Tony Ikäheimo

Auton tietojen kerääminen Raspberry Pi 2 -tietokoneella OBD-II-järjestelmästä

Opinnäytetyö
Kevät 2016
SeAMK Tekniikka
Tietotekniikan Tutkinto-ohjelma

SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Tekniikan yksikkö

Tutkinto-ohjelma: Tietotekniikka

Suuntautumisvaihtoehto: Sulautetut järjestelmät

Tekijä: Tony Ikäheimo

Työn nimi: Auton tietojen kerääminen Raspberry Pi 2 -tietokoneella OBD-II-järjestelmästä

Ohjaaja: Heikki Palomäki

Vuosi: 2016 Sivumäärä: 44 Liitteiden lukumäärä: 6

Opinnäytetyön tavoitteena on kerätä tietoa auton OBD-II-järjestelmästä Raspberry Pi 2 -ohjelmointialustalle ja luoda Windows-ohjelma, jolla saadaan esitettyä tallennettu lokitiedosto graafisesti.

Raspberry Pi 2:lle piti asentaa käyttöjärjestelmä sekä tarvittavat ohjelmat, jotka selvennetään tässä työssä. Tämän jälkeen kaikki tarvittavat ohjelmat ja laitteet testattiin ennen varsinaista asennusta. Raspberry Pi 2 -alusta ohjelmoitiin aloittamaan tallennus automaattisesti, kun auto käynnistetään.

Saatu lokitiedosto siirretään Windows-pohjaiselle tietokoneelle, johon on asennettu tässä työssä tehty ohjelma. Ohjelmasta valitaan haluttu lokitiedosto ja ohjelma muuntaa sen graafiseksi esitykseksi.

Avainsanat: Raspberry Pi, OBD-II, Raspbian, pyOBD, ELM327

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree programme: Information Technology

Specialisation: Embedded systems

Author: Tony Ikäheimo

Title of thesis: Using Raspberry Pi 2 for reading data from the OBD-II-system of a car

Supervisor: Heikki Palomäki

Year: 2016 Number of pages: 44 Number of appendices: 6

The objective of this Bachelor's thesis is to use a Raspberry Pi 2 -computer for reading and logging data from a car's OBD-II -system via an ELM327-adapter. The aim was to log data to a Raspberry Pi 2 -computer and create a Windows program which presents the raw data as graphic diagrams. Hereby, for example, a hobbyist can see understandable diagrams presenting the information conveyed by the car's sensors and decide where the fault is.

The ELM327-adapter is a cable which connects a Raspberry Pi 2 -computer to a car's OBD-system. On the Raspberry Pi 2 -computer there is a pyOBD-program that is written in the python-language. This is an open source program which can be downloaded from the GitHub-website. The pyOBD-program makes log files to the Raspberry Pi 2 -computer and these files are then moved to another computer which has a Windows operating system. In that computer the created program shows the user clear and easily understandable diagrams.

Keywords: Raspberry Pi, OBD-II, Raspbian, pyOBD, ELM327

SISÄLTÖ

Kuvaluettelo

Käytetyt termit ja lyhenteet

C#	Ohjelmointikieli
Cron	Unix-pohjaisilla käyttöjärjestelmillä oleva ajastuspalvelu
ELM327	Adapteri, joka luo virtuaalisen portin OBD-järjestelmän ja USB-portin välille
gitHub	Git-versionhallintaa tarjoava verkkosivusto
GPIO	General Purpose I/O, portti, joka voidaan ohjelmoida ottamaan vastaan signaalia tai lähettämään signaalia
HAT	Hardware Attached on Top, lisäkortti, joka laajentaa Raspberry Pi-tietokonetta jättäen alkuperäiset liittimet käytettäviksi
IoT	Internet of Things, esineiden internet
MAF	Mass Air Flow, ilmamassamittari autoissa
microSDHC	Korkean nopeusluokan muistikortti
OBD-II	**(**On Board Diagnostic) tietoliikenneportti autossa
Ohjelmointialusta	Valmis piirilevy komponentteineen, joka on ohjelmoitavissa oman tarpeiden mukaan
PnP	Plug and Play, laite, joka ei tarvitse erillisiä ajureita toimiakseen
pyOBD	OBD-II-yhteensopiva auton diagnostiikkatyökalu
Python	Ohjelmointikieli
Raspberry Pi 2	Toisen sukupolven ohjelmointialusta

Raspberry Pi Foundation	Rekisteröity englantilainen opetushyväntekeväisyysjärjestö, Raspberry Pi-tietokoneiden kehittäjä ja julkaisija
Raspbian	Raspberry Pi Foundationin virallisesti tukema käyttöjärjestelmä
RS232	Sarjaportti tietokonelaitteissa
USB OTG	On-The-Go, USB-portti, joka toimii sekä isäntänä että asiakkaana tarpeen vaatiessa
Win32 Disk Imager	Windows-ohjelma, jolla voidaan kirjoittaa image-tiedostoja irrotettavalle laitteelle

1 JOHDANTO

1.1 Työn tausta

Kasvava mielenkiinto elektroniikkaan ja sulautettuun järjestelmiin sekä harrastuksena oleva autojen korjaaminen johtivat ideaan, jonka tuotoksena syntyi tämä opinnäytetyö. Halu tietää enemmän auton ”sielunelämästä” johti ELM327 USB -adapterin hankintaan, jolla pääsee käsiksi auton OBD-järjestelmään ja siinä oleviin tietoihin. Tällä laitteella ja tavallisella kannettavalla tietokoneella päästään jo lukemaan auton vikatiedot sekä nollaamaan ne.

Raspberry Pi 2 -tietokone, vähäisen virrankulutuksensa takia, voidaan sijoittaa autoon ja käyttää tätä yhdessä ELM327 USB -adapterin kanssa keräämään tietoja autosta ajon aikana ja tallentamaan ne lokitiedostoon. Lokitiedostoja voidaan myöhemmin tarkastella esimerkiksi kotona tietokoneella omassa rauhassa. Tavoitteena on, että ei rikota lakia ja käytetä tietokonetta samaan aikaan kun keskitytään ajamiseen.

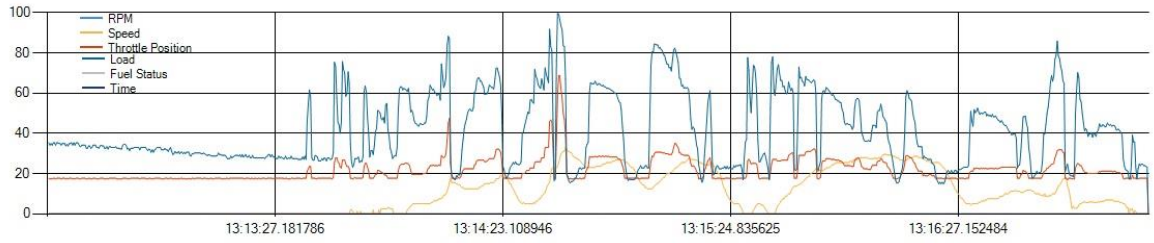
1.2 Työn tavoite

Ensisijaisena tavoitteena on lukea Raspberry Pi 2:n avulla reaaliaikaista dataa autosta ELM327 USB -adapterin avulla ja kirjoittaa niistä lokitiedostoa Raspberry Pi 2:n muistiin. Tämän jälkeen saatu ”raakadata” tuodaan Windows-pohjaiselle tietokoneelle, johon tehdään ohjelma purkamaan tuotu data ja näyttämään se graafisessa muodossa. Näitä graafisia käyriä voidaan sitten tutkia ja helposti huomata, jos jossain on ollut vikaa.

```
13:12:31.910435,1192,0.0,17.6470588235,35.2941176471,0200
13:12:32.209190,1196,0.0,17.6470588235,35.2941176471,0200
13:12:32.508023,1193,0.0,17.2549019608,34.1176470588,0200
13:12:32.778847,1195,0.0,17.6470588235,35.2941176471,0200
13:12:33.36705,1195,0.0,17.6470588235,35.2941176471,0200
13:12:33.290538,1200,0.0,17.6470588235,34.9019607843,0200
13:12:33.544380,1203,0.0,17.6470588235,34.9019607843,0200
13:12:33.815151,1206,0.0,17.2549019608,34.1176470588,0200
13:12:34.109965,1216,0.0,17.6470588235,34.9019607843,0200
13:12:34.408858,1214,0.0,17.6470588235,34.9019607843,0200
13:12:34.707642,1207,0.0,17.6470588235,34.9019607843,0200
13:12:35.2520,1212,0.0,17.2549019608,33.7254901961,0200
13:12:35.260388,1220,0.0,17.6470588235,34.5098039216,0200
13:12:35.560177,1203,0.0,17.2549019608,33.7254901961,0200
13:12:35.858935,1194,0.0,17.6470588235,34.9019607843,0200
13:12:36.153903,1179,0.0,17.6470588235,34.9019607843,0200
13:12:36.456781,1188,0.0,17.6470588235,35.2941176471,0200
13:12:36.751550,1193,0.0,17.2549019608,34.1176470588,0200
13:12:37.46377,1187,0.0,17.6470588235,35.2941176471,0200
```

Kuva 1. Raakadata

Raakadata (Kuva 1) muutetaan ohjelman avulla selkokielisemmäksi, eli tehdään siitä graafinen esitys (Kuva 2).



Kuva 2. Esimerkki tulevasta ohjelmasta

Kuvia 1 ja 2 verratessa huomaa helposti, että graafista dataa on paljon helpompi tulkita. Tästä graafisesti kuviosta voidaan helposti huomata, jos esimerkiksi autoa kiihdyttäessä kierrokset (sininen käyrä) nousee ja vauhti (keltainen käyrä) ei nouse. Tästä voidaan päätellä, että kytkin luistaa ja kytkinremontti on edessä.

Toissijaisena tavoitteena on saada kaikki toimimaan automaattisesti internetin välityksellä. Raspberry Pi 2:n tallentamat lokitiedostot ladattaisiin automaattisesti palvelimelle, kun auto on ajettu esim. kotipihaan, ja Raspberry Pi 2 ottaa yhteyden langattomaan tukiasemaan.

Palvelimelle tehtäisiin ohjelma, joka toimisi internetin välityksellä ja tuloksia voitaisiin tarkastella mistä tahansa laitteelta, jossa on pääsy internetiin.

1.3 Työn rakenne

Ensimmäisessä luvussa kerrotaan opinnäytetyön taustasta, tavoitteista ja rakenteesta. Toisessa luvussa on kerrottu Raspberry Foundationista, Raspberry Piin historiasta, teknisistä tiedoista sekä Raspberry Pi 2:een asennettavista olevista käyttöjärjestelmistä yleisesti. Kolmannessa luvussa perehdytään OBD-II-järjestelmään ja sen historiaan sekä toimintaan. Neljäs luku kertoo, mitä kaikkea tarvittavaa tarvitaan Raspberry Pi 2:n asentamiseen autoon sekä tarvittavat lisälaitteet. Viides luku kertoo itse ohjelmien käytöstä ja konfiguroinnista Raspberry Pi 2:een sekä itse asennuksesta. Viidennessä luvussa on myös esitelty itse tehty Windows-ohjelma, joka näyttää saadut tiedot graafisessa muodossa. Kuudennessa luvussa on esitelty tulokset. Seitsemännessä luvussa on pohdintaa ja yhteenveto työstä.

