Tampereen yliopisto

Anton Palm

IOT-OVIKELLON SUUNNITTELU JA TOTEUTUS

Informaatioteknologian ja viestinnän tiedekunta
Diplomityö
Tammikuu 2020

TIIVISTELMÄ

Anton Palm: IoT-ovikellon suunnittelu ja toteutus
Diplomityö
Tampereen yliopisto
Sähkötekniikka
Tammikuu 2020

Pienemmissä yritysrakennuksissa ei välttämättä ole vastaanottotiskiä vierailijoita varten, jolloin rakennuksen ulko-ovia voi joutua pitämään lukossa. Tällainen tilanne on esimerkiksi toimistorakennuksessa, jossa Sasken Finland Oy:n Tampereen toimipiste sijaitsee. Vieraiden vastaanottoa halutaan helpottaa ilmoittautumisjärjestelmällä, jonka tarkoituksena on mahdollistaa vierailijoiden itseilmoittautuminen.

Ilmoittautumisjärjestelmä koostuu ilmoittautumislaitteesta, palvelimesta sekä ovikellolaitteista. Tämä diplomityö on tehty ovikellolaitteen suunnittelusta ja toteutuksesta. Ovikellolaite sijoitetaan yrityksen tiloihin, jossa se hälyttää saapuvasta vieraasta, jolloin yrityksessä tiedetään, että vieras on saapunut.

Diplomityössä suunniteltiin ja toteutettiin fyysinen ovikellolaite, sekä esiteltiin laitteen vaatiman ohjelman rakennetta. Laitteen suunnittelussa tutkittiin erilaisia mikropiirejä ja sähkökytkentöjä, kuten galvaanisesti erotettu muuntaja sekä Power over Ethernetin mahdollistavat kytkennät.

Työssä on myös tutkittu ohjelmistossa tarvittavia AVR-mikrokontrollerin ominaisuuksia, joista eniten on keskitytty keskeytyksiin ja ajastin/laskureihin, joita käytetään äänen tuottamiseen. Ohjelmakoodi tehtiin C- ja C++-ohjelmointikielillä. Työssä on myös tutkittu erilaisia tekniikoita, kuten tiedonsiirtotekniikoita, sekä laitteessa käytettäviä protokollia, kuten TCP/IP, HTTP ja MQTT. Valmis laite saatiin vastaamaan sille asetettuja vaatimuksia.

Avainsanat: IoT, sulautetut järjestelmät, mikrokontrolleri, AVR, Ethernet, Power over Ethernet, MQTT

ABSTRACT

Anton Palm: The design and implementation of an IoT-doorbell
Master's thesis
Tampere University
Electrical Engineering
January 2020

Smaller office buildings do not necessarily have a reception desk for visitors, so the external doors to the building may have to be locked. This is the case for example in the office building where Sasken Finland Oy's Tampere office is located. To make receiving visitors easier, a registration system is being planned, which is meant to enable the visitors to self-register as a visitor.

The registration system consists of a registration device, a server and doorbell devices. This master's thesis describes the design of the doorbell device. The device is placed to a company's office, where it alerts when a visitor is arriving, so the workers know a visitor has arrived.

In this master's thesis the physical device of the doorbell was designed and implemented. The structure of the program for the device was also presented. In the electronics design, a few different microchips and electrical connections were introduced, such as a galvanically isolated transformer and the necessary circuitry for using Power over Ethernet.

Some features of an AVR-microcontroller, which were needed by the program, were studied. The emphasis was on AVR interrupts and timer/counters, which were used to produce audio. The program was written using C- and C++-programming languages. A few different technologies were explored, such as different data transfer techniques, and protocols used by the device, such as TCP/IP, HTTP and MQTT. The finished device met its requirements.

Keywords: IoT, embedded systems, microcontroller, AVR, Ethernet, Power over Ethernet, MQTT

ALKUSANAT

Tämä työ on tehty Sasken Finland Oy:lle osana Respamax-ilmoittautumisjärjestelmäprojektia. Aloittaessani diplomityötä minulle annettiin muistaakseni vain aihe sekä ainoana vaatimuksena Power over Ethernetin hyödyntäminen, joten sain hyvin vapaat kädet laitteen suunnitteluun. Diplomityön kirjoitus ja sitä myötä valmistuminen yliopistosta on hieman venähtänyt, sillä muut työt yrityksessä ovat päässeet varastamaan huomioni. Kuten töissä tämänhetkisessä projektissa on tapana sanoa, homma etenee kuin hunajassa juoksisi.

Yliopistolta haluan kiittää työni vastuuohjaajaa professori Jukka Vanhalaa työhön annetuista kommenteista sekä korjailusta. Jukkaa sekä toista tarkastajaa, professori Karri Palovuorta, kiitän myös heidän luennoimistaan kursseista, jotka ovat avanneet minulle ovet mikrokontrollerien ihmeelliseen maailmaan.

Saskenin puolelta haluan kiittää ohjaajaani Jani Turpeista avusta etenkin työn alkuvaiheessa, sekä muuta Respamax-tiimiä ideointiavusta ja parannusehdotuksista työn edetessä.

Suuret kiitokset myös kavereilleni ja perheelle. Haluan etenkin kiittää pitkäaikaista tyttöystävääni Suvia tuesta ja turvasta, jota olen aina tarvittaessa saanut. Erityismainінta vielä meidän äijien WhatsApp-ryhmälle, jossa päivittäin heitetyt huumoriletkautukset ovat auttaneet jaksamaan synkimpinäkin hetkinä.

Tampereella, 29.1.2020

Anton Palm

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

ACK	Acknowledgement, vahvistusviesti
AES	Advanced Encryption Standard, salausmenetelmä
bitti	tietotekniikassa tiedon pienin käsiteltävä osa, 1 tai 0
C	ohjelmointikieli
C++	ohjelmointikieli
CTC	Clear Timer on Compare, AVR-ajastimen toimintamoodi
HTTP	HyperText Transfer Protocol, tiedonsiirtoprotokolla
HVPP	High Voltage Parallel Programming, AVR-mikrokontrollerien ohjelmointitapa
HVSP	High Voltage Serial Programming, AVR-mikrokontrollerien ohjelmointitapa
I/O	Input/Output, Sisäänmeno/ulostulo
IEEE	Institute of Electrical and Electronics Engineers, kansainvälinen tekniikan alan järjestö
IoT	Internet of Things, esineiden internet
ISP	In-system programming, AVR-mikrokontrollerien ohjelmointitapa
LQFP	Low profile Quad Flat Package, mikropiirin kotelotyyppi
MISO	Master In Slave Out, isäntälaitteen kuunteleva tiedonsiirtolinja SPI:ssä
MOSI	Master Out Slave In, isäntälaitteen lähettävä tiedonsiritolinja SPI:ssä
MQTT	MQ Telemetry Transport, tiedonsiirtoprotokolla
PCINT	Pin Change Interrupt, Pinnin arvon muutoksesta seuraava keskeytys mikrokontrollerissa
PCM	Pulse-code modulation, analogisen signaalin modulointitekniikka
PoE	Power over Ethernet, virransyöttö Ethernetin kautta
PSE	Power Sourcing Equipment, virrantuottolaitteisto Power over Ethernetissä
PWM	Pulse-width modulation, signaalin modulointitekniikka
QFN	Quad Flat No-leads package, mikropiirin kotelotyyppi
QoS	Quality of Service, tietoliikenteen priorisointia kuvaava termi
RAM	Random Access Memory, hajasaantimuisti
RJ45	yleisesti Ethernetissä käytetty liitin
ROM	Read-only Memory, lukumuisti
SCLK	Serial clock, kellosignaali SPI:ssä
SPI	Serial Peripheral Interface, sarjamuotoinen oheislaiteväylä
SS	Slave Select, kuuntelevan laitteen valintasignaali SPI:ssä
SSOP	Shrink small-outline package, mikropiirin kotelotyppi
tavu	8 bittiä
TCP/IP	Transmission Control Protocol / Internet Protocol, tietoliikenneprotokollien yhdistelmä
UDP	User Datagram Protocol, yhteydetön tiedonsiirtoprotokolla
USART	Universal Syncronous/Asynchronous Receiver-Transmitter, sarjaliikenteen lähetys- ja vastaanottopiiri
USB	Universal Serial Bus, Sarjaväyläarkkitehtuuri
Wi-Fi	Wireless Fidelity, IEEE-standardiperhettä käyttävä langaton verkkotekniikka
WLAN	Wireless Local Area Network, langaton lähiverkko

1. JOHDANTO

Suurissa yritysrakennuksissa on vastaanottovirkailijoita, jotka hoitavat saapuvien vieraiden sisään kirjaamisen ja varmistavat ettei alueella oleskele asiattomia ihmisiä. Kaikissa pienemmissä rakennuksissa ei ole niin suurta määrää vierailijoita, että olisi tarvetta erillisille vastaanottovirkailijoille. Rakennuksiin ei myöskään todennäköisesti haluta päästää asiattomia ihmisiä sisälle, joten ulko-ovet on pidettävä lukossa.

