Tuisku Tomi

CAN-väylä

Raskaankaluston standardi SAE J1939

Opinnäytetyö
Kevät 2012
Tekniikan yksikkö
Tietotekniikka koulutusohjelma
Sulautetut järjestelmät

Seinäjoen ammattikorkeakoulu
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Tekniikan yksikkö

Koulutusohjelma: Tietotekniikan koulutusohjelma

Suuntautumisvaihtoehto: Sulautetut järjestelmät

Tekijä: Tuisku, Tomi

Työn nimi: CAN-väylä: Raskaankaluston standardi SAE J1939

Ohjaaja: Palomäki, Heikki

Vuosi: 2012 Sivumäärä: 33 Liitteiden lukumäärä: 1

Tässä opinnäytetyössä tutkitaan CAN-väylän toimintaa sekä siihen perustuvaa raskaankaluston sovellutusta SAE J1939 -standardia. Työssä kerrotaan CAN-väylän perusteita sekä perehdytään J1939-standardin mukaiseen sanomakehykseen.

CAN-väylä on nykyisin yleinen ajoneuvoväylä, jota käytetään usein jonkin muun väylän rinnalla. Lisääntyneen elektroniikan vuoksi CAN-väylä tulee todennäköisesti kasvattamaan suosiotaan entisestään. CAN-väylässä liikkuvat kaikki tiedot, mitä ajoneuvon elektroniset ohjainyksiköt tarvitsevat. Kaikki laitteet, jotka tarvitsevat jonkin tiedon, voivat lukea sen yhtä aikaa. Väylään voidaan lisätä laitteita lukemaan tietoa.

Työn tarkoituksen on tutustua CAN-väylään sekä kuinka mikroprosessorilla voidaan lukea ajoneuvon tietoja. Työ tehtiin Deware Oy:lle tutkimuksena kuinka nykyisen seurantalaitteen voisi liittää lukemaan työsuoritteen optimointiin tarvittavia tietoja työkoneen CAN-väylästä.

Työn tuloksena on selvitys CAN-väylän sanomakehyksen rakenteesta ja tieto mistä löytyy J1939-standardin mukaiset parametrien määritykset, ilman määrityksiä on mahdotonta yrittää lukea väylältä haluamaansa tietoa. Lisäksi tutkittiin mitä tarvitaan mikroprosessorin liittämiseksi CAN-väylään.

Avainsanat: CAN-väylä, SAE J1939

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree programme: Information Technology

Specialisation: Embedded Systems

Author: Tomi Tuisku

Title of the thesis: CAN bus: SAE J1939 standard for heavy duty vehicles

Supervisor: Heikki Palomäki

Year: 2012 Number of pages: 33 Number of appendices: 1

The aim of this thesis was to introduce the operation of a CAN bus, and the SAE J1939 standard. The J1939 standard is a CAN bus application for heavy duty vehicles.

A CAN bus delivers all data that a vehicle's electronic control units need. All electronic control units which are connected to the bus can read data simultaneously. More electronic control units to read data can be added to a CAN bus.

The purpose of this thesis was to introduce the CAN bus, and find out how information can be read from a CAN bus with microchip. And also research how a microchip must be connected to a CAN bus. The thesis was commissioned by Deware Oy, a tracking device manufacturer, to research how a tracking device can be added to read information from a heavy duty vehicle's CAN bus.

The result of the thesis was a report of the CAN bus message frame, and information about where the J1939 standard parameter determinations can be found. Without these determinations it is almost impossible to read the required information.

Keywords: CAN bus, SAE J1939

SISÄLTÖ

KÄYTETYT TERMIT JA LYHENTEET

ACK	Acknowledge.
CAN-H	CAN-high väyläjohto.
CAN-L	CAN-low väyläjohto.
CAN	Controller Area Network.
CRC	Cyclic Redundancy Check.
DLC	Data Length Code.
DP	Data page.
EOF	End Of Frame.
ID	Identifier.
IDE	Identifier Extension.
IFS	Interframe Space, kenttä joka erottaa kaikki CAN-viesti toisistaan.
J1939	CAN-väylään pohjautuva sovelluskerroksen protokolla, jota käytetään kuorma-autoissa.
PDU	Protocol Data Unit.
PF	PDU Format.
PGN	Parameter Group Number, parametriryhmän tunniste numero.
PS	PDU Specific.
R	Reserved, tulevaisuuden käyttöön varattu.

RTR	Remote Transmission Request.
SAE	Society Of Automotive Engineers, autoalan standardisointi järjestö.
SOF	Starf Of Frame.
Solmu	CAN-väylään liitetty laite.
SPN	Suspect Parameter Number.

KUVIO- JA TAULUKKOLUETTELO

1 JOHDANTO

1.1 Aiheen esittely

CAN-väylä on suunniteltu 80-luvun alkupuolella vähentämään ajoneuvoissa tarvittavaa johdotusta lisääntyneen elektroniikan vuoksi. Nykyään ajoneuvoissa, niin henkilöautoissa kuin isoissa työkoneissakin, on paljon elektroniikkaa. Ajoneuvoissa erilaisilta antureilta mitatut arvot välitetään CAN-väylää pitkin kaikille elektronisille ohjainyksiköille, jotka kyseistä tietoa tarvitsevat.

Vuodesta 2008 eteenpäin CAN-väylä on ainoa diagnostiikassa hyväksytty väylätyyppi. Ajoneuvoissa voi olla myös muita väyliä samanaikaisesti käytössä, mutta CAN-väylän kautta täytyy pystyä lukemaan katsastuksessa muun muassa saastetiedot. Tämän vuoksi useissa ajoneuvoissa diagnostiikka tapahtuu CAN-väylän kautta. (Heikkilä 2007.)

Tämän opinnäytetyö idea tuli Deware Oy:ltä, jolla on kehitteillä työkoneiden seurantajärjestelmä. Seurantalaitteen tulisi lukea työkoneiden käytön optimointiin liittyvää tietoa ajoneuvojen CAN-väylästä.

1.2 Työn tavoite

Tässä opinnäytetyössä on tarkoitus perehtyä yleisesti CAN-väylän toimintaan sekä raskaankaluston käyttämään J1939-standardiin. Päätavoite on selvittää miten työkoneen CAN-väylästä pystytään lukemaan tarvittavia tietoja, kuten moottorin kierrosnopeus ja ajoneuvon nopeus. Lisäksi halutaan tutkia mitä muutoksia seurantalaitteen prototyyppi vaatisi, että sillä voisi lukea tietoa CAN-väylästä. Toissijaisena tavoitteena on rakentaa yksinkertainen mikrokontrollerilla toimiva testilaite, joka lukee CAN-väylästä tietoa.

1.3 Työn rakenne

Toisessa luvussa tutustutaan CAN-väylän toimintaan.

Kolmannessa luvussa tutustutaan CAN-väylään pohjautuvaan raskaassa kalustossa käytettävään J1939-standardiin.

Neljännessä luvussa esitellään mikroprosessorin liittäminen CAN-väylään sekä pohditaan mitä mikroprosessoria voitaisiin käyttää seurantalaitteessa.

Viimeisessä luvussa käydään läpi työn tulokset ja yhteenveto.