2 RASPBERRY PI

2.1 Raspberry Pi Foundation

Raspberry Pi Foundation on rekisteröity englantilainen opetushyväntekeväisyyssäätiö. Säätiön tavoitteena on edistää aikuisten ja lasten koulutusta erityisesti tietotekniikan osa-alueella. (Raspberry Pi Foundation [Viitattu 13.4.2016].)

Idea lapsille suunnattuun edulliseen ja pieneen tietokoneeseen syntyi vuonna 2006. Eban Upton, Rob Mullins, Jack Lang ja Alan Mycroft huolestuivat tietotekniikan opiskelijoiden numeroiden ja taitojen jatkuvasta laskemisesta. (Raspberry Pi Foundation [Viitattu 13.4.2016].)

1990-luvulla monet lapset tulivat työhaastatteluun kokeneina harrastelijaohjelmoijina. 2000-luvulla asiat ovat aivan toisin, tyypillinen hakija on tehnyt vain vähän nettisivuja. (Raspberry Pi Foundation [Viitattu 13.4.2016].)

Jokin on muuttanut nuorten ja tietokoneiden vuorovaikutusta. Monia ongelmia on havaittu: Tietotekniikan opetussuunnitelma keskittyy Wordin, Excelin ja nettisivujen opetukseen: IT-kuplan loppu, tietokoneiden ja pelikonsoleiden suosion nouseminen, jotka korvasit Amigat, BBC Microt, Spectrum ZX:n ja Commodore 64:n, joilla edellinen sukupolvi oli oppinut ohjelmoimaan. (Raspberry Pi Foundation [Viitattu 13.4.2016].)

Vuonna 2011 Raspberry Pi Model B otettiin massatuotantoon element 14/Premier Farnell ja RS Electronics yritysten kanssa tehtyjen sopimusten kautta. Kahden vuoden päästä laitetta oli myyty yli 2 miljoonaa kappaletta. (Raspberry Pi Foundation [Viitattu 13.4.2016].)

Raspberry Pi Foundation ei väitä, että heillä olisi kaikkia vastauksia. He eivät myöskään ajattele, että Raspberry Pi korjaisi kaikki maailman tietokoneongelmat. Ajatuksena on, että Raspberry voi toimia katalyyttina. Raspberry Pi Foundation haluaa nähdä edullisen ja ohjelmoitavan tietokoneen kaikkialla. (Raspberry Pi Foundation [Viitattu 13.4.2016].)

2.2 Raspberry Pi -alustan kehitys

Uusin Raspberry Pi -tietokone on kolmannen sukupolven Raspberry Pi 3. Tämän Raspberry Pi Foundation julkaisi helmikuussa 2016. Ulkomuodoltaan Raspberry Pi 3 on identtinen Raspberry Pi 2:n kanssa. Liittimien ja komponenttien sijoittelut ovat identtiset, ja vanhan sukupolven kotelot ovat yhteensopivia uudemman kanssa. (Raspberry Pi Foundation [Viitattu 6.5.2016].)

Muutokset Raspberry Pi 2-versioon verrattuna ovat seuraavat:

- A 1,2 GHz:n 64-bittinen neliytiminen ARMv8-prosessori
- 802.11n langaton sovitin
- Bluetooth 4.1
- BLE (Bluetooth Low Energy).

(Raspberry Pi Foundation [Viitattu 6.5.2016].)

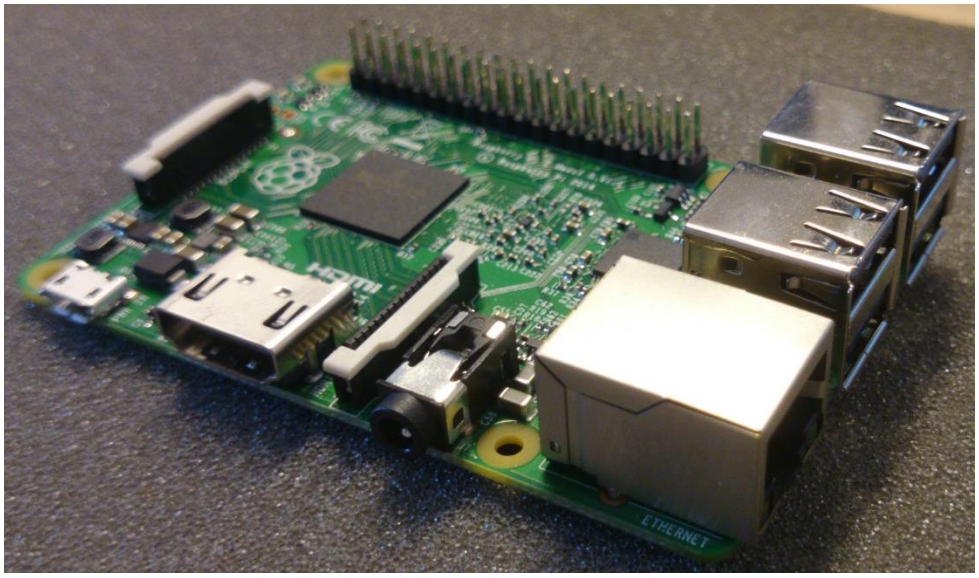
Ominaisuudet ja liitynnät, jotka ovat pysyneet samana Raspberry Pi 1 Model B+-versiosta lähtien:

- 4 USB-porttia
- 40 GPIO-pinniä
- HDMI-portti
- Ethernet-portti
- CSI (Camera Interface)
- DSI (Display Interface)
- Micro SD-korttipaikka

- VideoCore IV 3D -grafiikkasuoritin
- 3,5 mm:n audioliitin, joka tukee kuvaa ja ääntä.

(Raspberry Pi Foundation [Viitattu 6.5.2016].)

Helmikuussa 2015 julkaistiin Raspberry Pi 2 Model B -ohjelmointialusta (Kuva 3), joka korvasi ensimmäisen sukupolven Raspberry Pi -ohjelmointialustat. (Raspberry Pi Foundation [Viitattu 25.4.2016].)



Kuva 3. Raspberry Pi 2

Raspberry Pi 2 on paljon tehokkaampi kuin edeltäjänsä Raspberry Pi 1 Model B+. Siihen tuli lisää muistia ja paljon edellistä parempi prosessori. (Raspberry Pi Foundation [Viitattu 12.1.2016].)

Ensimmäisissä Raspberry-tietokoneissa muistia oli 256 Mt, lokakuussa 2012 määrä päivittyi 512 Megatavuun. Tämä nopeutti Raspberry Piin käyttöä todella paljon. Sillä pystyi tekemään entistä monipuolisempia asioita. 2015 helmikuussa, kun Raspberry Pi 2 julkaistiin, muisti oli päivitetty 1 Gigatavuun. Sama muistimäärä on käytössä

uusimmassa versiossa, Raspberry Pi 3:ssa. (Raspberry Pi Foundation [Viitattu 12.1.2016].)

Ensimmäisen sukupolven Raspberry Pi -tietokoneissa oli prosessorina yksiytiminen 700 MHz:n ARM11-prosessori, joka on suorituskyvyltään noin kuusi kertaa nykyistä neliytimistä 900 MHz:n ARM Cortex-A7 -prosessoria heikompi. (Raspberry Pi Foundation [Viitattu 25.4.2016].)

Arm Cortex-A7 on prosessori, jossa on neljä ydintä. Ytimet ovat tehdasasetuksena 900 MHz, mutta valmistajan mukaan voidaan käyttää jopa 1,6 GHz:n taajuutta. Virtaa prosessori kuluttaa alle 100 mW, mikä tekee tästä hyvän prosessorin moneen eri sovellukseen vähäisen virrankulutuksen myötä. (ARM Ltd [Viitattu 23.12.2015].)

Raspberry Pi 3 -alustassa on käytetty uudempaa A 1,2 GHz:n 64-bittistä neliytimistä ARMv8-prosessoria. Tämä prosessori on hieman paranneltu versio prosessorista, jota Raspberry Pi 2 -alustassa käytetään. Ytimien taajuudet ovat saaneet hiukan nopeutta lisää ja virrankulutusta ollaan saat vähennettyä entisestään. (Raspberry Pi Foundation [Viitattu 6.5.2016].)

Prosessori ja muisti yhdessä mahdollistavat ARM GNU/LINUX -versioiden ajamisen laitteessa. Myös Windows 10 IoT Core on mahdollista asentaa Raspberry Pi 2- ja 3 –alustoihin. (Raspberry Pi Foundation [Viitattu 12.1.2016].)

Uusin tulokas Raspberry perheeseen on Raspberry Pi Zero (Kuva 4).

Kuva 4. Raspberrry Pi Zero
(Upton 2015.)

Raspberry Pi Zero on kooltaan puolet Raspberry Pi Model A+-versiosta. Tekniikan kehittyessä on saatu pienennettyä piirilevyn kokoa ja samalla saatu komponenteista tehokkaampia. Raspberry Pi Zero on hintansa ja kokonsa puolesta todella monipuolinen mihin tahansa projektiin, sillä sen hinta on vain 5 dollaria. (Raspberry Pi Foundation [Viitattu 1.5.2016].)

Raspberry Pi Zeron ominaisuuksia ovat:

- 1 GHz:n yksiytiminen prosessori
- 512 Mt keskusmuistia
- Mini-HDMI-portti
- USB OTG -portti (On-The-Go)
- HAT-yhteensopivat 40-pinniset liittimet

- Komposiittivideo ja reset-pinnit.

(Raspberry Pi Foundation [Viitattu 1.5.2016].)

2.3 Käyttöjärjestelmät

Raspberry Pi 2 –ohjelmointi alusta tukee monia eri käyttöjärjestelmiä kuten Raspbian, Snappy Ubuntu Core, Windows 10 IoT Core, OSMC yms. (Raspberry Pi Foundation [Viitattu 25.4.2016].)

Lyhenne IoT (Internet of Things), esineiden internet, on nykypäivää. Microsoft on tullut mukaan kilpailuun ja tarjoaa ilmaiseksi oman käyttöjärjestelmän IoT-sovelluksiin. Kaikki yritykset haluavat nykyään tuotteensa toimivan internetissä ja luoda niille sovelluksia, jotta lopullinen käyttäjä voi esimerkiksi tarkistaa pesukoneensa tilan kotimatkallaan. (Microsoft Corp 2015.)

2.3.1 Raspbian

Raspbian (Kuva 5) on Raspberry Pi Foundationin virallisesti tukema käyttöjärjestelmä. Se on ilmainen käyttöjärjestelmä, joka pohjautuu Debianiin ja on optimoitu juuri Raspberry Piin laitteistolle. (Raspbian [Viitattu 1.5.2016].)

Kuva 5. Raspbian-logo
(Raspbian [Viitattu 1.5.2016].)

Raspbian sisältää ohjelmia ja työkaluja Raspberry Piin käynnistämiseen ja toimimiseen. Se on helppo asentaa ja ottaa käyttöön, sillä sen mukana tulee yli 35 000 pakettia sekä esikoottuja ohjelmapaketteja (yksi ohjelmapaketti voi aiheuttaa riippuvuuksia moniin eri paketteihin). Se on vakaa ja tehokas käyttöjärjestelmä Raspberry Pi -ohjelmointialustoille. (Raspbian [Viitattu 1.5.2016].)

