Jesse Väinämö

Korielektroniikan ohjainlaitteen suunnittelu ja valmistus

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Ajoneuvotekniikka

Insinöörityö

29.11.2021

Tiivistelmä

Tekijä:	Jesse Väinämö
Otsikko:	Korielektroniikan ohjainlaitteen suunnittelu ja valmistus
Sivumäärä:	41 sivua + 1 liite
Aika:	29.11.2021
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Ajoneuvotekniikka
Ammatillinen pääaine:	Ajoneuvosuunnittelu
Ohjaajat:	Lehtori Sanna Heikkinen

Insinöörityön aiheena oli suunnitella ja valmistaa korielektroniikan ohjainlaitteen prototyyppi Tritium-sähköurheiluautoon. Valmistuneen ohjainlaitteen käyttötarkoitus ajoneuvossa on ohjata ovi- ja ikkunamoottoreita sekä CAN-väylän välityksellä sumuvaloja. Insinöörityö kuvaa yleisesti ohjainlaitteen suunnitteluprosessia vaatimustenmäärittelystä ohjainlaitteen ohjelmointiin.

Suunnittelun alussa tutkittiin ajoneuvoympäristön asettamia haasteita elektroniikalle ja niiden hallitsemisen avuksi luotuja AEC-Q100- ja Q200-standardeja, jotka määrittelevät elektroniikan komponenttien soveltuvuuden ajoneuvokäyttöön. Ohjainlaitteen ominaisuudet ja vaatimukset määritettiin näiden haasteiden perusteella, minkä jälkeen siirryttiin komponenttivalintoihin. Ohjainlaitteeseen valittiin AEC-Q100- ja Q200-hyväksyttyjä aktiivi ja passiivikomponentteja.

Piirilevysuunnittelussa otettiin huomioon ajoneuvoympäristön vaatimukset elektroniikalle. Ohjainlaite suojattiin ylijännitteeltä ja vastanapaiselta kytkennältä PTC-sulakkeesta, TVS-diodista ja Schottky-diodista muodostuvalla kytkennällä. Mikrokontrollerin lukemien painikkeiden signaalit suodatettiin myös kestävällä Schmitt-kytkimiä hyödyntävällä suodatuselektroniikalla. Ajoneuvoelektroniikan vaatimukset näkyivät myös ikkunamoottoreiden ohjausjärjestelmässä, joiden anti-pinch-turvamekanismi toteutettiin moottorin virranmittaukseen perustuvalla ratkaisulla.

Ohjainlaitteen ohjelmiston suunnittelu aloitettiin sille asetettujen ominaisuuksien pohjalta. Ohjainlaitteen viiveetön ja robusti toiminnallisuus saavutettiin laitteistokeskeytyksiä hyödyntävällä ohjelmalla. Lopuksi työssä kuvataan mikrokontrollerin asetusten ohjelman lähdekoodin pohjan luonti STM32CubeMX-ohjelmistossa ja ohjelman kääntäminen Makefile-tiedostolla.

Insinöörityön lopputuloksena oli toimiva ja vaatimustenmukainen ohjainlaitteen prototyyppi, jolla voidaan ohjata ajoneuvon ovien ja ikkunoiden nostimia sekä valaistusta CAN-väylän välityksellä.

Avainsanat: ajoneuvotekniikka, ohjainlaite, sulautetut järjestelmät

Abstract

Author:	Jesse Väinämö
Title:	Designing and Building a Prototype of an Automotive Body Control Module
Number of Pages:	43 pages + 1 appendix
Date:	29 November 2021
Degree:	Bachelor of Engineering
Degree Programme:	Automotive Engineering
Professional Major:	Automotive Design Engineering
Supervisors:	Sanna Heikkinen, Lecturer

The objective of this Bachelor's thesis was to design and build a prototype of a body control module for the electric supercar developed by Tritium Automotive. The function of the body control module in the vehicle is to control various actuators and lights through button inputs and via the CAN-bus of the vehicle. The overall design and building process of an automotive control module is examined starting from the specification of the requirements to the programming of the control module.

At the beginning of the design process the challenges posed by the automotive environment regarding vehicle electronics were studied. A look was then taken at the AEC-Q100- and Q200-standards which are used to define and test the eligibility of a component for automotive use. The requirements and specifications for the control module were then defined and the design process followed with the selection of components. Circuit design was then started, where the control module's over voltage and reverse voltage protection circuit was implemented. The filtering and protection electronics for the microcontroller inputs were also designed. The anti-pinch safety mechanism required by the window function was then solved with a current measuring system.

Next the firmware of the control module was designed, where robust and timely operation demanded by the automotive environment was achieved by using external interrupts. Programming of the STM32-microcontroller in a Linux-environment was then demonstrated by using the STM32CubeMX-software to create the source code and a Makefile which is used to compile the source code.

The result was a control module built for automotive use which can be used to drive four electric DC motors and four 12 V outputs via the CAN-bus of the vehicle.

Keywords: automotive, control module, embedded system

Sisällys

Lyhenteet

AEC: Automotive Electronics Council, ajoneuvokomponenttien standardisointijärjestö.

ADAS: Advanced driver-assistance system, kuljettajaa avustavat järjestelmät.

CAN: Controller Area Network, ajoneuvoelektroniikan kommunikaatioväylä.

DMA: Direct Memory Access eli oikosiirto, joka tarkoittaa tiedon kopioimista tietokoneen sisällä käyttämättä suoritinta.

EMC: Electromagnetic Compatibility, sähkömagneettinen yhteensopivuus.

GPIO: General Purpose Input Output, mikrokontrollerin yleiskäyttöön konfiguroitava pinni.

HSE: High Speed External, mikrokontrollerin kellolähdetyyppi.

IC: Integrated Circuit, mikropiiri.

JEDEC: Solid State Technology Association, puolijohdevalmistajien standardointiorganisaatio.

LSE: Low Speed External, mikrokontrollerin kellolähdetyyppi.

TVS: Transient Voltage Suppression, transienttijännitesuoja.

PWM: Pulse Width Modulation, pulssinleveysmodulaatio.

PTC: Polymeric Positive Temperature Coefficient device, uudelleenkytkeytyvä vastus.

PTC: Power and Temperature Cycling, komponenttien testausprosessi.

1 Johdanto

Insinöörityön aiheena oli suunnitella ja valmistaa korielektroniikan ohjainlaitteen prototyyppi sähköajoneuvoon. Idea sai alkunsa tarpeesta saada tiettyä tarkoitusta varten rakennettu ohjainlaite. Työn tilaajana on Tritium Automotive, jonka valmistamaan sähköurheiluautoon ohjainlaite asennettiin. Ohjainlaitteen suunnittelun lisäksi työ vaati ajoneuvon väyläarkkitehtuurin suunnittelun. Työ sisälsi myös ohjainlaitteen kokoamisen, ohjelmoinnin, testaamisen ja käyttöönoton.

Korielektroniikka on käsitteenä laaja, ja sillä voidaan tarkoittaa yleisesti ajoneuvon sähköisiä toimilaitteita, kuten valoja (ajovalot, vilkut, jarruvalot, sumuvalot), moottoreita (ikkunanostin, ovien hydrauliikka) ja solenoideja (ovien lukot) mutta myös näitä toimilaitteita ohjaavia ohjainlaitteita.

Tritium Automotive on suomalaisen komposiittialan yrityksen Finnluxury Oy:n luoma projekti, joka keskittyy kehittämään piensarjana valmistettavaa pitkälle kustomoitavaa sähköurheiluautoa. Projekti alkoi vuonna 2014, ja polttomoottorikäyttöinen prototyyppi valmistui vuonna 2018. Tämän jälkeen aloitettiin ajoneuvon muuttaminen sähkökäyttöiseksi.

Opinnäytetyössä hyödynnettiin laajasti projektisuunnittelussa -ja toteutuksessa käytettävää V-prosessimallia, joka on esitetty kuvassa 1. V-malli on muokattavissa projektin tarpeiden mukaan, mutta yleisesti se määrittelee projektin vaiheet ja niistä vastuussa olevat henkilöt. V-mallin tavoitteena on minimoida projektin riskit, varmistaa laadukas lopputulos, pienentää kustannuksia ja parantaa kommunikointia työympäristössä [1, s. 30.]

Kuva 1. V-prosessimallin vaiheet.

2 Ohjainlaitteen ominaisuudet ja vaatimukset

2.1 Ajoneuvo ympäristönä

Ajoneuvo on vaativa ympäristö elektroniikalle, ja siksi se olikin yksi merkittävimmistä seikoista ohjainlaitteen ominaisuuksien määrittelyssä. Ajoneuvoelektroniikka pitää suunnitella monella eri tavalla vikasietokykyiseksi ajoneuvon käyttöturvallisuuden takaamiseksi. Ohjainlaitteiden on siedettävä suuria lämpötilanvaihteluita, tärinää ja toimia laajalla jännitealueella kestäen jännitepiikit ja vastanapaisuuden kytkennän. Lisäksi ohjainlaitteen pitää läpäistä sähkömagneettisen säteilyn emissio- ja immuniteettitestit. [2, s. 18.]

Esimerkki lämpötilaltaan vaativasta ympäristöstä ajoneuvossa olisi polttomoottorin sylinterinkannessa sijaitseva sytytyspuola. Moottorin käynnistystilanteessa puolestaan akun jännite voi laskea hetkellisesti tai apuvirralla käynnistäminen voi aiheuttaa jännitepiikin. Elektromagneettisen säteilyn lähteitä polttomoottorikäyttöisessä ajoneuvossa on esimerkiksi sytytysjärjestelmään kuuluvat virranjakaja ja puola. [3, s. 16.]

Sähköajoneuvossa merkittävimmät elektromagneettisen säteilyn lähteet ovat voimansiirron komponentit kuten akusto, invertteri ja sähkömoottori [4, s. 2].

Sähkömagneettinen yhteensopivuus *(Electromagnetic Compatibility)* on olennainen osa ajoneuvon sähköjärjestelmän suunnittelua, ja sillä on merkittävä vaikutus ajoneuvon toimintaan. Etenkin nykyajoneuvon turvallisuuskriittisten järjestelmien (moottorin- ja vaihteistonohjaus, ADAS) toimintavarmuus on taattava kaikissa tilanteissa.

Seuraavilla tavoilla voidaan parantaa ohjainlaitteen häiriösuojausta suunnitteluvaiheessa [3, s. 28]:

- I/O-pinnien suojaus ulkoisella suodatuskytkennällä
- korkean tehon johtimien erotus optoerottimella
- ohjainlaitteen jännitelähteen kanssa sarjaan kytketyllä ferriittihelmellä
- ohjainlaitteen kotelointi sähköä johtavalla materiaalilla.

Seuraavat suunnitteluvaiheet vaikuttavat merkittävästi ajoneuvokokonaisuuden sähkömagneettiseen yhteensopivuuteen [3, s. 36]:

- ohjainlaitteiden sijoittaminen ajoneuvoon häiriölähteet huomioiden
- johtosarjan suunnittelu ja reititys huomioiden merkittävät häiriölähteet
- häiriösuojattujen johtimien käyttö johtosarjassa
- ajoneuvon korin rakenteen hyödyntäminen EMC-suunnittelussa.

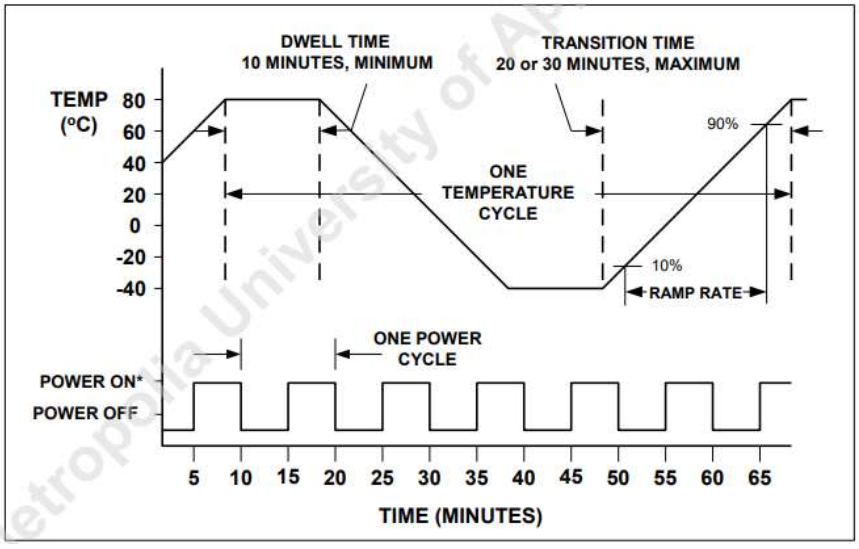
2.2 AEC-standardit

Vaativa ympäristö asettaa niin piirilevysuunnittelulle sekä ohjainlaitteen komponenttivalinnoille korkeat vaatimukset. Tästä tarpeesta ovat syntyneet AEC-Q100–104- ja AEC-Q200-standardit, jotka määrittelevät komponenttien sopivuuden ajoneuvokäyttöön. Näistä AEC-Q100–104 käsittelevät aktiivikom-

ponentteja ja AEC-Q200 passiivikomponentteja. AEC-hyväksynnät jaetaan sovellutustensa mukaan seuraaviin luokkiin: *AEC-Q100 Integrated Circuits*, mikropiirit, *AEC-Q101 Discrete Semiconductors*, erilliset puolijohdekomponentit, *AEC-Q102 Discrete Optoelectronic Semiconductors*, erilliset optoelektroniset puolijohteet, *AEC-Q103 Sensors*, anturit, *AEC-Q104 Multichip Modules*, mikrokontrollerit ja *AEC-Q200 Passive Components*, passiivikomponentit. [5.]

Liitteessä 1 on havainnollistettu AEC-hyväksynnän laajaa testausprosessia, joka jakautuu olosuhde-, mekaanisiin ja sähköisiin testeihin. Lisäksi komponentin sähköstaattisen purkauksen kesto ja EMC-ominaisuudet mitataan. Itse testimenetelmät ovat useimmille testeille JEDEC-standardointisorganisaation määrittelemiä. [6, s. 13.]

Olosuhdetesteihin kuuluu esimerkiksi kuvassa 2 esitetty JEDEC 22A105D -standardissa määritelty *power and temperature cycling* eli PTC-testi. PTC-testissä mitattavan komponentin ympäristön lämpötilaa vaihdellaan sen maksimi- ja minimikäyttölämpötilan välillä ja samanaikaisesti sen virta kytketään päälle ja pois tietyin väliajoin [7, s. 6].

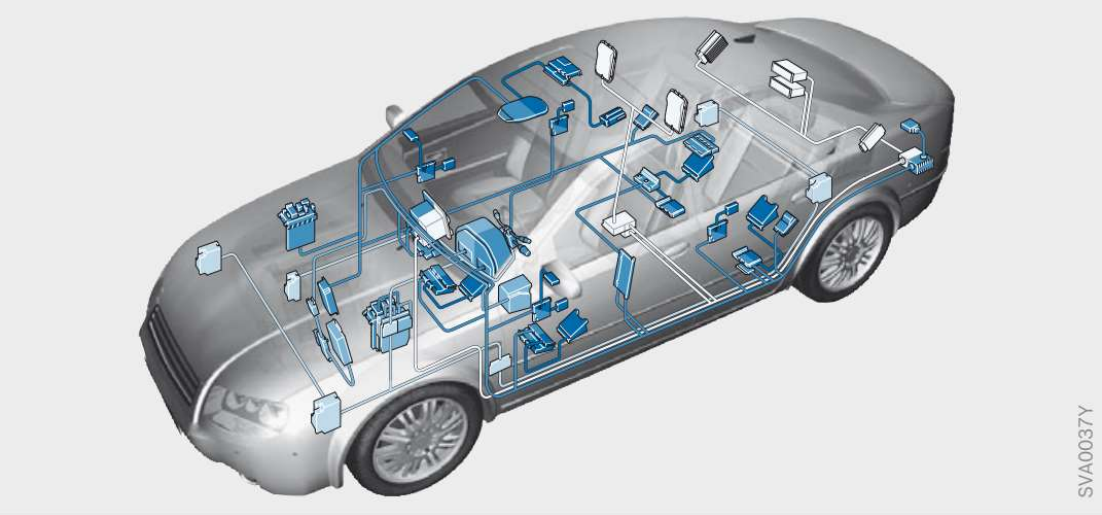


Kuva 2. PTC-testausprosessi. Testauslämpötilat ja kytkentäsyklin pituus ovat komponenttikohtaisia [7, s. 8].

