Tonino Casagrande

Raspberry Pi CAN-väyläohjaimena

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Automaatiotekniikka

Insinöörityö

21.5.2012

Metropolia

Tekijä(t) Otsikko Sivumäärä Aika	Tonino Casagrande Raspberry Pi CAN-väyläohjaimena 19 sivua + 2 liitettä 21.5.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Automaatiotekniikka
Suuntautumisvaihtoehto	Kappaletavara-automaatio
Ohjaaja(t)	Lehtori Antti Liljaniemi

Insinöörityössä tutkittiin Raspberry Pi:n toiminnallisuutta CAN-väyläohjaimena ja lopuksi tutkittiin sen käyttämistä yhdessä CANopen-protokollaan pohjautuvan I/O-moduulin kanssa. Työn alussa perehdyttiin CAN-väylän teknisiin tietoihin ja CAN-viestin rakenteeseen. Samalla tutustuttiin CANopen-protokollaperheen eri protokollien toimintaan ja niiden ominaisuuksiin.

Työssä rakennettiin Raspberry Pi:n ja CAN-väylän välille väylämuunnin, joka perustuu täysin muiden harrastajien kokeiluihin ja suunnitteluun. Tämä väylämuunnin kääntää Raspberry Pi:n tukeman SPI-väylän CAN-väyläksi, SPI:n toimiessa taustalla rajapintana.

Raspberry Pi:n ja I/O-moduulin välinen kommunikointi yritettiin toteuttaa avoimeen lähdekoodin perustuvaan Libcanopen-ohjelmakirjaston avulla, joka sisältää kaikki tarvittavat CANopen-protokollat.

Vaikkakaan työssä ei onnistuttu kommunikoimaan CANopen-protokalla I/O-moduulin kanssa. Tämä ei kuitenkaan tarkoita etteikö kommunikaatio olisi ollut mahdollista toteuttaa, mutta se olisi vaatinut suunniteltua enemmän aikaa asian perehtymiseen. Työssä kuitenkin onnistuttiin lähettämään CAN-viestejä väylään sekä monitoroimaan väylän liikennettä Raspberry Pi:n avulla.

Avainsanat	Raspberry Pi, CAN, CANopen

Author(s) Title Number of Pages Date	Tonino Casagrande Raspberry Pi as CAN controller 19 pages + 2 appendices 21 May 2015
Degree	Bachelor of Engineering
Degree Programme	Automation technology
Specialisation option	Manufacturing automation
Instructor(s)	Antti Liljaniemi, Lecturer

The aim of this bachelor's thesis was to research Raspberry Pi's suitability to work as CAN bus controller and build a working communication between it and the CANopen protocol based I/O module. The thesis was started by examining CAN bus technical information and its message format. And also studied the CANopen protocol family's content and their features.

To be able transmit data between Raspberry Pi and CAN bus, it was necessary to build a bus converter that can convert Raspberry Pi's supported SPI bus to the CAN bus. This bus converter is based on the planning and creations of other hobbyists.

There was an attempt to create communication between Raspberry Pi and I/O module by using the libcanopen library, which is an open-source library for CANopen. Libcanopen contains all the necessary communication protocols that were needed during this study.

As a result of this study it was not possible to successfully transmit CANopen protocol data between Raspberry Pi and I/O. However, this does not mean that such communication would not be possible to execute, but it would have required more time to get familiar with the system. In conclusion, it turned out to be possible to send CAN messages to the bus and monitor its traffic.

Keywords	Raspberry Pi, CAN, CANopen

Sisällys

Liitteet

Liite 1. SPI/CAN-väylämuunnin

Liite 2. Raspberry pi GPIO layout

Lyhenteet

CANH *CAN High*. on väyläsignaaleista se, minkä dominanttia tasoa vastaa resessiivistä tasoa korkeampi jännite.

CANL *CAN Low*. on väyläsignaaleista se, minkä dominanttia tasoa vastaa resessiivistä tasoa matalampi jännite.

SPI *Serial* Peripheral Interface. SPI-väylä on synkronoitu sarjaväylä, missä tieto liikkuu bitteinä kahden tai useamman laitteen välillä.

CiA *CAN in Automation*. Organisaatio joka tuottaa informaatiota ja markki nointimateriaalia CAN tekniikasta ja CAN tuotteista.

OSI *Open Systems Interconnection*. Kuvaa tiedonsiirtoprotokollien yhdistelmän seitsemässä kerroksessa

RPI *Raspberry Pi.* Edullinen yhden piirin tietokone, jonka on suunnitellut Raspberry Pi Foundation.

1 Johdanto

Edullisten yhden piirin tietokoneiden yleistyessä kuluttajamarkkinoilla, niiden soveltaminen eri käyttötarkoituksiin on kasvanut vauhdilla harrastajien keskuudessa. Tunnetuin tällainen tietokone on Raspberry Pi. Tässä opinnäytetyössä on tarkoitus tutkia Raspberry Pi B-mallin soveltuvuutta kommunikoimaan Beckhoffin CANopen-protokollaan pohjautuvaa LC5100 I/O-moduulin kanssa. Jotta tämä olisi edes mahdollista, tarvitaan rajapinta CAN-väylälle ja erilaisten kirjastojen sekä ajureiden asentamista Raspberry Pi:lle.

Työn alussa selvitetään CAN-väylän tekniset tiedot ja käytettävät viestikehykset, sekä tutustutaan CANopen-protokollaperheen sisältöön ja niiden toiminnallisuuksiin yleisellä tasolla. Ohessa esitellään Raspberry Pi:n tukeman SPI-väylän muuntaminen CAN-väyläksi erillisellä itse rakentamalla väylämuuntimella. Lopussa esitellään vaiheittain Raspberry Pi:lle tehtävät konfiguraatiot ja tutustutaan avoimeen lähdekoodiin perustuvaan libcanopen -ohjelmakirjastoon.

Opinnäytetyön aihe perustuu Raspberry Pi Foundation ylläpitämiltä sivuilta löytyneeseen projektiin, jossa käyttäjä oli toteuttanut LED valo-ohjauksen Raspberry Pi:n ja CAN-ohjainpiirien avulla. Ja näiden välinen kommunikaatio toteutettiin CANopen-protokollaa hyödyntäen. Tästä projektista sitten suunniteltiin tämä opinnäytetyö, joka tehtiin Metropolia ammattikorkeakoulun tiloissa, josta löytyi työssä tarvittavat laitteistot.

