Jukka-Erik Antin

KNX-pohjaisen automaatiojärjestelmän toteutus kolmannen osapuolen ratkaisuilla

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Sähkö- ja automaatiotekniikan tutkinto-ohjelma

Insinöörityö

4.4.2022

Tiivistelmä

Tekijä:	Jukka-Erik Antin
Otsikko:	KNX-pohjaisen automaatiojärjestelmän toteutus kolmannen osapuolen ratkaisuilla
Sivumäärä:	28 sivua + 3 liitettä
Aika:	4.4.2022
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Sähkö- ja automaatiotekniikka
Ammatillinen pääaine:	Sähkövoimatekniikka
Ohjaajat:	Lehtori Jarno Nurmio

Opinnäytetyön tarkoituksena oli käydä lävitse huokeita tapoja laajentaa olemassa olevaa KNX-automaatioverkkoa. Laitteistoina käytettiin kolmannen osapuolen edullisia ratkaisuja, kuten Atmega328P-mikrokontrolleria sekä Raspberry PI-minitietokonetta. Ohjelmistoissa pitäydyttiin suurelta osin avoimen lähdekoodin tuotteissa sekä yritetään välttää kaupallisen ETS-ohjelmiston käyttämistä.

Opinnäytetyö perustuu aiemmin koulussa valmistuneen opetussalkkuun, mihin lisättiin paneeliohjaus sekä itsetehtyjä KNX-laitteita.

Opinnäytetyö osoittaa miten voidaan rakentaa yksinkertainen kotiautomaatiojärjestelmä edullisesti.

Avainsanat: KNX, Arduino, Raspberry PI, atmega328, etäohjaus

Abstract

Author:	Jukka-Erik Antin
Title:	Implementing a KNX-based Automation System with Third-party Solutions
Number of Pages:	28 pages + 3 appendices
Date:	4 April 2022
Degree:	Bachelor of Engineering
Degree Programme:	Electrical and Automation Engineering
Professional Major:	Electrical Power Engineering
Instructors:	Jarno Nurmio, Senior Lecturer

The purpose of the thesis work was to clarify several approaches to expand existing KNX network.

Hardware used was low-cost general-purpose hardware, such as an Atmega328p microcontroller and a Raspberry Pi computer. The software used was open source and it was tried to avoid using official ETS5 software.

The work is based on Chun Leung's thesis from 2015, to the result of which remote control and self-made devices were added.

As result, this study shows how to build a simple home automation system at low cost.

Keywords: KNX, Arduino, Raspberry PI, atmega328, remote control

Sisällys

Lyhenteet

1 Johdanto 1

2 KNX-verkon perusteet 2

2.1 KNX-verkon rakenne 2
2.2 KNX-topologia 2
2.3 Turvallisuus 3
2.4 Osoitteet 4
2.5 Ohjauskenttä 5
2.6 NPCI-kenttä 6
2.7 Viestin kuorma 7
2.8 ACPI 7

3 Arduinon kytkeminen KNX-verkkoon 8

3.1 Fyysinen kytkentä 8
3.2 Ohjelmisto väylältä lukemiseksi 9

4 Raspberry PI-tietokoneen kytkentä väylälle 10

4.1 Raspberry PI:n sähköinen kytkentä 11
4.2 KNXD-palvelinohjelmisto 12
4.3 Väylälle kirjoittaminen sekä salkun laitteiden ohjaus 13
4.4 Salkun laitteiden selvittäminen Raspberry Pi:llä 16

5 Raspberry PI Zero-tietokoneen käyttäminen releohjaukseen 17

6 Arduinon käyttäminen ohjauspainikkeiden lukemiseen 19

6.1 Arduinon ohjelmisto 21
6.2 Muutoksia ohjelmistoon 22

7 Atmega328:n käyttö releohjaukseen 23

8 Hyödyllisiä komentoja KNXD-ohjelmiston käyttöön 26

9 Yhteenveto 27

Lähdeluettelo 29

Liitteet

Liite 1. Testiverkon looginen kytkentä

Liite 2. groupWrite.ino esimerkkitiedosto

Liite 3. Relelogiikan kytkentäkaavio

Lyhenteet

ATmega328p: kahdeksan bittinen mikro-ohjain

Arduino: Aloittelijaystävällinen kehitysalusta.

ETS: Engineering Tool Software. Ohjelmisto KNX-verkon konfigurointiin, josta usein näkee myös käytettävän nimeä EIB Tool Software.

GPIO: General-purpose input/output. Yleisnimitys liitännälle, jonka toiminnot ovat käyttäjän valittavissa.

IDE: Integrated development environment. Ohjelmointiympäristö.

Linjatoistin: Ohjelmoitu ohjaamaan KNX-väylän kaikki viestit läpi.

Linjayhdistin: Linjayhdistin joko estää tai välittää viestienvälityksen päälinjan ja linjan välillä, riippuen ohjelmoinnista.

Optoerotin: Optoerottimella toteutetaan signaalien galvaaninen erotus.

SELV: Safety Extra Low Voltage. Pienoisjännite, jossa jännitteen ei tule ylittää pienoisjännitteen (ELV) raja-arvoja normaaliolosuhteissa mukaan luettuna maasulut toisissa piireissä [1].

Tavu: Tavu on pituudeltaan 8 bittiä eli oktetti.

1 Johdanto

Opinnäytetyön tavoitteena on käydä lävitse erilaista lähestymistapaa KNX-verkon toteutukseen, projektin tärkein tavoite on pystyä toteuttamaan edullisesti automaatioväylä kotiin, kesämökille tai ihan vain työpajaan. Työtä varten valmistettiin useita laitteita, joita lisättiin yksitellen valmiiseen automaatiokokonaisuuteen, sekä lopuksi koko verkon toteutus itse tehdyillä laitteilla (pois lukien teholähde). Työtä varten on koululta lainassa opinnäytetyönä rakennettu KNX-koulutuspaketti, johon on toteutettu toimiva KNX-verkko. [12.]

Opinnäytetyössä käsitellään KNX-verkon väyläliikennettä, muistiosoitteita sekä tarkastellaan erilaisia tapoja rakentaa laitteita KNX-verkkoon. Käytössä on kahdeksan bittiseen ATmega328p-piirisarjaan perustuva Arduino-kehitysalusta, vakio ATmega328p-mikropiiri ilman ylimääräistä logiikkaa sekä Raspberry PI-minitietokone, joka toimi releiden ohjauksessa sekä rajapintana Internetin ja KNX-väylän välillä mahdollistaen etäkäytön tabletilta.

Opinnäytetyötä varten oli koululta lainassa ETS5-ohjelmisto KNX-verkon konfigurointiin, mutta yritin välttää sen käyttöä, sillä kalliin ohjelman hankkiminen ei kuulu omatoimisen rakentelun luonteeseen ja sen tuoma hyöty olisi vähäistä.

Metropolian opetussalkkuun liittyviä KNX:n asetuksia ei käydä lävitse, eikä salkun kytkentöihin kosketa muuten kuin haaroittamalla yhdestä jakorasiasta parikaapelilähtö pistokkeille, josta KNX-väylän pystyy jatkamaan salkun ulkopuolella oleville komponenteille.

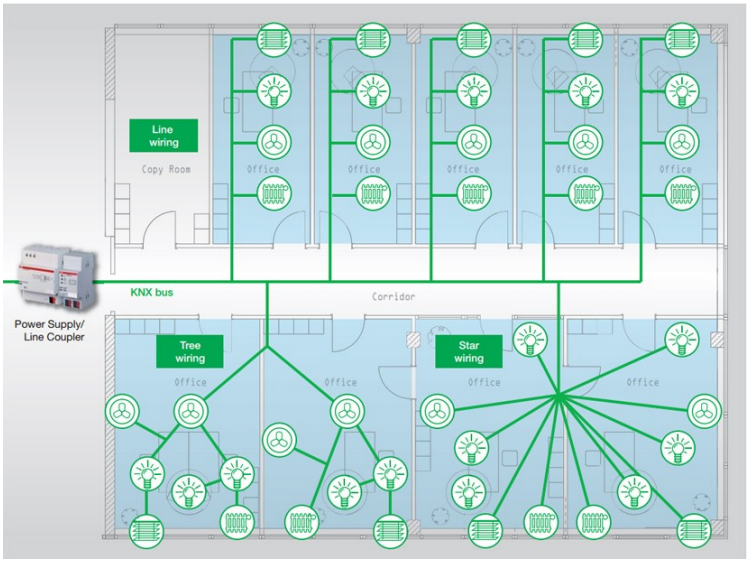
Opinnäytetyössä tiedonsiirto tulee tapahtumaan yksinomaan parikaapeliväylässä, sillä se on helpoin sekä halvin toteuttaa eikä itse verkon laajentaminen tarvitse käyttäjältä asennuslupia johtuen verkon virtalähteiden SELV-jännitteestä, jossa jännitteen ei tule ylittää pienoisjännitteen (ELV) raja-arvoja normaaliolosuhteissa. Muita mahdollisuuksia olisi sähköverkon sekä radioteitse ylitse tapahtuva liikenne. [2, s. 29; 2, s. 34; 2, s. 40.]

2 KNX-verkon perusteet

Opinnäytetyössä käydään kevyesti lävitse KNX-verkon fyysinen rakenne, linja-topologia, muistiosoitteet ja tiedonsiirtoon liittyviä parametrejä. Muistiosoitteiden tai ohjauskenttien opiskeleminen ei ole tarpeellista valmiita kirjastoja käyttäessä, mutta kirjoittamalla alusta alkaen kaiken itse on syytä sisäistää viestin tarvitsemat kentät.

2.1 KNX-verkon rakenne

Fyysinen KNX-verkko voi olla rakennettu linja-, puu- tai tähtitopologiana tai niiden sekoituksina, ainoastaan rengaslinjoja ei saa muodostua. Väylää voi haaroittaa joko omissa rasioissaan tai laitteiden luona (ks. kuva 1), verkon suunnittelijan tulee ainoastaan huolehtia, ettei yhden linjan tai linjasegmentin kokonaispituus ylitä 1000:ta metriä. [2, s. 26.]



Kuva 1. Linja-, tähti- ja puuhaaroitettu kytkentä [13].

2.2 KNX-topologia

KNX-verkossa yksinkertaisin väylätopologia on linja. Se voi pitää sisällään maksimissaan 64 laitetta, mutta yksittäisen linjan laitemäärän voi kasvattaa 255

laitteeseen kolmella linjatoistimella, joiden kanssa pystytään tekemään linjaan lisäksi kolme linjasegmenttiä, joissa jokaisessa on 64 laitepaikkaa. Linjatoistimet tulee asentaa linjaan, niitä ei voi kytkeä sarjaan, jokainen linjatoistin lasketaan myös laitteeksi ja vähentävät yhden vapaan laitepaikan linjalta. [2, s. 27.]

Kytkettäessä KNX-linjaan linjayhdistimellä toinen linja alkuperäisestä aletaan kutsumaan päälinjaksi. Yhteensä 15 edellä mainitun kaltaista linjaa voidaan lisätä päälinjaan. Linjatoistimia ei voi kytkeä päälinjaan, joten päälinjassa on mahdollista pitää maksimissaan 64 laitetta. [2, s. 27.]