PTC-testi testaa 30 kappaleen erän komponentteja, joista yksikään ei saa kytkeytyä pois päältä lämpötilan takia (*thermal shutdown*) [6, s.14].

2.3 CAN-väylä

Ajoneuvon elektronisten ohjainlaitteiden määrän moninkertaistuessa on tullut esille tarve ajoneuvokäyttöön sopivalle ohjainlaitteiden väliselle tietoverkolle, josta on syntynyt *Controller Area Network,* CAN-väylä. CAN-väylän fyysisen tason määrittelevät standardit ISO 11898-2 (high-speed CAN, 125 kbit/s – 1 Mbit/s) [8] ja ISO 11898-3 (low-speed CAN, 40 kbit/s – 125 kbit/s) [9]. Korkeamman nopeuden CAN-väylää käytetään perinteisesti turvallisuuskriittisemmissä ohjainlaitteissa, kuten moottorin ja vaihteiston ohjainlaitteistoissa. Hitaampi CAN-väylä on käytössä taas mukavuuselektroniikan ohjauksessa. CAN-väylä mahdollistaa tehokkaan ohjainlaitteiden välisen kommunikoinnin, joka on edellytys nykyaikaisen ajoneuvon sähköjärjestelmälle. Kuvassa 3 havainnollistetaan ohjainlaitteiden määrää ajoneuvossa ja niiden välistä tietoväylää. CAN-väylän fyysisestä tasosta kerrotaan enemmän luvussa 3.2.6.



Kuva 3. Ajoneuvon ohjainlaitteet ja niiden välinen tietoväylä [10, s. 82].

3 Ohjainlaitteen suunnittelu

Ohjainlaitteen suunnittelu aloitettiin kartoittamalla ohjainlaitteen ominaisuudet ja vaatimukset. Ominaisuuksien määrittelyn jälkeen voitiin aloittaa komponenttien etsintä ja osittain piirisuunnittelu. Piirisuunnittelun edetessä komponenttivalinnat tarkentuivat ja loppuvaiheessa ohjainlaitteen liittimet valittiin.

3.1 Ominaisuudet

Ohjainlaitteen käyttötarkoitus ajoneuvossa on ohjata ikkunoiden ja ovien moottoreita sisätilaan ja oviin sijoitetuilla painikkeilla sekä sumuvaloja CAN-väylän välityksellä. Käyttötarkoituksesta on johdettu seuraavat ohjainlaitteen ominaisuudet:

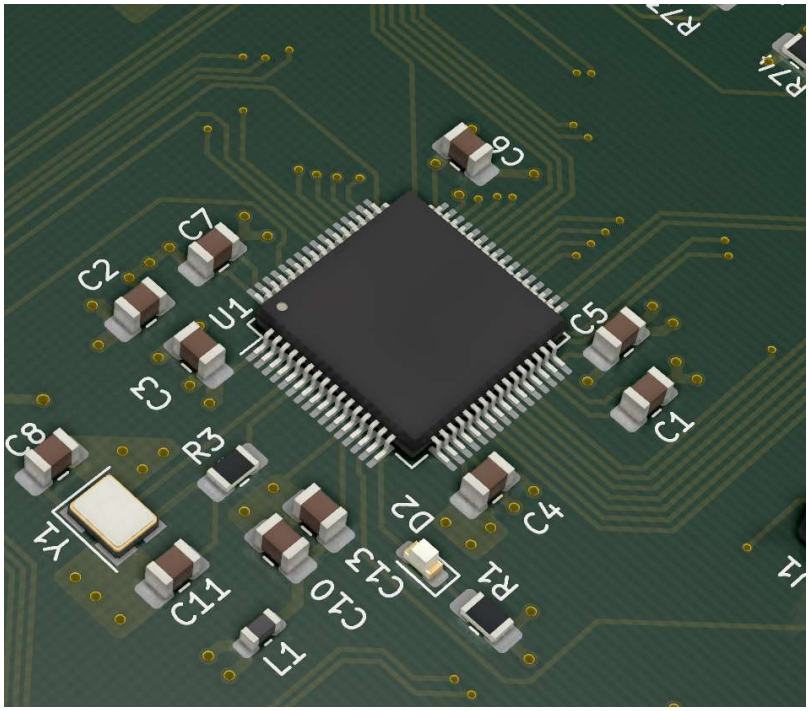
- ohjaus neljälle DC-moottorille, virrankesto 10 A per moottori
- moottorien käyttämän virran mittaus
- ohjaus neljälle 12 V:n lähdölle, virrankesto yhteensä 5 A
- CAN-väyläyhteys
- lisämuokattavuus, ylimääräisiä sisään- ja ulostuloja
- kokonaisvirrankesto, 20 A:n pääsulake.

3.2 Komponentit

Ohjainlaitteelta vaadituista ominaisuuksista johtuen se tarvitsee useita aktiivikomponentteja sekä näitä tukevia passiivikomponentteja. Ohjainlaitteeseen valittiin pääasiassa AEC-luokitettuja, ajoneuvokäyttöön sopivia pintaliitoskomponentteja.

3.2.1 Mikrokontrolleri

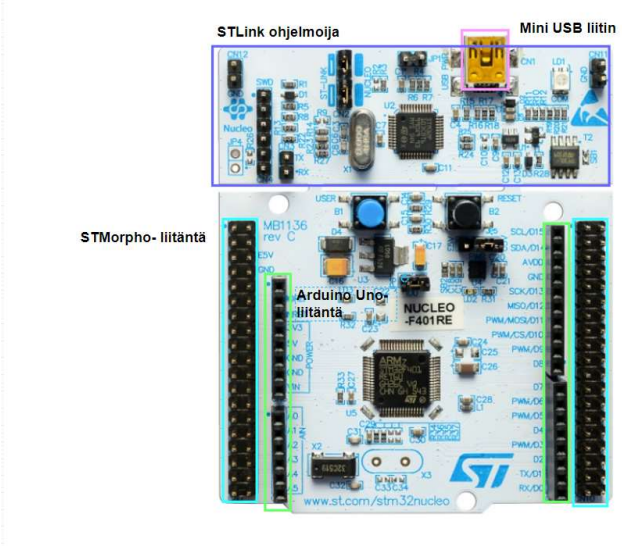
Ohjainlaitteen mikrokontrolleriksi valittiin STMicroelectronicsin valmistama, ARM Cortex-M4 -ytimeen perustuva, STM32F446RC-mikrokontrolleri sen hyvien ominaisuuksien takia: 256 kt Flash-muistia, kaksi CAN-väylälaitetta, useita GPIO-pinnejä, joissa laitteistokeskeytys ominaisuus, useita AD-muunninkanavia ja ajastimia, jossa PWM-lähtö (kuva 4).



Kuva 4. Kuvankaappaus STM32F446RE-mikrokontrollerista apukomponenttei-neen KiCAD-ohjelmiston 3D-näkymässä.

Mikrokontrolleri oli myös tuttu STMicroelectronics Nucleo F446RE -kehitysalus-tasta, joka helpotti valintaa. STM32 Nucleo -sarjan kehitysalustat ovat erinomai-nen työkalu prototyyppien rakentamiseen ja opiskeluun. Kehitysalustassa on

hyvät liitäntämahdollisuudet (Arduino Uno, STMorpho, Mini USB) sekä painikkeita ja LED-valoja, joita on havainnollistettu kuvassa 5.



Kuva 5. STM32 Nucleo -kehitysalusta [11].

Kehitysalusta sisältää myös irrotettavan STLink-ohjelmointipiirin, jota voidaan käyttää erillisenä ohjelmointilaitteena. STMicroelectronics tarjoaa useita esimerkkiohjelmia useille kehitysympäristöille mikrokontrollerin toimintojen testaukseen ja hyvän dokumentaation.

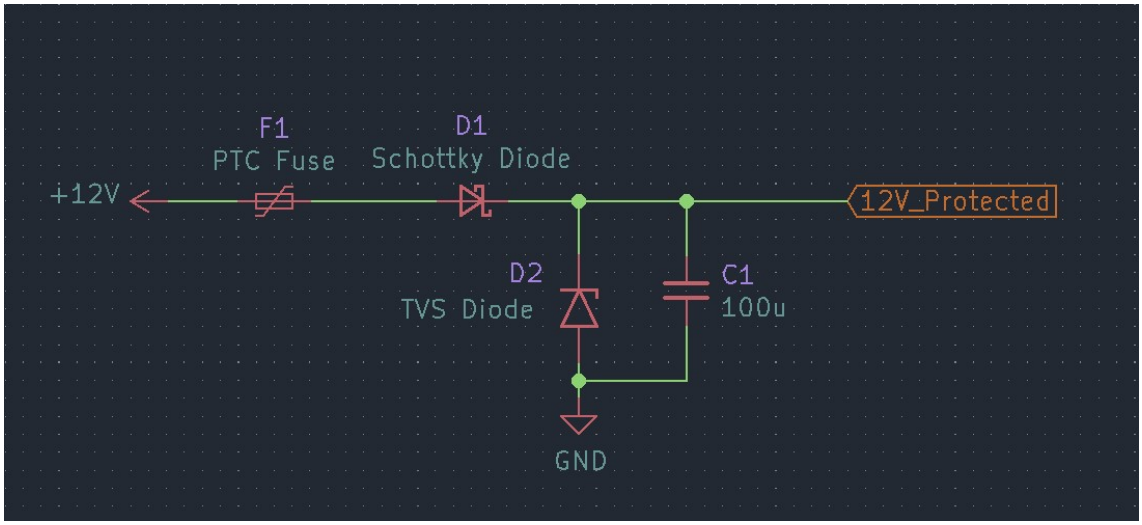
3.2.2 Kideoskillaattori

Ohjainlaitteen vaatimien kommunikaatioprotokollien (USB, CAN) takia mikrokontrollerin kellolähteeksi valittiin ulkoinen kideoskillaattori. Ulkoinen kideoskillaattori takaa näiden kommunikaatioprotokollien vaatiman korkeamman, tarkemman ja stabiilimman kellotaajuuden, jota sisäisellä kellolähteellä ei voida saavuttaa.

Komponentiksi valittiin valmistajan Kyocera, CX3225SA-sarjan 16 MHz:n AEC-Q200-hyväksytty kideoskillaattori. Komponentti tarjoaa hyvän kellotaajuuden tarkkuuden 15 PPM (*parts per million*) ja laajan käyttölämpötila-alueen -40–150 ºC, joita AEC-luokitetulta komponentilta voidaan odottaa [12, s. 1].

3.2.3 Jännitteensäätöelektroniikka

STMicroelectronics A5975D- ja LD39100-jännitteensäädinmikropiirit muuttavat akun 12 V:n jännitteen mikrokontrollerin ja CAN-vastaanottimen käyttämille 3,3 V:n ja 5 V:n jännitteille. Ohjainlaite on ylijännite- ja vastanapaiselta kytkennältä suojattu käyttäen palautuvaa sulaketta sekä Schottky- ja transienttisuojadiodeja, jotka on kytketty kuvan 6 kytkentäkaavion mukaisesti.



Kuva 6. Ylijännite- ja vastanapaisen kytkennän suojauspiiri.

Tulojännitteen noustessa transienttisuojadiodin läpilyöntijännitteen suuruiseksi se oikosulkee jännitteen ja laukaisee sulakkeen suojaten jännitteensäädintä ylijännitteeltä. Sisääntulojännitteeseen sarjaan kytketty Schottky-diodi estää jännitteen kulun väärään suuntaan vastanapaisen kytkennän tilanteessa.

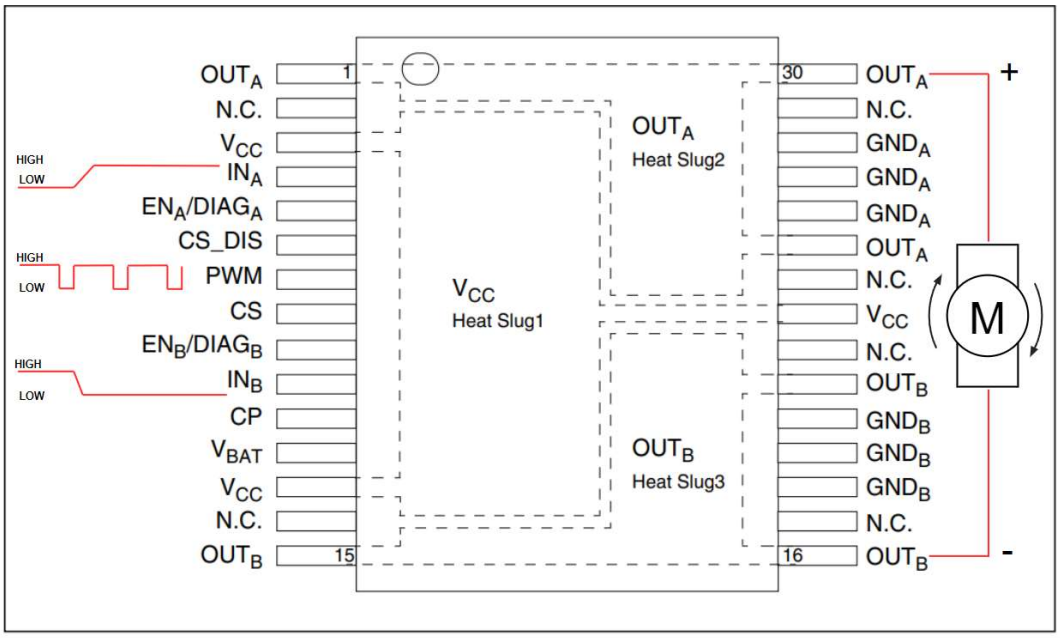
3.2.4 Moottorinohjainpiiri

STMicroelectronics VNH5019 -moottorinohjainpiirillä ohjataan ajoneuvon ovien ja ikkunoiden nostinmoottoreita. Tulojen IN_A ja IN_B tilat määrittävät logiikkataulukon 1 mukaisesti jännitteen suunnan lähtöjen OUT_A ja OUT_B välillä ja PWM-signaalin pulssisuhde jännitteen suuruuden.

Taulukko 1. Moottorinohjain mikropiirin logiikkataulu [13, s. 14].

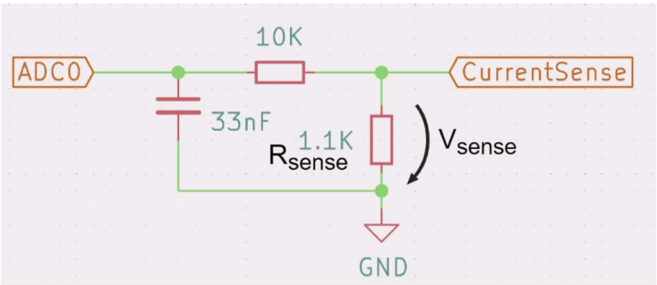
IN_A	IN_B	OUT_A	OUT_B	Operating Mode
1	1	H	H	Brake to VCC
1	0	H	L	Clockwise
0	1	L	H	Counterclockwise
0	0	L	L	Brake to GND

Kuva 7 havainnollistaa moottorinohjaimen toimintaperiaatetta. HIGH-signaali IN_A tulossa ja LOW-signaali IN_B tulossa ajaa moottoria myötäpäivään. Vaihtamalla signaalien järjestys moottori pyörii vastapäivään. HIGH-signaali tai LOW-signaali molemmissa tuloissa pysäyttää moottorin.



