivujen kehittämiseen
busy	signaali rinnakkaisväylässä, kertoo kun tulostin prosessoi vastaanottamaansa dataa
Centronics	tulostinvalmistaja, nykyään viittaa myös rinnakkaisväylään
ESP8266, ESP	lukijalaitteessa käytetty WiFi-mikrokontrolleri
GET	HTTP-metodi datan pyytämiseen palvelimelta
GPIO	mikrokontrollerin pinni, joka voi toimia lähtönä tai tulona
lukijalaite	työssä toteutettu tulostimen korvaava laite
MongoDB	palvelimen käyttämä tietokantaohjelmisto
Node.js	JavaScript-ajoympäristö
PCL	tulostinten käyttämä sivunkuvauskieli
POST	HTTP-metodi datan lähettämiseen palvelimelle
PostScript	tulostinten käyttämä sivunkuvauskieli
rinnakkaisväylä	käytöstä jo poistunut tulostinten käyttämä tiedonsiirtoväylä
RS-232	sarjaväylästandardi
sarjaväylä	yksinkertainen tiedonsiirtoväylä, myös tulostuksessa vanhastaan käytetty
SSID	langattoman lähiverkon identifioiva tunnus
strobe	signaali rinnakkaisväylässä, kertoo siirrettävästä tavusta
SQL	tietokantojen kyselykieli
TTL	digitaalitekniikan logiikkaperhe

1 JOHDANTO

Tämän opinnäytetyön lähtökohtana oli kehittää Tampereen Ammattikorkeakoulun tietoliikenteen osaston tietoliikennelaboratoriossa työskentelyä helpottava ja suoraviivaistava järjestelmä. Laboratoriokursseilla tehtävistä mittaustöistä tulee kirjoittaa kirjalliset raportit, joihin on usein tarpeen sisällyttää laboratorion mittalaitteilla saatuja mittaustuloksia. Tähänastinen toimintamalli on ollut ottaa mittaustulokset talteen tulostamalla ne paperille. Tulostimien kanssa toimiminen kuitenkin hankaloittaa ja hidastaa mittaustöiden tekemistä ja raportointityötä.

Edellä mainitut ongelmat muodostivat lähtöasetelman, jonka pohjalta uutta järjestelmää alettiin suunnittelemaan. Varsin nopeasti selveni, että tulostimille tarkoitetun datan kaappaaminen on kaikkein järkevin tapa saada mittaustulokset laitteilta sähköisessä muodossa talteen. Merkittävä osa työstä keskittyykin tulostimen korvaavan ns. lukijalaitteen suunnitteluun ja toteutukseen. Toinen painopiste on palvelimen ja käyttöliittymän ohjelmointityössä.

Kehitystyössä lähdettiin liikkeelle laitteiden käyttämien tiedonsiirtoväylien toimintaan ja tulostusdataformaatteihin perehtymisellä, joskin työ kokonaisuudessaan painottuu enemmän käytännön työhön tutkimuksen sijaan. Tämä opinnäytetyöraportti keskittyy toteutetun järjestelmän osasten toiminnan ja rakenteen tarkasteluun, edellä mainittujen teoreettisten seikkojen ohella.

2 MOTIIVIT JÄRJESTELMÄN TOTEUTTAMISELLE

2.1 Lähtötilanne

Tampereen Ammattikorkeakoulun tieto- ja viestintätekniikan koulutusohjelmaan sisältyy, opintosuuntauksesta riippuen, vaihteleva määrä tietoliikennetekniikan opintoja, joihin sisältyy myös laboratoriokursseja. Näillä kursseilla työskentely tapahtuu osaston tietoliikennelaboratoriossa. Pääosa kurssien töistä sisältää erinäisten mittausten tekemistä mittalaitteilla, joita laboratoriossa on useita erityyppisiä. Tyypillisiä laboratorion laitteita ovat esimerkiksi oskilloskoopit, radiokommunikaatiotesterit ja piirianalysaattorit, kuten kuvassa 1.

KUVA 1. Piirianalysaattori HP 8753E.

Laitteilla saatavat mittaustulokset ovat poikkeuksetta visuaalista informaatiota, siis erinäisiä kuvaajia (spektrikuvaajat ym.). Tällaiset kuvaajat muodostavat olennaisen osan kurssien mittaustöistä kirjoitettavista laboratoriotyöselostuksista, mikä luonnollisesti tarkoittaa sitä, että kuvaajat tulee saada jollakin tavalla talteen laitteilta. Muutamissa laboratorion uusimmista mittalaitteista on mahdollisuus mittaustulosten tallentamiseen USB-muistitikulle, mutta valtaosa laitteista on kuitenkin ajalta ennen USB-liitäntää, tai ainakin sen yleistä käyttöä. Näiden laitteiden tapauksessa ainoa tapa ottaa mittaustulokset talteen, on paperille tulostaminen. Tämän vuoksi tietoliikennelaboratoriossa onkin tulostimia lähemmäs parikymmentä kappaletta.

Paperiset tulosteet ovat kuitenkin varsin epäkäytännöllisiä, koska niitä ei voida hyödyntää suoraan laboratoriotöistä tehdyissä raporteissa, sillä saadut kuvaajat halutaan selostuksissa tekstin sekaan. Siksi tulosteet ovatkin lähinnä väliaikaisen ”tallennusmedian” roolissa, sillä ne päätyvät skannauksen jälkeen suoraan paperinkeräykseen. Lisäksi skannaus pakollisena välitoimenpiteenä tekee raportointityöstä työläämpää. Nämä seikat olivat pääasiallisina motivaattoreina vaihtoehtoisen järjestelmän ja toimintamallin kehittämiselle, vaikka toteutuksella saavutettiin toki muitakin lisäetuja.

2.2 Ekologisuus

On ilmiselvää, että kertaluontoisten tulosteiden käyttö ei ole kovinkaan ekologisella pohjalla, etenkin kun asiaa katsoo pidemmällä aikaskaalalla. Jo pitkään yleisenä trendinä on ollut paperin käytön vähentäminen, joten siinäkin mielessä uudenlaisen järjestelmän toteuttaminen sopii hyvin ajan henkeen. Toki vähäisempi paperin kulutus näkyy myös rahallisena säästönä, joskin ammattikorkeakoulun kokoisessa organisaatiossa tämä vaikutus jäänee melko merkityksettömäksi.

Yksi laboratoriokurssin työ saattaa käsittää parikymmentäkin mittausta ja siten yhtä monta tulostettavaa kuvaajaa. Kun otetaan huomioon ryhmien määrä, kurssin aikana tulee tulostetuksi satoja papereita jo yhden laboratoriotyön tiimoilta. Paperin kulutusta lisäävät myös virheellisistä mittauksista aiheutuneet turhat tulosteet. Tästä saa yleiskuvaa siitä, kuinka paljon paperia kokonaisuudessaan kuluu tietoliikennelaboratorion toimesta, ja kun huomioidaan tulosteiden lyhyt elinikä, voidaan nykyistä toimintamallia pitää paperin suorana tuhlaamisena.

2.3 Huoltovarmuus

Tulostimien käyttöä hankaloittaa niiden vaatima ylläpito. Etusijalla on mustekasettien vaihtaminen uusiin aina tarvittaessa, mutta toki tulostimet kuluvat käytössä muutenkin, useiden liikkuvien osiensa vuoksi. Tulostinmallien elinkaari on yleisesti ottaen varsin lyhyt, eikä laitteiden korjaamista tehdä käytännössä lainkaan, mikä taas luo tarpeen uusien laitteiden investoinnille. Koulun vuosia jatkuneet säästötarpeet huomioiden, on kuitenkin selvää, ettei tällaisia investointeja olla juurikaan valmiita tekemään.

Tietoliikennelaboratorion lukuisista tulostimista moni on tavalla tai toisella epäkunnossa. Yleisintä on kuluneiden osien myötä vikaantunut paperinsyöttö, toisaalta esimerkiksi joskus taas paperi tulee laitteesta ulos ryppyisenä. Joidenkin laitteiden tulostusjälki on hyvin suttuista, mikä toki saattaa johtua lähinnä kuluneista mustekaseteista. Aina silloin tällöin tulostin saattaa jumiutua myös täysin ohjelmallisista syistä. Tällaiset ongelmat hankaloittavat tulosteiden kanssa toimimista entisestään. Korvaava, täysin sähköinen järjestelmä on sen sijaan lähes huoltovapaa.

2.4 Raportointityön sujuvoittaminen

Edellä mainituista ongelmista saa yleiskuvan lukuisista raportointityötä hankaloittavista käytännön tekijöistä. Tulostimien ongelmat ja tulosteiden skannaaminen ovat opiskelijalle täysin turhia lisätyön aiheuttajia, ja siten myös suoraan pois itse mittauksiin ja raportointiyöhön keskittymisestä. Laboratoriokurssin työkerroilla saattaa kulua yllättävänkin paljon aikaa tulostusongelmien selvittelyyn.

Laboratorion ulkopuolella aikaa kuluu itse skannaamisen lisäksi monitoimilaitteelta saadun skannatun materiaalin käsittelyyn. Monitoimilaitteet luovat skannauksista monisivuisia PDF-tiedostoja, joista halutut kuvaajat täytyy kaapata talteen, rajata ja tallentaa kuvatiedostoiksi. Monitoimilaitteen arkkiskanneria käyttämällä itse skannausoperaatioon ei toki kauaa kulu, mutta arkkiskannerin käyttökään ei aina ole ongelmatonta; joskus lopputuloksessa skannatut sivut saattavat olla muutaman asteen vinossa, mikä taas tarkoittaa lisää työtä tulosteiden suoristamiseksi tai vaihtoehtoisesti uudelleen skannaamista. Tulosteiden talteenotto mittalaitteilta suoraan sellaisenaan poistaa edellä mainitut ongelmat ja siten tekee raportointityöstä tältä osin nopeampaa ja suoraviivaisempaa.

3 JÄRJESTELMÄKOKONAISUUDEN KUVAUS

3.1 Mittalaite

Koko järjestelmän keskiössä on tietenkin tulosteiden alkulähde, eli laboratoriossa käytetyt mittalaitteet. Yleisluontoisempiin elektroniikan mittaustehtäviin käytettäviä mittalaitteita ovat mm. oskilloskoopit ja spektrianalysaattorit. Osaa laitteista taasen käytetään hyvin spesifeihin mittauksiin, esimerkiksi GSM-järjestelmään liittyen. Mittaustulokset ovat yleensä erilaisia kuvaajia (käyrät ym.), mutta joidenkin laitteiden/mittausten tapauksessa ne voivat olla myös esimerkiksi taulukkomuotoista informaatiota.

Laboratorion mittalaitteista valtaosan valmistusajankohta ajoittuu 90-luvulle, alkupäähän painottuen. Ikää laitteille on kertynyt, koska niiden uusimiseen ei erityisemmin ole mitään tarvetta. Useimmilla laitteista mitataan sellaisia asioita, jotka eivät muutu ajan saatossa, siis esimerkiksi fysikaaliset ilmiöt. Vanhojen laitteiden kääntöpuolena on kuitenkin käytöstä poistuneiden liitäntöjen ja tallennusmedioiden käyttö. Tulostamisen ohella joillakin laitteilla on mahdollista tallentaa mittaustuloksia levykkeille, joka nykypäivänä on auttamatta vanhentunut teknologia, eikä levykeasemia enää tietokoneissa juuri näe. Tulostusdatan kaappaaminen jääkin ainoaksi järkeväksi tavaksi saada mittaustulokset talteen suoraan sähköisessä muodossa. Koska nämä vanhemmat mittalaitteet ovat ajalta ennen USB-liitäntää, kytketään tulostimet laitteissa rinnakkaisväylään (kuva 2), tai vaihtoehtoisesti joissakin laitteissa myös sarjaväylään.

