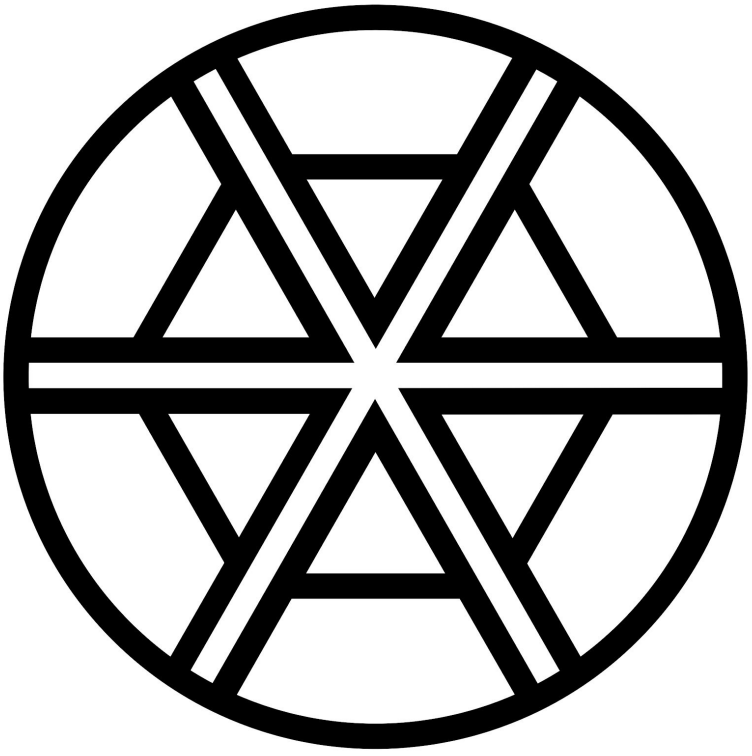
Aalto-yliopisto, Sähkötekniikan korkeakoulu
ELEC-D0301 Protopaja
2019

Loppuraportti

Projekti #2

Real-Time Data-Analyzing for Sports



Päiväys: 30.8.2019

Sampo Vänninen
Tiia-Maria Hyvönen
Olavi Uusitalo
Samuel Johansson

Tietosivu

Opiskelijat
Sampo Vänninen
Tiia-Maria Hyvönen
Olavi Uusitalo
Samuel Johansson

Projektipäällikkö
Tiia-Maria Hyvönen

Sponsoroiva yritys
Aboense Oy

Aloituspäivä
3.6.2019

Palautettu
30.8.2019

Tiivistelmä

Projektimme aiheena on lukea usealta urheilijalta hengitys- ja sydämen sykedataa ja esittää tämä data sekä urheilijan paikan sijainti nettisivun tai sovelluksen avulla reaaliajassa. Teemme projektia Aboense:lle ja sitä työstetään pääsääntöisesti sisäpelien pelaajia varten kuten esimerkiksi salibandyn. Jokaiselle pelaajalle kiinnitetään laite joka sisältää mm. Aboensen anturin joka mittaa sykettä ja hengitystä, lisäksi bluetoothin avulla saadaan tieto paikasta. Laite lähettää datan langattomasti palvelimelle, joka esittää sen visuaalisesti sovelluksen kautta reaaliajassa.

Abstract

The subject of our project is to read breath data and heart rate data from several athletes and display this data and the location of the athletes through a web page or application in real time. We are working on a project for Aboense and it is mainly being worked on for indoor sports such as floorball players. Each player is fitted with a device which includes eg. an Aboense’s sensor that measures heart rate and breathing, plus Bluetooth provides location information. The device wirelessly sends data to a server, which visually displays it through the application in real time.

Sisällysluettelo

1. Johdanto

Aboense on kehittänyt anturin, joka mittaa urheilijan sydämen sykettä ja hengitystiheyttä. Projektin tarkoitus on lukea näistä antureista dataa usealta urheilijalta samaan aikaan sekä ylläpitää tietoa urheilijoiden paikasta. Paikan, hengityksen ja sykkeen data esitetään reaaliajassa nettisivun kautta, johon se lähetetään langattomasti.