Kuva 7. Moottorin ohjaamisen toimintaperiaate [13, s. 6].

Moottorinohjainpiirissä on Current Sense -lähtö, joka mahdollistaa kuvan 8 mukaisen kytkennän avulla moottorin käyttämän virran mittaamisen mikrokontrollerin AD-muuntimella.



Kuva 8. Moottorin käyttämän virran mittaus Current Sense -lähdön avulla.

Moottorinohjaimen Current Sense -lähtö tuottaa virran, joka on suuruudeltaan verrannollinen moottorin käyttämään virtaan. Current Sense -lähdön tuottama virta luo vastuksen R_{sense} yli jännitehäviön V_{sense}, joka luetaan mikrokontrollerin AD-muuntimella. Moottorin käyttämä virta saadaan laskettua jännitteestä V_{sense}.

$$Iout = \frac{V_{sense} * K}{R_{sense}} \quad (1)$$

V_{sense} on jännitehäviö R_{sense} vastuksen yli
K on Current Sense -virran kerroinluku
R_{sense} on Current Sense -vastuksen arvo

Mikrokontrollerin lukema AD-muuntimen arvo saadaan muutettua ampeereiksi.

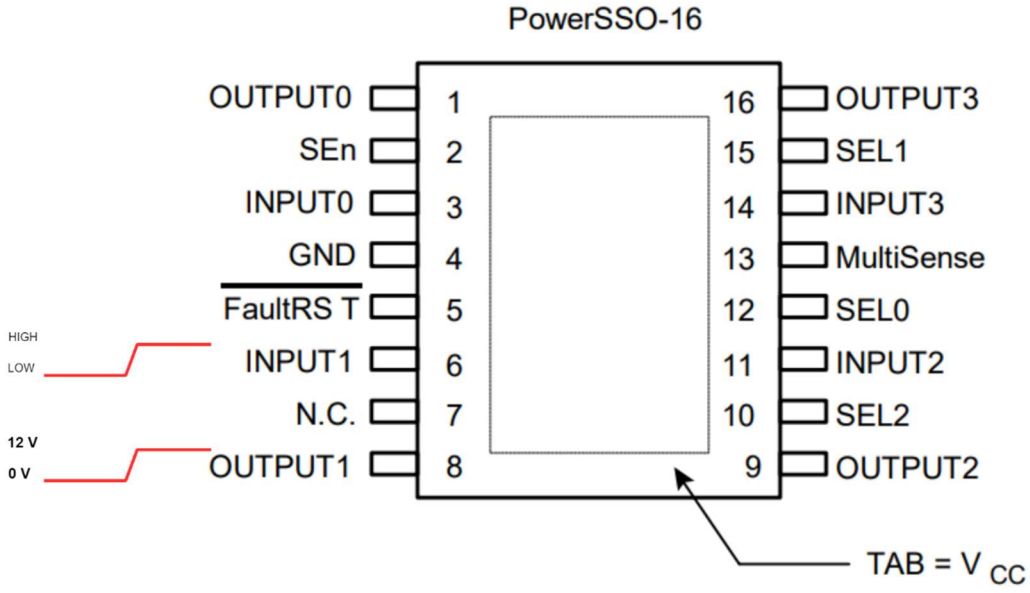
$$Iout = \frac{adc_{val} * V_{ref} * K}{4096 * R_{sense}} \quad (2)$$

$adcval$, mikrokontrollerin AD-muuntimen lukema arvo
V_{ref}, mikrokontrollerin AD-muuntimen referenssijännite
K, Current Sense -virran kerroinluku
R_{sense}, Current Sense -vastuksen arvo

Kaavaa hyödyntäen mikrokontrollerille voidaan esimerkiksi antaa moottorin virralle raja-arvo, jossa moottori kytketään pois päältä.

3.2.5 12 V:n lähtöjen ohjauspiiri

STMicroelectronics VNQ7140AJ on nelikanavainen kytkinpiiri, jolla voidaan ohjata 12 voltin jännite ajoneuvon toimilaitteille, esimerkiksi valaistukselle ja oven lukkojen solenoideille. Jokaisen kanavan jännitettä (OUT0-3) ohjataan kanavan tulopinnillä INPUT0-3. Kuva 9 havainnollistaa ohjauspiirin toimintaa, jossa INPUT1-tuloon johdettu HIGH-signaali kytkee lähdön OUTPUT1 päälle.

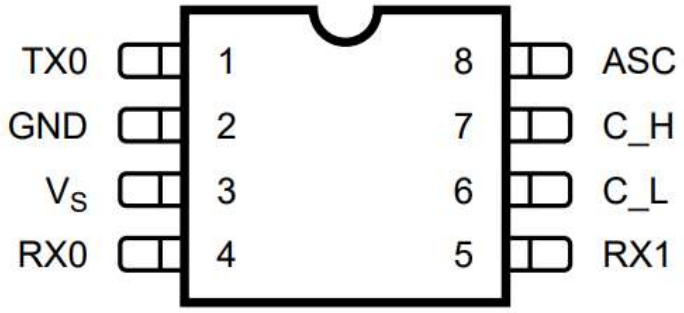


Kuva 9. VNQ7140-piirin konfiguraatiokaavio [14, s. 6].

VNQ7140AJ-piirissä on myös virranmittaukseen käytettävä MultiSense-lähtö sekä vikatilan ilmaisemiseen FaultRST-lähtö.

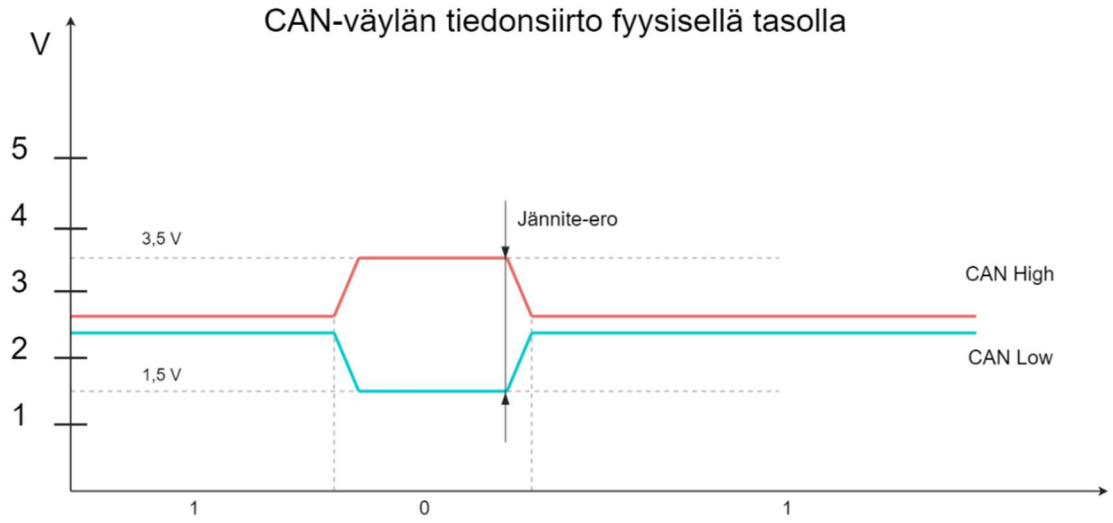
3.2.6 CAN-vastaanotin-lähetinpiiri

CAN-vastaanotin-lähetinpiiri L9615D toimii fyysisenä rajapintana ajoneuvon CAN-väylän ja mikrokontrollerin välillä. Kuvassa 10 on esitetty komponentin pinnikaavio.



Kuva 10. L9615 CAN-vastaanotin-lähettimen pinnikaavio [15, s. 2].

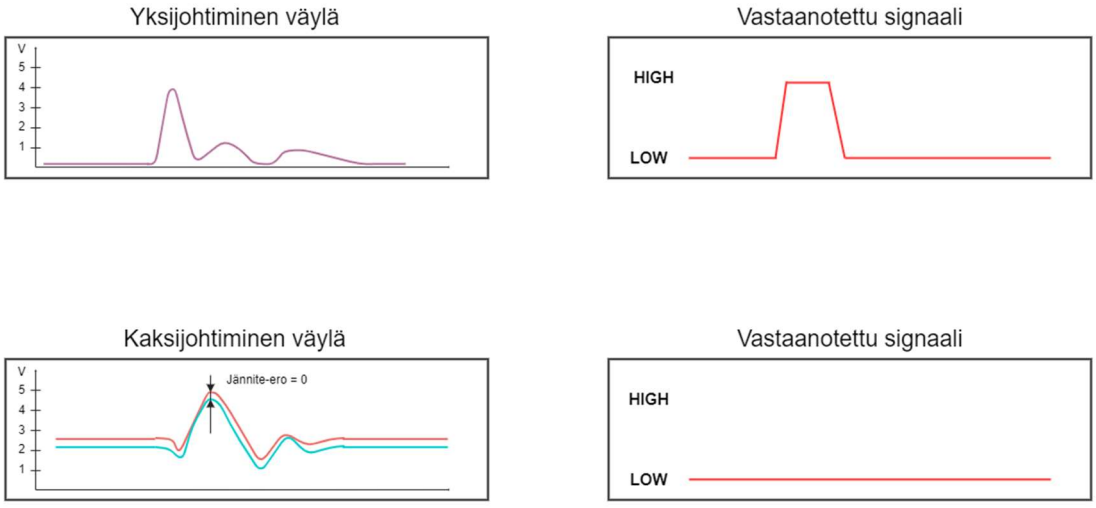
Lähettimen TX0- ja RX0-pinnit liitetään mikrokontrollerin CAN_Tx- ja CAN_Rx-pinneihin, jotka lähettävät ja vastaanottavat CAN-viestit CAN-väylään C_H - ja C_L-pinnien välityksellä. Tieto siirtyy CAN-väylässä differentiaalisignaalina kahdesta johtimesta (CAN High -ja CAN Low) kierretyn parikaapelin välityksellä, mitä havainnollistetaan kuvassa 11.



Kuva 11. Tiedon välittyminen CAN-väylässä kahden johtimen välisenä differentiaalisignaalina. Kuvassa CAN High-speed -väylän jännitetasot.

Väylän tila on näiden johtimien jännitteiden erotus. Väylän ollessa hallitsevassa *(dominant)* tilassa jännite-ero on 2 V, ja se vastaa binaareina arvoa 0. Väistyvässä *(recessive)* tilassa jännite ero on 0 V, joka vastaa binaareina arvoa 1. Hitaammassa CAN Low-speed -väylässä jännitetasot ovat CAN High -johtimen kohdalla 0–3,6 V ja CAN Low -johtimen 5–1,4 V. [10, s. 95.]

Kahdella johtimella lähetetyn signaalin ansiosta CAN-väylä on hyvin häiriösietoinen. Väylään syntyvä häiriö vaikuttaa molempiin johtimiin yhtä suuresti, jolloin johtimien välinen jännite-ero ei muutu ja häiriösignaali on käytännössä näkymätön vastaanottimelle [10, s. 94]. Kuvassa 12 vertaillaan häiriön vaikutusta yksi- ja kaksijohtimiseen väylään.



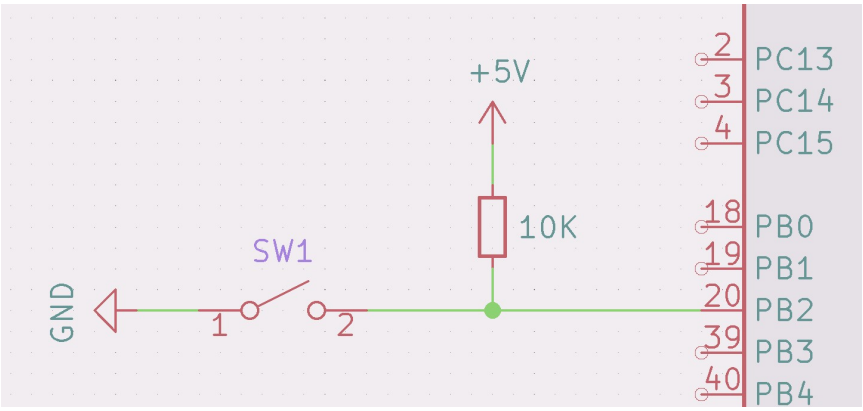
Kuva 12. Häiriön vaikutus yksi- ja kaksijohtimisen tiedonsiirtoväylän signaaliin.

CAN-väylän kaksijohtiminen rakenne mahdollistaa myös pitkän tiedonvälitysetäisyyden, joka riippuu väylälle asetetusta tiedonsiirtonopeudesta. Suositellut väylän maksimipituudet 250, 500 ja 1000 kbit/s CAN-väylille ovat vastaavasti 250, 100 ja 40 metriä [10, s. 95].

3.2.7 Painikkeiden signaalin suodatus

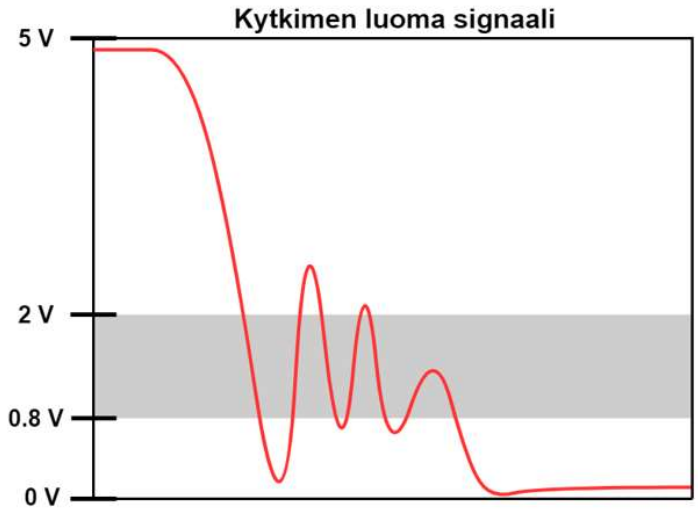
Mikrokontrollerin pinnit seuraavat TTL-logiikkaa, jossa yli kahden voltin tulojännite luetaan HIGH-signaalina ja alle 0,8 voltin jännite LOW-signaalina [16, s. 25]. Tuloihin liitetyt painikkeet voivat kuitenkin maadoittuessaan aiheuttaa vaihtelun jännitteeseen, joka usein aiheuttaa ongelmia. Tästä syystä painikkeiden signaali tulee suodattaa. Seuraavaksi tutkitaan suodattamattoman painikkeen signaalia ja verrataan sitä kahdella eri menetelmällä suodatetun painikkeen signaaliin mikrokontrollerin näkökulmasta. [17.]

Painike SW1 on kytketty mikrokontrollerin PB2-tuloon kuvan 13 kytkennän mukaisesti.



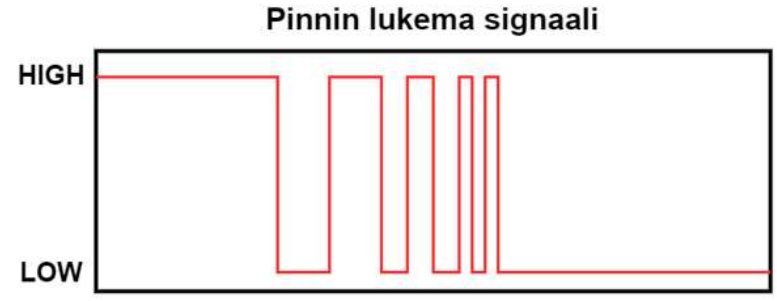
Kuva 13. Painikkeen SW1 kytkentä mikrokontrolleriin.