Tällainen tilanne on esimerkiksi rakennuksessa, jossa Sasken Finland Oy:n Tampereen toimipiste sijaitsee. Rakennuksessa toimii useita yrityksiä, mutta siellä ei kuitenkaan ole vastaanottotiskiä. Koska ovet ovat lukossa, rakennuksessa sijaitseviin yrityksiin tulevat vierailijat joutuvat ilmoittautumaan viereisen rakennuksen vastaanotossa. Vieras ilmoittautuu vastaanottovirkailijalle, joka ottaa yhteyttä kohdeyrityksessä olevaan vieraan isäntään.

Vieraiden ilmoittautuminen toisessa rakennuksessa on hieman epäkäytännöllistä. Työntekijät joutuvat hakemaan vierailijan kauempaa toisesta rakennuksesta. Rakennukseen on mentävä ulkokautta, joten matkalla toiseen rakennukseen voi myös esimerkiksi kastua.

On myös tilanteita, joissa vierailijan ei tarvitse varsinaisesti tulla yrityksen tiloihin sisälle asti. Näin voi olla esimerkiksi tavarantoimituksessa, jolloin riittäisi, että kuka vain työntekijä tulisi vastaanottamaan lähetyksen. Tällä hetkellä näissäkin tilanteissa tavarantoimittajat joutuvat käymään viereisessä rakennuksessa ilmoittautumassa. Tavarantoimituksissa ei välttämättä ole edes sovittu tapaamisaikaa, jolloin yhteyshenkilöt eivät mahdollisesti ole tavoitettavissa, mikä vaikeuttaa tilannetta vielä enemmän.

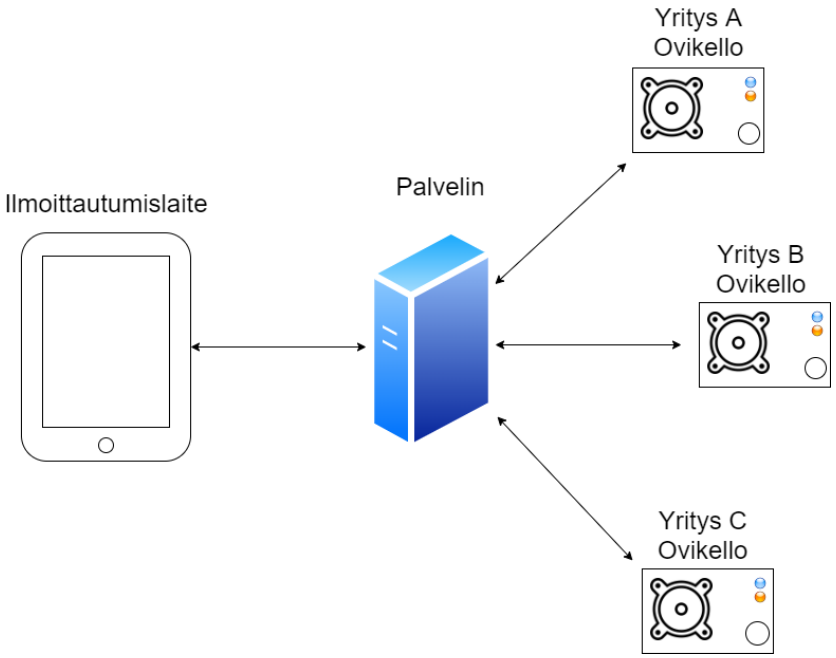
Ilmoittautumista halutaankin sujuvoittaa ilmoittautumisjärjestelmällä, jolla vierailijat voivat ilmoittautua tai soittaa ovikelloa, mikäli yritykseen tiloihin ei tarvitse tulla. Tämä diplomityö keskittyy ilmoittautumisjärjestelmässä tarvittavan ovikellolaitteen suunnitteluun. Työn tavoitteena on saada suunniteltua ja toteutettua toimiva ovikellolaite, joka kykenee kommunikoimaan järjestelmässä olevan ilmoittautumislaitteen kanssa ja ilmoittamaan saapuvista vierailijoista yrityksessä.

Työssä esitellään ilmoittautumisjärjestelmä lyhyesti, sekä esitellään muutamia keskeisiä teknologioita, joita laite käyttää. Ovikellolaitteen elektroniikan suunnittelua käydään läpi,

sekä esitellään valmistetut prototyypit. Myöhemmin esitellään valitun mikrokontrollerin ominaisuuksia ja kuinka niitä hyödynnetään mikrokontrollerin ohjelmoinnissa. Lopuksi esitellään testausympäristö, jossa laitteen ohjelmistoa kehitettiin, sekä kerrotaan kuinka laite toimi oikeaan palvelimeen yhdistettynä.

2. ILMOITTAUTUMISJÄRJESTELMÄ

Ilmoittautumisjärjestelmä on tulossa monikerroksiseen toimistorakennukseen, johon on kaksi ulko-ovea ja siinä toimii useita yrityksiä. Ilmoittautumisjärjestelmään kuuluvat ulko-ovien yhteydessä olevat ilmoittautumislaitteet, yritysten tiloissa sijaitsevat ovikellot, sekä laitteiden välisen kommunikaation hoitava palvelin. Yksinkertainen esimerkki on havainnollistettu kuvassa 1.



Kuva 1. *Esimerkki ilmoittautumisjärjestelmästä.*

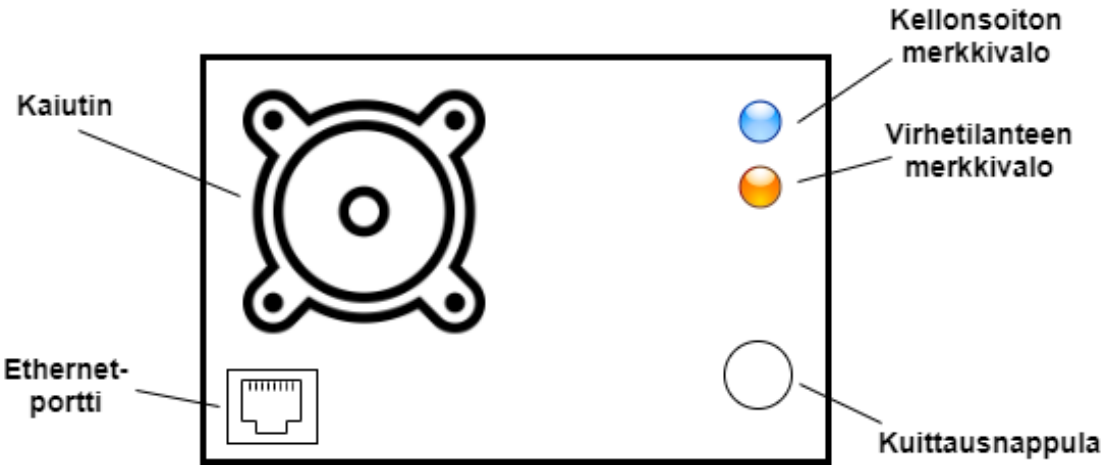
2.1 Ilmoittautumislaite

Ilmoittautumislaitetta käyttävä ihminen voi joko ilmoittautua yritykseen vierailijaksi, jolloin ilmoittautumislaite lähettää viestin palvelimelle, joka ilmoittaa vierailijasta isännälle esimerkiksi sähköpostilla tai Slack-pikaviestintäsovelluksella, tai soittaa ovikelloa, jolloin palvelin lähettää soittopyynnön yrityksen tiloissa sijaitsevaan ovikelloon.

Ilmoittautumislaitteena voi toimia esimerkiksi kioskitilaan asetettu tabletti, joka laitetaan suorittamaan ilmoittautumisohjelmaa. Ilmoittautumislaite sijoitetaan ulko-oven läheisyyteen, kuten tuulikaappiin.

