TAMPEREEN TEKNILLINEN YLIOPISTO

TIMO PIHLSTRÖM

LANGATON INERTIAMITTAUSYKSIKKÖ

Diplomityö

Tarkastajat: TkT Jussi Collin, prof. Jarmo Takala
Tarkastaja ja aihe hyväksytty
Tieto- ja sähkötekniikan tiedekuntaneuvoston kokouksessa 3.4.2013

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Sähkötekniikan koulutusohjelma
PIHLSTRÖM, TIMO: Langaton inertiamittausyksikkö
Diplomityö, 43 sivua, 9 liitesivua
Kesäkuu 2014
Pääaine: Sulautetut järjestelmät
Tarkastajat: TkT Jussi Collin, prof. Jarmo Takala
Avainsanat: inertiamittausyksikkö, bluetooth, kiihtyvyysanturi, gyroskooppi, magnetometri, GPS

Inertiamittausyksikkö on laite, joka pystyy mittaamaan liiketilaa ja asentoa. Yleensä näissä mittauksessa käytetään apuna kiihtyvyysanturia, gyroskooppia ja magnetometria. Jotta näistä useista antureista saatua mittaustietoa voidaan käyttää esimerkiksi asennon laskemiseen, täytyy ensin antureilta kysyä tieto, aikatahdistaa se muiden antureiden kanssa, kalibroida mittaustieto sekä lopulta suorittaa laskutoimenpide, joka kertoo asennon. Tämän työn tavoitteena on kuvata pienikokoisen ja langattomasti ohjattavan inertiamittausyksikön kehitystyötä, jonka käyttökohde on puettavissa sovelluksissa.

Työ on jaettu kolmeen osaan, joista ensimmäisessä kuvataan mittausyksikössä käytettyjen komponenttien valintaperusteita, piirilevysuunnittelua sekä mekaanista suunnittelua. Työn toisessa osassa kuvaillaan mittausyksikössä käytetty ohjelmistoarkkitehtuuri ja sen yksittäisten moduulien toiminta. Kolmas osio kertoo mittausyksikössä käytettyjen antureiden kalibrointimenetelmän ja tulokset menetelmän toimivuudesta. Tässä osiossa esitellään myös mittausyksikön käyttöä oikeissa mittausympäristöissä, joiden perusteella laitteen toimintakyvyn todetaan olevan riittävällä tasolla.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Electrical Engineering
PIHLSTRÖM, TIMO: Wireless Inertial Measurement Unit
Master of Science Thesis, 43 pages, 9 Appendix pages
June 2014
Major: Embedded Systems
Examiners: Dr.Tech. Jussi Collin, Prof. Jarmo Takala
Keywords: inertial measurement unit, bluetooth, accelerometer, gyroscope, magnetometer, GPS

Inertial measurement unit is a device that can measure orientation and the state of motion. Usually, these measurements combine the readings from accelerometer, gyroscope and magnetometer. Before the actual orientation can be calculated, the device needs the ask the information from the sensors, synchronize it with the data from other sensors and finally calibrate the sensor reading. The goal for this thesis is to describe the development process to produce a small-sized wirelessly controlled inertial measurement unit which is designed to work in wearable environments.

The thesis is divided in three parts. The first section describes the reasons for selecting the components, the printed circuit board design and mechanical design. The second part focuses on the software running on the device and describes the software architecture and the inner workings individual software components. The last part introduces a method which is used to calibrate the accelerometer and gyroscope readings. The results of this method are also presented. This section also describes how the measurement unit is used in real-life measurement situations. Based on these measurements, it can be said that the performance of the inertial measurement unit is at sufficient level.

ALKUSANAT

Haluan kiittää työn tarkastajia, Jarmo Takalaa ja Jussi Collinia, mahdollisuudesta kiinnostavaan aiheeseen, sekä vapaudesta laitteen tekemisen ja tämän työn kirjoitusvaiheen aikana. Lisäksi haluan kiittää Jussi Parviaista, jonka kanssa pääsin testaamaan toteutettua mittausyksikköä useissa mielenkiintoisissa mittauskohteissa. Kiitokset myös Jarkko Tuomelle ja Jayaprasad Bojjalle, joiden kanssa pohdin ratkaisuja työssä esiintyneisiin teknisiin ongelmiin.

Timo Pihlström

Päivämäärä

SISÄLLYS

TERMIT JA NIIDEN MÄÄRITELMÄT

SENSIN	Sensors in Mobile Consumer Devices -projekti
ESD	Electrostatic Discharge
SPP	Serial Port Profile
A/D-muunnin	Analogia-digitaalimuunnin
FAT	File Allocation Table
NMEA	National Marine Electronics Association
MAC	Media Access Control
ASCII	American Standard Code for Information Interchange
RS-232	Recommended Standard 232
SRAM	Static Random Access Memory
SDRAM	Synchronous Dynamic Random Access Memory
NiMH	Nikkelimetallihybridi
GPS	Global Positioning System
USB	Universal Serial Bus
MEMS	Micro Electro Mechanical Systems
I^2C	Inter-Integrated Circuit
SPI	Serial Peripheral Interface
SDIO	Secure Digital Input Output
JTAG	Joint Test Action Group

1. JOHDANTO

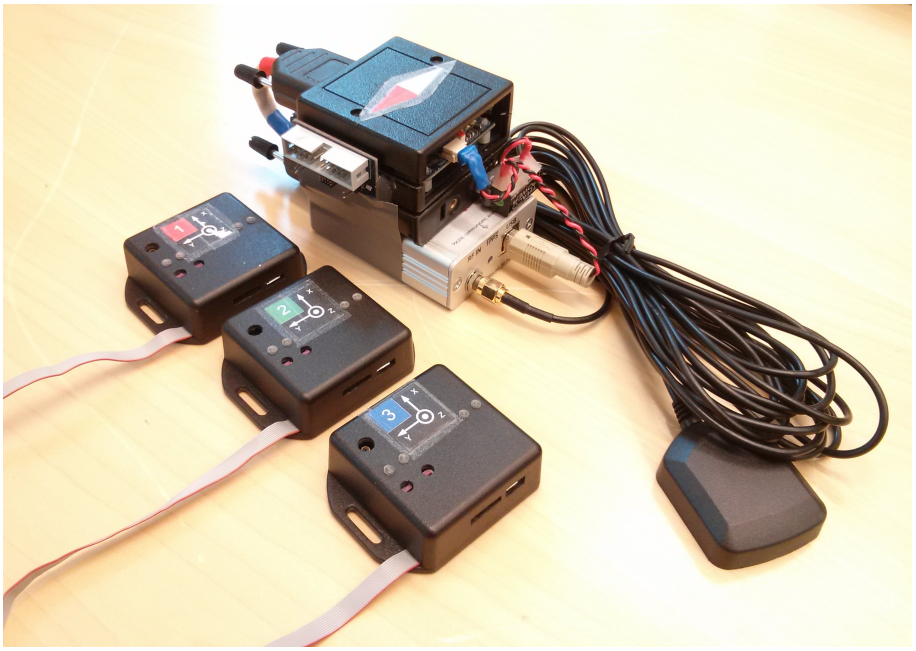
Tämä työ on tekninen raportti inertiamittausyksikön suunnitteluprosessista ja sen antureiden kalibroimista. Työ on toteutettu osana SENSIN-projektia (Sensors in Mobile Consumer Devices), jossa esiintyi tarve langattomasti ohjatuille ja aikasynkronoiduille pienikokoisille inertiamittausyksiköille. Tärkeä laitteen ominaisuus on myös se, että projektin yhdeydessä kehitettyjä algoritmeja voidaan toteuttaa itse laitteessa.

Työssä kuvattu laite pohjautuu ST Microelectronicsin STEVAL-MKI062V2 -kehitysalustaan [1], johon olen kehittänyt ohjelmistoa ja tehnyt laajennuksia vuoden 2012 toisen puoliskon aikana. Laajennukset kuitenkin lisäävät laitteen kokoa siihen pisteeseen, että laitetta ei pysty enää käyttämään tärkeimmissä käyttökohteissaan - esimerkiksi puettuna moottoripyöräkuskin vaatteisiin. Vanhan mittausyksikön kokoero verrattuna uusiin on esitetty kuvassa 1.1.

Laitteen ensimmäinen kehitysversio sisältää useita ongelmakohtia. Eräs niistä on laitteen teholähteenä toimivat neljä AA-kokoista NiMH-akkua (Nikkelimetallihybridi), jotka tuovat laitteelle kokoa ja painoa. Lisäksi akkujen lataus on vaikeaa sisäänrakennetun akkulaturin puuttumisen vuoksi. Toinen ongelmakohta on laitteen ulkoinen GPS-moduuli (Global Positioning System), joka liitetään kehitysalustaan sarjaliikennekaapelin ja tasomuuntimen kautta. GPS-laite tarvitsee myös ulkoisen antennin. Koska laite koostuu ulkoisilla kaapeleilla toisiinsa liitetyistä erillisistä komponenteista, on kokonaisuutta vaikea käyttää mittauskohteissaan. Laitteen käyttöliittymänä toimii yksi käyttäjänappi ja -ledi, jonka seurauksena siihen on vaikea toteuttaa useita käyttötiloja.

Uuden laitteen kehittämisen päämotivaationa toimii edellä mainittujen erillisten komponenttien integrointi yhdelle piirilevylle ja laitteen käyttöliittymän parantaminen Bluetooth-yhteyden avulla. Työn yhteydessä on toteutettu myös PC-ohjelma ja Android-sovellus laitteen ohjaamiseen ja monitorointiin joko USB-väylän (Universal Serial Bus) tai Bluetoothin yli, mutta niiden tarkempi kuvaus ei kuulu tämän työn laajuuteen. Työn yhtenä tärkeimpänä tavoitteena on luoda tehokas ohjelmistoarkkitehtuuri, joka pystyy tarjoamaan tarjoamaan luotettavia anturimittauksia useille mittausyksikössä ajettaville käyttäjäalgoritmeille.

Työ on jaettu neljään osaan. Kappaleessa 2 kerrotaan mitä työssä ollaan tekemässä, ja verrataan uuden laitteen ominaisuuksia jo olemassa oleviin kaupallisiin



Kuva 1.1: Uudet mittausyksiköt ja vanha kehitysalusta vierekkäin

tuotteihin. Kappaleessa 3 kerrotaan laitteen komponenttien valintaperusteista ja piirilevysuunnittelusta. Kappale 4 esittelee laitteen ohjelmistoarkkitehtuurin ja sovellusrajapinnan laitteessa toimiville algoritmeille. Kappale 5 on kokeellinen, ja siinä esitetään mittausjärjestely laitteessa käytettyjen antureiden kalibroimiseksi. Lisäksi kappaleessa esitellään tuloksia ja käyttökokemuksia todellisista mittausympäristöistä.

2. LÄHTÖKOHDAT

Tämä työ pohjautuu ST Microelectronicsin STEVAL-MKI062V2 -inertiakehitysalustaan, joka sisältää kiihtyvyysanturin, gyroskoopin, magnetometrin, barometrin sekä lämpötila-anturin. Kehitysalustan ulkoisiin liitäntöihin kuuluu muistikorttiliitäntä, USB-liitäntä sekä laajennusliitäntä sarjaliikenteelle.

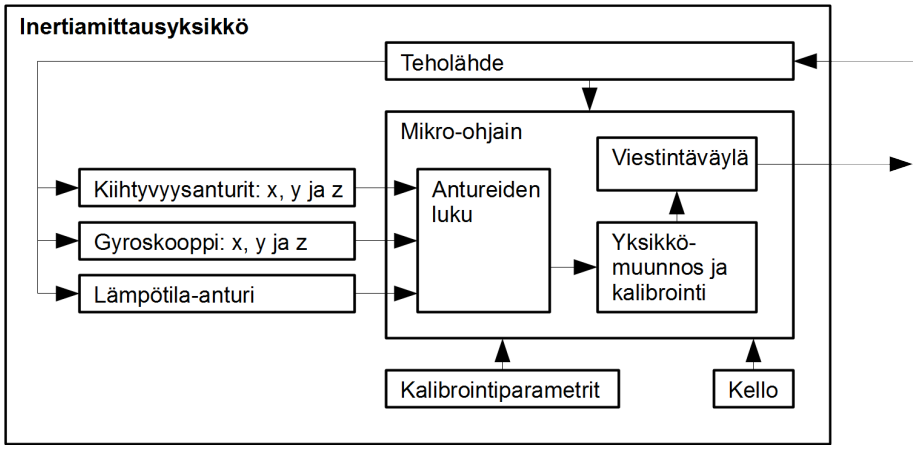
Kehiteltävän inertiamittausyksikön tavoitteena on sisältää STEVAL-MKI062V2 -alustan toiminnallisuus ja lisätä GPS-moduuli, Bluetooth-liitäntä sekä toiminta akun varassa. Seuraavassa listassa on esitetty uuden laitteen vaatimukset:

- Inertia-anturit: kiihtyvyysanturi ja gyroskooppi
- Muut anturit: magnetometri, barometri ja lämpötila-anturi
- Hyvä liitettävyys (muistikortti, Bluetooth, USB-liitäntä)
- GPS-paikannus ja aikasynkronointi
- Käyttäjäsovellusten suorittaminen laitteessa
- Pienikokoinen ja akkukäyttöisyys

Suunniteltavien mittausyksiköiden pääasiallinen käyttökohde tulee olemaan moottoripyörän ajotapahtumien havaitsemisessa. Mittaustilanteessa useita mittausyksiköitä on kiinnitetty moottoripyörään ja kuljettajaan, joista jokainen tutkii itsenäisesti ajotapahtumia. Tapahtumaliput siirretään langattomasti isäntäkoneelle, joka päättää millainen ajotapahtuma on kyseessä. Monen mittausyksikön samanaikainen toiminta asettaa haasteita ajastuksen ja laitteiston helppokäyttöisyydelle.

Yksiköiden ajastuksen tahdistaminen on saavutettavissa GPS-moduulilla, joka tukee aikapulssia. Tämän toiminnon avulla GPS-moduulit pystyvät tarjoamaan mikrosekuntiluokan tarkkuuksia [11]. Mittausyksikön helppokäyttöisyyden vuoksi on tärkeää, että GPS-moduuli integroidaan samalle piirilevylle muiden komponenttien kanssa. Lisäksi johdotuksien välttämiseksi GPS-moduulin tulee sisältää sisäinen antenni.

Mittausyksiköiden helppokäyttöisyyteen vaikuttaa niiden itsenäinen toiminta. Tämä tarkoittaa sitä, että johtojen vetämistä yksiköiden tai isäntäkoneen välillä tulee välttää. Langaton toiminta vaatii laitteelta akkukäyttöisyyttä sekä langatonta



Kuva 2.1: Rakennekuva inertiamittausyksiköstä

tiedonsiirtomahdollisuutta. Radiolinkin rajallisesta kaistasta johtuen kaikkea anturitietoa ei voi jatkuvasti siirtää isäntäkoneelle. Tämä asettaa vaatimuksia laitteen laskentateholle, sillä mittausyksikön pitää pystyä laskemaan anturitiedosta halutut suureet ja välittää vain nämä isäntäkoneelle. Suurikaistainen tieto voidaan tallettaa muistikortille myöhempää tarkastelua varten.

2.1 Inertiamittausyksikkö

Kuvassa 2.1 on esitetty inertiamittausyksikön rakenne. Se koostuu kolmen akselin kiihtyvyysanturista ja gyroskoopista, mikro-ohjaimesta, teholähteistä ja muistista, johon on tallennettu kalibrointivakioita. Mikro-ohjaimen tehtävänä on lukea antureita ja tehdä niille yksikkömuunnoksia ja kompensoida niiden tunnetut virheet kalibrointivakioiden avulla. Lisäksi mikro-ohjain tarjoaa prosessoidut anturilukemat eteenpäin viestitysväylän kautta.

Mittausyksikön sisältämät anturit ovat erilaisia valmistusteknisistä syistä. Antureiden antamiin lukemiin vaikuttavat muun muassa tehtaalta valmistunut erä, antureiden asettelu piirilevylle kalustusvaiheessa ja lämpötila. Lukemien korjaamiseksi inertiamittausyksikkö kalibroidaan virheiden varalta. Mittausyksiköt voidaan kalibroida jokainen erikseen tai asettaa kaikille samat kalibrointiparametrit. Yksittäinen kalibrointi on huomattavasti kalliimpaa.

Tässä työssä esiteltävä laite sisältää inertiamittausyksikön, mutta on kokonaisuudessaan laajempi sillä, siinä voidaan myös suorittaa käyttäjäsovelluksia, jotka hyödyntävät inertiamittaysyksikön tarjoamaa tietoa.

2.2 Vertailua kaupallisiin laitteisiin

Sovelluksessamme käytetyn mittausyksikön tärkein tehtävä on tarjota aikatahdistettuja mittauksia laitteen sisällä ajettaville käyttäjäsovelluksille. Lisäksi laitteen

(a) Hikob FOX (b) Xsens MTw (c) iSense AIST-350

Kuva 2.2: Kaupallisista inertiamittausyksiköitä

tulee toimia itsenäisesti ja pystyä keskustelemaan langattomasti isäntäkoneen kanssa. Ennen uuden mittausyksikön suunnittelun aloittamista tehty kaupallisten laitteiden kartoitus ei tuottanut täysin sopivia ehdokkaita käyttökohteeseemme. Tässä kappaleessa esitellään kartoituksen tuloksia.

Hikob FOX

Ranskalainen yritys Hikob on erikoistunut liikenteen ja rakennusten seurantaan pienivirtaisten langattomien anturiverkkojen avulla. Eräs yrityksen vahvuuksista on sen tarjoamien tuotteiden avoimuus OpenLab-yhteisön [3] kautta. OpenLab-yhteisö tarjoaa kehitystyökaluja, lähdekoodia ja esimerkkejä yrityksen tuotteiden ohjelmistokehitykseen.

Yritys valmistaa kuvan 2.2a Hikob FOX -tuotetta [2], joka sisältää lähes samat ominaisuudet kuin tässä työssä kehitetty mittausyksikkö. Erottava tekijä on käytetyn radiolinkin protokolla, joka perustuu IEEE 802.15.4 -standardiin. Tämä protokolla ei ole yhteensopiva älypuhelimien kanssa, joten se rajoittaa tuotteen käytettävyyttä sovelluksessamme. Rajoittavana tekijänä on myös GPS-moduulin puute, joka on tärkeä osa mittausten aikatahdistuksen kannalta. Tuotetiedustelun yhteydessä kävi ilmi, että yritys on valmistamassa GPS-tytärlevyä, mutta sen arvioitu valmistumisaika on vuoden 2013 toinen neljännes.

Xsens MTw

Hollantilainen yritys Xsens on erikoistunut kolmiulotteisen liikkeen seuraamiseen ja tarjoaa ratkaisuja teollisuuskoneiden liikkeen seuraamiseen sekä puettavia anturiyksiköitä ihmisen liikkeen tallentamiseen. Yrityksen tarjoamat tuotteet sisältävät laadukkaampia antureita verrattuna tässä työssä esitettyyn laitteeseen, mutta eivät tarjoa yhtä laajoja mahdollisuuksia omien reaaliaikaisen sovellusten kehittämiseen.

Yritys valmistaa kuvassa 2.2b esitettyjä langattomia MTw-mittausyksiköitä [4]

ihmisen liikkeen seurantaan. Mittauksia voidaan suorittaa samanaikaisesti usealla eri yksiköllä, ja ne kommunikoivat USB-väylään kiinnitettävän emoaseman kanssa. Emoaseman sisäinen protokolla tarjoaa aikatahdistuksen mittauksille. Käyttäjä pystyy tekemään tietokoneelle oman sovelluksen, joka lukee anturitietoa yksiköiltä yrityksen tarjoaman ohjelmistorajapinnan kautta. MTw soveltuu hyvin käyttökohteeseemme, mutta yksiköt maksavat useita tuhansia euroja kappaleelta. Täten niitä voidaan käyttää referenssimittauksissa käyttäjäsovelluksia kehitettäessä, mutta laajempaan käyttöön ne eivät sovellu.

iSense AIST-350

Venäläinen yritys iSense valmistamaa ja kalibroi laadukkaita MEMS-teknologiaan (Micro Electro Mechanical Systems) perustuvia inertiamittausyksiköitä. Yrityksen tuotteet eivät ole toiminnaltaan itsenäisiä ja vaativat aina isäntäkoneen vastaanottamaan ja käsittelemään tietoa.

Kuvassa 2.2c on esitetty yrityksen valmistama AIST-350 lämpötilaohjattu inertiamittausyksikkö. AIST-350:ä voidaan käyttää esimerkiksi ajoneuvoissa kiinteänä asennuksena, mutta sen suuren koon ja virrankulutuksen vuoksi sitä ei voida pukea ihmisen päälle. Käyttökohteessamme AIST-350 pystyy tarjoamaan tarkkoja referenssimittauksia ajoneuvon liiketilasta.