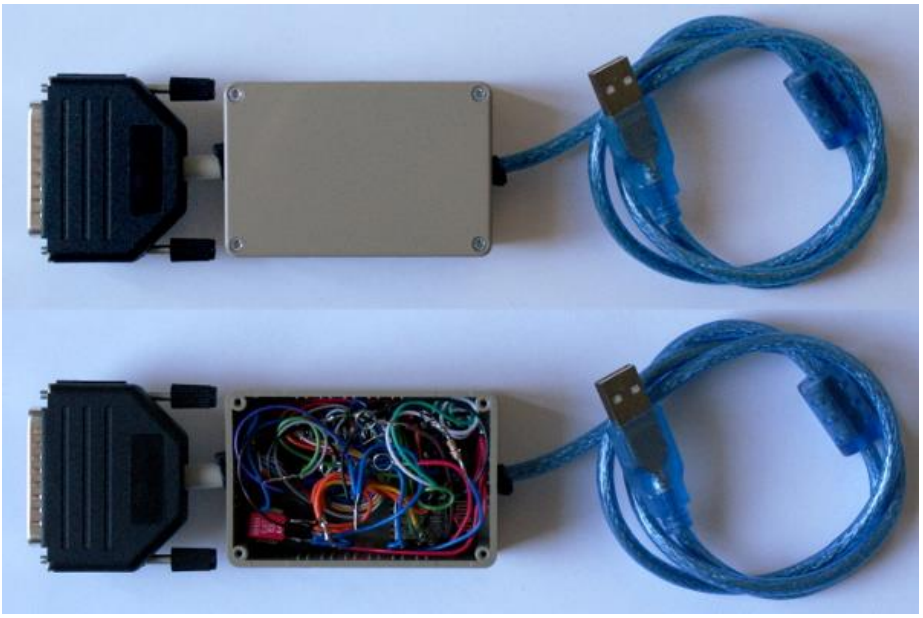
KUVA 2. Rinnakkaisväyläliitäntä mittalaitteessa.

3.2 Lukija

Lukija, eli lukijalaite, on järjestelmän toiminnallisuuden kannalta sen tärkeimpiä osasia. Kuten nimestä voi päätellä, lyhykäisyydessään kyseessä on laite, jonka tehtävä on kaapata normaalisti tulostimelle tarkoitettu data mittalaitteen rinnakkais- tai sarjaväylästä. Lukijalaite siis näyttäytyy mittalaitteelle tavallisena tulostimena, minkä vuoksi mittalaitteilta voidaan tulostaa kuten ennenkin.

Lukijalaite on langattoman verkon kautta yhteydessä palvelimeen, jolle se lähettää mittalaitteelta kaapatun tulostusdatan käsiteltäväksi ja näytettäväksi käyttöliittymässä. Käyttäjälle lukijalaite on melko näkymätön osa koko järjestelmää, käyttäjän tulee ainoastaan tietää lukijalaitteen tunnus/nimi. Koska lukijoita on järjestelmässä useita, tulee yksittäinen laite voida identifioida muiden joukosta. Siksi jokaisella lukijalaitteella on oma tunnuksensa. Tunnus syötetään käyttöliittymässä, jolloin käyttäjä aloittaa kyseisen lukijalaitteen (ja samalla mittalaitteen) ”monitoroinnin”, joka tarkoittaa halutun lukijan yhdistämistä käyttöliittymän näkymään. Toisin sanoen, tunnusten avulla palvelin tietää, mille käyttäjälle sen tulee miltäkin lukijalaitteelta vastaanottama data lähettää näytettäväksi.

Lukijalaite on siinäkin mielessä huomionarvoisessa asemassa, että sen suunnittelu ja toteutus oli tämän opinnäytetyöprojektin pisin ja työläin vaihe. Laitteen toteutuksessa tuli perehtyä vanhojen tiedonsiirtoväylien toimintaan, joka teetti ruohonjuuritason tutkimusta kehitystyössä. Kuvassa 3 nähdään valmis toteutettu lukijalaite. Musta liitin on mittalaitteeseen kytkettävä rinnakkaisväyläliitin, sinisen USB-kaapelin kautta laite saa käyttöjännitteensä. Lukijalaitteen kytkentäkaavio on nähtävillä liitteessä yksi.



KUVA 3. Toteutettu lukijalaite.

3.3 Langaton lähiverkko

Koska lukijalaitteita on järjestelmässä useita, WLAN-lähiverkko on käytännöllisin tapa em. laitteiden verkottamiseksi. Samaten langaton verkko luonnollisesti mahdollistaa laitteiden helpon liikuteltavuuden, mikäli tarve vaatii. Langatonta verkkoa käyttävän laitteiden rakentelu on viime vuosina muuttunut helpommaksi ja eritoten halvemmaksi, IoT-teknologian yleistyessä. Näistä syistä järjestelmä päätettiin toteuttaa langattomaan verkkoon perustuen.

Lukijalaitteille pystytettiin laboratorioon pelkästään niiden käyttöön tarkoitettu WLAN-lähiverkko, koska sopivia vaihtoehtoisia verkkoja ei ollut valmiina käytettäväksi. Toisaalta lukijalaitteiden liikenne on muutenkin hyvä (mm. turvallisuussyistä) pitää erillään muusta, yleisluontoisemmasta verkkoliikenteestä. Samasta syystä verkon SSID on piilotettuna, joten verkko on käyttäjille näkymätön. Langaton verkko toteutettiin Asuksen ”RT-N56U” -reitittimellä (kuva 4).

KUVA 4. Asus RT-N56U -reititin laboratoriossa.

3.4 Palvelin

Suurin osa järjestelmän funktionaalisuudesta keskittyy palvelimeen, jonka ensisijainen tehtävä on vastaanottaa lukijalaitteiden lähettämiä tulosteita ja muuntaa ne hyödyllisiin formaatteihin, pitäen samalla kirjaa järjestelmään kuuluvista lukijalaitteista ja niiden tilasta. Kyseessä on käyttäjän kannalta järjestelmän näkymättömin osa, palvelimella toteutettua käyttöliittymää lukuun ottamatta.

Suurin osa tämän projektin ohjelmointiosuudesta koostui palvelimen toteuttamisesta. Vaikka itse koodin kirjoittaminen sinänsä olikin suoraviivaista, vaati alkuun pääseminen jonkin verran tutkimustyötä, sillä toteutukseen valittiin käytettäväksi uusimpia nykypäivän ohjelmistoympäristöjä ja kirjastoja. Näin pyritään varmistamaan palvelimen ohjelmistojen ajantasaisuus myös pitkälle tulevaisuuteen.

3.4.1 Käyttöliittymä

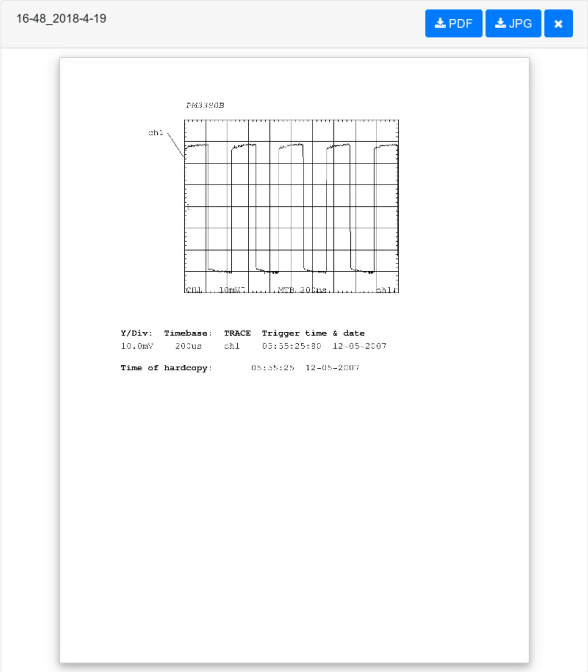
Käyttöliittymä on koko järjestelmäkokonaisuuden näkyvimpiä ja olennaisimpia osia käyttäjän kannalta. Käyttöliittymän päätehtävä on esittää mittalaitteilta saadut tulosteet käyttäjälle, samalla mahdollistaen niiden lataamisen käyttäjän omalle tietokoneelle, PDF

ja/tai JPG -tiedostoina. Se on toteutettu web-teknologioilla, eli kyseessä on selaimella käytettävä sovellus.

Järjestelmän käyttö vaatii käyttäjätunnuksen. Käyttöliittymässä onkin ensimmäisenä kirjautumisnäkymä, jonka kautta voi myös vaihtoehtoisesti rekisteröidä uuden tunnuksen, mikäli sellaista ei opiskelijalla vielä ole. Uuden tunnuksen luominen vaatii ”tamk.fi” -päätteisen sähköpostiosoitteen. Kirjauduttuaan sisään käyttäjän on mahdollista aloittaa haluamansa lukijalaitteen monitorointi.

Monitoroinnin aloittamiseksi käyttöliittymässä kysytään käyttäjältä lukijalaitteen tunnusta. Mikäli tunnus on oikea, eli laite on olemassa ja päällä, avautuu käyttäjälle monitorointinäkymä, jos laite on vapaana. Monitorointi varaa ko. laitteen käyttäjälle, joten jos toinen käyttäjä haluaa aloittaa saman laitteen monitoroinnin, tulee laitteen varanneen käyttäjän antaa tähän lupa. Tässä tilanteessa monitorointia yrittävä käyttäjä saa ilmoituksen, joka kertoo laitteen olevan varattuna, jolloin käyttäjän on jäätävä odottamaan laitteen vapautusta. Vastaavasti laitteen varannut käyttäjä saa ilmoituksen, jossa kerrotaan toisen käyttäjän haluavan monitoroida samaa laitetta. Käyttäjä voi hyväksyä tai hylätä pyynnön. Jos laitetta ei löydy tietokannasta tai laite on pois päältä, käyttäjä saa asianmukaisen virheilmoituksen. Käyttöliittymässä voidaan monitoroida useampaa laitetta yhtä aikaa; eri ”monitorit” ovat omissa välilehdissään.

Mittalaitteelta saadut tulosteet ilmestyvät käyttäjälle monitorinäkymään (kuva 5). Itse tulosteen lisäksi näkymässä on tulosteen nimi (päivämäärä) ja painikkeet tiedostojen lataamiseen, sekä tulosteen poistamiseen. Kun lukijalaitteelta saadaan uusi tuloste, korvataan vanha tuloste näkymässä automaattisesti uudella. Vanha tuloste jää kuitenkin talteen, ellei käyttäjä sitä erikseen poista.



KUVA 5. Kuvankaappaus tulosteesta monitorinäkymässä.

Käyttäjän on mahdollista tarkastella aiempia tulosteitaan historianäkymässä, jonka kautta on edelleen myös mahdollista yksittäisten tulosteiden lataaminen ja poistaminen, kuten monitorinäkymässäkin. Lisäksi historianäkymän kautta voidaan ladata kaikki tulosteet kerralla, yhteen zip-tiedostoon paketoituna. Järjestelmä tyhjentää käyttäjän tulostehistorian kuuden tunnin kuluttua siitä, kun käyttöliittymä on suljettu. Ajastettu poisto kuitenkin peruuntuu, mikäli käyttäjä tämän ajan sisällä kirjautuu sisään uudestaan.