2.3.2 Windows 10 IoT Core

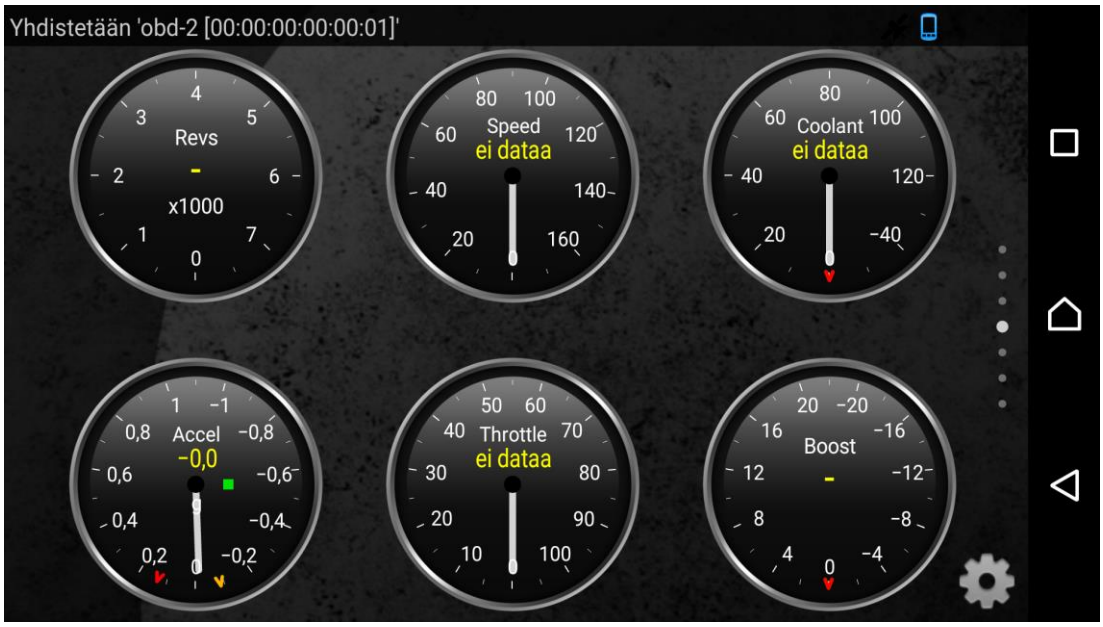
Windows 10 IoT Core on Microsoftin Windows 10 -sarjan versio, joka on optimoitu pienille laitteille, jotka mahdollisesti käyttävät näyttöä (ei-pakollinen etäohjausta käytettäessä). Windows 10 IoT Core toimii Raspberry Pi 2- ja 3-, Arrow DragonBoard 410c- ja MinnowBoard MAX -alustoilla, jotka ovat kaikki ohjelmointialustoja Raspberry Pi 2 -ohjelmointialustan tavoin. Windows 10 IoT Core hyödyntää rikasta ja laajennettavissa olevaa Universal Windows -ympäristön (UWP) ohjelmointirajapintaa (API (Application Programming Interface) rakentaakseen ohjelmia. (Microsoft Corp 2015.)

3 OBD-JÄRJESTELMÄ

On-Board Diagnostic, tunnetummin OBD, löytyy nykyisen lähes jokaisesta henkilöautoista, pakettiautoista sekä lava-autoista. 70-luvun ja 80-luvun alun välissä auton valmistajat alkoivat käyttää elektroniikkaa auton moottorin ohjaamiseen ja vikakoodien selvittämiseen. Näillä muutoksilla saatiin myös täytettyä Yhdysvaltain ympäristönsuojeluviraston pakokaasupäästöstandardien vaatimukset. Ajan myötä OBD-tekniikka on kehittynyt ja tullut entistäkin monipuolisemmaksi. 90-luvun puolivälissä otettiin käyttöön uusi standardi, OBD-II, joka tarjosi melkein täydellisen moottorinohjauksen ja samalla vahti sekä kontrolloi alustaa, koria ja lisävarusteita. (B & B Electronics 2011b.)

”Direktiivin 1999/102/EY mukainen OBD-järjestelmä tuli pääosin pakolliseksi ottomoottorilla varustettuihin M1- ja N1-luokan autoihin, jotka on otettu käyttöön 1.1.2001 tai myöhemmin” (Trafi 2011).

OBD-II-protokollaa tukevasta autosta voidaan lukea reaaliaikaista dataa OBD-II-adapterin avulla. Adaptereita löytyy monelta eri valmistajalta sekä monella eri tekniikalla toimivia. Yleisin on Bluetooth-tekniikkaa käyttävä ELM327 Bluetooth -adapteri, johon muodostetaan yhteys Bluetoothin avulla, esimerkiksi Android-puhelimella. Puhelimessa pitää olla asennettuna tätä adapteria tukeva sovellus, esim. Torque Pro -ohjelmisto, jolla pystytään katsomaan livedataa autosta reaaliajassa (Kuva 6).



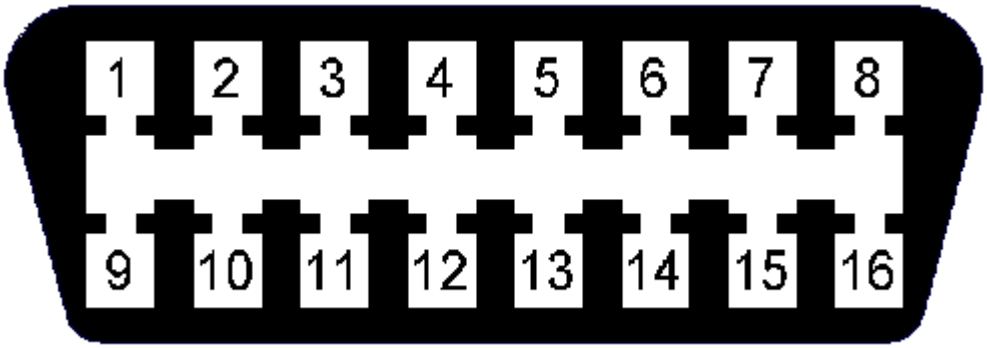
Kuva 6. Torque Pro -Android-sovellus

3.1 OBD-II-versiot

Eri autovalmistajat käyttävät eri OBD-II-protokollia. Käytössä on viisi erilaista protokollaa, joista jokainen poikkeaa toisistaan kommunikoinnin ja vikakoodinlukijan osalta. Muutaman vuoden sisällä on tapahtunut muutoksia valmistajien käyttämissä protokollissa, mutta pääosin Chrysler ja kaikki eurooppalaiset valmistajat ja useimmat aasialaiset valmistajat käyttävät joko ISO 9141- tai KWP2000-protokollaa. GM ja kevytkuorma-autojen valmistajat käyttävät SAE J1850 VPW-protokollaa ja Ford käyttää SAE J1850 PWM-protokollaa. Uusin protokolla on CAN, joka on määritelty OBD-II-järjestelmän laatuvaatimuksiin ja se on määrätty pakolliseksi kaikissa 2008 ja uudemmissa autoissa. (B & B Electronics 2011b.)

OBD-II-järjestelmän käyttämä protokolla voidaan selvittää kahdella tavalla. Ensimmäinen tapa on katsoa auton vuosi, merkki ja malli ja verrata niitä taulukkoon, jossa on kerrottu mitä protokollaa auto käyttää (Liite 1).

Toinen vaihtoehto on katsoa OBD-II-pistokkeen pinnejä. Jokainen eri protokolla käyttää eri pinnejä kommunikoimiseen. Näin voidaan käytettävissä olevista pinneistä todentaa käytettävä protokolla (Kuva 7).



Kuva 7. OBD-II-pistoke
(B & B Electronics 2011a)

Protokollien vaihtoehdot ovat.

- J1850 PWM käytettävät pinnit: 2, 4, 5, 10 ja 16
- J1850 VPW käytettävät pinnit: 2, 4, 5 ja 16, mutta ei 10
- ISO 9141 käytettävät pinnit: 4, 5, 7 ja 16
- ISO 14230 (KWP2000) käytettävät pinnit: 4, 5, 7 ja 16
- CAN käytettävät pinnit: 4, 5, 6, 14 ja 16.

(B & B Electronics 2011a.)

3.2 OBD-pistoke

OBD-pistokkeen sijainti riippuu auton merkistä ja mallista. Ensimmäisissä OBD-järjestelmää tukevissa autoissa pistoke saattoi olla konepellin tai kojelaudan alla. Ny-

kyisin valmistajat ovat sijoittaneet pistokkeet paikkaan, johon käyttäjä pääsee helposti käsiksi kuljettajan paikalta. Sijainteina voi olla keskikonsoli, tuhkakupin takana, kojelauta tai kuljettajan jalkatilat. (B & B Electronics 2011a.)

Vuoden 1997 Ford Mondeossa OBD-pistoke sijaitsee kuljettajan jalkatilassa ratin alapuolella. Pistoke ei ole suoranaisesti näkyvillä, vaan se sijaitsee suojakannen alla (Kuva 8).

Kuva 8. Ford Mondeo 1997 -mallin OBD-pistokkeen sijainti

Jotkut valmistajat eivät ole ottaneet valmistusvaiheessa OBD-II-pistokkeen sijaintia kunnolla huomioon. Mondeossa esimerkiksi ELM327 USB -adapterin käyttö on vaarallista ajon aikana, koska adapteri on pitkän mallinen ja osuu jalkoihin autoa ajaessa (Kuva 9). Tämä voisi luoda vaaratilanteita ajon aikana, eikä tämän mallista adapteria siis kannatakaan käyttää.

Kuva 9. OBD-II-adapteri kytketty pistokkeeseen

Huomattavasti pienempi laite on ELM327 Bluetooth-adapteri (Kuva 10), jonka käyttö onnistuu myös ajon aikana helposti sen pienuuden johdosta.