2 CAN

Vuonna 1986, Robert Bosch esitteli CAN-väylän (Controller Area Network) SAE kongressissa Detroidissa. Alunperin CAN suunniteltiin henkilöautojen käytettäväksi, mutta pian se otettiin käyttöön myös automaation puolella. Vuonna 1992 käyttäjät ja valmistajat perustavat kansainvälisen CAN in Automation (CiA) -yhdistyksen. CiA julkaisi samana vuonna CAN application layer (CAL) -protokollan, joka toimi pohjana CANopen -protokollalle, joka julkaistiin vuonna 1995. [6, s. xv]

CAN-väylää ei pelkästään käytetä autoteollisuudessa ja automaatiossa, vaan käyttökohteita ovat mm. raskaskalusto, junat, voimalaitokset, hissit, terveydenhuolto, tutkimuslaitteissa, viihde-elektroniikka sekä monia muita kohteita. [10]

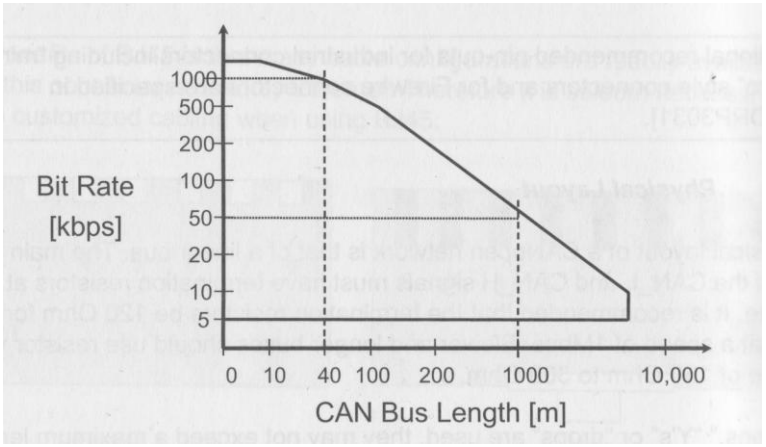
2.1 Toimintaperiaate

CAN-väylässä kaikki solmupisteet ovat kytketty toisiinsa kahdella johtimella, joita kutsutaan CANH ja CANL, ja näin ollen väylän liikenne välittyy kaikille solmupisteille. Jokaisella solmupisteellä on oma sanomatunniste (Node ID), jonka avulla solmupiste pystyy päättämään, että kuuluuko väylässä kulkeva viesti sille. Tällä tavoin solmupisteet voivat jakaa tietoaan muiden kesken yhdellä viestillä. Sanomatunnisteen avulla määritellään myös viestin prioriteetti. Jos useampi solmupiste ryhtyy lähettämään viestiä samanaikaisesti, pienimmän prioriteetin omaavat luopuvat lähetyksestä. [7]

2.2 CAN-väylän tiedonsiirtonopeudet

CAN-väylän tiedonsiirtonopeuteen suurin vaikuttava tekijä on väylän pituus. Kuvasta 1. on nähtävissä väylän pituuden vaikutus tiedonsiirtonopeuteen. Tämä ei kuitenkaan tarkoita että 40 metriä pitkä väylä toimisi 1 Mb/s siirtonopeudella, koska solmupisteiden laatu ja niiden määrä myös vaikuttavat siirtonopeuteen. [6, s.218-219]

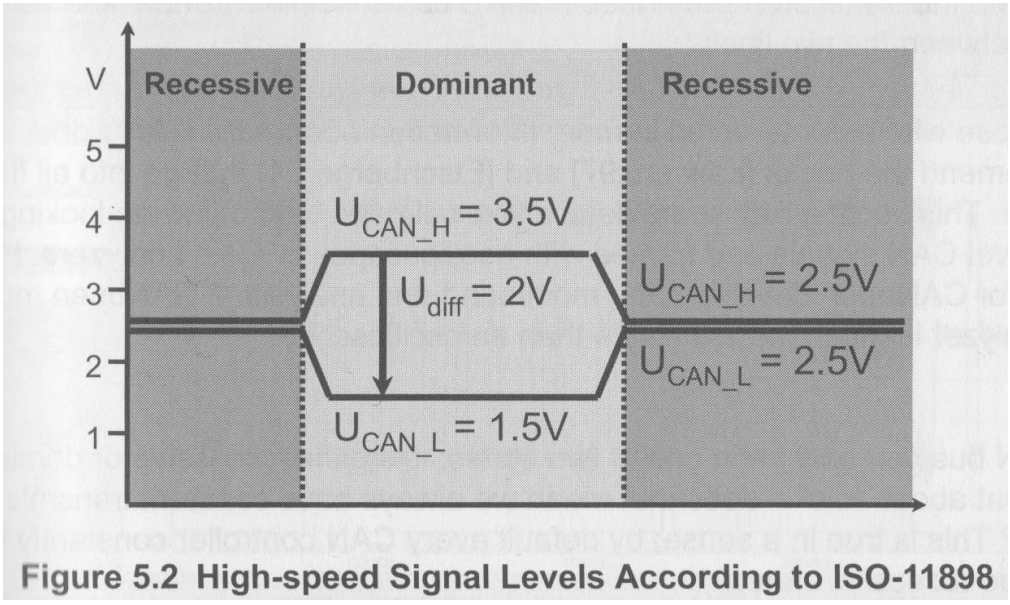
Esimerkiksi tässä työssä käytettävälle Beckhoff LC5100 I/O-moduulille suositellaan käytettäväksi 1 Mb/s siirtonopeudella lyhyempää väylää kuin 20 metriä. [11, s. 15]



Kuva 1. CAN-väylän tiedonsiirtonopeus suhteessa väylän pituuteen [6, s. 218]

2.3 Signaalitasot

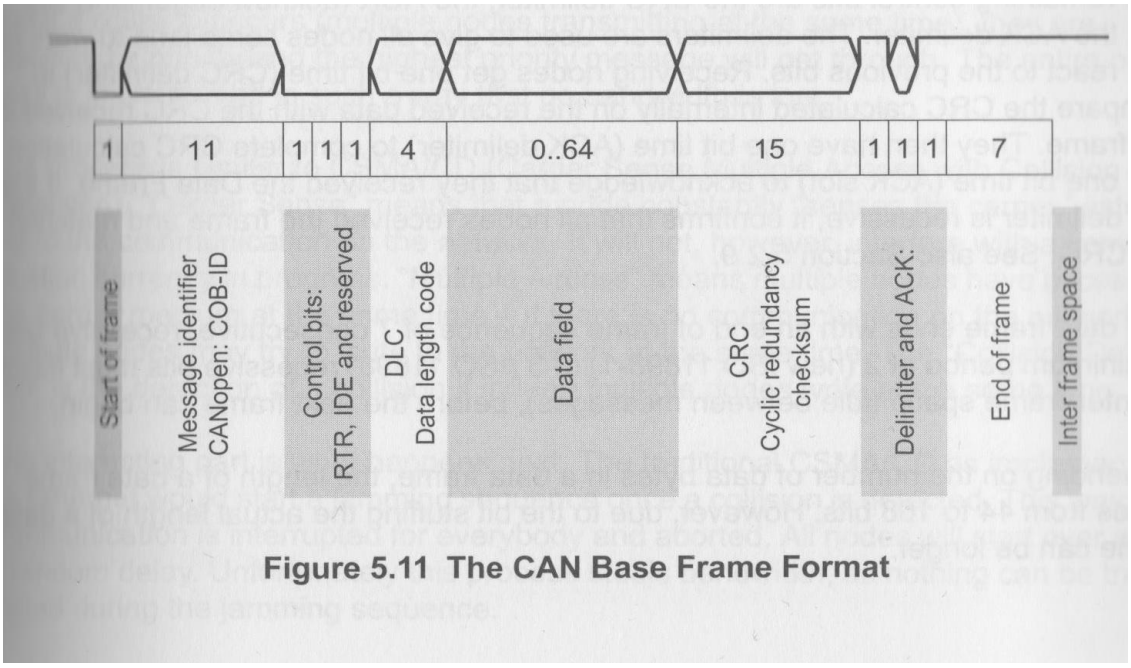
CANH- ja CANL-johtimien välinen jännite-ero määrittelee väylän loogisen tilan. Näistä tiloista käytetään ilmaisua dominantti ja resessiivinen. Dominantissa tilassa CANH-johtimen jännite on 3,5V ja CANL-johdin jännite 1,5V, eli johtimien välinen jännite-ero on 2V ja tämä vastaa loogista tilaa 0. Resessiivisessä tilassa CANH- ja CANL-johtimet ovat samassa 2,5V jännitteessä ja tämä vastaa loogista tilaa 1. Suurin hyöty käytettäessä jännite-eroon perustuvaa signaalia on sen hyvä häiriönsietokyky vastaan elektromagneettista häiriötä. [6, s. 207-209]



Kuva 2. Dominantin ja resessiivisen bitin signaalitasot CANH ja CANL johtimilla. [6, s. 209]

2.4 CAN-viestikehyset

Väylä voidaan konfiguroidan toimimaan kahdella erilaisella datakehysmuodolla, jotka ovat Bosch:in standardisoimat CAN 2.0A ja CAN 2.0B -kehykset. CAN 2.0A -kehys on esitelty kuvassa 3. ja se on yleisesti käytetympi datakehys. CAN 2.0B -kehys on muuten samanlainen kuin CAN 2.0A -kehys, mutta sen ID osuus on 11-bittisen sijaan 29-bittinen.