4 TIEDONSIIRTOVÄYLÄT JA SIVUNKUVAUS

4.1 Rinnakkaisväylä

Tulostimissa vanhastaan käytetyn rinnakkaisväylän historia ulottuu 1970-luvun alkupuoliin asti, jolloin tulostinvalmistaja Centronics alkoi kehittää rinnakkaisen datansiirron mahdollistavaa liitäntää ja signaalikättelyä omille tulostimilleen. Yksinkertaisuutensa vuoksi useat tietokonevalmistajat ottivat tämän nk. ”Centronics-liitännän” käyttöön laitteissaan, ja 1980-luvun alkuun mennessä se oli saavuttanut jo lähes standardinomaisen aseman tietokoneiden ja tulostinten välisessä tiedonsiirrossa. (Durda F. 2004)

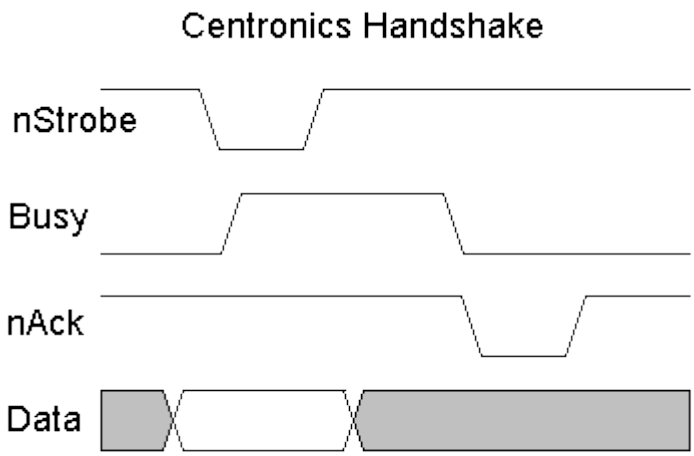
Myöhemmin IBM kehitti Centronics-väylästä oman, alkuperäisestä vain vähän muokatun version Personal Computer –kotitietokoneensa julkaisun myötä. Motiivina muutosten toteuttamiselle oli halu estää muiden kuin IBM:n omien tulostimien käyttö heidän omissa PC-tietokoneissaan. Tästä seurauksena markkinoilla oli hetken aikaa kaksi toistensa kanssa yhteensopimatonta rinnakkaisväylätoteutusta. Tulostinvalmistajat kuitenkin reagoivat tilanteeseen nopeasti lisäämällä tuen IBM:n väyläratkaisulle. Myöhemmin tämä ristiriita vaikutti väylässä käytettyihin liittimiin niin, että tulostimissa liittimenä pysyi alkuperäinen 36-pinninen ns. Centronics-liitin, kun taas tietokoneissa alettiin yleisesti nähdä IBM:n ratkaisun mukaista 25-pinnistä D-Sub liitintä. (Durda F. 2004)

Rinnakkaisväylässä perusajatuksena on nimensä mukaisesti siirtää dataa (bittejä) rinnakkain, tavu kerrallaan. Tämä tarkoittaa sitä, että väylässä on tavun kullekin kahdeksasta bitistä oma johtimensa, päinvastoin kuin sarjaväylässä, jossa dataa siirretään bitti kerrallaan yhdessä johtimessa. Myös perinteisestä sarjaväylästä poiketen, rinnakkaisväylässä käytetyt jännitetasot ovat TTL-logiikan mukaisia (0-5 volttia). Datansiirtoon vaaditaan em. kahdeksan datasignaalin lisäksi kolme erilaista kontrolli- ja tilasignaalia, joiden avulla toteutetaan datansiirtoa ohjaava Centronics-kättely. Lisäksi liitännässä on muutamia muita tilasignaaleita, joilla tulostin voi esimerkiksi signaloida vikatilanteesta tai paperin loppumisesta. Nämä muut signaalit eivät kuitenkaan ole tarpeellisia varsinaisessa datansiirrossa. (Peacock C. 2005)

Alkuperäinen Centronics-väylä tuki ainoastaan yksisuuntaista tiedonsiirtoa, maksiminopeudella 150 kt/s. Ajan saatossa väylästä kehitettiin monipuolisempia versioita, kuten EPP ja ECP. Nämä myöhemmät modifikaatiot tekivät mahdolliseksi mm. kahdensuuntaisen tiedonsiirron ja huomattavasti korkeammat tiedonsiirtonopeudet. Taaksepäin yhteensopivuus kuitenkin säilytettiin, ja vanha Centronics-tila alettiin myöhemmin tuntea yhteensopivuustilana (engl. ”compatibility mode”) tai SPP:nä (Standard Parallel Port). (Peacock C. 2005)

4.1.1 Centronics-kättely

Rinnakkaisväylän keskiössä on tiedonsiirtoa ohjaava signaalikättely, eli Centronics-kättely (engl. ”Centronics handshake”). Kättelyä käyttäen tietokone ja tulostin vaihtavat keskenään tietoa siitä, milloin väylässä on dataa luettavana, kun dataa luetaan ja kun data on vastaanotettu onnistuneesti. Kukin edellä mainituista käsittää oman signaalinsa liitännässä. (Peacock C. 2005)



KUVIO 1. Centronics-kättelyn ajoituskaavio. (Peacock C. 2005)

Kättely alkaa isäntälaitteen (tietokone, mittalaite) kirjoittaessa siirrettävän tavun bitit datalinjoihin (D0-D7). Seuraavana isäntälaite tarkistaa busy-signaalin olevan alatilassa varmistaakseen, että tulostin on kykenevä vastaanottamaan dataa. Lähettävä laite jää odottamaan, mikäli busy-signaali on ylätilassa. Kun tulostin on vastaanottavassa tilassa, isäntälaite signaloi luettavissa olevasta datasta negatiivisella pulssilla strobe-linjassa. (Peacock C. 2005)

Tulostin reagoi isäntälaitteen lähettämään strobe-pulssiin nostamalla busy-signaalinsa ylätilaan ja lukemalla tavun sisällön datasignaaleista. Tyypillisesti data luetaan strobe-pulssin nousevalla reunalla. Prosessoituaan vastaanotetun datan, tulostin palauttaa busy-signaalin takaisin alatilaan, sekä generoi negatiivisen pulssin ack-linjaan, jolla viestitään isäntälaitteelle, että data on vastaanotettu ja prosessoitu. Joissakin isäntälaitteiden toteutuksissa ack-signaali jätetään huomioimatta ajan säästämiseksi ja tällöin kättely tapahtuu pelkästään strobe ja busy -signaaleja käyttäen. Tulostimen palauduttua vastaanottavaan tilaan, siirrytään seuraavan tavun siirtoon, jolloin sykli alkaa alusta. Kättelyn toimintaa selventää ajoituskaavio yllä (kuvio 1). (Peacock C. 2005)

4.2 Sarjaväylä

Rinnakkaisväylän edeltäjänä tulostimissa käytettiin perinteistä sarjaväylää. Sarjaväylässä dataa siirretään bitti kerrallaan yhdessä johtimessa suuntaansa. Kaikkein yksinkertaisimmillaan (yksisuuntainen tiedonsiirto, ei vuonohjausta) sarjamuotoinen tiedonsiirto vaatii ainoastaan kaksi johdinta, signaali- ja maajohtimen. Tyypillisesti tiedonsiirto on kuitenkin kaksisuuntaista ja monesti tarvitaan myös vuonohjausta datansiirron hetkelliseen keskeyttämiseen, joten tavallisesti väylässä käytetään useampia johtimia. RS-232-standardin mukaisesti sarjaväylä käsittääkin tiedonsiirtosignaalien (Rx ja Tx) ohella yhteensä kuusi erilaista ohjaussignaalia. Vuonohjaussignaaleja lukuun ottamatta nämä muut signaalit ovat relevantteja lähinnä tietokoneiden ja modeemien välisessä kommunikoinnissa, joka on aikanaan ollutkin sarjaväylän pääkäyttökohde. (TAL Technologies 2018)

Tiedonsiirto väylässä voi tapahtua useilla erilaisilla konfiguraatioilla. Asetuksilla voidaan vaikuttaa tiedonsiirron nopeuteen, siirrettävien bittien määrään, lopetusbittien määrään, pariteettibittiin ja vuonohjaukseen. Jotta tiedonsiirto onnistuisi, on lähettäjällä ja vastaanottajalla oltava samat asetukset tiedonsiirrolle. Usein käytetty konfiguraatio on ”8N1”, eli kahdeksan databittiä, ei pariteettibittiä ja yksi lopetusbitti. (TAL Technologies 2018)

Alkuperäinen RS-232-standardi määrittelee väylässä käytettäväksi liittimeksi 25-pinnisen D-Sub liittimen. Käytännössä kuitenkin sarjaväylän perusliittimenä on jo pidempään ollut 9-pinninen D-Sub liitin. RS-232-standardin mukaisen sarjaväylän jännitetasot ovat -15 voltista +15 volttiin. Loogista ykköstilaa vastaa jännitearvot -5 voltin ja -15 voltin välillä ja looginen nollatila on vastaavasti jännitearvot 5 voltin ja 15 voltin välillä. Syynä

negatiivisten ja muutenkin näin suurien jännitteiden käytölle on parempi häiriönsietokyky ja pidemmät tiedonsiirtomatkat. Siispä vaikka itsessään sarjaväylä onkin perin yksinkertainen tiedonsiirtotekniikka, on sen käytössä silti omat hankaluutensa, jotka syntyvät tarpeista negatiivisille jännitteille ja jännitetasomuunnoksille. (Omega n.d.)

4.3 Sivunkuvauskielet

Isäntälaitteen tulostimelle siirtämä data sisältää informaation tulostettavan sivun (tai sivujen) sisällöstä ja asettelusta. Yksinkertaisimmillaan tulostimelle voidaan siirtää pelkkää tekstiä ASCII-merkkeinä, mutta tällöin ei tulosteelle luonnollisestikaan voida tehdä minkäänlaista asettelua/muotoilua (esim. tekstin koko, fontti tai sijainti), eikä sivu voi myöskään sisältää minkäänlaista grafiikkaa. Tähän tarpeeseen vastattiin aikanaan kehittämällä sivunkuvauskieliä. (FreeBSD n.d.)

Sivunkuvauskieliä on aikojen saatossa kehitetty lukuisia. Yleisesti käytettyjä ja tuettuja ovat pääasiassa kuitenkin vain HP:n PCL ja Adoben PostScript, sekä PDF. Näistä viimeksi mainittu on toki tunnetumpi tiedostoformaattina dokumenttien näyttämiseen tietokoneella kuin tulostimien sivunkuvauskielenä. Sekä PCL:n että PostScriptin historia ulottuu 80-luvun puolen välin tienoille. (Medeiros R. n.d.)

PostScript on PCL:ään verrattuna monin tavoin monipuolisempi. PostScriptillä on sivunkuvauskielen piirteet, mutta se on myös oma ohjelmointikielensä, sisältäen datatyypit, ehtolauseet ja silmukat jne. Monipuolisuuden kääntöpuolena on toki monimutkaisempi toteutus. Tästä syystä PCL sai jalansijaa etenkin edullisissa toimistotason tulostinlaitteissa, kun taas PostScript löysi tiensä julkaisutoimintaan. PCL on olennaisemmassa osassa tässä työssä, sillä lähes kaikki laboratorion mittalaitteista käyttää tulostuksen sivunkuvauskielenä juuri PCL:ää. (Medeiros R. n.d.)

5 LUKIJALAITTEEN TOTEUTUS

5.1 ESP8266

Lukijalaite rakentuu kiinalaisen Espressif Systemsin valmistaman ”ESP8266” WiFi-mikrokontrolleripiirin ympärille. ESP8266 on mikrokontrolleri, johon on sisällytetty WiFi-lähetin-vastaanotin ja koko TCP/IP-pakka. Halvan hintansa, pienen kokonsa ja vähäisen ulkopuolisten komponenttien tarpeen vuoksi siitä tullut erityisen suosittu IoT-alusta, etenkin maker-piirien keskuudessa. (Espressif Systems 2018)

Useimmat ESP8266-pohjaiset moduulit ovat kolmannen osapuolen valmistamia, joskin Espressifillä on omiakin moduuleja. Tunnetuin ESP-moduulien valmistaja lienee myöskin kiinalainen Ai-Thinker. Lukijalaitteessa käytetty moduuli on malliltaan Ai-Thinkerin ESP-12E (kuva 6).

KUVA 6. ESP-12E moduuli.