3. LAITTEISTO

3.1 Komponenttien valintaperusteet

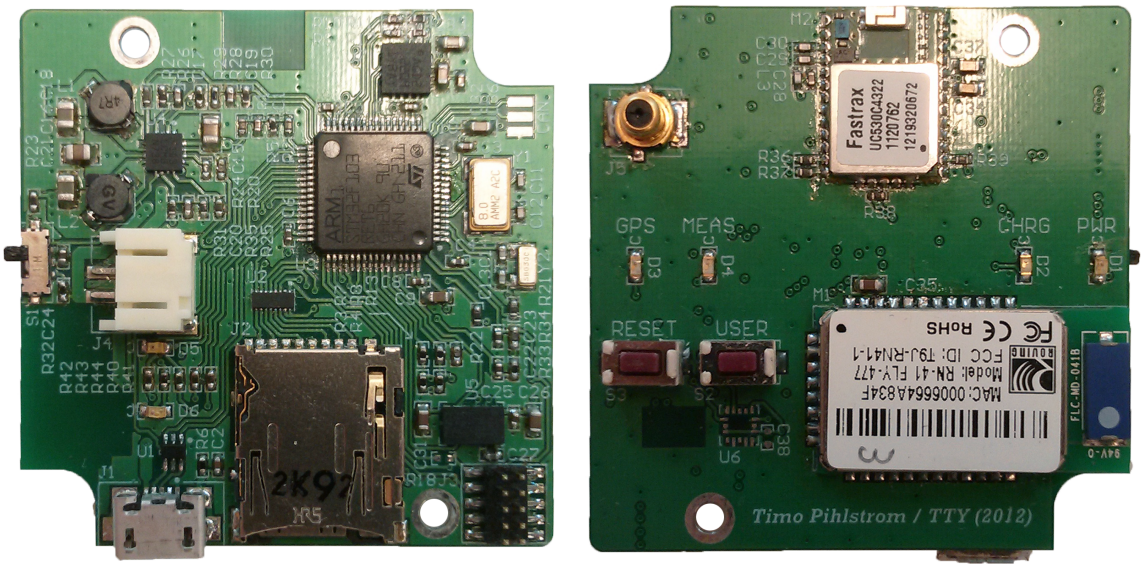
Mikro-ohjaimen tehtävänä on toteuttaa kuvassa 2.1 esitetyn inertiamittausyksikön toiminnot. Nämä toiminnot on mahdollista suorittaa 8-bittisellä mikro-ohjaimella, mutta samalla suorittimella tulee myös pystyä suorittamaan laskentaintensiivisiä käyttäjäsovelluksia. Tästä syystä valitun mikro-ohjaimen ytimen tulee olla 32-bittinen ja toimia riittävän korkealla kellotaajuudella. Toimiakseen osana muuta järjestelmää mikro-ohjain tarvitsee kaksi sarjaliikenneporttia ja vähintään yhden I^2C-väylän (Inter-Integrated Circuit). Laitteen kokorajoituksista johtuen, mikro-ohjaimen pakkauksen ulkomitat saavat olla korkeintaan 10 mm x 10 mm, joka asettaa rajoituksia ohjaimen kotelointityypille ja pinnien lukumäärälle.

Toteutettava laite on tehty projektiin, joka tutkii vähävirtaisia antureita kuluttajasovelluksissa – soveltuvia antureita voi löytää esimerkiksi peliohjaimista tai matkapuhelimista. Täten suurin rajoite anturien valinnalle on niiden hinta ja pakkauksen koko. Antureiden tulee myös toimia dynaamisissa tilanteissa, esimerkiksi törmäyksissä, joten lukutaajuuksien ja käyttöalueiden tulee olla riittävän suuret.

Jotta suunniteltava mittausyksikkö pystyy toimimaan useissa erilaisissa käyttötilanteissa, tulee sen sisältää erilaisia liitäntöjä ulkomaailmaan. Valittuihin liitäntöihin kuuluvat USB- ja Bluetooth-liitännät sekä mahdollisuus käyttää muistikorttia. USB-liitäntä on tarkoitettu jatkuvaan toimintaa, sillä liitäntä tarjoaa tiedonsiirtoväylän ja käyttöjännitteen. Muistikortin ja Bluetooth-liitännän avulla laite toimii langattomasti. Bluetooth-liitäntä tarjoaa pienikaistaisen tiedonsiirtoväylän laitteelle ja muistikortin avulla voidaan tallentaa suurikaistaista tietoa myöhempää tarkastelua varten.

Useiden mittausyksiköiden on toimittava samanaikaisesti aikatahdistetusti. Tämä voidaan toteuttaa lisäämällä laitteeseen GPS-moduuli, joka tukee aikapulssitoimintoa. Tämän lisäksi moduulin vaatimuksina tulee olla pieni koko, sekä sisäinen antenni hyvällä herkkyydellä. Moduulin toimintoja tulee myös pystyä testaamaan sisätiloissa, jolloin moduulin on tuettava ulkoista aktiivista antennia.

Eräs laitteen suunnitteluvaatimus on akkutoiminta, joka lisää monimutkaisuutta tehonhallintaan akkulaturin muodossa. Koska laite tarvitsee myös hyvällä hyötysuhteella toimivan jännitevakavointipiirin, kasvaa komponenttien määrä nopeasti. Tärkein valintaperiaate tehonhallintaan on sen vaatima piirilevypinta-ala ja riittävä



(a) Yläpuoli (b) Alapuoli

Kuva 3.1: Toteutetun mittausyksikön piirilevy

virranantokyky.

Viimeinen kappaleessa 2 esitetty suunnitteluvaatimus on laitteen pieni koko ja paino. Tämä pystytään saavuttamaan vain mikäli kotelon, piirilevyn ja komponenttien ulkomitat otetaan huomioon suunnittelun aikaisessa vaiheessa. Lisäksi laitteelle tulee määrittää toivottu käyttöaika akun varassa, koska liian suurikapasiteettisen akun valinta kasvattaa kotelon ulkomittoja. Kotelosta pitää myös löytyä kiinnityspisteitä.

3.2 Komponentit

3.2.1 Mikro-ohjain

Laitteessa käytetään ST Microelectronicsin valmistamaa STM32F103RET6 -mikro-ohjainta, joka perustuu 32-bittiseen ARM Cortex M3 -ytimeen. Mikro-ohjain sisältää 512 kilotavua ohjelmamuistia ja 64 kilotavua datamuistia. Ulkoisina liitäntöinä ohjaimessa on USB-liitäntä, kaksi sarjaliikenneväylää, kaksi I^2C- ja SPI-väylää (Serial Peripheral Interface) sekä SDIO-väylä (Secure Digital Input Output) muistikortille. Ohjelmointi- ja debuggausliitäntänä mikro-ohjain käyttää JTAG-liitäntää (Joint Test Action Group).

Ohjain käyttää kahta ulkoista kideoskillaatoria. Ohjaimen USB-lohko käyttää sisäistä 48 MHz kellotaajuutta, joten pääkiteen taajuuden on oltava tämän monikerta. Täten kiteen kellotaajuudeksi on valittu 8 MHz ja taajuusvakaudeksi 20 ppm. Toinen kide on pienitaajuuksinen kellokide, jonka avulla prosessori ylläpitää aikaa tai voi toimia pienivirtaisessa lepotilassa.

3.2.2 Anturit

Kiihtyvyysanturi ja magnetometri

Yhdistettynä kolmiakselisena kiihtyvyysanturina ja magnetometrina laitteessa toimii ST Microelectronicsin valmistama LSM303DLHC [6], joka kytkeytyy mikro-ohjaimen I^2C-väylään. Käyttäjä voi valita kiihtyvyysanturille toiminta-alueeksi $\pm$2g - $\pm$16g ja magnetometrille $\pm$1,3 - $\pm$8,1 Ga. Komponentin sijoitus piirilevyllä on tehty magnetometrin ehdoilla, ja täten se on sijoitettu etäälle tehonhallintapiireistä ja sen häiriötä aiheuttavista suodatuskeloista. Tämän lisäksi kaikkien kerrosten kuparitasot ja reititykset on poistettu suoraan komponentin alapuolelta, jotta niissä kulkevien virtojen aiheuttamat magneettikentät eivät häiritse mittauksia. Anturipiiri on pyritty myös sijoittamaan lähelle gyroskooppia, jotta näiden kahden komponentin tuottamat mittaukset kuvastaisivat samaan pisteeseen liittyviä suureita. Komponentti on sijoitettu kuvan 3.1a oikeaan alanurkkaan.

Gyroskooppi

Gyroskooppina laitteessa on ST Microelectronicsin valmistama L3GD20, joka voidaan liittää joko I^2C- tai SPI-väylään. Käyttäjä voi valita piirin toiminta-alueeksi $\pm$ 250 - $\pm$ 2000 °/s ja voi lukea anturia enintään 760 Hz taajuudella. Piiri on valittu, koska sen suorituskyky ja hinta ovat lähellä matkapuhelimissa esiintyviä gyroskooppeja.

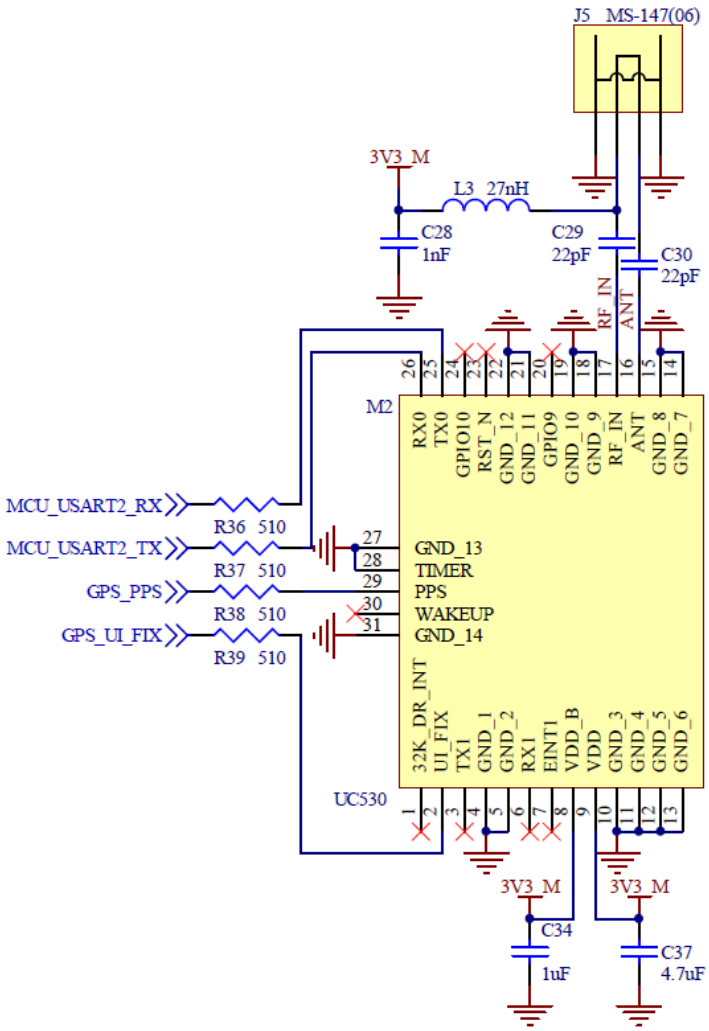
Paineanturi

Piirilevylle on varattu paikka ST Microelectronicsin valmistamalle LPS331AP -paineanturille, mutta sitä ei ole kalustettu laitteen kehitysversioissa. Komponentti sijaitsee kuvan 3.1b vasemmassa alanurkassa. Piirin mittausalue on 260-1260 mbar ja näytteistystaajuus on käyttäjän valittavissa 1-25 Hz. Paineanturia voidaan käyttää apuna korkeuden mittauksessa täydentämässä GPS-moduulilta saatua korkeustietoa.

3.2.3 Liitynnät ulkomaailmaan

Muistikortti

Muistikorttiliittimeksi on valittu Hirose Electricin DM3AT-SF-PEJM5 -liitin [9], joka toimii microSD-korttien kanssa. Muistikortti kytkeytyy mikro-ohjaimen SDIO-väylään, joka tarjoaa 4-bittisen tiedonsiirtoväylän 25 MHz nopeudella. Muistikorttiliittimen ja mikro-ohjaimen väliin on lisätty EMIF06-MSD02N16 -häiriönsuodatuspiiri [10], joka suojaa myös sähköstaattisia purkauksia vastaan. Suodatuspiiri on valittu



Kuva 3.2: GPS-moduulin kytkentäkaavio

sen pienen koon takia, sillä se on sijoitettu ahtaaseen paikkaan mikro-ohjaimen ja muistikorttiliittimen väliin.

GPS-liityntä

U-bloxin valmistama GPS-moduuli UC530 [11] mahdollistaa sisäisen ja ulkoisen antennin käytön. Moduuli on kytketty mikro-ohjaimen sarjaliikenneväylään ja lähettää aikapulssisignaalia ohjaimen IO-pinniin. Kytkentä mikro-ohjaimeen on esitetty kuvassa 3.2. Kytkennän GPS_PPS-signaali tuottaa 100 ms kanttiaaltopulsseja sekunnin välein, joiden luvattu tarkkuus on 1 μs sisällä. Pulssien tuottaminen alkaa muutama sekuntti ensimmäisen paikannuksen jälkeen ja tätä voidaan käyttää apuna useiden mittausyksiköiden aikatahdistamisessa. Signaali GPS_UI_FIX tuottaa kahden sekunnin välein 200 ms pulsseja, mikäli GPS-moduuli on saanut paikkaratkaisun. Muutoin tämä signaali pysyy alhaalla. Tämän signaalin avulla mikro-ohjaimen ohjelmisto tietää missä tilassa GPS-moduuli on.

Moduuli on ulkomitoiltaan 14 x 9,6 mm, mutta laitteen sisäinen antenni vaatii ympärilleen vähintään 45 x 20 mm katkeamattoman kuparialueen, joka kasvattaa huomattavasti moduulin käyttämää piirilevyalaa. Tämä asettaa vaatimuksia piirilevysuunnittelulle, ja kuvasta 3.1b nähdään että koko piirilevyn yläosa on varattu antennin maatasolle. Kuvassa näkyy myös ulkoisen aktiivisen antennin liitäntä vasemmassa yläkulmassa. Liitin toimii kytkimenä, joka liitoksen huomatessaan katkaisee sisäisen antennin signaalin ja ohjaa ulkoisen antennin signaalin moduulille. Liittimen kytkentä on esitetty kuvassa 3.2, jossa kela L3 sallii tasajännitteen siirtämisen aktiiviselle antennille, mutta estää radiotaajuisten signaalien pääsyn käyttöjännitelinjaan. C29 ja C30 ovat erotuskondensaattoreita, jotka päästävät vain radiotaajuiset signaalit moduulin antennisisäänmenoon.

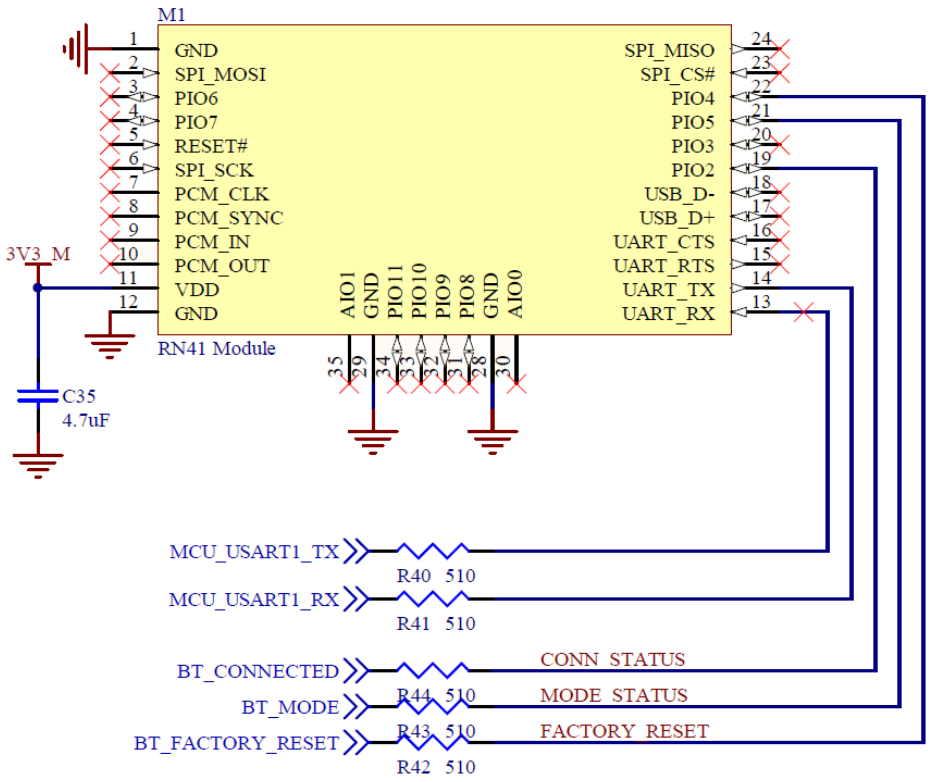
USB-liityntä

USB-liitännän tärkein tehtävä on tarjota tehoa akunlatauspiirille. Sen toinen tehtävä on toimia suurikaistaisena tiedonsiirtoväylänä, jota voidaan käyttää esimerkiksi mittausten tai tulosteiden siirtämiseen tietokoneelle. Liitintyypiksi on valittu micro-B, joka tarjoaa luotettavan liitännän pienessä koossa.

USB-väylä on differentiaalinen väylä, joka tarvitsee terminointivastukset sekä isännän että laitteen päähän. Tämän lisäksi laitteen tulee kytkeä toinen datalinjoista ylösvetovastuksella käyttöjännitteeseen, joka määrittää laitteen toimintatilan. Mittausyksikössä terminointi, ylösvetovastus ja ESD-suojaus (Electrostatic Discharge) on toteutettu ST Microelectronicsin USBUF02W6-piirillä [12].

Bluetooth-liityntä

Mittausyksikön Bluetooth-moduuliksi on valittu Microchipin valmistama RN41-moduuli [13], koska se sisältää oman Bluetooth-pinonsa ja valmiin SPP-profiilin (Serial Port Profile). Moduuli tukee Bluetooth 2.1 -standardia ja toimii luokassa 1. Moduulia käytetään sarjaporttiliitännän kautta ja kuvassa 3.3 on esitetty kytkentä mikro-ohjaimelle. Moduuli vetää signaalin BT_CONNECTED ylös silloin, kun moduuli on paritettuna toisen laitteen kanssa, tämä signaali on kytketty piirilevyllä olevaan lediin ja mikro-ohjaimelle. Signaali BT_MODE on kytketty pelkästään piirilevyllä olevaan lediin, ja vilkkuu 1 Hz taajuudella silloin, kun moduuliin voidaan ottaa yhteys; ja vilkkuu kymmenen hertsin taajuudella silloin, kun moduuli on mikro-ohjaimen ohjaamassa komentotilassa. BT_FACTORY_RESET signaali palauttaa moduulin tehdasasetustilaan mikäli siihen syöttää kolme pulssia. Tämä signaali on kytkettynä mikro-ohjaimelle ja sen kanssa moduuli voidaan palauttaa havaitusta virhetilasta.