2.2 Ovikellolaite

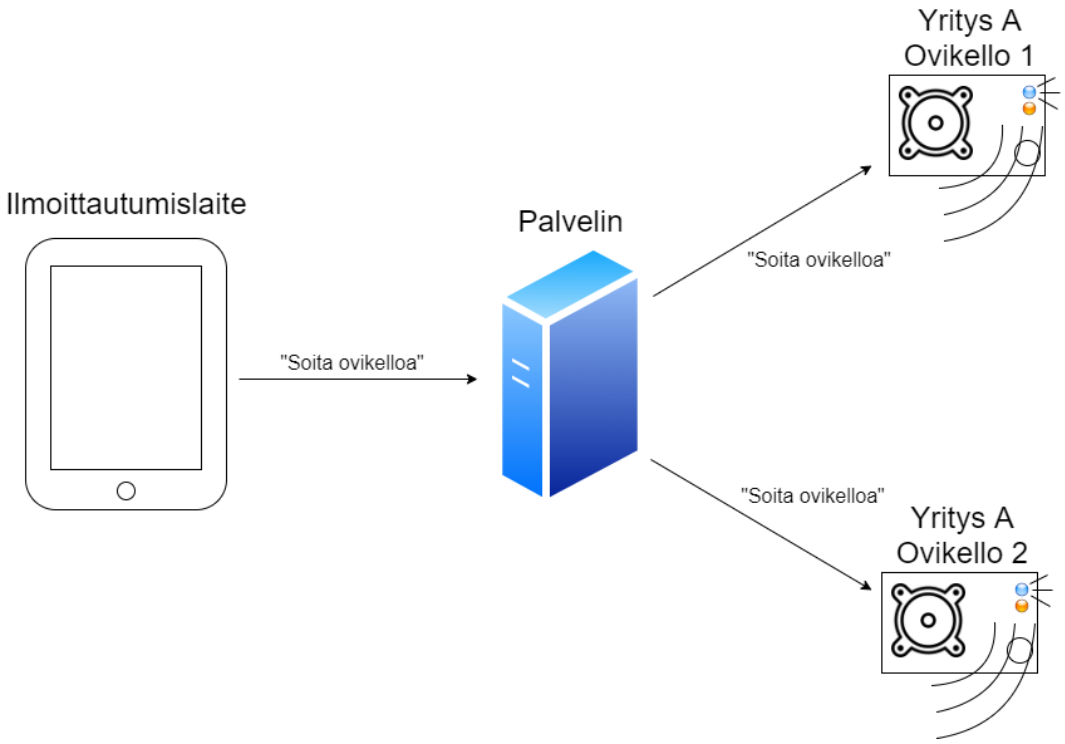
Yrityksen tiloissa sijaitsevan ovikellon tarkoitus on ilmoittaa yritykselle saapuvasta vierailijasta, sekä mahdollistaa vastaaminen vierailijalle kuittausnappulan avulla. Ovikellon ulkoiset ominaisuudet on esitelty kuvassa 2.



Kuva 2. *Ovikellolaitteen ulkoiset ominaisuudet.*

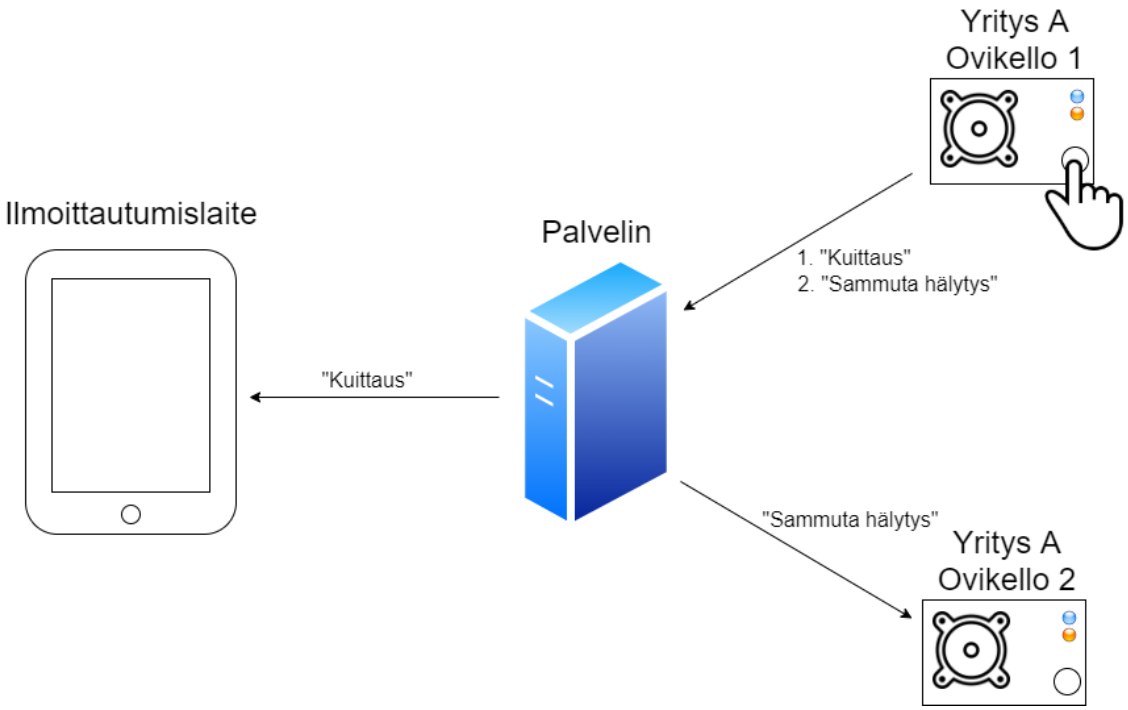
Ovikello kytketään Ethernet-kaapelilla lähiverkkoon ja sitä kautta internetin välityksellä palvelimeen. Ethernet-portin kautta laitteelle toteutetaan myös virransyöttö niin kutsuttua PoE:a (Power over Ethernet, virransyöttö Ethernetillä) käyttämällä. PoE:sta on kerrottu lisää luvussa 3.

Kun laitteelle tulee ovikellon soittopyyntö, se asettaa merkkivalon päälle ja alkaa soittamaan kaiuttimesta hälytysääntä. Järjestelmän on tarkoitus mahdollistaa useamman ovikellon käyttö yhdessä yrityksessä, joten palvelimen tulee lähettää soittopyyntö jokaiselle yritykseen liitetylle ovikellolle. Tätä on havainnollistettu kuvassa 3.



Kuva 3. *Ovikellon soittopyyntö.*

Kuittausnappulalla hälytys voidaan lopettaa, jolloin myös merkkivalo sammuu. Nappulan painamisesta lähetetään kuittausviesti palvelimen kautta ilmoittautumislaitteelle, jolloin vierailija tietää, että häntä ollaan tulossa hakemaan. Ovikellon pitää myös pystyä kertomaan muille ovikellolaitteille, että vierailijalle on jo vastattu ja hälytyksen voi lopettaa. Tätä on havainnollistettu kuvassa 4.



Kuva 4. *Ovikellolaitteella kuittaaminen.*

Virhetilanteita, kuten yhteyden katkeamista palvelimeen, varten on merkkivalo, jolla laite voi ilmoittaa olevansa virhetilanteessa. Laitteeseen tulee myös äänenvoimakkuuden säätö.