Kuva 10. ELM327-Bluetooth-adapteri

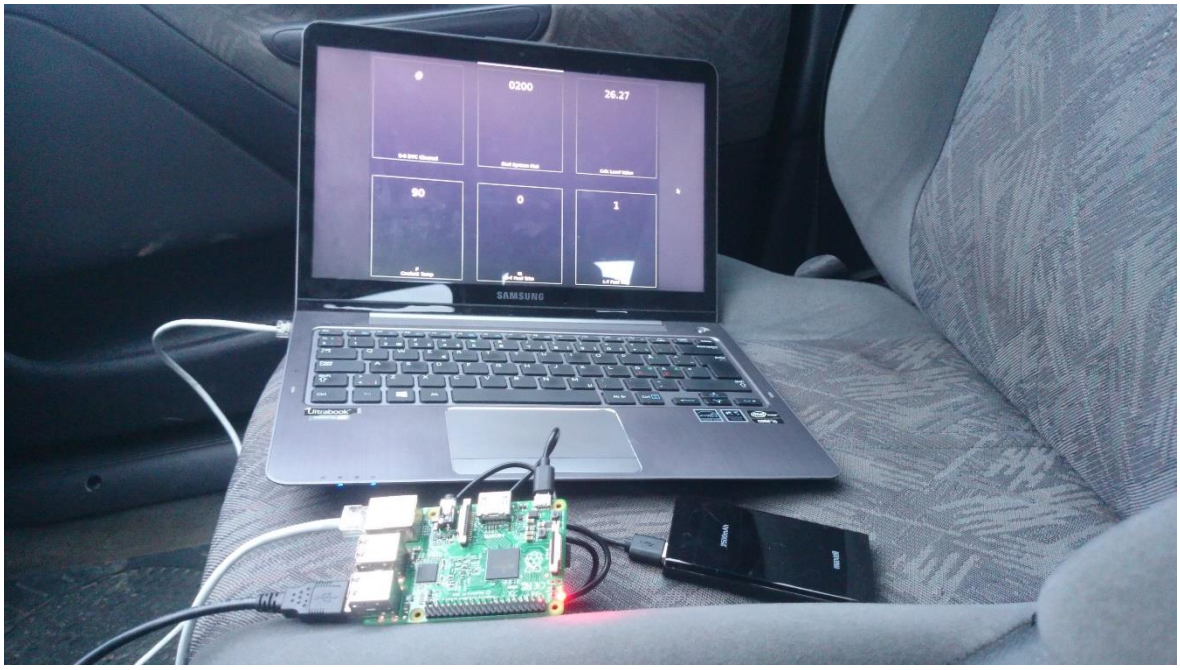
USB- ja Bluetooth-adaptereita vertaillessa voidaan huomata, että Bluetooth-versio on paljon pienempi ja huomattavasti mukavampi käyttää, koska se ei vie paljon tilaa. Bluetooth-versiossa ei myöskään ole ylimääräisiä johtoja, jotka olisivat käyttäjän tiellä autoa ajettaessa (Kuva 11).

Kuva 11. USB- ja Bluetooth-versiot

4 RASPBERRY PI 2:N VALMISTELU

4.1 Testiympäristö

Testiympäristönä (Kuva 12) toimi tässä opinnäytetyössä vuosimallia 1997 oleva Ford Mondeo -henkilöauto, jossa on käytetty jo OBD-II-protokollaa. Kaikkia mahdollisia tietoja tämän ikäisestä autosta ei saada luettua, koska ko. autossa ei ole ajotietokonetta tai mitään lisävarusteita.



Kuva 12. Testiympäristö

4.2 Raspberry Pi 2:n valmistelu

Raspberry Piin kotisivuilta löytyy asennusohjeet englanniksi (Windows-käyttöjärjestelmälle https://www.raspberrypi.org/documentation/installation/installing-images/windows.md), joilla saadaan asennettua käyttöjärjestelmä muistikortille käyttöjärjestelmästä riippumatta.

Ensin ladataan Raspbian-käyttöjärjestelmä tietokoneelle, jossa on microSDHC-kortinlukija osoitteesta https://www.raspberrypi.org/downloads/raspbian/.

Tämän jälkeen Windows-käyttöjärjestelmällä varustetulla tietokoneella ladataan erillinen Win32 Disk Imager -ohjelma (Kuva 13), jolla image saadaan purettua muistikortille.

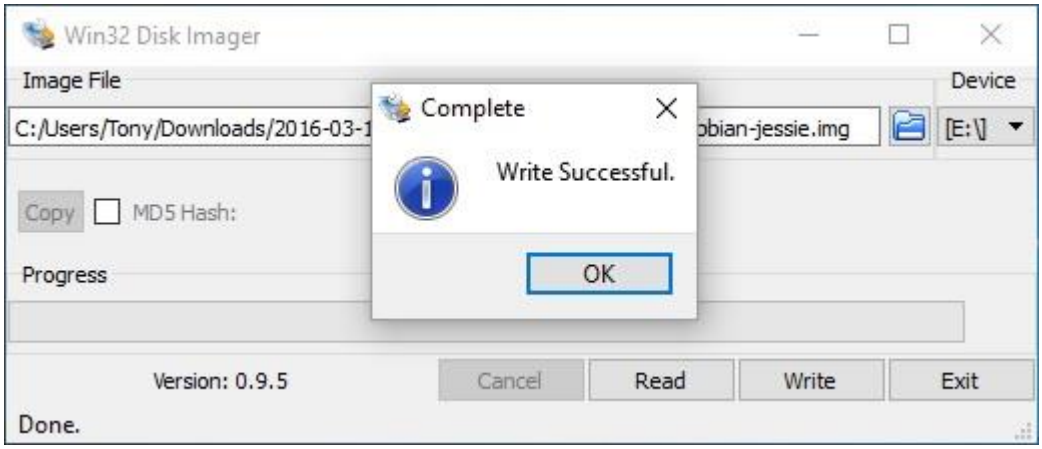
Kuva 13. Win32 Disk Imager -ohjelma

Ohjelmaan valitaan Raspbianin image-tiedosto kansiosta, johon se aikaisemmin ladattiin. Tämän jälkeen kohdasta Device valitaan muistikortin sijainti (E:\). Write-painikkeesta aloitetaan Raspbian-käyttöjärjestelmän asennus muistikortille. Tämän jälkeen asennus kysyy käyttäjältä: ”Oletko varma, että haluat jatkaa?”. Tähän vastataan kyllä, jotta päästään aloittamaan asennus (Kuva 14).

Kuva 14. Uudelleen kirjoituksen varmistaminen

Riippuen tietokoneen ja muistikortin nopeudesta, asennus kestää 5 minuutista puoleen tuntiin.

Kun asennus on valmis (Kuva 15), voidaan muistikortti irrottaa tietokoneesta ja liittää se Raspberry Pi 2 -ohjelmointialustaan.



Kuva 15. Asennus on valmis

Kun asennus on suoritettu, päivitetään Raspbianin tietolähteet (repository), päivitetään kaikki ohjelmat ja ajurit sekä poistetaan turhat tiedostot ja käynnistetään Raspberry Pi 2 uudelleen komennoilla:

- sudo apt-get update
- sudo apt-get upgrade
- sudo apt-get autoremove
- sudo apt-get reboot.

Tämän jälkeen voidaan asentaa tarvittavat ohjelmat:

- sudo apt-get install python-serial
- sudo apt-get install python-wxgtk2.8 python-wxtools wx2.8-i18n libwxgtk2.8-dev
- sudo apt-get install git-core
- sudo reboot.

Seuraavaksi haetaan pyOBD-ohjelma githubin sivuilta https://github.com/Pbartek/pyobd-pi. Ohjelma voidaan joko ladata suoraan sivulta tai hakea se terminaalin kautta komennolla:

- cd ~
- git clone https://github.com/Pbartek/pyobd-pi.git.

Tässä vaiheessa ollaan asennettu kaikki tarvittavat ohjelmat, ajurit ja lisäosat Raspberry Pi 2-alustaan ja voidaan seuraavaksi miettiä OBD2-adapterin asennusta.

4.3 pyOBD

pyOBD, pyOBD-II tai pyOBD2 on OBD-II-yhteensopiva auton diagnostiikkatyökalu. Se on suunniteltu näyttämään graafista työympäristöä ELM 32X OBD-II -adapterei-den lukemalle tiedoille, kuten ELM USB -adapteri (SeCons Ltd 2015).

4.4 ELM327 -adapterin asennus

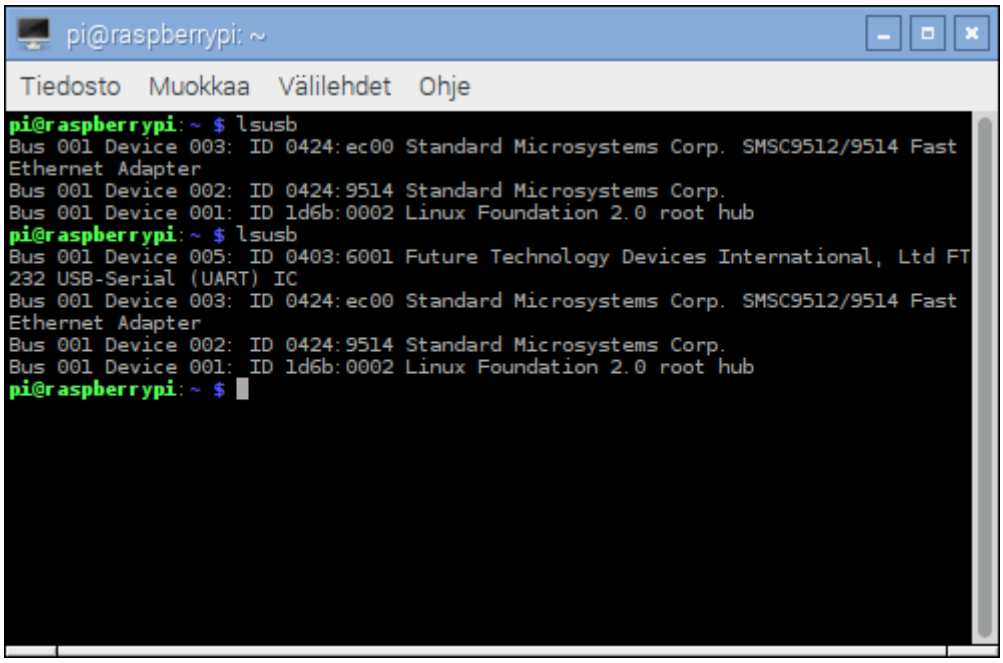
ELM327-adaptereita saa eri yhteysprotokollaa käyttävinä. Seuraavana käydään läpi Bluetooth ja USB-versiot. Saatavilla on myös wifin kautta toimivia adaptereita, mutta niitä ei käsitellä tässä opinnäytetyössä.

4.4.1 ELM327:n USB -versio

Adapterin asennus on todella helppo ja yksinkertainen. Raspbian-käyttöjärjestelmässä on sisään rakennetut ajurit ko. laitteelle, joten asennus tapahtuu PnP-menetelmällä. Laite asennetaan Raspberry Pi 2:n USB-porttiin ja käyttöjärjestelmä hoitaa loput automaattisesti.

Komennolla ”-lsusb” Raspberry Pi 2 -tietokone antaa tulosteena kaikki löytyvät laitteet, jotka on kytkettyinä tietokoneen USB-liittimiin (Kuva 16). Kuten huomataan, ensimmäisen kerran kun ajetaan komento, ei löydy OBD-II-adapteria, koska sitä ei

ole vielä kytketty. Kun toisen kerran ajetaan sama komento, OBD-II-adapteri on kytketty Raspberry Pi 2 -tietokoneeseen ja auton OBD-pistokkeeseen. Laite löytää adapterin numerona 005. Tämä on nyt kytketty OBD-II-adapteri, joka kommunikoi auton yksikön ja Raspberry Pi 2 -koneen kanssa ja välittää tiedot autosta Raspberry Pi 2 -koneelle.