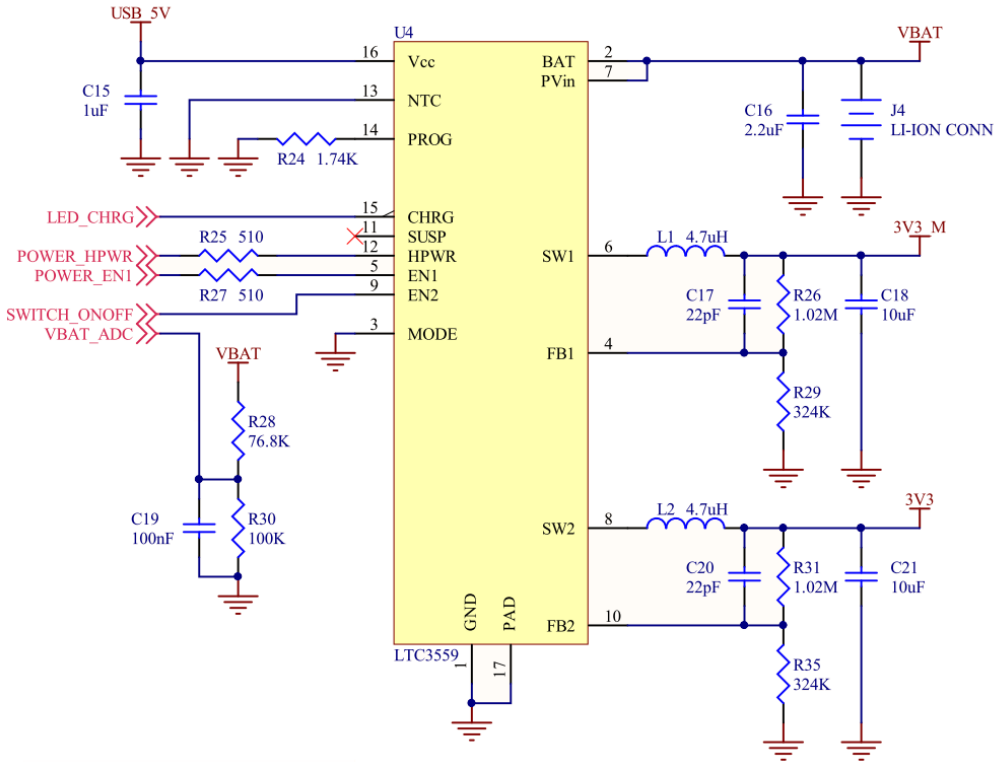
Kuva 3.3: Bluetooth-moduulin kytkentäkaavio

3.2.4 Tehonhallinta

Kuvassa 3.4 on esitetty mittausyksikön tehonhallintapiiri. Se perustuu Linear Technologyn LTC3559-piiriin [14], joka sisältää lineaarisen akkulaturin ja kaksi laskevaa hakkuriteholähdettä. LTC3559 saa käyttötehonsa USB-isännältä, joka tarjoaa asiakaslaitteille normaalitilassa 100 mA virtaa ja pyydettäessä 500 mA. Piirin ottama sisäänmenovirta määritetään vastuksella R24, joka kyseisellä vastusarvolla on asetettu 500 mA:iin. Mikro-ohjain voi ohjata piirin HPWR-pinniä, joka määrittää käytetäänkö asetetusta sisäänmenovirrasta 20 % vai 100 %.

Toinen laitteen hakkuriteholähteistä antaa virtaa mikro-ohjaimelle, antureille sekä muistikortille. Tämä hakkuri voidaan kytkeä päälle laitteen virtakytkimellä, joka nostaa EN2-pinnin tilan korkeaksi. Toinen hakkureista antaa virtaa GPS- ja Bluetooth-moduulille. Vastaavasti tämä hakkuri voidaan kytkeä päälle EN1-pinnillä, jota ohjataan mikro-ohjaimella. Molemmat hakkureista on asetettu antamaan 3,3 voltin ulostulojännite ja molempien maksimivirranantokyky on 400 mA.

Laitteen akuksi on valittu suojapiirillinen 400 mAh litiumpolymeeriakku [15], joka on kompromissi akun ulkomittojen ja mittausyksikön käyttöajan suhteen. Arvioitu laitteen virrankulutus aktiivisessa tilassa on noin 100-150 mA, joten akku tarjoaa käyttöajaksi 3-4 tuntia. Akku kytketään piirilevylle JST-liittimellä [16]. Lisäk-



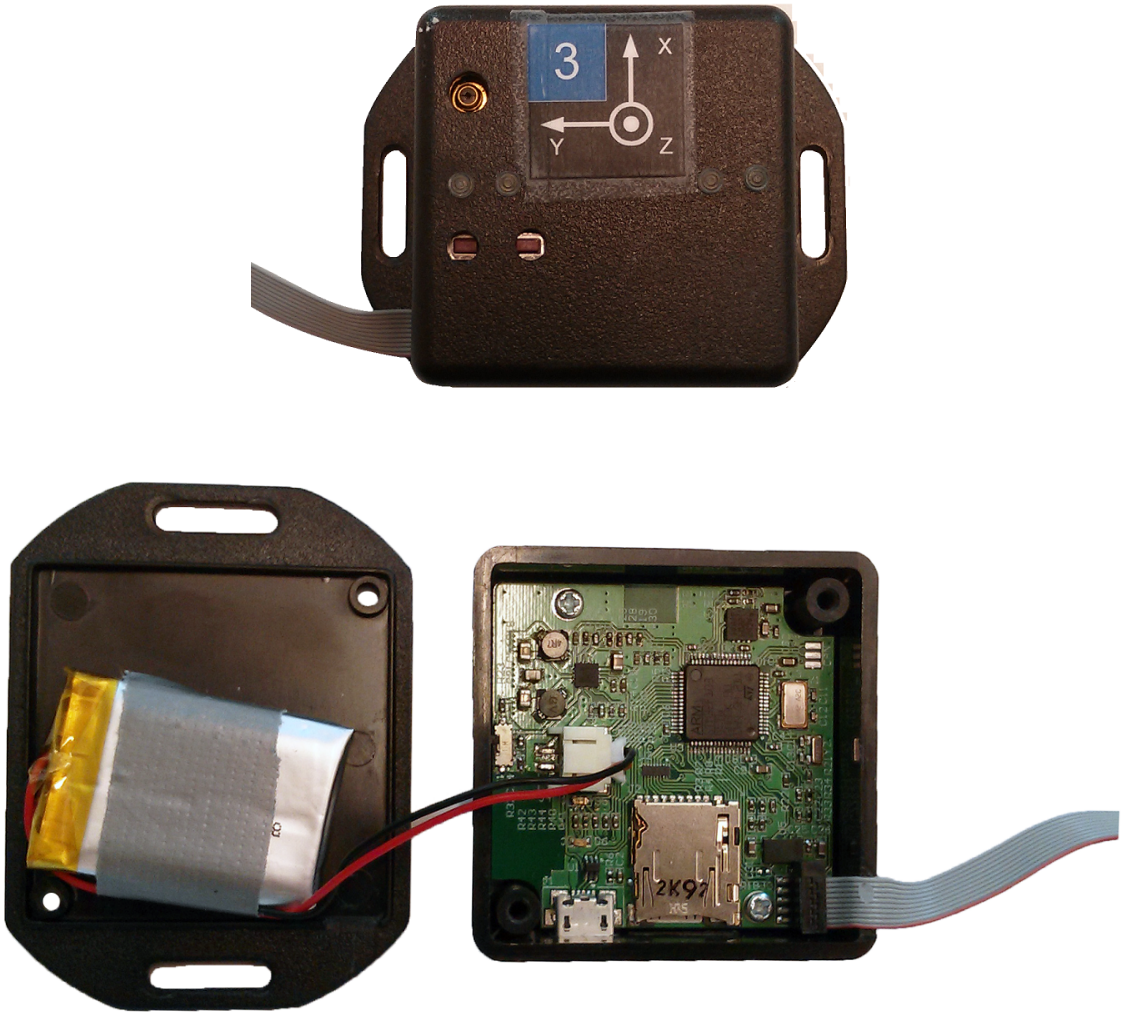
Kuva 3.4: Tehonhallintapiirin kytkentäkaavio

si tehonhallinta tarjoaa mikro-ohjaimelle jännitejaeon akkujännitteestä, joka A/D-muuntimen (analogia-digitaalimuunnin) avulla voidaan lukea ohjelmallisesti – tämän avulla laitteen toiminnot voidaan keskeyttää ennen akun tyhjenemistä.

3.2.5 Mekaaninen suunnittelu

Mittausyksikön kotelona toimii Hammond Mfg:n 1551FL-kotelo [17], joka on ulkomitoiltaan 50 x 50 x 20 mm. Piirilevyn ulkomitat ovat 44 x 44 mm ja sen muoto on suunniteltu valittuun koteloon käyttäen apuna 3D-mallinnusohjelmistoa. Kuva 3.5 esittää kotelon, piirilevyn ja litiumpolymeeriakun kiinnitettynä toisiinsa.

Kuva 3.5 esittää mittausyksikön koteloinnin ylhäältä katsottuna. Kuvassa näkyy ulkoinen liittin aktiiviselle GPS-antennille, neljä lediä ja kaksi käyttäjäpainiketta. Ledit ilmaisevat vasemmalta oikealle lukien GPS-moduulin tilan (oranssi), käynnissä olevan mittauksen (keltainen), käynnissä olevan akun latauksen (punainen) ja mikäli laite on päällä (vihreä). Ledien valo tuodaan kotelon pintaan lyhyillä valoputkilla, jotta aktiivisen ledin näkee paremmin suurestakin kulmasta. Kaksi käyttäjäpainiketta ovat noin millimetrin kotelon ulkoreunan sisällä, jotta niitä ei vahingossa paina mittausten ollessa käynnissä. Painikkeiden tehtävät vasemmalta oikealle lukien ovat laitteen uudelleenkäynnistys ja mittausten kytkeminen päälle tai pois. Jokaiseen mittausyksikköön on lisäksi lisätty tarra, joka kertoo mittausyksikön yksilöllisen



Kuva 3.5: Mittausyksikön kotelointi

numeron ja mitattavien akseleiden suunnan.

4. OHJELMISTO

4.1 Lohkorakenne

Kuvassa 4.1 on esitetty mittausyksikön ohjelmiston lohkorakenne. Ohjelmisto perustuu FreeRTOS-käyttöjärjestelmään ja sen tarjoamiin säie- ja jonopalveluihin. Jokainen kuvassa esiintyvistä moduuleista sisältää yhden FreeRTOS-työsäikeen ja mahdollisen viestintäsäikeen. Viestintäsäikeiden avulla moduulit liittyvät kuvassa esitettyyn sisäiseen viestintäväylään kuuntelemalla niille asetettuja viestijonoja.

Muut ohjelmiston ulkoiset ohjelmistokomponentit ovat ST Microelectronicsin tarjoamia ajureita mikro-ohjaimen sisäisille lohkoille, kuten USB:n virtuaalisarjaporttiajuri, I^2C-väylän ajuri, muistikorttiväylän ajuri ja siihen liittyvä FAT-tiedostojärjestelmäkirjasto (File Allocation Table) sekä ajurit kiihtyvyysanturille, magnetometrille sekä gyroskoopille. ST Microelectronicsin tarjoamien ajureiden lisäksi ohjelmistossa on käytetty NMEA-kirjastoa (National Marine Electronics Association) GPS-moduulilta tulevien viestien tulkitsemiseksi sekä OpenPilot-projektin Kalmansuodinta [22] laitteen suorituskyvyn testaamiseksi.

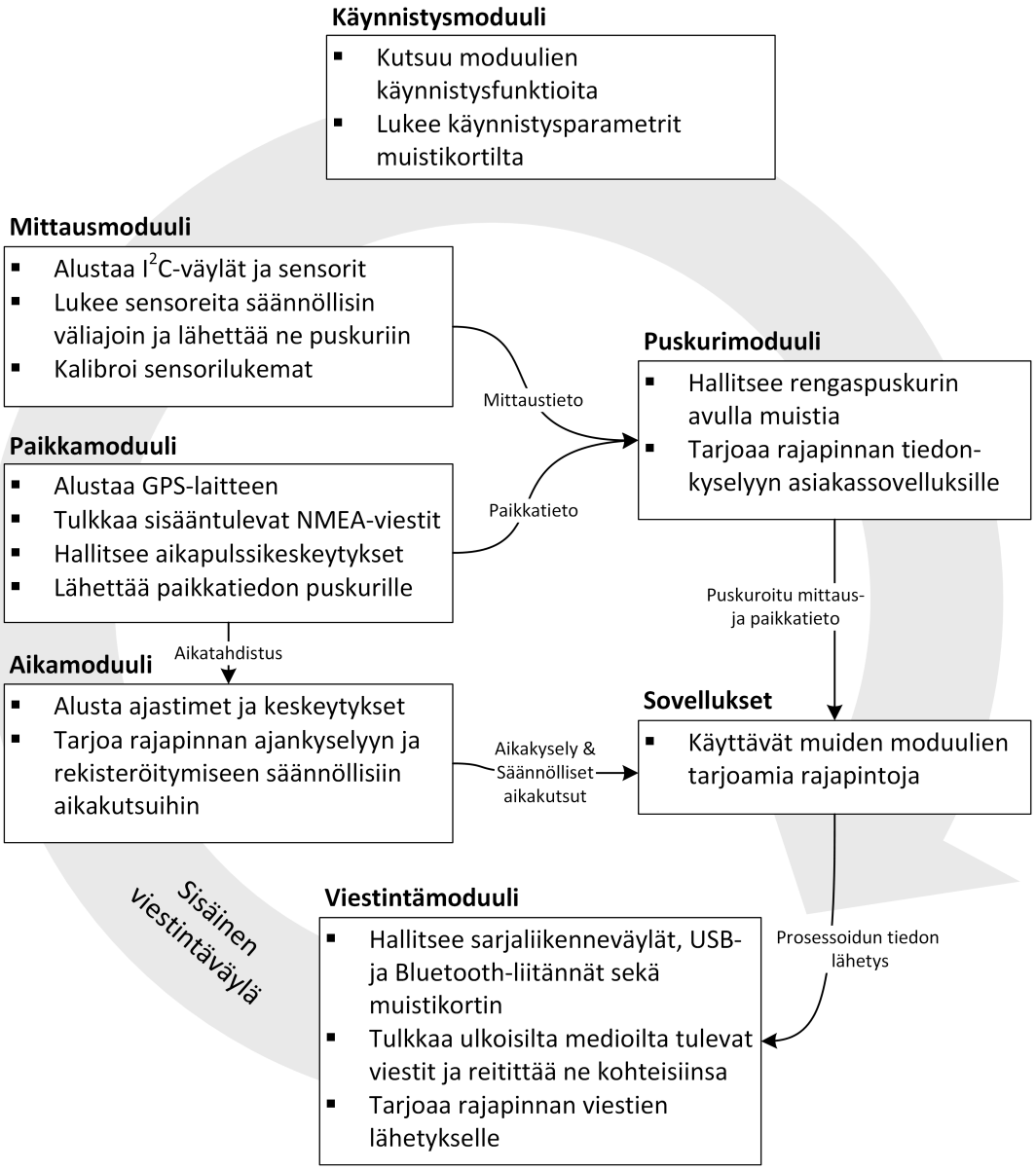
4.2 Toiminnallinen kuvaus

Laitteen käynnistyessä main-funktio asettaa mikro-ohjaimen kellot ja muistit käyttökuntoon, jonka jälkeen kutsuu se käynnistysmoduulin alustusfunktiota, jonka tehtävänä on käynnistää muut moduulit. Tämän lisäksi käynnistysmoduuli luo oman säikeensä, jonka tehtävänä on lukea käynnistyksen aikana muistikortilta asetustiedosto ja lähettää sen sisältämät parametrit muille moduuleille sisäisen viestintäväylän kautta. Säie toimii pienellä prioriteetilla, joten se on myös sopiva paikka debug-tulosteille.

4.2.1 Antureiden lukeminen

Mittausmoduulin tehtäviä ovat antureiden alustaminen laitteen käynnistyessä, anturien lukeminen tasaisin väliajoin sekä mittausten kalibrointien suorittaminen. Moduuli käyttää useita ST Microelectronicsin tarjoamia valmiita ohjelmistokomponentteja, kuten anturiajureita ja I^2C-kirjastoa.

Käynnistysmoduuli koostuu kahdesta säikeestä, joista toinen on viestintäsäie ja



Kuva 4.1: Ohjelmiston lohkorakenne

toinen työsäie. Viestintäsäikeen tehtävänä on kuunnella mittausmoduulin omaa viestijonoa, ja tulkita sekä vastata siihen saapuneita viestejä. Mittausmoduuli tukee viestejä antureiden lukunopeuden asettamiselle ja kyselylle sekä mittausten asettamiseksi päälle ja pois. Kaikki mittausmoduulin sisäisten toimintaparametrien asetus tapahtuu viestijonon kautta, jonka avulla on päästy eroon jaettujen resurssien ongelmasta, sillä vain yhtä viestiä käsitellään kerrallaan.

Työsäikeen tehtävänä on käynnistyessään alustaa I^2C-väylät käyttökuntoon ja sen jälkeen lähettää antureille alkuparametrit, jotka sisältävät muun muassa päivitysnopeuden ja mittausalueen laajuuden. Mittausten ajastamiselle on kaksi mahdollisuutta: FreeRTOS-käyttöjärjestelmän ajastuspalvelut tai mikro-ohjaimen sisältä-

män ajastinkeskeytyksen ajastama työsäikeen ajokerta. Ohjelmistossa on päädytty jälkimmäiseen ratkaisuun, sillä käyttöjärjestelmäpalveluita käyttämällä mittauskertoja saattaa hukkua tai viivästyä, mikäli muita korkeampiprioriteettisia säikeitä on ajossa. Ajastus toimii siten, että työsäie kuuntelee jonoa, johon ajastinkeskeytys lisää yhden työtilauksen, joka määrittää mitä ajureita työsäikeen tulee lukea. Tämän jälkeen työsäie kysyy ajureilta päivitetyt anturilukemat ja siirtää ne puskurimoduulille. Ajurien luku tapahtuu käyttämällä I^2C-väylän keskeytyksiä, jolloin työsäie keskeyttää toimintansa kunnes anturilukema on päivitetty. Tämän viiveen aikana muut säikeet voivat siirtyä suoritukseen.

```
1  typedef struct measurement {
2      union {
3          uint8_t i_payload[6];
4          float   f_payload[3];
5      };
6      uint32_t timestamp;
7      uint16_t source;
8      uint16_t pad1;
9  } measurement_t;
```

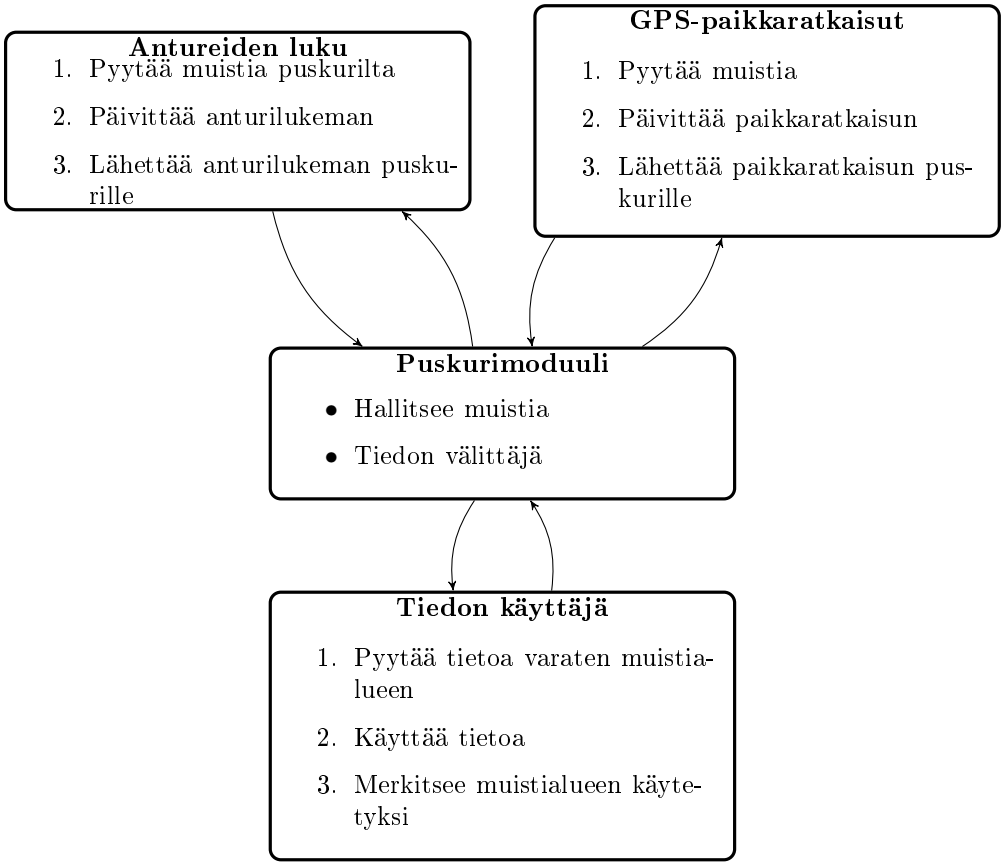
Listaus 4.1: Mittaustietorakenne

Listauksessa 4.1 on esitetty mittausmoduulin käyttämä sisäinen tietorakenne. Tietorakenne sisältää mittauksen aikaleiman, tietolähteen ja varsinaisen tiedon. Kenttä pad1 on lisätty varmistamaan että mittausrakenteen koko on jaollinen neljällä tavulla, jolloin tietorakenteen siirtäminen eri moduuleille tai viestintämedialle on luotettavampaa. Varsinainen mittaustieto voi olla antureilta tullessaan kokonaislukumuodossa, ja myöhemmin kalibroinnin jälkeen oikeassa yksikössään liukulukumuodossa. Tämä on toteutettu union-määritteellä, jolloin samaa muistialuetta voidaan käsitellä kahdella erilaisella tiedonesittämistavalla.