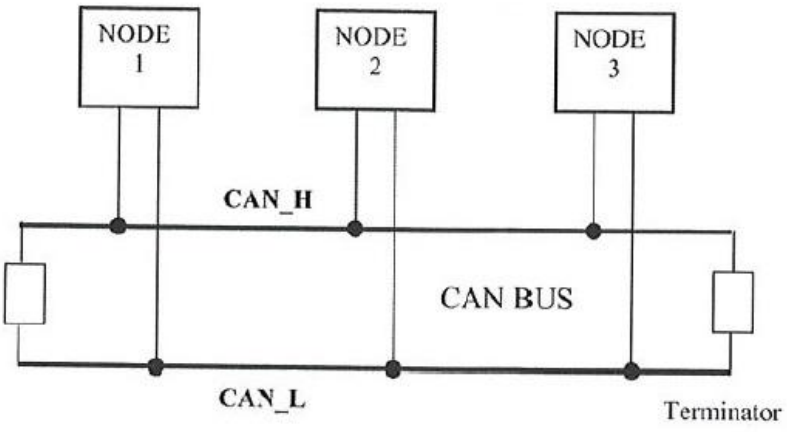
2 CAN-VÄYLÄ

CAN-väylä on Robert Bosch GmbH:n 1980-luvun alussa suunnittelema sarjamuotoinen verkkotekniikka ajoneuvoteollisuuden käyttöön. Myöhemmin siitä on tullut suosittu myös teollisuusautomaatiossa. Yleensä CAN-väylää käytetään sulautetuissa järjestelmissä nopeaan tiedon siirtoon mikrokontrollereiden välillä. (Wilfried 2005, 2-3.)

CAN-väylä on moni-isäntäinen väylä, jokainen solmu voi lähettää tietoa väylälle aina kun se on vapaa. Kaikilla viesteillä on oma tunnistenumero, joka määräytyy viestin sisällön mukaan. Kaikki solmut, jotka viestiä tarvitsevat, ottavat sen vastaan. CAN-väylällä ei siis lähetetä viestiä millekään tietylle solmulle. (Alanen 2000, 4-5.)

2.1 Fyysinen kerros

CAN-väylä koostuu kahdesta kaapelista, jotka ovat nimetty CAN-H ja CAN-L, väyläkaapelit kulkevat jokaisen solmun kautta. Yleisesti CAN-väylässä käytetään kierrettyä parikaapelia, jossa on molemmissa päissä 120 ohmin päätevastukset vähentämässä heijastuksia (kuvio 1). (Dogan 2011, 45.)

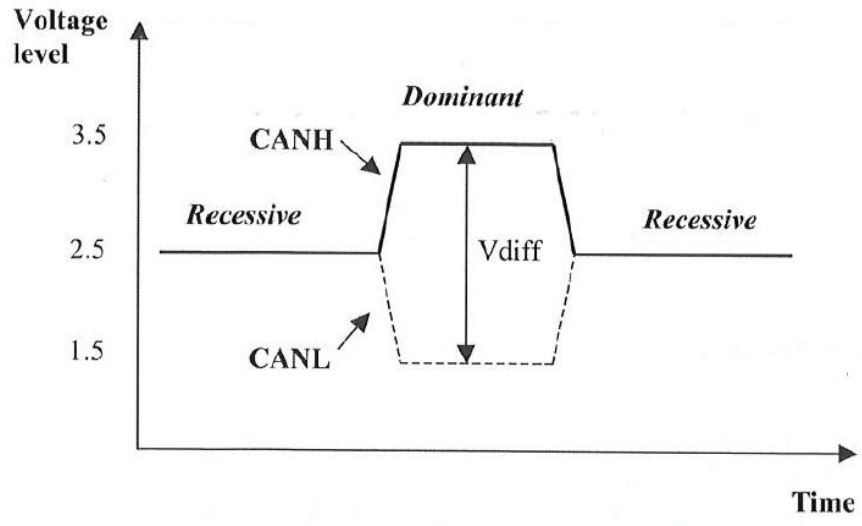


Kuvio 1: CAN-väylän periaatekuva kolmella solmulla (Dogan 2011, 45).

CAN-väylän kaapelin pituus riippuu väylässä käytetystä nopeudesta. Väylän nopeutta voidaan muuttaa ohjelmallisesti 10 kbit/s ja 1 Mbit/s välillä. Käytettäessä 1 Mbit/s tiedonsiirtonopeutta väylän maksimipituus on 40 m, hitaammalla 10 kbit/s nopeudella voidaan käyttää maksimissaan 6700 m pituista kaapelia. (Dogan 2011, 47.)

Kaapelien enimmäispituutta ei voi kasvattaa muutoin kuin pienentämällä tiedonsiirtonopeutta, koska maksimipituus on kiinni sähkömagneettisen aallon kulkunopeudesta. Solmujen on otettava näyte yksittäisestä bitistä samalla ajan hetkellä, minkä takia siirtotien viive ei saa olla liian suuri. Tätä tarvitaan, että CAN-protokollan kilpavarausperiaate toimisi. Kilpavarauksesta lisää luvussa 2.4. (Alanen 2000, 5.)

CAN-H- ja CAN-L-kaapelien välillä oleva jännite-ero määrää väylän loogisen tilan. Loogisia tiloja on kaksi: dominantti, joka vastaa loogista tilaa 0, ja resessiivinen, joka vastaa loogista tilaa 1. Dominantissa tilassa CAN-H-kaapelin jännite on 3,5 V ja CAN-L-kaapelin jännite on 1,5 V. Dominantissa tilassa kaapelin jännite-ero on 2 V. Resessiivisessä tilassa molempien johtimien jännite on 2,5 V, jännite-ero on tällöin 0 V. Väylän dominantti tila ja resessiivisen tila on esitettynä kuviossa 2. (Dogan 2011, 49.)



Kuvio 2: CAN-väylän tilat (Dogan 2011, 49).

2.2 Viestin kehysrakenne

CAN-protokolla määrittelee neljä eri viestikehystä:

- sanomakehys
- kyselykehys
- virhekehys
- viivekehys (Alanen 2000, 6).

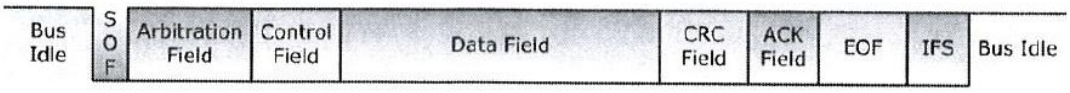
2.2.1 Sanomakehys

Sanomakehys sisältää varsinaisen tiedon, siinä voidaan välittää esimerkiksi anturin arvo tai vastata kyselykehyksen viestiin (Wilfried 2005, 37).