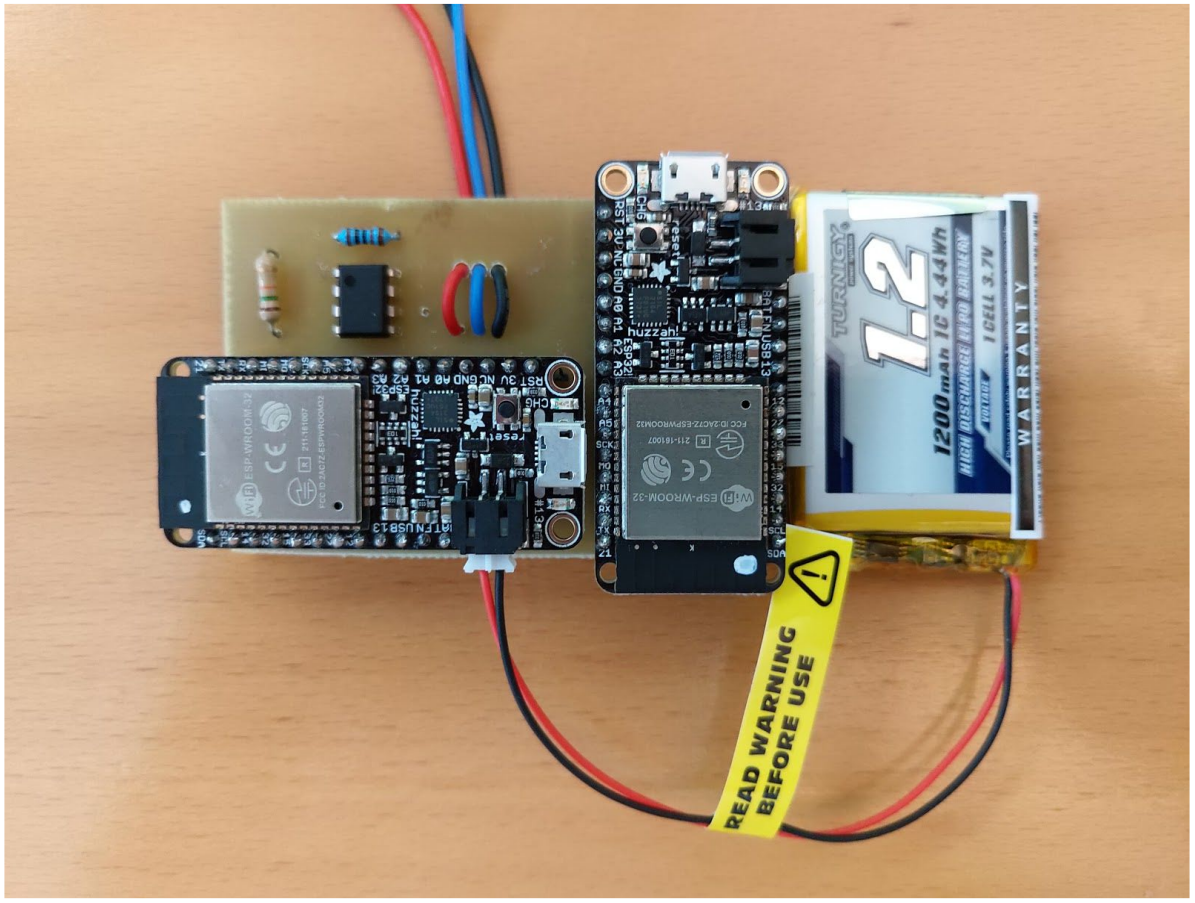
2. Tavoite

Ideaalissa lopputuloksessa voimme demonstroida kuinka usealla henkilöllä on kiinnitettynä anturi, ja näemme suoraan ruudulta jokaisen henkilön sykkeen, hengitystiheyden ja paikan reaaliajassa sekä hyvällä tarkkuudella. Tärkeintä onnistumisen kannalta olisi se että hengitykseen ja sykkeeseen liittyvä data saataisiin näkyviin selkeästi, samaan ikkunaan, reaaliajassa ja mahdollisimman tarkasti. Paikan esitys on myös hyvä tavoite mutta tulee vasta edellä mainittujen jälkeen. Data lähetetään Wi-Fi:n välityksellä langattomasti. Tällainen projekti toimii sisäurheilulajien, kuten salibandyn pelaajien seuraamiseen ja on mielenkiintoinen lisä urheilusuoritusten tutkimisessa.

3. Tulokset A - Mikrokontrollerit ja sensorit

3.1. Saavutettu toiminta

Valmiissa versiossa piirilevyyn on kiinnitettynä kaksi Adafruitin ESP32-Feather -mikrokontrolleria, joista toinen toimii Bluetooth-antennina ja toinen Wi-Fi -lähettimenä. Päädyimme käyttämään kahta mikrokontrolleria helppokäyttöisyyden ja yhdistettävyyden takia: Koodia oli helppo testata USB-portin kautta ja kontrollerit oli helppo liittää toisiinsa valmiiden pinnien avulla. Virransyöttö oli myös helppo toteuttaa Feathereissa mukana olevalla akkuliitännällä, ja virran syöttö toiselle mikrokontrollerille onnistui piirilevyn kautta.



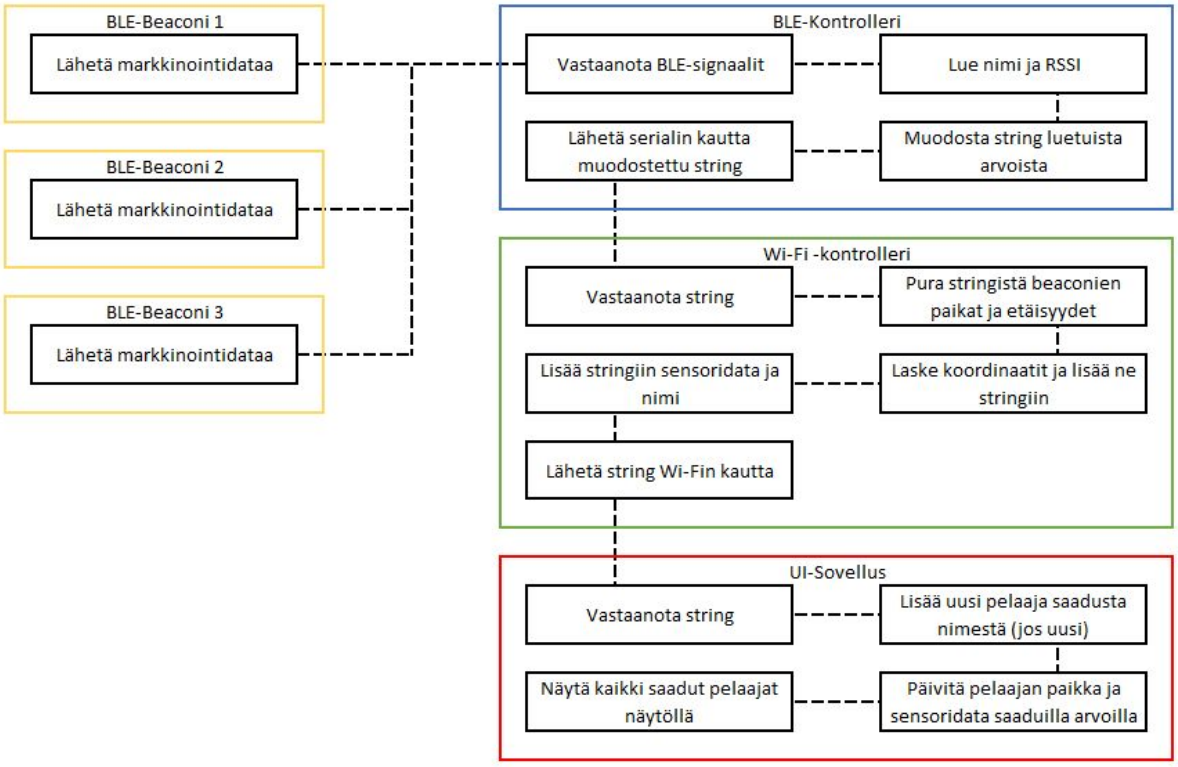
Kuva 1. Mikrokontrollerit piirilevyllä.

Beaconit olivat kohtuullisen yksinkertaisia toiminnaltaan. Mikrokontrollerina niissä toimi ESP32-WROOM sisäänrakennetun U.FL -liitännän takia, jolla saimme tehokkaamman antennin kiinnitettyä helposti. Päädyimme käyttämään beaconeiden virtalähteenä halpoja USB-akkupankkeja niiden helppokäyttöisyyden, saatavuuden, suuren kapasiteetin sekä hinnan takia. Beaconeiden kuoreksi saimme Aboenseltä 3D-mallit, joista tulostimme sekä laserleikkasimme kopan.

Kuva 2. BLE-beacon kopassaan.

Prototyypin toiminnan vaiheet ovat seuraavanlaiset:

1. Beaconit lähettävät signaalia
2. Bluetooth-kontrolleri vastaanottaa signaalin, lukee signaalista beaconin nimen sekä signaalinvoimakkuuden
3. BLE-kontrolleri pakkaa jokaisen löytämänsä beaconin stringiin muodossa #Nimi/RSSI#Nimi/RSSI...
4. BLE-kontrolleri lähettää Wi-Fi-kontrollerille hardware serialin kautta näin muodostetun stringin
5. Wi-Fi-kontrolleri vastaanottaa stringin, lukee siitä nimi-voimakkuusparit, laskee näistä X- ja Y-koordinaatit sekä lisää sensoreista tulevan datan.
6. Lähettää Wi-Fi:n kautta annettuun IP-osoitteeseen pakatun datan, jossa kontrollerin nimi, koordinaatit sekä sensoridata.



Kuva 3. Kaavio prototyypin toiminnasta.