5.1.1 Spesifikaatiot

ESP8266:n pohjana on Tensilican 32-bittinen ”L106” RISC-prosessori, jonka kellotaajuus on ohjelmallisesti asetettavissa maksimissaan 160 megahertsiin. Laitteessa on käytettävissä noin 50 kilobittiä muistia (SRAM). Flash-muistin määrä sen sijaan vaihtelee moduuleittain, sillä 8266-mallissa ei itsessään ole flash-muistia, vaan se sijaitsee omalla erillisellä piirillään. Maksimissaan flash-muistia voi kuitenkin olla 16 megatavua. (Espressif Systems 2018)

WiFi-lähetin-vastaanotin toimii 2,4 gigahertsin taajuudella ja tukee 802.11 b, g ja n protokollia. Radio toimii koko taajuusalueella, siis kaikilla neljällätoista kanavalla. Wi-Fi voi toimia BSS, Ad-Hoc tai SoftAP -tilassa, mahdollistaen useampia erilaisia käyttökohteita. Laite tukee WPA/WPA2 salausprotokollia, WEP, TKIP ja AES salausalgoritmejä käyttäen. (Espressif Systems 2018)

Laite toimii 3,3 voltin käyttöjännitteellä. GPIO-pinnejä on yhteensä 17 kappaletta, joista osalla on erityisrooli UART, SDIO, SPI, I^2C tai I^2S -väylinä. Pinneistä neljällä voidaan tuottaa PWM-signaaleja. Lisäksi piirissä on yksi 10-bittinen AD-muunnin. (Espressif Systems 2018) Selvennyksen vuoksi olennaisimmat spesifikaatiot ovat listattuna seuraavassa taulukossa (taulukko 1).

TAULUKKO 1. ESP8266 spesifikaatiot (Espressif Systems 2018)

Prosessori	Tensilica L106 32-bit
Kellotaajuus	160 MHz
Käyttöjännite	3,3 V
Taajuusalue	2,4 GHz (2400 MHz - 2483.5 MHz)
WiFi-protokollat	802.11 b/g/n
Salaus	WPA / WPA2 (WEP / TKIP / AES)
GPIO	17 kpl
ADC	1 kpl (10 bit)
PWM-lähdöt	4 kpl
Oheislaiteväylät	UART / SDIO / SPI / I^2C / I^2S / IR

5.1.2 GPIO

Yleiskäyttöisiä digitaalisia IO-pinnejä (GPIO, General Purpose Input Output) piirissä on edellä mainitut 17 kappaletta. Vapaaseen käyttöön näistä lopulta jää kuitenkin paljon pienempi määrä, sillä useilla pinneistä on jonkinlaisia niiden käyttöä rajoittavia erityispiirteitä. Esimerkiksi pinnit 6-11 ovat lähes poikkeuksetta varattuja erillisen flash-muistipiirin käyttöön, jättäen jäljelle 11 käytettävää pinniä. Kaikki GPIO-pinnit niiden erityisfunktioineen ovat listattuna oheisessa taulukossa (kuvio 2). (ESP Community Wiki 2015)

GPIO	Inst Name	Function 0	Function 1	Function 2	Function 3	Function 4	At Reset	After Reset	Sleep
0	GPIO0_U	GPIO0	SPICS2			CLK_OUT	oe=0, wpu	wpu	oe=0
1	U0TXD_U	U0TXD	SPICS1		GPIO1	CLK_RTC	oe=0, wpu	wpu	oe=0
2	GPIO2_U	GPIO2	I2SO_WS	U1TXD		U0TXD	oe=0, wpu	wpu	oe=0
3	U0RXD_U	U0RXD	I2SO_DATA		GPIO3	CLK_XTAL	oe=0, wpu	wpu	oe=0
4	GPIO4_U	GPIO4	CLK_XTAL				oe=0		oe=0
5	GPIO5_U	GPIO5	CLK_RTC				oe=0		oe=0
6	SD_CLK_U	SD_CLK	SPICLK		GPIO6	U1CTS	oe=0		oe=0
7	SD_DATA0_U	SD_DATA0	SPIQ		GPIO7	U1TXD	oe=0		oe=0
8	SD_DATA1_U	SD_DATA1	SPID		GPIO8	U1RXD	oe=0		oe=0
9	SD_DATA2_U	SD_DATA2	SPIHD		GPIO9	HSPIHD	oe=0		oe=0
10	SD_DATA3_U	SD_DATA3	SPIWP		GPIO10	HSPIWP	oe=0		oe=0
11	SD_CMD_U	SD_CMD	SPICS0		GPIO11	U1RTS	oe=0		oe=0
12	MTDI_U	MTDI	I2SI_DATA	HSPIQ MISO	GPIO12	U0DTR	oe=0, wpu	wpu	oe=0
13	MTCK_U	MTCK	I2SI_BCK	HSPID MOSI	GPIO13	U0CTS	oe=0, wpu	wpu	oe=0
14	MTMS_U	MTMS	I2SI_WS	HSPICLK	GPIO14	U0DSR	oe=0, wpu	wpu	oe=0
15	MTDO_U	MTDO	I2SO_BCK	HSPICS	GPIO15	U0RTS	oe=0, wpu	wpu	oe=0
16	XPD_DCDC	XPD_DCDC	RTC_GPIO0	EXT_WAKEUP	DEEPSLEEP	BT_XTAL_EN	oe=1,wpd	oe=1,wpd	oe=1

KUVIO 2. ESP8266:n GPIO-pinnit. (ESP Community Wiki. 2015.)

Pinneillä 0, 2 ja 15 on erityinen rooli laitteen käynnistyksessä. Niiden tila käynnistyshetkellä määrittää millä tavalla laite käynnistyy, ”boottaa”. Asettamalla pinnit ylös- ja alasvetovastuksilla ylä- tai alatilaan, voidaan valita, käynnistyykö laite (ohjelma) flash-muistilta, muistikortilta tai sarjaväylältä (UART). Käynnistys flash-muistilta tarkoittaa toisin sanoen ”normaalia” käynnistystä, sarjaväylävaihtoehto on tarkoitettu laitteen ohjelmointiin. Laitteen käynnistyttyä kyseiset pinnit ovat jälleen vapaasti käytettävissä, mutta vaadittujen ylös- ja alasvetovastusten vuoksi nämä pinnit soveltuvat kuitenkin vain lähdöiksi (output-tila). Käynnistysasetusten eri kombinaatiot ovat selvennyttynä seuraavassa taulukossa (taulukko 2). Taulukossa merkintä ”X” tarkoittaa kelluvaa tilaa (engl. ”floating"). (ESP Community Wiki 2015)

TAULUKKO 2. ESP8266 käynnistysasetukset.

Tila	**GPIO0**	**GPIO2**	**GPIO15**
UART	0	X tai 1	0
SDIO	X	X	1
Flash	1	X tai 1	0

Pinnit 1 ja 3 ovat sarjaväylän käytössä, joten usein varsinkaan kehitysvaiheessa ne eivät ole käytettävissä, mikäli ohjelmakoodin halutaan kirjoittavan lokitietoja terminaaliin. Lisäksi pinnin 1 kytkeminen maapotentiaaliin estää laitteen käynnistymisen. Pinneissä on sisäiset ylösvetovastukset, ja ne voidaan asettaa generoimaan keskeytyksiä. Poikkeuksena on pinni 16, joka sijaitsee piirillä täysin erillään muista IO-pinneistä, piirin RTC-yksikössä. Se ei tue keskeytyksiä, ja ylösvetovastuksen tilalla on alasvetovastus. (ESP Community Wiki 2015)

Näiden kaikkien mainittujen pinnien erityispiirteet huomioiden, voidaan todeta, että lopulta täysin vapaaseen käyttöön jääviä pinnejä on ainoastaan viisi kappaletta. Tämä luo omat haasteensa kehitystyölle, etenkin tämän kaltaisessa työssä, jossa lukumääräinen tarve IO:lle on suuri (esim. rinnakkaisväylän kahdeksan datasignaalia). Kuitenkin hyödyntämällä mm. multiplekseriä, onnistui rinnakkaisväylän lukeminen niukemmallakin määrällä IO-pinnejä.

5.1.3 Arduino-ohjelmointiympäristö

ESP8266-mikrokontrollerin ohjelmointiin on kehitetty useita sovelluskehitysympäristöjä (SDK). Yksi tunnetuimmista on alun perin lähinnä Atmelin AVR-mikrokontrollereille kehitetty Arduino-kehitysympäristö. Arduino-alustan suosion ja tunnettavuuden vuoksi se käännettiin kolmansien osapuolten toimesta myös ESP8266:lle. (Grokhotkov I. 2017b) Nykyisellään myös Arduinon omassa virallisessa IDE-ohjelmistossa on tuki myös muille kuin varsinaisille AVR-pohjaisille Arduino-mikrokontrollereille (Arduino 2015). Arduino SDK, virallisesti ”ESP8266 Arduino Core”, valittiin tähän toteutukseen juuri suosionsa ja aiempien käyttökokemusten perusteella.

Arduino-alusta tuo ESP8266:n ohjelmointiin digitalWrite ja digitalRead -tyyppiset, tutut ja yksinkertaiset funktiot. Yleisesti ottaen se mahdollistaakin ESP8266:n ohjelmoinnin kuten minkä tahansa muunkin Arduino-perheen mikrokontrollerin. Koska kirjaston tekninen toteutus on täysin irrallaan alkuperäisestä, on sen käytössä kuitenkin joitakin huomionarvoisia eroavaisuuksia. Laite esimerkiksi hoitaa kaikki WiFiin ja TCP/IP-protokolliin liittyvät tehtävät pääsilmukan (”loop()” -funktio) kierrosten välissä. (Grokhotkov I. 2017d) Tämä vuorostaan tarkoittaa sitä, että pääsilmukan suoritus (kierto) ei saa estyä liian pitkäksi aikaa. Mikäli näin käy, laite käynnistää itsensä uudelleen watchdogin toimesta. (Grokhotkov I. 2017c)

5.2 Kättelyn elektroninen toteutus

Suunnitteluvaiheessa ajatuksena oli Centronics-kättelyn toteutus täysin ohjelmallisesti, keskeytyksiä hyödyntäen. Idea tämän kaltaiseen ratkaisuun syntyi Arduino-keskustelu-

foorumille lähetetyn koodiesimerkin pohjalta. Kyseisessä toimintamallissa strobe-signaalin laskeva reuna aiheuttaa keskeytyksen, jolloin keskeytyspalvelussa hoidetaan datan lukeminen, sekä busy ja ack -signaalien toiminta. (Arduino Forum 2012)

Käytännön testeissä tämän kaltainen toteutus ei kuitenkaan osoittautunut toimivaksi. Vianetsinnän perusteella voitiin todeta ESP:n keskeytysten sisältävän liikaa latenssia. Piirin reagoinnissa strobe-pulssiin oli liikaa viivettä. Oheislaitteen on kyettävä reagoimaan strobe-pulssiin nostamalla busy-signaali ylätilaan maksimissaan mikrosekunnin sisällä strobe-signaalin laskevasta reunasta (Theiling H. 2013). Testeissä viivettä näiden kahden signaalin välille syntyi kuitenkin reilusti yli kaksi mikrosekuntia, mikä osoittautui ongelman syyksi. Syvällisemmälläkään tutkimuksella ei latenssin pienentämiseen löytynyt juurikaan keinoja, joten ainoaksi vaihtoehdoksi jäi kättelyn toteuttaminen ”raudalla”, eli elektronisella kytkennällä.

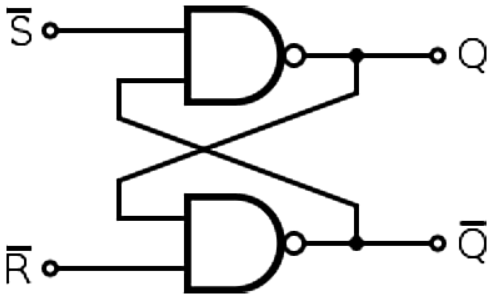
5.2.1 SR-salpa

Kättelyn toteutuksessa busy-signaali luodaan käyttäen SR-salpaa. SR-salpa on yksinkertainen sekvenssipiiri, jolla on set ja reset -tulot. Kun salvan set-tulo asetetaan ykköseksi, asettuu myös salvan lähtö ykköseksi. Lähdön tila säilyy ykkösenä niin kauan, kunnes reset-tulo asetetaan ykköstilaan, jolloin salvan lähtö nollautuu. Tulee kuitenkin varmistaa, ettei molempia salvan tuloista aseteta ylätilaan samaan aikaan, sillä tämä on ns. kielletty tila. Salpa toimii tässä tilanteessa virheellisesti, josta seurauksena on, ettei lähdön tilaa voida määrittää. SR-salvan toimintaa selventää oheinen totuustaulu (taulukko 3). (Plese L. 2016)

TAULUKKO 3. SR-salvan totuustaulu.