4.2.2 Anturitiedon siirto ja puskurointi

Puskurimoduulin tehtävä on ottaa vastaan tietoa mittaus- ja paikkamoduulilta sekä monistaa tämä data puskurimoduulin rekisteröityneille asiakassovelluksille. Puskurimoduulia tarvitaan, koska tietolähteitä ja asiakassovelluksia voi olla useita, jolloin muistinhallinta ja -jakaminen on ongelmallista.

Puskurimoduuli sisältää yhden työsäikeen eikä lainkaan viestintäsäiettä, joka on korvattu perinteisellä funktiorajapinnalla. Tämän rajapinnan avulla asiakassovellukset voivat rekisteröityä mittaus- ja paikkatietoon. Työsäie odottaa sisääntulevaa tietoa mittaus- tai paikkamoduuleilta, jonka siirto on toteutettu FreeRTOS-jonon



Kuva 4.2: Periaatekuva tiedonsiirrosta moduulien välillä

avulla ja sisältää osoittimia listauksen 4.2 mukaisiin tietorakenteisiin. Uuden tiedon saapuessa työsäie siirtää sen rekisteröityneille asiakkaille. Mikäli rekisteröityneitä asiakkaita on useita, tieto monistetaan, jotta jokainen asiakas saa käyttöönsä oman muistialueen. Kaiken siirretyn tiedon muistialue on esivarattu puskurimoduulin käynnistyessä, jonka avulla vältetään turhaa muistinkopiointia, eikä tehdä lainkaan kutsuja hitaaseen muistinvarausoperaatioon. Puskurimoduulin sisäinen toiminta perustuu rengaspuskuriin, josta muistia tarvitsevat mittaus- ja paikkamoduulit pyytävät sitä. Kun muistialue on täytetty uudella tiedolla, siirtyy se puskurimoduulin läpi mahdollisesti usealle asiakassovellukselle. Kun asiakas on lopulta käyttänyt tiedon, merkitsee se muistialueen tyhjäksi kutsumalla muistinvapautusoperaatiota puskurimoduulin funktiorajapinnasta. Tämä edellä kuvattu toiminta esitetään kuvassa 4.2.

Listauksessa 4.2 on esitetty puskurimoduulin käyttämä tietorakenne, jonka avulla voidaan välittää muistia mittausmoduuleilta puskurimoduuleille ja sieltä eteenpäin asiakassovelluksille. Tietorakenne sisältää lähdekentän, jonka avulla voidaan määrittää mistä mittaustieto on lähtöisin, sekä täytetavun varmistamaan että tietorakenteen koko on jaollinen neljällä tavulla. Koska puskurimoduuli tukee tietoraken-

teita sekä mittaus- että paikkamoduulilta, sisältää tietorakenne osoittimen tietoon union-määreen avulla. Tämän avulla samaa muistialuetta voidaan tulkita kahdella eri tavalla riippuen alkuperäisestä tietolähteestä.

```
1  typedef struct buff_item {
2      uint16_t source;
3      uint16_t pad;
4      union {
5          measurement_t*    meas;
6          GPSPositionData* gps;
7      };
8  } buff_item_t;
```

Listaus 4.2: Puskuritietorakenne

4.2.3 Aika- ja paikkapalvelut

Aikamoduuli sisältää palvelut ajan mittaamiseen ja säännöllisiin aikakutsuihin, joista molemmat palvelut perustuvat omiin mikro-ohjaimen sisäisiin ajastimiin. Aikamoduulin funktiorajapinta sisältää funktiot ajan kyselyyn ja päivittämiseen, sekä rekisteröitymisfunktiot aikakutsupalveluun. Aikakutsupalvelu kutsuu käyttäjän antamaa takaisinkutsufunktiota säännöllisin väliajoin niin kauan kunnes käyttäjä poistaa rekisteröintinsä. Käytetty aikaresoluutio on 1 ms ja aikaformaatti on GPS-aika, joka kuvastaa kuluneita millisekunteja viikon ensimmäisestä maanantaista lähtien. Aikamoduuli ja paikkamoduuli ovat läheisessä yhteistyössä, sillä paikkamoduuli päivittää sekunnin välein aikamoduulin aikaa kun GPS-moduuli on saanut paikannuksen.

Paikkamoduulin tehtävänä on alustaa GPS-moduuli käyttötilaan ja tulkita sieltä tulevat NMEA-muotoiset paikkaratkaisuviestit. Paikkamoduuli rakentuu yhdestä yhdistetystä työ- ja viestintäsäikeestä. Kaikki GPS-moduulilta saapuva NMEA-muotoinen tieto kulkee viestintämoduulin läpi, ja tulkitaan vasta paikkamoduulin säikeessä listauksessa 4.3 esitetyksi tietorakenteeksi. Tämän jälkeen tietorakenteen muistialue lähetetään puskurimoduulille, joka jakaa tietorakenteen rekisteröityneille asiakassovelluksille. Paikkamoduulin käyttämä tietorakenne sisältää paikka- ja nopeustiedon sekä päivämäärä- ja aikatiedon. Paikkamoduuli tukee komentoviestejä, joilla voidaan käynnistää tai sammuttaa GPS-moduuli, sekä asettaa sen lähettämien paikkaratkaisuviestien taajuus välillä 1-10 Hz.

```
1  typedef struct gps_data {
2      uint8_t fix_valid;  /* flag */
3      uint8_t date_valid; /* flag */
```

```
4       int32_t Latitude;    /* degrees * 1e7 */
5       int32_t Longitude;   /* degrees * 1e7 */
6       float Altitude;      /* m */
7       float velN;          /* m/s */
8       float velE;          /* m/s */
9       float velD;          /* m/s */
10      float Groundspeed;   /* m/s */
11      uint32_t GeoidSeparation;
12      int32_t Heading;     /* deg */
13      uint16_t year;
14      uint16_t month;
15      uint16_t day;
16      uint16_t updated;    /* flag */
17  } GPSPositionData;
```

Listaus 4.3: Paikkatietorakenne

4.2.4 Viestitysjärjestelmä

Mittausyksikkö voi keskustella kahden ulkoisen viestintäväylän kautta, jotka ovat USB- ja Bluetooth-liitäntä. USB-liitäntä toimii virtuaalisena sarjaväylänä, joka siirtää käytettyä viestipohjaista protokollaa tietokoneen ja mittausyksikön välillä. Sama viestipohjainen protokolla on käytössä myös Bluetooth-liitännän kanssa. Yhteistä molemmille viestintämedioille on käytetty yhteydetön viestintäprotokolla, joka on esitetty kappaleessa 4.2.4. Ulkoisten viestintäväylien lisäksi, samaa viestintäjärjestelmää käytetään prosessorin sisäisten säikeiden välisenä viestintäväylänä.

Viestintämoduulin tehtävänä on alustaa ja ohjata kaikkia käytettyjä ulkoisia viestintämedioita, joihin kuuluu USB-liitäntä, GPS- ja Bluetooth-laite ja muistikortti. Näistä GPS- ja Bluetooth-laitteet on kytketty kahteen erilliseen mikro-ohjaimen sarjaporttiväylään. Lisäksi viestintämoduuli osaa tulkita ulkoisista viestintäväylistä tulevaa sarjamuotoista tietoa protokollan mukaisiksi viesteiksi ja ohjata ne viestien reititystaulukon avulla oikeaan osoitteeseen. Tämän lisäksi viestintämoduuli sisältää funktiorajapinnan, joka helpottaa viestien lähettämistä. Viestintämoduuli on jaettu useaan säikeeseen, joista jokainen vastaa yhden ulkoisesta viestintäväylästä. Tällä toiminnan hajauttamisella pyritään estämään käyttöjärjestelmän lukkiintumisia, mikäli siirrettävät tietomäärät ovat suuria.

Viestintämoduulilla on lisäksi on tärkeä tehtävä yksilöimään mittausyksiköt, sillä jokainen Bluetooth-moduuli sisältää tehtaalta lähtiessään yksilöllisen MAC-osoitteen (Media Access Control). Viestintämoduuli lukee tämän osoitteen ja antaa laitteelle tunnistenumeron, jotka on esitetty taulukossa 4.1. Kirjoittamisen hetkellä laitteita on tehty seitsemän kappaletta.

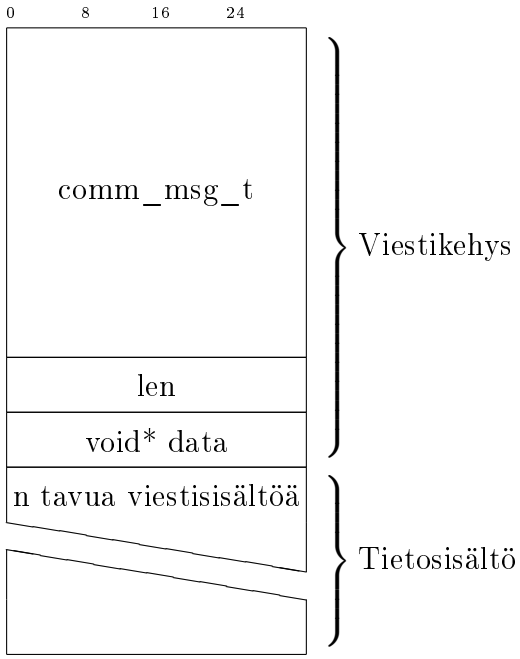
Taulukko 4.1: Mittausyksiköiden tunnistenumerot

Bluetooth MAC-osoite	Tunnistenumero
00:06:66:4A:82:43	1
00:06:66:4A:83:4E	2
00:06:66:4A:83:4F	3
00:06:66:4F:F8:FB	4
00:06:66:4F:F8:F6	5
00:06:66:60:01:49	6
00:06:66:4F:F8:F7	7

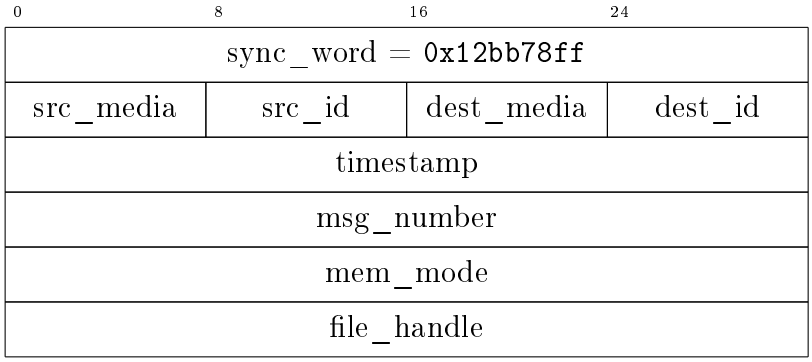
Viestien kehysrakenne

Jokainen viestitysjärjestelmän viesti sisältää alkukehyksen ja sitä seuraavan varsinaisen tietosisällön. Tämänlainen viesti on esitetty kuvassa 4.3. Kuvassa esiintyy myös void-tyyppinen osoitin, joka on käytössä mittausyksikön sisäisessä toiminnassa. Keskustellessaan ulkoisen viestintäväylän kanssa, tämä kenttä ei ole käytössä. Viestikehyksen pituus on 32 tavua ja tietosisällön pituutta ei ole rajoitettu. Kuitenkin käytännön rajana on mikro-ohjaimen kekomuistin koko, joka asettaa maksimikoon muutamaan kilotavuun.

Viestikehyksen rakenne on esitetty kuvassa 4.4. Viestin alkukehys kertoo millainen tietosisältö viestissä on ja miten se tulee tulkita. Koska viestiprotokolla kulkee sarjamuotoisena ulkoisten viestitysmedioiden yli, tulee jokaisen viestin alku tunnistaa tuntemattomien tavujen joukosta. Tämä on toteutettu lisäämällä protokollaan synkronisointisana, joka on määritetty heksadesimaalinumeroksi 0x12bb78ff. Kun viestitulkki vastaanottaa onnistuneesta synkronisointisanan, osaa se vastaanottaa



Kuva 4.3: Viestin rakenne (`comm_msg_t`)



Kuva 4.4: Viestikehyksen rakenne (`comm_msg_hdr_t`)

loput tavut oikein. Synkronisointisanaa seuraa neljä 8-bittistä kenttää, jotka sisältävät lähettävän ja vastaanottavan viestintämedian ja lähettävän ja vastaanottavan ohjelmistomoduulin. Jokainen viesti on lisäksi numeroitu kasvavasti ja aikaleimattu 32-bittisillä kentillä, joista timestamp-kenttä sisältää aikamoduulilta kysytyn GPS-ajan.

Kenttä mem_mode on käytössä ainoastaan viestitysmoduulin sisäisesti, ja kuvastaa millä tavalla viestitysmoduuli käsittelee tietosisällön muistialuetta. Koska viestitysjärjestelmän käyttökohteita on useita, esimerkiksi kilotavun kokoisen lohkon kirjoittaminen muistikortille työsäikeestä, tai muutaman tavun kokoisen viestisisällön lähettäminen keskeytyskontekstista, tulee viestitysjärjestelmän tukea useaa muistinkäsittelytapaa. Muistikäsittelytapoja on yhteensä neljä, joista yleisin käytössä oleva on varaus ja kopiointi, joka varaa dynaamisesti uuden muistialueen mikro-ohjaimen kekomuistista ja kopioi viestiä rakennettaessa annetun tietosisällön uudelle alueelle. Tämän jälkeen viestiä lähettävä taho on vapaa käyttämään tietosisällön muistia miten haluaa ja viestitysjärjestelmä vapauttaa automaattisesti varatun muistialueen, kun viesti on lähetetty. Toinen muistinkäsittelytapa on staattinen käyttäjämuisti, jolloin viestitysjärjestelmä lähettää annetun muistialueen, mutta ei tee mitään muuta. Tämän aikana käyttäjä ei saa koskea muistialueeseen, ja tämä tapa soveltuu esimerkiksi harvoin lähetettävään suurikokoiseen tietoon muistikortille. Kolmas tapa on käyttäjän dynaamisesti varaama muistialue kekomuistista. Viestin lähettämisfunktion kutsun jälkeen käyttäjä ei saa enää koskea muistialueeseen, ja viestitysjärjestelmä vapauttaa automaattisesti muistin kun viesti on lähetetty eteenpäin. Neljäs ja viimeinen tapa on käytössä, kun lähetetään pienikokoista tietosisältöä keskeytyskontekstista. Tällöin käytetään viestintämoduulin käynnistyessä esivarattua muistialuetta, jonka avulla vältetään dynaaminen muistinvarausoperaatio, mikä ei ole käytettävissä keskeytyksistä käsin.

Taulukossa 4.2 on esitetty src_id- ja dest_id-kenttien mahdolliset arvot. Ensimmäiset kahdeksan moduulinumeroa kuvastavat mittausyksikön sisäisiä ohjelmisto-

komponentteja, ja viimeiset neljä kuvastavat tietokoneelle tai tietokoneelta lähetettäviä viestityyppejä, jotka on tarkemmin kuvattu seuraavassa kappaleessa.

Taulukko 4.2: Moduulinumerot (`comm_module_id_e`)

Nimi	Arvo
ID_COMM	0
ID_STARTER	1
ID_GPS	2
ID_TIME	3
ID_BUFFERS	4
ID_APPS	5
ID_ACQ	6
ID_IRQ	7
ID_RUNTIME	8
NUMBER_OF_MODULES	9
PC_DEBUG	10
PC_NAV_SOLUTION	11
PC_SENSOR_DATA	12
MSG_CTRL	13

Taulukossa 4.3 on esitetty src_media- ja dest_media-kenttien mahdolliset arvot. Ensimmäiset neljä arvoa kuvastavat ulkoisia fyysisiä viestintäväyliä. Mittausyksikön sisäistä viestintäväylää kuvataan arvolla COMM_INTERNAL.

Taulukko 4.3: Viestintämediat (`comm_media_e`)

Nimi	Arvo
COMM_VCOM	0x00
COMM_UART1	0x01
COMM_UART2	0x02
COMM_SDCARD	0x03
COMM_INTERNAL	0x04
COMM_PC	0x05

Viestityypit

Mittausyksikkö tukee neljää erilaista viestityyppiä, jotka on esitetty taulukon 4.2 neljänä viimeisenä moduulinumerona. Nämä numerot eivät varsinaisesti ole moduulinumeroita, vaan kuvastavat enemmän viestikanavaa tai viestityyppiä. Ne on lisätty moduulinumeroiden joukkoon sen vuoksi, ettei viestikehykseen tarvitse lisätä yhtä ylimääräistä viestityyppiä kuvaavaa kenttää.

Tulosteviesti

Tulosteviesti sisältää käyttäjän mikro-ohjaimelta tulostamaa dataa ASCII-muodossa (American Standard Code for Information Interchange). Tämä viestityyppi vastaa PC_DEBUG-arvoa taulukossa 4.2. Viestintämoduuli tarjoaa printf-tyylisen apufunktion tulosteviestien lähettämiseen. Tulosteviestien tärkein käyttökohde on ohjelmistokehityksen tukena auttamaan näkemään mitä mikro-ohjaimella tapahtuu.

0	8	16	24
get_set	target	command	query_resp
data[0]	data[1]	data[2]	data[3]
data[4]	data[5]		

Kuva 4.5: Ohjausviestin rakenne (`comm_msg_t`)

Tämän viestin varsinainen tietosisältö on nolla-terminoitu ASCII-muotoinen merkkijono.

Mittaustietoviesti

Mittausyksikön eräänä käyttäjäsovelluksena on sovellus, joka kerää säännöllisin väliajoin tietoa anturimittauksista ja lähettää niitä viestinä eteenpäin tietokoneelle. Sovellus kerää useita mittauksia muistialueelle ja lähettää ne kerralla tietokoneelle. Usean mittauksen kerrallaan lähettämisen tavoitteena on pienentää viestikehyksien osuutta lähetetystä tietomäärästä. Sovellus lähettää viestin tietosisältönään listauksessa 4.1 esitettyjä tietorakenteita ja käyttää viestityyppinumeronaan taulukon 4.2 arvoa PC_SENSOR_DATA.

Ohjausviesti

Ohjausviesti on erityinen viestityyppi, jonka avulla ohjelmistomoduulien toimintaa voidaan ohjata. Ohjausviestit mahdollistavat esimerkiksi mittauksen aloittamisen tai GPS-laitteen paikannusten päivitysnopeuden asettamisen. Ohjausviestit kulkevat lähinnä ulkoisesta viestintämediasta laitteelle päin, mutta myös sisäisestä ohjelmistomoduulista muille sisäisille ohjelmistomoduuleille. Esimerkkinä laitteen sisällä kulkevista ohjauviesteistä on käynnistysmoduulin lähettämät viestit, jotka määrittävät antureiden ja GPS-laitteen lukunopeudet laitteen käynnistyessä.

Ohjausviesti on aina pituudeltaan 10 tavua, ja sen rakenne on esitetty kuvassa 4.5. Viestin neljä ensimmäistä tavua sisältävät tietoa pyydetystä toiminnosta, ja niiden sisältö on kuvattu taulukossa 4.4. Loput kuusi tavua ovat käytössä tietosisältönä mikäli ohjausviesti asettaa esimerkiksi uudet antureiden lukunopeudet. Taulukko 4.4 kuvaa mahdolliset arvot neljälle ensimmäiselle kentälle. Kaikki taulukossa esiintyvät arvot eivät sovi keskenään yhteen, ja tuleville käyttäjälle lienee helpointa tutkia ohjelmakoodista tietyn moduulin sisäinen toteutus. Esimerkkinä ohjausviestistä voidaan mainita ID_APPS moduulinumeroon lähetetty viesti, joka sisältää seuraavat tavut: CMD_SET, CMD_SD, CMD_FREQ, CMD_QUERY, 100, 0 ,0 ,0 ,0 ,0. Tämä viesti asettaa muistikortille kirjoittavan käyttäjäsovelluksen päivittämään anturilukemat 100 Hz taajuudella.

Taulukko 4.4: Ohjausviestin sisältö (`ctrl_cmd_get_set_e`)

get_set	target	command	query_resp
CMD_GET = 0x00 CMD_SET = 0x01	CMD_NOT_USED = 0x00 CMD_KALMAN = 0x01 CMD_SD = 0x02	CMD_MODE = 0x00 CMD_FREQ = 0x01 CMD_VOLTAGE = 0x02 CMD_FIX = 0x03	CMD_QUERY = 0x00 CMD_RESP = 0x01

4.2.5 Tiedon tallennus muistikortille

Tiedon tallentamisesta muistikortille vastaa käyttäjäsovellus, joka rekisteröityy puskurimoduulin tarjoamaan rajapintaan ja hakee sieltä tasaisin väliajoin anturitietoa. Aluksi muistikortille kirjoitus toteutettiin ASCII-muotoisena merkkijonona, mutta tämän ratkaisun suorituskyky ei ollut riittävällä tasolla suuren tietomäärän takia, kun jokainen anturilukema pitää muuttaa liukulukumuodosta tekstimuotoon. Ratkaisuksi suorituskykyongelmaan muistikortille kirjoitetaan tiedot binäärimuodossa.