2.3 Turvallisuus

KNX-verkon suojaus perustuu suurilta osin estämällä asiattomien henkilöiden pääsy verkkoon. Ulkona sijaitsevat laitteistot ovat erityisessä riskissä, ja väylälaitteet tulisi asentaa joko riittävän korkealle tai muuten estää asiattomien pääsy väylään. Kotelot voidaan lisäksi suojata hälytyksen aiheuttavalla koskettimella. [3, s. 3.]

Ulkopuolisen henkilön päästessä käsiksi KNX-väyläkaapeliin, on se sitten valokatkaisija, virtalähde, ilmanpaineanturi tai liiketunnistin, pystyy ko. henkilö lukemaan sekä kirjoittamaan omia komentoja verkkoon sekä ohjaamaan väylään kytkettyjä laitteita. Tästä syystä tulee varmistua, etteivät ulkopuoliset pääse käsiksi KNX-laitteisiin tai kaapelointeihin. Edellä mainitun avoimuuden positiivisena puolena on uusien laitteiden helppo rakentaminen sekä asentaminen verkkoon yksinkertaista.

Jos ulkopuolisten henkilöiden pääsyä KNX-väylään ei pystytä varmuudella estämään, voidaan asentaa turvalliseen paikkaan ennen liitäntälaitetta linjayhdistin, joka on ohjelmoituna päästämään vain ennalta määritettyihin ryhmäosoitteisiin menevät komennot läpi. [3, s. 3.]

Tiedon salaukseen on kehitetty KNX IP Secure sekä KNX Data -standardit, jotka mahdollistavat KNX-viestien allekirjoittamisen sekä datan salauksen tietoliikenneverkon sekä parikaapelin ylitse tapahtuvassa tiedonsiirrossa. [3, s. 6.]

2.4 Osoitteet

KNX-verkossa on käytössä niin kutsuttuja fyysisiä sekä ryhmäosoitteita. Osoitteiden tarkoitus ja käyttö eroaa huomattavasti toisistaan ja on erittäin tärkeää hahmottaa osoitteiden käyttötavat. Osoitteiden nimeäminen on kirjavaa, sillä englanninkielinen materiaali mainitsee fyysiset osoitteet, kun taas osa suomennetuista lähdeteoksista kutsuvat niitä joko fyysisiksi tai yksilöllisiksi osoitteiksi. Työssä käytetään ainoastaan fyysistä osoitetta, sillä se on lähempänä englanninkielistä vastinetta ja vähentää näin ollen erehdyksen mahdollisuutta.

Fyysinen osoite

Jokaisella KNX-laitteella on fyysinen osoite, joka saadaan laitteen paikasta KNX-verkossa. Fyysinen osoite muodostuu mallin alue.linja.linjalaite mukaisesti.

Ryhmäosoite

Ryhmäosoite ei ole sidoksissa fyysiseen osoitteeseen, vaan ne pystytään määrittelemään vapaasti. Yhdellä laitteella voi olla useitakin ryhmäosoitteita, sillä ryhmäosoite vastaa laitteessa yhden toiminnon osoitetta. Yleisimpiä tapoja määritellä osoite on 2- sekä 3-portainen. Kolmeportaista käytettäessä voi KNX-verkon suunnittelija määritellä osoitteen esimerkiksi muotoon kerros/toiminto/laite. Se ei kuitenkaan ole sidoksissa millään tavoin fyysiseen rakenteeseen. Tässä työssä käytetään ainoastaan kolmetasoista merkintätapaa.

Osoitteiden sijainti KNX-viestissä

Osoitteille varattu alue on KNX-viestin kahdeksan bittiä pitkät tavut 3–6. Viestin sisältämien osoitteiden ollessa ’11 09 09 07’ (kuva 2) saadaan siitä sekä fyysinen osoite että ryhmäosoite.

```
BC 11 09 09 07 E1 00 81 35
   11 09        = 0001 0001 0000 1001 (Fyysinen osoite)
         09 07  = 0000 1001 0000 0111 (Ryhmäosoite)
```

Kuva 2. Kirjoitusviesti, jossa valittuna fyysinen osoite sekä ryhmäosoite.

Fyysisen osoitteen muodostaminen

Fyysinen osoite sisältyy valitun alueen kahteen ensimmäiseen tavuun ’11 09’ (ks. kuva 2), ja se muodostuu seuraavan kaltaisesta binäärimuotoisesta sarjasta ’00010001 00001001’. Osoitteen neljä ensimmäistä bittiä kuvaavat aluetta, seuraavat neljä bittiä linjaa ja loput kahdeksan bittiä laitetta. Heksadesimaaliosoite ’11 09’ kuvaa siis fyysistä osoitetta 1.1.9.

Aluebiteillä pystytään osoittamaan runkolinja sekä 15 aluetta, runkolinjan ollessa bitit 0000 ja alueiden 0001–1111. Linjabiteillä määritellään päälinja sekä 15 linjaa, päälinjan ollessa bitit 0000 ja linjojen 0001–1111. Osoitteen jälkimmäinen tavu mahdollistaa 255 erillistä laitetta linjaa kohden. [2, s. 27.]

Ryhmäosoitteen muodostaminen

Ryhmäosoitteen ollessa kolmiportainen bittisarjaan tulee käyttää jakoa 5bit/3bit/8bit, joten valitun osoitteen tavujen kolme ja neljä ollessa ’09 07’ (ks. Kuva 2), on osoitteen heksadesimaali binäärimuodossa ’00001 001 00000111’ ja siitä muodostuu ryhmäosoite 1/1/7. Huomioitavaa on se, että vanhemmissa ETS-ohjelmistoissa (Eib Tool Software) ryhmäosoitteen eniten vaikuttava bitti tuli olla 0, joten osoitteeksi muodostui kolmiportaisena 4bit/3bit/8bit. [7, s. 11.]

2.5 Ohjauskenttä

Ohjauskenttä sisältää viestin pituuteen liittyvää informaatiota, prioriteetin sekä toistettavuuden. Viestin alkaessa BC-heksadesimaalilla vastaa se informaatiota ’1011 1100’ joka kuvaa taulukon 1 mukaista normaalia viestiä, jossa viestin

pituus on 8–23 tavua. Viesti ei ole toistettu ja sen prioriteetti on matala. [8, s. 15; 9.]

Taulukko 1. Ohjauskenttä.

	Byte 1								
	Control field								
Bit	7	6	5	4	3	2	1	0	
Fixed bits		0		1			0	0	
Telegram type	x								0: 9…263 Bytes, 1: 8..23 Bytes
Repetition status			x						0: Repeated / 1: Not repeated
Priority					x	x			00 system, 10 urgent 01 normal, 11 low

2.6 NPCI-kenttä

NPCI-kenttä (taulukko 2) sijaitsee viestin kuudennessa tavussa. Kentän ensimmäinen bitti kuvaa vastaanottavan osoitteen tyyppiä. Bitin arvon ollessa yksi, on kyseessä normaali ryhmäosoite, muuttamalla bitin nollaksi voidaan viesti osoittaa suoraan fyysiseen osoitteeseen. Seuraavat kolme bittiä ovat ns. routing counter jolla indikoidaan monenko linjalaitteen kautta viesti on kulkenut, alkaen kuudesta ja vähentyen aina yhdellä laitteiden läpi kulkiessa. Viimeiset neljä bittiä ilmaisee datakentän pituutta.

Taulukko 2. NPCI-tavun sisältö.

	Byte 6								
	NPCI								
Bit	7	6	5	4	3	2	1	0	
Address type	x								0 group, 1 device address
Routing counter		n	n	n					0…7
Lenght					n	n	n	n	0...15

2.7 Viestin kuorma

Kuorma sijaitsee tavuissa 7 ja 8 (taulukko 3) ja alkaa TPCI-kentällä, jossa ensimmäinen bitti ilmaisee, onko kyseessä data- vai kontrollipaketti. Seuraava bitti kertoo, käytetäänkö sekvenssinumeroa ja bitin ollessa 0 voidaan sekvenssikenttä jättää huomioimatta.

Taulukko 3. Kuorman sisältö, ACPI voi olla neljä tai kymmenen bittiä pitkä.

	Byte 7								Byte 8							
	TCPI / ACPI								APCI							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
ACPI							a	a	a	a	b	b	b	b	b	b
Sequence number			n	n	n	n										
TPCI	x	x														

2.8 ACPI

ACPI-kenttä alkaa tavun 7 lopusta (ks. Taulukko 3), sillä määritellään datan tyyppi. Yleisimmin käytetyt ACPI-koodit ovat ryhmäosoitteen tilatiedon kysymiseen tarkoitettu 0000 (GroupValueRead), vastausviesti ryhmäosoitteen arvon kyselyyn 0001 (GroupValueResponse) sekä ryhmäosoitteen tilan muuttamiseen tarkoitettu 0010 (GroupValueWrite). Käytännössä muita taulukon 4 ACPI-komentoja ei tarvitse käyttää laitteiden ohjauksissa.

Taulukko 4. ACPI-kentän koodeja [11, s. 28].

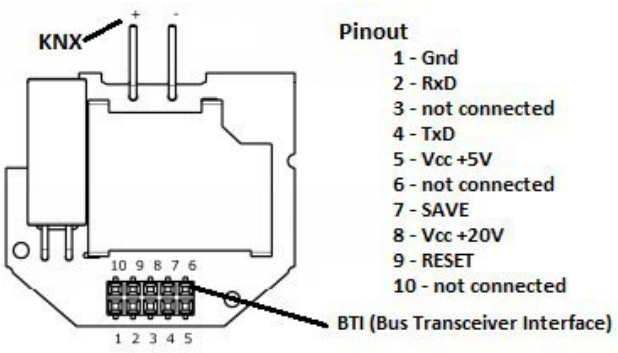
APCI	Name		
0000	GroupValueRead	1000	MemoryRead
0001	GroupValueResponse	1001	MemoryResponse
0010	GroupValueWrite	1010	MemoryWrite
0011	IndividualAddrWrite	1011	UserMessage
0100	IndividualAddrRequest	1100	MaskVersionRead
0101	IndividualAddrResponse	1101	MaskVersionResponse
0110	AdcRead	1110	Restart
0111	AdcResponse	1111	Escape

APCI-kentän ollessa neljä bittiä pitkä ja datan alle kuusi bittiä se voidaan sisällyttää kaksi tavuisen kuorman loppuun. Datan ollessa pidempi kuin kuusi bittiä tai käytetään 10 bittistä APCI kenttää, tulee lisätä tavuja kuorman jälkeen tiedon pituuden vaatima määrä, tavujen määrä ei kuitenkaan saa ylittää tavussa kuusi (taulukko 2) määriteltyä pituutta [10].