Perusidea on toimiva ja toimii halutusti, mutta projekti ei vastaa suunnitelmaa muutaman suuren ongelman takia:

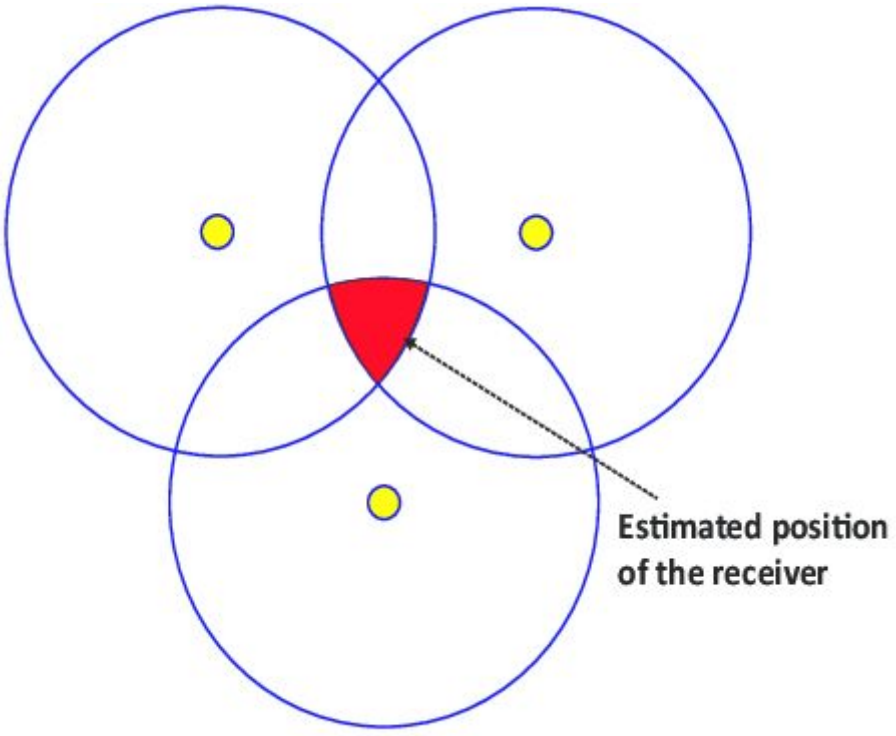
1. BLE-Signaali heijastuu arvaamattomasti eri huoneissa eri tavalla, muodostaen sitä lukiessa tasaisesti kasvattaen etäisyyttä eräänlaisen sinimuotoisten käyrän RSSI-signaalin arvoista, joka aiheuttaa koodissa sen, että esimerkiksi 3 metrin ja 10 metrin kohdalla RSSI-arvo on sama, mutta muutosfunktio on yksikäsitteinen, johtaen luetun etäisyyden heittävän 7 metrillä funktion palauttaessa etäisyydeksi 3 metriä molemmilla arvoilla.
2. Edellistä vielä enemmän haitaten BLE-signaalin voimakkuus riippuu suuresti antennin asennosta sekä vastaanottimen ja beaconin välissä olevista kappaleista, joka aiheuttaa liikkessä olevan kontrollerin lukeman dataan hyppimisen täysin ennalta-arvaamattomasti.
3. Sensoreista tuleva data on vaikeaselkoista, eikä sitä olla pystytty prosessoimaan, joten Wi-Fi-kontrollerin lähettämä sensoridata on koodissa lukittu vakioksi.

3.2. Paikannuskoodin toiminta

Trilateraatioon käyttämämme koodi löytyi valmiina julkisesta GitHubista Navigine-yrityksen tekemänä. Se käyttää pienimmän neliösumman menetelmää laskemaan annetuista majakkapisteistä ja etäisyyksistä näihin pisteisiin kaikkein todennäköisimmät koordinaatit. Tämä vaati siis kahta eri asiaa.

1. Majakoiden paikan tietäminen
2. Etäisyydet majakoihin

Majakoiden paikat on prototyypissä koodattuna suoraan Wi-Fi-kontrollerin koodissa, joka tarkoittaa kontrollerin kalibrointia esimerkiksi muuttaessan toimintaympäristöä.
Etäisyydet oli tarkoitus saada BLE-signaalin voimakkuudesta, mutta kuten kävi ilmi, signaalinvoimakkuus on hyvin epäluotettava tapa tehdä tämä, joka aiheuttaa neliösummamenetelmän näyttämään lähes aina suurinpiirtein beaconeiden keskipistettä, jos yksikään beaconeiden lukemista eroaa merkittävästi “oikeasta” lukemasta.



Kuva 4. Trilateraatiokoodin toiminta ideaalitapauksessa.

3.3. Kommunikaatiokoodin toiminta

3.3.1. Wi-Fi -kommunikaatio

Mikrokontrolleri ja käyttöliittymä kommunikoivat wifi:n kautta. Käyttoliittymä luo pistokkeen python socket -luokkaa käyttäen ja avaa portin käytettäväksi. Mikrokontrolleri ottaa yhteyden wifiin ja wifin kautta paikalliseen, käyttöliittymän koneen, ip-osoitteeseen. Mikrokontrolleri lähettää dataa tietyin väliajoin ja käyttöliittymän pistoke vastaanottaa sen.

3.3.2. Serial-väylä

Mikrokontrollereiden välinen kommunikatio tehtiin Arduinon valmiilla Serial-kirjastolla, jossa kahden kontrollerin RX (Receive) ja TX (Send) -pinnit yhdistetään ristiin. Piirilevy yhdistää Feathereiden pinnit, jolloinka datan lähettäminen on helppoa: Tarvitsee ainoastaan kirjoittaa serialiin, ja vastaanottava kontrolleri saa datan. Lähetysnopeutena kontrollerien välillä käytimme kohtuullisen hidasta 9600 baudia datan pienen määrän sekä varman toiminnan takia.

3.3.3. BLE-antenni

Bluetooth-signaalin lukeminen tehdään aktiivisella skannauksella, joka kuluttaa huomattavasti enemmän virtaa kuin passiivinen skannaus (huomaa esimerkiksi BLE-kontrollerin huomattavasta lämpenemisestä. Tämä kuitenkin parantaa signaalin luettavuutta huomattavasti, ja testeissä maksimietäisyys oli noin kolminkertainen verrattuna passiiviseen skannaamiseen. Käytetyssä arduinon BLE-kirjastossa minimiskannausaika oli 1000ms, josta tulee prototyypin paikkadatan päivitystaajuus, hiukan alle 1 Hz prosessointiaika mukaanluettuna.

3.4. Beaconeista

Beaconit olivat yksi projektin helpoimmista osista. Koodin osalta Arduino ESP32-paketista löytyy valmiina lähes suoraan esimerkki luoda uusi advertising-dataa lähettävä Beaconi, jota muokkaamalla saatiin haluttu toiminta, ja fyysisen beaconin kokoaminen ei vaatinut muuta kuin antennin ja akkupankin kiinnittämisen kontolleriin. Ulkokuorien tulostamiseen kului eniten aikaa, lähemmäs 20 tuntia per kuori, mutta tämä ei vaikuttanut beaconeiden toimintaan, joten emme tulostaneet kahta enempää.

Luodessa uutta beaconia koodissa täytyy vaihtaa yksi muuttuja createBeacon-funktiossa

AdvertisementData.setName("Beacon 3"); //Configures the beacon name

Bluetooth-koodi etsii ainoastaan beaconeita, joiden nimessä löytyy “Beacon”, joten beaconille annetun nimen on oltava muotoa “Beacon x”.