Sulkemalla painike SW1, tulon PB2 jännite maadoittuu, mutta jännite ei painikkeen mekaniikasta johtuen putoa siististi, vaan aiheuttaa kuvan 14 esittämän jännitteenvaihtelun signaalissa



Kuva 14. Painikkeen SW1 maadoituksen aiheuttama vaihtelu signaalin jännitteessä.

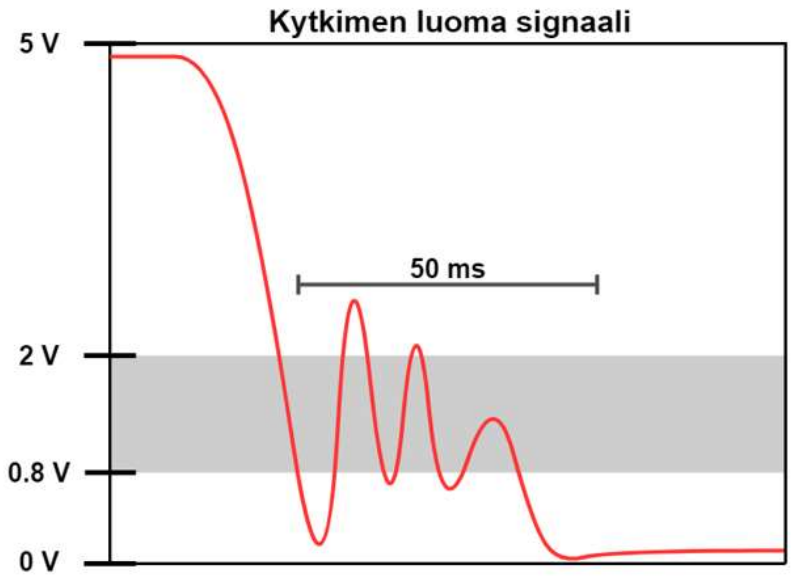
Signaalin jännitteenvaihtelusta johtuen mikrokontrollerin lukema arvo vaihtelee kuvassa 15 esitetyllä tavalla. TTL-logiikassa ei myöskään ole määritelty logiikan tilaa signaalin kelluessa harmaalla alueella 0,8 ja 2 voltin välillä, mikä tuo lisää arvaamattomuutta mikrokontrollerin lukemaan arvoon.



Kuva 15. Mikrokontrollerin tulon lukeman arvon vaihtelu.

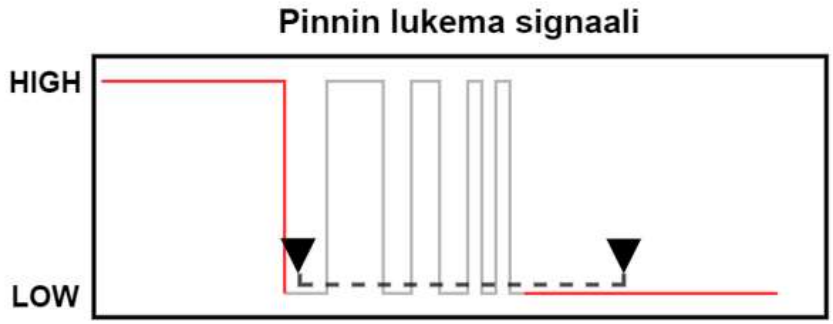
Painikkeen signaali voidaan suodattaa mikrokontrollerin ohjelmistossa. Tulon muuttaessa tilaansa, mikrokontrolleri lukee tulon tilan uudelleen pienen aikaviiveen jälkeen. Jos tila on sama kuin ensimmäisellä kerralla luettaessa, painik-

keen painallus hyväksytään. Kuvassa 16 on jälleen painikeen signaali, jossa aiheutuu sen maadoituksesta johtuen häiriö.



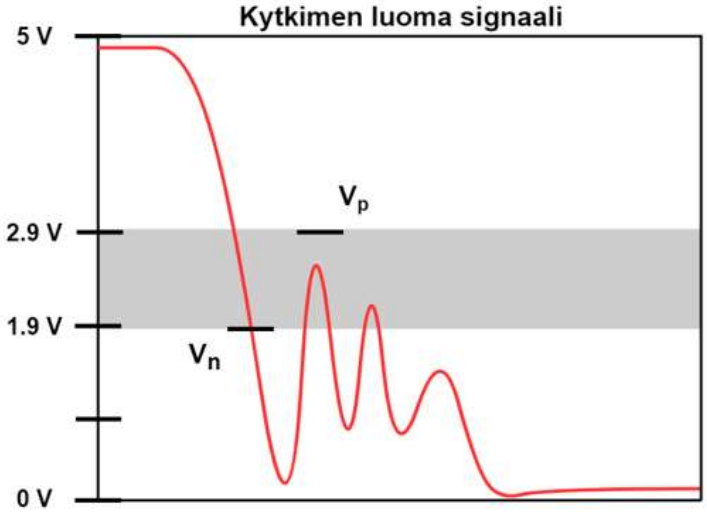
Kuva 16. Painikkeen signaalin häiriönsuodatus ohjelmistossa käyttäen 50 ms:n aikaviivettä.

Tulon tila vaihtuu kuvassa 17 ensimmäisen nuolen kohdalla korkeasta matalaksi ja mikrokontrolleri aloittaa viiveen laskemisen. Aikaviiveen jälkeen mikrokontrolleri lukee tilan uudelleen toisen nuolen kohdalla ja hyväksyy painikkeen painalluksen.



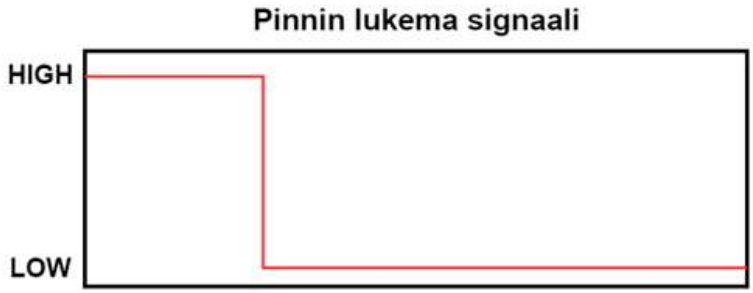
Kuva 17. Mikrokontrollerin lukema signaali.

Seuraavaksi havainnollistetaan signaalin suodatusta Schmitt-kytkimen avulla. Schmitt-kytkin tuo painikkeisiin hystereesin, joka suodattaa tehokkaasti signaalin. Hystereesillä tarkoitetaan käytännössä sitä, että kytkin pyrkii tilanvaihdon jälkeen säilyttämään uuden tilansa. Kuva 18 havainnollistaa suodatusta Schmitt-kytkintä käyttäen.



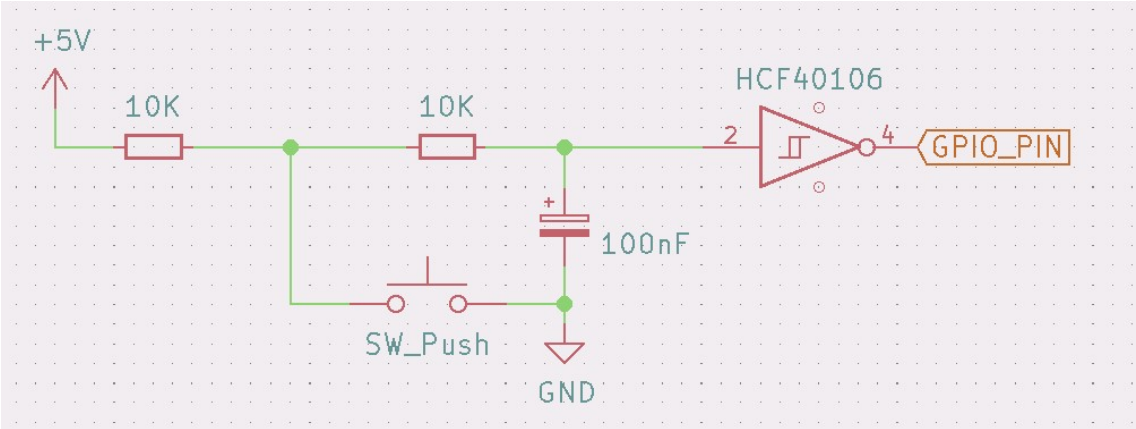
Kuva 18. Signaalin suodatus Schmitt-kytkimellä.

Schmitt-kytkimen tilan vaihtamiseksi korkeasta tilasta matalaan sen tulosignaalin on laskettava negatiivisen rajajännitteen V_n alle. Sen sijaan tilan vaihtamiseksi matalasta korkeaan signaalin on noustava positiivisen rajajännitteen V_p yli. Kuvassa 19 nähdään Schmitt-kytkimeltä lähtevä, mikrokontrollerin lukema signaali.



Kuva 19. Schmitt-kytkimeltä lähtevä mikrokontrollerin lukema signaali.

Ohjainlaitteen painikkeiden suodatuksessa päädyttiin kuusikanavaisesta HCF40106 Schmitt -kytkinmikropiiristä koostuvaan, kuvan 20 kaavion mukaiseen kytkentään.



Kuva 20. Painikkeiden suodatuselektroniikka.

3.3 Piirisuunnittelu

Piirisuunnittelu jakautui piirikaavio ja -levysuunnitteluun, jotka toteutettiin avoimen lähdekoodin KiCAD-ohjelmiston Eeschema- ja Pcbnew-moduuleilla.

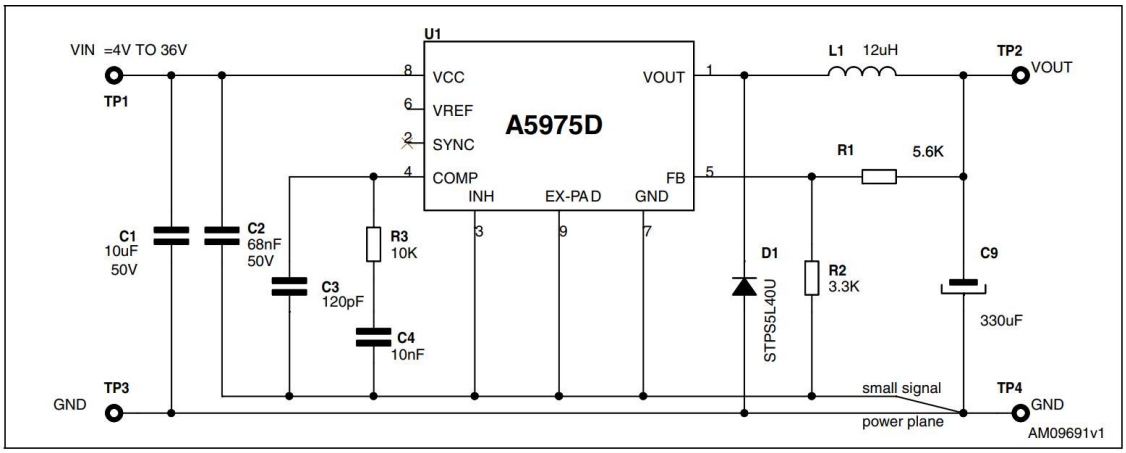
KiCAD on avoimen lähdekoodin piirisuunnitteluohjelmisto, joka on ollut kehitteillä vuodesta 1992 alkaen. KiCAD sisältää moduulit piirikaavio- ja piirilevysuunnitteluun sekä gerber-tiedostokatselimen. Ohjelman käytön oppii nopeasti, käyttöliittymä on yksinkertainen, sekä siinä on riittävät toiminnot (piirisimulointi, differentiaaliparien johdotus) ja intuitiivinen suunnitteluprosessi. Ohjelma on myös ilmainen, mikä tekee siitä erinomaisen vaihtoehdon piirikaavio- ja piirilevysuunnitteluun niin uusille kuin kehittyneemmille käyttäjille.

Useimmat ohjainlaitteeseen valitut komponentit eivät olleet ennestään tuttuja, joten suunnittelussa hyödynnettiin niiden referenssikytkentöjä ja piirilevyasetteluja. Useimpien komponenttien piirikaaviosymbolit ja piirilevykuviot eli footprintit löytyivät joko KiCADin omasta kirjastosta, tai ne olivat ladattavissa komponentin valmistajalta. Puuttuvat symbolit luotiin KiCADin Symbol- ja Footprint-editoreilla.

3.3.1 Piirikaaviosuunnittelu

Piirikaaviosuunnittelu aloitettiin jännitteensäädinelektroniikan suunnittelulla. Ohjainlaitteen jännitteensäädinelektroniikka muodostuu kahdesta sarjaan kytketystä jännitteensäätimestä. Ensimmäinen, A5975D, muuttaa akun jännitteen 5 volttiin, joka säädetään toisessa säätimessä mikrokontrollerin käyttämään 3,3 volttiin.

Jännitteensäädin A5975D kytkettiin kuvan 21 referenssikytkentäkaavion mukaisesti.



Kuva 21. A5975D-jännitteensäätimen referenssipiirikaavio [18, s. 1].

Kaaviossa nähdään jännitteensäätimen tulojännitteen suodatukseen tarvittavat kondensaattorit C1 ja C2. Pinniin 5 kytketty kahden vastuksen R1 ja R2 muodostama takaisinkytkentä säätää jännitteensäätimen lähtöjännitteen kaavan 3 mukaan [18, s. 18].

$$V_{OUT} = V_{ADJ} * (1 + \frac{R_1}{R_2}) \quad (3)$$

V_{ADJ} on jännitteensäätimen sisäinen referenssijännite, 1,235 V
R_1 on takaisinkytkentävastuksen arvo
R_2 on toisen takaisinkytkentävastuksen arvo

A5975D-jännitteensäätimen lähtöjännite asetettiin 5 voltin nimellisjännitteeseen 5,6:n ja 1,8 kilo-ohmin vastuksilla. Jännitteensäädinelektroniikan jälkeen siirryttiin määrittelemään mikrokontrollerin ja sen apukomponenttien sekä kideoskillaattorin kytkennät.

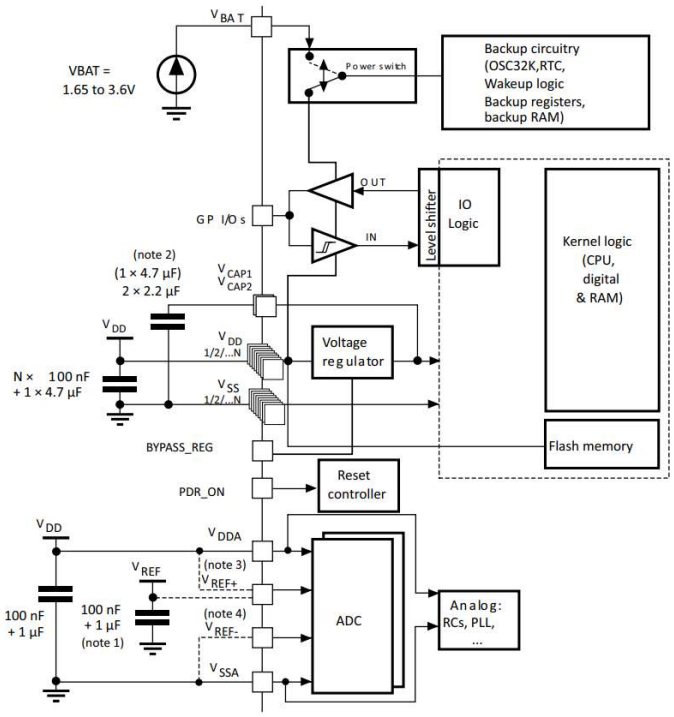
Kuvan 22 virtalähdekaaviossa on esitetty STM32F4-sarjan mikrokontrollerin vaatimat apukomponentit ja niiden kytkennät:

- Jokaista V_{DD} pinniä kohden sijoitettiin 100 nF:n kondensaattori pinnin lähelle sekä yhden pinnin lähelle lisäksi 4,7 µF:n kondensaattori.