Kuva 3. CAN 2.0A -viestikehys. [6, s. 219]

Seuraavaksi lähdetään avaamaan CAN 2.0A -datakehyksen sisältöä ja toimintoja kuvan 3. vasemmasta laidasta alkaen. Kaikki datakehykset alkavat dominantilla bitillä ja tätä bittiä kutsutaan: start of frame (SOF).

Seuraavaksi on tulee 11-bittinen sanomatunniste, jonka avulla määritellään myös viestin prioriteetti törmäystilanteessa. CANopen-protokollaa käytettäessä tämä osuus tunnetaan COB ID:nä, josta kerrotaan tarkemmin luvussa 3.2.

Control bits. CAN-verkossa näillä biteillä määritellään viestissä käytettävä kehys. RTR-bitin ollessa dominantti on kyseessä etäkehys. IDE-bittillä määritellään viestikehyksen muoto. Resessiivisessä tilassa viestikehys on 29-bittistä muotoa (ts. CAN 2.0B). RES-bitti on varattu ja se on aina dominantti. Control bitit CANopen-protokollassa ajetaan dominanttiin tilaan ja eikä niitä suositella käytettäväksi CANopen-verkossa.

Data length code (DLC) -kentällä kerrotaan datakehyksen sisältämän datan määrä, mikä voi olla 0..8 tavua.

Datakenttä (Data field) sisältää DLC-kentän ilmaiseman määrän datatavuja (0..64 bittiä).

Cyclic redundancy check (CRC) -kentän avulla tarkastetaan ettei viestissä ole bittivirheitä.

Jäljelle jäävät kolme ohjausbittiä ovat CRC delimiter, acknowledgement (ACK) ja ACK delimiter. CRC delimiter on tarkistussumman erotusbitti, joka on aina resessiivinen. ACK on kuittausbitti, jonka jokainen datakehyksen vastaanottanut solmupiste tulee pakottaa dominanttiin tilaan. ACK delimiter on kuittausbitin jälkeinen erotusbitti, joka tulee olla resessiivisessä tilassa, mikä tarkoittaa että kaikki solmupisteet vastaanottivat viestin ja CRC on hyväksytty.

Datakehys päätetään end of frame (EOF) -kentällä, joka ajaa väylän resessiiviseen tilaan seitsemän bitin ajaksi. Tämän jälkeen väylä on vapaa uusille viestikehyksille. [6, s. 219-220]

Vaikkakin CAN-väylässä liikkuvat viestit ovat useinkin datakehyksiä, kulkee siellä myös virhe-, etä- ja viivekehyksiä. Seuraavaksi esitellään kyseisten kehysten toimintaa solmupisteillä sekä väylässä.

Virhekehys (Error frame) lähetetään kun väylän jokin solmupiste havaitsee viestissä virheen. Virheen havainnut solmu lähettää aktiivisen kehyksen, joka koostuu kuudesta peräkkäisestä dominantista bitistä. Passiivinen kehys koostuu taas kuudesta peräkkäisestä resessiivisestä bitistä, jonka lähettävät virhekehyksen havainneet solmupisteet.

Etäkehyksellä (Remote frame) voidaan pyyttä väylän toiselta solmupisteeltä datakehyksen. Kehys ei sisällä lainkaan dataa, vaan DLC (Data lenght Code) osoittaa lähetettävän datakehyksen sisältämien datatavujen määrän.

Viivekehyksellä (Overload frame) solmupiste voi varata väylän yhden tai useamman kehyksen ajaksi. Tämä kehys on tarkoitettu hitaille solmupisteille, jotka voivat pitää väylää varattuna edellisen vastaanotetun viestikehyksen käsittelyn ajan. Viivekehys voidaan lähettää vain ainoastaan välittömästi datakehyksen jälkeen ja se koostuu kuudesta dominantista bitistä. [5, s. 11]

2.5 Terminointi

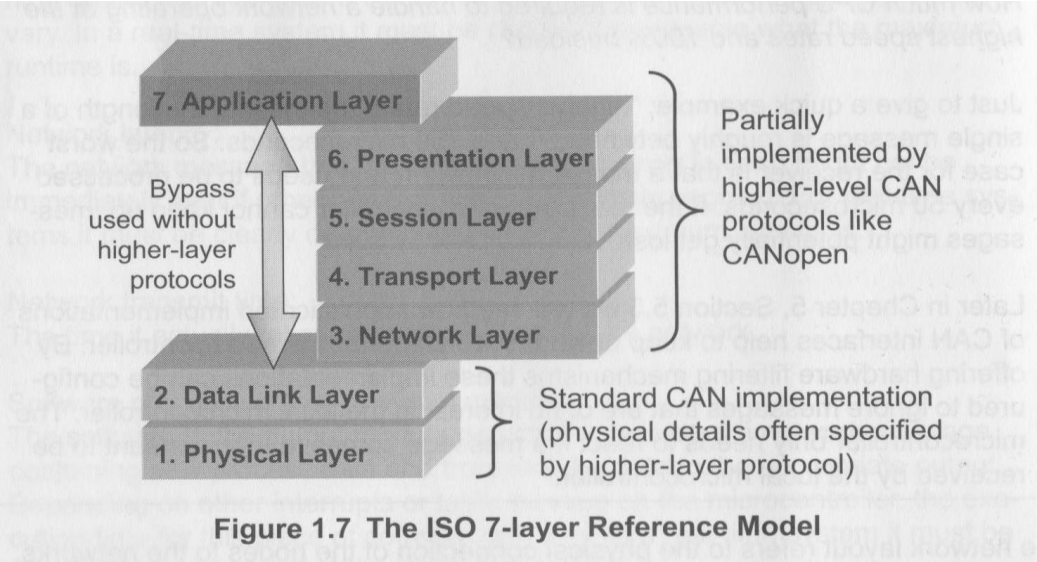
Vaimentaaksemme signaalien reunojen aiheuttamaa heijastusta, pitää väylä terminoida molemmista päistä. 1 Mb/s siirtonopeudella riittävä terminaattori on CANH:n ja CANL väliin kytkettävä 120 ohmin päätevastus. Käytettäessä pidempää väylää ja hitaampaa siirtonopeutta tulisi käyttää päätevastusta 150-300 ohmin väliltä. [5, s. 8][6, s. 217]