3.5. Sensorit

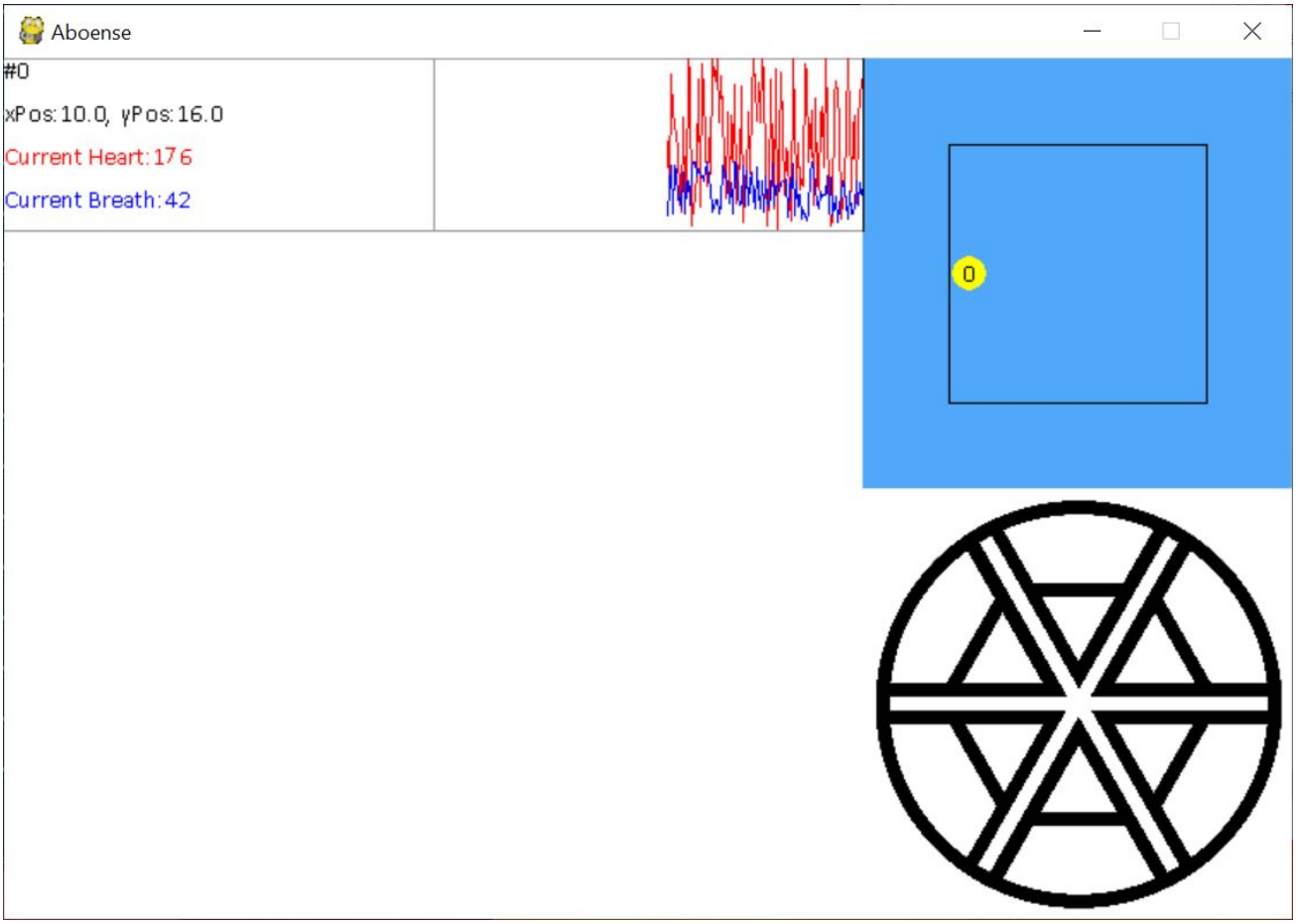
Projektissa käytetyt sensorit saimme Aboenselta valmiina. Kävi kuitenkin ilmi, että sensorien antama data oli kohtuullisen vaikeaselkoista varsinkin sydämen pulssin osalta, jonka parissa kului huomattavasti enemmän aikaa kuin oletimme. Lopulta myös hengitysdata oli sen verran kaoottista, että emme saaneet edes toista osaa sensorien antamasta signaalista prosessoitua.

4. Tulokset B - Käyttöliittymä

Käyttöliittymän toteutus onnistui ja käyttöliittymä saa dataa mikrokontrollerilta Wi-Fi -yhteyden kautta. Sisäisesti käyttöliittymällä on päätiedosto main.py, kaksi luokkaa player ja drawPlayer sekä data_parse datan jäsennykseen. Päätiedosto luo Wi-Fi -pistokkeen, ylläpitää pelaajien datalistaa ja kutsuu piirtokomentoja. Player -luokka on datan säilytysluokka. DrawPlayer piirtää player luokan datan ruudulle käyttäen pygamea.

Alkuperäisen suunnitelman mukaan meidän oli tarkoitus hyödyntää Amazon Web Services -pilvipalvelua käyttöliittymän toteuttamiseen, mutta projektin edetessä huomasimme sen käytön olevan liian työlästä ilman aiempaa kokemusta sen tarjoamista palveluista. Toteutimme lopulta käyttöliittymän Pythonilla PyGame nimisen grafiikkakirjaston avulla. PyGame on tarkoitettu varsinkin pelien ohjelmointiin, mutta se soveltui erittäin hyvin myös tähän projektiin. Toteutus oli lopulta paljon suoraviivaisempi kuin AWS:llä ja onnistuimme UI:n kannalta kaikessa suunnitellussa.

Käyttöliittymän oikeassa yläkulmassa on pelikentän muotoinen alue, mihin pelaajia kuvaavat keltaiset pisteet ilmestyvät kun heissä kiinni olevat laitteet saavat yhteyden muodostettua käyttöliittymään. Kentän kulmissa on Bluetooth -beaconit ja pelaajat (keltaiset pisteet) liikkuvat kentällä majakoiden mittaamassa/laskemassa kohdassa. Kentän alapuolella on sponsori yrityksen logo. Vasemmalle puolelle ilmestyy suorakulmion muotoisia laatikoita 0- 5 kappaletta riippuen siitä, että kuinka monta pelaajaa on yhdistettynä käyttöliittymään. Jokaisesta yhdistetystä pelaajasta tulee esille nimi, pelinumero, paikan koordinaatit, sydämen syke ja hengitystiheys. Lisäksi näiden tietojen oikealla puolella näkyy punaisella sydämen sykkeen ja sinisellä hengitystiheyden piirtämää käyrää.



Kuva 5. Käyttöliittymä.

5. Tulokset C - Komponentit ja piirilevy

5.1. Komponentit

Komponentti	Määrä	Käyttötarkoitus
ESP32-DevKitC-32U	10	BLE-beaconin mikrokontrolleri
Linx ANT-2.4-PML-UFL	10	2.4 GHz BLE-beaconin antenni
Clas Ohlson 3350 mAh	10	virtalähde BLE-beaconille
Sparkfun CAB-12970	1	elektrodikaapeli anturille
Aboense anturi	1	käsittelemämme raakadata
Adafruit HUZZAH32	2	piirilevyn mikrokontrolleri
TLV2372	1	operaatiovahvistimet piirilevyllä
pintaliitosvastus 0805	4	piirilevyn kytkennät
pintaliitosvastus 0603	1	piirilevyn kytkennät
läpireikävastus axial	2	piirilevyn kytkennät
pintaliitoskondensaattori 0805	6	piirilevyn kytkennät
Turnigy 1200 mAh LiPoly	1	virtalähde piirilevylle

Listattuna ovat kaikkien kymmenen beaconin komponentit. Muilta osin komponenttien määräksi on merkitty yhdelle piirilevylle/pelaajalle tulevat komponentit.