- V_{DDA}-pinnin läheisyyteen sijoitettiin 100 nF:n ja 1 µF:n kondensaattorit, lisäksi lisättiin 39 nH:n kela jännitteen tasoitusta varten.

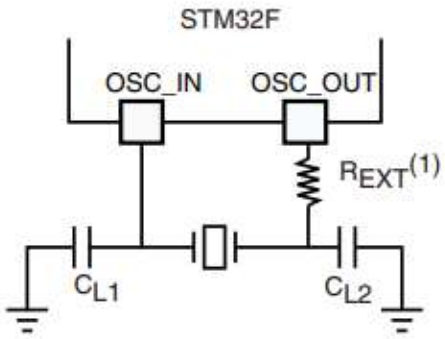
- V_{SS}-pinnit on liitettävä maahan.

- V_{CAP}-pinnin läheisyyteen on liitettävä 4,7 µF:n kondensaattori.



Kuva 22. STM32F446x:n virtalähdekaavio [19, s. 10].

Seuraavaksi määriteltiin mikrokontrollerin kellolähteeksi valitun kideoskillaattorin kytkentä mikrokontrollerin tuloihin kuvan 23 mukaisesti.



Kuva 23. Kideoskillaattorin kytkentä mikrokontrolleriin [19, s. 32].Kideoskillaattori tarvitsee toimiakseen kaksi kondensaattoria C_{L1} ja C_{L2} sekä tehoa rajoittavan vastuksen R_{EXT}. Kondensaattorien arvot riippuvat kideoskillaattorin kuor-

makapasitanssista (*load capacitance*) sekä oskillaattorikytkennän kapasitanssista (*stray capacitance*). Kondensaattorien arvot lasketaan kaavalla 4, joka olettaa niiden olevan yhtä suuret. [20, s.12.]

$$CL1 = 2(CL - CS) \quad (4)$$

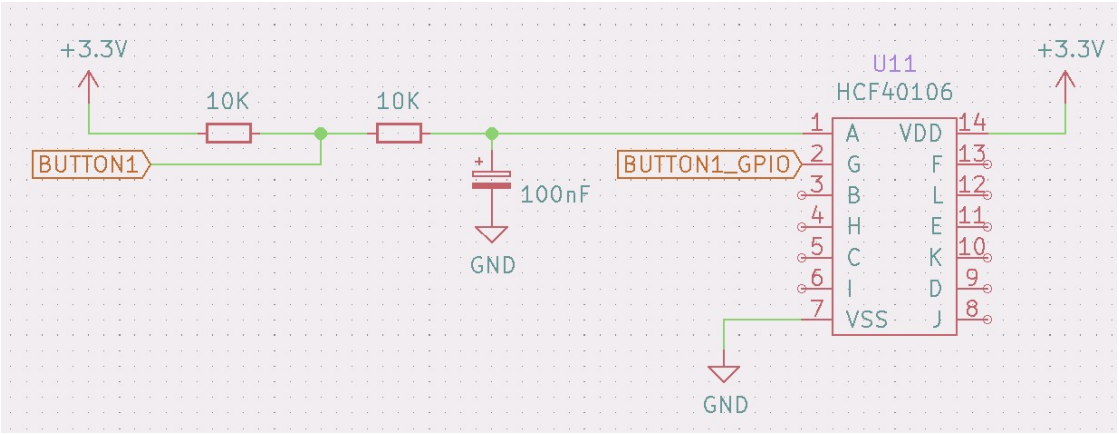
C_{L12} on kondensaattoreiden arvo
C_L on kideoskillaattorin kuormakapasitanssi, ilmoitettu komponentin tietolehdellä, 8 pF
C_S on *stray capacitance,* joka muodostuu mikrokontrollerin tulojen välille ja on arvioitu 4 pF:n suuruiseksi

Sijoittamalla kaavaan tunnetut arvot saadaan laskettua kondensaattoreiden arvo.

$$C_{L12} = 2(8\,pF - 4\,pF) = 8\,pF \quad (4)$$

Kideoskillaattorikytkennän jälkeen mikrokontrolleriin liittyvä elektroniikka oli valmis ja siirryttiin painikkeiden signaalien suodatukseen.

Ohjainlaitteen painikkeiden signaalit suodatetaan Schmitt-kytkimestä koostuvalla kytkennällä. Yksi Schmitt-kytkin liittää kuusi mikrokontrollerin tuloa kuuteen painikkeeseen kuvan 24 kaavion mukaisesti.



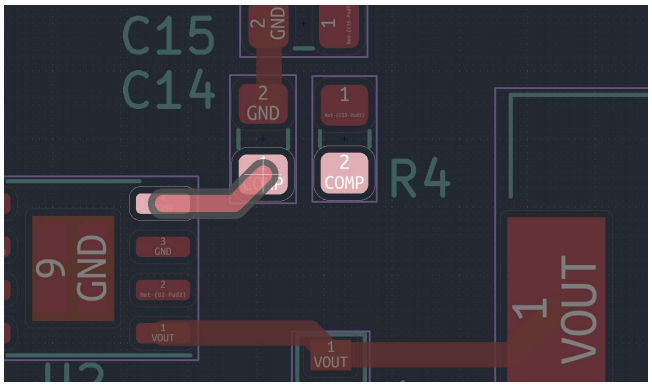
Kuva 24. Kytkimien signaalin suodatuselektroniikan kytkentäkaavio.

Tämän jälkeen piirikaaviosuunnittelu viimeisteltiin liittämällä moottorinohjainpiirit, 12 voltin lähtöjen kytkimet ja CAN-vastaanotin-lähetin mikrokontrolleriin.

3.3.2 Piirilevysuunnittelu

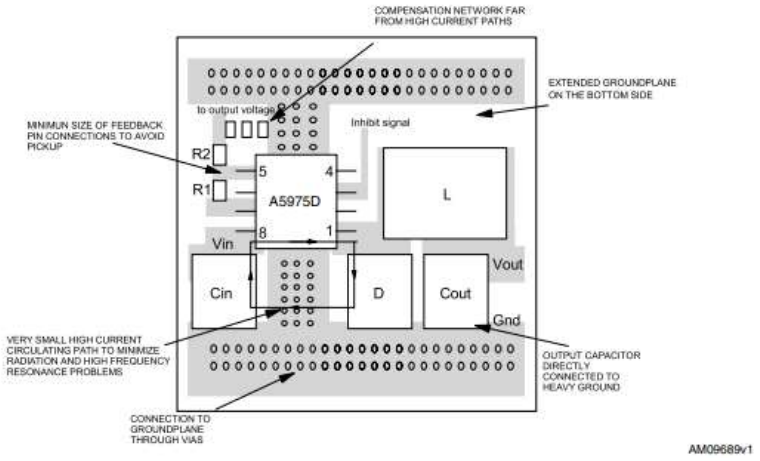
Piirikaavion valmistumisen jälkeen siirryttiin piirilevysuunnitteluun KiCAD Pcbnew -moduulilla. Piirilevystä tehtiin nelitasoinen, kahden keskimmäisen ollessa maa ja 3,3 voltin tasot. Ylin ja alin taso määritettiin signaalitasoiksi. Tämä yksinkertaistaa komponenttien reitittämistä ja estää maaerojen syntymistä, koska komponenteilla on lyhyt reitti maahan läpivientien kautta.

Työ aloitettiin mikrokontrollerin ja sen apuelektroniikan asettelulla piirilevyyn. Tämän jälkeen sijoitettiin moottorinohjaimet, 12 voltin High Side -kytkin, jännitteensäädinelektroniikka, CAN-vastaanotin-lähetin ja painikkeiden suodatuselektroniikka. Lopuksi liittimet sijoitettiin piirilevyyn. Komponenttiryhmien sijoittamisen jälkeen niiden osat reititettiin kuvan 25 esittämällä tavalla.



Kuva 25. Jännitteensäädinelektroniikan komponenttien reititys KiCAD-ohjelmassa.

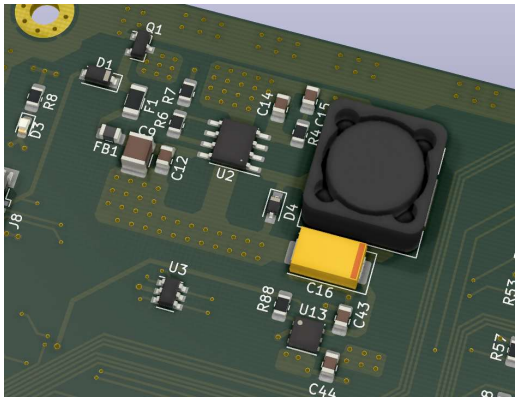
Komponenttien sijoittelussa ja reitityksessä hyödynnettiin niiden tietolehtiä, esimerkkinä kuvan 26 jännitteensäätimen A5975D referenssiasettelu, jota käytettiin mallina jännitteensäätöelektroniikan sijoittelussa.



Kuva 26. 12 voltin jännitteensäätimen referenssiasettelu [18, s. 27].

Asettelukaaviossa huomattavaa on apukomponenttien sijoittelu lähelle säädintä sekä maan liittäminen piirilevyn maatasoon läpivientien avulla. Myös reitti tulo- ja lähtöpinnien välillä pyritään pitämään lyhyenä häiriöiden minimoimiseksi.

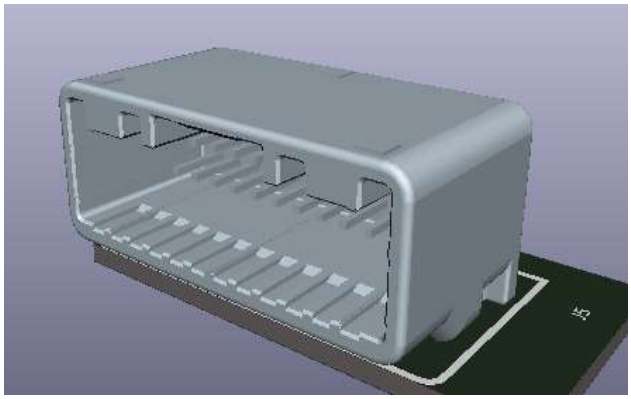
Kuvassa 27 on valmis jännitteensäätöelektroniikka KiCAD-ohjelmassa. Kuvassa näkyvät 5 V:n ja 3,3 V:n jännitteensäätimet (U2 ja U13).



Kuva 27. Ohjainlaitteen jännitteensäädinelektroniikka KiCAD-ohjelmiston 3D-näkymässä.

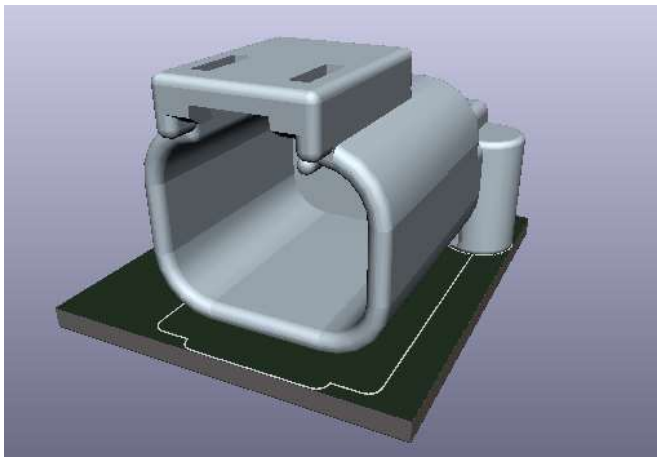
3.4 Liittimet

Ohjainlaitteen liittimien valinnassa otettiin huomioon ajoneuvoympäristö asettamat haasteet, kuten kosteus, pöly ja vaihtelevat lämpötilat. Ohjainlaitteen toiminnot jaettiin kolmeen erityyppiseen piirilevykiinnitteiseen liittimeen. Matalan virran tulot ja lähdöt kuten painikkeet, rajakytkimet ja CAN-väyläjohtimet liitettiinyhteen TE-Connectivityn TE1318853–2-mallin ajoneuvoliittimeen (kuva 28). Liitin tarjoaa 24-kontaktia kompaktissa kahden rivin asettelussa. Liittimen vastakappaleena on TE-1318917–1-liitin.



Kuva 28. Matalan virran liitin TE1318853–2.

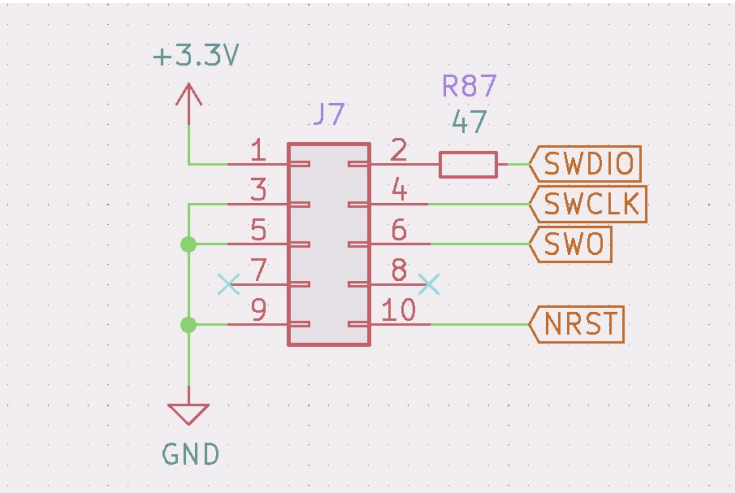
Moottoreiden liitäntään valittiin neljä kahden kontaktin Deutsch DTF13-2P-liitintä, jotka ovat ihanteellisia kevyiden kuormien ohjaamiseen (kuva 29). Liittimen virrankesto on 13 A, ja se kestää hyvin vettä IP67-luokituksen ansiosta. Liittimeen liitetään vastakappaleeksi DT06-2S-liitin. 12 V:n lähtöjen liitäntään valittiin saman sarjan nelikontaktinen Deutsch DTF13-4P-liitin ja sen vastakappaleeksi DT06-4S-liitin.



Kuva 29. Deutsch DTF13-2P, 2-napainen ajoneuvokäyttöön sopiva liitin ovi- ja ikkunamoottoreille.

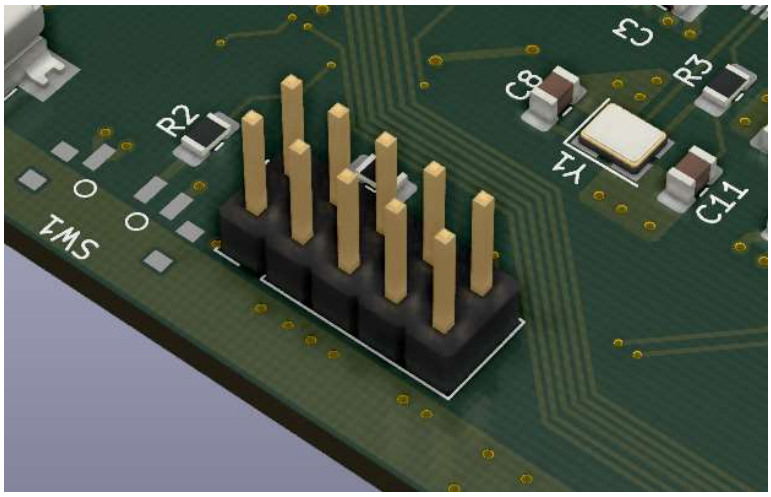
Mikrokontrolleri kommunikoi tietokoneen kanssa USB-yhteyden välityksellä, jota voidaan käyttää esimerkiksi virheenjäljitykseen ohjelmistokehityksen aikana. Yhteyden liittimeksi valittiin yleinen Mini USB -liitin.