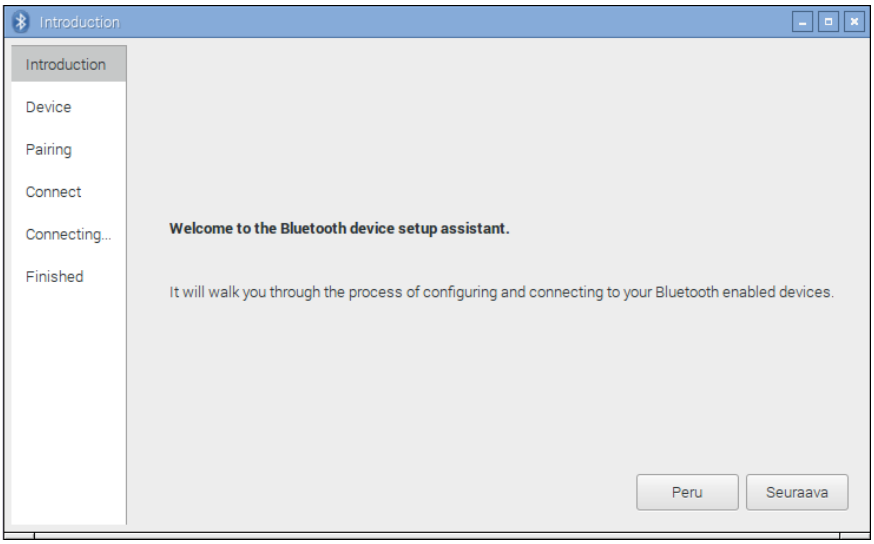
Kuva 16. Raspberry Pi 2:n USB-laitteet

4.4.2 ELM327:n Bluetooth-versio

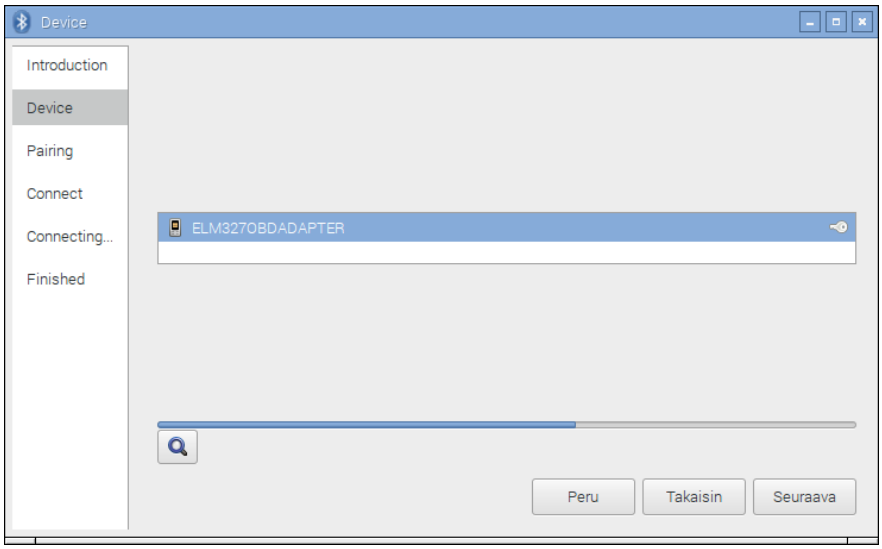
Bluetooth-version asennusta ei voi tehdä ennen kuin OBD-II-adapterissa on virtaa, mutta ohjelmat ja ajurit voidaan asentaa jo etukäteen.

Ensin täytyy ladata Raspbianille seuraavat ohjelma, jotka saadan ladattua komennoilla ”sudo apt-get install bluetooth”, ”sudo apt-get install bluez” ja “sudo apt-get install blueman”.

Tämän jälkeen avataan Bluetooth-valikko ja valitaan Add new device. Näytöllä aukeaa Bluetooth-laitteen asennusohjelma (Kuva 17). Seuraavassa vaiheessa etsitään OBD-II-Bluetooth-adapteri, johon halutaan muodostaa yhteys (Kuva 18). Tämän jälkeen asennusohjelma pyytää antamaan laitteen salasanan (ko. laitteessa oletussalasana on OBDII). Salasanan ollessa oikein asennusohjelma ilmoittaa, että laite lisättiin onnistuneesti. Tämän jälkeen käydään vielä Bluetoothin asetuksista lisäämässä laite luotetuksi laitteeksi, jotta Raspberry Pi 2 -tietokone yhdistäisi siihen automaattisesti, kun havaitsee laitteen olevan päällä ja kantaman sisäpuolella.



Kuva 17. Asennusohjelma



Kuva 18. Löydetty Bluetooth-laite

4.5 ELM327-adapterin toiminnallisuuden testaus

Toiminnan testauksen voi tehdä esimerkiksi asentamalla OBD-II-adapteri autoon ja kääntämällä auton virta-avain ON-asentoon ja käynnistämällä Raspberry Pi 2-alustasta pyOBD-ohjelma terminaalista käsin komennolla:

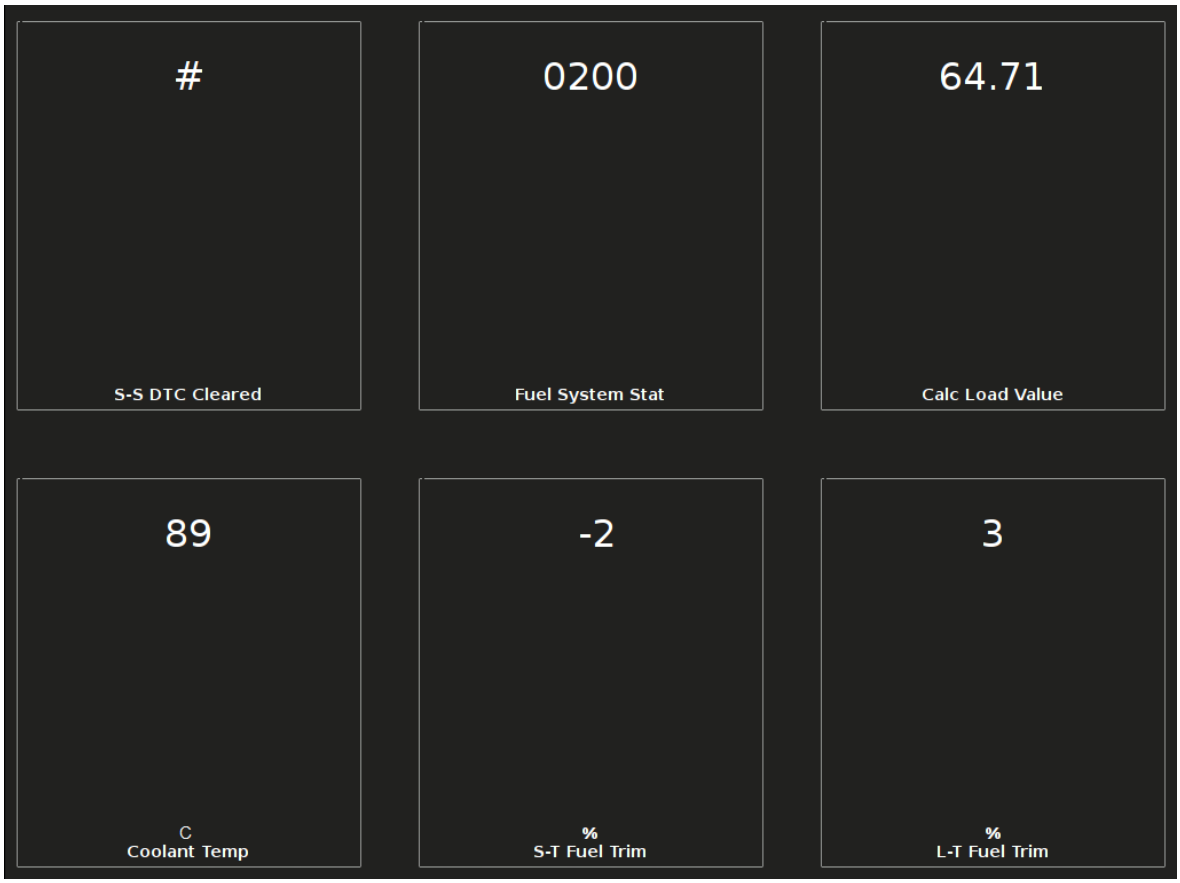
- cd /home/pi/pyobd
- sudo su
- python obd_gui.py.

Nyt pitäisi näytössä näkyä seuraava kuva (Kuva 19).

Kuva 19. pyOBD_gui.py, aloitussivu

Kuvassa 19 on pyOBD-ohjelman graafisen käyttöliittymän aloitussivu. Tällä sivulla voidaan seurata reaaliaikaista dataa autoa ajettaessa. Sivulla näkyvät seuraavien antureiden tiedot: kierrosnopeus, nopeus, sytytyksen ennakko, imuilman lämpötila, MAF-anturin ilmanvirtaus sekä kaasuläpän asento.

Näitä oletussivuja on ohjelmassa kolme kappaletta. Toisella sivulla näkyy muun muassa jäähdytyksen lämpötila ja muiden antureiden arvoja (Kuva 20). Jokaisella sivulla on eri antureiden tiedot oletuksina. Sivuja voi itse muokata haluamakseen, kunhan vain osaa Python-ohjelmointikieltä.



Kuva 20. pyOBD_gui.py, toinen sivu

5 TOTEUTUSTAPA

Ensin ohjelmat ladataan koneelle, koska asennusvaiheessa tiedostojen lataaminen saattaisi kestää todella kauan niiden koosta ja internet yhteyden nopeudesta riippuen.

Seuraavaksi asennetaan Raspberry Pi 2-alustaan Raspbian-käyttöjärjestelmä, joka on Raspberryn virallinen käyttöjärjestelmä. Tämän jälkeen asennetaan käyttöjärjestelmään kaikki mahdolliset päivitykset ja tarvittavat ohjelmat, kuten Python ja sen tarvitsemat kirjastot yms. Sitten asennetaan itse ohjelma, joka keskustelee OBD-II-järjestelmän kanssa, pyOBD. Tämä on ilmainen ohjelma ja se on kirjoitettu Python-kielellä. pyOBD-ohjelman saa ladattua githubista (ja se on vapaasti muokattavissa ja käytettävissä). Tämä ohjelma valitaan sen takia, koska se on helppokäyttöinen ja ilmainen sekä helposti muokattavissa omien tarpeiden mukaan.

Tämän jälkeen, kun kaikki on saatu asennettua Raspberry Pi 2-alustaan, muokataan pyOBD-ohjelmaa sen verran, että saadaan luettua autosta ne tiedot, jotka halutaan. Lisäksi muokataan koodia vielä sen verran, että saadaan muutettua nopeus näyttämään kilometrejä mailien sijasta sekä lämpötilaa kuvattua celsius-asteina fahrenheit-asteiden sijasta.