3 CANopen

CANopen on kansainvälisesti standardisoitu CAN-standardiin pohjautuva korkeamman luokan protokolla, joka on suunniteltu etenkin sulautetuille ohjausjärjestelmille. CANopen ei ole pelkästään yksittäinen protokolla, vaan TCP/IP:n tavoin protokollaperhe. Vaikkakin CANopen on suunniteltu varta vasten toimimaan CAN-väylässä, sitä voidaan käyttää myös muissa tietoliikennetekniikoissa, kuten I2C ja ethernetissä. [6, s. 65] [8, s. 1]

3.1 CAN ja CANopen OSI-mallissa

Open Systems Interconnection (OSI) -malli on standardi verkkokommunikaatio mallinnukselle, joka on jaettu seitsemään kerrokseen ja käsittää alueet aina fyysisestä kerroksesta sovelluskerrokseen (kuva 4). CAN-standardi toteuttaa näistä kerroksista fyysisen kerroksen (Physical layer) ja osittain siirtokerroksen (Data link layer). Ylempien kerrosten toiminnallisuus on usein toteutettu ohjelmistojen avulla. Protokollat jotka toteuttavat nämä kerrokset osittain tai kokonaan, kutsutaan korkeamman tason protokolliksi (higher layer protocols). CANopen-protokollat toteuttavat yhdessä kerrokset 3-6 ja täten sitä voidaan kutsua korkeamman luokan protokollaksi. [6, s. 18]



Kuva 4. OSI-malli. Havainnollistaa CAN ja CANopen-stadardien toteuttavat kerrokset. [6, s. 18]

3.2 CANopen-tunnisteet

Kaikilla CANopen-väylän kytketyllä solmupisteellä on yksilöllinen tunniste, Node-ID. Solmujen tunnisteiden arvot voivat olla väliltä 1 – 127. Tunniste 0 on varattu kaikille solmuille tarkoitetuille NMT-komennoille. [8, s. 1]

Connection Object ID (COB-ID) on 11-bittinen sanomatunniste CAN-kehyksessä, joka priorisoi väylän liikenteen törmäystilanteessa (pienemmän ID:n omaava saa jatkaa lähetystä). Kehyksen 4 ensimmäistä bittiä määrittelee CANopen-protokollan (NMT, SDO, PDO, yms...) ja loput 7 bittiä ovat aiemmin esitelty Node-ID. [9, Luku 1.1]

3.3 Objektikirjasto

CANopen-protokollan oleellisin osa on objektikirjasto (Object Dictionary). Jokaisella solmupisteellä on oma objektikirjasto. Objektikirjasto toimii CANopen-solmussa tietovarastona, jota voivat lukea tai kirjoittaa muut solmupisteet. Se myös sisältää solmupisteen kuvauksen konfiguraatiosta ja toiminallisuudesta.

Objektikirjasto sisältää indexejä. Indexi on 16-bittinen osoite, joka tarkoittaa että indeksejä on yhteensä 65 536, joista jokainen on vielä jaettu ali-indeksiin. Ali-indeksi on 8-bittinen osoite, jolloin niitä voi olla 256. Jokaisella indeksillä on oltava vähintään 1 ali-indeksiin. [6, s. 42 - 47]

3.4 EDS ja DCF -tiedostot

Electronic Datasheets (EDS) -tiedostolla kuvataan solmupisteen koko objektikirjaston rakenne. Tiedosto avulla solmun muuttaminen ja liittäminen CANopen-järjestelmään on mahdollista. [6, s. 56]

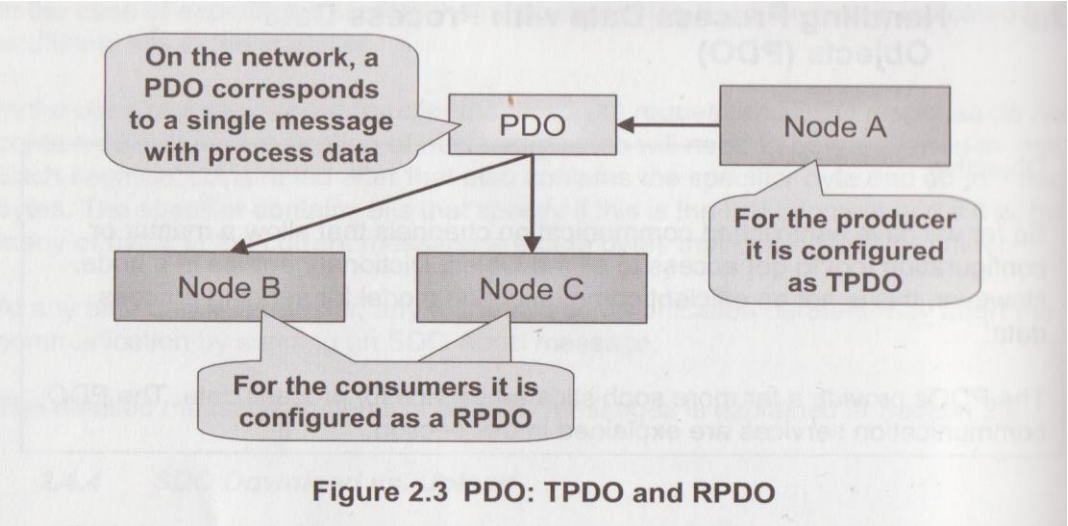
Device Configuration files (DCF) -tiedosto on hyvin saman kaltainen kuin EDS-tiedosto. Väylällä saattaa olla useampi samanlainen solmupiste, jotka käyttävät samaa EDS-tiedostoa. Jotta jokaiselle näistä solmupisteistä saataisiin omat asetukset, DCF-tiedostot tehdään jokaiselle solmupisteelle EDS-tiedostoon perustuen. [6, s. 60]

3.5 SDO-protokolla

Service Data Object (SDO) -protokollan avulla CANopen solmupisteiden objektikirjastoista voidaan lukea tai kirjoittaa. Viestin avulla master tai muu CANopen-solmupiste voi pyytää tietyltä CANopen-solmupisteeltä, että mitä dataa sillä on esim. Indeksi 1241h, ali-indeksi 01h:ssa. Solmupisteen tunnistettuaan pyynnön itselleen, se lähettää kyseisen indeksin/ali-indeksin tiedot riippumatta kysyjästä. SDO-protokollan heikkous on sen suuri kulutus kaistanleveydestä. [6, s. 61 - 64]

3.6 PDO-protokolla

Process Data Object (PDO) -protokolla on tarkoitettu prosessidatan jatkuvaan viestintään. Yhdellä viestillä voidaan lähettää 1 – 8 tavua dataa. PDO sisältää kahta eri tyypistä viestiä: Transmit Process Data Objects (TPDO) ja Receive Process Data Objects (RPDO). PDO viestinnässä on siis yksi TPDO solmupiste, joka lähettää viestiä väylään. Joten on oltava yksi tai useampi RPDO solmupiste, jotka vastaanottavat ja käyttävät tätä viestiä.



Kuva 5. PDO viestinnän toimintaperiaate. [6, s. 66]

PDO:ssa käytetään kahdenlaista konfiguraatio parametria: kommunikaatio ja kartoitus parametria. Kommunikaatio parametrissa (communication parameter) määritellään käytettävä CAN-viesti ja kuinka se laukaistaan. Kartoitus parametrilla (mapping parameter) osoitetaan, mitkä objektikirjaston merkinnät sisältyvät PDO:ssa.