Mikrokontrolleri ohjelmoidaan Serial Wire Debug-yhteydellä, joka vaatii seuraavat kuvan 30 esittämät liitännät STLink-ohjelmointipiirin ja mikrokontrollerin välillä: *SWDIO* ja *SWCLK*, joiden välillä itse kommunikaatio tapahtuu, *NRST* eli resetointipinni, 3,3 V:n käyttöjännite sekä maa. *SWO* eli Single Wire Output on valinnainen liitäntä, jota voidaan käyttää ohjelman virheenjäljitykseen.[21.]



Kuva 30. Serial Wire Debug -ohjelmointiliitäntä.

Mikrokontrollerin ohjelmointiliittimenä käytettiin kuvan 31 kahta yleiskäyttöistä 10 kontaktin pinnirimaa, joka on käytössä myös ST-Link-ohjelmointipiirissä.



Kuva 31. Mikrokontrollerin Serial Wire Debug -ohjelmointiliitin.

4 Ohjainlaitteen ohjelmointi

4.1 Ohjelman suunnittelu

Ohjelman suunnittelu alkoi tarkastelemalla ohjainlaitteen vaatimuksia, joiden pohjalta ohjelman ominaisuudet ja vaatimukset määritettiin. Ohjelmalta vaadittujen ominaisuuksien perusteella määriteltiin ohjelman käyttämät mikrokontrollerin oheis- ja ohjainlaitteet (*peripherals*). Näitä ovat esimerkiksi analogia-digitaalimuuntimet, ajastimet sekä kommunikaatioliittymät kuten USB ja CAN. Oheislaitteiden alustaminen käydään läpi työn luvussa 4.2

Tarvittavien oheislaitteiden selvityksen jälkeen ohjainlaitteen toiminnot, kuten ovien ja ikkunoiden ohjaus, jaettiin funktioihin. Samalla ohjainlaitteen pääohjelmarakenne määriteltiin. Ohjelma alkaa muuttujien alustamisella, jonka jälkeen ohjelma käy läpi oheislaitteiden ja GPIO-pinnien alustusfunktiot. Alustusten jälkeen ohjelma siirtyy silmukkaan, johon on sijoitettu ohjainlaitteen toimintojen funktiot. Ohjelman viiveettömän toiminnan takia painikkeiden tiloja ei lueta pääohjelman silmukassa vaan tilat luetaan mikrokontrollerin laitteistokeskeytyksiä käyttämällä, joita tarkastellaan luvussa 4.2.

4.1.1 CAN-viestiprotokollan määrittely

Ohjainlaite asennetaan ajoneuvon tietoväylään, jossa se kommunikoi muiden ohjainlaitteiden kanssa. Ohjainlaitteiden välisen kommunikoinnin säännöt määritetään CAN-viestiprotokollassa, joka selittää CAN-viestien lähettäjän, vastaanottajat, rakenteen ja käyttötarkoituksen. Taulukossa 2 on kuvattu osa ohjainlaitteen CAN-viestiprotokollasta.

Taulukko 2. CAN-viestiprotokolla

CAN Bus protocol, CAN High Speed, 500 kbit/s						
Tx	**Rx**	**ID**	**DLC**	**Period/On**	**Signal**	
Steering Column Control Module	Central Body Control Module	0x88	2	On state change	Byte 0	Bit 0: Fog light state 0 - off, 1 - on
					Byte 1	0xFF
Steering Column Control Module	Central Body Control Module	0x89	1	On state change	Byte 0	Bit 0: right indicator, 1 - on, 0 - off Bit 1: left indicator, 1 - on, 0 - off Bit 2: High beam, 1 - on, 0 - off Bit 3: Bit 4 - 6: 010 - DLR, 001 - park, 101 - headlight on, 000 - light off

Yllä on kuvattu kaksi ohjauspylvään ohjainmoduulin (*Steering Column Control Module*) lähettämää CAN-viestiä, jotka ohjaavat ajoneuvon korin valaistusta. Korielektroniikan ohjainlaitteen vastaanottaessa CAN-viestin tunnisteella 0x88 se lukee datakentän ensimmäisen tavun ensimmäisen bitin, joka kertoo sumuvalon kytkentätilan. Ohjainlaite kytkee sitten sumuvalon päälle tai pois riippuen kytkentätilasta. Edellä mainittu CAN-viesti lähetetään aina ohjauspylvään painikkeiden kytkentätilan muuttuessa, joka säästää CAN-väyläliikennettä.

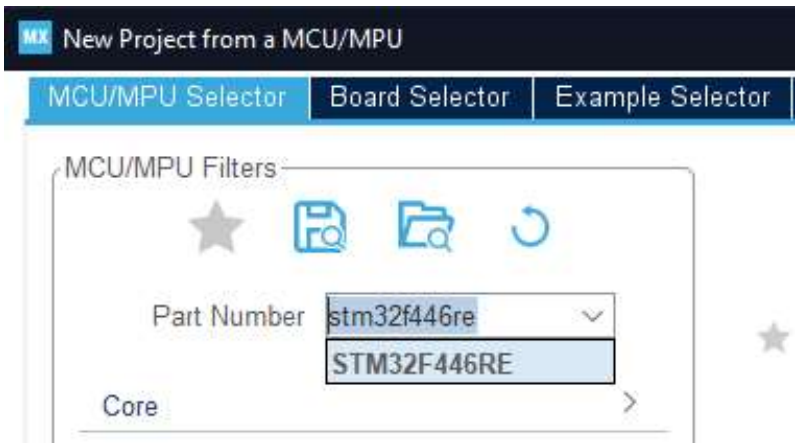
4.2 STM32-ohjelmointi Linux-ympäristössä

STM32-mikrokontrollereiden ohjelmointiin on useita eri ohjelmistotyökaluja ja kehitysympäristöjä (Integrated Development Environment), kuten IAR Embedded Workbench, Keil MDK-ARM, ja STM32 System Workbench [11, s. 1]. Edellä mainitut kehitysympäristöt ovat usein parhain vaihtoehto ja tarjoavat suoraviivaisen kehitysprosessin. Valmis kehitysympäristö ei kuitenkaan välttämättä toimi kaikilla käyttöjärjestelmillä, minkä takia ohjelmistokehitys tehtiin erillisillä

työkaluilla Linux-käyttöjärjestelmässä. Ohjelmistokehitykseen käytettävät työkalut eli *toolchain* muodostui STM32CubeMX-ohjelmistosta, Makefilesta, kääntäjästä (*GCC Arm Embedded*), ST Flash -työkalusta sekä tekstieditorista. Useimmat työkalut ovat julkaistu avoimen lähdekoodin lisenssillä, mikä helpottaa kehitysprosessia.

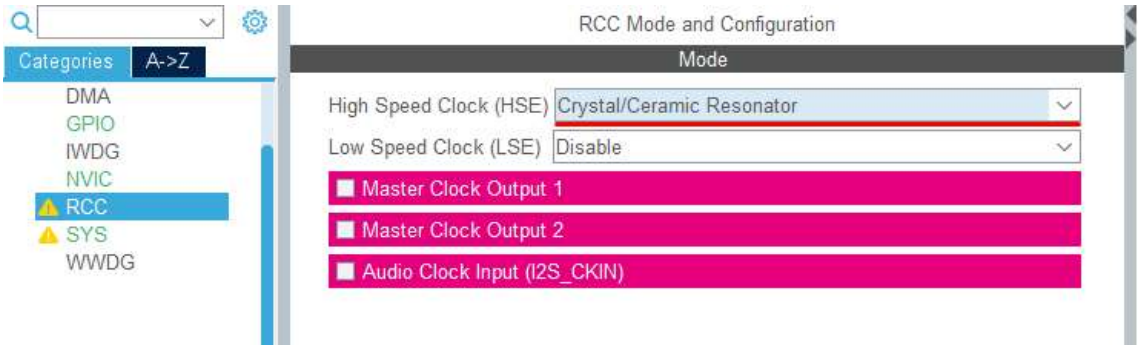
4.2.1 STM32CubeMX

Mikrokontrollerin oheislaitteiden ja pinnien alustaminen tehdään STM32CubeMX-ohjelmistoa käyttäen. Aluksi luodaan projekti ja valitaan käytettävä mikrokontrolleri tai kehitysalusta (kuva 32).



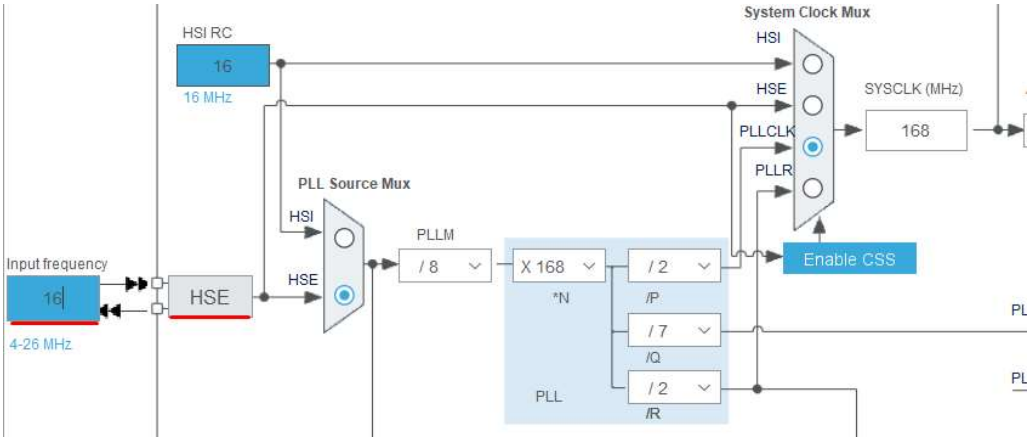
Kuva 32. STM32CubeMX-projektin luominen mikrokontrollerin mallin STM32F446-pohjalta.

Mikrokontrolleri voi käyttökohteesta riippuen käyttää useita eri kellolähteitä, joten alustus kannattaa aloittaa niistä. Valitsemamme STM32F446-mikrokontrolleri voi esimerkiksi käyttää mikrokontrolleriin sisäänrakennettuja (*High Speed Internal, Low Speed Internal*) tai ulkoisia (*High Speed External, Low Speed External*) kellolähteitä. Tässä projektissa käytettiin ulkoista 16 MHz:n kideoskillattoria (kuva 33).



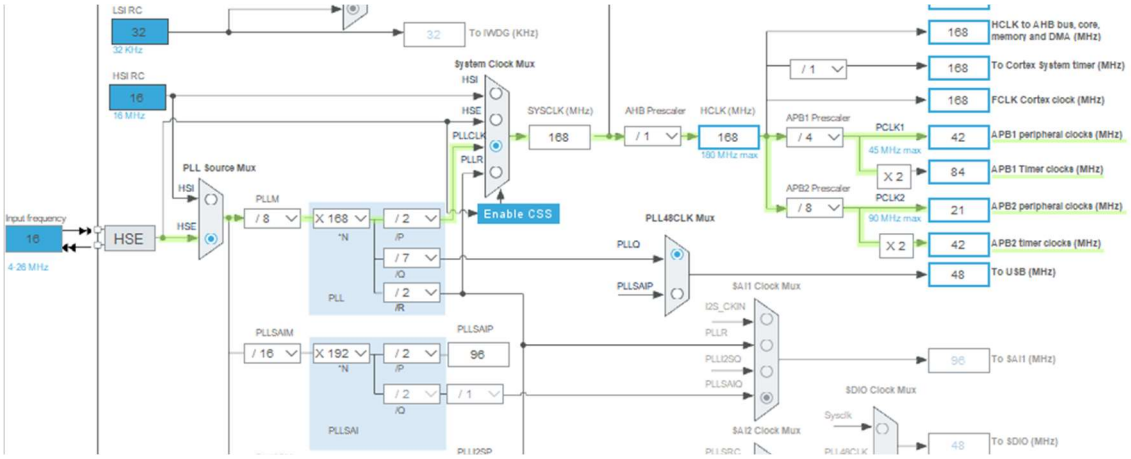
Kuva 33. Mikrokontrollerin kellolähteen asettaminen.

Ulkoisen kellotaajuuslähteen (*HSE, High Speed External crystal*) valinnan jälkeen avataan mikrokontrollerin kellotaajuuskaavio (*Clock Tree*) ja asetetaan HSE-kellotaajuus, 16 MHz, kuvan 34 mukaisesti.



Kuva 34. Ulkoisen kellolähteen taajuuden asetus 16 MHz:iin kellotaajuuskaaviossa.

Tehtyjä muutoksia voidaan tarkastella kuvan 35 kellotaajuuskaaviossa, joka näyttää käytetyn kellolähteen polun sekä mikrokontrollerin oheislaitteille johdetut kellotaajuudet.



Kuva 35. Mikrokontrollerin kellotaajuuskaavio.

Kellotaajuuskaaviosta näkee esimerkiksi APB1 (*Advanced Peripheral Bus*) -väylässä sijaitsevan CAN-väyläoheislaitteelle johdetun kellotaajuden, joka on asetettu 42 MHz:iin. Kellolähteen asettamisen jälkeen siirryttiin määrittelemään mikrokontrollerin tulot painikkeita varten.

GPIO *(General Purpose Input Output*) -pinnit ovat mikrokontrollerin yleiskäyttöisiä pinnejä, jotka ovat konfiguroitavissa tuloiksi tai lähdöiksi. Projektissa käytetyssä STM32F446RE-mikrokontrollerissa niitä on 50 kappaletta, joista jokainen on asetettavissa laitteistokeskeytys (*external interrupt*) tilaan. Ohjainlaitteeseen liitetyt painikkeet kytketään suodatuselektroniikan läpi näihin tuloihin, joiden tilaa mikrokontrolleri voi tarkkailla seuraavaksi esitellyin menetelmin.

Tulot voidaan lukea käyttämällä kyselymenetelmää (*polling*). Kyselymenetelmällä toteutettu ohjelma lukee jokaisella ohjelmasyklillä tulopinnin tilan, esimerkkikoodin 1 mukaisesti. Jos painike on painettu, ohjelma kytkee ledin päälle. Tämä menetelmä ei kuitenkaan ole tehokas tapa, etenkään jos tuloja on useita.

```
while (1){

    if (HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_4)){

         HAL_GPIO_TogglePin(GPIOA, UserLed1);

    }

}
```

Esimerkkikoodi 1. Painikkeen tilan lukeminen kyselymenetelmällä

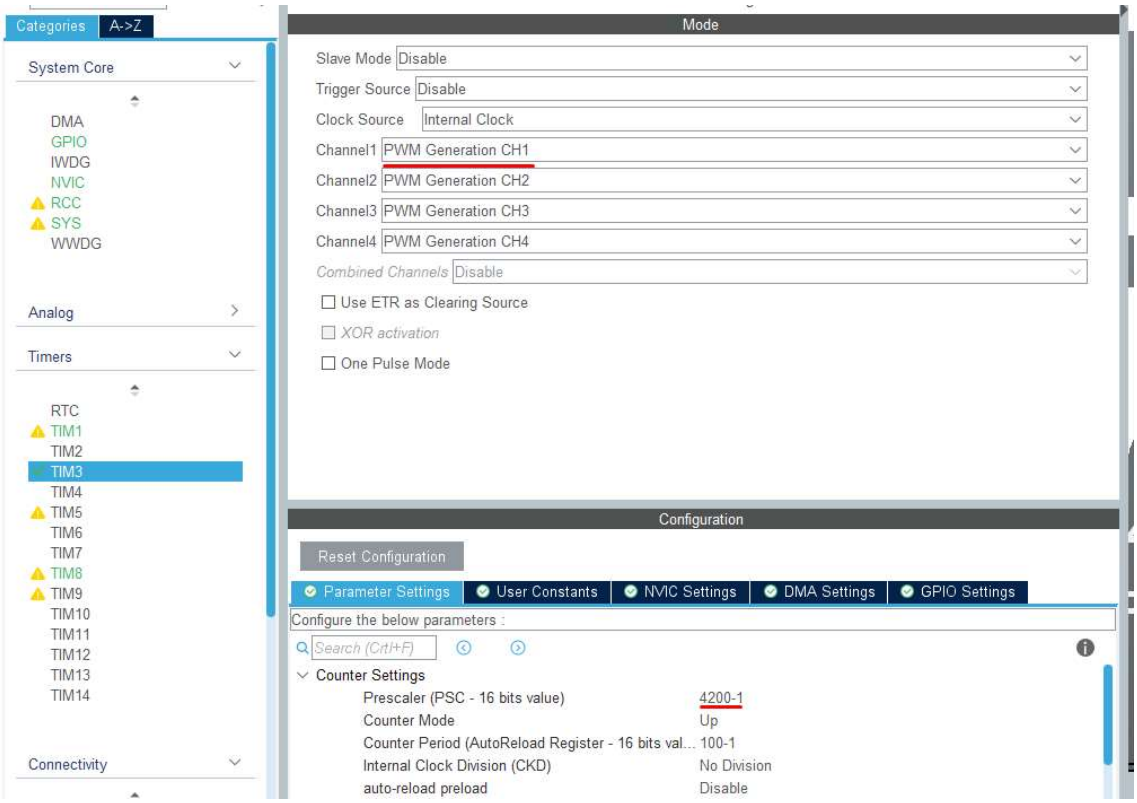
Huomattavasti parempi vaihtoehto on lukea tulot käyttäen laitteistokeskeytyksiä (*external interrupt*). Laitteistokeskeytystilaan asetetun tulon signaalin muuttuminen laukaisee keskeytystapahtuman, jossa suoritin keskeyttää pääohjelman ja suorittaa keskeytyskäsittelijään (*interrupt handler*) määritetyn funktion. Tämän jälkeen suoritin jatkaa pääohjelman suorittamista, kohdasta missä keskeytys tapahtui. Esimerkkikoodissa 2 on mikrokontrollerin keskeytyskäsittelijään määritetty funktio.

```
void HAL_GPIO_EXTI_Callback( uint16_t GPIO_Pin ){

    switch (GPIO_Pin){

         case (GPIO_PIN_5){

             HAL_GPIO_TogglePin(GPIOA, UserLed1);

             break;

         }

    }

}
```

Esimerkkikoodi 2. Kesketyskäsittelijän kutsuma funktio laitteistokeskeytyksen tapahtuessa.