S	**R**	**Q**	**Q’**	**Tila**
0	0	Q	Q’	Ei muutu
1	0	1	0	Asetettu
0	1	0	1	Nollattu
1	1	0	0	Kielletty tila

SR-salpa voidaan toteuttaa joko NOR tai NAND logiikkaporteilla. Edeltävä totuustaulu (taulukko 3) itseasiassa kuvaakin NOR-tyyppisen salvan toimintaa. Erona NAND-porteilla toteutetussa salvassa on tulojen looginen inversio (tulossa alatila tulkitaan loogiseksi ykköseksi). NAND-porteilla toteutetun SR-salvan kytkentää kuvaa oheinen logiikkadiagrammi (kuvio 3). (Plese L. 2016)



KUVIO 3. NAND-tyyppisen SR-salvan logiikkadiagrammi. (Plese L. 2016)

Koska rinnakkaisväylässä strobe ja ack -signaalit ovat invertoituja, tarvitaan kättelyn toteuttamiseen juurikin NAND-tyyppinen salpa. Kytkennässä strobe-signaali tuodaan salvan set-tuloon ja ack-signaali reset-tuloon. Salvan ei-invertoitu lähtö ”Q” on haluttu busy-signaali.

Negatiivinen strobe-pulssi salvan set-tulossa ajaa salvan lähdön, eli tässä tilanteessa busy-signaalin ylätilaan. Koska tämä tapahtuu täysin elektronisesti, on pulssiin reagoiminen lähes välitöntä. Keskeytysten sijaan ohjelmakoodissa testataan (eli ”pollataan”) jatkuvasti busy-signaalin tilaa. Kun busy-signaalin havaitaan olevan ylätilassa, voidaan päätellä isäntälaitteen lähettäneen strobe-pulssin ja näin ollen jatkaa datan lukemiseen. Kun tavun bitit on saatu luettua onnistuneesti, signaloidaan tästä isäntälaitteelle generoimalla ack-linjaan negatiivinen pulssi. Koska ack-signaali on kytketty myös salvan reset-tuloon, aiheuttaa em. pulssi myös salvan lähdön vapautumisen alatilaan.

Salpakytkentä toteutettiin käyttäen Motorolan valmistamaa ”SN74LS00N” -mallista IC-piiriä. Kyseinen piiri valittiin, sillä niitä oli helposti saatavilla koulun omista valikoimista. Tämä piiri sisältää neljä kappaletta NAND-portteja (Texas Instruments 1983). SR-salpa saadaan rakennettua kytkemällä kaksi piirin porteista kuvion 3 osoittamalla tavalla.

5.3 Multiplekseri

ESP8266:n käytettävissä olevien GPIO-pinnien vähäisen määrän vuoksi kaikkia rinnakkaisväylän signaaleja ei voitu kytkeä mikrokontrolleriin suoraan. Ratkaisuna oli kytkeä väylän datasignaalit mikrokontrolleriin multiplekserin avulla. Multiplekseriä hyödyntämällä saatiin datasignaalien lukemiseen tarvittavien IO-pinnien määrä puolitettua, siis kahdeksasta pinnistä neljään pinniin.

Multiplekseri on elektroninen laite (piiri) joka koostuu kanavista, valitsintuloista ja lähdöstä. Valitsintulojen biteillä valitaan, minkä kanavan multiplekseri kytkee lähtöönsä. Esimerkiksi nelikanavaisessa multiplekserissä valitsintuloja on kaksi kappaletta, sillä neljän arvon esittämiseen tarvitaan kaksi bittiä. Kun halutaan valita ensimmäinen (nollas) kanava, molemmat valitsintulot asetetaan nolliksi. Vastaavasti viimeinen (kolmas) kanava valitaan asettamalla valitsintulojen bitit ykkösiksi (koska binaarissa 11 = 3). (Agarwal T. n.d.) Tähän toteutukseen piiriksi valikoitui Texas Instrumentsin kahdeksankanavainen multiplekseri, malliltaan ”74HC4051”.

5.4 Puskurirekisteri

Yleisen käytännön mukaan oheislaitteen tulee lukea data väylästä strobe-pulssin nousevalla reunalla. Riippuen isäntälaitteen toteutuksesta, validi data saattaakin hävitä datalinjoista nopeasti pulssin nousureunan jälkeen. (Theiling H. 2013) Tämä luo datan lukemiselle potentiaalisesti tiukat aikakriteerit. Datan lukeminen multiplekseriä käyttäen vie luonnollisesti enemmän aikaa kuin GPIO-pinnien suora lukeminen, joten lisäämällä kytkentään ennen multiplekseriä puskurirekisteri, voidaan varmistaa datan pysyminen tallessa koko lukemisoperaation ajan.

Puskurirekisteri on piiri, joka koostuu useammasta D-kiikusta. D-kiikku on digitaalitekniikan yksinkertaisimpia kiikkupiirejä. Sillä on datatulo ”D” ja kellotulo, sekä lähtö ”Q”. D-kiikku kopio datatulonsa arvon lähtöönsä kellotulon nousevalla (tai laskevalla/molemmilla, riippuen mallista) reunalla. Tällä tavoin datatulossa kellon nousureunan hetkellä ollut bitin arvo säilyy kiikun lähdössä riippumatta siitä, mitä datatulossa kellopulssin jälkeen tapahtuu. Täten bitin arvon lukemiseen jää enemmän aikaa. (Texas Instruments 1975)

Koska D-kiikkuja tarvitaan jokaiselle väylän datasignaaleista, puhutaankin tällöin kahdeksanbittisestä puskurirekisteristä. Tällainen piiri sisältää kahdeksan kappaletta D-kiikkuja, jotka jakavat yhteisen kellosignaalin. Kellosignaaliksi kytketään tässä tapauksessa väylän strobe-signaali. Täten jokainen rekisterin kiikuista kopioi tulon tilan lähtöönsä strobe-pulssin nousevalla reunalla. Puskurireksiteriksi valittiin Motorolan valmistama ”SN74LS374N” -mallinen IC-piiri.

5.5 Sarjaväyläadapteri

Tietoliikennelaboratorion mittalaitteista määrällisesti merkittävän osan muodostaa Fluken valmistamat oskilloskoopit. Näitä oskilloskooppeja on muutamaa eri mallia, mutta kaikkia niitä yhdistää tulostus sarjaväylän kautta, sillä näissä laitteissa ei ole rinnakkaisväyläliitäntää lainkaan. Koska tällaisia laitteita on laboratoriossa useita, olisi epätarkoituksenmukaista jättää nämä laitteet tulostusjärjestelmän ulkopuolelle.

Yksi mahdollisuus olisi luoda lukijalaitteesta pelkästään sarjaväylään tarkoitettu versio. Ratkaisua ei kuitenkaan voida pitää kovin ihanteellisena, sillä kaksi erilaista laitetoteutusta tekisi järjestelmäkokonaisuudesta turhaan monimutkaisemman. Suoraviivaisempaa olikin toteuttaa adapteri, jota käyttämällä ensisijaisesti rinnakkaisväylään tarkoitettu lukijalaite voidaan kytkeä myös sarjaväylään.

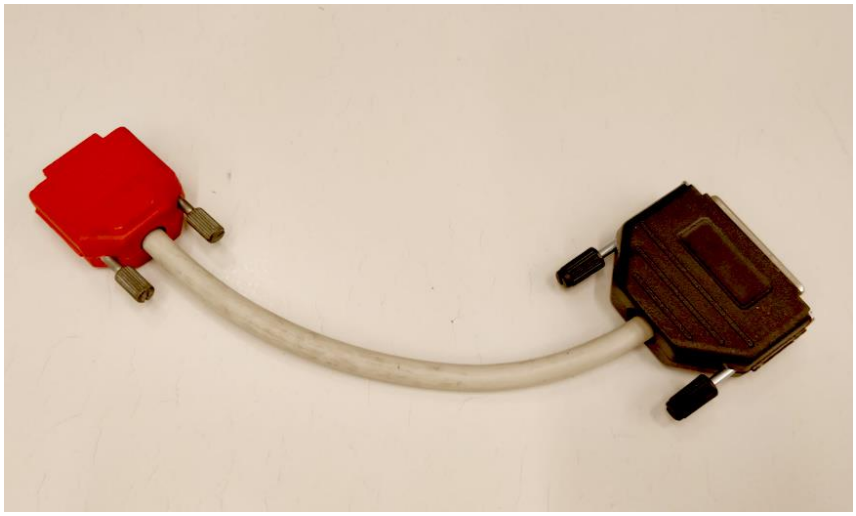
Sarjaväylän yksinkertaisuuden johdosta, pääosa adapterin toteutuksesta hoidetaan täysin ohjelmakoodissa. Adapterin fyysinen toteutus käsittää ainoastaan jännitetasomuunnoksen, sekä kytkennän, jonka perusteella lukijalaite osaa tunnistaa juuri adapterin kytketyksi. Tunnistaminen on toteutettu siten, että adapterissa on kytketty väylän datasignaalit siten, että niistä muodostuu tietty tavu, jota lukijalaite tietää väylästä etsiä.

5.5.1 Tasomuutos

Jotta mikrokontrolleriin voitaisiin kytkeä RS-232 sarjaväylästandardia käyttävä oheislaite, tulee sarjaväylän datasignaaleille tehdä jännitetasomuunnos, sekä looginen inversio (NOT-operaatio). Tasomuutoksessa ylätilajännite (5 - 15 V) alennetaan viiteen volttiin,

alatilajännite (-12 - -5 V) nostetaan maatasoon, siis nollaan volttiin. Looginen NOT-operaatio tarvitaan, koska RS-232 standardissa loogista ykköstilaa vastaa alatila, nollatilaa ylätila, siis päinvastoin kuin TTL-logiikassa. (Omega n.d.)

Tasomuunnos toteutettiin käyttäen Motorolan valmistamaa ”MC1489” IC-piiriä. Kyseessä on RS-232-sarjaväylän vastaanottopuolen tasomuunnokseen tarkoitettu piiri. (ON Semiconductor 2009) Koska tässä tapauksessa liikenne on yksisuuntaista (mittalaitteelta mikrokontrollerille), ei tasomuunnosta vastakkaiseen suuntaan tarvita lainkaan. Piiri kytkentöineen sisällytettiin rinnakkaisväyläliittimen koteloon (kuva 7). Adapterin kytkentäkaavio on nähtävillä liitteessä kaksi.



KUVA 7. Toteutettu sarjaväyläadapteri.

5.6 Regulaattorikytkentä

Lukijalaite suunniteltiin toimimaan viiden voltin jännitteellä. Viiden voltin jännitelähteistä on tullut varsin yleisiä USB-laturien myötä, joten viisi volttia oli luonteva vaihtoehto. ESP8266:n käyttöjännite on kuitenkin 3,3 volttia, joten syöttöjännite tulee alentaa sopivaksi (Espressif Systems 2018). Tämä on toteutettu käyttäen lineaarista jänniteregulaattoria.