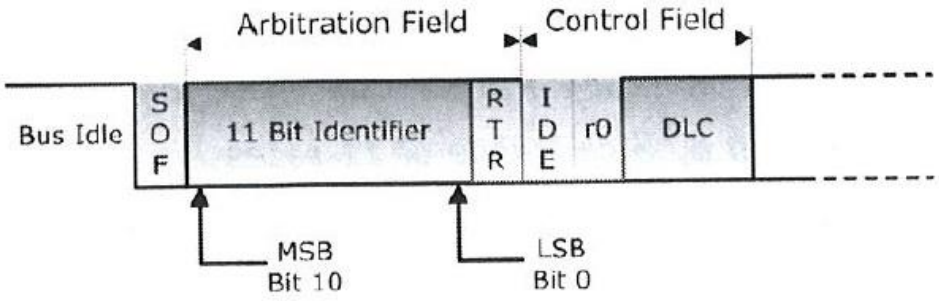
Sanomakehys (kuvio 3.) sisältää seuraavat osat:

- SOF (Start of Frame): Yhden dominantin bitin mittainen ja aloittaa viestikehyksen. Ennen aloitusbittiä väylän täytyy olla vapaana.
- Sovittelukenttä (Arbitration field): Koostuu kahdesta osasta, tunnistekentästä (Identifier (ID)) ja RTR (Remote Transmission Request) -bitistä. RTR-bitti on sanomakehyksessä aina 0. Tunnistekenttä yksilöi jokaisen viestin ja se lähetetään eniten merkitsevä bitti ensin. Tunnistekenttä on 11 bittiä pitkä standard CAN -formaatissa (kuvio 4) ja 29 bittiä pitkä laajennetussa CAN-formaatissa (kuvio 5).
- Ohjauskenttä (Control field): Kuusi bittiä pitkä. Sisältää IDE (Identifier Extension)-bitin, joka määrittää tunnistekentän pituuden. IDE-bitin ollessa 0, tunnistekenttä on 11 bittä pitkä, ja jos IDE-bitti on 1, tunnistekenttä on 29 bittiä pitkä. IDE-bitin jälkeen tulee yksi varalla oleva bitti, joka on varattuna tulevaisuuden käyttöön ja sen tulee olla 0. Viimeiset neljä bittiä ovat DLC (Data Length Code) -kenttä, joka kertoo tietokentän pituuden (kuvio 6).

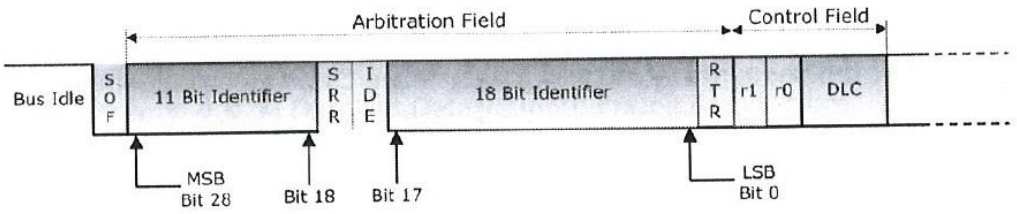
- Tietokenttä (Data field): Enintään kahdeksan tavun mittainen, 0-64 bittiä. Lähetetään eniten merkitsevä bitti ensin.
- CRC (Cyclical Recovery Checking): 16 bittiä pitkä kenttä, joka sisältää 15 bittiä pitkän tarkistussumman, ja kenttä päätetään yhdellä resessiivisellä bitillä.
- ACK (Acknowledgemet): Kahden bitin mittainen kuittauskenttä. Solmut, jotka ovat ottaneet viestin vastaan, kirjoittavat tähän dominantin bitin. Jälkimmäinen bitti on aina resessiivinen.
- EOF (End Of Frame): Lopetuskenttä on seitsemän bitin mittainen ja kaikki seitsemän bittiä ovat resessiivisiä.
- IFS (Interframe Space): Kolmen bitin mittainen, kaikki ovat resessiivisiä. Nämä bitit erottavat viestikehykset toisistaan. (Wilfried 2005, 43-52.)



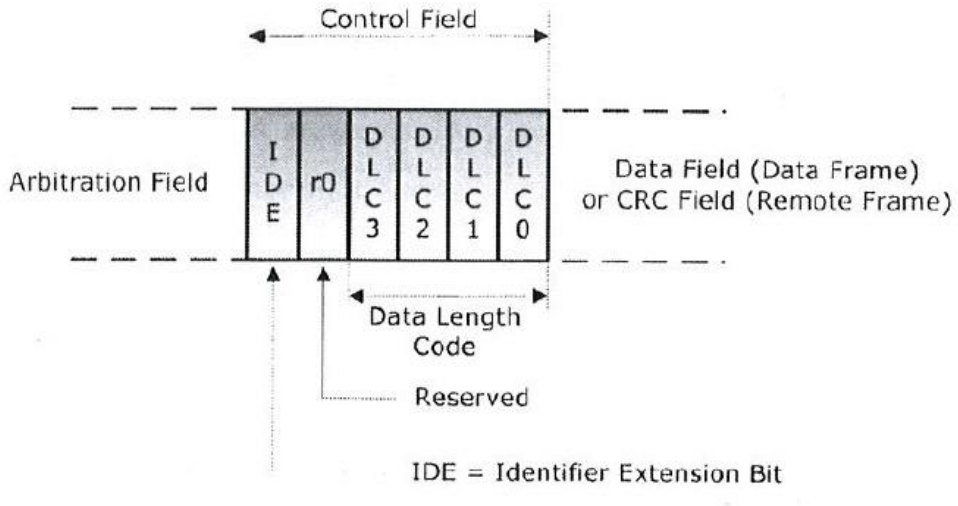
Kuvio 3: Sanomakehyksen rakenne (Wilfried 2005, 43).



Kuvio 4: Sovittelukentän rakenne (Wilfried 2005, 46).



Kuvio 5: Sovittelukentän rakenne laajennetussa CAN-formaatissa (Wilfried 2005, 46).



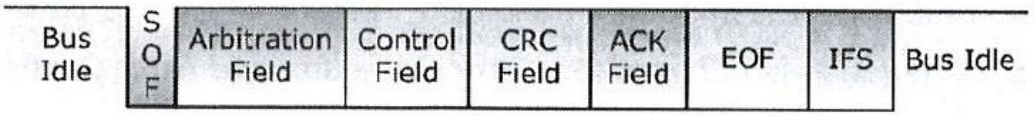
Kuvio 6: Ohajuskentän rakenne (Wilfried 2005, 47).

2.2.2 Kyselykehys

Kyselykehys lähetetään silloin kun jokin solmu haluaa saada tietyn viestin. Kyselykehyksellä pyydetään toista solmua lähettämään viestikehys. (Dogan 2011, 62.)

Kyselykehys on rakenteeltaan viestikehyksen kaltainen. Kyselykehyksestä puuttuu tietokenttä ja RTR-bitti on 1. (Wilfried 2005, 39.)

Kyselykehyksen ja tähän kyselyyn vastaavan viestikehyksen tietokentän pituuden kertovien DLC-kentien täytyy olla molemmissa kehyksissä samat. (Dogan 2011, 62.)



Kuvio 7: Kyselykehyksen rakenne (Wilfried 2005, 39).

2.2.3 Virhekehys

Virhekehyksen ensimmäinen osa koostuu kuuden bitin mittaisesta virhelippukentästä ja jälkimmäinen osa koostuu kahdeksan bitin mittaisesta erotuskentästä. Erotuskentän kahdeksan bittiä ovat resessiivisiä. (Dogan 2011, 62-63.)