3. LAITTEEN SUUNNITTELU

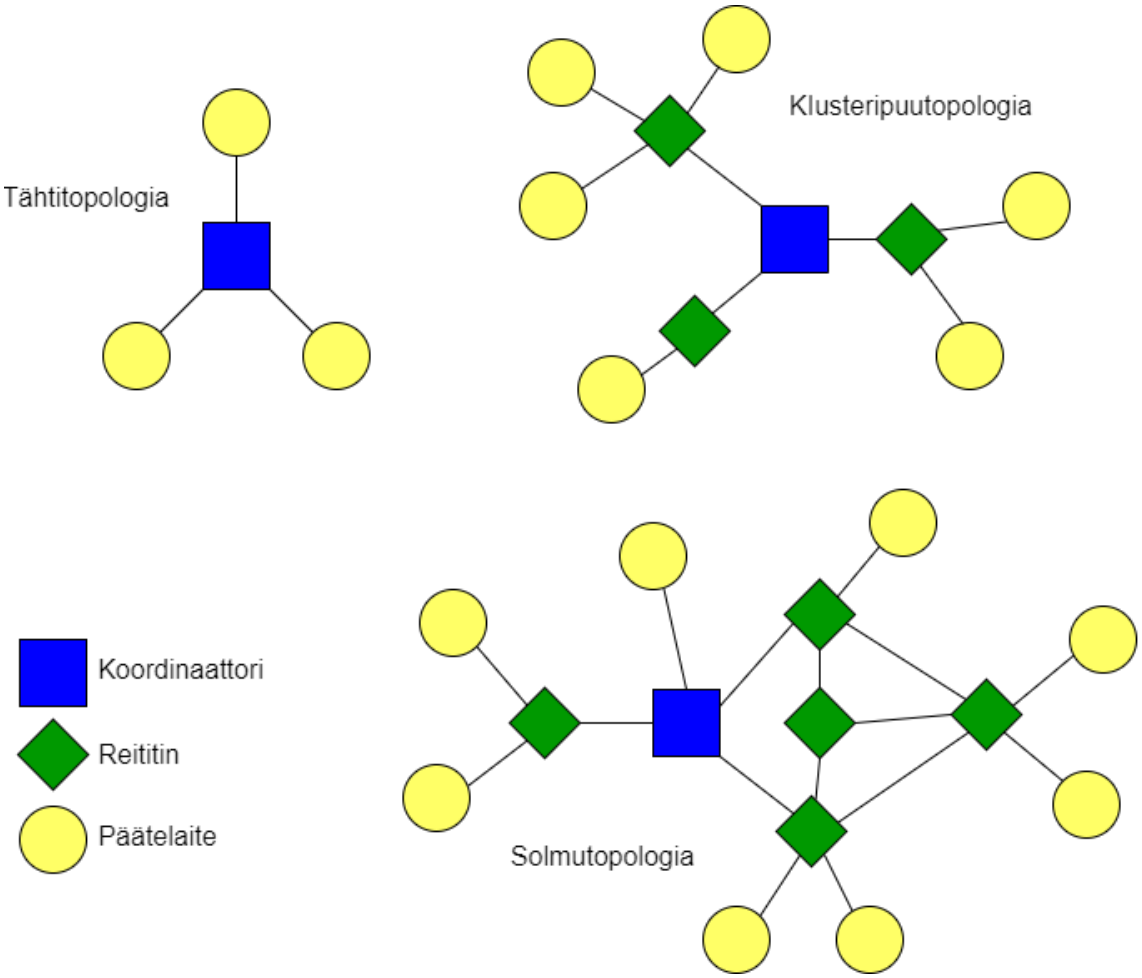
Tässä luvussa kerrotaan eri tiedonsiirtotekniikoista ja valitaan niistä ovikellolaitteelle sopivin. Lisäksi luvussa esitellään laitteeseen tulevia mikropiirejä ja niiden ominaisuuksia, sekä erilaisia ovikellolaitteen käyttämiä sähkökytkentöjä. Luvussa kerrotaan myös, miten ovikellolaite saadaan tuottamaan ääntä mikrokontrollerin avulla.

3.1 Tiedonsiirtotekniikka

Rakennuksessa, johon ilmoittautumisjärjestelmä on tulossa, on neljä kerrosta, joten tiedonsiirtotekniikkaa valitessa tulee varmistua siitä, että data saadaan siirtymään ilmoittautumislaitteita kuuntelevalta palvelimelta ovikelloille luotettavasti. Seuraavassa on esitelty muutamia vaihtoehtoja laitteen käytettäväksi tiedonsiirtotekniikaksi.

3.1.1 Zigbee

Zigbee on Zigbee Alliancen kehittämä IEEE 802.15.4–standardiin perustuva langaton tietoliikenneteknologia. Se on pienivirtainen ja mahdollistaa lyhyen viiveen, mikä tekee teknologiasta sopivan IoT-laitteille. Yhdessä Zigbee-verkostossa voi olla jopa 65 000 laitetta [1]. Kuvassa 5 on esitelty erilaisia Zigbeen verkkotopologioita.



Kuva 5. *Erilaisia Zigbeen verkkotopologioita.*

Zigbeen tukemat verkkotopologiat ovat tähti-, klusteripuu- ja solmutopologia. Tähtitopologiassa, kuvassa vasemmalla ylhäällä, on yksi koordinaattori, joka vastaa verkoston muodostamisesta. Kaikki muut päätelaitteet yhdistyvät siihen ja kaikki tieto kulkee sen kautta. Klusteripuutopologiassa, kuvassa oikealla ylhäällä, on lisäksi reitittimiä, jotka voivat välittää päätelaitteiden dataa eteenpäin muille reitittimille tai päätelaitteille. [1]

Monimutkaisin topologia on solmutopologia, kuvassa oikealla alhaalla, jossa on lisäksi reitittimiä, jotka voivat muodostaa useamman yhteyden toisiin reitittimiin. Reitittimien avulla laitteet voivat lähettää pitkiltäkin etäisyyksiltä viestejä toisilleen. Useammalla yhteydellä verkko voi valita optimaalisimman lähetysreitin datalle, mikä parantaa verkon luotettavuutta ja nopeutta. [2]

Zigbeen solmuverkossa tieto verkkoon yhdistetyistä laitteista päivittyy automaattisesti, kun siihen liittyy uusia laitteita tai vanhoja poistuu. Zigbee kykenee 250 kbit/s nopeuteen 2,4 GHz:n taajuudella. Zigbee tukee AES-salausta (Advanced Encryption Standard), joka on erittäin vahva suojaus. Kyseistä suojausta käyttää myös esimerkiksi

Yhdysvaltojen hallitus [3]. Zigbeen kantama voi olla jopa yli 300 metriä esteettömissä olosuhteissa, sisätiloissa alle 100 metriä [4].

Zigbee-päätelaitteet tukevat virransäästötilaa. Artem Dementyevin et al. tekemässä tutkimuksessa [5] virrankulutus virransäästötilassa oli hyvin alhainen, vain noin 4,2 µA. Aktiivisena virrankulutus oli testitilanteessa 9,3 mA. Kokeessa 10 sekunnin päivitysintervalleilla laite oli aktiivisena 2,5 % ajasta, jolloin keskimääräiseksi virrankulutukseksi saadaan

$$I_{avg} = \frac{2{,}5}{100} * 9{,}3\ \text{mA} + \frac{97{,}5}{100} * 4{,}2\ \mu\text{A} \approx 0{,}24\ \text{mA}.$$

Esimerkiksi 3,3-volttisen CR2032-nappipariston kapasiteetti on mallista riippuen noin 235 mAh [6], jolla laite pysyisi toiminnassa noin 37 päivää.

3.1.2 WLAN

Toinen vaihtoehto langattomalle tiedonsiirrolle on WLAN (Wireless local area network, langaton lähiverkko). Se perustuu IEEE 802.11 -standardiin. Standardin mahdollistamat tiedonsiirtonopeudet voivat olla useita gigatavuja sekunnissa [7]. Käytännössä pienet IoT-sovelluksiin tarkoitetut WLAN-moduulit eivät kuitenkaan kykene näin suuriin nopeuksiin. Esimerkiksi eräässä ESP8266-järjestelmäpiirissä on sisäänrakennettu WLAN-toiminnallisuus, joka IoTBitsin tekemän tutkimuksen [8] mukaan kykenee 7 Mbit/s tiedonsiirtonopeuteen. Verrattuna Zigbeen 250 kbit/s nopeuteen se on kuitenkin huomattavasti nopeampi.

Toisin kuin Zigbeessä, jossa ilmoittautumislaitteet ja ovikellolaitteet muodostavat keskenään verkon, WLAN:ia käyttämällä laitteet voivat muodostavat yhteyden toisiinsa lähiverkon tai internetin välityksellä. Uudempaa 802.11n standardia käyttävien WLAN-laitteiden kantama voi olla noin 70 metriä sisätiloissa [9].

WLAN ei ole yhtä vähävirtainen kuin Zigbee. Esimerkiksi ESP8266 kuluttaa datalehden mukaan [10] virtaa dataa vastaanottaessa noin 50 mA. Lepotilassakin virrankulutus on noin 1 mA, joka on yli 200-kertainen Zigbeen virransäästötilaan verrattuna. Mikäli järjestelmäpiiri olisi dataa vastaanottavassa tilassa 2,5 % ajasta, kuten Zigbeen vertailussa, saataisiin keskimääräiseksi virrankulutukseksi

$$I_{avg} = \frac{2{,}5}{100} * 50\ \text{mA} + \frac{97{,}5}{100} * 1\ \text{mA} \approx 2\ \text{mA}.$$

Tällä kulutuksella pelkkä verkkoyhteys tyhjentäisi nappipariston alle 5 päivässä.

3.1.3 Ethernet

Ethernet on IEEE-802.3 standardiin perustuva teknologia. Se on yleisesti tunnettu nimellä langallinen lähiverkko. Ethernet ja WLAN käyttää samoja tiedonsiirtoprotokollia, joiden erona on lähinnä fyysinen tiedonsiirtomuoto. Koska Ethernet käyttää kaapelia tiedonsiirtoon, laitteen kantaman rajoitteena on käytännössä verkkokaapelin pituus. [11]

3.2 Tietoliikenne ja virransaanti

Suuressa rakennuksessa ulko-ovella sijaitsevan ovikellolaitteen lähettämät langattomat viestit joutuvat kulkemaan lattioiden ja seinien läpi, joten Zigbeetä tai WLAN:ia käytettäessä ei voida varmistua siitä, että ovikellolaitteet ympäri rakennusta vastaanottaisivat kaikki viestit luotettavasti. Viestikatkojen estämiseksi langattoman tiedonsiirtotekniikan valitsemisesta luovutaan ja laite päädytään toteuttamaan Ethernetiä käyttämällä.