5.1.1. BLE-beaconin komponentit

Bluetooth Low Energy beacon koostuu ESP32-DevKitC-32U mikrokontrollerista, Linxin ANT-2.4-PML-UFL antennista sekä Clas Ohlsonin 3350 mAh varavirtalähteestä. Mikrokontrolleri saa virtansa Micro-USB -liitännän kautta uudelleen ladattavalta varavirtalähteeltä. Antennille on liitäntä mikrokontrollerissa ja se on helppo kiinnittää ja irrottaa.

Aboensen Hugo suunnitteli beaconille koteloinnin, jonka valmistimme pajalla. Katso Kuva 2.

5.1.2 Piirilevyn komponentit

Valitsimme piirilevyn mikrokontrolleriksi Adafruitin HUZZAH32:den ensisijaisesti sen Wi-Fi- ja Bluetooth-ominaisuuksien vuoksi. Se on näppärän kokoinen. Mikrokontrollerissa on myös latauspiiri ja liitäntä LiPo-akulle. LiPo-akku on ladattavissa mikrokontrollerin mikro-usb-liitännän kautta. Käytämme Turnigyn 1200 mAh LiPo-akkua, jonka kapasiteetin pitäisi riittää ainakin kolmen tunnin käyttöön.

Adafruitin HUZZAH32 mikrokontrollereja piirilevyllä on kaksi, joista toinen ottaa vastaan anturin antaman signaalin ja on yhteydessä Wi-Fiin sekä hoitaa kaikki laskutoimitukset, kuten sydän- ja hengityssignaalin lukuarvon laskun ja paikan määrittämisen. Se jakaa myös LiPo-akun virtaa toiselle mikrokontrollerille, joka kuuntelee BLE-beaconien signaalia ja jakaa sen eteenpäin ensimmäiselle mikrokontrollerille käsiteltäväksi.

Aboensen anturi on kiinnitettynä piirilevylle Sparkfunin CAB-12970 kolmipäisellä elektrodikaapelilla.

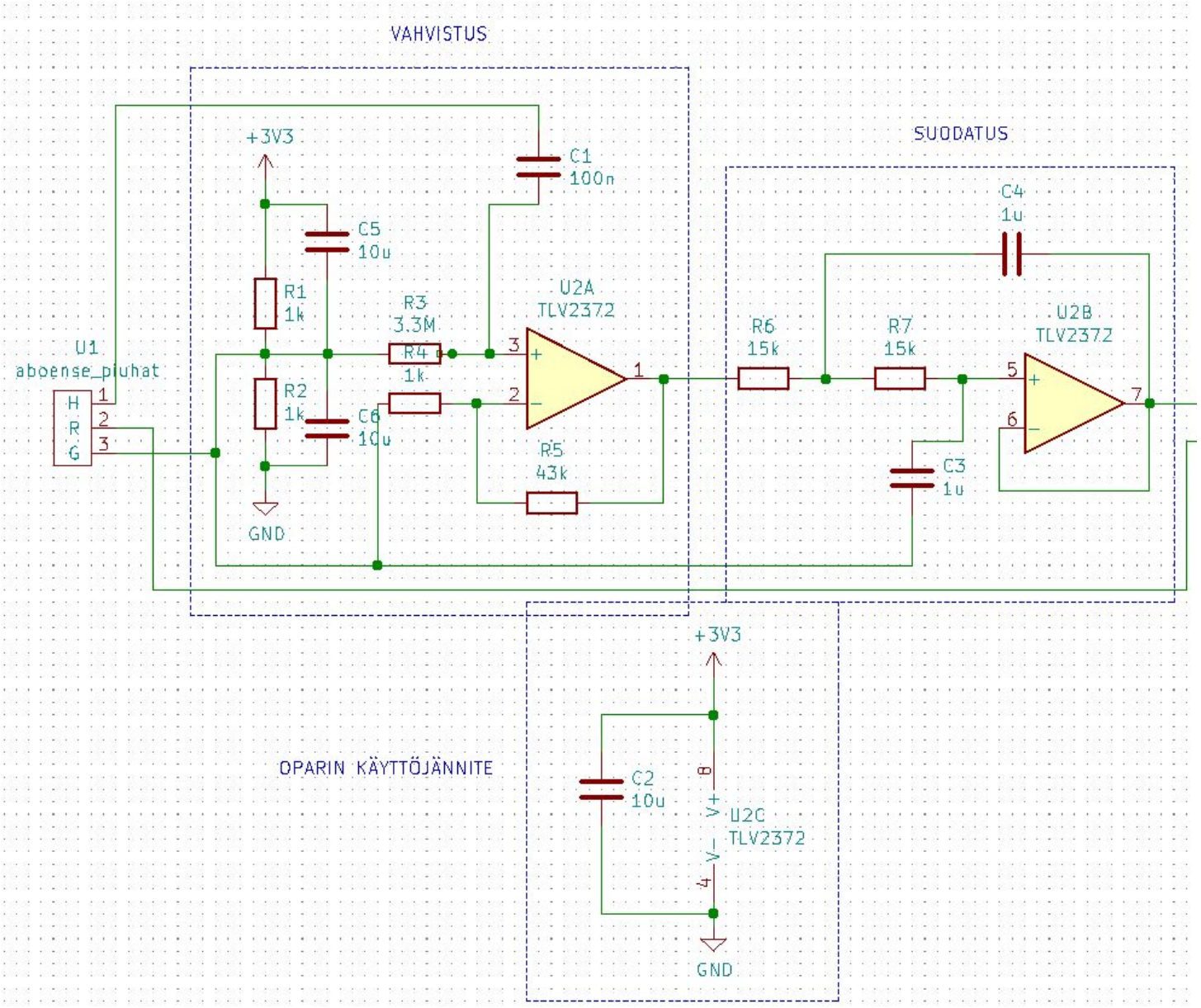
TLV2372 operaatiovahvistinta käytetään sydänsignaalin vahvistukseen ja suodatukseen.

Piirilevyllä kondensaattoreina ja vastuksina on käytetty ensisijaisesti pintaliitoskomponentteja niiden pienen koon vuoksi. Vastuksista kaksi on läpireikämallisia, jotta kupariyhteyksiä saatiin vedettyä niiden alta. Yksi pintaliitosvastuksista on erityisen pieni (0603), koska tarvittavaa vastusta ei ollut pajalla suurempana.

5.2 Piirilevy

Valmistimme yhteensä kolme piirilevyä, joista ensimmäisessä ei ollut signaalien suodatusta tai virtuaalimaata ja signaalin laatu oli huonoa. Jatkossa puhutaan vain lopullisesta piirilevystä ja sen kytkennästä.