Funktio saa parametriksi pinnin, jossa signaalinmuutos tapahtuu, jolloin samalla funktiolla voidaan käsitellä useaa tuloa. Jos keskeytyksen laukaissut tulo on GPIO_PIN_5, ohjelma vaihtaa ledin tilan. Tämä säästää suorittimen resursseja, sillä tuloja ei lueta jokaisella ohjelmasyklillä. Tulojen määrittelyn jälkeen siirryttiin mikrokontrollerin ajastimen alustamiseen.

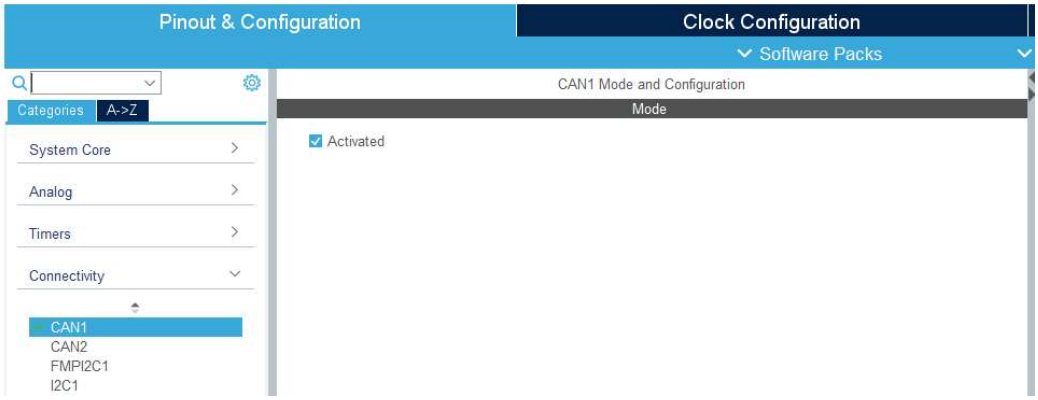
VNH5019- moottorinohjauspiirejä ohjataan 20 kHz:n PWM (Pulse Width Modulation) -signaalilla, joka generoidaan TIM3-ajastimessa. Ajastimen TIM3 käyttämä kellotaajuus on 84 MHz, ja käyttämällä kellotaajuuden jakolukua (*prescaler*) 4200 saadaan 20 kHz:n taajuudella oleva PWM-signaali. Kuvassa 36 asetetaan TIM3-ajastimen neljä kanavaa PWM-lähtötilaan, joilla ohjataan neljää VNH5019-moottorinohjainpiiriä.



Kuva 36. TIM3-ajastimen asettaminen PWM-signaalitilaan.

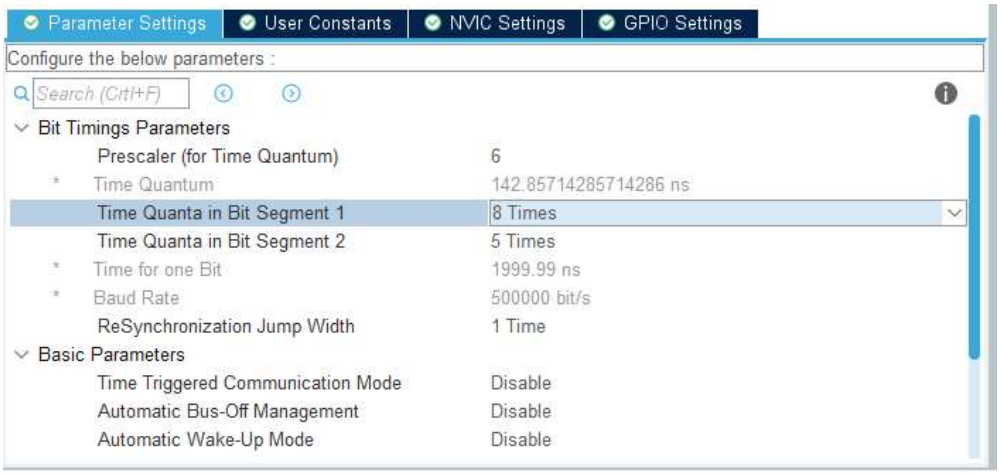
Ajastimen PWM-lähtöjen asettamisen jälkeen siirryttiin alustamaan ohjainlaitteen kommunikointilaitteistoja.

Ohjainlaite kommunikoi ajoneuvon muiden ohjainlaitteiden kanssa CAN-väylän välityksellä, jonka ohjaimen asetukset määriteltiin seuraavaksi. Määrittely aloitettiin mikrokontrollerin CAN1-ohjaimen aktivoinnilla kuvan 37 mukaisesti.



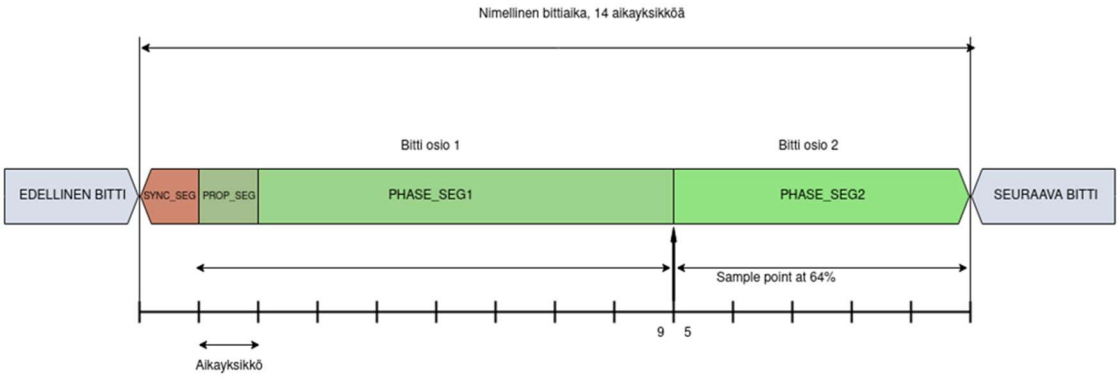
Kuva 37. CAN1-oheislaitteen aktivointi.

CAN-ohjain tarvitsee toimiakseen oikeat bittiajoitusparametrit, jotka riippuvat ohjaimen käyttämästä kellotaajuudesta ja vaaditusta CAN-väylän tiedonsiirtonopeudesta. Kuvassa 38 on CAN1-oheislaitteelle valitut bittiajoitusparametrit, joita tutkitaan seuraavaksi.



Kuva 38. Bittiajoitusparametrit, bitin pituus on 8 + 5 + 1 aikayksikköä (*time quanta).*

CAN-väylän bittien ajoitus (*bit timing*) määritetään ISO-11898-2 standardissa seuraavanlaiseksi. Yksi CAN-väylässä lähetetty bitti muodostuu kuvan 39 mukaisesti neljästä osasta, *SYNC_SEG, PROP_SEG, PHASE_SEG1 ja PHASE_SEG2*. Osat koostuvat bitin perusaikayksiköistä (*time quanta*). Aikayksikön pituuden määrittää mikrokontrollerin CAN-ohjaimen käyttämä kellotaajuus. [21.]



Kuva 39. CAN-bitin muodostavat osiot.

CAN-väylässä lähetetty bitti alkaa SYNC_SEG-osiolla, jossa CAN-ohjain synkronoi itsensä väylän taajuuteen sen signaalin laskevalla reunalla. Sitä seuraa kiinteä osio PROP_SEG, joka kompensoi CAN-väylästä johtuvaa viivettä signaalin lähettäjän ja vastaanottajan välillä. PHASE_SEG1- ja PHASE_SEG2-osioita käytetään ohjainlaitteiden kellolähteiden toleranssien kompensointiin. CAN-ohjain voi signaalinerosta riippuen pidentää PHASE_SEG1-osiota tai lyhentää PHASE_SEG2-osiota. Osioiden ja itse aikayksikön pituudella vaikutetaan CAN-väylän tiedonsiirtonopeuteen ja väylän maksimipituuteen.

CAN-ohjaimen kellotaajuudesta voidaan laskea CAN-väylän tiedonsiirtonopeus kaavalla 5.

$$bus\_bps = \frac{can\_peripheral_{hz}}{prescaler * (nomin \quad \_bit\_time)} \tag{5}$$

$can\_peripheral_{hz}$ on mikrokontrollerin CAN-ohjaimen käyttämä kellotaajuus
$prescaler$ on CAN-oheislaitteen kellotaajuuden jakaja
$nominal\_bit\_time$ on bitin pituus aikayksiköissä

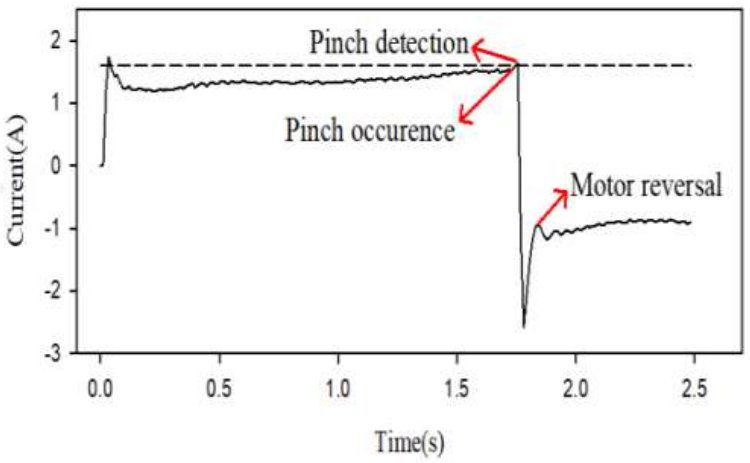
Näin ohjainlaitteen CAN-väylän nopeudeksi saadaan

$$bus_{bps} = \frac{42000000\ Hz}{6*(8+5+1)} = 500\ kbit/s \quad (5)$$

Edellisestä kaavasta nähdään myös, että nostamalla aikayksiköiden määrää ($nominal\_bit\_time$), väylän tiedonsiirtonopeus pienenee. CAN-ohjaimen asetusten jälkeen siirryttiin moottorien virranmittaukseen käytettävän analogia-digitaalimuuntimen alustamiseen.

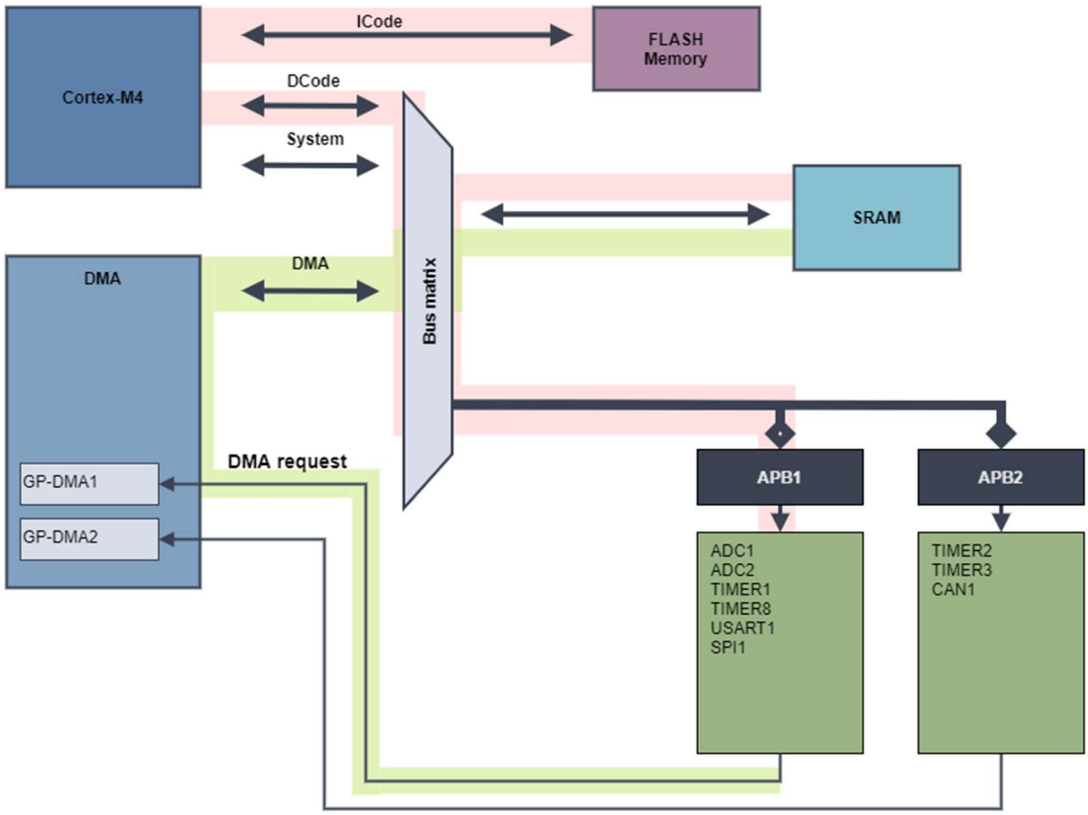
AD-muunnin eli analogia-digitaalimuunnin on mikrokontrollerin oheislaite, jolla voidaan mitata jännitteen suuruutta. Mikrokontrolleri käyttää yhden AD-muuntimen neljää kanavaa mitatakseen mittauselektroniikan avulla neljän moottorinohjainpiirin käyttämää virtaa. Virranmittauksella voidaan toteuttaa ikkunoiden *anti-pinch*-turvamekanismi, joka pysäyttää moottorin sen kohdatessa epänormaalin vastuksen.

Ohjainlaite mittaa ikkunaa sulkiessa ikkunamoottorin käyttämää virtaa. Virran noustessa yli sille asetetun raja-arvon ikkuna pysäytetään ja avataan. Kuva 40 havainnollistaa pinch-tapahtumaa.