Regulaattoriksi valittiin Texas Instrumentsin ”LM3940IT”. Kyseessä on ns. ”low dropout” -regulaattori, joka tarkoittaa, että regulaattori pystyy toimimaan silloinkin, kun syöttöjännite on hyvin lähellä lähtöjännitettä. Tämän regulaattorin tapauksessa alhaisin

jännite, jolla regulointi vielä onnistuu, on 4,5 volttia. Regulaattorin lähtöpuolelle tulee lisäksi vielä kytkeä kondensaattori (33 µF) tasaamaan mahdollisia virtapiikkejä. (Texas Instruments 1999)

5.7 Ohjelmallinen toteutus

Lukijalaitteen ohjelmakoodin päätehtävä on datan lukeminen rinnakkais- tai sarjaväylästä ja sen lähettäminen eteenpäin serverille HTTP-protokollalla. Lisäksi laite lähettää palvelimelle käynnistyessään ja sen jälkeen tasaisin väliajoin (5 min) päivitysviestin, jonka perusteella palvelin tietää kyseisen lukijalaitteen olevan edelleen päällä. Viesti lähetetään myös silloin, kun kytketty mittalaite vaihtaa tilaa (päällä/pois päältä, tai ei kytketty), tai kun lukijaan kytketään sarjaväyläadapteri. Päivitysviesti sisältää seuraavat tiedot laitteesta:

- Nimi
- Laitteen luomisen (ohjelmoinnin) ajankohta
- Sijainti
- Kytketyn mittalaitteen tila

Osaa näistä tiedoista tarvitaan vain siinä tilanteessa, kun laite on järjestelmälle uusi ja sen tiedot halutaan lisätä palvelimen tietokantaan. Luomisajankohta ja sijaintitieto ovat hyödyllistä lisäinformaatiota listattaessa kaikki järjestelmän laitteet. Sekä tulosteet, että päivitysviestit lähetetään palvelimelle käyttäen HTTP:n POST-metodia.

Aluksi datan lukeminen toteutettiin siten, että kaikki väylästä luettu data puskuroitiin ensin flash-muistille, ja vasta sen jälkeen lähetettiin palvelimelle yhdessä erässä. Tästä oli se etu, että HTTP-otsikkotietoihin vaadittu ”Content-length” -kenttä, eli lähetettävän datan määrä tavuissa, voitiin laskea helposti, koska kaikki luettu data oltiin puskuroitu talteen ennen lähettämistä. (Fielding, et al. 1999)

Datan puskuroiminen flash-muistille osoittautui kuitenkin varsin hitaaksi, samalla hidastaen koko lukuoperaatiota merkittävästi. Lisäksi jatkuva flash-muistille kirjoittaminen lyhentää muistipiirin elinikää. Näistä syistä ohjelman toimintaa muutettiin siten, että flash-

muistin sijasta dataa puskuroidaankin RAM-muistille kuudentoista kilobitin erissä kerrallaan ja lähetetään serverille osissa. Tästä kuitenkin seuraa ettei ”Content-length”-tietuetta voida käyttää, koska data lähetetään useassa osassa ja näin ollen datan kokonaismäärää ei voida tietää lähetyksen alussa. Ratkaisu oli implementoida nk. ”chunked” -siirtokoodaus (engl. ”chunked transfer encoding”), joka mahdollistaa POST-datan lähettämisen osissa, ilman että lähetettävän datan määrää tarvitsee tietää etukäteen. (Fielding, et al. 1999) Kokonaisuudessaan lukijalaitteen ohjelmakoodin toimintaa kuvaa vuokaavio liitteessä kolme.

5.7.1 Wifi-kirjastot

ESP8266-mikrokontrolleria ohjelmoitaessa olennaisimpia kirjastoja ovat Arduino-ohjelmointiympäristöön sisältyvät WiFi-kirjastot. Tyypillisin käyttö, kuten tässäkin tapauksessa, on olemassa olevaan verkkoon yhdistäminen. Kirjastot mahdollistavat myös monenlaista muuta toiminnallisuutta, kuten esimerkiksi saatavilla olevien verkkojen etsiminen ja oman yhteyspisteen luomisen. (Grokhotkov I. 2017b)

Serveriin lukijalaite on yhteydessä käyttämällä WiFi-kirjaston WifiClient-luokan metodeja. WiFiClient mahdollistaa TCP-yhteyden muodostamisen haluttuun osoitteeseen ja porttiin. Luotu yhteys on käytettävissä Stream-tyypin oliona, joka tuntee mm. print() ja write() -metodit (vrt. ”Serial.print()”). POST-viestit koostetaankin staattisesti käyttäen em. metodeja. (Grokhotkov I. 2017a)

5.7.2 Setup() –alustusfunktio

Alustusfunktiossa ensimmäisenä asetetaan käytettävien IO-pinnien tilat halutuiksi (tulo/lähtö, alasvetovastusten käyttö), jonka jälkeen tarkastetaan, onko kontrolleriin mahdollisesti kytketty sarjaväyläadapteri. Mikäli adapteri on kytkettynä, alustetaan sarjaväylä duplex-tilaan, tällöin Rx-pinniä (GPIO3) käytetään datan vastaanottamiseen mittalaitteelta ja Tx-pinniä (GPIO1) voidaan edelleen käyttää lokitietojen välittämiseen konsolille. Mikäli adapteri ei ole kytketty, alustetaan sarjaväylä simplex-tilaan. Tällöin käytössä on vain Tx-pinni, sillä Rx-pinniä käytetään rinnakkaisväylän ack-signaaliin.

Seuraavana funktiossa hoidetaan asetusten lukeminen flash-muistilta. Laitteen vaatimat konfiguraatiot ovat tallennettuna ”initconf” -tiedostoon haihtumattomalla muistilla. Tiedosto sisältää päivitysviestin sisältämän tietojen lisäksi seuraavat asetukset:

- Langattoman verkon SSID
- Langattoman verkon salasana
- Palvelimen verkko-osoite
- Palvelimen portti

Kun konfiguraatiot on luettu, jatketaan langattoman yhteyden muodostamiseen. Kun yhteys on saatu muodostettua onnistuneesti, lähetetään ensimmäinen päivitysviesti, jonka jälkeen alustusfunktio päättyy ja ohjelman suoritus siirtyy pääsilmukkaan.

5.7.3 Loop() –pääsilmukka

Pääsilmukan ensisijainen tehtävä on reagoida mittalaitteelta saapuvaan dataan. Kuten todettu, rinnakkaisväylän tapauksessa tämä on toteutettu lukemalla salvan lähdön, eli busysignaalin tilaa. Mikäli sarjaväyläadapteri on kytkettynä, voidaan käyttää tavallisia Serial-luokan metodeja (mm. ”available()”).

Lukemisoperaatio on toteutettu sikäli hyvin yksinkertaisella tavalla, että ohjelma ei tulkitse luetun datan sisältöä millään tavalla. Tästä johtuen aikakatkaisu on ainoa tapa, jolla lukijalaite voi tietää mittalaitteen lopettaneen tiedonsiirron. Mikäli mittalaite ei signaloi uudesta tavusta 300 millisekunnin aikana (sarjaväylällä 1 s), lukijalaite päättelee tiedonsiirron loppuneen, päättäen samalla yhteyden palvelimelle.

Kaikki luettu data saatetaan myös hylätä kokonaan, mikäli luetun datan määrä jää alle tietyn rajan (512 tavua). Tällä on tarkoitus estää virheellisen datan lähettäminen serverille. Lukijalaite saattaa tulkita mittalaitteen aloittaneen tiedonsiirron, vaikka oikeasti kyseessä olisikin jonkinlainen häiriö rinnakkaisväylässä, mikä on saanut salvan vaihtamaan tilaansa. Häiriö saattaa syntyä esimerkiksi kytkettäessä lukijalaite mittalaitteeseen.

Kun datansiirto ei ole käynnissä, silmukassa testataan kolmen sekunnin välein kytketyn laitteen tilaa. Mikäli laitteen tila vaihtuu, esimerkiksi mittalaite kytketään pois päältä, lähetetään serverille päivitysviesti. Samoin toimitaan, mikäli lukijaan kytketään sarjaväyläadapteri. Lukijaan siis voidaankin kytkeä laitteita ”lennosta”, ilman tarvetta uudelleenkäynnistykselle.

Viimeisenä silmukassa hoidetaan päivitysviestin lähettäminen palvelimelle viiden minuutin välein. Palvelin päivittää tietokantaansa viimeisimmän päivitysviestin ajankohdan ja mikäli viimeisimmästä viestistä on kulunut reilusti yli viisi minuuttia, tulkitaan laitteen olevan pois päältä.

6 PALVELINOHJELMOINTI

6.1 Virtuaaliserveri

Työn palvelinosuuden alustana käytettiin koulun tarjoamaa virtuaaliserveriä. Tietoliikenteen koulutusohjelma tarjoaa servereitä oppilaillensa mm. kurssi- ja projektikäyttöön. Serverit kuuluvat tietotekniikan osaston hallinnoimaan nk. ”titeverkkoon”, joka on osaston tarpeisiin pystytetty intranet-tyyppinen verkko. Virtuaalikoneisiin voi muodostaa SSH-yhteyden ainoastaan em. verkosta. Verkkoon pääsee mm. VPN-yhteydellä. (Parkkila E. n.d.)

Virtuaalikoneita ajetaan Citrixin XenServer -virtualisointialustalla (engl. ”hypervisor”). Tämän virtuaaliserverin käyttöjärjestelmäksi valittiin ja asennettiin Debian GNU/Linuxin versio 9.4 (Stretch). Virtuaalikoneen ja julkisen verkon välissä on lisäksi käänteinen välityspalvelin (”nGinx”-palvelinohjelmisto). (Parkkila E. n.d.)

6.2 Taustaosa

Taustaosa, eli ”back-end”, sisältää kaiken sen toiminnallisuuden, joka yhdistää lukijalaitteen lähettämän datan selainpohjaiseen käyttöliittymään. Taustaosa toteuttaa rajapinnan, jota sekä käyttöliittymä, että lukijalaite käyttävät. Rajapinta toteutettiin pääosin REST-arkkitehtuurin mukaisesti.

Lukijalaite on yhteydessä rajapintaan lähettäessään tulosteen tai päivitysviestin. Vastaanottaessaan tulosteen, palvelin muuntaa mittalaitteelta luetun datan pdf- ja kuvatiedostoksi. Tulosteen tiedot lisätään tietokantaan ja samalla uudesta tulosteesta välitetään tieto myös käyttöliittymälle. Vastaavasti lukijan tiedot päivitetään tietokantaan, kun palvelin vastaanottaa päivitysviestin laitteelta. Tietokantaan tallennettuja laitteen tietoja tarvitaan käyttäjän pyrkiessä aloittamaan halutun laitteen monitoroinnin.

Käyttöliittymän toiminnallisuus nojaa täysin taustaosan rajapinnan käyttämiseen, sillä käyttöliittymän HTML-koodia ei generoida palvelimen puolella lainkaan (vrt. PHP), vaan tiedostot ovat täysin staattisia. Käyttöliittymän dynaaminen toiminnallisuus toteutetaan

siis kokonaan asiakkaan puolella selaimessa. Tämän johdosta rajapintaan vaaditaan useita nk. päätepisteitä (engl. ”api endpoint”). Rajapinnan päätepisteet ovat uri-osoitteita, jotka tarjoavat pääsyn resurssien lukemiseen ja manipulointiin, käyttäen HTTP-metodeja (GET, POST, PUT jne.). Yksi esimerkki tällaisesta resurssista tämän järjestelmän tapauksessa on tuloste. Tulosteen tiedot saadaan GET-metodilla, sen nimi voidaan päivittää PUT-metodilla, tai tuloste voidaan myös poistaa kokonaan DELETE-metodilla. Rajapinnan kautta käyttöliittymä suorittaa mm. seuraavia toimenpiteitä:

- Käyttäjän luonti (rekisteröityminen)
- Käyttäjän autentikointi (kirjautuminen)
- Lukijalaitteen monitorointi
- Tulosteiden lataaminen
- Käyttäjien listaaminen (ylläpitäjä)
- Laitteiden listaaminen (ylläpitäjä)

Pääsyä rajapinnan resursseihin rajoitetaan token-tyyppisellä autentikoinnilla. Käyttäjä kirjautuu sisään lähettämällä rajapinnalle käyttäjätunnuksen ja salasanan. Mikäli ne ovat validit, rajapinta lähettää vastauksena tokenin, joka antaa käyttöliittymälle pääsyn rajapinnan käyttöön. Kun käyttöliittymä tekee kutsun rajapintaan, lähetetään samalla token HTTP:n otsikkotiedoissa, ”Authentication” -kentässä. Mikäli em. tietue jää puuttumaan tai on virheellinen, palvelin vastaa ”Unauthorized” -virheilmoituksella.