3 Arduinon kytkeminen KNX-verkkoon

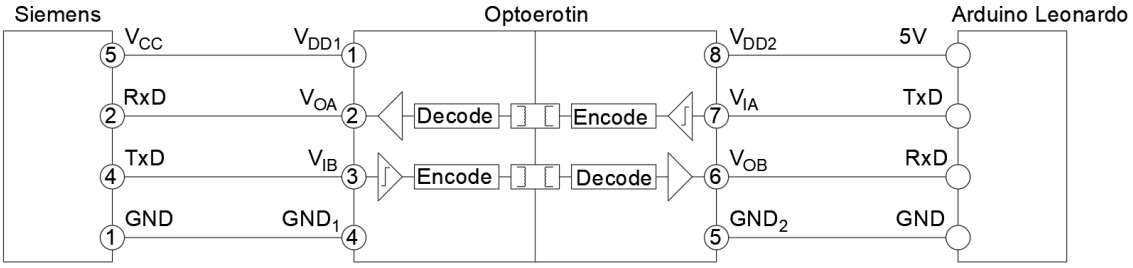
3.1 Fyysinen kytkentä

Arduino Leonardon kytkemiseksi väylään tarvitaan lisäksi Siemensin väylämuunnin (kuva 3) sekä optoerotin galvaanisen erotuksen tuottamiseksi. Galvaaninen erottaminen on tarpeen, jottei KNX-väylän ja tietokoneen välille pääse syntymään potentiaalieroa, joka pahimmillaan voi vaurioittaa tietoliikennepiirejä.



Kuva 3. Siemens-väylämuunnin [16].

Siemensin valmistama väylämuunnin on kytketty KNX-väylän nastoista KNX-verkkoon ja sarjaliikenne nastat puolestaan optoerottimen kautta Arduino Leonardo -mikrokontrolleriin. Optoerottimen molempia puolia tulee syöttää virtalähteillä, jotka ovat samassa potentiaalissa tietoliikennepiirien kanssa, Kytkentään toteutettiin kuvan 4 mukaisella kytkennällä.



Kuva 4. Optoerottimen kytkentäkaavio väylämuuntimen ja Arduino Leonardon välillä.

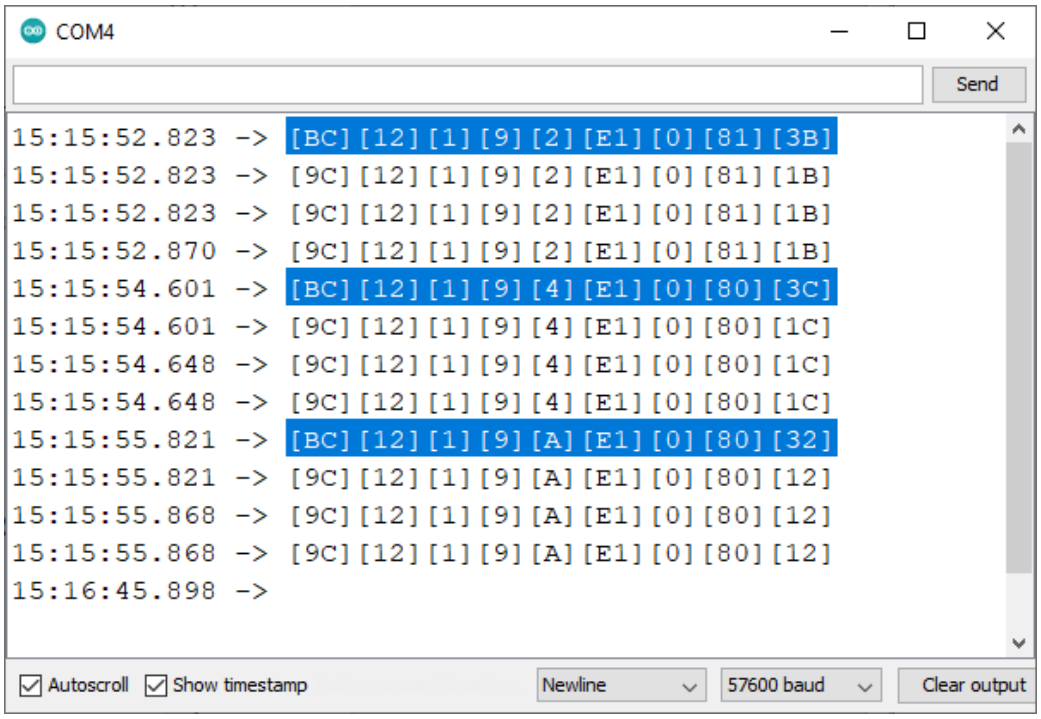
3.2 Ohjelmisto väylältä lukemiseksi

Yksinkertaisin testi väylän lukemiseksi on käyttää arduino:n mukana tulevaa ohjelmointiympäristoä eli IDE:ä (Integrated development environment), sekä mukana tullutta MultiSerial.ino-esimerkkiohjelmistoa ja muuttaa sitä siten, että ainoastaan luettiin KNX-väylää eikä kirjoiteta sinne mitään (ks. esimerkkikoodi 1).

```
float aika = micros();
void setup() {                    // Alustetaan sarjaportit:
  Serial.begin(57600);            //USB (serial0)
  Serial1.begin(19200);           //KNX (serial1)
}
void loop() {
  if (Serial1.available()) {      //Jos serial1 puskurissa on dataa.
    aika = micros();              //Otetaan datan aika talteen.
    int inByte = Serial1.read(); //luetaan tavu muuttujaan.
    Serial.print("[");            //tulostetaan serial0 (USB) väylälle
    Serial.print(inByte, HEX);    //vastaanotettu tavu HEX muodossa
    Serial.print("]");            //lisäten sulkeet ympärille: [XX]
  }
  if ((aika + 2000) < micros()){ //Jos viimeisimmästä tavusta on yli
    Serial.print("\n");           //2ms, tulostetaan rivinvaihto
    aika = micros() + 50000000;   //ja oletetaan että viesti loppui.
  }                               //Lopuksi annetaan ajalle 50s arvo
}
```

Esimerkkikoodi 1. Arduinolle kirjoitettu koodi väyläliikenteen kuuntelemiseen.

Painettaessa KNX-salkun nappeja tulisi Arduino IDE:n sarjaliikennekonsoliin tulostua kuvan 5 mukaista viestiliikennettä.



Kuva 5. Arduino IDE-sarjaliikennekonsoli, jossa näkyvillä vastaanotettuja KNX-viestejä heksadesimaalina sekä haettavat viestit vahvistettuna.

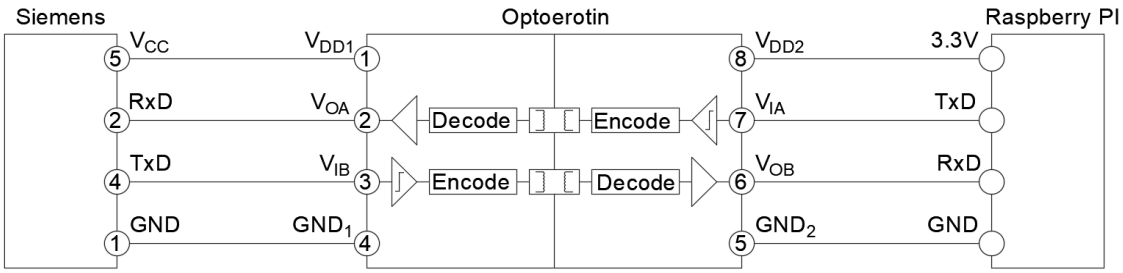
Kuten kuvasta 5 voi päätellä, viestiliikenne vaikuttaa järkevältä sillä valituissa riveissä havaitaan laitteen 1.2.1 yrittävän kirjoittaa ryhmäosoitteisiin 1/1/2, 1/1/4 sekä 1/1/10. ’9C’ alkuiset viestirivit kuvastavat samaa viestiä, mutta ohjauskentän (ks. taulukko 1) kolmas bitti on nolla ja se kuvastaa toistettua viestiä, mikä ilmenee laitteiston tunnistaessa virheen ryhmäosoitteellisen viestin lähetyksessä linjalle. [14, s. 23.]

4 Raspberry PI-tietokoneen kytkentä väylälle

Raspberry PI-mikrotietokoneen käyttöönotto KNX-laitteeksi alkoi lataamalla uusin levykuva mikrotietokoneen valmistajan nettisivulta ja kirjoittamalla se mikrotietokoneen muistikortille. Palvelinohjelmistoksi valikoitui KNXD-palvelin ja Michael Albertin tekemä KNXD-komentosarja, joka automatisoi asennusta asettamalla Rapberry PI-tietokoneen porttien asetukset sopiviksi KNX-väyläkommunikointia varten. Tarvittavat laitteet Raspberry PI:n lisäksi ovat samat kuin Leonardolla eli Siemensin väylämuunnin ja optoerotin.

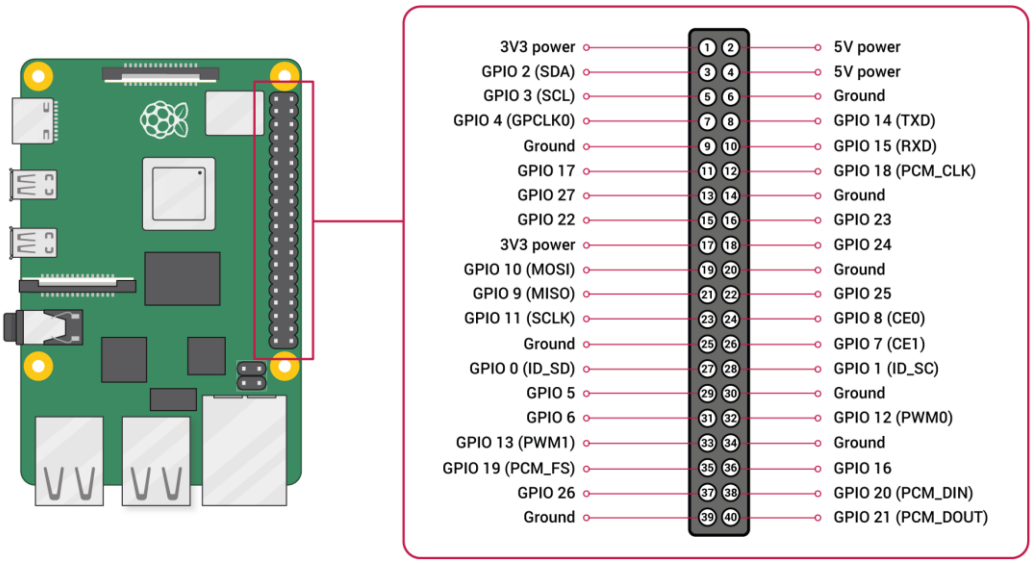
4.1 Raspberry PI:n sähköinen kytkentä

Raspberry Pi-tietokoneen sarjaliikennepiiri on suunniteltu toimimaan 3,3 voltin jännitteellä ja Siemensin väylämuuntimen ulostulo on 5 volttia, Jännitteiden eroavaisuuden sekä mahdollisten potentiaalierojen takia tulee laitteiden välille kytkeä optoerotin (kuva 6). Kytkentä on lähes identtinen Arduinon kytkennän kanssa.