PDO-viestin lähetyksen liipaisu voidaan toteuttaa neljällä eri tapaan: event drive (change-of-state), time driven, individual polling ja synchronized.

Event drive -liipaisua käytettäessä solmupiste tunnistaa muutoksen seurattavassa suureessa ja tallentaa tiedon objektikirjastoon, jonka jälkeen lähettää tiedon PDO-viestissä eteenpäin. Tämä tapa on yksi nopeimpia vasteajaltaan.

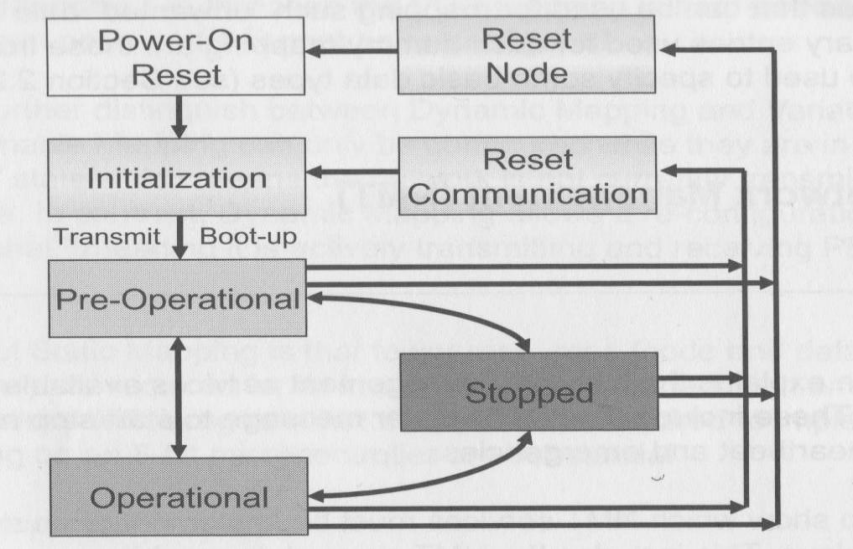
Time driven -liipaisu voidaan konfiguroida lähettämään PDO-viesti kiinteä aikaisesti esimerkiksi 50 millisekunnin välein. Tämä tapa helpottaa ennakoimaan väylän kuormitusta.

Individual polling -liipaisu tapahtuu vain silloin, kun jokin muu väylän solmupisteistä pyytää PDO-viestiä toiselta solmulta. Tätä tapaa ei suositella käytettäväksi, koska kaikki valmistajat eivät toteuta tapaa samalla tavalla CAN-ohjaimissaan.

Synchronized -liipaisu on suunniteltu toimimaan SYNC-signaalin kanssa. Sync-signaali on viesti, joka ei sisällä lainkaan dataa, vaan sitä käytetään ainoastaan PDO-viestien tahdistukseen. [6, s. 65 - 70]

3.7 NMT-protokolla

Network Management -protokollan (NMT) avulla voidaan ohjata CANopen-verkon orjalaitteita. NMT-viestejä voidaan lähettää yhdelle tietylle solmupisteelle tai kaikille solmupisteille yhtäaikaisesti. Viestit sisältävät uuden tilatiedon, johon solmupisteen tulee siirtyä. Jotkin tilavaihdokset voidaan toteuttaa solmupisteellä automaattisesti. Solmupisteillä on 3 erillaista päätilaa: pre-operational, operational ja stopped.



Kuva 6. NMT-tiladiagrammi solmupisteellä. [6, s. 84]

Käynnistettäessä CANopen-verkkoa solmupisteet aloittavat power-on reset -tilasta, jonka jälkeen ne alustetaan (Initialization). Tällöin koko sovellutus, CANopen-rajapinta ja kommunikaatio alustetaan. Alustuksen jälkeen solmupiste yrittää lähettää boot-up -viestiä, jonka onnistuttua solmupiste siirtyy pre-operational -tilaan. Pre-operational -tilassa solmupiste osallistuu kaikkeen kommunikaatioon, jotka liittyvät SDO, hätätila, aikaleima tai heartbeat/Node guardian -viestintään.

Operational -tila on muuten samanlainen kuin pre-operational -tila, mutta tilaan on lisätty PDO-kommunikaatio, joka sallii solmupisteen vaihtaa ja käsitellä prosessidataa. Tätä tilaa voidaan pitää ns. normaali tilana, jolloin solmupisteet voivat lähettää ja vastaanottaa viestejä.

Stopped -tilassa solmupiste keskeyttää kaiken kommunikaation, paitsi heartbeat ja nodeguard -toiminnot.

NMT-isänällä on myös kaksi erilaista alustuskäskyä: Reset Communication ja Reset Node. Reset communicaton -tilassa solmupiste alustaa CAN ja CANopen kommunikaatio rajapinnat. Reset node -tilassa alustetaan solmupiste ja sen kaikki

oheislaitteet. Molemmat alustuskäskyt johtavat uuteen boot-up -viestiin solmupisteeltä, jonka jälkeen se siirtyy takaisin pre-operational -tilaan odottamaan NMT-isännältä uutta tilakäskyä. [6, s. 83-86]

3.8 Heartbeat ja Node Guardian -protokollat

Node Guardian-protokollassa NMT-isäntä käyttää pollaus metodia selvittääkseen verkon solmupisteiden NMT-tilan. Mikäli solmupiste ei vastaa tietyn ajan kuluessa tiedustelusta, saattaa NMT-isäntä tulkita solmupisteen kadonneeksi ja ajaa täten järjestelmän hallitusti alas. Tämä protokolla on vanhentunut ja se korvaajaksi suositellaan uudempaa Heartbeat-protokollaa.

Heartbeat-protokolla etuja vanhentuneeseen Node Guardian-protokollaan verrattuna on pienempi kaistanleveys, joustavampi ja turvallisempi. Jos väylältä yksi solmupiste katoaa, se ei välttämättä aja koko järjestelmää alas. Heartbeat-protokolla avulla CANopen-solmupiste voi lähettää tilatietoa itsestään väylän muille solmuille. Samalla muut solmut ja NMT-isäntä voivat valvoa kyseistä tilatietoa. [6, s. 86-87]

4 Käytettävä Laitteisto

Opinnäytetyön fyysinen osuus koostuu pääasiassa kolmesta osasta. Ensimmäinen osa on Raspberry Pi, jonka avulla ohjataan CAN-väylää. Toinen osa on SPI/CAN-väylämuunnin, joka on rakennettu käyttäen pohjana muiden harrastajien suunnitteluita ja kokeiluja. Kolmas on CAN-väylälaite, joka on tässä työssä Beckhoffin LC5100 I/O -moduuli.

4.1 Raspberry Pi

Raspberry Pi (RPI) on britanialaisen Raspberry Pi Foundationin kehittämä yhden piirilevyn tietokone. Ensimmäinen tuoteversion julkaistiin 29.2.2012, jonka jälkeen ensimmäisestä mallista säätiö on julkaissut 3 eri versiota. Viimeisin malli Rasberry Pi 2 julkaistiin helmikuussa 2015. [2]

Tässä työssä käytetään RPI B -versiota, johon on asennettu viimeisin Rasbian-käyttöjärjestelmä. Luvun 5. konfigurointi ohjeet soveltuvat myös uudemmalle Raspberry Pi 2 -mallille.