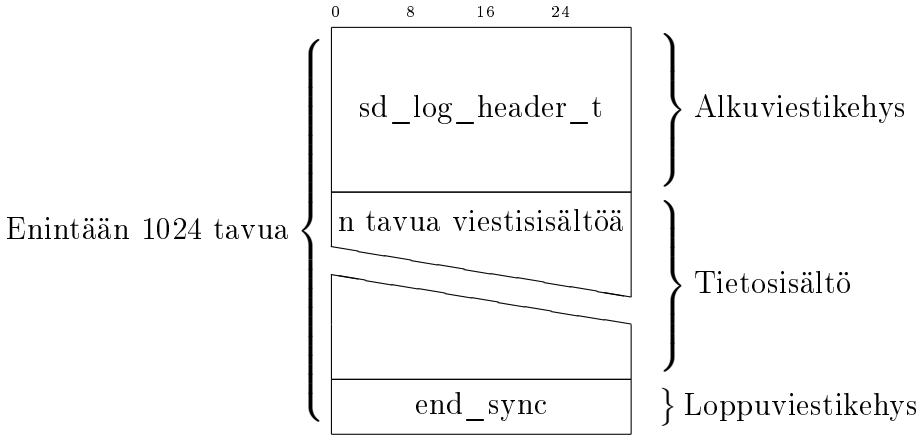
Kirjoitusformaatti

Muistikortille kirjoitettava tietopaketti tukee anturimittauksia ja supistettua GPS-paikkamittausta. Ongelma tiedontallennusformaatin tekemisessä on, että eri antureilta ja paikkamoduulilta saadaan mittauksia eri tahdissa. Muistikorttisovelluksen tiedonkeruun ajanjaksona on tyypillistä että kiihtyvyysanturin mittauksia tulee kolme kertaa enemmän kuin gyron mittauksia, ja normaalisti vain joka kymmenenteen tiedonkeruujaksolle osuu GPS-paikkapäivitys. Tämä ongelma on ratkaistu muodostamalla listauksessa 4.4 esiintyvä tietopaketin alkukehys, joka sisältää tiedon montako anturimittausta tietopaketissa on ja esiintyykö siinä yhtäkään GPS-paikkamittausta. Mikäli GPS-mittaus löytyy tietopaketista, se sijaitsee aina viimeisenä tietosisällön tietorakenteena. Kentät meas_amount ja gps_existent kuvastavat näitä arvoja.

```
1 typedef struct sd_log_header {
2     uint32_t start_word;
3     uint32_t timestamp;
4     uint8_t meas_amount;
5     uint8_t gps_existent;
6     uint8_t kalman_existent;
7     uint8_t pad_byte;
8 } sd_log_header_t;
```

Listaus 4.4: Muistikortin mittaustietopaketin alkukehys

Kuvassa 4.6 on esitetty kokonaisen tietopaketin rakenne. Tietopaketin koko on rajattu yhteen kilotavuun ja aina, kun tämä raja tulee täyteen, lähetetään tietorakenne



Kuva 4.6: Muistikorttiformaatin rakenne

muistikortille. Muistikortille kirjoituksen aikana samaa muistialuetta ei voi käyttää, ja tästä syystä sovellus osaa automaattisesti vaihtaa toiseen esivarattuun muistialueeseen. Yhteensä näitä esivarattuja kilotavun muistialueita on seitsemän kappaletta, joten sovelluksella on seitsemän kilotavun puskuri muistikortille kirjoittamisessa. Binäärimuotoinen tietopaketti muutetaan tietokoneohjelmalla ASCII-muotoiseksi tekstitiedostoksi, joka voidaan lukea esimerkiksi Matlab-ohjelmassa. Jotta tietokoneen tulkki osaa etsiä yhden kokonaisen tietopaketin binäärimuotoisesta tietovirrasta, sisältää yksi tietopaketti alku- ja loppusynkronointisanan.

Tietopaketin varsinainen tietosisältö sisältää listauksessa 4.1 esiintyviä mittaustietorakenteita ja listauksessa 4.5 esiintyviä typistettyjä GPS-paikkaratkaisurakenteita.

```
1  typedef struct sd_log_gps {
2      int32_t Latitude;    /* degrees * 1e7 */
3      int32_t Longitude;   /* degrees * 1e7 */
4      float Groundspeed;   /* m/s */
5  } sd_log_gps_t;
```

Listaus 4.5: Muistikortin GPS-tietorakenne

Parseriohjelma tietokoneella

Tietokoneella ajettava binäärimuotoisen mittaustiedoston purkuohjelma on toteutetty Python-kielellä. Tietoformaatin tulkkausalgoritmi on esitetty listauksessa 4.6.

```
1  Etsi aloitussynkronointi
2  Lue tavuja muistiin niin kauan kunnes lopetussynkronointi löytyy
3  Lue alkukehyksen aikaleima
4  Lue kentät meas_amount, gps_existent ja kalman_existent
5  Lue tavuista meas_amount määrä tietorakennetta measurement_t
6  Mikäli lippu gps_existent on asetettu, lue tietotyyppi sd_log_gps_t
7  Merkitse tavut käytetyksi ja palaa kohtaan 1
```

Listaus 4.6: Algoritmi muistikortin tietoformaatin tulkitsemiseksi

4.2.6 Sovellusrajapinta

Sovellukset ovat laitteessa ajettavia algoritmeja, jotka käyttävät reaaliaikaista anturitietoa ja laskevat siitä esimerkiksi laitteen asennon. Laitteen ohjelmistokomponentit tarjoavat useita rajapintoja, joiden avulla käyttäjä voi toteuttaa sovelluksensa laitteeseen. Seuraavissa kappaleissa esitellään nämä rajapinnat ja annetaan käyttöesimerkkejä.

Rekisteröityminen anturitietoon

Sovellus pääsee käsiksi mittausdataan rekisteröitymällä puskurimoduuliin, joka palauttaa sovellukselle uusimmat mittaukset sisältävän tietorakenteen. Tämä tietorakenne sisältää jokaiselle anturille yksipaikkaisen FreeRTOS-jonon, jonka kautta pääsee käsiksi uusimman mittauksen muistialueeseen. Mikäli jono on jo tyhjä, tarkoittaa se että edellisen lukukerran jälkeen anturilta ei ole kysytty uutta mittausta. Tämä tietorakenne on esitetty listauksessa 4.7.

```
1  typedef struct buff_data {
2      xQueueHandle acce;
3      xQueueHandle magn;
4      xQueueHandle gyro;
5      xQueueHandle baro;
6      xQueueHandle gps;
7  } buff_data_t;
```

Listaus 4.7: Mittaustietorakenne

Seuraavana on esitelty puskurimoduulin funktiorajapinta, joita käyttämällä asiakassovellus pääsee käsiksi anturitietoon.

Esittely	`buff_data_t* buff_register(void)`
Selitys	Asiakassovellukset voivat käyttää tätä funktiota rekisteröityäkseen puskurimoduuliin. Funktiokutsun palauttaman tietorakenteen avulla sovellukset pääsevät käsiksi anturitietoon ja GPS-paikkaratkaisuihin.
Parametrit	-
Palautusarvo	Osoitin `buff_data_t`-rakenteeseen.

Esittely	`uint32_t buff_unregister(buff_data_t** data)`
Selitys	Kutsumalla funktiota asiakassovellus voi poistaa itsensä puskurimoduulien palveltavien asiakkaiden listasta. Kutsun jälkeen parametrina annettu osoitin ei ole enää käytettävissä.
Parametrit	`data` Osoitin `buff_data_t`-rakenteeseen.
Palautusarvo	Palauttaa numeron 1, mikäli operaatio on onnistunut.

Esittely	`uint32_t buff_free_memory(buff_item_t** b_item)`
Selitys	Asiakassovelluksen tulee kutsua tätä funktiota anturitiedon käyttämisen jälkeen. Funktiokutsun jälkeen anturitiedon käyttämä muistialue merkitään vapaaksi ja sen sisältöä ei voi enää käyttää.
Parametrit	`b_item` Osoitin `buff_item_t`-rakenteeseen.
Palautusarvo	-

Listauksessa 4.8 on esitetty käyttöesimerkki puskurimoduulin tyypillisestä käytöstä. Siinä asiakas rekisteröityy puskurimoduuliin ja saa sieltä paluuarvokseen osoittimen listauksen 4.7 mukaisen tietorakenteeseen. Käyttämällä tämän tietorakenteen jonoja FreeRTOS-jonofunktion xQueueReceive() kautta, voi asiakas kysellä viimeisimmän mittausdatan kultakin anturityypiltä. Jono sisältää listauksen 4.2 mukaisia puskuritietorakenteita, joiden kautta varsinaiseen mittaustietoon pääsee käsiksi.

```
1  /* Rekisteröidy puskurimoduulin tietovirtaan */
2  buff_data_t* sd_buff = buff_register();
3
4  buff_item_t* b_item = NULL;
5
6  /* Lue FreeRTOS-jonosta buff_item-tyyppinen osoitin. */
7  if( xQueueReceive(sd_buff->acce, &b_item, 0) == pdTRUE ) {
8      /* Prosessoi kiihtyvyysanturin uusin anturilukema */
9      process_acce_data(b_item->meas.f_payload);
10
11     /* Merkitse käytetty muistiyksikkö vapaaksi */
12     buff_free_memory( &b_item );
```

```
13  }
14
15  /* Poista rekisteröinti puskurimoduulista */
16  uint32_t error = buff_unregister(sd_buff);
```

Listaus 4.8: Käyttöesimerkki

Rekisteröityminen aikakutsuihin

Aikamoduuli tarjoaa sovellukselle ajoituksiin liittyviä palveluita. Sovellus voi rekisteröityä säännöllisiin aikakutsuihin, jotka toimivat takaisinkutsufunktioiden avulla. Takaisinkutsufunktiota kutsutaan keskeytyskontekstista, joten siellä ei voi tehdä aikaa vieviä operaatioita. Palvelu kuitenkin soveltuu esimerkiksi ajan mittaamiseen tai työsäikeen herättämiseen. Aikakutsujen parametrit asetetaan listauksessa 4.9 esitetyllä tietorakenteella. Tämä tietorakenteen ensimmäinen kenttä target_interval_ms sisältää käyttäjän pyytämän aikakutsujen aikavälin millisekunneissa. Seuraava kenttä sisältää aikakutsupalvelun sisäisesti käyttämän kentän current_ms, joka kertoo montako millisekuntia on kulunut viimeisimmästä aikakutsusta. Viimeinen kenttä number_to_send sisältää numeron, joka lähetetään takaisinkutsufunktiolle. Tämän numeron avulla yksi takaisinkutsufunktio voi palvella useaa aikakutsua.

```
1  typedef struct time_timeout_ctrl {
2      uint32_t target_interval_ms;
3      uint32_t current_ms;
4      uint32_t number_to_send;
5  } time_timeout_t;
```

Listaus 4.9: Käyttöesimerkki

Seuraavissa taulukoissa on esitelty aikamoduulin aikakutsupalvelun rajapintafunktiot. Funktioita on palveluun rekisteröitymiseen, aikakutsujen muokkaamiseen ja palvelusta poistumiseen.

Esittely	`uint32_t time_register(time_cb callback_func, time_timeout_t* timeouts, uint32_t num)`
Selitys	Asiakassovellukset voivat rekisteröityä aikapalvelumoduuliin tämän funktion avulla. Funktiokutsun jälkeen aikamoduuli kutsuu käyttäjän ilmoittamaa takaisinkutsufunktiota säännöllisin väliajoin.
Parametrit	`callback_func` Takaisinkutsufunktio, joka on tyyppiä `typedef void (*time_cb)(uint32_t)` `timeouts` Tietorakennetaulukko, joka sisältää tiedon palveltavista aikakutsuista `num` Aikapyyntörakenteiden lukumäärä taulukossa
Palautusarvo	-

Esittely	`uint32_t time_change_timeout(time_cb callback_func, time_timeout_t* timeouts, uint32_t num)`
Selitys	Tämä funktio muuttaa jo olemassa olevan rekisteröinnin parametreja. Tämän funktion avulla voidaan muuttaa aikakutsujen määrää ja taajuutta.
Parametrit	`callback_func` Takaisinkutsufunktio, joka on tyyppiä `typedef void (*time_cb)(uint32_t)` `timeouts` Tietorakennetaulukko, joka sisältää tiedon palveltavista aikakutsuista `num` Aikapyyntörakenteiden lukumäärä taulukossa
Palautusarvo	-

Esittely	`uint32_t time_unregister_timeout(time_cb callback_func)`
Selitys	Tämä funktio poistaa asiakassovelluksen rekisteröinnin aikamoduulista. Funktiokutsun jälkeen aikakutsuja ei enää tapahdu.
Parametrit	`callback_func` Takaisinkutsufunktio, joka on tyyppiä `typedef void (*time_cb)(uint32_t)`
Palautusarvo	-

Listaus 4.10 sisältää esimerkin aikakutsupalveluun liittymisestä. Tässä esimerkissä ilmenee miten aikakutsupalveluilla voidaan herättää FreeRTOS-pohjainen työsäie säännöllisin väliajoin. Funktio apps_init luo työsäikeen ja rekisteröi takaisinkutsufunktion apps_timeout_cb ajastinpalveluihin. Ajastinpalvelulle annetaan parametrit tietorakenteella timeout_param, joka aiheuttaa takaisinkutsufunktion kut-

sumisen 20 millisekunnin välein. Takaisinkutsufunktio suoritetaan keskeytyspalvelijan kontekstissa, joten se vain herättää työsäikeen lisäämällä numeron jonoon worker_wakeup_queue. Varsinainen työ tullaan tekemään työsäikeessä apps_worker_task, joka siirretään suoritukseen heti kun jonoon on tullut elementti.

```
1  /* Globaalit muuttujat */
2  time_timeout_t timeout_param = { .target_interval_ms = 20,
3                                   .current_ms = 0,
4                                   .number_to_send = 0 };
5  xQueueHandle worker_wakeup_queue = NULL;
6
7  void apps_init(void) {
8      xTaskCreate(apps_worker_task, "apps_worker", 384, NULL,
9                  configMAX_PRIORITIES - 1, NULL);
10     worker_wakeup_queue = xQueueCreate( 1, sizeof(uint32_t) );
11
12     /* Rekisteröinti aikakutsupalveluun */
13     time_register(apps_timeout_cb, &timeout_param, 1);
14 }
15
16 void apps_timeout_cb(uint32_t num) {
17     portBASE_TYPE xHigherPriorityTaskWoken = 0;
18     xQueueSendFromISR(worker_wakeup_queue, &num,
19                       &xHigherPriorityTaskWoken);
20 }
21
22 void apps_worker_task(void *pvParameters) {
23     for(;;) {
24         uint32_t timeout_id;
25         if( xQueueReceive(worker_wakeup_queue, &timeout_id,
26                           portMAX_DELAY) == pdTRUE ) {
27             /* Tätä koodia kutsutaan tasaisin väliajoin */
28         }
29     }
30 }
```

Listaus 4.10: Käyttöesimerkki

Sovellustiedon tallennus

Viestintämoduuli sisältää funktiorajapinnan tiedoston luomiselle tai avaamiselle muistikortille ja siihen kirjoittamiseen. Tämä rajapinta on esitetty seuraavissa taulukoissa.

Esittely	`file_handle_t* comm_open_file(char* filename, uint8_t mode)`
Selitys	Asiakassovellukset voivat luoda uuden tiedoston tai avata jo olemassa olevan tiedoston tällä funktiolla.
Parametrit	`filename` Tiedoston nimi nolla-terminoituna merkkijonona `mode` Tiedoston avaamisparametrit
Palautusarvo	Osoitin file_handle_t tyyppiseen tiedostokahvaan

Esittely	`void comm_sd_send_data(file_handle_t* fd, void* data, uint32_t len)`
Selitys	Tämä funktio kirjoittaa osoittimen päässä olevan datan tiedostoon. Funktio toimii ainoastaan mikäli annettu tiedostokahva on avattu.
Parametrit	`fd` Osoitin tiedostokahvaan `data` Osoitin tiedostoon kirjoitettavaan tietoon `len` Kirjoitettavien tavujen määrä
Palautusarvo	-

Esittely	`void comm_close_file(file_handle_t* fd)`
Selitys	Sulkee avoinaisen tiedoston.
Parametrit	`fd` Osoitin avonaiseen tiedostokahvaan
Palautusarvo	-

Listauksessa 4.11 on esitetty miten viestintämoduulin tiedostopalvelua käytetään. Ensin käyttäjän tulee esitellä file_handle_t-tyyppinen osoitin, johon pyydetään avonainen tiedostokahva funktiolla comm_open_file(). Tämän jälkeen käyttäjä voi kirjoittaa avaamaansa tiedostoon funktiolla comm_sd_send_data(), jonka parametreina on avattu tiedostokahva, osoitin kirjoitettavaan tietoon ja tiedon pituus tavuissa. Lopuksi käyttäjän tulee sulkea tiedosto kutsumalla funktiota comm_close_file(). Mikäli käyttäjä ei sulje tiedostoa ennen kuin mittausyksikkö sammutetaan, kaikki tiedostoon kirjoitettu sisältö katoaa.

```
1  /* Alustettu merkkijono */
2  char esim_str[7] = "tietoa";
3
4  /* comm_sd.h:ssa esitelty tietorakenne */
5  file_handle_t* apps_fd = NULL;
6
7  /* Luo uusi tiedosto, jonka nimi on sovellustietoa.txt */
8  apps_fd = comm_open_file("sovellustietoa.txt", FA_CREATE_NEW|FA_WRITE);
9
10 /* Lähetä edellämääritelty merkkijono tiedostoon */
```

```
11  comm_sd_send_data(apps_fd, esim_str, strlen(esim_str));
12
13  /* Sulje tiedosto */
14  comm_close_file(apps_fd);
```

Listaus 4.11: Käyttöesimerkki

5. KALIBROINTI JA TESTAUS

5.1 Kalibrointi

5.1.1 Virhemallit

Kuvassa 5.1 on esitetty virhemalli, jossa anturimittaus $\mathbf{y}$ korjataan matriisilla $\mathbf{S}$ ja vektorilla $\mathbf{b}$. Lopputuloksena saadaan kalibroitu mittaus $\mathbf{u}$. Anturin lukemaan vaikuttaa akselin skaalauskerroin, akseleiden välinen ristikuuluvuus ja niiden poikkeama nollakohdasta. Matriisilla $\mathbf{S}$ korjataan kaksi ensimmäistä virhelähdettä ja vektorilla $\mathbf{b}$ korjataan poikkeama seuraavasti

$$\mathbf{u} = \mathbf{S}(\mathbf{y} - \mathbf{b}) \quad , jossa\ \mathbf{S} = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{xy} & S_{yy} & S_{yz} \\ S_{xz} & S_{yz} & S_{zz} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}. \tag{5.1}$$

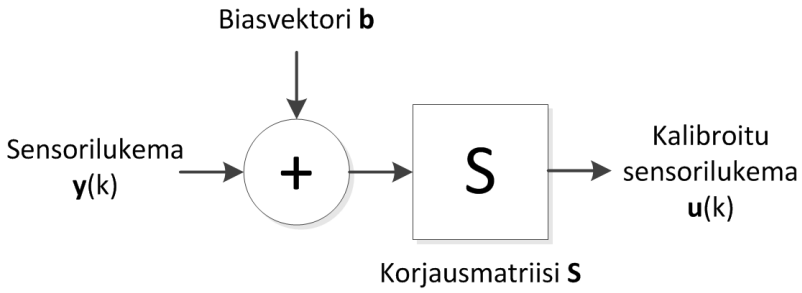
5.1.2 Pyörityspöytä ja kalibrointikuutio

Kalibrointimittaus suoritetaan pyörityspöydän avulla, jonka tavoitteena on asettaa mittausyksikkö tunnettuihin asentoihin tai pyörittää mittausyksikköä tunnetulla kulmanopeudella.

Pyörityspöytä on Velmexin valmistama BTS5990 [19], joka sisältää askelmoottorin ja suhteella 90:1 laskevan kulmavaihteen. Mittauksissa käytetyn askelmoottorin kanssa kokonaiseen kierrokseen kuluu 36000 askelta, jolloin yksi askel kääntää pyöritystasoa 0,01 astetta. Käytetty askelmoottoriohjain on Velmex VXM [18], joka on mahdollista ohjelmoida RS-232 -väylän (Recommended Standard 232) kautta. Pyörityspöytään on lisäksi kiinnitetty alumiinikuutio, joka mahdollistaa mittausyksikön kiinnittämisen kolmeen asentoon kuution eri sivuille.