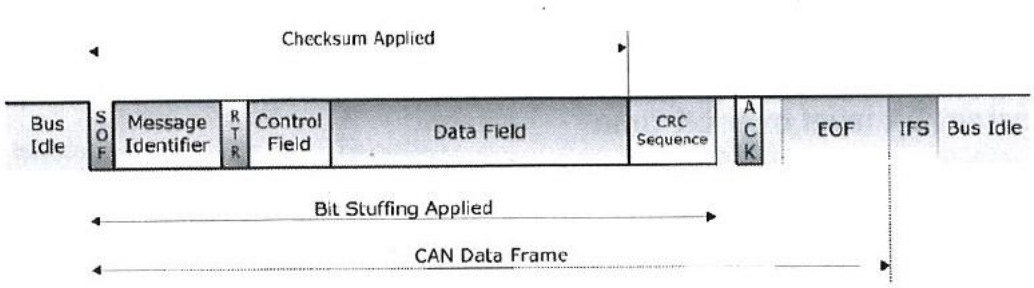
Virhekehys lähetetään välittömästi kun virhe on huomattu. Aktiivinen virhekehys koostuu kuudesta peräkkäisestä dominantista bitistä, tämä rikkoo bit-stuff-säännön. Viimeistään tässä vaiheessa kaikki solmut huomaavat virheen ja hylkäävät viestin. On mahdollista, että osa solmuista ei huomaa virhettä kuin vasta kuudennen peräkkäisen samassa tilassa olevan bitin jälkeen, jolloin ne alkavat lähettää omaa virhekehystä. Tällöin virhekehys voi kasvaa 12 peräkkäisen dominantin bitin mittaiseksi, minkä jälkeen lähetetään vielä kahdeksan resessiivistä bittiä. (Dogan 2011, 63.)

2.2.4 Viivekehys

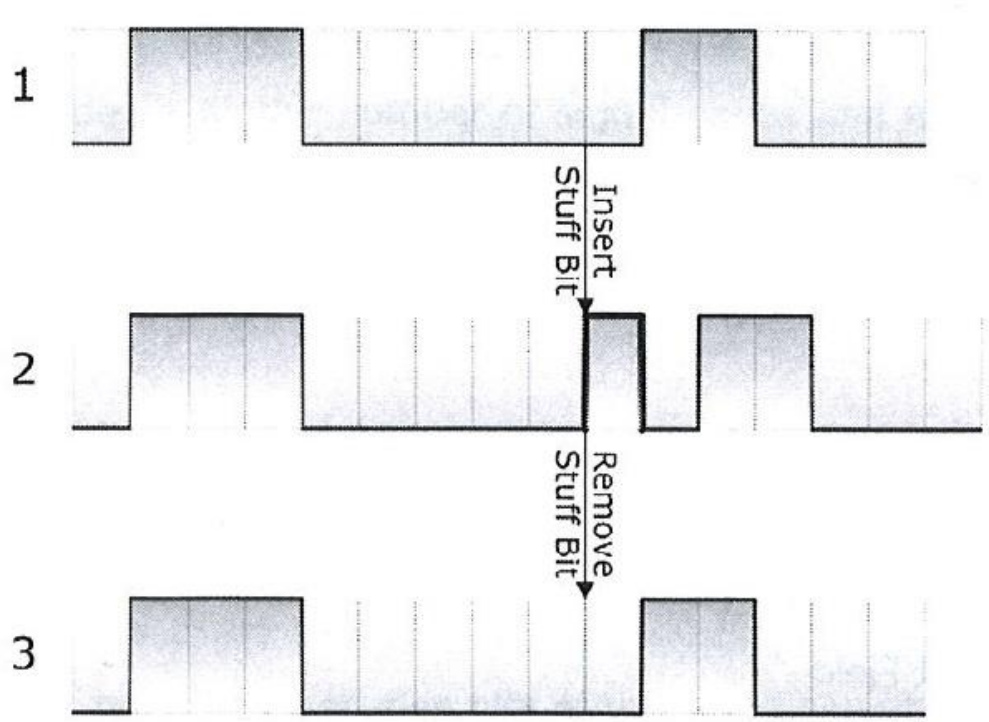
Viivekehys on samantyylinen kuin virhekehys, viivekehys ei aiheuta edellisen kehyksen uudelleen lähetystä, koska viivekehys lähetetään vain sanoma- tai kyselykehyksien välissä. Viivekehyksellä solmu aiheuttaa viivettä sanoma- tai kyselykehyksien välissä, joten sille jää enemmän aikaa käsitellä edellinen vastaanotettu sanoma. (Wilfried 2005, 66-67.)

2.3 Bit-stuff-sääntö

CAN-protokolla sallii vain viisi peräkkäistä samaa bittiä SOF-bitistä alkaen ja jatkuen CRC-kentän loppuun (kuvio 8). Jos lähetettävä tieto sisältää enemmän kuin viisi peräkkäistä bittiä, lähettävä solmu lisää viidennen bitin jälkeen päinvastaisen bitin. Vastaanottava solmu taas poistaa tämän lisätyn bitin (kuvio 9). Tämä ominaisuus auttaa solmuja pysymään oikeassa tahdissa lähetyksen aikana, jos lähetetään monta samaa bittiä peräkkäin. (Wilfried 2005, 96-98.)



Kuvio 8: Sallittu bit-stuff-alue (Wilfried 2005, 96).

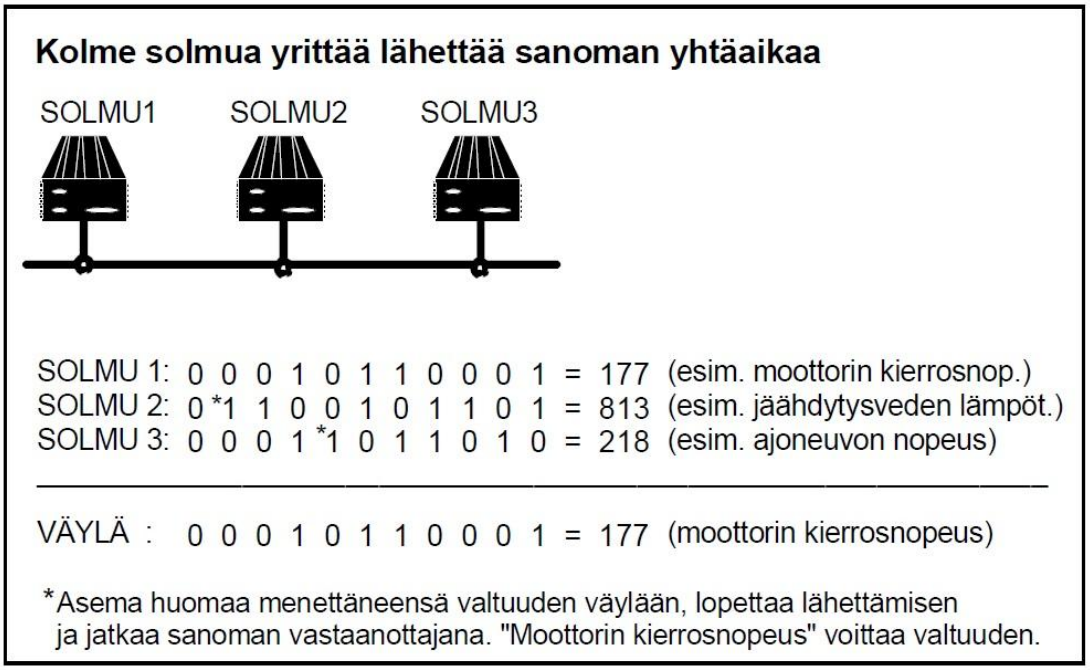


Kuvio 9: Esimerkki stuff-bitin lisäämisestä (Wilfried 2005, 98).

Kuviossa 9 on esitetty stuff-bitin lisääminen. Ensimmäisessä kohdassa on lähetettävä viesti, joka sisältää kuusi peräkkäistä dominanttia bittiä. Toisessa kohdassa lähettävä solmu lisää viiden dominantin bitin jälkeen yhden resessiivisen bitin. Viimeisessä kohdassa vastaanottava solmu poistaa resessiivisen bitin ja näin lähetetty ja vastaanotettu viesti on sama sisältäen kuusi peräkkäistä dominanttia bittiä. (Wilfried 2005, 98.)