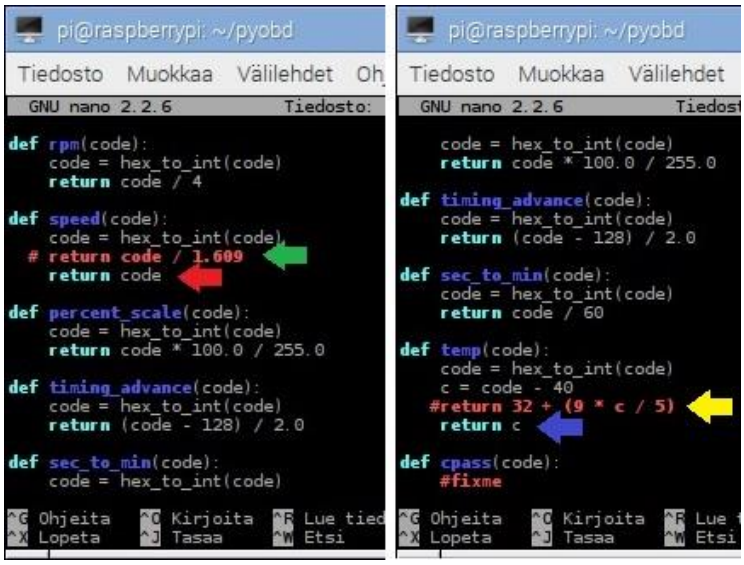
Sitten tehdään C#-kielellä Windows Forms –ohjelma, joka näyttää saadut tiedot graafisina käyrinä. Tämä helpottaa käyttäjää tulkitsemaan tietoja paremmin.

5.1 pyOBD-ohjelman muokkaus

pyOBD-ohjelman asennuksen jälkeen voidaan koodia muokkaamalla muuttaa nopeus maileista kilometreiksi ja fahrenheitasteista celsiusasteiksi pienillä muokkauksilla (Kuva 21).

Kuvassa 21 on merkitty vihreällä nuolella alkuperäinen nopeuden kaava, joka palautti nopeuden arvon maileina tunnissa. Kommentoimalla tämä koodi merkillä # ja tekemällä uusi rivi koodia sen alle (punainen nuoli) saatiin nopeus tulostettua muodossa km/h.

Lämpötilat saatiin fahrenheitasteista celsiusasteiksi kommentoimalla keltaisen nuolen osoittama rivi ja lisäämällä sinisen nuolen osoittama koodirivi. Tämän jälkeen tallennettiin tiedosto ja pyOBD-ohjelman muokkaus oli valmis.



Kuva 21. obd_sensors.py-tiedoston muokkaukset

5.2 Loki-tiedostojen automatisointi

PyOBD-ohjelmalla voidaan automaattisesti tallentaa ajon aikana saadut datat loki-tiedostoon, jota voidaan myöhemmin tarkastella.

Ohjelma tallentaa lokitiedoston oletuksena kansioon /home/username/pyobd-pi/log. Lokitiedoston nimi mukautuu päivämäärän ja ajan mukaan, jotta haluttu loki-tiedosto on helposti haettavissa myös myöhemmin.

pyOBD saadaan automaattisesti tallentamaan tietoja, kun käynnistetään obd_recorder.py-Python-skripti, joka ensin tarkastaa yhteyden ja alkaa sen jälkeen tallentamaan tietoja noin neljä kertaa sekunnissa.

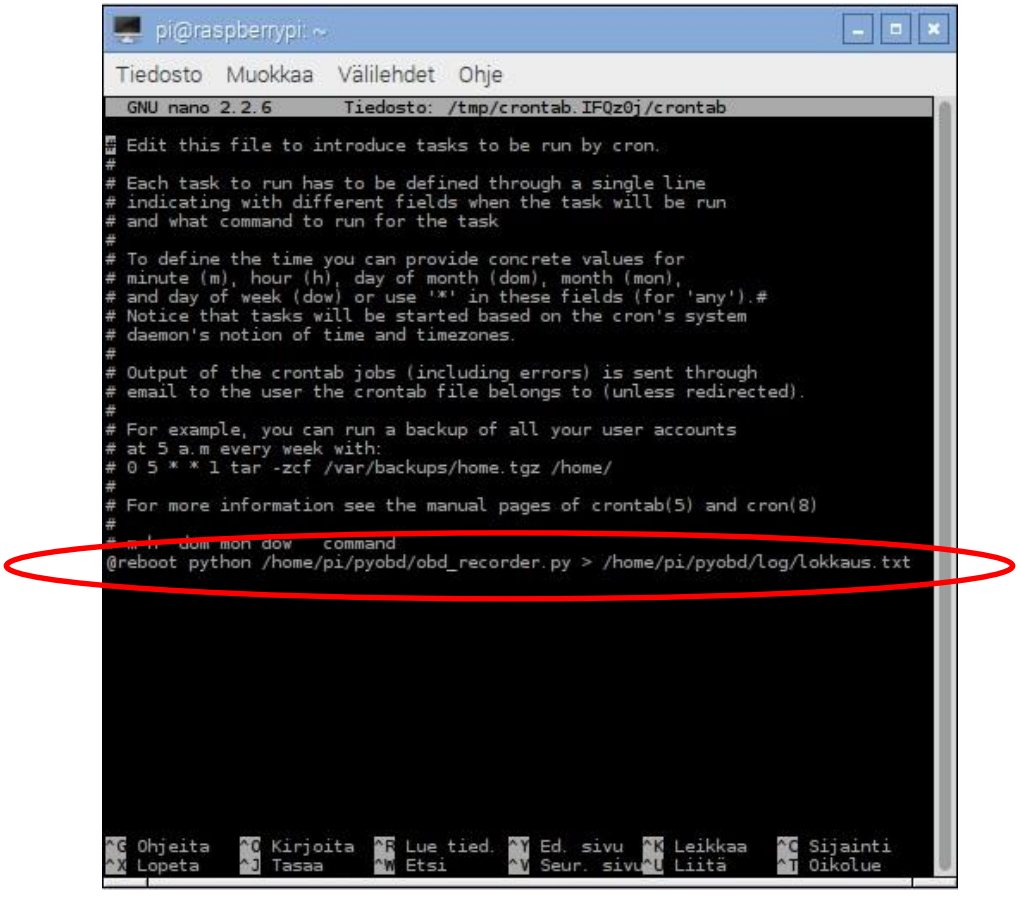
Yksi tapa saada automaattinen tallennus toimimaan, on ajaa obd_recorder.py-skripti aina kun Raspberry Pi 2 -tietokone käynnistetään. Sen voi tehdä esimerkiksi,

kuten tässä työssä, terminaalissa komennolla ”crontab –e”, joka aukaisee cron-nimisen (Kuva 22) ajastuspalvelun tekstitiedoston, jota muokkaamalla voidaan ajastaa Unix-järjestelmien tehtäviä.

Tähän tiedoston viimeisimmäksi riviksi kirjoitetaan seuraava komento:

- @reboot python /home/pi/pyobd/obd_recorder.py > /home/pi/pyobd/log/lokkaus.txt

Tämä komento käynnistää edellä mainitun Python-skriptin aina kun Raspberry Pi 2 -ohjelmointialusta käynnistetään.



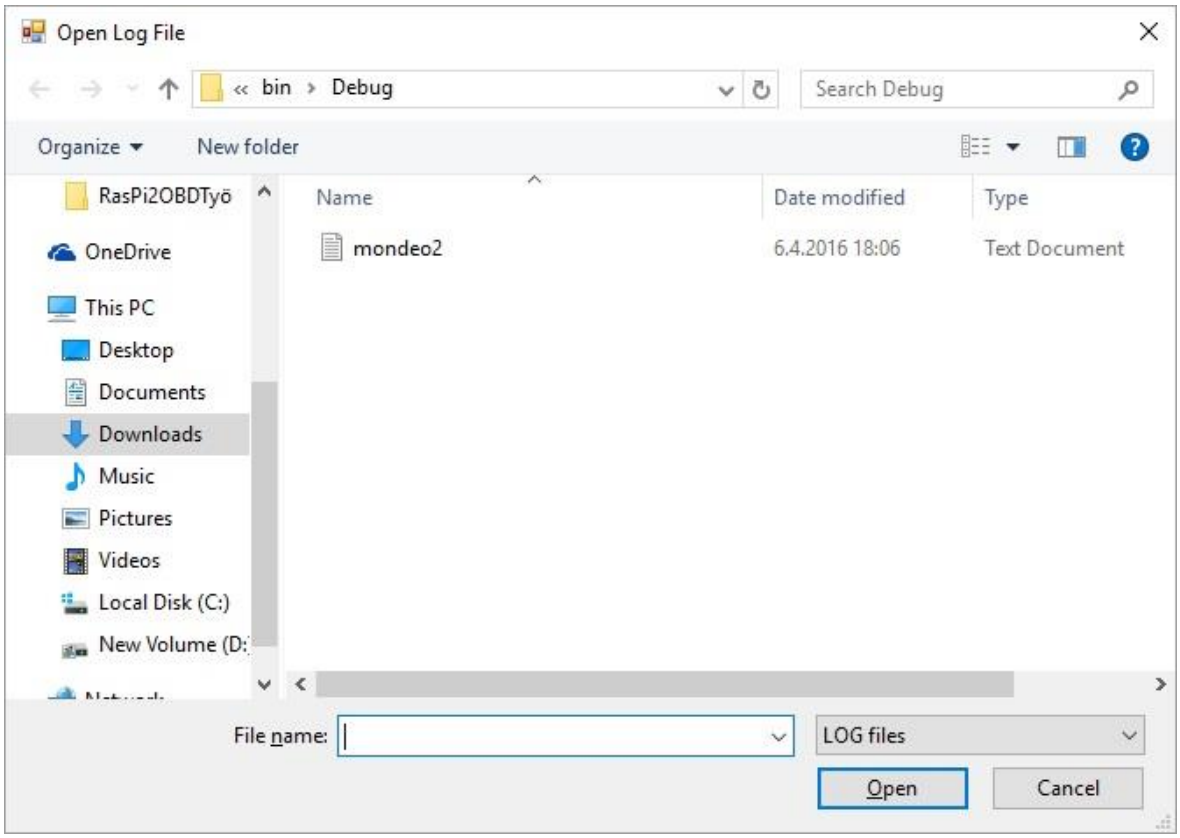
Kuva 22. Crontab

5.3 Windows Forms -projekti

Opittujen ohjelmointitaitojen puitteissa kieleksi valittiin C# ja projekti tehtiin Windows Forms -projektina. Tämän jälkeen alkoi käyttöliittymän suunnittelu ja objektien asettelu, jotta ohjelmasta saadaan helppo ja mukava käyttää (Kuva 23). Objektien sijoittelu on tehty käyttäjää helpottaen, jotta saataisiin mukava käyttöliittymä kokonaisuudessaan. Painikkeet on sijoitettu oletussijoihin, joihin tietokoneen käyttäjät ovat jo tottuneet.

Kuva 23. Windows-ohjelma

Datan tuonti ohjelmaan on toteutettu aukaisemalla käyttäjälle oletuksena kansio, johon lokitiedostot tulevat. Virhetilanteita on karsittu ohjelmoimalla koodi, joka sallii vain .log-tiedostojen avaamisen (Kuva 24). Tähän ohjelmaan eivät muut tiedostotyypit sovi niiden tekstinsijoittelun ja ohjelmoidun koodin vuoksi.



Kuva 24. Lokitiedoston hakeminen

Valittuaan lokitiedoston ohjelma muuntaa lokitiedoston raakadatan käyttäjälle ystävällisemmäksi graafiseksi muodoksi.