6.2.1 Node.js

Taustaosa toteutettiin ”Node.js”-ohjelmistoalustalla. Kyseessä on palvelinohjelmointiin suunnattu runtime-ympäristö JavaScriptille. Vanhastaan JavaScriptiä on totuttu suorittamaan lähinnä selaimissa, mutta kielen monipuolistuttua ajan saatossa, se löysi tiensä myös palvelinten puolelle. Node.js:n etuna perinteisempiin ratkaisun pidetään erityisesti sen skaalautuvuutta suuriinkin järjestelmiin. (Node.js Foundation n.d. a)

Skaalautuvuus saavutetaan yksisäikeisellä prosessoinnilla, joka taasen on mahdollista koodin asynkronisen suorituksen johdosta. IO-kutsut ovat Node.js:ssä luonteeltaan ei-estäviä (engl. ”non-blocking”) ja niiden suoritus tapahtuu käyttäjälle näkymättömässä nk.

tapahtumasilmukassa (engl. ”event loop”). Ohjelmakoodissa siis lähinnä rekisteröidäänkin callback-funktioita tapahtumasilmukan suoritettavaksi. Tämän vuoksi palvelinohjelmointi Node.js:llä noudattaakin usein enemmän funktionaalisen ohjelmoinnin paradigmaa, kuin tavallista proseduraalista ohjelmointia. (Node.js Foundation n.d. b)

Node.js:n perusasennukseen sisältyy kirjastot HTTP-yhteyksien käsittelyyn. Käytännössä kuitenkin useimmiten suositaan kolmannen osapuolen kirjastoja web-serverien toteutuksessa, koska Node.js:n omat kirjastot ovat ominaisuuksiltaan jokseenkin suppeat. Tähän toteutukseen valittiin käytettäväksi ”Express.js”, joka on yksi suosituimmista kolmannen osapuolen web-sovellusalustoista Node.js:lle. (StrongLoop, et al. 2017a) Express.js:n avulla rajapinnan päätepisteiden luonti on hyvin yksinkertaista. Tämä voidaan demonstroida ”Hello World” -tyyppisellä esimerkillä:

```
app.get('/resurssi', (req, res) => {
        res.send('Hello World');
});
```

App-olion get-metodille annetaan ensimmäisessä parametrissä uri-polku, jolle rekisteröidään toisena parametrinä oleva callback-funktio. Kun palvelimelle tehdään GET-pyyntö kyseiseen polkuun (esim. ”www.example.com/resurssi”), suoritetaan em. callback-funktio. Kyseisen funktion parametreinä on serverin vastaanottama pyyntö (req-olio) ja lähetettävä vastaus, eli res-olio. Tässä tapauksessa vastaus on pelkkä merkkijono, joka lähetetään käyttämällä res-olion metodia ”send()”. Jos haluttaisiin käsitellä samaan polkuun tulevia POST-pyyntöjä, käytettäisiin get-jäsenfunktion sijasta post-funktiota. Samalla periaatteella voidaan käsitellä kaikki muutkin HTTP:n metodit. (StrongLoop, et al. 2017b)

6.2.2 MongoDB

Palvelimen tietokantaohjelmistoksi valittiin MongoDB, joka on hyvin suosittu juuri Node.js:llä tehdyissä sovelluksissa. Se lukeutuu niin kutsuttuihin ”NoSQL” -tietokantoihin, tarkoittaen sitä, että tietokanta perustuu johonkin muuhun kuin perinteiseen SQL-relaatiotietokantamalliin. MongoDB:ssä tieto tallennetaan kantaan JSON-tyylisinä dokumentteina, minkä vuoksi MongoDB:n ja Node.js:n yhteiskäyttö onkin hyvin saumatonta. Samasta syystä dataa voidaan tallentaa MongoDB-tietokantaan hyvin joustavalla tavalla, etenkin kun verrataan perinteisiin SQL-tietokantoihin. (MongoDB, Inc. 2018)

Vaikka Node.js sisältääkin natiivin tuen MongoDB-tietokantojen käytölle, on tässäkin tapauksessa usein yleisempää käyttää erillistä kirjastoa. Tässä toteutuksessa tietokantaa käytettiin Mongoose.js-kirjastolla, joka tuo kannan käyttämiseen lisää hyödyllistä abstraktiota. MongoDB:ssä tallennettaville dokumenteille ei voida etukäteen määritellä minkäänlaista mallia, jota dokumentin tulisi noudattaa, vaan kaikki data tallennetaan suoraan sellaisenaan. Tämä tekee mm. datan validoinnista hankalaa. Näihin tarpeisiin vastaa Mongoose.js, joka mahdollistaa tiedon tallentamisen kantaan ns. skeemojen mukaisesti, joita voisi verrata SQL-tietokantojen tauluihin. Skeemassa voidaan määritellä, mitkä tietueet dokumentin malliin kuuluvat, mitkä tietueet ovat pakollisia, oletusarvoja tietueille jne. Myös datan validointi helpottuu merkittävästi, sillä skeemoissa tietueet ovat vahvasti tyypitettyjä. (Mongoose n.d)

6.2.3 GhostPCL

Kuten aiemmin todettu, tietoliikennelaboratorion mittalaitteiden pääsääntöisesti käyttämä sivunkuvauskieli on Hewlett-Packardin kehittämä PCL. Toisin kuin esimerkiksi PDF, PCL ei ole nykykäyttöjärjestelmien yleisesti ymmärtämä tiedostoformaatti, joten mittalaitteilta kaapattu tulostusdata tulee muuntaa muihin käytännöllisempiin formaatteihin. Palvelin hoitaa muunnoksen käyttäen GhostPCL-ohjelmistoa. Kyseessä on Artifex Softwaren kehittämä tulkki PCL-sivunkuvauskielelle ja se kuuluu osana laajempaan GhostScript-ohjelmistoperheeseen. (Artifex Software 2016b)

GhostPCL:n toiminta perustuu nk. laitteisiin, joita ohjelma käyttää muunnetun datan generointiin. Esimerkki tällaisesta laitteesta on tässäkin toteutuksessa käytetty ”pdfwrite”, joka generoi ulostulonaan PDF-tiedostoja. Joskus on kuitenkin kätevämpää käsitellä tulosteita suoraan kuvatiedostoina, joten palvelin muuntaa tulosteet myös JPEG-tiedostoiksi. Tällöin ulostulolaitteeksi valitaan suoraviivaisesti nimetty ”jpeg”. (Artifex Software 2016a)

6.3 Käyttöliittymä

Käyttöliittymällä (”front-end”) tarkoitetaan sitä osuutta ohjelmistosta, jota suoritetaan selaimessa asiakaspuolella. Käyttöliittymän tehtäväksi jää informaation (etusijalla tulosteiden) esittäminen ja siihen suoraan liittyvä toiminnallisuus, taustaosan hoitaessa järjestelmän varsinaisen logiikan. Käyttöliittymä on käyttäjää lähimpänä oleva ja näkyvin osa koko järjestelmää. Siksi sen suunnittelussa ja toteuttamisessa tuleekin ottaa huomioon myös käyttökokemuksen kaltaiset seikat, perusfunktionaalisuuden ohella.

Selainpohjaisen käyttöliittymän etuja ovat alustariippumattomuus ja siten myös suoraviivainen kehitystyö. Käyttäjän ei myöskään tarvitse ladata ja asentaa ohjelmistoa tietokoneellensa. Tällainen ratkaisu mukailee nykypäivän yleistä trendiä; yhä useammat sovellukset siirtyvät pilvipalveluihin ja web-applikaatioihin. Näiden seikkojen johdosta selainpohjaisen käyttöliittymän toteuttaminen oli lähes itsestäänselvyys.

Käyttöliittymän ensisijainen tehtävä on näyttää käyttäjälle mittalaitteilta saatuja tulosteita, sekä mahdollistaa kyseisten tulosteiden lataaminen käyttäjän omalle tietokoneelle. Toiminnallisuuteen kuuluu luonnollisesti myös sisään kirjautuminen, ja mahdollisuus rekisteröidä uusi käyttäjä. Käyttöliittymä koostaa myös listausnäkymän käyttäjän viimeisimmistä tulosteista. Pääkäyttäjille käyttöliittymässä on myös listausnäkymät järjestelmän kaikista laitteista ja käyttäjistä.

6.3.1 Angular

Käyttöliittymä toteutettiin Angularilla, joka on Googlen kehittämä web-sovelluskehys (engl. ”web framework”) yhden sivun web-applikaatioiden kehittämiseen. Angular-sovellukset kirjoitetaan käyttäen HTML:ää ja TypeScript-kieltä, joka on JavaScriptin laajennos. Se lisää JavaScriptiin tuen tyypitykselle, luokille ja moduuleille. Selaimet eivät tue TypeScriptiä suoraan, joten koodi käännetään JavaScriptiksi ennen käyttöä. (Google 2018a)

Angularin arkkitehtuuri on modulaarinen. Sovellukset rakentuvat joukosta komponentteja. Jokainen komponentti määrittelee itselleen näkymän, joka sisältää käyttäjälle näkyviä HTML-elementtejä. Komponentti voi myös sisältää ohjelmalogiikkaa, jolla näkymän

sisältöä voidaan muokata. Yksinkertaistaen voidaan todeta Angular-sovellusten koostuvan useista sisäkkäisistä komponenteista, joiden näkymät muodostavat käyttäjän selaimessaan näkemän kokonaisuuden, sekä kyseisten komponenttien ohjelmallisesta logiikasta, jolla luodaan käyttöliittymän varsinainen toiminnallisuus. Tämä tiivistys jättää toki monia asioita huomioimatta, mutta aiheen syvällisempi tarkastelu ei tässä tapauksessa ole kovinkaan olennaista, erityisesti kun otetaan huomioon Angular-alustan monipuolisuus ja siten myös aihealueen laajuus. (Google 2018b)

6.3.2 Bootstrap

Käyttöliittymän HTML-osuudessa hyödynnettiin Bootstrap-sovelluskehystä. Bootstrap on kirjasto, joka sisältää liudan valmiita komponentteja responsiivisten web-sivujen luontiin. Tällaisia komponentteja ovat esimerkiksi erilaiset painikkeet, navigointipalkit ja valikot. (Refsnes Data 2018a) Komponenttien lisäksi Bootstrapin keskeinen ominaisuus on taulukkojärjestelmä (engl. ”grid”), joka tekee elementtien asettelusta sivulle hyvin suoraviivaista, samalla toteuttaen responsiivisuuden kriteerit (Refsnes Data 2018b).

Bootstrap koostuu pääosin CSS-muotoiluista, mutta muutamat komponenteista vaativat myös JavaScriptiä toiminnallisuuteensa. Tässä tapauksessa alkuperäisten JavaScript-kirjastojen sijaan hyödynnettiin kolmannen osapuolen kehittämää ”ng-bootstrap” Angular-moduulia, joka tuo Bootstrap-komponentit käytettäviksi Angular-komponentteina, siten tehden Bootstrapin käytöstä Angularin kanssa yhtenäisempää. (Ng-bootstrap n.d.)

7 POHDINTA

Työn lopputulos vastasi onnistuneesti sille asetettuihin lähtövaatimuksiin. Vaikka laajamittaista käyttöönottoa ei tältä osin vielä tehtykään, loi lukijalaitteen ja palvelinohjelmiston toteutus pohjan, jonka perusteella järjestelmä voidaan myöhemmin laajentaa koskemaan kaikkia haluttuja laboratorion mittalaitteita. Laajempi käyttöönotto tarkoittaakin lähinnä uusien lukijalaitteiden rakentamista, sillä muilta osin järjestelmä on valmis käytettäväksi.

Laboratoriossa työskentely muuttuu tuleville opiskelijoille sujuvammaksi sähköisen tulostusympäristön myötä, millä voi hyvinkin olla myös positiivinen vaikutus tehtävien laboratoriotyöselostusten laatuun. Koulun kannalta uusi järjestelmä on aiempaa helpompi ylläpitää. Järjestelmä on myös tulevaisuuden kannalta varmempi ratkaisu vähäisen huoltotarpeensa vuoksi; laitteita ei tarvitse uusia, eivätkä ne kulu käytössä. Tulostusympäristöllä toivotaan olevan yleinen positiivinen vaikutus työskentely- ja opiskelumukavuuteen tietotekniikan laboratoriossa.