5.1.3 Kalibrointialgoritmit

Kalibrointimittaukset koostuvat kolmesta erillisestä mittauksesta, joiden välillä mittausyksikön asentoa vaihdetaan alumiinikuutiossa – tämän avulla antureiden jokaiselta akselilta saadaan mittausdataa. Jokainen mittaus asettaa kiihtyvyysanturin neljään esimääriteltyyn asentoon. Tällöin kiihtyvyysanturin antama lukema asennossa n on $\mathbf{y}_n$ ja esimääritellyn asennon oletettu lukema on $\mathbf{a}_n$. Koska mittauksia on



Kuva 5.1: Antureiden virhemalli

kolme ja asentoja neljä, saadaan kiihtyvyysanturin lukemia yhteensä 12 kappaletta.

Käytetty kalibrointialgoritmi on esitetty lähteessä [20]. Algoritmi perustuu kumulatiivisen virhefunktion muodostamiseen tunnetuista mittauksista. Virhefunktio muodostetaan kaavan 5.1 avulla vähentämällä siitä oletettu anturilukema $\mathbf{a}_n$. Mikäli tämä funktio palauttaa arvon nolla, vastaa anturilukema esimääritellyn asennon oletettua lukemaa. Lisäämällä jokaisen lukeman n mittaukset yhteen, saadaan virhefunktioksi kiihtyvyysanturin tapauksessa seuraava kaava:

$$\mathbf{e}_a = \begin{cases} \sum_{n=1}^{12} [\mathbf{S}_a(\mathbf{y}_n - \mathbf{b}_a) - \mathbf{a}_n] \\ \sum_{n=1}^{12} [||\mathbf{S}_a(\mathbf{y}_n - \mathbf{b}_a)|| - a_g] \end{cases} . \tag{5.2}$$

Vastaavasti gyroskoopin tapauksessa saadaan seuraava kaava

$$\mathbf{e}_\omega = \begin{cases} \sum_{n=1}^{9} [\mathbf{S}_\omega(\mathbf{y}_n - \mathbf{b}_\omega) - \boldsymbol{\omega}_n] \\ \sum_{n=1,4,7} [||\mathbf{S}_\omega(\mathbf{y}_n - \mathbf{b}_\omega)|| - \omega_0] \end{cases} . \tag{5.3}$$

Tällä kertaa mittauksia on vain yhdeksän kappaletta, joten $n = 1..9$. Virhefunktiot voidaan minimoida esimerkiksi Matlabin fsolve-työkalun avulla. Lopputuloksena fsolve palauttaa ne kalibrointivakioiden arvot, joiden avulla anturin virhettä kuvaava summafunktio saavuttaa pienimmän arvonsa.

5.1.4 Tulokset

Kuvassa 5.3 on esitetty yhden mittausyksikön kalibrointimittaus kahden akselin suhteen. Kuvasta ilmenee, että mittausyksikköä on ensin pyöritetty z-akselin suhteen kulmanopeudella 30 °/s viiteen eri asentoon aikajaksolla 0-1800 s. Tämän jälkeen mittausyksikköä on pyöritetty vastakkaiseen suuntaan kulmnanopeudella 60 °/s siten, että mittausyksikön alkuasento on sama kuin mittauksen lähtötilanteessa. Kulmanopeudet kuvaajassa on esitetty yksikössä radiaania sekunnissa. Yhden akselin mittauksen jälkeen, mittaus on katkaistu ja mittausyksikkö on siirretty toiselle alu-

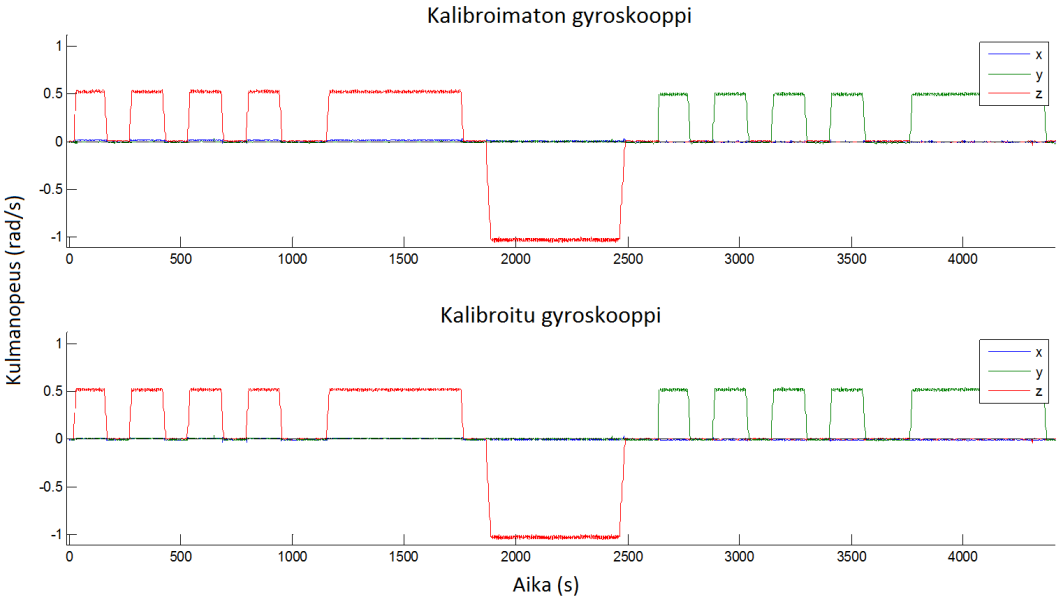
Kuva 5.2: Mittausjärjestely antureiden kalibrointiin

miinikuution sivulle siten, että mittausyksikkö pyörii nyt y-akselin suhteen. Nämä kaksi mittausta on liitetty peräkkäin, että koko mittaustilanne saadaan esitettyä kuvaajassa. Toisen akselin mittaus löytyy aikajaksolta 2500-5000 s.

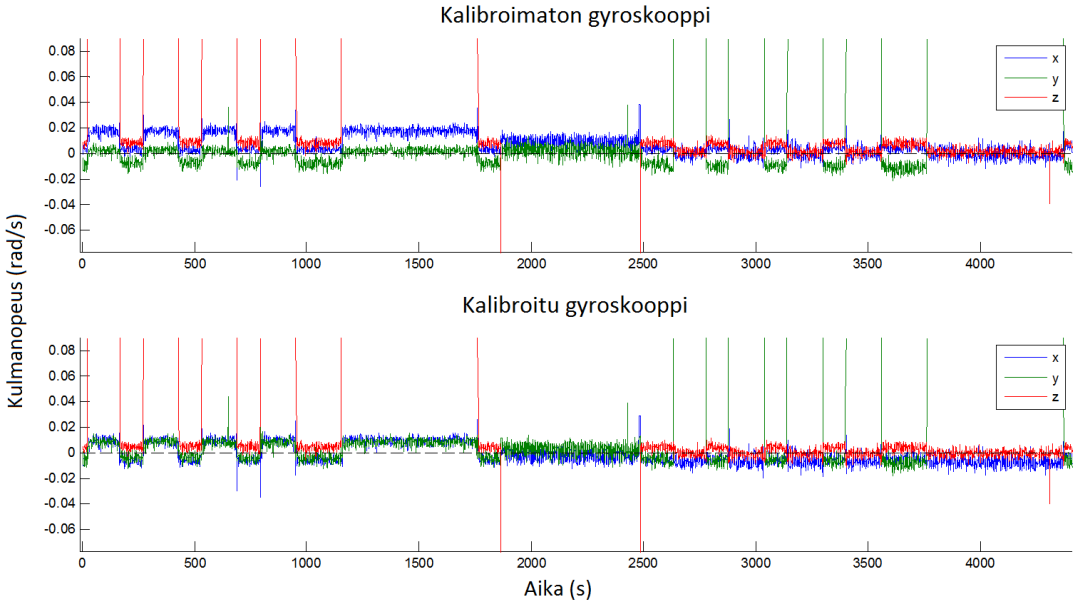
Kuva 5.4 esittää samaa mittausta kuin aikaisemminkin, mutta tällä kertaa tarkastellaan vain pieniä kulmanopeuksia. Gyroskoopin liittäminen piirilevylle ja gyroskoopin sisäinen rakenne aiheuttavat akselien välistä ristikuuluvuutta, joka näkyy selvästi ylemmästä kuvaajasta mittausyksikön pyöriessä. Tällöin z-akselin kokema kulmanopeus näkyy myös muuttuvina x- ja y-akselin arvoina. Ylemmästä kuvaajasta huomataan myös kaikkien akseleiden poikkeama nollakohdasta mittausyksikön ollessa paikallaan.

Kuvan 5.4 alemmasta kuvaajasta huomataan, miten kalibrointivakioiden käyttöönotto vaikuttaa ristikuuluvuuteen ja poikkeamaan nollakohdasta. Akseleiden välinen ristikuuluvuus on vähentynyt, mutta se ei ole kokonaan poistunut. Poikkeama nollakohdasta on nyt myös vähentynyt, ja se on nyt tasaisemmin jakautunut nollakohdan molemmille puolille.

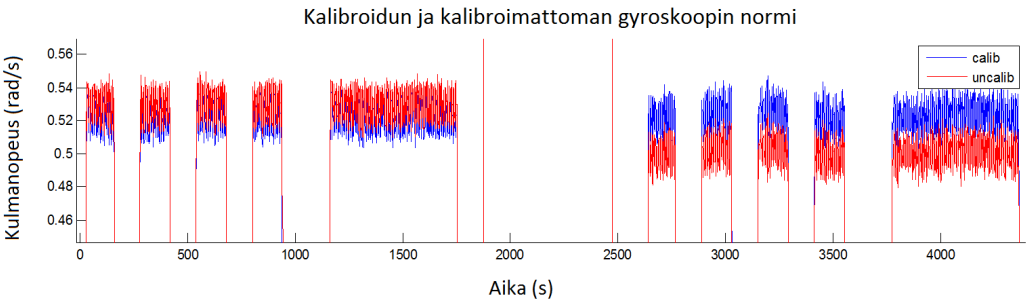
Aikaisemmista kuvista kävi ilmi kahden virhetekijän pieneneminen kalibrointivakioiden käyttöönoton jälkeen. Nämä olivat akselien ristikuuluvuus ja poikkeama nollakohdasta. Kolmas virhetekijä on akselin skaalauskerroin, joka ilmenee pyöritettäessä mittausyksikköä tunnetulla vakionopeudella. Ennen kalibrointia mittausyk-



Kuva 5.3: Gyroskoopin kalibroinnin tulokset



Kuva 5.4: Gyroskoopin kalibroinnin tulokset pienellä kulmanopeudella



Kuva 5.5: Gyroskoopin kalibroinnin tulokset suurella kulmanopeudella normitettuna

sikön eri akselien ilmaisemat kulmanopeudet eroavat tästä tunnetusta kulmanopeudesta. Kuva 5.5 esittää miten mittausyksikkö toimii ennen ja jälkeen kalibroinnin tässä tilanteessa. Mittausyksikön kulmanopeudet on normitettu ja piirretty päällekkäin, jotta kalibroinnin vaikutus saadaan selkeästi esiin. Punainen käyrä esittää kalibroimatonta tilannetta ja sininen käyrä tilannetta kalibroinnin jälkeen. Kuten aikaisemmissakin kuvissa, mittausyksikköä on pyöritetty z-akselin suhteen aikajaksolla 0-2500 s ja y-akselin suhteen ajanjaksolla 2500-5000 s.

Kuvaajasta huomataan että kyseisen mittausyksikön z-akselin skaalauskerroin on ollut hyvin lähellä oikeaa, mutta y-akselin skaalaus on ollut liian pieni. Kalibroinnin jälkeen molempien akseleiden kulmanopeudet ovat lähellä oikeaa, eli 0,523 rad/s joka vastaa pyöritysnopeutta 30 °/s.

5.2 Testaus

Kalibroinnin lisäksi laitetta on testattu useissa eri käyttöympäristöissä ja -kohteissa. Näistä ensimmäisenä käyttökohteena voidaan mainita moottoripyörämittaukset Tampereen keskustassa. Mittausten tarkoituksena oli saada kiihtyvyysanturilta ja gyroskoopilta referenssidataa tärinäisestä ajotilanteesta mukulakivetyksellä. Mittaustilanteessa kuljettajan kypärään, selkään ja moottoripyörän takaosaan on kiinnitetty mittausyksiköt, joiden aika on tahdistettu GPS:n avulla. Usean mittausyksikön mittaukset käynnistettiin ja sammutettiin samanaikaisesti matkapuhelimella Bluetooth-yhteyden yli. Kuvassa 5.6 on esitetty mittausyksiköiden sijoitus. Kuvassa ei näy kuljettajan selkään kiinnitettyä kolmatta mittausyksikköä. Mittausyksiköt suoriutuivat hyvin kahdesta noin kymmenen minuutin mittaisesta mittausjaksosta ja niistä saatua tietoa on käytetty ajotapaluokittelijan kehittämisessä.

Toisena mittauskohteena esitellään nuken tiputusmittaukset Tampereen Hervannassa. Mittaustilanteessa noin 80 kilon painoiseen paininukkeen on kiinnitetty kaksi mittausyksikköä päähän sekä selkään, joka on esitetty kuvassa 5.7. Mittausyksiköiden kellot on tahdistettu GPS:n avulla. Mittauksen tavoitteena on saada kiihtyvyysanturilta ja gyroskoopilta referenssidataa siitä, kun nukkea tiputetaan 1-2 metrin

(a) Kypärässä (b) Moottoripyörän takaosassa

Kuva 5.6: Mittausyksikön sijoittelut moottoripyörämittauksissa

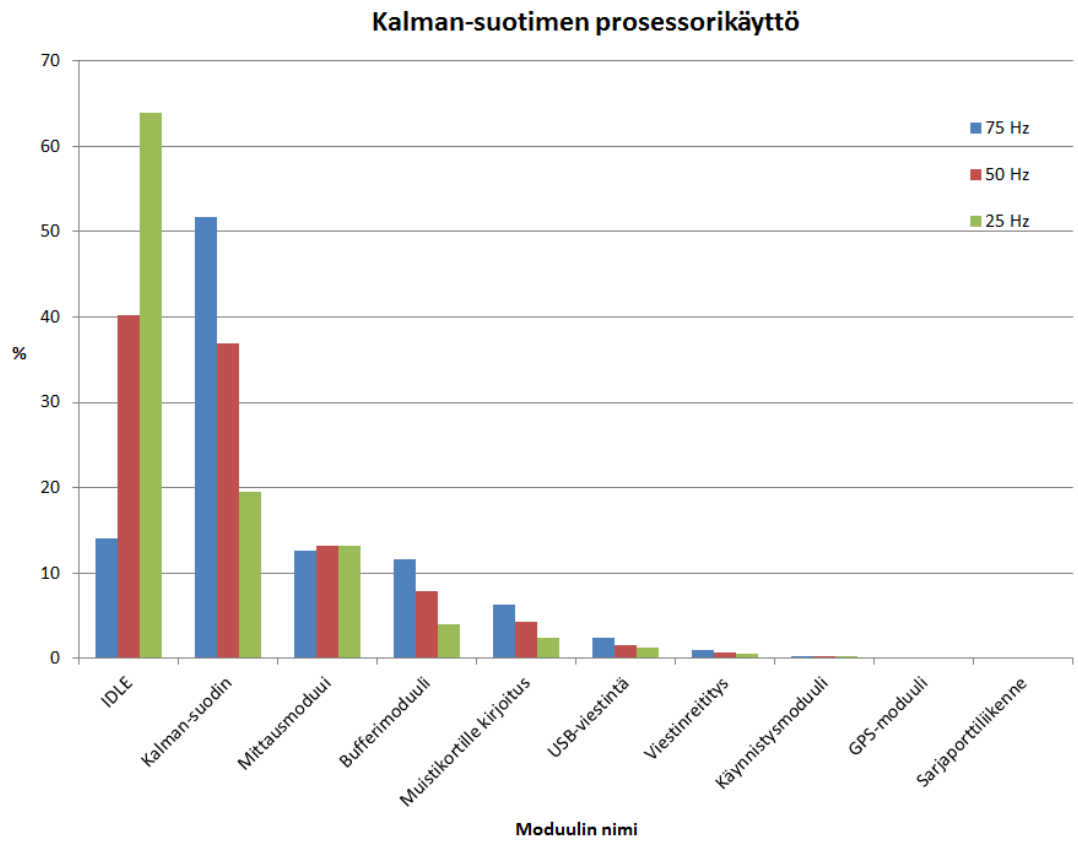
korkeudesta maahan. Kokeen tarkoituksena on mallintaa mitä moottoripyöräkuljettajalle tapahtuu törmäystilanteessa. Mittausyksiköt suoriutuivat rajusta mittaustilanteesta hyvin, ja mittauksesta saatua tietoa on käytetty ajatapaluokittelijan kehittämisessä.

Mittausyksikön suorituskykyä on mitattu käyttämällä apuna OpenPilot-projektissa kehiteltyä Kalman-suodinta [22]. Suotimen tehtävänä on yhdistää kiihtyvyysanturin, magnetometrin, gyroskoopin ja GPS-laitteen antama tieto ja poistaa siitä kohinaa ja häiriöitä. Suodin antaa ulostulona tilastollisesti parhaan arvion mittausyksikön asennosta ja sijainnista. Kalman-suodin on matemaattisesti raskas algoritmi, joten se soveltuu hyvin mittausyksikön suorituskyvyn mittaamiseen. Kuvassa 5.8 on esitetty suotimen käyttämä prosessoriaika päivitysnopeuksilla 25, 50 ja 75 Hz. Kuvaajan aineisto on kerätty FreeRTOS-käyttöjärjestelmän palveluilla, joilla yksittäisen säikeen käyttämää suoritinaikaa voidaan mitata. Jokaista tapausta on suoritettu noin kymmenen minuuttia mittausyksikössä.

Kuvasta huomataan, että Kalman-suotimen päivitysnopeuden kasvaessa, sen vaa-

(a) Nuken päässä (b) Nuken selässä

Kuva 5.7: Mittausyksikön sijoittelut nuken tiputusmittauksissa



Kuva 5.8: Kalman-suotimen käyttämä prosessoriaika

tima prosessoriaika kasvaa huomattavasti. Havaittu maksimipäivitysnopeus on noin 75 Hz, joka jättää vielä noin kymmenen prosenttia suorittimen vapaa-ajaksi. Tämä päivitysnopeus on riittävä navigointisovelluksiin, joten mittausyksikkö pystyy myös inertiaalinavigointiin.

6. JOHTOPÄÄTELMÄT

Työssä toteutettu mittausyksikkö on osoittautunut käytännölliseksi ja tehokkaaksi mittausvälineeksi, sillä laitteen pieni koko ja robusti kotelointi mahdollistavat käytön useissa erilaisissa mittaustilanteissa. Mittausyksikön käyttöliittymä on toteutettu työn yhteydessä tehdyn Android-sovelluksen kautta, joka mahdollistaa usean mittausyksikön samanaikaisen valvonnan sekä mittausten käynnistämisen tai lopettamisen. Mobiilisovellus yhdistettynä GPS-laitteen tarjoamaan aikasynkronointiin helpottaa käyttöä mittaustapahtuman aikana ja sen jälkeen.

Mittausyksiköiden toiminta on testattu useissa erilaisissa haastavissa mittaustilanteissa, jotka on kuvattu kappaleessa 5.2. Näistä mittauksista saatujen tulosten pohjalta on tämän työn ulkopuolella kehitetty ajotapaluokittelija, joka havaitsee erilaiset ajotilanteet moottoripyörän tapauksessa. Tämä luokittelija on toteutettu mittausyksikössä suoritettavaksi käyttäen apuna kappaleessa 4.2.6 kuvattua sovellusrajapintaa. Tämän luokittelijan toteutus on kuvattu lähteessä [21].

Kappaleessa 2 esitetyt tavoitteet on siis täytetty, sillä laite pystyy tuottamaan ja tallentamaan ajallisesti luotettavia ja kalibroituja mittauksia. Laite toimii siis inertiamittausyksikkönä, mutta kuten kappaleessa 5.2 on todettu, laitteessa on mahdollista myös suorittaa monimutkaisia käyttäjäalgoritmeja. Nämä käyttäjäalgoritmit mahdollistavat laitteen toiminnan myös inertianavigointiyksikkönä.