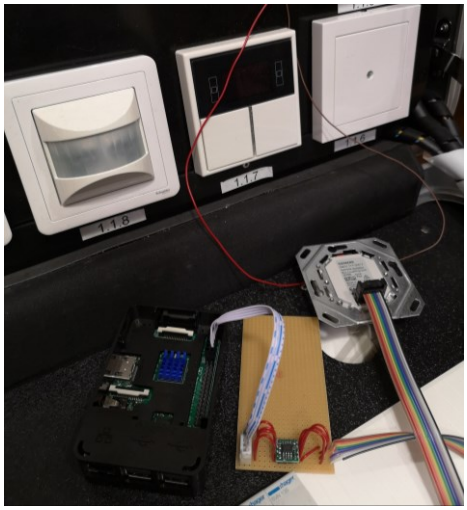
Kuva 6. Raspberry Pi-minitietokoneen kytkentä KNX-väylään.

Optoerottimen ja Raspbery Pi'n välisessä kytkennässä on huomioitava, että pinnien numerointi ei ole yhtenevä liittimen numeroinnin kanssa, pinnit tx(14) sekä rx(15) ovat liittimessä paikassa 8 ja 10 (kuva 7), väärinkäsitysten välttämiseksi numerointia ei ole merkitty kytkentäkuviin.



Kuva 7. Raspberry Pi:n yleiskäyttöinen liitäntärima eli GPIO (General purpose input/output) [17].

Laitteen toiminnan tarkastamista varten tehty kytkentä oli nopeasti kytkentälangalla tehty asennus, jossa galvaaninen erotin on juotettuna väliin (kuva 8).



Kuva 8. Laitteen toiminnan tarkastaminen, vasemmalla Raspberry Pi, keskellä optoerotin ja oikealla siemensin väylämuunnin, Takana KNX-laitteistoa.

Ensimmäiset testit olivat sarjaliikenneportin lukemista sekä siirrettyjen viestien tulostus näytölle, kuten Arduinon kanssa meneteltiin aiemmin.

4.2 KNXD-palvelinohjelmisto

Käyttämällä Raspberry Pi:ssä KNXD-ohjelmiston komentoa ”knxtool vbusmonitor1 ip:localhost”, tulisi ohjelmiston ryhtyä kuuntelemaan KNX-verkkoa ja tulostamaan väyläliikennettä näytölle. Painettaessa KNX-verkossa olevaa valokytkintä, minkä fyysinen osoite on 1.1.1, lähettää kytkin loogisen tilan ’1’ ryhmäosoitteelle 1/1/2, joka tässä tapauksessa aktivoi valoa ohjaavan releen ja näin ollen saa ohjattavan valaisimen syttymään.

> L_Busmon: BC 11 01 09 02 E1 00 81 38 :L_Data low from 1.1.1 to 1/1/2 hops: 06 T_Data_Group A_GroupValue_Write (small) 01

Kytkimen napin painaminen toiseen suuntaan lähettää muilta osin saman viestin, mutta tilana on ’0’

L_Busmon: BC 11 01 09 02 E1 00 80 39 :L_Data low from 1.1.1 to 1/1/2 hops: 06 T_Data_Group A_GroupValue_Write (small) 00

KNXD-ohjelmiston mukana tulevat ohjelmat ovat täysin riittäviä jo yksinään, kirjoitin silti yksinkertaisen python-ohjelmiston (kuva 9) ja putkitin KnxTool:n syötteen sille, näin pääsin helpommin sisään verkkoliikenteeseen.

```
pi@raspberrypi: ~/knx
BC 11 01 09 01 E1 00 81 3B      last telegram message
   11 01                        source address (Physical)
                                0001 . 0001 . 00000001
                                  1  .  1  . 1

         09 01                  target address (group)
                                00001 / 001 / 00000001
                                  1   /  1  /  1

               E1               Routing counter ja viestin koko
                                11100001
                                1         Type of the destination address
                                 110      Routing counter (6)
                                    0001  data lenght (1 tavua pitka)

                  00 81         Short data
                                0000000010000001
                                000000             Reserved
                                      0010         command (Send value on bus)
                                          000001   data


      1/1/1 : 1      1/1/2 : 0      1/1/4 : 0      1/1/6 : 0      1/1/7 : 1
      1/1/8 : 0      1/2/2 : 1      1/3/2 : 0
```

Kuva 9. capture.py-ohjelmiston tulostus, napin lähettäessä ’1’ tilan valaisinta ohjaavalle kontaktorille.

Ohjelmisto tulostaa kuvassa 9 saman datan kuin KnxTool, mutta helpommin hahmotettavissa olevassa muodossa. Ohjelmiston alareunassa on näkyvillä osoitteet, joihin on väylällä kirjoitettu sekä niiden viimeinen tila. Ohjelmisto otti huomioon ainoastaan viestit, joiden datan pituus on alle kuusi bittiä pitkä. Ohjelma oli lähinnä opettelua varten ohjelmoitu, enkä nähnyt tarpeelliseksi jatkaa ohjelman kehittämistä enempää.

4.3 Väylälle kirjoittaminen sekä salkun laitteiden ohjaus

Yksinkertainen KNX-väylälle kirjoittaminen tapahtuu Raspberry PI:n konsolista komennolla ”knxtool groupswrite ip:localhost 1/1/2 1”, millä lähetetään lyhyt

kuusibittinen heksadesimaali ’1’ ryhmäosoitteeseen 1/1/2, tämän KNX-konfiguraation tapauksessa osoitetta 1/1/2 kuunteleva rele vetää ja releen ohjaama lamppu syttyy. Lampun sammuttamiseksi tule kirjoittaa ryhmäosoitteeseen heksadesimaali ’0’.

Lyhyen viestin lähetys mahdollistaa arvojen 0–63 lähettämisen KNX-väylälle, suuremmat arvot pyörähtävät takaisin nollaksi, sillä vain 6 vähiten merkitsevää bittiä huomioidaan, desimaaliluvun 64 binaarimuotoinen esitys on 01000000 ja kuusi vähiten merkitsevää bittiä ovat nollia, joten lähetettäväksi viestiksi muodostuu nolla.

Opetussalkussa oli mukana himmennettäviä valo-ohjaimia, joiden ohjaukseen on määritelty useampia ryhmäosoitteita, päälle ja pois toimintojen osoite on 1/1/4 ja lähettämällä nolla tai yksi osoitteeseen 1/1/4, se vastaavasti sammutti tai sytytti lampun.

Himmennyksen tapauksessa tuli kytkintä pitää pohjassa sammutusasennossa, jolloin kytkin lähettää komennon ’01’ osoitteeseen 1/1/5 ja lampun kirkkausarvo alkaa laskemaan, käyttäjän vapauttaessa napin lähettää kytkimen ohjain arvon ’00’ samaan osoitteeseen pysäyttäen valaisimen himmentymisen. Lampun kirkkaammaksi säätäminen kuvastaa painonapin pitämistä päällä -asennossa, mikä lähettää osoitteeseen 1/1/5 koodin ’09’, joka käynnistää lampun kirkastumisen. Toiminnon vapauttaminen puolestaan lähettää koodin ’08’, pysäyttäen kirkastumisen.

Himmennystoimintoa testatessa Raspberry Pi:llä käytettiin esimerkkikoodia 2, joka ohjaa lampun kirkastumaan ja himmentymään.

```
#/bin/bash
# ohjataan lamppua kirkkaammaksi kaksi sekunttia
knxtool groupswrite ip:localhost 1/1/5 9
sleep 2
knxtool groupswrite ip:localhost 1/1/5 8

sleep 6 #Odotetaan 6 sec

# himmennetään lamppua kaksi sekunttia
knxtool groupswrite ip:localhost 1/1/5 1
sleep 2
knxtool groupswrite ip:localhost 1/1/5 0
```

Esimerkkikoodi 2. Bash-koodi lampun kirkkaustason muuttamiseksi.

Komentojen välillä oleva kahden sekunnin viive on sopivaksi haettu ja valaisimen himmennystoiminto ohjautui ääripäästä toiseen. Askeleittain ajettaessa voi myös lähettää heksadesimaali arvon ’0F’, joka ohjaa lampun kirkastumaan yhden askeleen verran, ja koodilla ’07’ valo himmeni. [5.]

Tilatiedon kysyminen KNX-laitteelta

Käyttöliittymän teossa on lähes yhtä tärkeää tietää laitteen sen hetkinen tila kuin ohjaussignaalin lähettäminen. Yksinkertainen lampun tilatiedon kysely on muotoa ”knxtool groupread ip:localhost 1/1/2”, joka saa laitteiston lähettämään väylälle viestin, jossa kysytään ryhmäosoitteen 1/1/2 tilatietoa.

Valaisin itsessään ei vastaa, sillä valaisimelle menee pelkästään sähkönsyöttö, jossa on mukana vaihekulmaohjaus himmennystä varten. Joten vastaus ’00’ tulee osoitteessa 1.1.4 olevalta himmentimeltä seuraavassa muodossa.

L_Busmon: BC 11 04 09 02 E1 00 40 FC :L_Data low from 1.1.4 to 1/1/2 hops: 06 T_Data_Group A_GroupValue_Response (small) 00

Himmennin ei reagoinut kyselyihin kuin päälläolon puolesta, himmennyksen voimakkuutta ei pystynyt lukemaan.

4.4 Salkun laitteiden selvittäminen Raspberry Pi:llä

Koululta lainassa olevasta salkusta tuli saada selville ryhmäosoitteet, useimmat pystyy lukemaan väylältä kokeilemalla kytkimiä ja kuuntelemalla väylää. Tähän tarkoitukseen kirjoitettiin pieni ohjelmisto automatisoimaan testausta (esimerkkikoodi 3), joka lähettää ryhmien tilatieto kyselyitä 0,5 sekunnin välein ennalta määritellyllä alueella.

```
import os, time
for a in range(1, 7):
  for b in range(1, 8):
    for c in range(1, 256):
      time.sleep(0.5)
      address = (str(a) + "/" + str(b) + "/" + str(c))
      string = ("knxtool groupread ip:localhost " + address)
      print(string)
      os.system(string)
```

Esimerkkikoodi 3. ping_groups.py on pythonilla tehty ohjelma, joka lähettää kyselyitä välillä 1/1/1–6/7/255.

Edellä mainittu pythonilla tehty ohjelma käy ryhmäosoitteet läpi, alkaen osoitteesta 1/1/1 ja lopettaen osoitteeseen 6/7/255. Suurin mahdollinen osoite olisi 31/7/255, mutta haun suorittaminen ottaisi turhan kauan, johtuen 0,5 sekunnin tauosta hakujen välillä.

Vaikka liian nopea hakujen lähettäminen verkkoon ei tuntunut vaikuttavan itse verkon toimintaan, se jätti osan viesteistä lähettämättä ja näin ollen 0,5 sekunnin pituinen tauko valikoitui sopivaksi.