Toinen peruste Ethernetin valintaan on virrankulutus. Vaikka laitteet saataisiin käyttämään suhteellisen vähän sähköä, tarvitsisi akkuja kuitenkin vaihtaa tai latailla, mistä aiheutuu ylimääräisiä huoltotoimenpiteitä.

3.2.1 Power over Ethernet

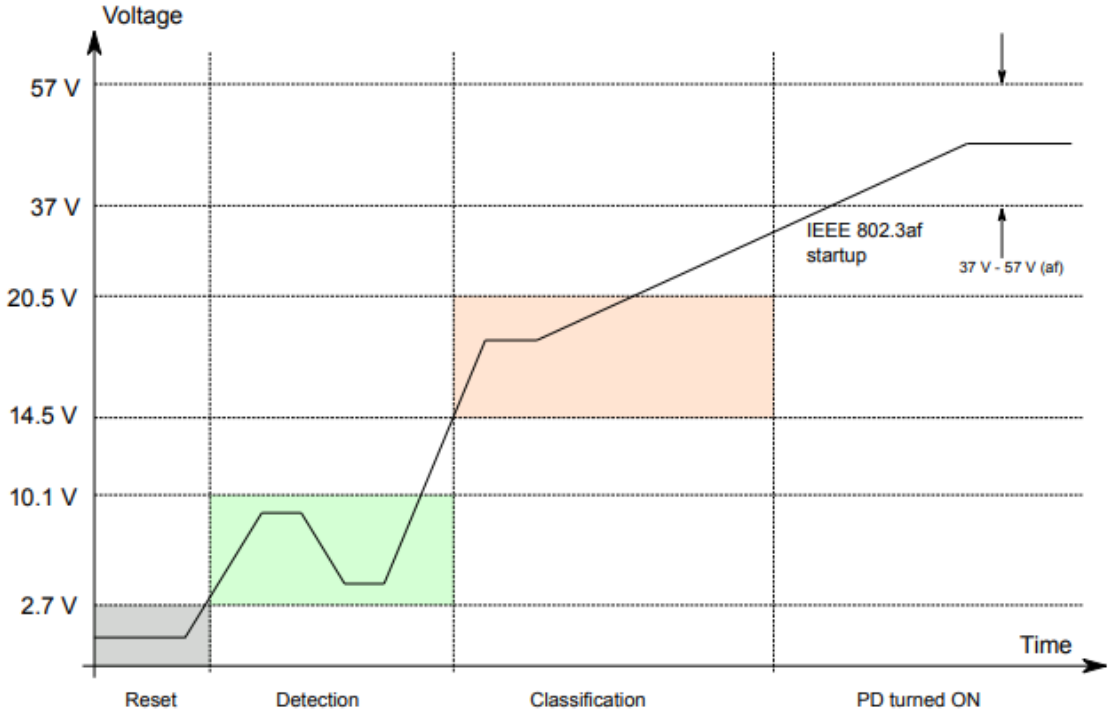
Ethernetiä käytettäessä virransyöttö voidaan toteuttaa käyttämällä PoE-teknologiaa (Power over Ethernet, virransyöttö Ethernetin kautta). Tavallisessa Ethernet-kaapelin RJ45-liittimessä on 8 liitintä, joista 4 liitintä on varattu tiedonsiirtoon. Liittimet 4,5,7 ja 8 ovat normaalisti käyttämättömiä [12]. IEEE on julkaissut Power over Ethernet –standardit IEEE 802.3af/at/bt, joissa käyttämättömiä pinnejä käytetään virran syöttöön. PoE-tyyppejä on neljä, joiden keskeisimmät ominaisuudet on esitelty taulukossa 1.

Taulukko 1. Power over Ethernet –tyypit [13].

PoE-tyyppi	Maksimiteho (W)	Jännite laitteella (V)	Maksimivirta (mA)
Tyyppi 1	12,95	37–57	350
Tyyppi 2	25,5	42,5–57	600
Tyyppi 3	51	42,5–57	600 (paria kohden)
Tyyppi 4	71	41,1–57	960 (paria kohden)

Laite tulee kuluttamaan alle 13 wattia tehoa, joten valitaan käytettäväksi tyypiksi tyyppi 1.

Power over Ethernet jaotellaan aktiiviseen ja passiiviseen PoE:en. Aktiivisessa PoE:ssa PSE (Power sourcing equipment, virrantuottolaitteisto), kuten PoE:ta tukeva verkkokytkin tai reititin, tekee laitteen kanssa kättelyn, missä sovitaan mitä PoE-tyyppiä -ja luokkaa käytetään [14]. Tyypin 1 PoE-kättelyä on havainnollistettu kuvassa 6.



Kuva 6. *Power over Ethernet -kättely* [15].

Kun laite yhdistetään PSE:hen, se alkaa syöttää 2,7–10,1 V jännitettä laitteelle, jolloin laitteessa tulee olla 23,7–26,3 kΩ resistanssi. Kun reititin havaitsee resistanssin, se nostaa jännitteen yli 14,5 V:hen ja alkaa mittaamaan virrankulutusta, jonka perusteella se määrittelee käytetyn luokan.

Tyypin 1 PoE:ssa on 4 luokkaa, jotka määrittävät laitteelle sallitun maksimitehon. Esimerkiksi PoE-kontrolleri Si3404 määrittää luokitusvirran I_{CLASS} luokitusvastuksen R_{CLASS} perusteella. Jos halutaan käyttää tiettyä luokkaa, resistanssi saadaan kaavasta

$R_{CLASS} = \frac{1{,}35\ \mathrm{V}}{I_{CLASS}}$ [15].

Kyseisen PoE-kontrollerin suunnitteluohje suosittelee käytettäväksi taulukossa 2 esiintyviä vastuksen arvoja.

Taulukko 2. Erään PoE-kontrollerin luokitusvastuksen R_{class}:n valitseminen [16].

PoE-luokka	I_{CLASS} min (mA)	I_{CLASS} max (mA)	R_{CLASS} (Ω)	Sallittu teho (W)
Luokka 0	0	4	avoin	0,44–12,95
Luokka 1	9	12	140	0,44–3,84
Luokka 2	17	20	75	3,84–6,49
Luokka 3	26	30	48,7	6,49–12,95

Luokitus on vapaavalintainen prosessi, jota käytettäessä laite voi kertoa PSE:lle tarvitsemansa tehon määrän, jolloin esimerkiksi teholtaan rajoitettu PSE voi allokoida virtaa laitteille paremmin [17]. Mahdollisen luokituksen jälkeen liittimille aletaan tuottaa luokalle sopivaa jännitettä.

Passiivinen PoE ei ole varsinainen PoE-standardi. Passiivisessa PoE:ssa RJ45-portin liittimille vain tuodaan tasajännite, ilman minkäänlaista kättelyä. Passiivista PoE:a käytettäessä tulee olla tarkka käytettävän laitteen jännitevaatimuksesta. On esimerkiksi 24 V jännitettä käyttäviä PoE-kameroita, jotka 47 V:n passiivinen PoE voi rikkoa. [18]

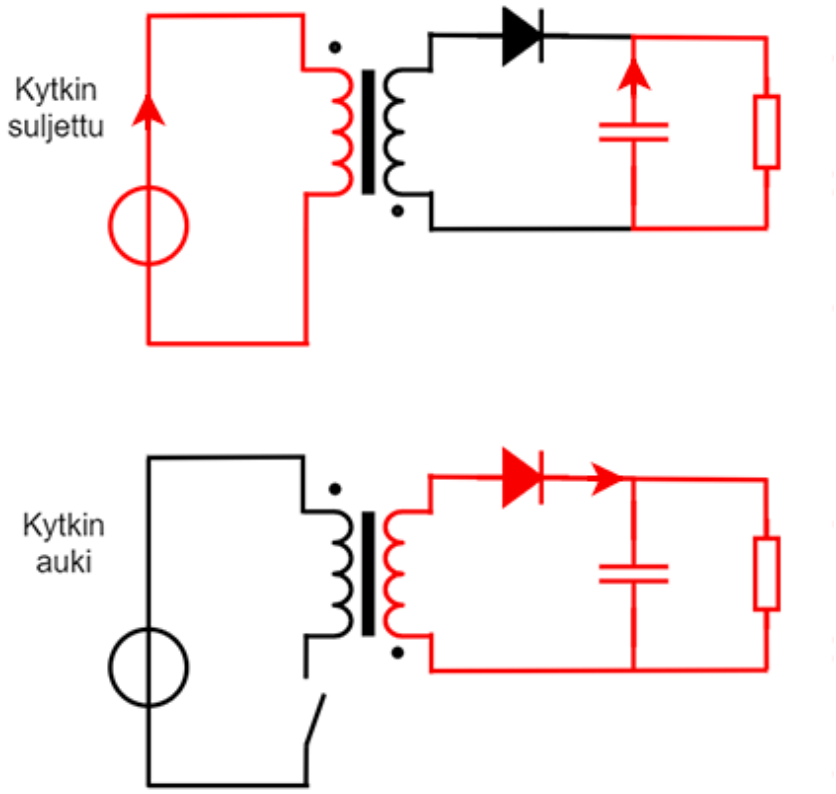
Passiivisen PoE:n voi toteuttaa PoE-injektoria käyttämällä. PoE-injektori yhdistää siihen tuotavan verkkokaapelin tiedonsiirtojohtimet ja erillisestä virtalähteestä tuotavan jännitteen samaan RJ45-porttiin [19].