2.4 Kilpavaraus

CAN-väylään liitetyt solmut voivat lähettää sanomia väylälle aina kun se on vapaa. Useamman solmun yrittäessä lähettää samaan aikaan, lähetysvuoro ratkaistaan kilpavarausmenetelmällä sanoman tunnistekentän avulla. Tunnistekentältään pienin on prioriteetiltään suurin ja se pääsee väylälle ensimmäisenä (kuvio 10). (Alanen 2000, 7.)



Kuvio 10: Esimerkki kilpavarauksesta (Alanen 2000, 7).

3 SAE J1939

3.1 J1939

SAE (Society Of Automotive Engineers) on autoalan standardisointijärjestö, joka on kehittynyt raskaan kaluston standardin J1939. J1939 on sovelluskerroksen protokolla, joka käyttää CAN-standardin fyysistä kerrosta. J1939 käyttää laajennettua CAN-standardia, jossa on 29 bittinen viestin tunnistekenttä. (Wilfried 2008, 5.)

J1939-standardi määrittelee väylän maksimipituudeksi 40 metriä, suurimmaksi nopeudeksi 250 kbit/s ja väylällä olevien solmujen suurin sallittu määrä on 30. CAN-standardin mukaan 40 metriä pitkällä väylällä voidaan käyttää 1 Mbit/s nopeutta, mutta SAE haluaa omilla maksimimäärityksillään varmistaa väylän toimivuuden sekä luotettavuuden. (Wilfried 2008, 17-18.)

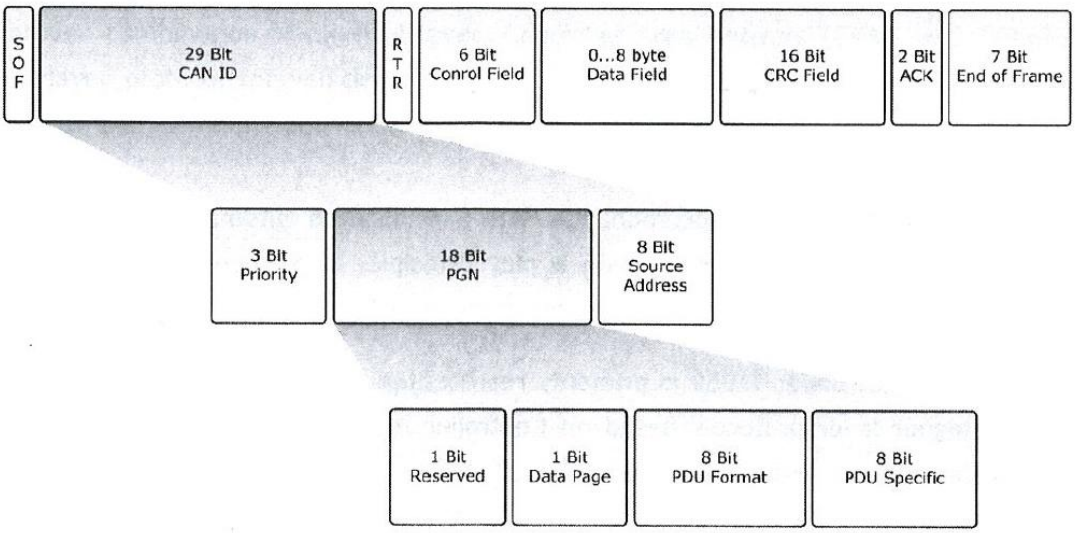
J1939 standardin pääominaisuudet ovat PGN (Parameter Group Number) ja SPN (Suspect Parameter Number). Näihin parametreihin on määriteltynä ajoneuvon tietoja ja valmiiksi määritellyt parametrit löytyvät SAE J1939 -standardista. (Wilfried 2008, 19.)

3.2 J1939-sanomakehys

J1939-standardin sanomakehys sisältää samat osat kuin luvussa 2.2.1 esitelty CAN-standardin sanomakehys. CAN-standardista poiketen J1939-standardin tunnistekenttä sisältää lähettävän solmun osoitteen ja joissakin tapauksissa myös vastaanottavan solmun osoitteen. (Wilfried 2008, 21).

J1939-standardin sanomakehyksen tunnistekenttä (kuvio 11) sisältää seuraavat osat:

- Priority: Kolmen bitin mittainen tunniste kertoo viestin tärkeyden, 0 on korkein prioriteetti ja 8 matalin prioriteetti.
- PGN (Parameter Group Number): 18 bittiä pitkä kenttä, joka koostuu R (Reserved) -bitistä, DP (Data Page) -bitistä, kahdeksasta PF (PDU Format) -bitistä ja kahdeksasta PS (PDU Specific) -bitistä.
- Reserved (R) -bitti on varattuna tulevaisuuden käyttöön, on aina 0.
- Data Page (DP) -bitti on nykyään aina 0 ja se osoittaa sivulle 0. Kun 0 sivu on täynnä, siirrytään määrittelemään parametrejä sivulle 1. Käytännössä tämän avulla pysytään tulevaisuudessa lisäämään parametriryhmien määrää.
- PDU Format (PF) -kenttä on yksi kentistä, jotka määrittelevät parametriryhmän numeron.
- PDU Specific (PS) -kentän arvo riippuu PF-kentän arvosta. Jos PF-kentän arvo on väliltä 0-239, PS-kentän arvo on vastaanottavan solmun osoite. Osoite 255 on yleinen osoite ja se on tarkoitettu kaikille solmuille. Jos PF-kentän arvo on välillä 240-255, PS-kentän arvon avulla määritellään lisää parametriryhmän numeroita.
- Source address: Kahdeksan bittiä pitkä lähettävän solmun osoite. Jokaisella väylällä olevalla solmulla täytyy olla oma osoite. (Wilfried 2008, 40-50.)



Kuvio 11: J1939-standardin sanomakehys (Wilfried 2008, 21).

PDU (Protocol Data Unit) koostuu 29-bittisestä tunnistekentästä ja itse datakentästä. PDU on nimitys tärkeille tietokentille ja sillä ei sinällään ole muuta merkitystä. PDU Format (PF) ja PDU Specific (PS) ovat määriteltäviä kenttiä, ne kuuluvat jokaiseen J1939-standardin mukaiseen sanomakehykseen. (Wilfried 2008, 40.)

3.3 SPN

SPN (Suspect Parameter Number) on parametrin tunnistenumero. Valmiiksi määritellyt parametrit ovat esiteltynä SAE J1939/71 -standardissa. Parametrien määrittelyt sisältävät seuraavat tiedot:

- Data Length: Viestin pituus, kertoo tavuina viestin pituuden.
- Resolution: Kertoo yhden bitin arvon, kuinka paljon yhden bitin muutos vaikuttaa arvoon.
- Offset: Mistä arvosta lähdetään liikkeelle.
- Data Range: Mikä on arvojen vaihteluväli, minimi- ja maksimiarvot.

- Type: Mitä tyyppiä lähetettävä arvo on. Tyyppi voi olla esimerkiksi mitattu tai tila. Mitattu arvo on jokin suure, esimerkiksi paine, nopeus, lämpötila. Tila on esimerkiksi päällä/pois tai auki/kiinni.
- Reference: Kertoo mihin parametriryhmään se kuuluu. (Wilfried 2008, 46-47.)