Vastauksien lukemiseen käytin samankaltaista ohjelmaa (esimerkkikoodi 4), ohjelmalle on putkitettu KnxTool-ohjelmiston antama data ja ohjelma tulosti ruudulle vain laitteilta takaisinpäin tulevat vastaus viestit, joiden kahdeksannen tavun neljä ensimmäistä bittiä muodostivat heksadesimaalin ’4’ (ks. Kuva 10). Huomioitavaa että koodi lukee pelkästään kahdeksannen tavun alun joka sisältää ACPI-kentän loppuosan, koodilla voi saada vääriä vastauksia, mutta se on riittävän luotettava tähän tarkoitukseen.

```
#!/usr/bin/env python
import sys
try:
    while True:
        line = raw_input()
        message_type= line[31:32]
        if message_type == "4":
           print (line)
except EOFError:
    pass
```

Esimerkkikoodi 4. Verkkoon lähetettyjä vastausviestejä kuunteleva ohjelma.

Ohjelman tulostamaa dataa ei käytetty sellaisenaan mihinkään, se oli lähinnä testi, jolla todetaan toiminta. Koska tulokset olivat usean ajon perusteella samankaltaiset, voidaan olettaa, ettei ryhmäosoitteita ollut enempää käytettävissä ja ohjelma toimi riittävän luotettavasti.

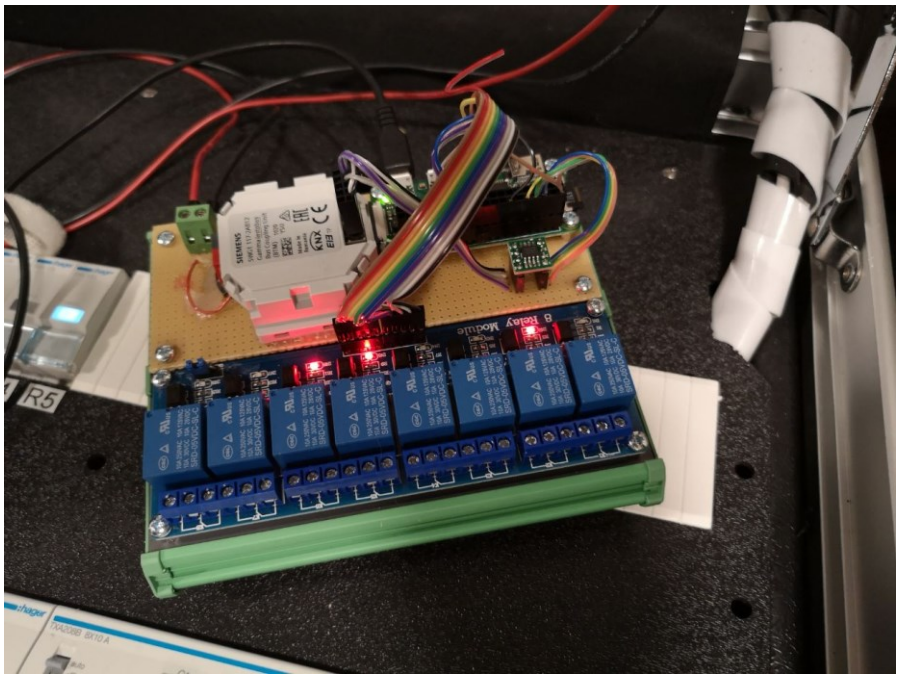
```
pi@raspberrypi: ~/knx
pi@raspberrypi:~/knx $ pico capture_response.py
pi@raspberrypi:~/knx $ knxtool vbusmonitor1 ip:localhost | python capture_response.py
L_Busmon: BC 11 02 09 01 E1 00 40 F9 :L_Data low from 1.1.2 to 1/1/1 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
L_Busmon: BC 11 04 09 02 E1 00 40 FC :L_Data low from 1.1.4 to 1/1/2 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
L_Busmon: BC 11 04 09 03 E1 00 40 FD :L_Data low from 1.1.4 to 1/1/3 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
L_Busmon: BC 11 02 09 06 E1 00 40 FE :L_Data low from 1.1.2 to 1/1/6 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
L_Busmon: BC 11 02 09 07 E1 00 40 FF :L_Data low from 1.1.2 to 1/1/7 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
L_Busmon: BC 11 04 09 08 E2 00 40 00 F5 :L_Data low from 1.1.4 to 1/1/8 hops: 06 T_Data_Group A_GroupValue_Response 00
L_Busmon: BC 11 02 0B 01 E1 00 40 FB :L_Data low from 1.1.2 to 1/3/1 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
L_Busmon: BC 11 02 0B 02 E1 00 40 F8 :L_Data low from 1.1.2 to 1/3/2 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
L_Busmon: BC 11 04 0C 01 E1 00 40 FA :L_Data low from 1.1.4 to 1/4/1 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
L_Busmon: BC 11 02 09 01 E1 00 40 F9 :L_Data low from 1.1.2 to 1/1/1 hops: 06 T_Data_Group A_GroupValue_Response (small) 00
```

Kuva 10. capture_response.py-ohjelman käynnistäminen sekä ohjelmiston tulostus, kun taustalla ajetaan ping_groups.py-ohjelmisto.

Kuten kuvasta 10 voidaan havaita, yhdeksän ryhmäosoitetta vastasi takaisin ping_groups.py ohjelmiston (ks. esimerkkikoodi 3) lähettämiin kyselyihin ja vain yhdellä vastauksen pituus ei ollut pituudeltaan ns. lyhyt, vaan pituutta indikoiva heksadesimaali on ’2’ ja vastaus yhden tavun pituinen.

5 Raspberry PI Zero-tietokoneen käyttäminen releohjaukseen

Raspberry Pi Zero (kuva 11) valikoitui lopulta palvelinkäyttöön sekä yhteyspisteeksi tablettiohjaukselle. Kokonsa, sähkönkulutuksen sekä alle 13 euron hinnalla se on ihanteellinen projektiin. Konfigurointi ja toiminta on identtinen Raspberry Pi-laitteiston kanssa, ainoastaan sarjaliikenneväylä tuli ottaa pois käytöstä, sillä se on joissain tapauksissa varattu järjestelmälle oletuksena.



Kuva 11. Raspberry Pi Zero, Siemens väyläohjain sekä relekortti DIN-kisko kotelossa.

Mikrotietokoneelle valmistettiin yksinkertainen palvelinohjelmisto, joka kuunteli ja kirjoitti väylälle. Itse ohjelmaa ei avata tarkemmin, sillä se on sekavasti ohjelmoitu. Ohjelma oli riittävän luotettava ja pystyi uudelleenkäynnistämään KNXD-palvelun tilanteista, joissa olin sammuttanut KNX-väylästä sähköt ja palvelinohjelmisto taustalla sammui.

Lyhyesti sanottuna, ohjelmisto kuunteli sille putkitettuja viestejä KnxTool-ohjelmistolta ja toteutti toimintoja saamiensa viestien perusteella. Väylälle kirjoitettaessa kutsutaan os.system() -metodia, jonka avulla käynnistetään KnxTool sopivilla parametreillä.

Väylälle kirjoittamista tarvitaan mm. tablettiohjauksen kysellessä laitteiden tilatietoja. Tabletin kysyessä osoitteen 1/1/51 tilaa, joka on määriteltynä tähän Raspberry Pi -laitteistoon, käyttää python-ohjelmisto os.system() -metodia lähettäessään viestin KNXtool-ohjelmistolla. Komento on muotoa ’os.system("knxtool groupsresponse ip:localhost 1/1/51 1 > /dev/null");’. Tämä lähettää

väylälle tiedon osoitteen 1/1/51 tilasta, joka tässä tapauksessa olisi ’1’, jolloin tabletti pystyy päivittämään sen hetkisen ryhmäosoitteen tilan ruudulle.

6 Arduinon käyttäminen ohjauspainikkeiden lukemiseen

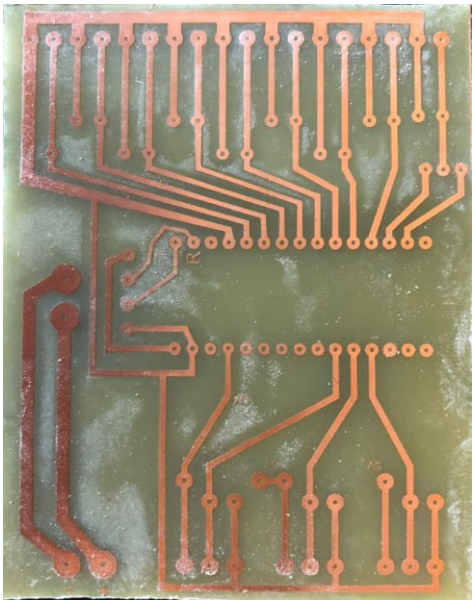
Arduinolle on kirjoitettu useampia kirjastoja KNX-ohjausta varten, projektia varten kokeiltiin Arduino Nano-, UNO- sekä Leonardo-alustoja, lopulliseksi laitteeksi muodostui kiinalainen kopio Nanosta, sillä Kiinasta tilattuna sen hinta on 3–5 euroa ja se on kokonsa puolesta sopiva.

Arduinon Nanon kanssa aloitettiin tulostettavalla maskilla (ks. kuva 12), jota käytettiin piirilevyn valotuksessa. Piirilevyn suunnittelussa tapahtui virhe, Arduinon käyttöjännite ohjattiin Arduinon regulaattorille eikä 5 V:n nastaan.

Kuva 12. Piirilevyn valoitusmaski.

Kuvassa 13 näkyvälle piirilevylle on suunniteltu yhdeksän digitaalista sisääntuloa. Koodia muutettiin sen mukaan, oliko kytkettynä impulssinappeja vai tilaa vaihtavia kytkimiä sekä lisäksi kolme analogista porttia, joihin pystyi laittamaan lämpötila sensorin tai potentiometrin. Ohjelmisto oli alun perin kirjoitettu Leonardo-alustalle, jossa oli käytössä optoerotin sekä oma virransyöttö, sillä testatessa verkkoliikenne toimi hyvin. Siirrettäessä ohjelma Zero-alustalle, poistettiin samalla optoerottin ja virransyöttö otettiin KNX-väylästä. Siirron jälkeen lämpötilamittaus lopetti toimimisen ja syyksi paljastui lopulta Arduinon käyttöjännite,

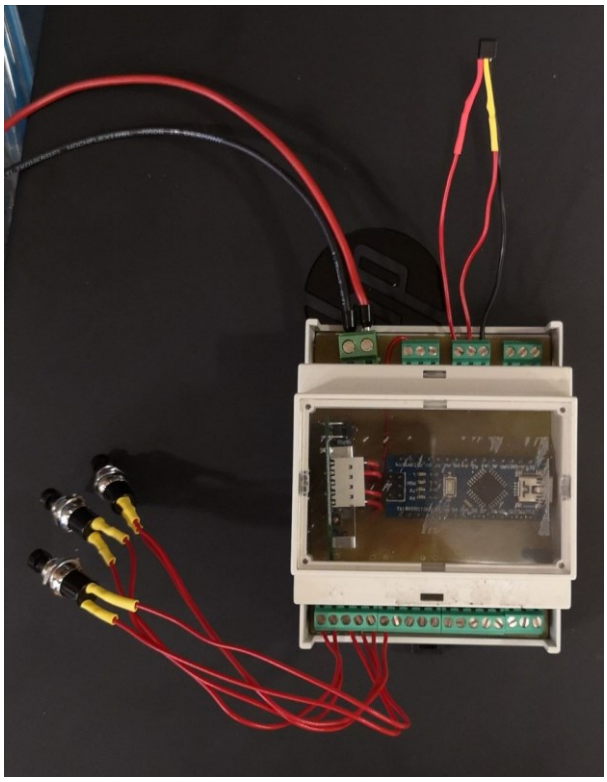
joka oli laskenut liian alhaiselle tasolle. Ohittaen regulaattorin ja kytkettäessä jännite suoraan Arduinon 5 V:n jännite pinniin, aiheutui enää 200 mV jännitteen alenema. Lämpötila anturin ulostulojännitteen ollessa 0–1,75 V mikä vastaa lämpötila-aluetta -50–125 C°, on 0,2 voltin pudotus referenssijännitteessä riittävä haittaamaan mittausta.