Käyttäjä voi tarkastella tuomiaan lokitiedostoja samaan aikaan sekä listamuodossa että diagrammeina. Vietäessä hiiren kursori diagrammin viivan päälle näyttää ohjelma sen hetkisen valitun datan arvon. Klikattaessa diagrammia, kohdistuu listview-komponentti sen hetkisiin arvoihin ja korostaa taustavärillä sen erottumaan käyttäjälle helposti.

6 TULOKSET

Valmista projektia voidaan hyödyntää kotimekaanikon näkökulmasta monella eri tavalla. Kun Raspberry Pi 2 viedään autoon ja liitetään se auton OBD-II-järjestelmään ja käydään ajamassa autolla lenkki, voidaan etsiä autosta mahdollisia vikoja. Windows-ohjelmaan siirretään saatu raakadata ja puretaan se näyttämään graafisia käyriä. Käyristä voidaan katsoa mahdollisia vikoja. Käyrästä huomaa, jos vaikka sytytyksen ajoitus on hetkellisesti mennyt nollaan, vaikka näin ei pitäisi. Tämä helpottaa kotimekaanikon korjaustöitä huomattavasti.

Raspberry Pi voidaan asentaa autoon myös kiinteästi ja hyödyntää sen kaikki mahdollisuudet. Raspberry Piihin voidaan asentaa esim. peruutuskamera, navigointi tai muita lisälaitteita ja yhdistää ne Raspberry Piihin. Näin saadaan Raspberry Pi-alustasta tehtyä autoon monipuolinen viihdejärjestelmä monipuolisine ominaisuuksineen.

7 POHDINTAA JA YHTEENVETO

Tässä opinnäytetyössä tehdyn Raspberry Pi -kokonaisuuden voisi asentaa uudempaan autoon paljon helpommin. Uusissa autoissa on usein näytöt valmiina ja niihin on yleensä sisääntulo video-signaalille, jota voisi käyttää Raspberry Pi 2-alustan signaalin tuomiseen auton integroidulle näytölle. Virta voitaisiin ottaa suoraan auton savukkeensytyttimestä sopivalla adapterilla. OBD-II-Bluetooth-laite voitaisiin asentaa auton OBD-pistokkeeseen pysyvästi. Näin saataisiin helposti piilotettava kokonaisuus, joka kuitenkin loisi autoon täydellisen viihdekeskuksen keskusyksikön. Lisäominaisuuksina voitaisiin asentaa Raspberry Pi 2-alustaan vielä XBMC-lisäosa, jolla voidaan toistaa videoita, musiikkia, valokuvia sekä selata internetiä. Jos autoon integroitu radio ei täyty odotuksia, voidaan Raspberry Pi -laitteeseen asentaa navigointi, radio yms. sovellukset ja korvata koko auton oma järjestelmä.

Myös peruutuskameran asentaminen olisi helppoa nykyautoon. Raspberry Pi 2 voitaisiin ohjelmoida automaattisesti kytkemään peruutuskamera päälle ja näyttämään sen videokuva näytöstä, kun vaihdetaan peruutusvaihteelle. Tämä voidaan toteuttaa lukemalla OBD-II-järjestelmästä nykyisen vaihteen tila. Jos vaihteen tila muuttuu peruutusvaihteeksi, kytkettäisiin kamera päälle ja näytettäisiin videokuva näytöltä.

Vanhemmassa autossa, kuten Ford Mondeon vuosimallissa 1997, ei saada OBD-II-järjestelmä vastaavaa signaalia. Tämä voitaisiin tehdä vetämällä Raspberry Pi –tietokoneelle suoraan vaihdelaatikon anturilta tai vaihtoehtoisesti peruutusvaloilta lisäkaapeli. Tältä kaapelilta saataisiin signaali, jota voitaisiin käyttää kytkemään peruutuskamera päälle.

Windows-ohjelmaan voitaisiin vielä lisätä ominaisuus, joka näyttäisi listview-komponentin valitun tallennusajan perusteella chart-komponentissa punaisen pystyviivan. Näin saataisiin todella tarkasti tarkasteltua valitun ajanhetken arvoja.

LÄHTEET

Alpha-Bid. 2016. OBD2 Communication Protocols by Manufacturer. [pdf-lähde]. Alpha-Bid. [Viitattu 9.1.2016]. Saatavana: http://www.alpha-bid.com/media/Shared-Pics/OBD2_protocols.pdf

ARM Ltd. Ei päiväystä. Cortex-A7 Processor. [www-lähde]. ARM Ltd. [Viitattu 23.12.2015]. Saatavana: http://www.arm.com/products/processors/cortex-a/cortex-a7.php

B & B Electronics. 2011a. Does My Car Have OBD-II?. [www-lähde]. B & B Electronics. [Viitattu 16.11.2015]. Saatavana: http://www.obdii.com/connector.html

B & B Electronics. 2011b. OBD-II Background Information. [www-lähde]. B & B Electronics. [Viitattu 16.11.2015]. Saatavana: http://www.obdii.com/background.html

Microsoft Corp. 2015. Learn About Windows 10 IoT Core. [www-lähde]. Microsoft Corp. [Viitattu 14.2.2016]. Saatavana: http://ms-iot.github.io/content/en-US/IoTCore.htm

Raspberry Pi Foundation. Ei päiväystä. About us. [www-lähde]. Raspberry Pi Foundation. [Viitattu 13.4.2016]. Saatavana: https://www.raspberrypi.org/about/

Raspberry Pi Foundation. Ei päiväystä. Raspberry Pi 2 Model B. [www-lähde]. Raspberry Pi Foundation. [Viitattu 12.1.2016]. Saatavana: https://raspberrypi.org/products/raspberry-pi-2-model-b/

Raspberry Pi Foundation. Ei päiväystä. Raspberry Pi 2 on sale now at $35. [www-lähde]. Raspberry Pi Foundation. [Viitattu 25.4.2016]. Saatavana: https://www.raspberrypi.org/blog/raspberry-pi-2-on-sale/

Raspberry Pi Foundation. Ei päiväystä. Raspberry Pi 3 Model B. [www-lähde]. Raspberry Pi Foundation. [Viitattu 6.5.2016]. Saatavana: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

Raspberry Pi Foundation. Ei päiväystä. Raspberry Pi Zero. [www-lähde]. Raspberry Pi Foundation. [Viitattu 1.5.2016]. Saatavana: https://www.raspberrypi.org/products/pi-zero/

Raspbian. Ei päiväystä. Welcome to Raspbian. [www-lähde]. Raspbian. [Viitattu 1.5.2016]. Saatavana: https://www.raspbian.org/

SeCons Ltd. 2015. pyOBD. [www-lähde]. SeCons Ltd. [Viitattu 9.1.2016]. Saatavana: http://www.obdtester.com/pyobd

Trafi. 2011. Ottomoottorikäyttöisten ajoneuvojen pakokaasupäästöjen tarkastus [pdf-lähde]. Liikenteen turvallisuusvirasto. [Viitattu 02.02.2016]. Saatavana: http://www.trafi.fi/file-bank/a/1325147177/3fb9d1c954c8aab89c3d40b8cce5ca26/4756-Ottomoottori-kayttoisten_pakokaasupaastojen_tarkastus.pdf

Upton, L. 27.11.2015. Did you get a Raspberry Pi Zero?. [www-kuva]. Raspberry Pi Foundation. [Viitattu 5.5.2016]. Saatavana: https://www.raspber-rypi.org/blog/did-you-get-a-raspberry-pi-zero/

LIITTEET

Liite 1. OBD2 Communication Protocols by Manufacturer

Liite 2. OBD-II-loggaajaohjelman lähdekoodi

Liite 1. OBD2 Communication Protocols by Manufacturer [Alpha-Bid 2016]