Suurimmat työssä koetut ongelmat liittyvät anturikomponenttien liittämiseen piirilevylle ja komponenttivalintoihin. Piirilevyn pienen koon saavuttamiseksi useat komponenteista ovat LGA- tai QFN-koteloisia, joten niiden liittäminen piirilevylle käsin on aikaavievää työtä. Tämä ongelmakohta voidaan kuitenkin ohittaa komponenttien asettelulaitteella ja reflow-uunilla. Toinen käsinasetteluun liittyvä ongelma on antureiden lopullinen orientaatio, joka saattaa vaihdella mittausyksiköstä toiseen muutaman asteen verran. Tämä näkyy erilaisissa mittaustuloksissa eri yksiköiden välillä, mutta erojen vaikutusta on pyritty vähentämään kappaleessa 5 kuvatulla kalibrointimenetelmällä.

Komponenttivalintoihin liittyen suurimpana ongelmakohtana oli muistikorttistandardin riittämätön tutkiminen, sillä standardi sallii muistikortin viettää 500 ms viivetilassa lohkokirjoitusoperaation jälkeen. Tämän viiveen aikana muistikortti tyhjentää sisäisen muistilohkonsa ja kirjoittaa puskurinsa tähän tilaan. Valittu mikro-ohjain ei sisällä riittävästi muistia, jotta suurikaistaiselle kirjoitukselle pystytään

tekemään 500 ms puskuria. Ratkaisuna ongelmaan on valita mikro-ohjain, joka sisältää ulkoisen muistiväylän johon voidaan liittää ulkoinen SRAM- (Static Random Access Memory) tai SDRAM-piiri (Synchronous Dynamic Random Access Memory). Toinen komponenttivalintoihin liittyvä parannusehdotus on valita mikro-ohjain liukulukuyksiköllä, joka nopeuttaa laitteessa toteutettujen käyttäjäsovellusten suoritusta huomattavasti.

LÄHTEET

[1] STEVAL-MKI062V2, iNEMO demonstration board based on MEMS devices and STM32F103RE. ST Microelectronics. 2010. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user_manual/CD00271225.pdf

[2] HiKoB FOX. HiKoB. 2012. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.hikob.com/hikob-fox

[3] Hikob Openlab [WWW]. [viitattu 25.5.2013] Saatavissa: http://openlab.hikob.com/development-kit

[4] Xsens MTw. Xsens. 2013. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.xsens.com/products/mtw-development-kit-lite/

[5] STM32F103RE ARM Cortex-M3 MCU with 512 Kbytes Flash, 72 MHz CPU. ST Microelectronics. 2011. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.st.com/web/catalog/mmc/FM141/SC1169/SS1031/LN1565/PF164485

[6] LSM303DLHC 3D accelerometer and 3D magnetometer. ST Microelectronics. 2013. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.st.com/web/catalog/sense_power/FM89/SC1449/PF251940

[7] L3GD20 3-axis digital gyroscope. ST Microelectronics. 2013. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.st.com/web/catalog/sense_power/FM89/SC1288/PF252443

[8] LPS331AP MEMS pressure sensor. ST Microelectronics. 2012. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.st.com/web/catalog/sense_power/FM89/SC1316/PF251601

[9] DM3 microSD Card Connectors. Hirose Electric Group. 2010 [WWW]. [viitattu 25.5.2013] Saatavissa: https://www.hirose.co.jp/cataloge_hp/e60900232.pdf

[10] EMIF06-MSD02N166-line EMI filter and ESD protection for T-Flash and micro SD card interfaces. ST Microelectronics. 2008. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.st.com/web/catalog/sense_power/FM139/CL1432/SC1432/PF219457

[11] UC530 GPS antenna module. Ublox. 2013. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.u-blox.com/en/positioning-antennas/gnss-antenna-modules/previous-generations/uc530.html

[12] USBUFxxW6 EMI filter and line termination for USB upstream ports. ST Microelectronics. 2006. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.st.com/web/catalog/sense_power/FM139/CL1432/SC1434/PF65578

[13] RN41/RN41N Class 1 Bluetooth Module. Microchip. 2013. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.microchip.com/wwwproducts/Devices.aspx?product=RN41

[14] LTC3559 Linear USB Battery Charger with Dual Buck Regulators. Linear Technology. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.linear.com/product/LTC3559

[15] GSP062530 Li-Polymer. Great Power Battery Co. 2012. [WWW]. [viitattu 25.5.2013] Saatavissa: https://www.sparkfun.com/products/10718

[16] PH connector. JST. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.jst-mfg.com/product/pdf/eng/ePH.pdf

[17] 1551FL Miniature Plastic Enclosures. Hammond Manufacturing. 2011. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.hammondmfg.com/dwg9FL.htm

[18] VXM Stepping Motor Controller. Velmex. 2013. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.velmex.com/motor_control_vxm.html

[19] B5990TS Motorized Rotary Table. Velmex. 2013. [WWW]. [viitattu 25.5.2013] Saatavissa: http://www.velmex.com/motor_rotary_tables.html

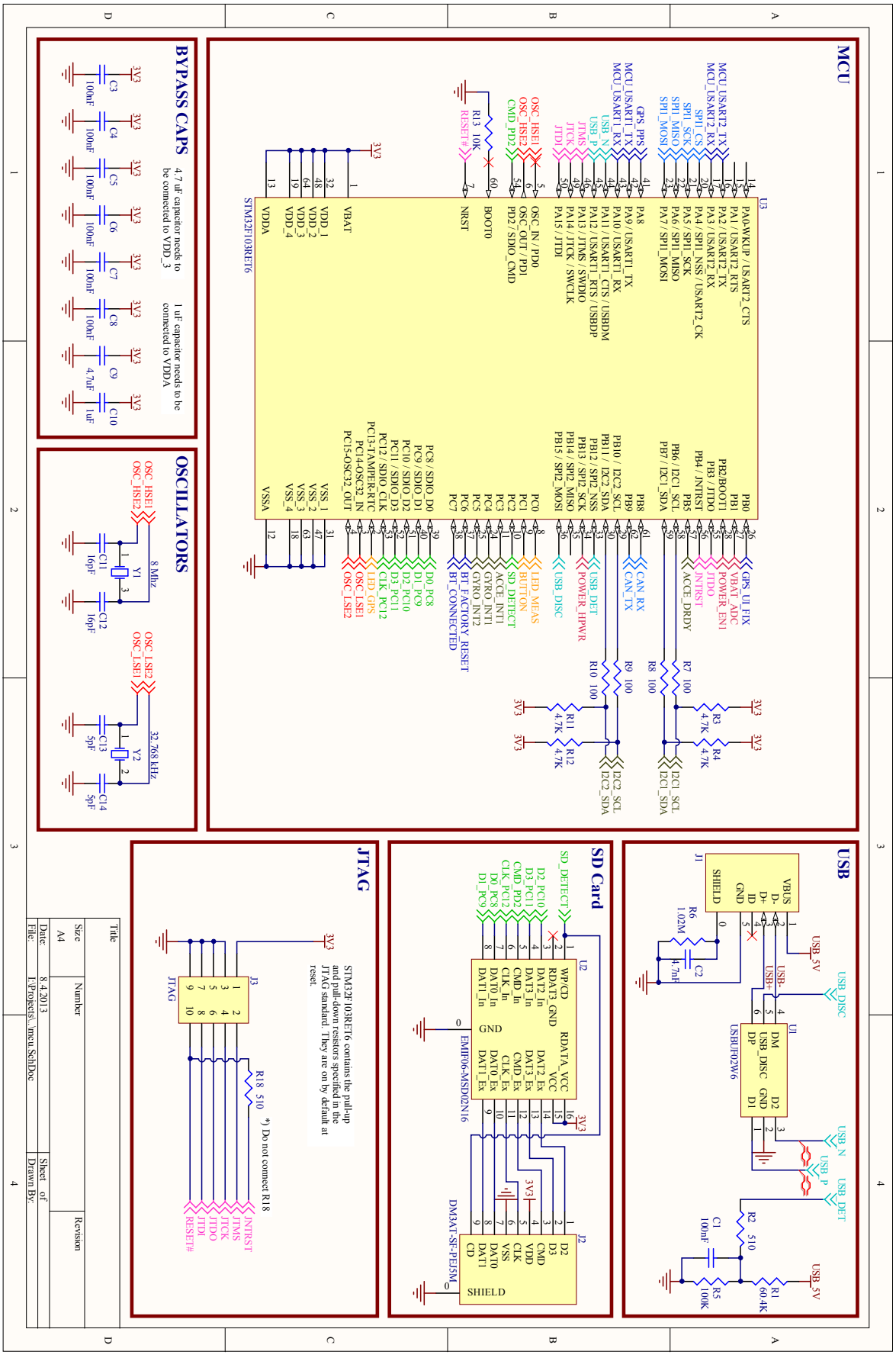
[20] I. Frosio, F. Pedersini, ja N. Alberto Borghese. Autocalibration of MEMS Accelerometers. Instrumentation and Measurement, IEEE Transactions, Volume 58 Issue 6. 2009, s. 2034-2041

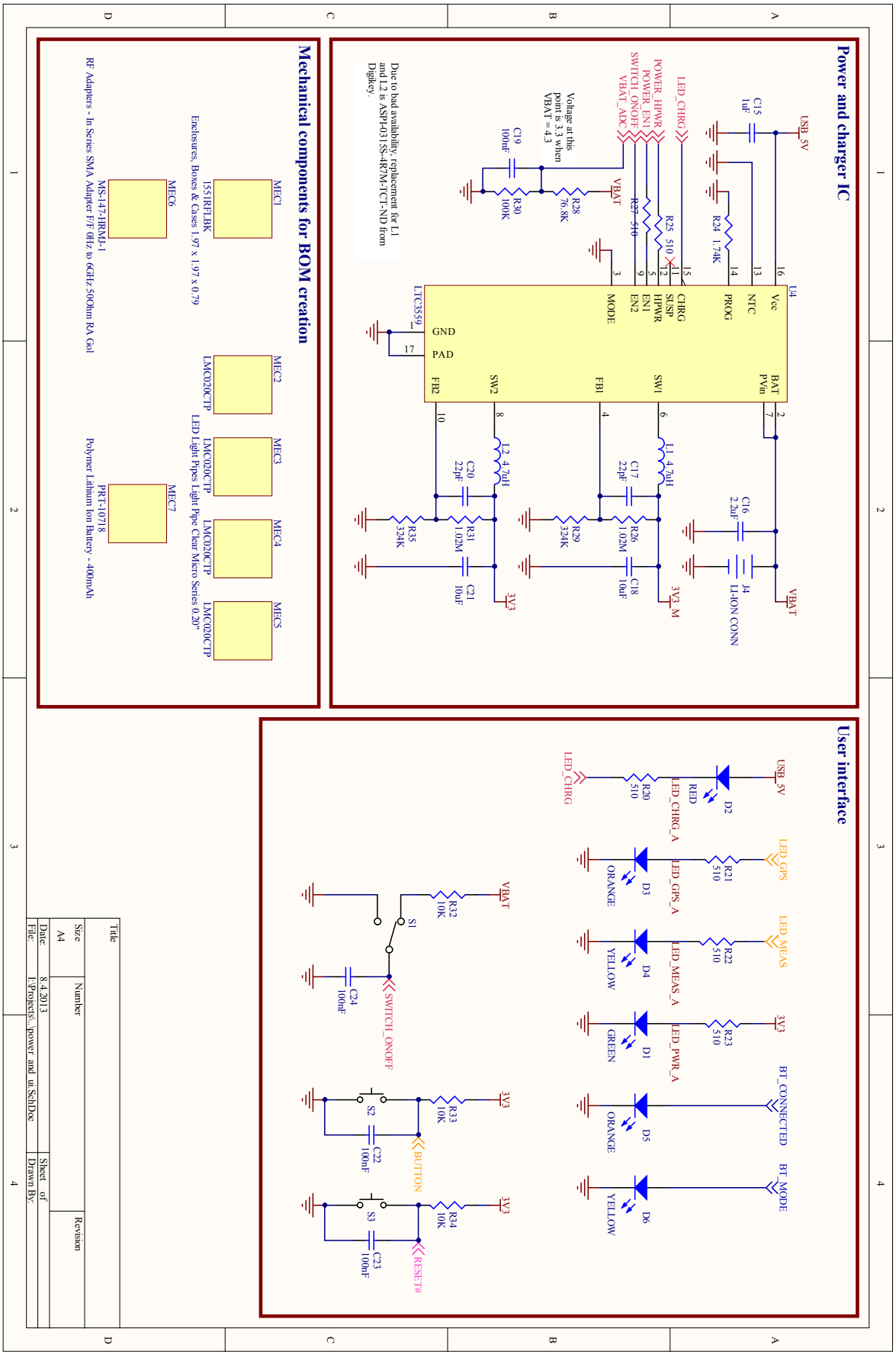
[21] J. Parviainen, J. Collin, T. Pihlström, J. Takala, K. Hanski, A. Lumiaho, Automatic Crash Detection for Motor Cycles. Tampere 2013, Tampereen teknillinen yliopisto. Julkaisematon artikkeli.

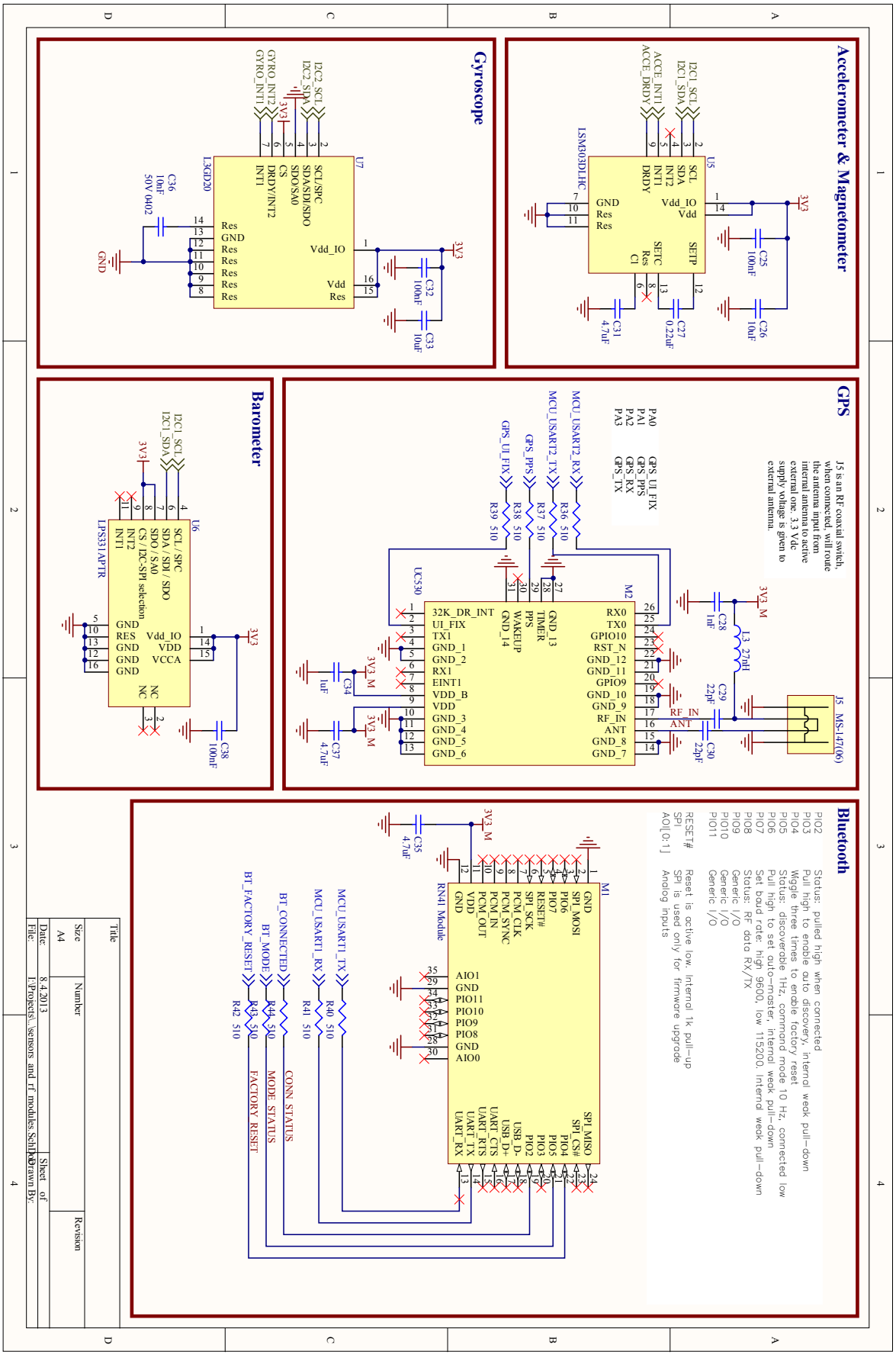
[22] Dale E. Schinstock, GPS-aided INS Solution for OpenPilot [WWW]. [viitattu 5.5.2014] Saatavissa: http://wiki.openpilot.org/display/Doc/INSGPS+Algorithm

A. LIITTEITÄ

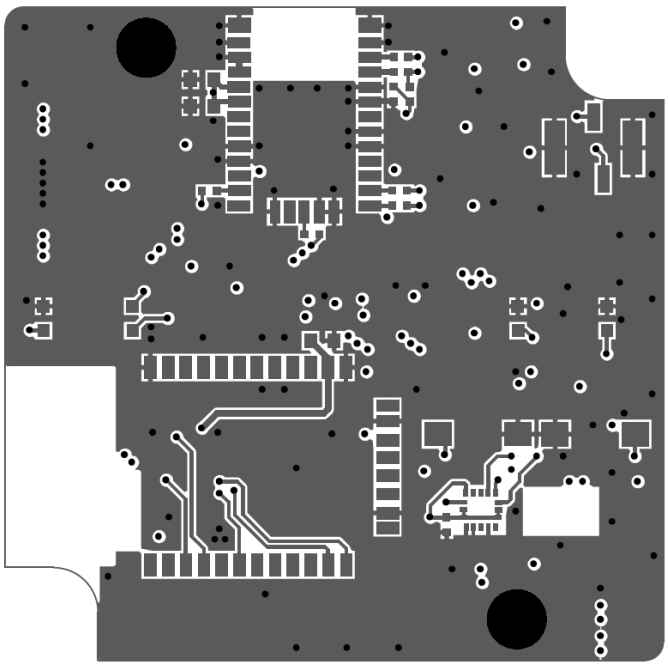
A.1 Kytkentäkaavio


MCU
U3
STM32F103RET6
PA0-WKUP / USART2_CTS
PA1 / USART2_RTS
PA2 / USART2_TX
PA3 / USART2_RX
PA4 / SPI1_NSS / USART2_CK
PA5 / SPI1_SCK
PA6 / SPI1_MISO
PA7 / SPI1_MOSI
PA8
PA9 / USART1_TX
PA10 / USART1_RX
PA11 / USART1_CTS / USBDM
PA12 / USART1_RTS / USBDP
PA13 / JTMS / SWDIO
PA14 / JTCK / SWCLK
PA15 / JTDI
PB0
PB1
PB2/BOOT1
PB3 / JTDO
PB4 / JNTRST
PB5
PB6 / I2C1_SCL
PB7 / I2C1_SDA
PB8
PB9
PB10 / I2C2_SCL
PB11 / I2C2_SDA
PB12 / SPI2_NSS
PB13 / SPI2_SCK
PB14 / SPI2_MISO
PB15 / SPI2_MOSI
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
PC8 / SDIO_D0
PC9 / SDIO_D1
PC10 / SDIO_D2
PC11 / SDIO_D3
PC12 / SDIO_CLK
PC13-TAMPER-RTC
PC14-OSC32_IN
PC15-OSC32_OUT
VBAT
VDD_1
VDD_2
VDD_3
VDD_4
VSSA
VSS_1
VSS_2
VSS_3
VSS_4
OSC_IN / PD0
OSC_OUT / PD1
PD2 / SDIO_CMD
BOOT0
NRST
OSC_HSE1
OSC_HSE2
CMD_PD2
JTMS
JTCK
JTDI
USB_N
USB_P
RESET#
R13 10K
GPS_UI_FIX
VBAT_ADC
POWER_EN1
JTDO
JNTRST
ACCE_DRDY
CAN_RX
CAN_TX
USB_DET
POWER_HPWR
USB_DISC
LED_MEAS
BUTTON
SD_DETECT
ACCE_INT1
GYRO_INT1
GYRO_INT2
BT_FACTORY_RESET
BT_CONNECTED
LED_GPS
OSC_LSE1
OSC_LSE2
R3 4.7K
R4 4.7K
R7 100
R8 100
R9 100
R10 100
R11 4.7K
R12 4.7K
I2C1_SCL
I2C1_SDA
I2C2_SCL
I2C2_SDA
3V3
BYPASS CAPS
4.7 uF capacitor needs to be connected to VDD_3
1 uF capacitor needs to be connected to VDDA
C3 100nF
C4 100nF
C5 100nF
C6 100nF
C7 100nF
C8 100nF
C9 4.7uF
C10 1uF
OSCILLATORS
Y1 8 Mhz
C11 16pF
C12 16pF
Y2 32.768 kHz
C13 5pF
C14 5pF
USB
J1
VBUS
D-
D+
ID
GND
SHIELD
USB_5V
USB_DISC
USB-
USB+
R6 1.02M
C2 4.7nF
U1
USBUF02W6
DM
USB_DISC
DP
D2
GND
D1
USB_N
USB_P
USB_DET
R2 510
C1 100nF
R5 100K
R1 60.4K
SD Card
U2
EMF06-MSD02N16
SD_DETECT
D2_PC10
D3_PC11
CMD_PD2
CLK_PC12
D0_PC8
D1_PC9
WP/CD
RDAT3_GND
DAT2_In
DAT3_In
CMD_In
CLK_In
DAT0_In
DAT1_In
GND
RDATA_VCC
VCC
DAT2_Ex
DAT3_Ex
CMD_Ex
CLK_Ex
DAT0_Ex
DAT1_Ex
J2
DM3AT-SF-PEJ5M
D2
D3
CMD
VDD
CLK
VSS
DAT0
DAT1
CD
SHIELD
JTAG
J3
JTAG
STM32F103RET6 contains the pull-up and pull-down resistors specified in the JTAG standard. They are on by default at reset.
R18 510
*) Do not connect R18
JNTRST
JTMS
JTCK
JTDO
JTDI
RESET#
Title
Size
A4
Number
Revision
Date
8.4.2013
File
I:\Projects\..\mcu.SchDoc
Sheet of
Drawn By:

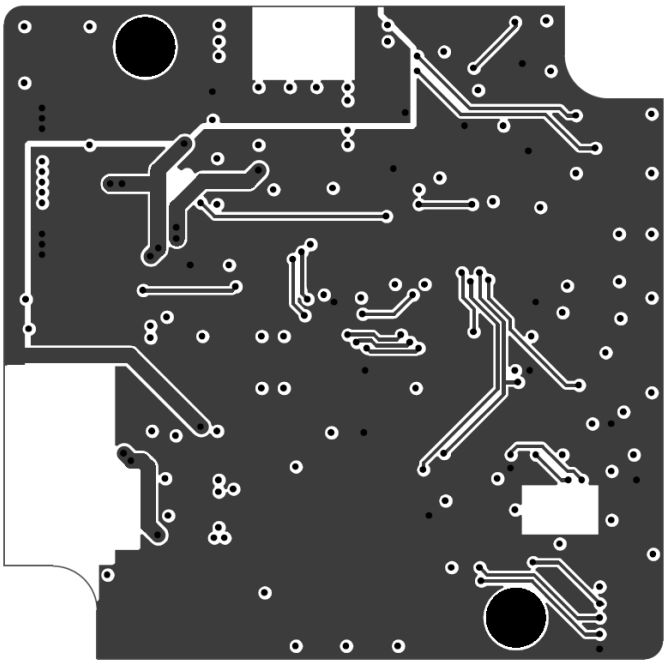

Power and charger IC
USB_5V
C15 1uF
R24 1.74K
LED_CHRG
R25 510
POWER_HPWR
POWER_EN1
SWITCH_ONOFF
VBAT_ADC
R27 510
VBAT
R28 76.8K
R30 100K
C19 100nF
Voltage at this point is 3.3 when VBAT = 4.3
Due to bad availability, replacement for L1 and L2 is ASPI-0315S-4R7M-1CT-ND from Digikey.
U4
LTC3559
Vcc
NTC
PROG
CHRG
SUSP
HPWR
EN1
EN2
MODE
GND
PAD
FB2
SW2
FB1
SW1
BAT
PVin
L1 4.7uH
L2 4.7uH
C16 2.2uF
C17 22pF
C18 10uF
C20 22pF
C21 10uF
R26 1.02M
R29 324K
R31 1.02M
R35 324K
J4
LI-ION CONN
3V3_M
3V3
Mechanical components for BOM creation
MEC1
155JRFLBK
Enclosures, Boxes & Cases 1.97 x 1.97 x 0.79
MEC2
LMO20CTP
LED Light Pipes
MEC3
LMO20CTP
Light Pipe Clear Micro Series 0.20"
MEC4
LMO20CTP
MEC5
LMO20CTP
MEC6
MS-147-HRMJ-1
RF Adapters - In Series SMA Adapter F/F 0Hz to 6GHz 50Ohm RA Gol
MEC7
PRT-10718
Polymer Lithium Ion Battery - 400mAh
User interface
USB_5V
D2
RED
LED_CHRG_A
R20 510
LED_CHRG
LED_GPS
R21 510
D3
ORANGE
LED_GPS_A
LED_MEAS
R22 510
D4
YELLOW
LED_MEAS_A
3V3
R23 510
D1
GREEN
LED_PWR_A
BT_CONNECTED
D5
ORANGE
BT_MODE
D6
YELLOW
VBAT
R32 10K
S1
C24 100nF
SWITCH_ONOFF
S2
R33 10K
C22 100nF
BUTTON
S3
R34 10K
C23 100nF
RESET#
Title
Size
A4
Number
Revision
Date
8.4.2013
Sheet of
File
I:\Projects\...\power and ui.SchDoc
Drawn By:


Accelerometer & Magnetometer
LSM303DLHC
U5
I2C1_SCL
I2C1_SDA
ACCE_INT1
ACCE_DRDY
3V3
C25 100nF
C26 10uF
C27 0.22uF
C31 4.7uF
Gyroscope
L3GD20
U7
I2C2_SCL
I2C2_SDA
GYRO_INT2
GYRO_INT1
C32 100nF
C33 10uF
C36 10nF 50V 0402
GPS
J5 is an RF coaxial switch, when connected, will route the antenna input from internal antenna to active external one. 3.3 Vdc supply voltage is given to external antenna.
MCU_USART2_RX
MCU_USART2_TX
GPS_PPS
GPS_U1_FIX
R36 510
R37 510
R38 510
R39 510
UC530
M2
J5 MS-147(06)
L3 27nH
C28 1nF
C29 22pF
C30 22pF
RF_IN
ANT
C34 1uF
C37 4.7uF
Barometer
LPS331APTR
U6
C38 100nF
Bluetooth
RN41 Module
M1
SPI_MOSI
SPI_CS#
PIO6
PIO7
RESET#
SPI_SCK
PCM_CLK
PCM_SYNC
PCM_IN
PCM_OUT
AIO1
PIO11
PIO10
PIO9
PIO8
AIO0
SPI_MISO
PIO4
PIO5
PIO3
PIO2
USB_D-
USB_D+
UART_CTS
UART_RTS
UART_TX
UART_RX
C35 4.7uF
MCU_USART1_TX
MCU_USART1_RX
R40 510
R41 510
R44 510
R45 510
R42 510
BT_CONNECTED
BT_MODE
BT_FACTORY_RESET
CONN STATUS
MODE STATUS
FACTORY RESET
PIO2 Status: pulled high when connected
PIO3 Pull high to enable auto discovery, internal weak pull-down
PIO4 Wiggle three times to enable factory reset
PIO5 Status: discoverable 1Hz, command mode 10 Hz, connected low
PIO6 Pull high to set auto-master, internal weak pull-down
PIO7 Set baud rate: high 9600, low 115200. Internal weak pull-down
PIO8 Status: RF data RX/TX
PIO9 Generic I/O
PIO10 Generic I/O
PIO11 Generic I/O
RESET# Reset is active low. Internal 1k pull-up
SPI SPI is used only for firmware upgrade
AIO[0:1] Analog inputs
Title
Size A4
Number
Revision
Date 8.4.2013
Sheet of
File: I:\Projects\...\sensors and rf modules.SchDoc
Drawn By:

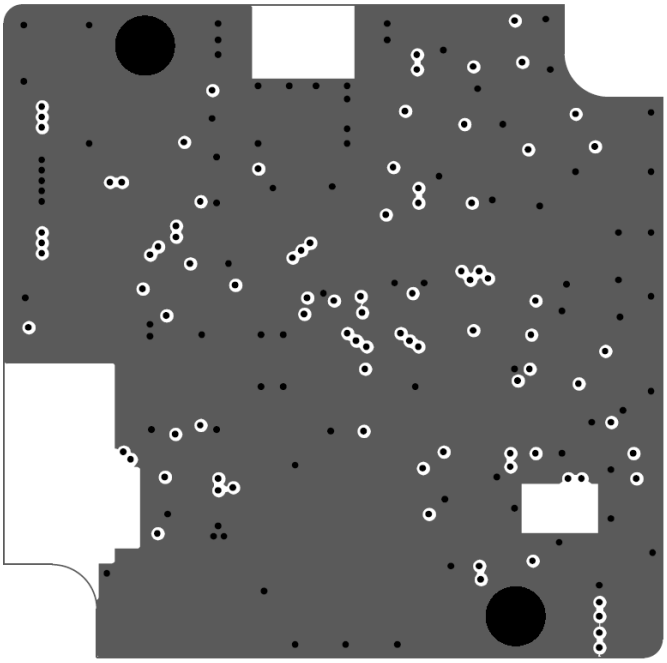
A.2 Osasijoittelukuvat ja kuparitasot



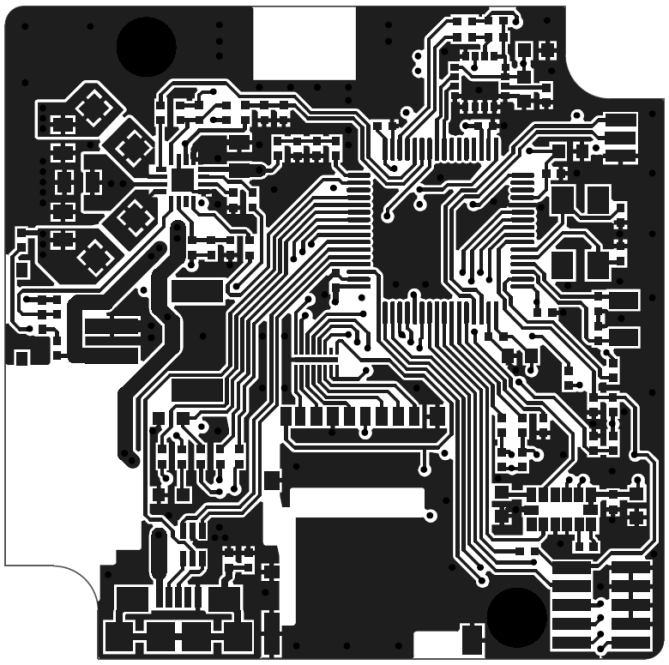
Kuva A.1: Piirilevyn yläkerros (reititys ja maa)



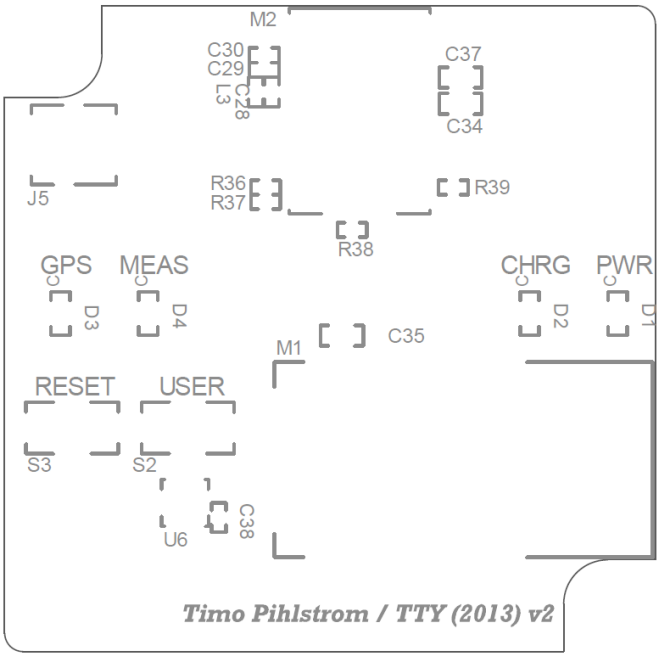
Kuva A.2: Piirilevyn välikerros (käyttöjännitteet ja reititys)



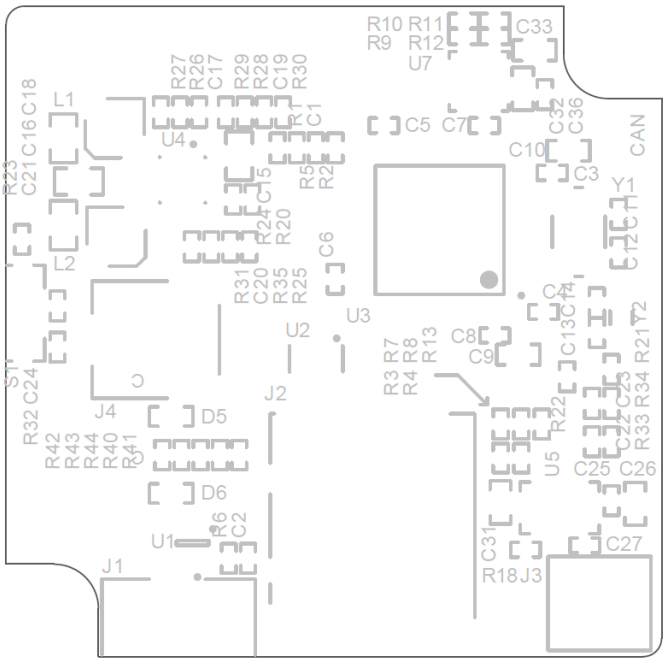
Kuva A.3: Piirilevyn välikerros (maa)



Kuva A.4: Piirilevyn pohjakerros (reititys)



Kuva A.5: Komponenttien sijoitus yläpuolella



Kuva A.6: Komponenttien sijoitus alapuolella

A.3 Komponenttilistaus

Designator	Description	Quantity	Supplier 1	Supplier Part Number 1	Supplier Stock 1
C1, C3, C4, C5, C6, C7, C8, C19, C22, C23, C24, C25, C38	CAP CER 0.1UF 16V 10% X5R 0402	13	Digi-Key	587-1226-1-ND	323916
C10, C34	CAP CER 1UF 16V 10% X5R 0603	2	Digi-Key	445-1416-1-ND	466290
C11, C12	CAP CER 16PF 50V 5% NP0 0402	2	Digi-Key	399-6171-1-ND	23599
C13, C14	CAP CER 5PF 50V NP0 0402	2	Digi-Key	445-4881-1-ND	46638
C15	CAP CER 1UF 25V 10% X5R 0805	1	Digi-Key	587-1291-1-ND	94432
C16	CAP CER 2.2UF 25V 10% X5R 0805	1	Digi-Key	587-1292-1-ND	194165
C17, C20	CAP CER 22PF 50V 5% NP0 0402	2	Digi-Key	587-1203-1-ND	208276
C18, C21	CAP CER 10UF 25V 10% X5R 0805	2	Digi-Key	587-2985-6-ND	336238
C2	CAP CER 4700PF 50V 10% X7R 0402	1	Digi-Key	587-2237-1-ND	20627
C26, C33	CAP CER 10UF 10V 10% X5R 0603	2	Digi-Key	445-7486-1-ND	438403
C27	CAP CER 0.22UF 10V 10% X5R 0402	1	Digi-Key	587-1228-1-ND	433997
C28	CAP CER 1000PF 50V 10% X7R 0402	1	Digi-Key	587-1220-1-ND	111510
C29, C30	CAP CER 22PF 50V 5% NP0 0402	2	Digi-Key	399-6174-1-ND	25327
C32	CAP CER 0.1UF 25V 10% X7R 0603	1	Digi-Key	445-1314-1-ND	1853117
C36	CAP CER 10000PF 50V 10% X7R 0402	1	Digi-Key	587-2238-1-ND	330919
C9, C31, C35, C37	CAP CER 4.7UF 16V 10% X5R 0603	4	Digi-Key	445-7478-1-ND	235760
J2	CONN MICRO SD R/A PUSH-PUSH SMD	1	Digi-Key	HR1964CT-ND	21283
J4	CONN HEADER PH SIDE 2POS 2MM SMD	1	Digi-Key	455-1749-1-ND	71181
L3	INDUCTOR MULTILAYER 27NH 0402	1	Digi-Key	535-11508-1-ND	23728
R1	RES 60.4K OHM 1/16W 1% 0402 SMD	1	Digi-Key	RMCF0402FT60K4CT-ND	22376
R13, R32, R33, R34	RES 10K OHM 1/16W 5% 0402 SMD	4	Digi-Key	RHM10KCECT-ND	63396
R2, R18, R20, R21, R22, R23, R25, R27, R36, R37, R38, R39, R40, R41, R42, R43, R44	RES 510 OHM 1/16W 5% 0402 SMD	17	Digi-Key	RHM510JCT-ND	9670
R24	RES 1.74K OHM 1/16W 1% 0402 SMD	1	Digi-Key	541-1.74KLCT-ND	79169
R28	RES TF 76.8K OHM 1% 1/16W 0402	1	Digi-Key	RMCF0402FT76K8CT-ND	1812
R29, R35	RES 324K OHM 1/16W 1% 0402 SMD	2	Digi-Key	541-324KLCT-ND	56886
R3, R4, R11, R12	RES 4.7K OHM 1/16W 5% 0402 SMD	4	Digi-Key	RHM4.7KCECT-ND	28964
R5, R30	RES 100K OHM 1/16W 1% 0402 SMD	2	Digi-Key	RHM100KBHCT-ND	13011
R6, R26, R31	RES 1.02M OHM 1/16W 1% 0402 SMD	3	Digi-Key	541-1.02MLCT-ND	40931
R7, R8, R9, R10	RES 100 OHM 1/16W 5% 0402 SMD	4	Digi-Key	RHM100CECT-ND	5335
U2	IC EMI FILTER ESD 16-MICRO QFN	1	Digi-Key	497-8751-1-ND	3828
U3	MCU ARM 512KB FLASH MEM 64-LQFP	1	Digi-Key	497-6444-ND	1340
U4	IC USB CHARGER 16-QFN	1	Digi-Key	LTC3559EUD#PBF-ND	663
U6	IC PRESSURE SENSOR PIEZO 16HCLGA	1	Digi-Key	497-12927-1-ND	1593
D1	Standard LED - SMD GREEN WATER CLEAR	1	Mouser	604-APT1608SGC	38233
D2	Standard LED - SMD HI EFF RED WTR CLR	1	Mouser	604-APT1608EC	36850
D3, D5	Standard LED - SMD ORANGE WATER CLEAR	2	Mouser	604-APT1608SECK	1754
D4, D6	Standard LED - SMD YELLOW WATER CLEAR	2	Mouser	604-APT1608YC	204
J1	USB Connectors MICRO B RECEPT RA SMT BTTM MNT	1	Mouser	798-ZX62-B-5PA11	3937
J3	Headers & Wire Housings 5+5 DIL PIN HDR SMT Au/Sn	1	Mouser	855-M50-3600542	4757
J5	RF Connectors REC COAX SWITCHING SMT	1	Mouser	798-MS14706	2174
L1, L2	Fixed Inductors 4.7uH 900mA	2	Mouser	851-CDRH3D16NP4R7NC	4230
M1	MODULE BLUETOOTH W/ANT CLASS1	1	Mouser	765-RN-41	830
M2	GPS Modules Low Pwr GPS Module w/Int Antenna	1	Mouser	916-UC530	295
MEC1	Enclosures, Boxes & Cases 1.97 x 1.97 x 0.79	1	Mouser	546-1551RFLBK	1586
MEC2, MEC3, MEC4, MEC5	LED Light Pipes Light Pipe Clear Micro Series 0.20"	4	Mouser	593-LMC020CTP	6429
MEC6	RF Adapters - In Series SMA Adapter F/F 0Hz to 6GHz 50Ohm RA Gol	1	Mouser	798-MS-147-HRMJ-1	17
S1	Slide Switches slide switch without location lug	1	Mouser	688-SSAJ120100	4360
S2, S3	Tactile & Jog Switches 6.0x3.5x5mm 260gF	2	Mouser	688-SKQMAR	7120
U1	EMI Filters EMI FILTER IPAD	1	Mouser	511-USBUF02W6	5119
U5	Board Mount Accelerometers E-compass 3D accel & 3D magnetometer mod.	1	Mouser	511-LSM303DLHC	5476
U7	Gyroscopes MEMS 3-Axis GRYO 16-Bit Data 1.8V	1	Mouser	511-L3GD20	2608
Y1	CRYSTAL 8.000 MHZ 18PF SMD	1	Mouser	815-ABMM2-8-E2T	3990
Y2	CRYSTAL 32.768KHZ 7PF SMD	1	Mouser	815-ABS07-32.768K7T	9517
MEC7	Polymer Lithium Ion Battery - 400mAh	1	Sparkfun	PRT-10718	