Aboensen anturilla on yksi sisääntulo piirilevyn groundista ja ulostulot sydän- ja hengityssignaaleille. Hengitys on kytketty suoraan mikrokontrollerin analogipinniin A4. Sydänsignaali vahvistetaan ensin 44-kertaiseksi, jonka jälkeen se viedään alipäästösuodattimen läpi (noin 10 Hz rajataajuus) mikrokontrollerin analogipinniin A2. Teimme myös toisen piirilevyn samalla kytkennällä testausta varten, jonka alipäästösuodattimen rajataajuus on noin 30 Hz. Tämä tehtiin pienentämällä alipäästösuodattimen vastusten arvoja (Kuva 7, R6 ja R7). Testeissä tämä pienempi suodatus ei tuottanut parempia tuloksia.



Kuva 6. Kytkentäkaavio.

Yllä olevasta kytkentäkaaviosta on jätetty pois mikrokontrollereiden kytkennät selkeyden vuoksi. +3v3- ja GND-liput on vedetty mikrokontrollerin vastaavista pinneistä, jonka lisäksi mikrokontrollereiden välillä on veto BAT-pinnien välillä virransyöttöä varten sekä kaksi vetoa TX-RX pinnien väleillä mikrokontrollereiden keskinäistä tiedonsiirtoa varten.

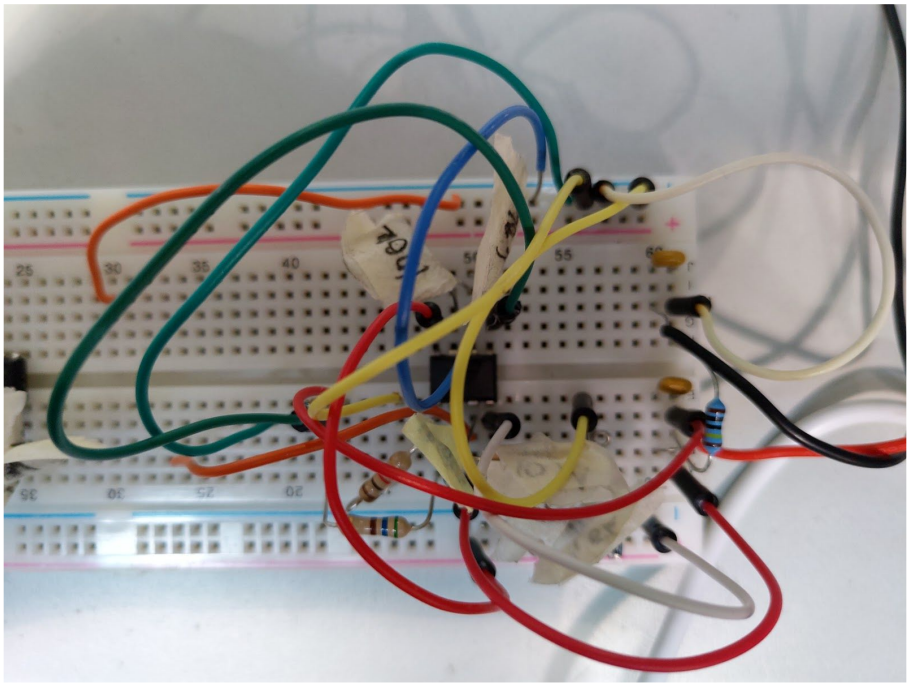
5.2.1 Suunnittelu

Tämä kytkentä on paras yrityksemme anturin signaalin lukukelpoiseksi saamiseksi. Aikaa toteutukselle jäi vähän, koska signaalin käsittelyn vaativuus yllätti meidät. Ajatus kuvan 6 mukaisesta vahvistuskytkennästä tuli Scott Hardenin DIY ECG-projektista, jossa Scott sai sykekäyränsä näkyviin yksinkertaisella vahvistuksella. Erona meidän projektiimme on hänen käyttämänsä 3 elektrodia käyttämämme kahden sijaan ja suodatuksen hoitaminen koodilla. TLV2372 sisältää kaksi operaatiovahvistinta, joista toista käytetään ensin sydänsignaalin vahvistamiseen ja toista vahvistetun signaalin suodattamiseen.

Vahvistuskytkennässä groundina käytetään virtuaalimaata, joka saadaan aikaiseksi vastusten R1 ja R2 muodostamalla jännitteen jakajalla. Virtuaalimaa nostaa jännitteen referenssitason 3.3 V:n käyttöjännitteen puoleenväliin. Tämä on tarpeellista, koska sykepulssissa on piikki referenssitason kummallekin puolelle. Varsinaisen vahvistuksen määrittävät vastukset R4 ja R5 kaavan $V_{out} = V_{in}(1 + R5/R4)$ mukaan. Meidän piirilevymme vahvistus on siis 44.

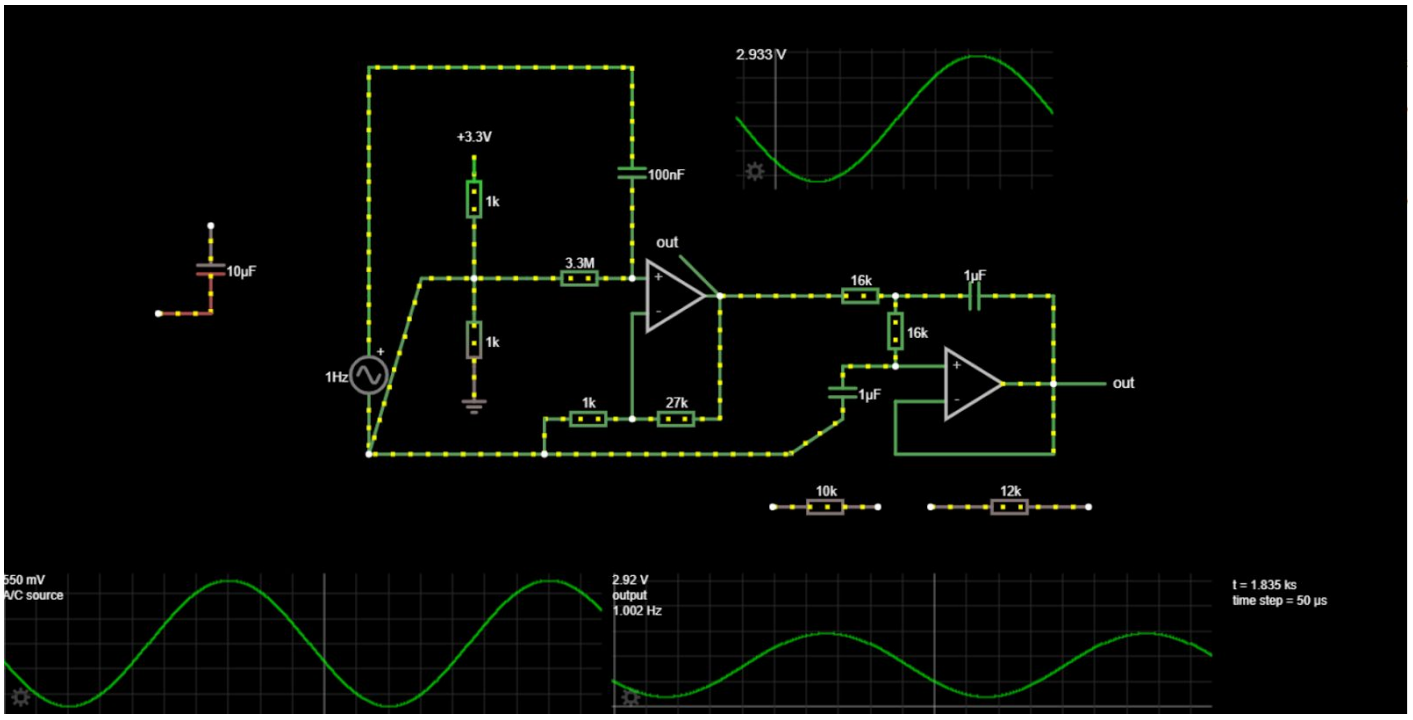
Signaalin suodattamista varten meillä on toisen asteen alipäästösuodatin, jonka rajataajuus on noin 10 Hz. Alipäästösuodatin toimii päästämällä rajataajuutta pienemmät taajuudet läpi vaimentaen korkeampia taajuuksia, joista signaalin häiriöt pääosin koostuvat. Sykkeen taajuus vaihtelee välillä 1 - 4 Hz. Lisäsimme kondensaattorit C2, C5 ja C6 suodattamaan käyttöjännitettä ja kondensaattorin C1 suodattamaan sydänsignaalia ennen vahvistusta.