Kuva 13. Arduino-mikrokontrollerille valmistettu piirilevy.

Jos käytössä on Arduino ATmega328P-piirisarja, komennolla ’analogreference(internal)’ pystyy analogiaportin referenssijännitteen asettamaan 1.1 V:n tasolle. Samalla muutos rajoittaa TMP36-sensorin lämpötilamittauksen alueelle -50–59 C°. Siemensin väylämuuntimessa on myös 20 V:n ulostulo, jota pystyy kuormittamaan 25mA virralla, mutta se vaatisi erillisen DC-DC-muuntimen eikä piirilevylle ollut varattu riittävästi tilaa.

Kuvassa 14 on lopullinen versio painonappisovellutuksesta rakennettuna DIN-kiskoon asennettavaan koteloon. Ainoastaan ohjelmointi on hankala toteuttaa, johtuen Nano-alustan irroittamisen hankaluudesta.



Kuva 14. Arduino asennettuna DIN-koteloon, alareunassa nappien lähdöt, yläpuolella KNX-väylä sekä lämpötila anturi.

6.1 Arduinon ohjelmisto

Arduinossa käytettiin pohjana Git Hubista löytyvää Arduino-tpuart-kirjastoa. Kirjasto sisältää huomattavan määrän toimintoja viestien lähettämiseen, vastaanottamiseen sekä paljon esimerkkejä. Työssä käydään lyhyesti lävitse liitteen 2 esimerkkikoodin tärkeimmät ominaisuudet.

KnxTpUart knx(&Serial1, "15.15.20"); Määritellään kirjastolle Serial1 -niminen sarjaportti käytettäväksi KNX-väylän kanssa kommunikointiin sekä asetetaan KNX-kirjastolle fyysinen osoite 15.15.20. Koodi on tehty toimivaksi Arduino MEGA-alustalla, jossa on useita sarjaportteja. Käyttäessä Nano tai Uno-alustoja, tulee sarjaportiksi määritellä ensimmäinen sarjaportti, joka kulkee nimellä Serial ja samalla tulee poistaa nykyiset debug-tulostukset.

void setup() -funktion sisällä oleva Serial1.begin(19200, SERIAL_8E1) alustaa Serial1-sarjaportin asettamalla nopeudeksi 19200 bps sekä antaa sille tiedonsiirtoon asetukset kahdeksan bittiä pariteettibitin kanssa sekä yksi stop bitti. Myös tämä sarjaliikennelaite tulee nimetä Serial -nimiseksi, jos laitteistona on Nano tai Uno.

void loop() on yksinkertainen funktio, jonka sisälle on kirjoitettu koodi tarkkailemaan digitaaliportin 32 tilaa. Huomattuaan portin tilan muuttuvan LOW-tilaan sytyttää ohjelmisto ledin ja lähettää kerran viestin KNX-verkkoon. Viestin lähetys toteutetaan komennolla knx.groupWriteBool("0/0/3", !onSent), arvo on eri kuin edellisellä kerralla lähetettiin, viestiä ei myöskään lähetetä kuin kerran. Napin vapautuessa painalluksesta ja virtapiirin avautuessa, sisääntulo palautuu pullup-vastuksen johdosta korkeamman potentiaalin tilaan ja lähetetyn viestin indikaattori resetoidaan sekä ledi sammutetaan. Tämän jälkeen viesti voidaan lähettää uudestaan nappia seuraavan kerran painaessa.

6.2 Muutoksia ohjelmistoon

Esimerkkikoodi toimii suoraan, mutta on syytä havaita, ettei koodissa ole huomioitu ns. Bounce-efektiä [15], joka tyypillisesti ilmenee nappia painaessa. Napin mekaniikka saattaa aiheuttaa useita kytkentöjä yhtä painamista kohden ja ohjelmistossa tulisi olla tarkistus asian varalle, jotta vältyttäisiin ylimääräisiltä viesteiltä. Esimerkkikoodista poiketen asetin osoitteet ja painonappien pinnit yksinkertaiseen taulukkoon (kuva 15), jota luetaan silmukassa.

```
#define SWITCH 0
#define IMPULSE 1
#define buttonTime 100
String  OutAddrBool[]        = { "1/1/2"  , "1/1/4"   , "1/1/51"  , "1/1/52"  , "1/1/53"  };
int     digitalIn[]          = { 2        , 3         , 4         , 5         , 6         };
int     digitalInStatus[]    = { -1       , -1        , -1        , -1        , -1        };
int     digitalInPullup[]    = { HIGH     , HIGH      , HIGH      , HIGH      , HIGH      };
int     digitalInType[]      = {IMPULSE   , IMPULSE   , IMPULSE   , IMPULSE   , IMPULSE   };
int  outAddrBoolInverted[]   = {true      , true      , true      , true      , true      };
unsigned long lastChange[]   = {millis()  , millis()  , millis()  , millis()  , millis()  };
int outAddrBoolSended[20]    = {0};
int outAddrBoolLastSended[20]   = {0};
```

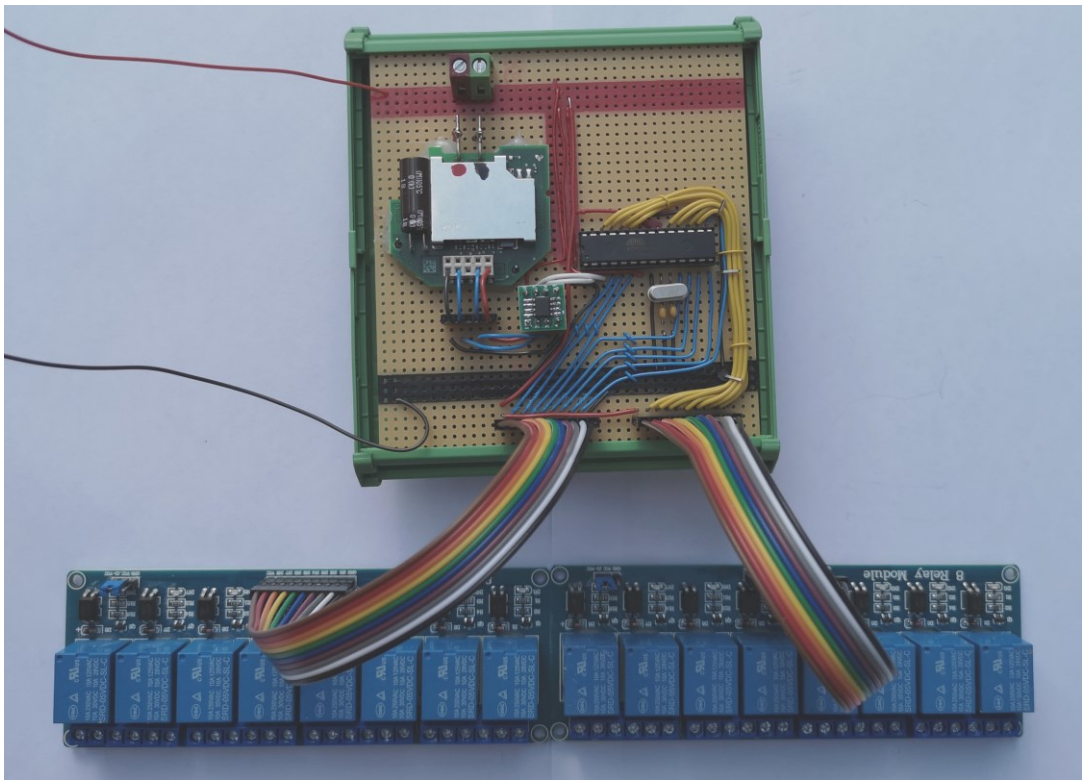
Kuva 15. Painonappien määrittely sekä ohjattavat osoitteet.

Ohjelmistossa oleva FOR-silmukka käy yksitellen digitaalipinnien tilan lävitse. Painonappia painaessa pinnin tilan laskee LOW-tasolle, ohjelmisto tarkistaa pinnille määritellyn ryhmäosoitteen sekä mitä sinne on viimeksi lähetetty. Jos kyseessä on BOOL-muotoinen data, se lähettää vain käänteisen arvon. Valmista ohjelmistoa ei sisällytetä opinnäytetyöhön, sillä kirjaston mukana tulee hyvät esimerkki koodit ja on suositeltavaa kirjoittaa ohjelmisto itse ajan tasalla olevilla kirjastoilla.

Painonappisovellus toimi kohtuullisesti Arduinossa, mutta laite ei pystynyt lukemaan väylältä tietoa takaisinpäin, joten painonapit toimivat hyvin vain, jos samaa osoitetta ei ollut ohjattu toisaalta.

7 Atmega328:n käyttö releohjaukseen

Relelogiikan tekemiseen valikoitui lopulta Atmega328p-piiri, Siemensin väyläohjain sekä optoerotin, sillä Kiinasta tilattuja Nanoja en onnistunut saamaan toimimaan kunnolla sarjaliikenteen osalta, kuten ohjauspainikkeiden kanssa tuli ilmi. Atmega328P-piirin kanssa mitään edellä mainittuja ongelmia ei ollut ja tiedonsiirto toimi hyvin. Piirilevyä ei valmistettu kiireen johdosta, vaan kytkentä on koottuna kuvassa 16 näkyvälle reikälevylle, joka mahdollistaa nopeat muutokset kytkennöissä. Lopullinen kytkentä toteutettiin liitteen 3 mukaisesti mutta ulkoisella 5 V:n jännitelähteellä.



Kuva 16. Atmega328-mikropiiri, Siemens-väylämuunnin, optoerotin sekä relekortit.

Releohjaus vaatii ulkopuolisen jännitelähteen, kuten Raspberry PI'tä käytettäessä, sillä KNX-väylälaitteita ei ole suunniteltu kuluttamaan kuin 10 mA laitetta kohden ja Arduinoon kytketyt 16 kappaletta 5 V:n releitä, kuormittaa virtalähdettä noin yhden ampeerin virralla kaikkien releiden ollessa aktiivisena.