SPN 110	**Engine Coolant Temperature**
Temperature of liquid engine cooling system	
Data Length	1 Byte
Resolution	1 deg C / Bit
Offset	-40 deg C
Data Range	-40 to 210 deg C
Type	Measured
Reference	PGN 65262

Kuvio 12: Parametrin määrittely (Wilfried 2008, 47).

Esimerksi kuviossa 12 oleva moottorin jäähdytinnesteen lämpötilan määrittely. Parametrin tunnistenumero (SPN) on 110 ja se kuuluu parametriryhmään (PGN) 65262. Viestin pituus on yksi tavu ja se on tyypiltään mitattu. Yhden bitin arvo on yksi celsiusaste. Arvojen vaihteluväli on –40 – 210 celsiusasetetta. Jos viestissä lähetetyn tavun arvo on nolla, se tarkoittaa lämpötilaa –40 celsiusastetta. Jos tavun arvo on yksi, se vastaa lämpötilaa –39 celsiusastetta. Lämpötila lasketaan kaavan yksi mukaan.

$$t = data - 40 \qquad (1)$$

jossa t on lämpötila celsiusasteina
data on tietokentässä lähetetyn tavun arvo.

3.4 PGN

PGN (Parameter Group Numbers) on parametriryhmä, ryhmään kuuluu useita parametrejä. Kun sanomakehys lähetetään, kehykseen sisältyy kerrallaan yksi parametriryhmä, joka sisältää monen parametrin tiedot. Parametriryhmät ovat ryhmiteltynä siten, että samantyyppiset tiedot sisältyvät yhteen ryhmään. Esimerkiksi moottorin lämpötila ryhmään kuuluu parametrejä jotka sisältävät lämpötiloja (kuvio 13). (Wilfried 2008, 42-51.)

PGN 65262		**Engine Temperature**	
Transmission Rate		1 sec	
Data Length		8 bytes	
Data Page		0	
PDU Format (PF)		254	
PDU Specific (PS)		238	
Default Priority		6	
PG Number		65262 ($FEEE_{hex}$)	
Description of Data			**SPN**
Byte	1	Engine Coolant Temperature	110
	2	Fuel Temperature	174
	3, 4	Engine Oil Temperature	175
	5, 6	Turbocharger Oil Temperature	176
	7	Engine Intercooler Temperature	52
	8	Engine Intercooler Thermostat Opening	1134

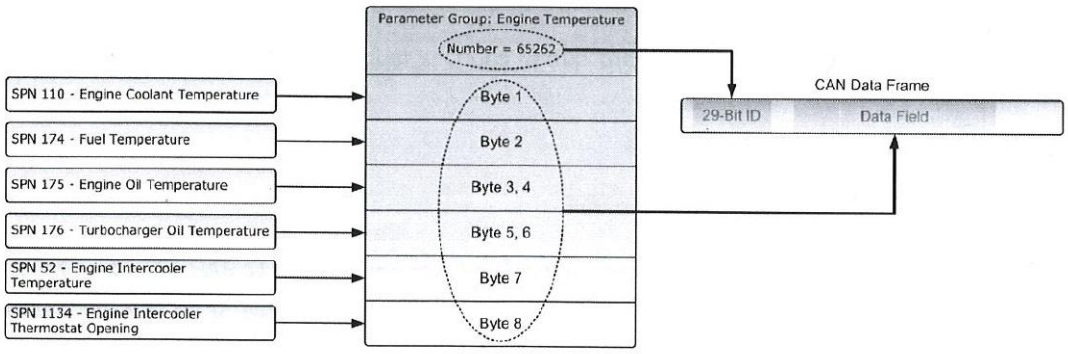
Kuvio 13: Parametriryhmän määrittely (Wilfried 2008, 46).

Esimerkki parametriryhmän määrittelystä (kuvio 13):

- Transmission Rate: Kertoo kuinka usein tieto lähetetään. PGN 65262 lähetetään sekunnin välein.
- Date Length: Kertoo kuinka monta tavua lähetettävä tietokenttä sisältää.

- Data Page: Nykyään 0, tulevaisuudessa tämän avulla voidaan määrittää lisää parametriryhmiä.
- PDU Format: Määrittää parametriryhmän numeron, tästä arvosta riippuu PDU Specific -kentän arvo.
- PDU Specific: PDU Format -kenttä on yli 240, joten PDU Specific -kenttä määrittää myös parametriryhmän numeroa.
- Default priority: Kertoo viestin prioriteetin. Pienin luku vastaa suurinta prioriteettia.
- PG Number: Parametriryhmän numero.

Kuviossa 14 on esitettynä kuinka parametrit liittyvät parametriryhmään ja CAN-sanomakehykseen.



Kuvio 14: Parametrien sijainti parametriryhmässä (Wilfried 2008, 47).

4 J1939-STANDARDIN SOVELTAMINEN KÄYTÄNNÖSSÄ

Varsinainen työ on teoriapohjainen suunnitelma siitä miten Deware Oy:n nykyistä seurantajärjestelmää tarvitsee muuttaa, jotta sillä voisi lukea työkoneen J1939-standardin mukaisia sanomia.

4.1 Mikroprosessorin liittäminen CAN-väylään

Osa mikropiireistä sisältää CAN-kontrollerin, jolloin se on helppo liittää CAN-väylään. Tässäkin tapauksessa tarvitaan CAN-lähetin-vastaanotin-piiri, mikroprosessorin ja CAN-väylän väliin. Yksi tällainen lähetin-vastaanotin-piiri on Microchip Technologyn valmistama MCP2551.

CAN-väylän kanssa voi käyttää mikroprosessoria, joka ei sisällä CAN-kontrolleria, mutta silloin tarvitaan CAN-lähetin-vastaanotin-piirin ja CAN-väylä johtimien väliin erillinen CAN-kontrolleri. CAN-kontrolleri kommunikoi mikroprosessorin kanssa SPI-väylää pitkin. Kuitenkin CAN-sovelluksissa kannattaa käyttää mikroprosessoria, jossa on sisäänrakennettu CAN-kontrolleri, koska tällöin muun muussa ohjelmointi helpottuu, eikä tarvitse sovittaa piirilevylle erillistä CAN-kontrolleria. (Dogan 2011, 119-127.)

4.2 Mikroprosessorin valinta

Deware Oy:n ensimmäinen seurantalaitteen prototyyppi toimii Atmelin valmistamalla AtMega128-piirillä, tämä mikroprosessori ei sisällä CAN-kontrolleria, joten tuotteen jatkokehitystä ajatellen on järkevää vaihtaa mikroprosessori CAN-kontrollerin sisältämään mikroprosessoriin. Uudesta mikroprosessorista täytyy löytyä samat ominaisuudet kuin AtMega128-mikroprosessorista ja lisäksi siinä täytyy olla CAN-väylän tuki. CAN-väylän myötä käsiteltävän datan määrä kasvaa melkoisesti, joten mikroprosessorin olisi hyvä olla myös tehokkaampi.

Atmelin piirien ohjelmoinnissa on käytössä MikroElektronikan ohjelmistoympäristö. Tähän ohjelmistoympäristöön ollaan oltu tyytyväisiä, koska se on helppo käyttää ja se sisältää valmiita kirjastoja. Esimerkiksi LCD-näytön käyttöön on valmis kirjasto, jonka avulla voidaan näyttö saadaan nopeasti käyttöön ja voidaan keskittyä itse ohjelman tekemiseen. ARM-arkkitehtuurin piireihin siirtyminen on päätetty jo aikaisemmin, ja MikroElektronikan ohjelmistoympäristö on hankittuna myös ARM-prosessoreille. Tämä ohjelmistoympäristö sisältää tällä hetkellä tue Cortex M3 –prosessoreille, joten prosessori valinta tulee olemaan ARM Cotex M3.