Testasimme kytkentöjä leipälaudalla saamatta selkeitä tuloksia. Jäljellä olevan vähäisen ajan vuoksi siirryimme kuitenkin eteenpäin toivoen, että leipälautakytkentöjen keräämät ylimääräiset häiriöt signaalissa vähenisivät piirilevyllä.



Kuva 7. Sydänsignaalin testikytkentöjä leipälaudalla.

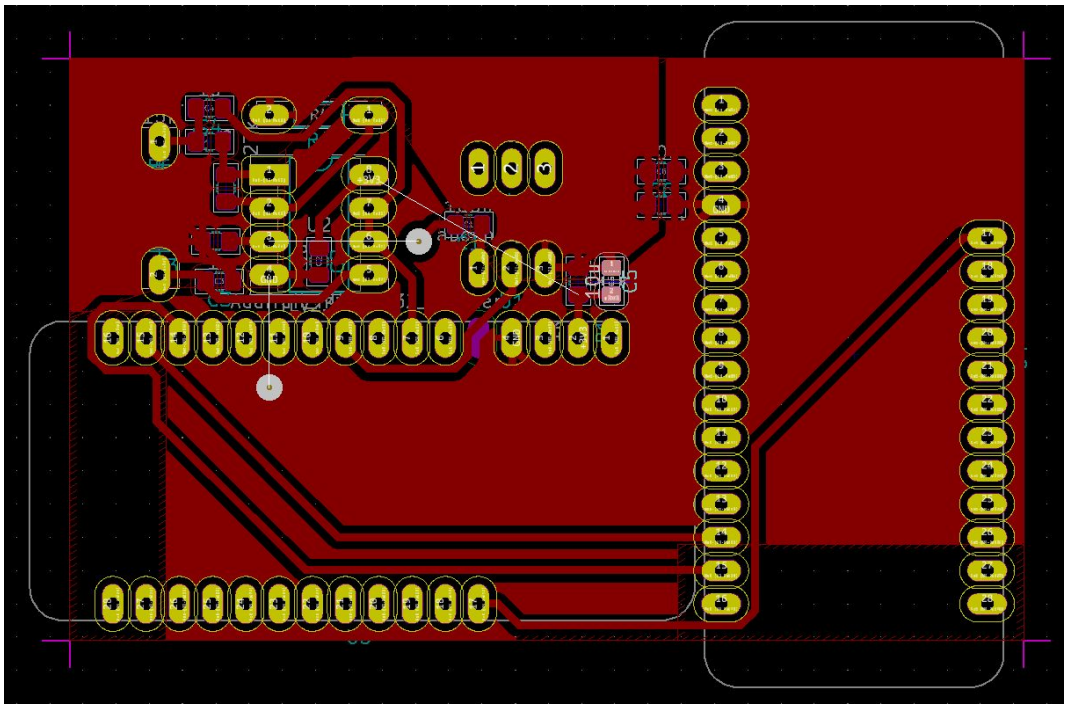
Kokeilujen jälkeen simuloimme sydänkytkentää Falstadin virtapiirisimulaattorilla. Tässä ensisijaisesti varmistuttiin komponenttien arvoista ja siitä, ettei signaali missään vaiheessa ylitä 3.3 V käyttöjännitteen rajaa ja leikkaannu. Otimme simulaatiossa huomioon myös mahdollisen anturin ja ihon välisen offset-jännitteen.



Kuva 8. Sydänkytkennän simulointia Falstadin avulla.

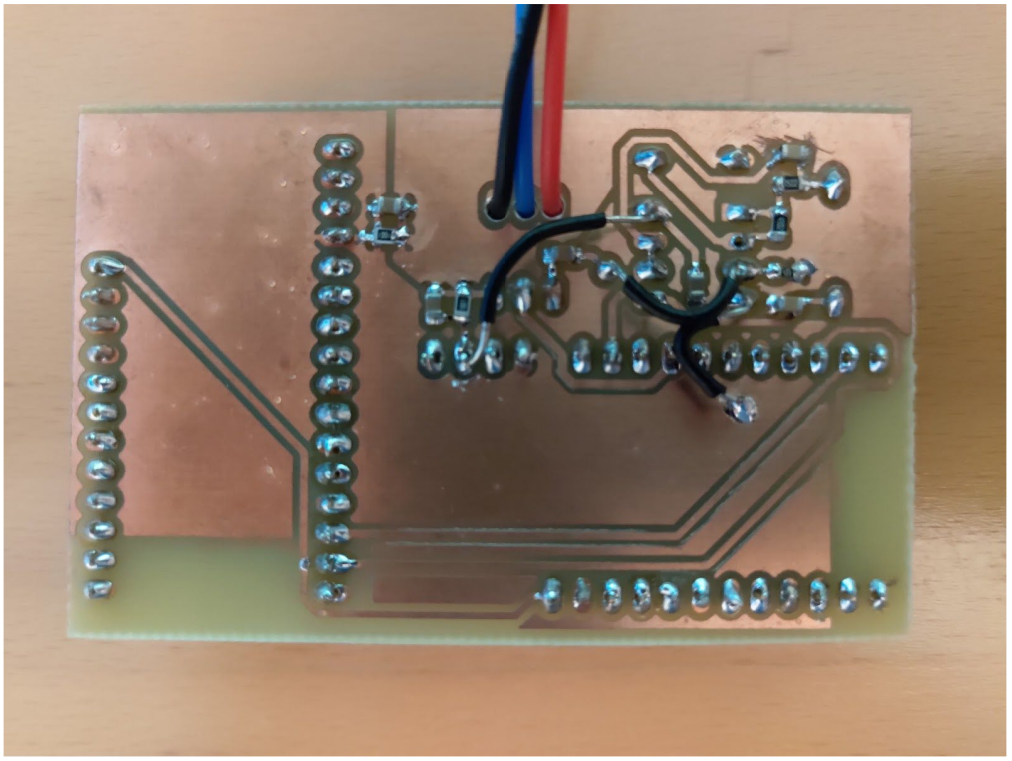
5.2.2 Valmistus

Piirilevy piirrettiin KiCadilla pitäen mielessä pajalla valmistamisen vaatimukset: kuparivetojen leveys ja tyhjät välit ovat leveydeltään 0.6 mm, läpireikäkomponenttien kuparitassut ovat ovaalin muotoisia ja 3 mm leveitä ja muuten tyhjäksi jääville alueille on asetettu kuparitäyttö, jotta syövytysliuos pysyisi hyvänä pidempään. Kuparitäyttö poistettiin vain alueilta, jotka ovat lähellä mikrokontrollerien antenneja häiriöiden välttämiseksi.