4.2 SPI/CAN -väylämuunnin

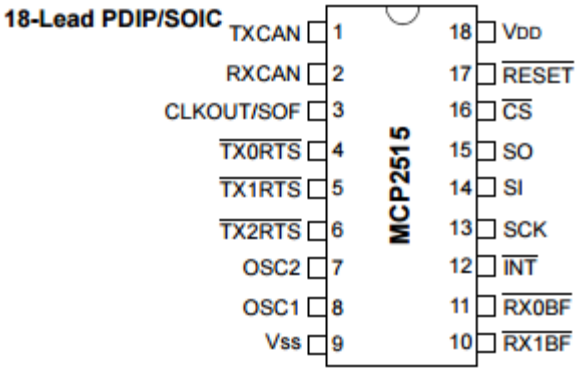
RPI:n GPIO pinneistä ei löydy suoraa liitäntää CAN-väylälle, joten tarvitaan väylämuunnin, joka kääntää RPI:n tukeman rajapinnan CAN-väyläksi. Parhaiten rajapinnaksi sopii RPI:n tukema SPI-väylä. Tiedonsiirrossa käytetään SPI-väylän mosi ja miso pinnejä. Väylämuuntimen MCP2515 ja MCP2551 -piirit saavat käyttöjännitteensä RPI:n GPIO pinneistä. Kaikki väylämuuntimen ja RPI:n väliset johdotukset löytyvät liitteestä 1. Raspberry Pi:n GPIO -pinnijärjestys löytyy liitteestä 2.

MCP2515 -piirin kiteen kellotaajuudeksi suositellaan tässä työssä 16 Mhz, mutta jos haluaa käyttää muuta arvoltaan olevaa kidettä, se tulee ottaa huomioon konfiguroidessa RPI:tä.

MCP2515 ja MCP2551 -piirien käyttöjännite eroista johtuen pitää lisätä "jännitteen puolittaja" MCP2515 pin-2 (RXCAN) ja MCP2551 pin-4 (RXD) välille (katso liite 1.). Mikäli käytettäisiin piirejä jotka toimivat 3,3V käyttöjännitteellä, ei tätä lisäystä tarvittaisi.

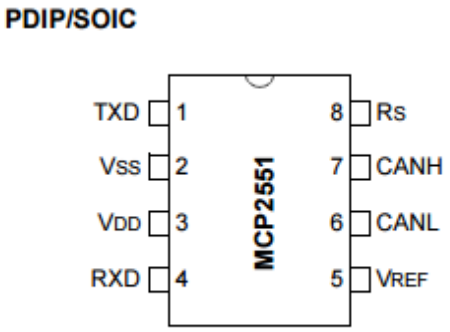
Tarvittaessa RPI:lle löytyy myös kaupallisia versioita SPI/CAN-väylämuuntimista. Yksi tällainen väylämuunnin on PICAN -piirilevy, jotka sijoitetaan suoraan RPI:n GPIO pinneihin kiinni. PICAN -väylämuuntimessa käyttää samoja mikropiirejä kuin tässä opinnäytetyössä.

4.2.1 MCP2515 Stand-alone CAN controller



MCP2515-piiri on erillinen CAN-ohjain. Piirin käyttöjännite 2,7V-5,5V. Suositellaan käytettävän 3,3V käyttöjännitettä. Tukee molempia CAN-standardeja ja 1 Mb/s siirtonopeutta. High speed SPI interface (10 Mhz). Piiri soveltuu vastaanottamaan ja lähettämään data-, etä- sekä laajennettuja kehyksiä. [3, s. 1]

4.2.2 MCP2551 High-speed CAN transeiver



MCP2551-piiri on vikasietoinen High-speed CAN laite, joka toimii rajapintana CAN-ohjaimelle ja fyysiselle CAN-väylälle. Piirin käyttöjännite 4.5V-5.5V. Tukee 1 Mb/s siirtonopeutta. Soveltuu 12V ja 24V järjestelmille. Liitettävyys jopa 112:lle solmupisteelle. [4, s. 1-3]

Piirillä on kolme erilaista toimintatilaa: High-speed, Slope-control ja Standby. Tässä työssä käytetään ainoastaan Standby -tilaa, mutta on myös hyvä silti ymmärtää muiden tilojen ominaisuudet ja niiden kytkennät ongelmien varalta.

High-speed -tila valitaan kytkemällä Rs -pinni Vss:än. Tässä tilassa lähettimen ulostulon ohjaimella on käytössä nopeammat nousu- ja laskureunat. Tätä tilaa on käytettävä, kun väylän tiedonsiirtonopeus on 1 Mb/s.

Slope-control -tilalla vähennetään elektromagneettista häiriötä CANH- ja CANL-johtimien välillä, rajoittamalla nousu- ja laskureunojen aikaa. Tila valitaan kytkemällä Rs ja Vss:än väliin vastus, jonka arvo on 10 k ja 120 k ohmin väliltä. Käyttämällä suurempaa vastusarvoa saavutetaan nopeampi Slew Rate (SR).

Standby -tilassa piiri voidaan asettaa joko valmiustilaan tai SLEEP -tilaan. SLEEP -tilassa lähetin on pois päältä, mutta vastaanotin on yhä toiminnassa. Ohjaimen puoleinen vastaanotin (RXD) on yhä toiminnassa, mutta toimii hitaalla vasteajalla. Piiriin kanssa toimiva CAN-ohjain pystyy siis yhä monitoroimaan väylää RXD -pinnin kautta ja asettamaan lähettimen valmiustilaan tarvittaessa. Tila valitaan kytkemällä Rs ja Vss:än väliin vastus, jonka arvo on esim. 4,7 k ohmia [4, s. 3-4]

4.3 LC5100 I/O -moduuli

Beckhoffin LC5100 on CANopen-protokolalla toimiva edullinen I/O-moduuli. Moduuliin voidaan liittää maksimissaan 64 kappaletta k-tyypin I/O-lisäkorttia. Moduulin kyljestä löytyy dippikytkimet joiden avulla voidaan valita väylän nopeus 4 eri nopeudesta: 1 Mb/s, 500 kb/s, 250 kb/s ja 125 kb/s. Laiteesta löytyy tuki PDO-protokollan kaikille viestin lähetys muodoille. Moduulin käyttöjännite on 24V. Moduuli noudattaa CiA:n DS401-sertifikaattia. [11]

5 Raspberry Pi:n konfigurointi

Maaliskuussa 2015 Raspberry Pi Foundation julkaisi ajurin, joka tukee SPI-väylän ja MCP2515-piirin välistä kommunikointia. Aiemmin käyttäjien tuli ladata ja asentaa muiden harrastajien muokkaamat kernel tiedostot omalle RPI:lle. Ajuria kuitenkaan ole valmiiksi asennettuna vasta käyttöön otetulla RPI:lla. Ajuri asennetaan päivittämällä RPI seuraavilla LXterminaaliin syötettävillä komennoilla. Varmista ennen aloittamista, että RPI:llä on yhteys Internetiin. Ensiksi päivitetään RPI:n käyttöjärjestelmä, jonka jälkeen käynnistetään käyttöjärjestelmä uudestaan.

```
sudo apt-get update
sudo apt-get upgrade
sudo reboot
```

Seuraavaksi päivitetään RPI:n ajurit.

```
sudo rpi-update
```

Tämän jälkeen RPI:n hakemistosta /boot/overlays löytyy tiedosto mcp2515-can0-overlay.dtb. Tämä tarkoittaa että mcp2515 -piiriä tukema ajuri on asennettu.