Kuva 40. Ikkunan anti-pinch-toiminto [23].

Mikrokontrolleri voi lukea AD-muuntimelle johdetun jännitteen arvon muistiin usealla eri tavalla, joista kuvan 41 kaaviossa vertaillaan kahta. Tiedon luku voidaan toteuttaa ohjelmakoodissa, jolloin suoritin lukee suoritusohjeen flash-muistista, lukee AD-muuntimen arvon ja kirjoittaa sen käyttömuistiin. Tämä ei kuitenkaan ole tehokas tapa, sillä suoritin joutuu tekemään useita laskutoimituksia. Ratkaisu on käyttää oikosiirtoa (*Direct Memory Access*). DMA-ohjain mahdollistaa tiedon lukemisen oheislaitteelta, tässä tapauksessa AD-muuntimelta, suoraan mikrokontrollerin käyttömuistiin (*SRAM*), kuljettamatta sitä suorittimen (*Cortex-M4*) kautta. Tällöin tieto siirtyy nopeasti ja suoritin on vapaa tekemään muita laskutoimituksia.

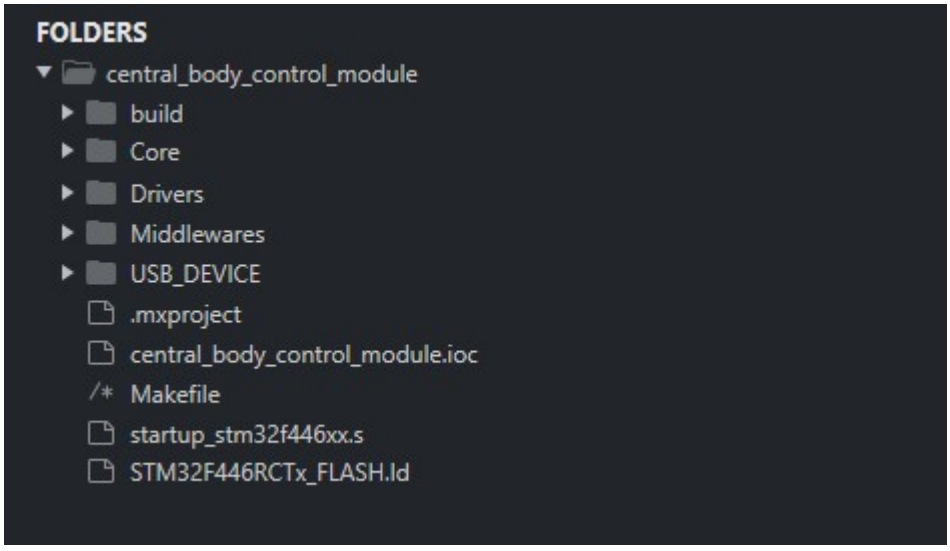


Kuva 41. AD-muuntimen luku oikosiirto- ja ohjelmistomenetelmällä. Ohjelmistomenetelmä merkitty punaisella ja oikosiirtomenetelmä vihreällä.

4.2.2 Lähdekoodin luominen projektitiedostoista

Mikrokontrollerin pinnien ja oheislaitteiden määrittelyn jälkeen siirrytään Project Manager -välilehdelle, jossa valitaan toolchain/IDE-valikosta käytettäväksi työkaluksi Makefile. Tämän jälkeen lähdekoodit luodaan valitsemalla Generate Code. STM32CubeMX-ohjelma luo määritysten perusteella tarvittavat lähdekooditiedostot ja oheislaitteiden käyttämiseen tarvittavat ajurit (*drivers*), sekä ohjelman kääntämiseen käytettävän Makefile-tiedoston (kuva 42).

Makefile-tiedosto on Linux-käyttöjärjestelmän Make-työkalun suorittama tiedosto, johon on määritelty muuttujia ja komentorivillä suoritettavia käskyjä [24]. STM32CubeMX-ohjelman luomassa Makefilessa on määritelty muuttujat, ja niitä käyttävät komentorivikäskyt ohjelman kääntämiseksi ja kirjoittamiseksi mikrokontrollerin flash-muistiin. Komento *make all* suorittaa Arm Embedded GCC-kääntäjän, joka kääntää ohjelman lähdekoodin mikrokontrollerin suoritettavaksi binaaritiedostoksi. Komento *make flash* puolestaan suorittaa ST Flash -työkalun, jolla kirjoitetaan ohjelma mikrokontrollerin flash-muistiin.

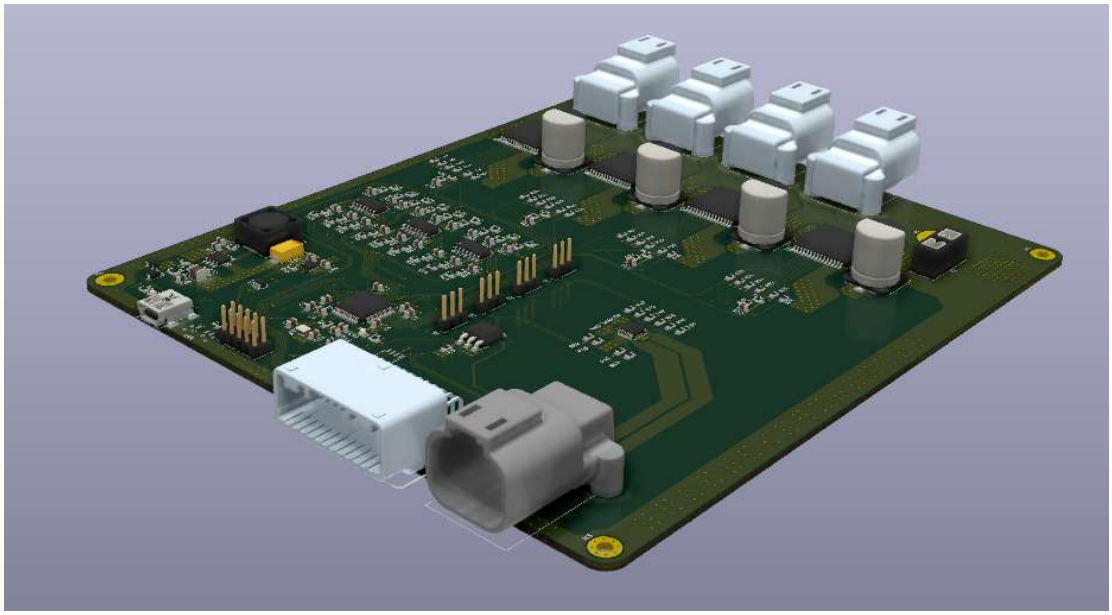


Kuva 42. STM32CubeMX-projektin luomat lähdekoodi- ja ajuritiedostot sekä kansiorakenne.

5 Yhteenveto

Ohjainlaitteen prototyypin suunnittelu ja valmistus opinnäytetyönä oli hyvin laaja ja siinä yhdistyi useita aihealueita. Projekti alkoi ohjainlaiteen ominaisuuksien määrittämisellä, jossa merkittävimmät vaatimukset asetti ajoneuvoympäristö. Ohjainlaitteen soveltuvuus ajoneuvokäyttöön saavutettiin projektin eri vaiheissa tehdyillä suunnitteluratkaisuilla. Sen komponenteiksi valittiin AEC-Q100- ja Q200-hyväksyttyjä, ajoneuvokäyttöön suunniteltuja pintaliitoskomponentteja sekä ajoneuvoliittimiä. Ohjainlaite suojattiin ylijännitteeltä ja vastanapaiselta kytkennältä erillisellä suojauselektroniikalla ja mikrokontrollerin tulot suodatettiin Schmitt-kytkimiä hyödyntävällä suodatuselektroniikalla. Ohjainlaitteen ohjelman suunnittelussa hyödynnettiin mikrokontrollerin laitteistokeskeytyksiä ja DMA-ohjainta vikasietoisen ja viiveettömän toiminnan takaamiseksi.

Lopputuloksena oli ajoneuvokäyttöön sopiva ja vaatimustenmukainen ohjainlaitteen prototyyppi, jolla voidaan ohjata ajoneuvon ovien ja ikkunoiden moottoreita, sekä valaistusta CAN-väylän välityksellä (kuva 43).



Kuva 43. Ohjainlaitteen 3D-malli KiCAD-ohjelmistossa.

Lähteet

1. Mathur, Sonali & Malik, Shaily. 2010. Advancements in the V-Model. International Journal of Computer Applications, Volume 1 – No. 12.

2. Interference Technology. 2020. Automotive EMC Guide. Verkkoaineisto. <https://interferencetechnology.com/wp-content/uploads/2020/05/2020_Automotive_EMC_Guide.pdf>. Luettu 19.11.2021.

3. Steffka, Mark. 2008. Automotive EMC Introduction and Overview. University of Michigan-Dearborn Electrical and Computer Engineering Department.

4. Guttowski, Stephan. 2003. EMC Issues in Cars with Electric Drives. 2003 IEEE Symposium on Electromagnetic Compatibility.

5. AEC Documents. Verkkoaineisto. Automotive Electronics Council. <http://www.aecouncil.com/AECDocument.html>. Luettu 1.6.2021.

6. Failure Mechanism Based Stress Test Qualification for Multichip Modules In Automotive Applications. 2017. Verkkoaineisto. Automotive Electronics Council. <http://www.aecouncil.com/Documents/AEC-Q104_Rev-.pdf>. Luettu 23.5.2021.

7. JESD22-A105D. 2020. JEDEC Solid State Technology Association.

8. ISO 11898-2:2016. Road vehicles — Controller area network (CAN) — Part 2: High-speed medium access unit. 2016. International Organization for Standardization.

9. ISO 11898-3:2006. Road vehicles — Controller area network (CAN) — Part 3: Low-speed, fault-tolerant, medium-dependent interface. 2006. International Organization for Standardization.

10. Bosch Automotive Electrics and Automotive Electronics. 2014. E-kirja. Wiesbaden: Springer Fachmedien.

11. Data brief for STM32 Nucleo-64 boards. Verkkoaineisto. STMicroelectronics. <https://www.st.com/en/evaluation-tools/nucleo-f446re.html>. Luettu 11.2.2021.

12. Crystal Units Surface Mount Type CX3225SA. 2021. Verkkoaineisto. Kyocera. <https://ele.kyocera.com/assets/products/crystal-device/cx3225sa_e.pdf>. Luettu 10.6.2021.

13. VNH5019A-E Automotive Fully Integrated H-Bridge Motor Driver. 2017. Verkkoaineisto. STMicroelectronics. <https://www.st.com/resource/en/datasheet/vnh5019a-e.pdf>. Luettu 24.5.2021.

14. VNQ7140AJ Quad channel high-side driver with MultiSense analog feedback for automotive applications. 2015. Verkkoaineisto. STMicroelectronics. <https://www.st.com/resource/en/datasheet/vnq7140aj.pdf>. Luettu 20.5.2021.

15. L9615 CAN Transceiver. 2013. Verkkoaineisto. STMicroelectronics. <https://www.st.com/resource/en/datasheet/l9615.pdf>. Luettu 2.5.2021.

16. Harris, David & Harris, Sarah. 2007. Digital Design and Computer Architecture. E-kirja. Morgan Kaufmann Publishers.

17. Debouncing Fun with Schmitt Triggers and Capacitors. 2018. Verkkoaineisto. <https://mansfield-devine.com/speculatrix/2018/04/debouncing-fun-with-schmitt-triggers-and-capacitors/>. Luettu 30.6.2021

18. A5975D Up to 3 A step-down switching regulator for automotive applications. 2011. Verkkoaineisto. STMicroelectronics. <https://www.st.com/resource/en/datasheet/a5975d.pdf>. Luettu 16.4.2021.

19. Getting started with STM32F4xxxx MCU hardware development. 2018. Verkkoaineisto. STMicroelectronics. <https://www.st.com/resource/en/application_note/dm00115714-getting-started-with-stm32f4xxxx-mcu-hardware-development-stmicroelectronics.pdf>. Luettu 22.5.2021.

20. Oscillator design guide for STM32 MCUs. 2021. Verkkoaineisto. STMicroelectronics. <https://www.st.com/resource/en/application_note/cd00221665-oscillator-design-guide-for-stm8afals-stm32-mcus-and-mpus-stmicroelectronics.pdf>. Luettu 14.5.2021.

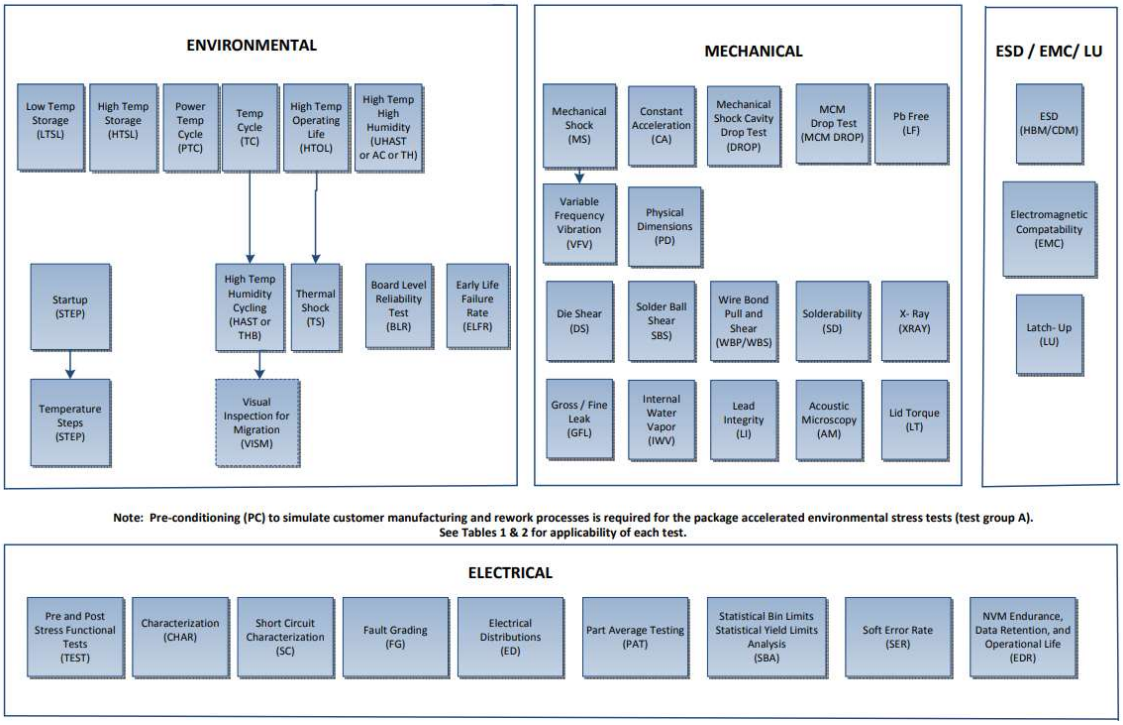
21. STM32 microcontroller debug toolbox. 2021. Verkkoaineisto. STMicroelectronics. <https://www.st.com/resource/en/application_note/dm00354244-stm32-microcontroller-debug-toolbox-stmicroelectronics.pdf>. Luettu 19.4.2021

22. Hartwich, Florian & Bassemir, Arming. 1999. The Configuration of the CAN Bit Timing. 6th International CAN Conference.

23. Wibowo, Wahyu Kunto. 2018. Automatic Threshold on Current based Anti-pinch Mechanism for Power Windows. Institute of Advanced Engineering And Science.

24. What is a Makefile? 1998. Verkkoaineisto. University of Pittsburgh. <http://www.sis.pitt.edu/mbsclass/tutorial/advanced/makefile/whatis.htm>. Luettu 9.11.2021.

AEC-hyväksynnän testausprosessi



Kuva 1. AEC-hyväksynnän testausprosessi [6, s. 13].