3.2.2 PoE-kytkentä

PoE:n noin 47 V jännite on liian korkea laitteessa käytettäville mikropiireille. Esimerkiksi käytettävän Ethernet-kontrollerin suositeltu käyttöjännite on 3,3 V [20]. Jännite lasketaan muulle elektroniikalle sopivaksi muuntimen avulla. PoE-standardin mukaan laitteen sisältäessä ulkoisia liitäntöjä, tulee PoE-kytkennän ja muun laitteen välillä tulee olla galvaaninen erotus, eli kytkentöjen välillä ei saa olla suoraa sähköyhteyttä [21].

Jännitteen muuntaminen galvaaninen erotus ylläpitäen toteutetaan flyback-muuntimella. Muuntimessa on ferriittiydin, johon ensiö- ja toisiopuoli ovat käämittynä vastakkaisiin suuntiin, jolloin indusoituva jännite on vastakkaissuuntaista. Puolten väliset jännitteet ovat suoraan verrannollisia käämittyjen kierrosten lukumäärään. [22]

Flyback-muuntimen toiminta on esitetty kuvassa 7.



Kuva 7. Flyback-muuntimen toiminta.

Kun kytkin on suljettu (kuvassa ylhäällä), muuntimen ensiöpuoli kytkeytyy suoraan jännitelähteeseen, jolloin virta ja magneettivuo kasvavat ja muuntimen energia kasvaa. Toisiopuolelle indusoitunut jännite on negatiivinen, jolloin diodi on estosuuntainen, ja toisiopuoleen kytketty kondensaattori syöttää virtaa kuormalle.

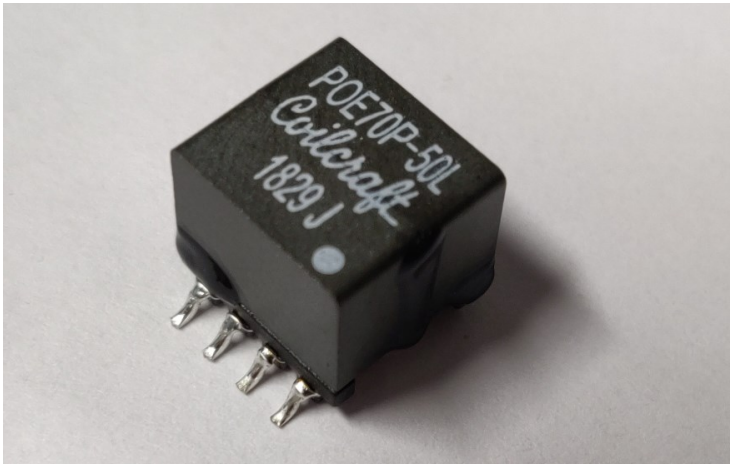
Kun kytkin avataan, ensiöpuoli ei ole enää kytkettynä jännitelähteeseen ja muuntimeen varastoitunut energia alkaa purkautua toisiopuolelle positiivisena. Tällöin diodi on myötäsuuntainen, jolloin muuntimen energia lataa kondensaattoria ja syöttää virtaa kuormalle. [22]

Laitteeseen tulevan muuntajan voisi tehdä itse käämimällä kuparilankaa ferriittiytimen ympärille. Mikäli RJ45-liittimeen tuleva jännite olisi 47 V ja muun laitteen lähdejännitteeksi halutaan 5 V, tulisi muuntajan käämittyjen kuparilankojen kierrosten suhdeluku olla

$$\frac{N1}{N2} = \frac{U1}{U2} = \frac{47\,\mathrm{V}}{5\,\mathrm{V}} = 9{,}4.$$

PoE:ssa syötettävä jännite voi standardin mukaan vaihdella aiemmin esitellyn taulukon mukaisesti 37…57 V välillä. Tällöin tarkasta muunnoksesta ei ole hyötyä, ja jännitettä ohjataankin PoE-kontrollerilla, jolle voidaan tehdä takaisinkytkentä muunnetun jännitteen puolelta. Kontrolleri käyttää takaisinkytkettyä jännitettä vertausarvona, jonka perusteella

se kytkee jännitteen syötön muuntajalle päälle tai pois päältä ylläpitäen sopivaa jännitettä.

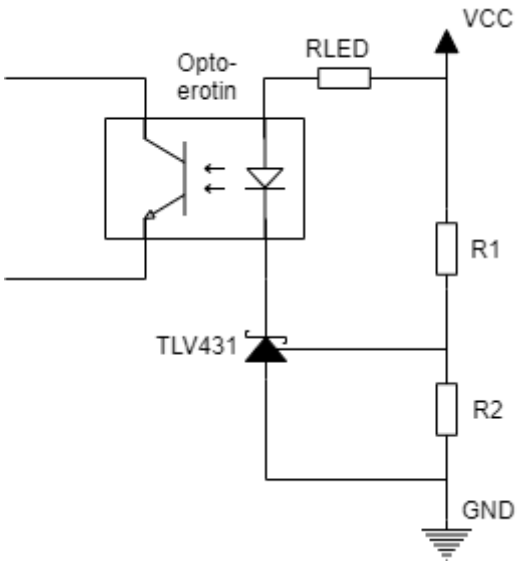


Kuva 8. Flyback-muunnin POE70P.

Flyback-muuntimeksi valitaan PoE-kytkentöjä varten suunniteltu kuvassa 8 näkyvä POE70P.

3.2.3 Muunnetun jännitteen takaisinkytkentä

Kuvassa 9 on esitetty PoE-kontrollerille takaisinkytkennän mahdollistava kytkentä.



Kuva 9. Jännitteen takaisinkytkentä.

Takaisinkytkentää varten tarvitaan jännitereferenssi, joka tuottaa tasaisen jännitteen kuormasta tai jännitteestä riippumatta [23]. Jotta galvaaninen eristys säilyy, takaisinkytkentään käytetään optoerotinta. Optoerotin on suljettuun koteloon sijoitettu

ledi ja fototransistori. Kun optoerottimen ledin puolelle tuodaan jännite, ledi alkaa tuottamaan valoa, joka saa fototransistorin johtavaan tilaan [24].

Kytkentään valitun TLV431-jännitereferenssin jännite on tyypillisesti 1,24 V [25]. Koska jännite pysyy vakiona, vastuksen R2 yli oleva jännite on myös oltava 1,24 V. Jännitteenjakosäännön perusteella kytkennän ulostulojännite on

$$V_{out} = 1{,}24\ \mathrm{V} + \frac{R1}{R2} * 1{,}24\ \mathrm{V}.$$

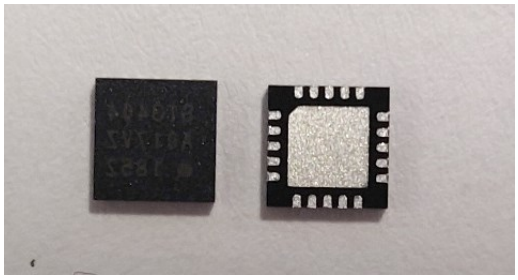
Kun ulostulojännitteeksi halutaan 5 V, saadaan vastusten R1 ja R2 suhteeksi

$$\frac{R1}{R2} = \frac{V_{out}}{1{,}24\ \mathrm{V}} - 1 = \frac{5\ \mathrm{V}}{1{,}24\ \mathrm{V}} - 1 \approx 3{,}032.$$

Eli valitsemalla ylemmäksi vastukseksi resistanssiltaan noin kolminkertainen alempaan nähden, saadaan ulostulojännitteeksi noin 5 V.