Ohjelmistona käytettiin samaa kirjastoa kuin painonappien kanssa ja ohjelmiston tarvitsemat muutokset lisättiin kirjaston mukana tulleeseen ReceiveKNXTelegrams.ino-esimerkkikoodiin.

Releiden ja osoitteiden tiedot tallennettiin taulukon sijaan struct-tietorakenteeseen, joka koostui relenumerosta, Atmegan pinnin numerosta johon rele on kytkettynä, KNX-ryhmäosoitteesta sekä portin senhetkisestä tilatiedosta.

```
struct data_struct {
  int relay;          //Relay number
  int port;           //atmega port number / name
  char *addr;         //KNX address
  int value;          //Current relay status (default all off)
} fields[] {
  {1, 2, "1/1/10", 0}, //connector 1, pin 2
  {2, 3, "1/1/11", 0},
  {3, 4, "1/1/12", 0},
  {4, 5, "1/1/13", 0},
  {5, 6, "1/1/14", 0},
  {6, 7, "1/1/15", 0},
  {7, 8, "1/1/16", 0},
  {8, 9, "1/1/17", 0}, //connector 1, pin 9

  {9, 10, "1/1/18", 0}, //connector 2, pin 2
  {10, 11, "1/1/19", 0},
  {11, 12, "1/1/20", 0},
  {12, 13, "1/1/21", 0},
  {13, A0, "1/1/22", 0}, //remember that you have led also in
  {14, A1, "1/1/23", 0},             // this port on nano/uno
  {15, A2, "1/1/24", 0},
  {16, A3, "1/1/25", 0} //connector 2, pin 9
};
```

Kuva 17. Releiden määritteleminen koodissa.

Tiedot annetaan käynnistymisessä kirjastolle kuvassa 18 olevalla for-silmukalla, KNX-osoite määritellään addListenGroupAddress() funktiolla, pinnien tilan asettaminen ohjattavaksi (Output) ja lopuksi asetetaan releiden tila.

```
//Initialize ports and addresses
for (auto& field : fields) {
  knx.addListenGroupAddress(field.addr);  //Initialize address
  pinMode(field.port, OUTPUT); //Sets the Pin to output
  digitalWrite(field.port, !field.value); //setup pins: lastValue 1 sets pin to 0 (activates it)
  delay(50);
}
```

Kuva 18. Ryhmäosoitteiden määrittäminen KNX-kirjastolle.

Releiden ohjaukseen on määritelty serialEvent() funktiossa silmukka, jossa verrataan Struck-rakennelman sisältämiä osoitteita viimeisen viestin osoitteeseen. Jos osoite täsmää ja viestityyppi on muotoa KNX_COMMAND_WRITE (ks. Taulukko 3, APCI koodi 0010), muutetaan releen tila viestin lähettämän tiedon mukaiseksi. Kyseinen tapa koodata ei ole mitenkään esimerkillinen, tarkoituksena oli vain saada nopeasti koodia aikaan, jotta pääsin testaamaan laitteistoa.

8 Hyödyllisiä komentoja KNXD-ohjelmiston käyttöön

sudo systemctl restart knxd.service:

Käynnistää palvelun uudelleen, sarjaliikenteen katketessa /etc/systemd/system/knxd.service sijaitsevaan tiedostoon voi lisäksi asettaa asetukset Restart=always sekä RestartSec=10.

knxtool vbusmonitor1 ip:localhost:

Monitoroidaan liikennettä väylällä.

knxtool vbusmonitor2 ip:localhost:

Monitoroidaan liikennettä väylällä, tulostetaan ainoastaan heksadesimaali viesti.

knxtool off ip:localhost 2/0/7:

Kytkee ryhmän 2/0/7 pois päältä.

knxtool on ip:localhost 2/0/7:

Kytkee ryhmän 2/0/7 päälle.

knxtool groupwrite ip:localhost 1/2/3 ff:

Lähettää yhden tavun mittaisen viestin ’ff’ osoitteeseen 1/2/3.

9 Yhteenveto

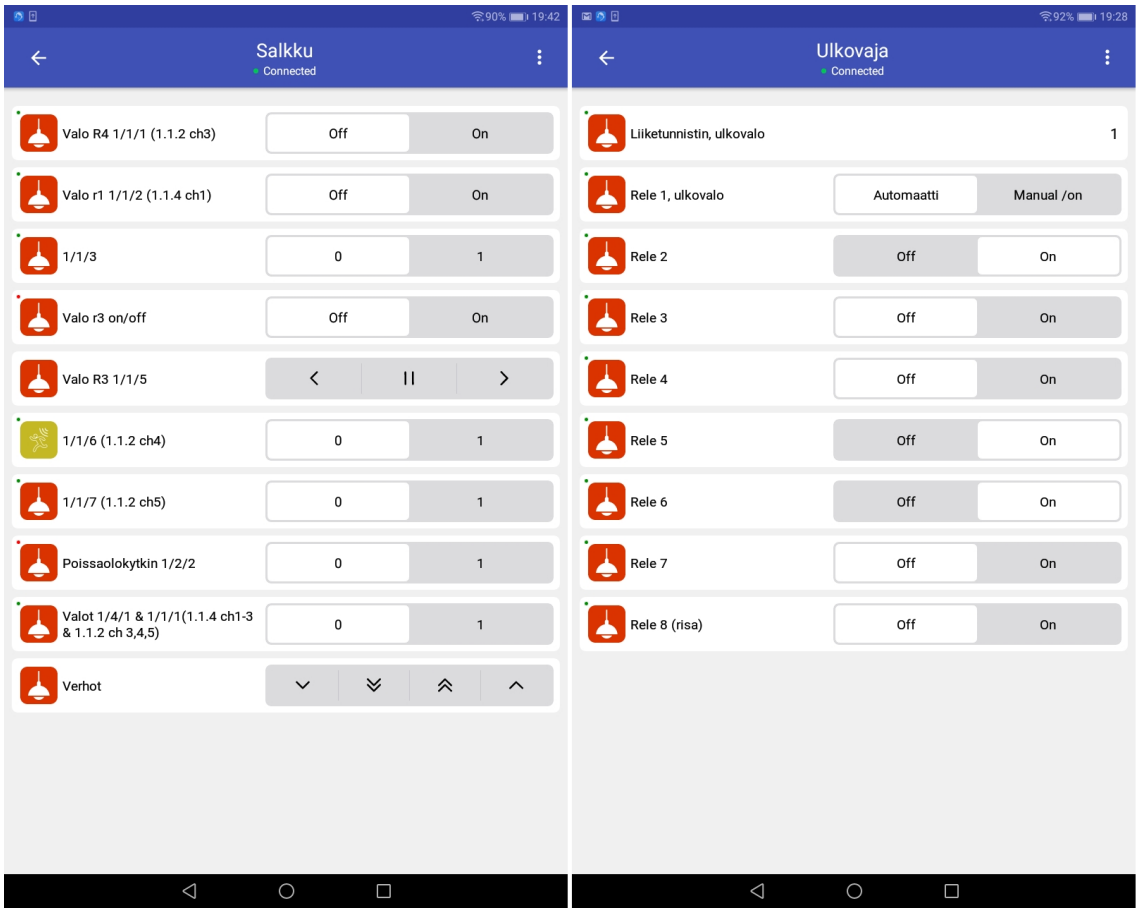
KNX-salkun palautuksen jälkeen jäi koululta lainaan ainoastaan virtalähde, jonka ympärille rakensin yksinkertaisen KNX-verkon, ideana oli myös oman virtalähteen tekeminen, tai tarkemmin ottaen filtterin rakentamista valmiiseen virtalähteeseen, sillä halpoja tasajännitevirtalähteitä saisi vanhoista laitteista ilmaiseksi. Filtterin teko olisi ollut projektina kiva, mutta sen toimintaan perehtyminen ei ollut työn kannalta järkevää. Tästä syystä testit salkun palautuksen jälkeen tuli tehtyä kaupallisella virtalähteellä, Raspberry PI-minitietokoneella sekä kahdella mikrokontrollerilla, joista toinen toimi sisääntulona painonapeille ja toinen ohjasi releitä. Lopullinen looginen KNX-verkko jäi liitteen 1 kaltaiseksi.

Ikävänä puolena laitteistoiden ohjelmoinnista tuli turhan hankalaa, sillä kytkimien toteuttamia toimintoja tai niiden ohjaamia osoitteita ei voi vielä uudelleen ohjelmoida KNX-verkon ylitse, vaan laitteessa oleva Arduino Nano tulee irrottaa kotelosta ja ohjelmoida tietokoneessa USB-liitynnän avulla. Atmega328:n ohjaamien releiden kanssa on sama puute, mutta sen osoitteiden vaihto ei ole tarpeellista muuten kuin esteettisistä syistä tai jos on useampia linjoja ja vaihtaa laitetta linjasta toiselle.

Koska projektissa ei käytetty ETS5-ohjelmistoa, rajoittaa se asennuksen kohteisiin, joissa maksimissaan 255 fyysistä osoitetta. Ohjelmistoa tarvitaan linjayhdistimen ohjelmointiin, mikä yhdistää linjat eikä sitä pysty tekemään ilman kaupallista ohjelmistoa. Toisaalta koska tämä projekti nojaa suurelta osin itse tekemiseen, on erittäin epätodennäköistä, että tarvitsisi yli 255 fyysistä KNX-osoitetta.

Projektin alussa oli ideana kirjoittaa etäkäyttöohjelmisto Android-pohjaiselle tabletille, mutta kokeillessa Google play-kaupassa saatavilla olevia ohjelmistoja, huomasin niiden toimivan kohtuullisesti ja ajatus ohjelmiston tekemisestä loppui samalla, ei ollut järkeä yrittää tehdä parempaa. Kokeiltuani noin kymmentä erilaista KNX-käyttöliittymää Easy KNX Lite-ohjelmisto vaikutti parhaiten toimivalta ja ostin siitä maksullisen version. Vaikka ohjelmisto olikin turhan simppeli ja

varmasti monet vieroksuvat sen vanhahtavaa käyttöliittymää (kuva 19), ostohetkellä ohjelmiston hinta oli 15 euroa, mikä oli toimivasta ohjelmistosta kohtuullinen hinta.



Kuva 19. Vierekkäin salkun ja Raspberry PI:n ohjaukset Easy KNX-ohjelmistossa.