OBD2 Communication Protocols by Manufacturer

Make - Model	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008
Audi/VW	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO9141 KWP2000	ISO9141 KWP2000	ISO9141 KWP2000	ISO9141 KWP2000 CAN	ISO9141 KWP2000 CAN	ISO9141 KWP2000 CAN	CAN
Bentley	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141 (50%) KWP2000 (50%)	ISO 9141 (35%) KWP2000 (65%)	ISO 9141 (20%) KWP2000 (80%)	CAN	CAN
BMW - Mini	N/A	N/A	N/A	N/A	N/A	N/A	KWP2000	KWP2000	KWP2000	KWP2000	N/A	N/A	CAN
BMW 3-Series, 5-Series, X5, Z3, Z4, Z8, 740i, 740iL, 750iL	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	CAN
BMW 5-Series; next generation (E60)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	KWP2000	KWP2000	N/A	N/A	CAN
BMW 745i, 745Li, 765Li	N/A	N/A	N/A	N/A	N/A	N/A	KWP2000	KWP2000	KWP2000	KWP2000			
Chrysler	ISO 9141 (100%)	ISO 9141 (100%)	ISO 9141 (95%) J1850-10.4 (5%)	ISO 9141 (85%) J1850-10.4 (15%)	ISO 9141 (75%) J1850-10.4 (25%)	ISO 9141 (35%) J1850-10.4 (65%)	ISO 9141 (15%) J1850-10.4 (85%)	ISO 9141 (5%) J1850-10.4 (95%)	CAN (5%) J1850-10.4 (95%)	CAN (15%) J1850-10.4 (85%)	CAN (35%) J1850-10.4 (65%)	CAN (85%) J1850-10.4 (15%)	CAN
Daewoo-Lanos	N/A	N/A	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	N/A					
Daewoo-Leganza	N/A	N/A	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	KWP2000	KWP2000					
Daewoo-Nubira	N/A	N/A	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	KWP2000	KWP2000					
Ford (w/o exceptions)	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	5% CAN	5% CAN	50% CAN	85% CAN	90% CAN	CAN	CAN
exceptions:													
Aspire	ISO 9141	ISO 9141											
Escort 1.8L	ISO 9141												
Probe 2.5L	ISO 9141	ISO 9141											
Villager	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141						
GM (w/o exceptions)	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	80% J1850-10.4 15% CAN 5% KWP2000	55% J1850-10.4 40% CAN 5% KWP2000	25% J1850-10.4 70% CAN 5% KWP2000	25% J1850-10.4 70% CAN 5% KWP2000	CAN
exceptions:													
Saturn LS 3.0L	N/A	N/A	N/A	N/A	KWP2000	KWP2000	KWP2000	KWP2000					
Saturn VUE 3.0L	N/A	N/A	N/A	N/A	N/A	N/A	KWP2000	KWP2000					
Saturn VUE 3.5L	N/A	N/A	N/A	N/A	N/A	N/A	N/A	KWP2000					
Saturn ION	N/A	N/A	N/A	N/A	N/A	N/A	N/A	CAN					
Cadillac Catera	N/A	KWP2000	KWP2000	KWP2000	KWP2000	KWP2000	N/A	N/A					
Cadillac CTS	N/A	N/A	N/A	N/A	N/A	N/A	N/A	KWP2000					
GEO Metro	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	N/A	N/A					
GEO Prism	J1850-10.4	J1850-10.4	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	N/A					
GEO Tracker	ISO 9141	ISO 9141	ISO 9141	KWP2000	KWP2000	KWP2000	KWP2000	KWP2000					
Pontaic Vibe	N/A	N/A	N/A	N/A	N/A	N/A	N/A	ISO 9141					
Saab 9-3	N/A	N/A	N/A	N/A	N/A	N/A	N/A	CAN					
Honda/Acura	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	70% ISO 9141 30% CAN	40% ISO 9141 60% CAN	CAN
Hyundai	ISO 9141	ISO 9141	ISO 9141	KWP2000	ISO 9141 KWP2000	ISO 9141 KWP2000	ISO 9141 KWP2000	ISO 9141 KWP2000	ISO 9141 KWP2000	ISO 9141 KWP2000	ISO 9141 KWP2000	ISO 9141 KWP2000	CAN
Isuzu	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	J1850-10.4	N/A	N/A	CAN
Jaguar	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141		
Kia - Optima	N/A	N/A	N/A	N/A	N/A	KWP 2000	KWP 2000	KWP 2000	KWP 2000	KWP 2000			
Kia - Rio	N/A	N/A	N/A	N/A	N/A	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141			
Kia - Sephia	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	N/A	N/A	N/A	N/A	CAN
Kia - Spectra	N/A	N/A	N/A	N/A	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141			
Kia - Sportage	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141			
Kia Sedona	N/A	N/A	N/A	N/A	N/A	N/A	ISO 9141	ISO 9141	ISO 9141	ISO 9141			
Kia Sorento	N/A	N/A	N/A	N/A	N/A	N/A	N/A	KWP 2000	KWP 2000	KWP 2000			
Land Rover	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	0% CAN	0% CAN	0% CAN	62% CAN	CAN	CAN	CAN
Mazda	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	75%-IS0 9141 25%-CAN	60%-CAN 40%-ISO 9141	60%-CAN 40%-ISO 9141	CAN	CAN	CAN
Mazda-Ford	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	J1850-41.6	50%-J 1850 50%-CAN	50%-J 1850 50%-CAN	Tribute-CAN Truck?	CAN
Mercedes	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141 KWP2000	ISO 9141 KWP2000	ISO 9141 KWP2000	KWP2000	KWP2000	KYP2000 CAN	KYP2000 CAN	KYP2000 CAN	CAN
Mitsubishi	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141 CAN	N/A	CAN
Nissan/Infiniti	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	N/A	CAN
Porsche	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141					
Rolls Royce	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	KWP 2000	KWP 2000	KWP 2000	N/A	CAN
Subaru	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141 KWP 2000	ISO 9141 KWP 2000	KWP 2000	CAN	CAN	CAN
Suzuki Aerio 2.0L	N/A	N/A	N/A	N/A	N/A	N/A	KWP 2000	KWP 2000	KWP 2000	KWP 2000	KWP 2000	CAN	CAN
Suzuki Esteem 1.6L	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	N/A	N/A	N/A	CAN
Suzuki Esteem 1.8L	N/A	N/A	N/A	ISO 9141	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	N/A	N/A	N/A	CAN
Suzuki Grand Vitara 2.5L	N/A	N/A	N/A	ISO 9141	ISO 9141	ISO 9141	ISO 9141	KWP 2000	KWP 2000	KWP 2000	KWP 2000	CAN	CAN
Suzuki Grand Vitara 2.7L	N/A	N/A	N/A	N/A	N/A	KWP 2000	KWP 2000	KWP 2000	KWP 2000	KWP 2000	KWP 2000	CAN	CAN
Suzuki Sidekick 1.6L	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	CAN
Suzuki Sidekick 1.8L	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	CAN
Suzuki Swift 1.3L	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	N/A	N/A	N/A	N/A	CAN
Suzuki Vitara 1.6L	N/A	N/A	N/A	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	N/A	N/A	N/A	N/A	CAN
Suzuki Vitara 2.0L	N/A	N/A	N/A	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	CAN	CAN
Suzuki X90 1.6L	ISO 9141	ISO 9141	ISO 9141	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	CAN
Toyota/Lexus/Scion	J1850-10.4	J1850-10.4 ISO 9141	J1850-10.4 ISO 9141	J1850-10.4 ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141 CAN	ISO 9141 CAN	ISO 9141 CAN	N/A	CAN
Volvo	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	ISO 9141	90% ISO 9141 10% CAN	5% ISO 9141 95% CAN	CAN	CAN	CAN

Liite 2. OBD-II-loggaajaohjelman lähdekoodi

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.Windows.Forms.DataVisualization.Charting;
using System.Windows.Forms.DataVisualization.Charting.Data;

namespace OBD_II_loggaaja
{
    public partial class Form1 : Form
    {
        ListViewItem lastItem;
        string[] otsikot;

        public Form1()
        {
            InitializeComponent();
            ChartArea CA = chart1.ChartAreas[0];
            CA.AxisX.ScaleView.Zoomable = true;
            CA.CursorX.AutoScroll = true;
            CA.CursorX.IsUserSelectionEnabled = true;
            CA.AxisY.ScaleView.Zoomable = true;
            CA.CursorY.AutoScroll = true;
            CA.CursorY.IsUserSelectionEnabled = true;
        }


        private void Form1_Load(object sender, EventArgs e)
        {
            listView1.View = View.Details;
            listView1.LabelEdit = true;
            listView1.AllowColumnReorder = true;
            listView1.CheckBoxes = false;
            listView1.FullRowSelect = true;
            listView1.GridLines = true;
        }


        private void timer1_Tick(object sender, EventArgs e)
        {
            label6.Text = DateTime.Now.ToString();
        }


        private void chart1_MouseMove(object sender, MouseEventArgs e)
        {
            foreach (Series s in chart1.Series)
            {
                chart1.Series[s.Name].ToolTip = s.Name + ": #VALY";
            }
        }
```

```
        private void poistuOhjelmastaToolStripMenuItem_Click(object sender, Even-
tArgs e)
        {
            this.Close();
        }


        private void muunnaDiagrammiksiToolStripMenuItem_Click(object sender, Even-
tArgs e)
        {
            chart1.Series.Clear();
            if (listView1.Items.Count == 0)
            {
                MessageBox.Show("Hae ensin lista!");
            }
            else
            {
                for(int i = 1; i < otsikot.Length; i++)
                {
                    chart1.Series.Add(otsikot[i]);
                    chart1.Series[otsikot[i]].ChartType = SeriesChartType.Line;
                    for (int x = 0; x < listView1.Items.Count; x++)
                    {
                        chart1.Series[otsi-
kot[i]].Points.AddXY(listView1.Items[x].SubItems[0].Text, listView1.Items[x].Sub-
Items[i].Text);
                    }
                }
                MessageBox.Show("Antureiden lukemat tiedot muutettu diagrammiksi!" +
"\n\n" + "Diagrammia voi suurentaa maalaamalla hiirellä haluttu alue.");
            }
        }

        private void haeLokitiedostoToolStripMenuItem1_Click(object sender, Even-
tArgs e)
        {
            OpenFileDialog theDialog = new OpenFileDialog();
            theDialog.Title = "Avaa lokitiedosto";
            theDialog.Filter = "Lokitiedostot | *.log";
            theDialog.InitialDirectory = @".\";


            try
            {
                if (theDialog.ShowDialog() == DialogResult.OK)
                {
                    FileStream textFile = new FileStream(theDialog.FileName, File-
Mode.OpenOrCreate, FileAccess.ReadWrite);
                    StreamReader reader = new StreamReader(textFile);

                    listView1.Clear();

                    string line1 = reader.ReadLine(); ;
                    otsikot = line1.Split(',');

                    foreach (string s in otsikot)
                    {
                        listView1.Columns.Add(s, 100, HorizontalAlignment.Left);
                    }

                    listView1.View = View.Details;
                    int count = 1236 / listView1.Columns.Count;
```

```
                    for (int i = 0; i < listView1.Columns.Count; i++)
                    {
                            listView1.Columns[i].AutoResize(ColumnHeaderAuto-
ResizeStyle.None);
                            listView1.Columns[i].Width = count;
                    }

                    string line = "";
                    string[] items;
                    ListViewItem listItem;

                    while ((line = reader.ReadLine()) != null)
                    {
                        items = line.Split(',');
                        listItem = new ListViewItem();

                        for (int i = 0; i < items.Length; i++)
                        {
                            if (i == 0)
                            {
                                listItem.Text = items[i];
                            }
                            else
                            {
                                listItem.SubItems.Add(items[i]);
                            }
                        }

                        listView1.Items.Add(listItem);
                    }

                    reader.Close();
                    textFile.Close();

                }
            }
            catch (Exception ex)
            {
                MessageBox.Show("Virhe: Ei voitu lukea tiedostoa levyltä.\n Alkupe-
räinen virhe: " + ex.Message);
            }
        }

        private void haeAnturitiedototsikotToolStripMenuItem_Click(object sender,
EventArgs e)
        {

            OpenFileDialog theDialog = new OpenFileDialog();
            theDialog.Title = "Avaa lokitiedosto";
            theDialog.Filter = "Lokitiedostot | *.log";
            theDialog.InitialDirectory = @".\";


            try
            {
                if (theDialog.ShowDialog() == DialogResult.OK)
                {
                    FileStream textFile = new FileStream(theDialog.FileName, File-
Mode.OpenOrCreate, FileAccess.ReadWrite);
                    StreamReader reader = new StreamReader(textFile);
```

```
                    string line1 = reader.ReadLine(); ;
                    string[] otsikot = line1.Split(',');
                    string line = "";
                    int i = 1;
                    foreach (string s in otsikot)
                    {
                        line = line + "\n" + i + ". " + s;
                        i++;
                    }

                    MessageBox.Show(line);

                    reader.Close();
                    textFile.Close();
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show("Virhe: Ei voitu lukea tiedostoa levyltä. Alkuperäi-
nen virhe: " + ex.Message);
            }
        }

        private void palautaDiagrammiToolStripMenuItem1_Click(object sender, Even-
tArgs e)
        {
            this.chart1.ChartAreas[0].AxisX.ScaleView.ZoomReset(0);
            this.chart1.ChartAreas[0].AxisY.ScaleView.ZoomReset(0);
        }

        private void tyhjennäToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            listView1.Clear();

            foreach (var series in chart1.Series)
            {
                series.Points.Clear();
            }
        }

        private void tallennaDiagrammiToolStripMenuItem_Click(object sender, Even-
tArgs e)
        {
            saveFileDialog1.ShowDialog();
        }

        private void saveFileDialog1_FileOk(object sender, CancelEventArgs e)
        {
            string name = saveFileDialog1.FileName;
            this.chart1.SaveImage(name + ".png", ChartImageFormat.Png);
        }


        private void chart1_MouseClick(object sender, MouseEventArgs e)
        {
            HitTestResult result = chart1.HitTest(e.X, e.Y);
            if (result.ChartElementType == ChartElementType.DataPoint)
            {
                var selectedValue = chart1.Series[0].Points[result.PointIn-
dex].AxisLabel;
                ListViewItem foundItem = listView1.FindItemWithText(selectedValue,
false, 0, true);
```

```
                    if (foundItem != null)
                    {
                        listView1.TopItem = foundItem;
                        if (foundItem.Text == selectedValue)
                        {
                            if (lastItem != null)
                                lastItem.BackColor = Color.Transparent;
                            foundItem.BackColor = Color.LightSteelBlue;
                            lastItem = foundItem;
                            return;
                        }

                    }
                }
            }
        }
    }
```