Jotta mcp2515 -piiri toimisi SPI-väylän kanssa, avata SPI-rajapinta seuraavilla komennoilla.

```
cd /usr/bin
sudo ./raspi-config
```

Komentojen jälkeen RPI:lle avautuu konfiguraatio -ikkuna, josta valitaan ensiksi kohta 8 (Advanced Options). Tämän jälkeen valitaan kohta A6 (SPI). Hyväksytään SPI-rajapinta käyttöön (enable) ja hyväksytään myös SPI kernel moduuli ladattavaksi aina RPI:tä käynnistettäessä. Viimeiseksi käynnistetään käyttöjärjestelmä uudestaan.

```
sudo reboot
```

Seuraavaksi avataan hakemisto, johon lisätään overlayt ja syötetään mcp2515 piirin kellotaajuus.

```
sudo nano /boot/config.txt
```

Lisätään tiedoston perään seuraavat 2 riviä.

```
dtoverlay=mcp2515-can0-overlay,oscillator=16000000,interrupt=25
dtoverlay=spi-bcm2835-overlay
```

Tallennetaan muutokset ja käynnistä RPI uudelleen.

```
sudo reboot
```

Tämän jälkeen voidaan aktivoidaan CAN-rajapinta ja asetetaan sille haluttu väylänopeus, joka on tässä tapauksessa 125 kb/s (yleisimmät väylänopeudet 33,333 b/s, 50 kb/s, 83,333 b/s, 100 kb/s, 125 kb/s, 250 kb/s, 500 kb/s, 800 kb/s ja 1 Mb/s). Komento on syötettävä aina uudelleen RPI:lle käynnistyksen jälkeen.

```
sudo /sbin/ip link set can0 up type can bitrate 125000
```

Tämän jälkeen CAN-portti pitäisi tulla näkyville verkkoliitäntöjen listauksessa. Liitännän nimi on can0 ja sen voi tarkistaa komennolla.

```
ifconfig
```

Seuraavaksi on hyvä asentaa CAN-kommunikaation perustyökalut hakemistoon /usr/local/bin.

```
git clone https://git.gitorious.org/linux-can/can-utils.git
make -C can-utils
sudo make -C can-utils install PREFIX=/usr/local
rm -rf can-utils
```

Asennetuilla ohjelmilla voidaan mm. lähettää CAN-viestejä väylään ja monitoroida väylää. Ennen kuin CAN-ohjelmia voidaan käyttää täytyy siirtyä hakemistoon, jossa ohjelmat ovat.

```
cd /usr/local/bin
```

Tämän jälkeen yksinkertaisen CAN 2.0A -viestin lähettäminen väylään onnistuu komennolla.

```
sudo ./cansend can0 7DF#0201050000000000
```

5.1 Väylän monitorointi

Yksinkertainen tapa testata laitteiston toiminta on monitoroida väylällä kulkevia viestejä. Tämä onnistuu käyttämällä edellisessä luvussa asennettuja työkaluja. Seuraavalla komennolla aktivoidaan CAN-väylän monitorointi, jossa ohjelma tulostaa riveittäin väylällä havaitut viestit, jotka voidaan tarvittaessa tallentaa .

```
sudo ./candump can0
```

Vaihtoehtoisesti väylää voidaan monitoroida Wireshark -ohjelman avulla, jonka asentaminen onnistuu seuraavilla komennoilla.

```
sudo apt-get install wireshark
```

Tämän jälkeen Wireshark -ohjelma on käytettävissä RPI:n visuaalisessa käyttöympäristössä. Jotta CAN-väylä näkyisi Wireshark -ohjelman listaamissa rajapinnoissa, täytyy aina ensin ajaa CAN-väylä aktiiviseksi seuraavalla komennolla.

```
sudo /sbin/ip link set can0 up type can bitrate 125000
```

5.2 VNC:n asennus

Tämä luvun ohjetta ei ole pakollista suorittaa, koska se käsittää ainoastaan graafisen etäyhteyden muodostamisen RPI:lle. On kuitenkin suositeltavaa suorittaa ohjelman asennus, koska suoraan RPI:llä työskentely ei ole aina mahdollista tai kätevää. Virtual Networkin Computing (VNC) on protokolla tietokoneen graafiseen etäkäyttöön.

Aloitettaan asentamalla tightVNC -paketti komennolla

```
sudo apt-get install tightvncserver
```

Tämän jälkeen suoritetaan VNCserver, jonka yhteydessä ohjelma pyytää syöttämään käyttäjän itse valitseman salasanan etäyhteydelle. Viimeiseksi ohjelma kysyy salasanaa view-only -tilalle, jota ei välttämättä tarvita lainkaan. Komento syötettävä uudelleen jokaisen RPI:n käynnistyksen jälkeen.

```
tightvncserver
```

Seuraavalla komennolla määritellään etäyhteyden istunto ja sen resoluutio.

```
vncserver :0 -geometry 1920x1080 -depth 24
```

Komennon jälkeen etäyhteyden muodostaminen muulla päätteellä voidaan luoda VNCviewer -ohjelman avulla.

Mikäli RPI:n ip-osoite ei ole tiedossa, sen saa selville seuraavalla komennolla.

```
ifconfig
```

5.3 CANopen-ohjelmakirjasto

Voidaksemme kommunikoida CANopen-solmupisteen kanssa tarvitaan tätä varten sovellus tai ohjelmakirjasto, joka toimii RPI:n Linux-käyttöjärjestelmässä sekä SPI-rajapinnan kanssa. Tähän tarkoitukseen soveltuu Libcanopen -ohjelmakirjasto, joka on avoin CANopen-kirjasto ja on suunniteltu toimimaan yhdessä linuxin SocketCAN -ajurin kanssa. Kirjasto sisältää kaikki yleisimmät CANopen-protokollat ja se on ladattavissa seuraavalla komennolla.

```
git clone https://github.com/rscada/libcanopen
```

Tämän lisäksi tutkittiin myös kahta muuta mahdollista CANopen-ohjelmakirjastoa, jotka niin ikään perustuvat avoimeen lähdekoodiin: CANFestival ja CANopenNode. CANFestival-ohjelmakirjasto on suunniteltu käyttämään muita rajapintoja kuin SPI-rajapintaa, esimerkiksi USB-väylää ja sarjaporttia. Tämän takia CANFestival-ohjelmakirjasto ei sovellu tässä työssä käytettäväksi. CANopenNode-ohjelmakirjasto on puolestaan suunniteltu toimimaan tietyn mallisten mikropiirien kanssa, joka estää kyseisen ohjelmakirjaston toiminnan tässä työssä.

Jotta Libcanopen-ohjelmakirjaston sisältöä olisi edes mahdollista käyttää, tarvitaan RPI:lle C-kieltä tukeva ohjelmointityökalu. Tällainen työkalu on esimerkiksi Geany, jota voidaan käyttää RPI:n visuaalisessa käyttöympäristössä. Ohjelman asentaminen RPI:lle onnistuu seuraavalla komennolla.

```
sudo apt-get install geany
```

Valitettavasti opinnäytetyötä tehtäessä ei onnistuttu asentamaan tai käyttämään Libcanopen-ohjelmakirjastoa tarkoitetulla tavalla, koska Linux-pohjainen käyttöjärjestelmä ei ollut riittävät tuttu ympäristö työntekijälle, eikä ohjeita tai manuaali ole saatavana. Näin ollen kommunikaatiota CANopen-protokolalla jäi tässä tapauksessa saavuttamatta.