Lähdeluettelo

1 Sesko: Sähkötekniikan peruskäsitteet. 2020. Verkkoaineisto. Sesko. https://www.sesko.fi/files/79/Peruskasitteet_osa1_jannitteet_2020.pdf Luettu 29.8.2021

2 KNX Perusperiaatteet. 2006. KNX Association cvba

3 KNX security. 2015. Verkkoaineisto. KNX Association http://knx.fi/doc/esitteet/KNX-Security-Position-Paper_en.pdf Luettu 29.8.2021

4 Raspberry Pi Documentation. Verkkoaineisto. https://www.raspberrypi.org/documentation/computers/os.html#gpio-and-the-40-pin-header Luettu 29.8.2021

5 Gustavo Andrés Lizio: Easy KNX-ohjelmiston tekijä.

6 KNX System arguments. Verkkoaineisto. KNX Association http://knx.com.ua/attachments/article/132/KNX-basic_course_full.pdf Luettu 29.8.2021

7 KNX ETS5 and group addresses. 2020. Verkkoaineisto. ABB. https://library.e.abb.com/public/02d03f979e59478c845af18759dd2030/OnlineTraining-KNX-ETS-Groupaddresses-April-2020_PR_EN_V1-0_9AKK107680A5823.pdf Luettu 23.9.2021

8 Egor Litvinov 2018: Industrial Cybersecurity. 2018. Verkkoaineisto. Kaspersky lab https://ics.kaspersky.com/media/ics-conference-2018/Egor-Litvinov-Cybersecurity-for-building-management-systems-En.pdf Luettu 24.9.2021

9 Control Field. 2018. Verkkoaineisto. Knx / Christophe Parthoens https://support.knx.org/hc/en-us/articles/115003188429-Control-Field/ Luettu 24.9.2021

10 Payload. Verkkoaineisto. Knx / Christophe Parthoens. https://support.knx.org/hc/en-us/articles/115003188529-Payload Luettu 24.9.2021

11 Serial Data Transmission and KNX Protocol. Verkkoaineisto. KNX Association https://cdn.chipkin.com/assets/uploads/imports/resources/Serial%20Data%20Transmission%20and%20KNX%20Protocol%2005_Serial%20Data%20Transmission_E0808f.pdf Luettu 23.8.2021

12 KNX-koulutusympäristö. 2015. Verkkoaineisto. Chun Leung https://urn.fi/URN:NBN:fi:amk-2015112217092

13 KNX topology. Verkkoaineisto. ABB. https://new.abb.com/low-voltage/products/building-automation/product-range/abb-i-bus-knx/system/installation Luettu 11.10.2021

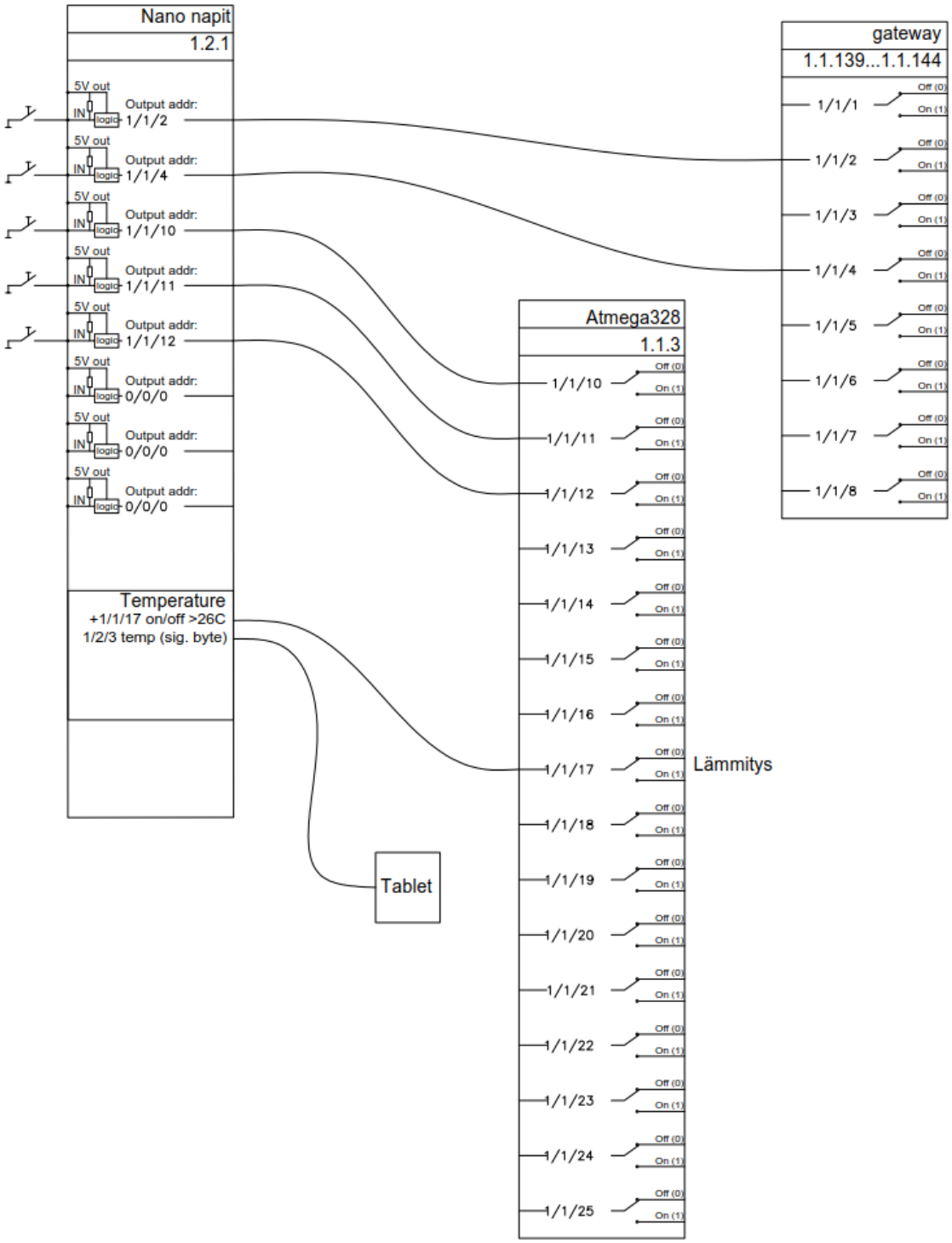
14 Line coupler S. Verkkoaineisto. Theben. https://www.theben.de/ocsmedia/optimized/full/o5685v42%20Linienkoppler%20S%20KNX%20-%20Hand%20book.PDF Luettu 11.10.2021

15 Debounce https://www.arduino.cc/en/Tutorial/BuiltInExamples/Debounce

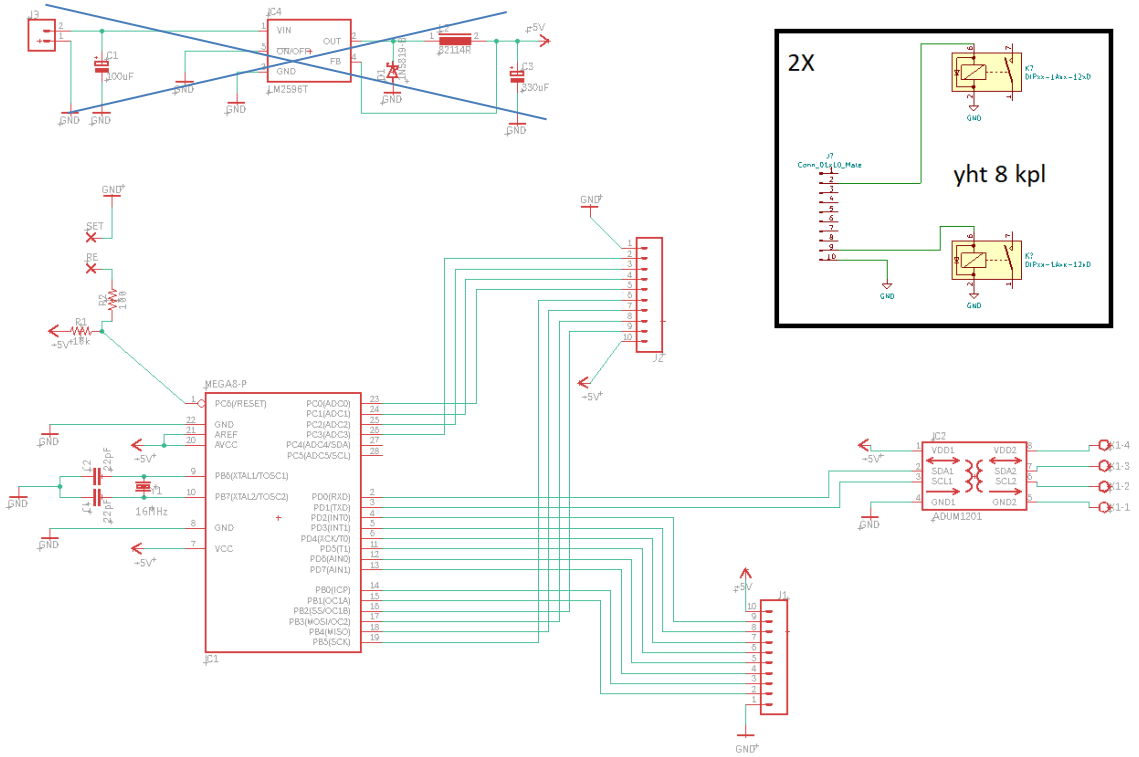
16 Siemens 5wg1 117-2ab12. verkkoaineisto. Konnekting. https://www.konnekting.de/konnekting-lernen/l1-knx-mit-arduino/ Luettu 8.3.2022

17 Raspberry PI. verkkoaineisto. Raspberry Pi. https://www.raspberrypi.com/documentation/computers/os.html Luettu 3.3.2022

Liite 1. Testiverkon looginen kytkentä



Liite 2. groupWrite.ino esimerkkitiedosto

```
// File: GroupWrite.ino
// Author: Daniel Kleine-Albers (Since 2012)
// Modified: Thorsten Gehrig (Since 2014)
// Modified: Mag Gyver (Since 2016)
// Test constellation = ARDUINO MEGA <-> 5WG1 117-2AB12

#include <KnxTpUart.h>
// Initialize the KNX TP-UART library on the Serial1 port of ARDUINO
MEGA
KnxTpUart knx(&Serial1, "15.15.20");

// Define input pin
int inPin = 32;

// Remember if we have sent on current keypress
boolean haveSent = false;

// Remember if we sent ON last or OFF
boolean onSent = false;


void setup() {
  pinMode(inPin, INPUT);
  digitalWrite(inPin, HIGH); // Turn on pullup
  pinMode(13, OUTPUT); // LED
  digitalWrite(13, LOW);
  Serial.begin(9600);
  Serial.println("TP-UART Test");
  Serial1.begin(19200, SERIAL_8E1);
  Serial.print("UCSR1A: ");
  Serial.println(UCSR1A, BIN);
  Serial.print("UCSR1B: ");
  Serial.println(UCSR1B, BIN);
  Serial.print("UCSR1C: ");
  Serial.println(UCSR1C, BIN);
  knx.uartReset();
}

void loop() {
  if (digitalRead(inPin) == LOW) { // Button is pressed
    digitalWrite(13, HIGH);
    if (!haveSent) {
      // Send the opposite of what we have sent last
      bool success = knx.groupWriteBool("0/0/3", !onSent);
      Serial.print("Successfully sent: ");
      Serial.println(!onSent);

      onSent = !onSent;
      haveSent = true;
    }
  }
  else {
    digitalWrite(13, LOW);
    haveSent = false;
  }
}
```

Liite 3. Relelogiikan kytkentäkaavio