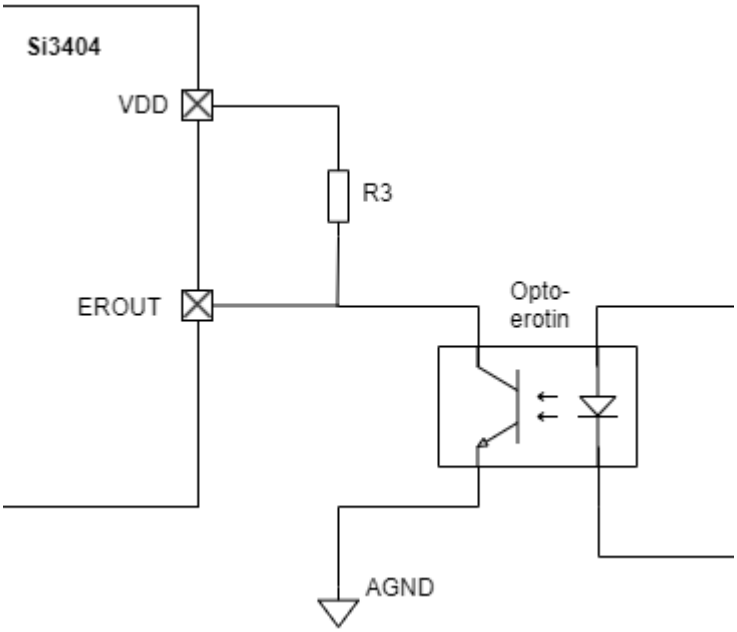
3.2.4 PoE-kontrolleri

PoE-jännitteen ohjaamiseen käytetään sitä varten tehtyä PoE-kontrolleria, joiden tehtävänä on integroida tehonhallinta ja kytkennän ohjaaminen [15]. Laitteeseen valitaan PoE-kontrolleriksi kuvassa 10 näkyvä Si3404.



Kuva 10. PoE-kontrolleri Si3404.

Galvaanisesti eristetyssä optoerottimessa olevan fototransistorin kollektori kytketään PoE-kontrollerin tuottamaan 5 V lähdejännitteeseen *VDD* ja vertailupinniin *EROUT*, kuvassa 11 esitetyn kytkentäkaavion mukaisesti.



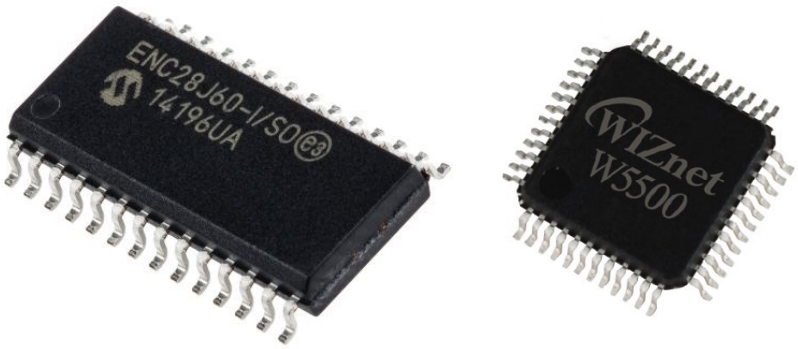
Kuva 11. Takaisinkytkentä PoE-kontrollerille.

Kun flyback-muuntimen tuottama jännite on tavoiteltua ulostulojännitettä suurempi, optoerottimen ledi menee päälle ja fototransistori menee johtavaan tilaan. Tällöin jännite VDD kytkeytyy maahan vastuksen kautta, jolloin myös EROUT-signaali on kytkettynä maahan. Tämä saa kontrollerin lyhentämään flyback-muuntimessa olevan kytkimen päälläoloaikaa, jolloin muuntimen toisiopuolen jännite pienenee [16]. Kun flyback-muuntimen jännite on pienempi kuin tavoiteltu ulostulojännite, ledi menee pois päältä ja fototransistori lopettaa johtamisen, jolloin EROUT-signaali muuttuu aktiiviseksi. Tämä saa PoE-kontrollerin pidentämään muuntimeen liitetyn kytkimen päälläoloaikaa, ja jännite kasvaa.

PoE-kontrolleri toimii 250 kHz taajuudella, jolloin takaisinkytkennän toimintakin on hyvin nopeaa, joten jatkuvasta muutoksesta huolimatta ulostulojännite tasaisena [15].

3.3 Ethernet-kontrolleri

Ethernet-kontrollerin tehtävä on vastaanottaa lähiverkosta ovikellolaitteelle tulevaa dataa ja välittää ne edelleen mikrokontrollerille prosessoitavaksi. Ethernet-kontrollerin valinnassa käytetään apuna muita internetiä käyttäviä laitteita. Esimerkiksi Arduino-mikrokontrollerialustalle löytyy ohje [26], jossa Arduino UNO:on kytketään Ethernet-kontrolleri ENC28J60, joka on kuvassa 12 vasemmalla. Ohje sisältää myös valmiin ohjelmointikirjaston. Toisaalta Arduinoille on myös virallinen Ethernet-lisälevy, joka käyttää W5500-piiriä, joka on kuvassa 12 oikealla. Sille on myös olemassa valmis ohjelmointikirjasto [27].



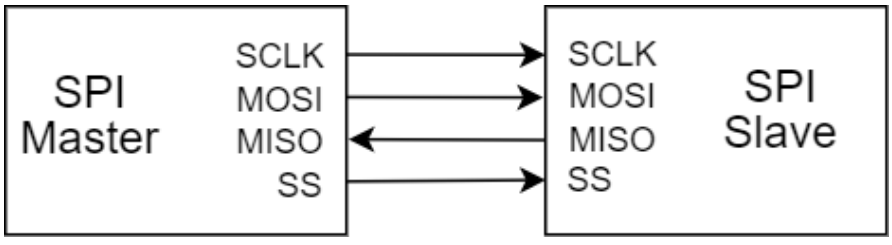
Kuva 12. Ethernet-kontrollerit ENC28J60 ja W5500 [27] [28].

Molemmat kontrollerit vaikuttavat sopivilta toteutukseen, joten ne otetaan tarkempaan vertailuun. Ethernet-kontrollerien ominaisuuksia on vertailtu taulukossa 3.

Taulukko 3. Ethernet-kontrollerien vertailu [31] [32].

Ethernet-kontrolleri	W5500	ENC28J60/SS
Sisäänrakennettu TCP/IP	Kyllä	Ei
Sisäinen muisti (kt)	32	8
Tiedonsiirtonopeus (Mbit/s)	100	10
SPI-väylän nopeus (MHz)	80	20
Käyttöjännite (V)	2,97–3,63	3,1–3,6
Kotelointi	LQFP-48	SSOP-28
Hinta (€)	2,23	2,51

Molemmat Ethernet-kontrollerit käyttävät SPI:tä (Serial Peripheral Interface, sarjamuotoinen oheislaiteväylä) kommunikoimiseen mikrokontrollerin kanssa. Kahden laitteen välinen SPI-kytkentä on esitetty kuvassa 13.



Kuva 13. SPI-väylän kytkentä.

SPI-väylässä on isäntä (master), jolla on yksi tai useampi isäntää kuunteleva laite (slave). Isäntälaite tuottaa kellosignaalin (SCLK), joka mahdollistaa synkronisen tiedonsiirron. Datalinjoja on kaksi, isäntälaitteelta lähtevä (MOSI, Master Out Slave In) ja

isäntälaitteeseen tuleva (MISO, Master In Slave Out). Kahden väylän ansiosta laitteet voivat lähettää ja vastaanottaa viestejä samanaikaisesti. Neljättä signaalia (SS, Slave Select) voidaan käyttää valitsemaan kuuntelevista laitteista se, jonka kanssa isäntä haluaa keskustella. Mikäli väylässä on useampi kuunteleva laite, isäntälaitteen tulee tuoda jokaiselle kuuntelevalle laitteelle oma Slave Select -signaali. [31]

W5500-kontrollerissa on sisäänrakennettu tuki TCP/IP:lle (Transmission Control Protocol / Internet Protocol). Kontrollerille tulevia TCP/IP-paketteja pystytään purkamaan jo Ethernet-kontrollerissa, joten pakettien käsittely helpottuu mikrokontrollerissa. Ilman valmista tukea protokollille, jollaista ENC28J60:lla ei ole, tulee pakettien purkaminen tehdä mikrokontrollerissa, mikä lisää mikrokontrollerin kuormitusta [30]. Sen lisäksi vaadittava ohjelmakoodi joudutaan sijoittamaan mikrokontrollerin muistiin, joka on yleensä hyvin pieni.