6 Yhteenveto

Opinnäytetyössä oli tavoitteena luoda kommunikaatio yhteys Raspberry Pi:n ja Beckhoffin CANopen-protokollaan perustuvan LC5100 I/O-moduulin välille. Tavoite jäi kuitenkin toteuttamatta, koska opinnäytetyössä ei onnistuttu käyttämään Libcanopen-ohjelmakirjastoa Linux-käyttöjärjestelmässä siihen varatussa ajassa. Kommunikaatio yhteys on kuitenkin täysin mahdollista toteuttaa SPI-rajapintaa hyödyntäen, koska työssä onnistuttiin kuitenkin todistamaan sen toiminta lähettämällä CAN-väylälle viestejä ja monitoroimaan verkossa liikkuvaa dataa. Saavutetulla työllä on mahdollista esimerkiksi monitoroida CAN-väylällä varustetun henkilöauton väyläviestejä. Se että viestien datat saataisiin käännettyä ymmärrettävään muotoon, vaatisi auton valmistajalta manuaalin tai ajurin.

Työn jatkokehitys vaatisi toimivaa CANopen-ohjelmakirjastoratkaisua, joka voidaan mahdollisesti toteuttaa jo esitellyllä Libcanopen-ohjelmakirjastolla tai muulla vastaavalla sovellutuksella. Mikäli tämä saataisiin toimimaan, voitaisiin I/O -moduuli liittää esimerkiksi johonkin prosessiin, josta se tuottaisi input dataa Raspberry Pi:lle. Ja minkä perusteella Raspberry Pi voisi ohjata jotakin output -toimilaitetta.

Loppupäätelminä voidaan todeta ettei Raspberry Pi sovellu nykyisellään ohjaamaan CANopen-protokollaan pohjautuvia automaatiolaitteita, mutta pitää huomioida että SPI/CAN-rajapinta on myös jatkuvassa kehityksessä Rasbian -käyttöjärjestelmässä ja sen käyttämistä on jo helpotettu merkittävästi viimeisen puolen vuoden aikana. Tilanne saattaa muuttua merkittävästi tulevaisuudessa, kun Raspberry Pi 2 -mallille optimoitu Windows 10 -käyttöjärjestelmä julkaistaan, joka avaa taas lisää mahdollisuuksia hyödyntää jo Windows -laitteille suunniteltujen ohjelmistojen käyttöä.

Lähteet

1 Heikki Saha. CAN Dictionary. 2008. Verkkodokumentti. <http://www.can-cia.org/fileadmin/cia/pdfs/CANdictionary-v4_fi.pdf>

2 Raspberry Pi. Verkkodokumentti. <http://fi.wikipedia.org/wiki/Raspberry_Pi>

3 MCP2515 Datasheet. Verkkodokumentti. Microchip. <http://ww1.microchip.com/downloads/en/DeviceDoc/21801e.pdf>

4 MCP2551 Datasheet. Verkkodokumentti. Microchip. <http://users.ece.utexas.edu/~valvano/Datasheets/MCP2551.pdf>

5 Heikki Saha. CAN-väylä. Verkkodokumentti. <http://www.canopen.fi/artikkelit/CAN.pdf>

6 Olaf Pfeiffer, Andrew Ayre, and Christian Keydel. 2003 Embedded networking with CAN and CANopen. RTC Books.

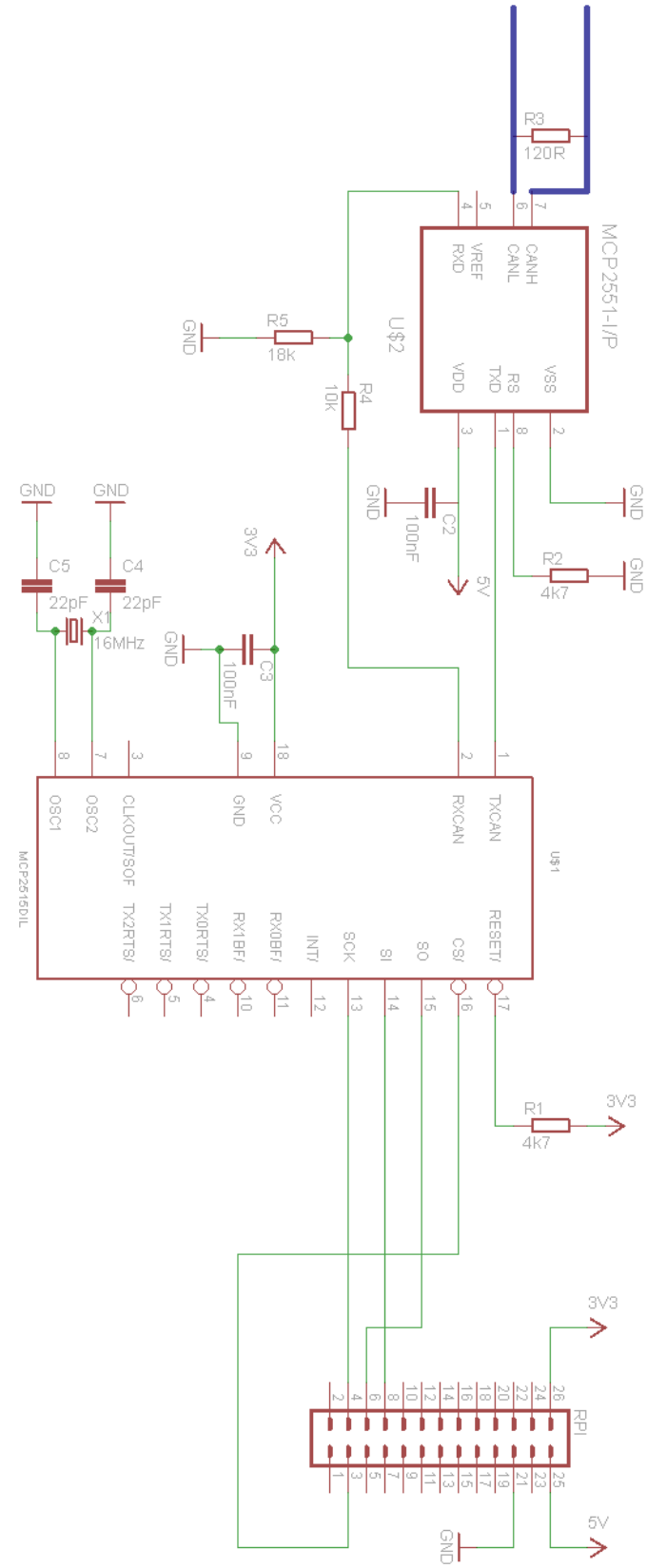
7 CAN-väylä. Verkkodokumentti <http://fi.wikipedia.org/wiki/CAN-v%C3%A4yl%C3%A4>

8 Heikki Saha. CANopen perusteet. Verkkodokumentti <http://www.canopen.fi/artikkelit/CANopen.pdf>

9 CANopen. Verkkodokumentti. <http://en.wikipedia.org/wiki/CANopen>

10 Application domains. Verkkodokumentti. CIA.<http://www.can-cia.org/index.php?id=30%3E.>

11 CANopen Coupler BK5120, BK5110, LC5100. Verkkodokumentti. Beckhoff <http://pdf.datasheetarchive.com/indexerfiles/Datasheets-UD1/DSAUD00378.pdf>



Liite 1. SPI/CAN-väylämuunnin

Liite 2. Raspberry Pi GPIO pinnit