Texas Instumentin valmistamalla ARM Cortex M3 LM3S2918 -mikroprosessorilla on vaadittavat ominaisuudet. Prototyypin AtMega128 toimii 8 MHz:n kellotaajuudella. LM3S2918-piiri toimii taas 50 MHz:n kellotaajuudella. LM3S2918-mikroprosessori sisältää samat ominaisuudet kuin AtMega128 ja se tukee myös CAN-väylää. Digi-Keyn (Www.digikey.com) sivustolta katsottuna prosessoreiden hinnat ovat: Atmel AtMega128 17,21 €/kpl ja Texas Instrument LM3S2918 11,63 €/kpl. CAN-väylää varten tarvitaan lisäksi lähetin-vastaanotin-piiri, tähän sovelias voisi olla Microchip Tecnologyn MCP2551. MCP2551 on melko yleisesti käytetty piiri ja sen hankita hinta on edullinen 1,12 €/kpl (Digi-Key). Hintojen perusteella ARM-prosessoriin olisi kannattanut siirtyä jo aikaisemmin.

4.3 Ohjelmoinnin suunnittelu

J1939-standardin mukaiset sanomat ovat melko monimutkaisia ja tiedot pystytään muuttamaan järkevään muotoon vain tietämällä kuinka parametrit ovat määriteltynä. Ilman J1939-standardin osaa 71 on lähes mahdoton yrittää saada luettua haluttua tietoa väylältä. Yhdessä sanomakehyksen tietokentässä lähetetään peräkkäin monen eri parametrin tiedot, pitää tietää montako bittiä mikäkin parametri sisältää, ja on tiedettävä mikä on yhden bitin painoarvo. SAE (Society of Automotive Engineer) myy standardeja ja

J1939/71-standardin hankitahinta on 66 dollaria (Vehicle Application Layer, 2011). CAN-väylän luku on suunnitteluvaiheessa ja koko J1939-standardin hinta on noin 800 dollaria, joten standardi on vielä jätetty ostamatta. Tämän työn aikana kuitenkin selvisi, että parametrien määrittelyt ovat standardin osassa 71, joten sen hankintahinta ei ole suuri.

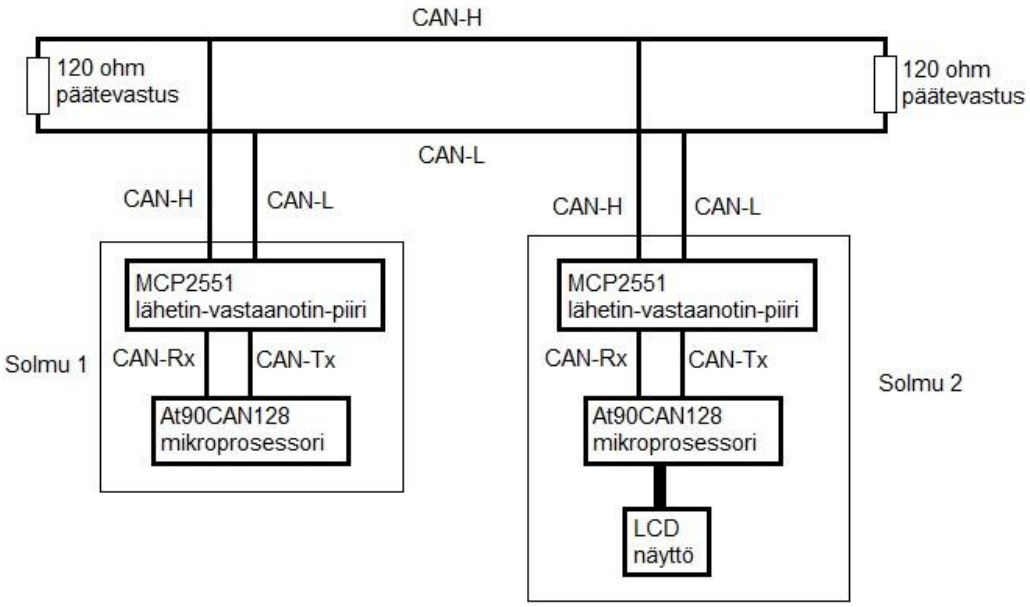
CANCapture on CAN-väylän simulointiohjelma. Ohjelmasta on saatavilla demoversio (CANCapture 2009). Ohjelmassa on määriteltynä J1939-standardin mukaiset sanomat, jotka ovat myös näkyvillä. CANCapture-ohjelmasta katseltuna saa hyvän käsityksen mitä kaikkea J1939-standardin mukaisessa CAN-väylässä liikkuu. Tämä auttaa suunnittelemaan laitetta kun tiedetään mitä tietoa väylältä löytyy. Ohjelmalla pystyy myös rakentamaan oman CAN-väylän simulointia varten.

4.4 Käytännön testaus

ARM-prosessoriin siirtyminen on päätetty, mutta niitä ei ole vielä hankittuna. CAN-väylän testaus kokeillaan aluksi Atmelin At90CAN32-mikroprosessorilla.

CAN-väylän testauksessa käytetään kahta mikroprosessoria, jotka ovat kytkettynä kuvion 15 esittämällä tavalla. Solmu 1 koostuu mikropiiristä ja CAN-lähetin-vastaanotin-piiristä. Solmu 1 lähettää aluksi viestin, jossa tietokenttä on nolla. Joka lähetyksen jälkeen tietokentän arvoa kasvatetaan yhdellä. Kun tietokentän arvo on kasvanut lukemaan 100, sitä aletaan pienentämään joka lähetyksen jälkeen yhdellä, kunnes se on saavuttanut arvon 0.

Solmu 2 koostuu mikropiiristä, CAN-lähetin-vastaanotin-piiristä ja LCD-näytöstä. Solmu 2 vastaanottaa solmun 1 lähettämän tiedon ja näyttää sen LCD-näytöllä. Solmun 2 lähdekoodi on esitettynä liitteessä 1.



Kuvio 15: CAN-väylän testauksen rakennekuva.

MikroElektronikan ohjelmistoympäristössä on CAN-väylään käyttöön valmis kirjasto. CAN-väylän testauksessa valmiit kirjastot osoittautuivat vaikeakäyttöiseksi. Valmiiden funktioiden esittelyt olivat vaikeaselkoisia ja osan toimintaa piti kokeilla ennen kuin sen sai toimimaan. Tämän testin perusteella tultiin siihen päätökseen, että CAN-väylän käyttöön tarvittavat funktiot pitää tehdä itse. Vaikuttaa siltä, että valmis kirjasto on tarkoitettu CAN-väylän yksinkertaisempaan käyttöön, esimerkiksi välittämään napin tila mikroprosessorilta toiselle. Tämän testauksen jälkeen J1939-standardin mukaisten viestien lähettämiseen ja vastaanottamiseen valmis kirjasto ei tunnu sopivalta.