Kuva 9. Valmis piirros KiCadilla.

Piirilevy valmistettiin pajalla syövyttämällä ohjeistuksen mukaan. Poraa ja kolvia käytettiin. Kolmessa kohdassa oli vedettävä yhteys johdolla, koska vedot olivat ristikkäin. Anturin piuhat juotettiin suoraan piirilevyyn kiinni, ja niille porattiin reiät vedonpoistoiksi. Mikrokontrollereita varten piirilevylle juotettiin piikkirimat, jotta niiden irrottaminen ja kiinnittäminen kävisi helposti.



Kuva 10. Valmiin piirilevyn kuparipuoli.

6. Projektitoiminta

6.1. Tavoitteiden saavuttaminen

Alunperin tavoitteenamme oli pystyä loppunäytöksessä demonstroimaan projektia, jossa useammalla kuin yhdellä pelaajalla on kiinni prototyyppimme: Aboensen sensori kyljessä, kytkettynä johdoilla piirilevyyn, mikä on kiinnitettynä vyölle. Nämä laitteet mittaavat hengitystä ja sykettä samaan aikaan kun mikrokontrolleri laskee paikan koordinaatteja bluetoothin avulla. Data lähetetään käyttöliittymälle Wi-Fi:n välityksellä ja koneelta voidaan seurata pelaajien sykettä, hengitystä sekä paikkaa kentällä reaaliajassa.

Lopputuloksena saimme usealta pelaajalta tutkittua paikkaa ja esitettyä useat eri arvot käyttöliittymällä, mutta minkään saatu data ei ollut järkevää. Paikan sijaintitiedot hyppäsivät jatkuvasti kentän keskelle, koska Beaconeiden avulla toteutus oli altis signaalin heijastumiselle sekä estymiselle. Lisäksi sydämen sykkeen data oli heikkoa ja epäsäännöllistä, josta oli vaikea saada ulos selvää sykearvoa. Hengitysdatasta ei saatu myöskään järkeviä arvoja. Datan lähetys Wi-Fi:n avulla toimi ja UI kykeni esittämään datan kuten oli tarkoitus.

6.2. Aikataulu

Ajankäyttöä oli vaikea arvioida jo alussa, koska suunnitelmat muuttuivat useaan otteeseen. Osa projektin vaiheista vei enemmän tai vähemmän aikaa kuin oletus oli ja kaikkea ei lopulta ehditty edes saada toimimaan. Alkukesällä käytimme paljon aikaa, ehkä liian paljon, toteutuksen suunnitteluun ja kun lopulta muutimme joitakin suunniteltuja asioita kokonaan, aika ei lopulta riittänyt. Työskentelimme pitkälti yksin tai pareittain kesän aikana, joten kaikilla oli vastuu omasta ajankäytöstä. Jatkoa varten opimme, että vaikka huolellinen suunnittelu on tärkeää, se on parempi pitää ajallisesti tiiviimpänä. Monesti projektia työstäessä tulee esille uusia ongelmia, joita ei osaa ottaa huomioon suunnitteluvaiheessa.

6.3. Riskianalyysi

Riski	Vaarallisuus	Riskin toteen käyminen	Analyysi
Paikannuksen toiminta & tarkkuus	Suuri	Kyllä	BLE:n käyttäminen yksinään ei ole riittävä ratkaisu sisätiloissa (ainakaan ilman jatkokehitystä) Toiminta OK, tarkkuus puuttuu oikeastaan kokonaan
Sensorien toiminta	Pieni	Kyllä	Ilmeni, että sensori ei todennäköisesti pysty antamaan signaalia sydämen osalta pienen pinta-alansa vuoksi. Aboense ottaa tämä huomioon jatkokehityksessä.
Koodibugit	Keskisuuri	Ei	Ei huomattavia bugeja koodissa. Koodin testaamisen avulla saatiin ohjelmistot toimivaksi.
Aika	Keskisuuri	Ehkä	Projektin alkupäässä työtunteja hieman vähemmän, kun suunnitelmat muuttuivat useasti ja ei tiedetty mitä tehdä. Kesän loppupuolella tunteja tuli kaikille runsaasti enemmän.

7. Yhteenveto ja johtopäätökset

Emme lopulta saaneet prototyyppiä toimimaan kaikin puolin ja kesän aikana projekti osoittautui paljon haastavammaksi kuin alunperin osasimme odottaa. Loppunäytöksessä meillä oli esittää osia jotka toimivat ja osia jotka eivät toimineet, mutta opimme paljon uutta kesän aikana ja pääsimme todella perehtymään projektin eri osa-alueisiin. Jaoimme jo alussa vastuita projektiin liittyen, joten kaikki painivat lähtökohtaisesti jonkin oman ongelman kanssa ja pidimme paljon palavereita, joissa sitten annoimme tilannepäivityksen muille ryhmäläisille ja mietimme eri ratkaisumahdollisuuksia ongelmiin.

Projektin suunnitteluvaiheessa olisi ollut hyvä perehtyä tarkemmin aikaisempiin samankaltaisiin projekteihin ja niissä ilmi tulleisiin ongelmiin. Esimerkiksi BLE:n heikkous ainoana paikkadatan antajana oli selvästi esiin tullut trendi tutkimuksissa, joita tuli loppupäässä luettua yrittäessä etsiä ratkaisua ongelmiimme. Jos ongelmat olisi tiennyt jo valmiiksi, olisimme voineet valita jonkin toisen tavan toteuttaa paikkadata tai esimerkiksi yhdistää monta eri tapaa, joka oli liian myöhäistä ongelmien ilmetessä.

Meillä oli hyvä projektiryhmä ja sponsoriyritys, joiden kanssa työskentely oli mukavaa ja kurssista jatkamme eteenpäin hyvillä mielin, vaikka projekti ei toteutunut halutulla tavalla.

8. Linkit

Projektin git repository:
https://github.com/Lerrike/Protopaja2019

Falstadin virtapiirisimulaattori:
http://falstad.com/circuit/

KiCad:
http://www.kicad-pcb.org/

9. Lähteet

Naviginen trilateraatioalgoritmi:
https://github.com/Navigine/Indoor-navigation-algorithms/blob/master/README.md

ESP32 Arduino BLE:
https://github.com/nkolban/ESP32_BLE_Arduino

Scott Harden DIY ECG:
https://www.swharden.com/wp/2016-08-08-diy-ecg-with-1-op-amp/