Työn tekijälle järjestelmän toteutus toi arvokasta kokemusta etenkin uusimmista webteknologioista, mutta toki myös yleisemmällä tasolla elektroniikasta ja hieman laajemman projektin ohjelmoinnista. Täten projektin toteuttaminen antaa myös hyviä eväitä työelämään. Potentiaalisesti järjestelmästä saattaa myöhemmin hyötyä myös kolmannet osapuolet, sillä projektin lähdekoodit on mahdollista tulevaisuudessa julkaista vapaasti internetissä, avoimen lähdekoodin lisenssin alaisena.

LÄHTEET

Agarwal T. N.d. Operations of multiplexer and de-multiplexer circuit and applications. Luettu 11.4.2018.
https://www.elprocus.com/what-is-multiplexer-and-de-multiplexer-types-and-its-applications/

Arduino. 2015. Environment. Julkaistu 9.7.2015. Luettu 11.4.2018.
https://www.arduino.cc/en/Guide/Environment

Arduino Forum. 2012. Read Data From Parallel Port. Foorumikeskustelu. Julkaistu 30.5.2012. Luettu 11.4.2018.
https://forum.arduino.cc/index.php?topic=107996.0

Artifex Software. 2016. Details of Ghostscript output devices. Julkaistu 26.9.2016. Luettu 14.4.2018.
https://www.ghostscript.com/doc/9.20/Devices.htm

Artifex Software. 2016. What is Ghostscript? Julkaistu 26.9.2016. Luettu 14.4.2018.
https://www.ghostscript.com/doc/9.20/WhatIsGS.htm

Durda F. 2004. Centronics and IBM Compatible Parallel Printer Interface Pin Assignment Reference. Luettu 5.3.2018.
http://nemesis.lonestar.org/reference/computers/interfaces/centronics.html

ESP8266 Community Wiki. 2015. Basic Wiring Functions. Päivitetty 10.11.2017. Luettu 10.4.2018.
http://www.esp8266.com/wiki/doku.php?id=esp8266_gpio_pin_allocations

Espressif Systems. 2018. ESP8266EX Datasheet. Datalehti. Julkaistu 1.2.2018. Luettu 10.4.2018.
https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

Fielding, et al. 1999. Hypertext transfer protocol -- HTTP/1.1. Protokollakuvaus. Julkaistu 1.6.1999. Luettu 13.4.2018.
http://www.ietf.org/rfc/rfc2616.txt

FreeBSD. N.d. Common Page Description Languages. Luettu 7.4.2018.
https://www.freebsd.org/doc/handbook/printing-pdls.html

Google. 2018. Architecture overview. Luettu 13.4.2018.
https://angular.io/guide/architecture

Google. 2018. Introduction to components. Luettu 13.4.2018.
https://angular.io/guide/architecture-components

Grokhotkov I. 2017. Client. ESP8266 Arduino Core documentation. Luettu 14.4.2018
http://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/client-examples.html

Grokhotkov I. 2017. ESP8266WiFi library. ESP8266 Arduino Core documentation. Luettu 14.4.2018.
http://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/readme.html

Grokhotkov I. 2017. My ESP crashes running some code. How to troubleshoot it? ESP8266 Arduino Core documentation. Luettu 10.4.2018.
http://arduino-esp8266.readthedocs.io/en/latest/faq/a02-my-esp-crashes.html

Grokhotkov I. 2017. Reference. ESP8266 Arduino Core documentation. Luettu 10.4.2018.
https://arduino-esp8266.readthedocs.io/en/latest/reference.html

Medeiros R. N.d. Page Description Languages. Luettu 7.4.2018.
https://opentextbc.ca/graphicdesign/chapter/6-6-page-description-languages-pdl/

MongoDB, Inc. 2018. MongoDB Architecture. Luettu 15.4.2018.
https://www.mongodb.com/mongodb-architecture

Mongoose. N.d. Schemas. Luettu 13.4.2018.
http://mongoosejs.com/docs/guide.html

Ng-bootstrap. N.d. Getting started. Luettu 16.4.2018.
https://ng-bootstrap.github.io/#/getting-started

Node.js Foundation. N.d. About Node.js. Luettu 15.4.2018.
https://nodejs.org/en/about/

Node.js Foundation. N.d. Overview of blocking vs non-blocking. Luettu 15.4.2018.
https://nodejs.org/en/docs/guides/blocking-vs-non-blocking/

Omega. N.d. The RS232 Standard. Pdf-dokumentti. Luettu 5.4.2018.
https://www.omega.com/techref/pdf/RS-232.pdf

ON Semiconductor. 2009. Quad line EIA-232D receivers. Datalehti. Julkaistu 1.12.2009. Luettu 13.4.2018.
https://www.onsemi.com/pub/Collateral/MC1489-D.PDF

Parkkila E. N.d. Titeverkko. Päivitetty 25.11.2017. Luettu 12.4.2018.
https://gitlab.tamk.cloud/examples/gitlab-guide-wiki/wikis/titeverkko-fi

Peacock C. 2005. Interfacing the Standard Parallel Port. Julkaistu 15.6.2005. Luettu 4.3.2018.
http://retired.beyondlogic.org/spp/parallel.htm

Plese L. 2016. Digital logic 2. Julkaistu 3.8.2016. Luettu 12.4.2018.
http://scienceup.org/computer-science/digital-logic-2/

Refsnes Data. 2018. Bootstrap 4 get started. Luettu 16.4.2018.
https://www.w3schools.com/bootstrap4/bootstrap_get_started.asp

Refsnes Data. 2018. Bootstrap 4 grids. Luettu 16.4.2018.
https://www.w3schools.com/bootstrap4/bootstrap_grid_basic.asp

StrongLoop, et al. 2017. Express. Luettu 12.4.2018.
https://expressjs.com/

StrongLoop, et al. 2017. Routing. Luettu 12.4.2018.
https://expressjs.com/en/guide/routing.html

TAL Technologies. 2018. RS232 and Serial Communications. Luettu 10.4.2018.
http://www.taltech.com/datacollection/articles/serial_intro

Texas Instruments. 1975. Octal d-type transparent latches and edge-triggered flip-flops. Datalehti. Julkaistu 1.10.1975. Päivitetty 1.8.2002. Luettu 12.4.2018.
http://www.ti.com/lit/ds/symlink/sn54ls373.pdf

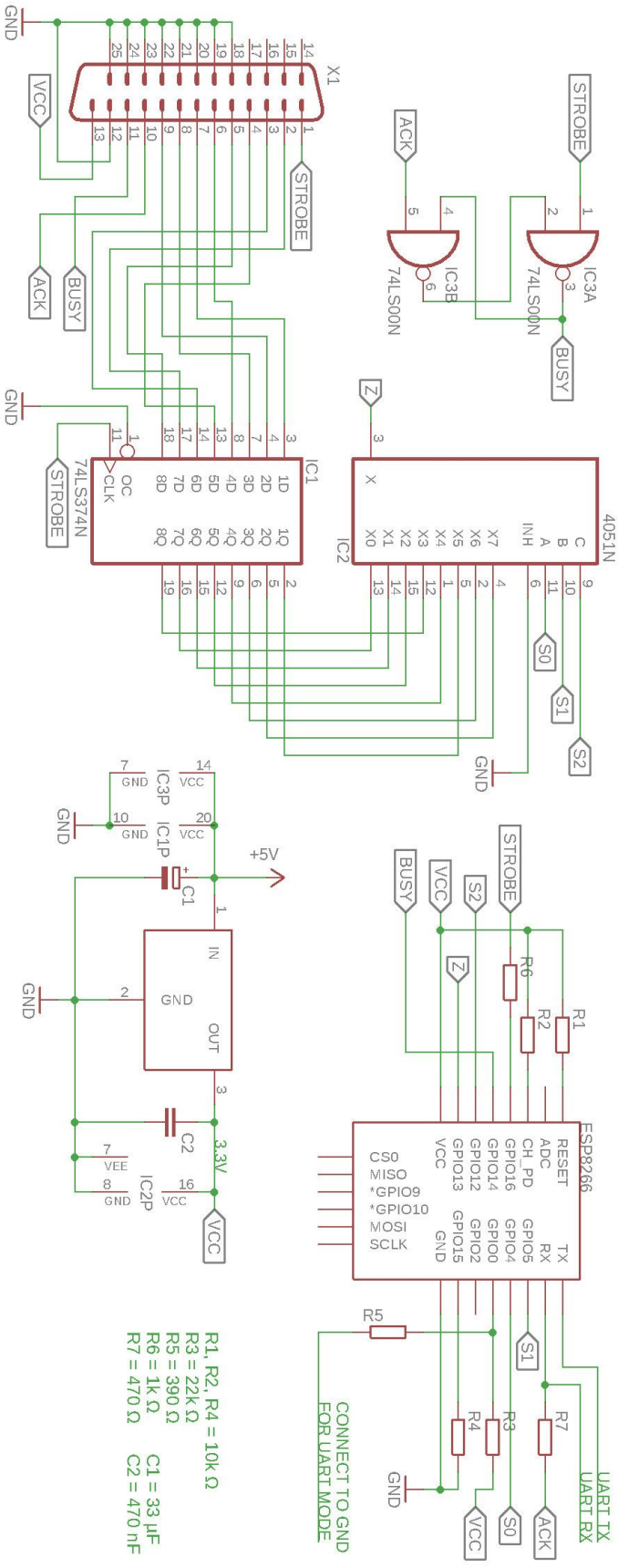
Texas Instruments. 1983. SNx400, SNx4LS00, and SNx4S00 quadruple 2-Input positive-NAND gates. Datalehti. Julkaistu 1.12.1983. Päivitetty 1.5.2017. Luettu 10.4.2018.
http://www.ti.com/lit/ds/symlink/sn5400.pdf

Texas Instruments. 1999. LM3940 1-A low-dropout regulator for 5-V to 3.3-V conversion. Datalehti. Julkaistu 1.5.1999. Päivitetty 1.2.2015. Luettu 13.4.2018.
http://www.ti.com/lit/ds/symlink/lm3940.pdf

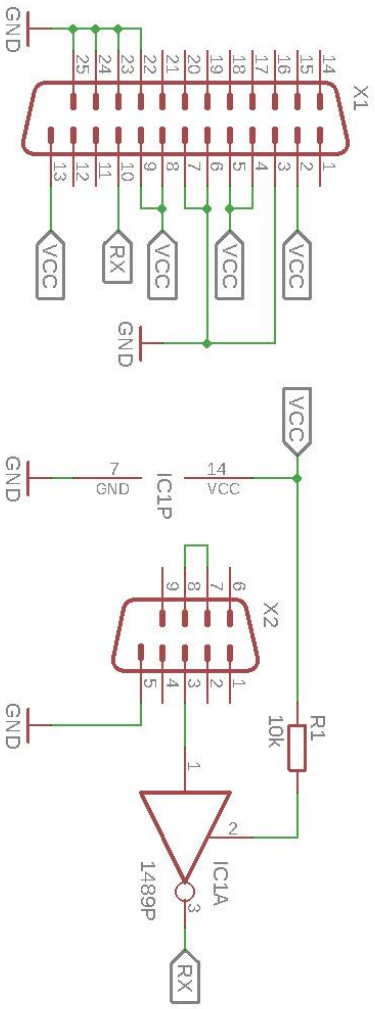
Theiling H. 2013. Theiling Online: Parallel Port: Centronics. Julkaistu 14.1.2013. Luettu 4.3.2018.
http://www.theiling.de/parport/centronics.html

LIITTEET

Liite 1. Lukijalaitteen kytkentäkaavio.



Liite 2. Sarjaväyläadapterin kytkentäkaavio.



Liite 3. Lukijalaitteen toiminnan vuokaavio.