5 YHTEENVETO JA TULOKSET

CAN-väylä on kehitetty jo 1980-luvulla mutta vasta 2000-luvulla se on alkanut yleistyä. 2008 se tuli henkilöautoihin pakolliseksi diagnostiikka väyläksi. Alun perin CAN-väylä on kehitetty ajoneuvoväyläksi, mutta nykyään CAN-väylä on levinnyt sen alkuperäisestä käyttötarkoituksestaan muun muassa teollisuus automaatioon. Elektroniikan määrä ajoneuvoissa lisääntyy edelleen, joten CAN-väylä tulee olemaan tärkeä osa ajoneuvoteollisuutta vielä pitkään.

Tavoitteena oli tutustua ja selvittää CAN-väylän ja raskaankaluston standardin J1939 toimintaa. Tähän tavoitteeseen päästiin hyvin, selvitettiin CAN-väylän teoriaa ja J1939-standardin mukaiset sanomakehykset tulivat tutuiksi. CANCapture-ohjelman avulla pystyttiin tutustumaan mitä kaikkea väylällä liikkuu.

Toissijaisena tavoitteena oli rakentaa yksinkertainen mikroprosessorilla toimiva laite, jolla olisi voinut lukea CAN-väylästä tietoa. Tähän tavoitteeseen päästiin osittain. Selvitettiin miten mikroprosessori pitää liittää CAN-väylään, sekä tehtiin kahden mikroprosessorin testiympäristö. Testissä lähetettiin CAN-väylään viesti ja vastaanotettiin se toisella mikroprosessorilla. CAN-väylän toimintaa testattiin itse keksityllä sanomalla. Ohjelmaa, joka lukisi ajoneuvon CAN-väylästä tietoa, ei ehditty tekemään. Tähän vaikutti osaltaan myös se, että pitäisi ostaa J1939-standardi tai ainakin osa numero 71, mistä näkisi parametriryhmien numerot sekä parametrien määrittelyt. Standardia ei lähdetty suin päin ostamaan aluksi sen suuren hankintahinnan vuoksi. Työn aikana kuitenkin selvisi, että parametrien määrittely ovat standardin osassa 71, standardin yhden osan hinta ei ole kohtuuton.

LÄHTEET

Wilfried, V. 2005. A Comprehensible Guide to Controller Area Network. 2. p. Greenfield: Copperhill Media Corporation.

Wilfried, V. 2008. A Comprehensible Guide to J1939. Greenfield: Copperhill Media Corporation.

Alanen, J. 2000. CAN-ajoneuvojen ja koneiden sisäinen paikallisväylä. [pdf-tiedosto]. Tampere: VTT Automaatio. [Viitattu 26.4.2012]. Saatavana: http://www.oamk.fi/~eeroko/Opetus/Ohjausjarjestelmat/CAN/CAN-perusteet_AlasenMateriaalia.pdf

Dogan, I. 2011. Controller Area Network Projects. United Kingdom: Elektor International Media BV.

CANCapture. 2009. [Pc-ohjelma, Demo]. EControls Inc. [Viitattu 26.4.2012]. Saatavana: http://www.cancapture.com/downloads/doc_download/45-cancapture-demo-v243.html

Heikkilä, J. 2007. EOBD, OBD2 ja valmistajakohtaiset protokollat. [Verkkosivu]. Kangasala: OBD Automotive Oy. [Viitattu 26.4.2012]. Saatavana: http://www.obd.fi/index.php?option=com_content&task=view&id=32&Itemid=2

Vehicle Application Layer. 15.3.2011. J1939/71-standard [Verkkosivu]. SAE International. [Viitattu 17.5.2012]. Saatavana: http://standards.sae.org/j1939/71_201103

LIITTEET

Liite 1. Testauksessa käytetyn Solmun 2 lähdekoodi.

```
sbit LCD_RS at PORTC2_bit;      //LCD-näytön alustukset
sbit LCD_EN at PORTC3_bit;
sbit LCD_D4 at PORTC4_bit;
sbit LCD_D5 at PORTC5_bit;
sbit LCD_D6 at PORTC6_bit;
sbit LCD_D7 at PORTC7_bit;

sbit LCD_RS_Direction at DDC2_bit;
sbit LCD_EN_Direction at DDC3_bit;
sbit LCD_D4_Direction at DDC4_bit;
sbit LCD_D5_Direction at DDC5_bit;
sbit LCD_D6_Direction at DDC6_bit;
sbit LCD_D7_Direction at DDC7_bit;

unsigned char init_liput,lah_liput,vast_liput;
unsigned char Data_pituus;
char viesti_saapunut;
char Rx_Tx_Data[8];
char viesti_naytolle[20];
long ID;
long Node1 = 0x123;
long Node2 = 0x321;
long viesti;

void main() {
   Lcd_Init();         //näytön alustus

   init_liput = 0;     //lippujen nollaus
   lah_liput = 0;
   vast_liput = 0;

   lah_liput = _CAN_IDE_XTD_FRAME & //lähetyksen liput
               _CAN_NO_RTR_FRAME;   //ei RTR-kehystä, extented viestikehys

   init_liput = _CAN_CONFIG_SAMPLE_THRICE &  //alustuksen liput
                _CAN_CONFIG_XTD_MSG //extented viestikehys, kolme bitti näytettä

   CANInitialize(1, 1, 3, 3, 1, init_liput);        //CAN-väylän alustus

   CANSetOperationMode(_CAN_MODE_STANDBY,0xFF);

   /***Filtereiden alustus, asetetaan filteriin Solmun1 ID, -1 = 0xFFFF ***/
   CANSetFilter(CAN_RX_FILTER_3,  Node1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_4,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_5,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_6,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_7,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_8,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_9,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_10, -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_11, -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_12, -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_13, -1, _CAN_CONFIG_XTD_MSG);
   CANSetFilter(CAN_RX_FILTER_14, -1, _CAN_CONFIG_XTD_MSG);

   CANSetMask(CAN_RX_MASK_3,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetMask(CAN_RX_MASK_4,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetMask(CAN_RX_MASK_5,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetMask(CAN_RX_MASK_6,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetMask(CAN_RX_MASK_7,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetMask(CAN_RX_MASK_8,  -1, _CAN_CONFIG_XTD_MSG);
   CANSetMask(CAN_RX_MASK_9,  -1, _CAN_CONFIG_XTD_MSG);
```

```
    CANSetMask(CAN_RX_MASK_10, -1, _CAN_CONFIG_XTD_MSG);
    CANSetMask(CAN_RX_MASK_11, -1, _CAN_CONFIG_XTD_MSG);
    CANSetMask(CAN_RX_MASK_12, -1, _CAN_CONFIG_XTD_MSG);
    CANSetMask(CAN_RX_MASK_13, -1, _CAN_CONFIG_XTD_MSG);
    CANSetMask(CAN_RX_MASK_14, -1, _CAN_CONFIG_XTD_MSG);

    CANSetOperationMode(_CAN_MODE_ENABLE,0xFF);

    while(1) {
       /***Viestin luku***/
       viesti_saapunut = CANRead(&ID, Rx_Tx_Data, &Data_pituus, &vast_liput);
       /***tähän mennään jos viesti tullut, ja tulee Solmulta 1  ***/
       if ((ID == Node1) && viesti_saapunut) {
          Lcd_Cmd(_LCD_CLEAR);
          viesti_saapunut = 0;
          viesti = Rx_Tx_Data[0];
          sprintf(viesti_naytolle,"%u",viesti);
          Lcd_Out(1,1,viesti_naytolle);   //tulostetaan saapunut viesti näytölle
          Delay_ms(10);
          CANWrite(Node2, Rx_Tx_Data, 8, lah_liput);
          Delay_ms(1000);
       }
    }
}
```