Lisäksi W5500 tukee suurempia internet- ja SPI-väylän nopeuksia kuin ENC28J60, sekä siinä on enemmän sisäistä muistia siirrettäviä datapaketteja varten. Ovikellosovellus ei kuitenkaan tarvitse suuria tiedonsiirtonopeuksia ja ENC28J60:n hitaammatkin nopeudet riittävät, joten nämä ominaisuudet eivät ole niin merkittäviä kontrollerin valinnan kannalta.

W5500:n LQFP-48-kotelon (low-profile quad flat package), pinnien etäisyys toisistaan on hieman pienempi kuin ENC28J60:lla, joten se on hieman vaikeampi juottaa. Parempien teknisten ominaisuuksien lisäksi W5500 on kuitenkin myös halvempi, joten se valitaan toteutukseen.

3.4 Mikrokontrolleri

Mikrokontrollerin tehtävä ovikellossa on kommunikoida Ethernet-kontrollerin kanssa, tuottaa ääntä kaiuttimen avulla sekä toteuttaa painonapin ja ledien toiminnallisuus. Mikrokontrollerin valinnassa on huomioitava ainakin

- Yhteensopivuus SPI-väylän kanssa
- Muistin määrä
- Ohjelmoitavien I/O-pinnien lukumäärä.

Mikrokontrollerin on tuettava Ethernet-kontrollerin tukemaa SPI-väylää, jotta ne voivat kommunikoida keskenään. Valmis ovikello-ohjelma käyttää noin 30 kilotavua ROM-muistia (Read-only Memory, lukumuisti) ja yhden kilotavun RAM-muistia (Random Access Memory, luku- ja kirjoitusmuisti), mikä tulee ottaa huomioon mikrokontrollerin valinnassa.

Tarkempaan vertailuun valitaan kuvassa 14 näkyvät kaksi mikrokontrolleria, ATmega328PB ja EFM8BB31F32, sekä aiemmin esitelty ESP8266EX-järjestelmäpiiri.

Kuva 14. Vertailtavat mikrokontrollerit [32] [33] [34].

Mikrokontrollereista otetaan vertailuun tärkeimpiä ominaisuuksia, jotka on esitelty taulukossa 4.

Taulukko 4. Mikrokontrollerien ominaisuuksien vertailu [35] [36] [10].

Laite	ATmega328PB	EFM8BB31F32	ESP8266EX
Väylän leveys (bittiä)	8	8	32
Kellotaajuus (MHz)	20	50	160
ROM-muisti (kt)	32	32	1024
RAM-muisti (kt)	2	2,25	50
Verkkoyhteys	Ei	Ei	Wi-Fi
I/O -liittimet	23	21	17
Kotelo	TQFP-32	QSOP-24	QFN-32
Hinta (€)	1,21	0,78	1,40

Vertailun perusteella ESP8266EX vaikuttaa piireistä selkeästi tehokkaimmalta 160 MHz:n kellotaajuudella. Piirissä on hieman vähemmän I/O-liittimiä (Input/Output, sisäänmeno/ulostulo) kuin muissa. I/O-liittimet kuitenkin riittäisivät toteutukseen, sillä mikrokontrolleriin kytketään Ethernet-kontrolleria käyttävän SPI-väylän 4 liitintä, 2 lediä ja äänilähtö. Piiriä valmistetaan vain QFN-32-kotelossa (Quad Flat No-leads package), joka on vaikea juottaa käsin puuttuvien jalkojen takia [10]. Sen lisäksi piirin alla piilossa olevaa padia ei voi juottaa kolvilla, joten juottamiseen tarvitsisi käyttää kuumailmaa tai uunia. Piiri on myös hieman kalliimpi kuin kaksi muuta vertailtavaa.

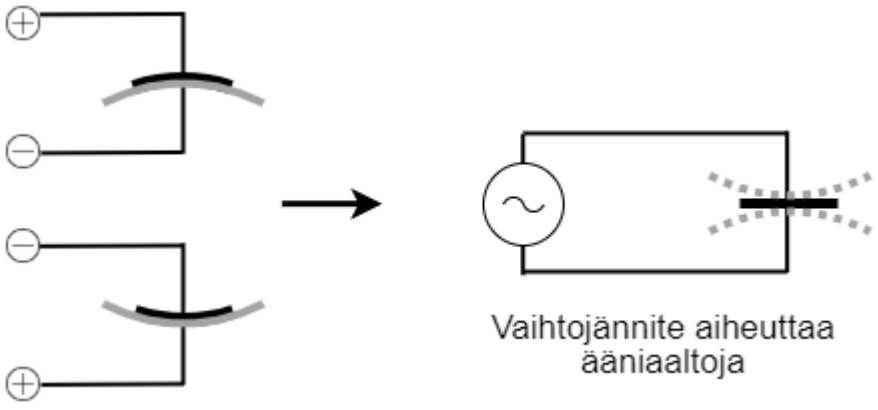
EFM8BB31F32-mikrokontrolleri on huomattavasti halvempi kuin muut piirit ja sen kellotaajuus on ATmega328PB:hen verrattuna 2,5-kertainen. RAM-muistia on myös hieman enemmän kuin ATmega328PB:ssä. Näyttää kuitenkin siltä, että EFM8-piireillä ei ole samanlaista avointa yhteisöä kuin ATmegalla, jolla on esimerkiksi ATmega-mikropiireihin perustuva Arduino-yhteisö [37]. Laajasta yhteisöstä voi olla apua ohjelmoinnin ongelmatilanteissa. ATmega-mikrokontrollereille on myös valmiita ohjelmointikirjastoja esimerkiksi Ethernetin käyttöön [27]. EFM8-piireillä sellainen täytyisi mahdollisesti tehdä itse, joten mikrokontrolleriksi valitaan ATmega328PB.

3.5 Ääni

Laitteen tarvitsee tuottaa ääntä hälytystilanteessa. Mikrokontrollerisovelluksissa voidaan käyttää äänentuottoon esimerkiksi pietsosummereita tai kaiuttimia. Äänen tuotto perustuu tietyllä taajuudella vaihtelevaan jännitteeseen, mikä saa äänielementissä olevan kalvon värähtelemään. Seuraavissa alaluvuissa esitellään pietsosummerin ja kaiuttimen toimintaa ja niiden eroja, sekä miten mikrokontrolleri saadaan tuottamaan äänisignaali.

3.5.1 Pietsosummeri

Pietsosummereissa on pietsosähköinen levy kiinnitettynä metallilevyyn. Kun elementtiin aiheutetaan sähkökenttä, se taivuttaa levyä kuvan 15 mukaisesti. Vaihtojännitteellä levy taipuu jännitteen taajuuden tahdissa, jolloin värähtelevä kalvo alkaa tuottamaan ääniaaltoja. [38]

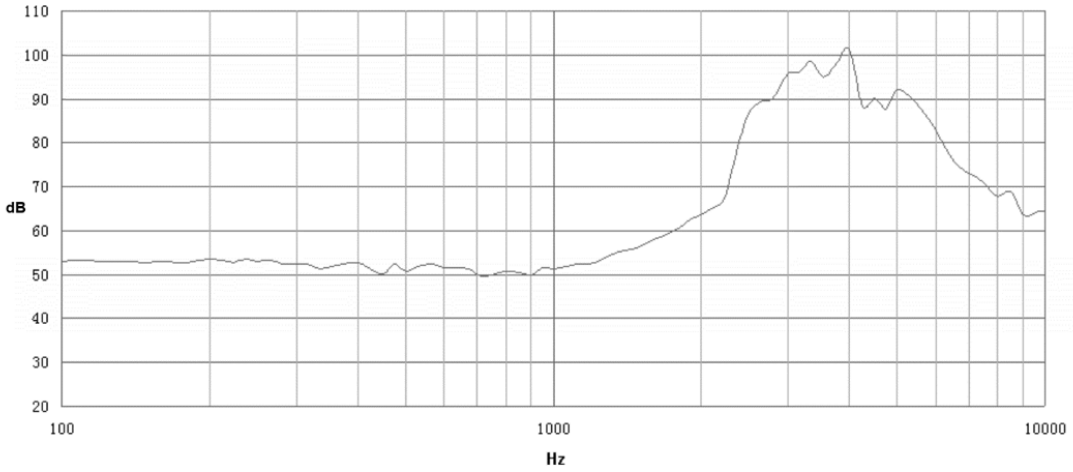


Kuva 15. Pietsosummerin toimintaperiaate.

Pietsosummereissa voi olla valmiina pietsoelementin ohjauspiiri, joka tuottaa sille tuotavasta tasajännitteestä ääntä tietyllä taajuudella. Näitä summereita voidaan käyttää esimerkiksi palovaroittimissa tai mikroaaltouuneissa, joissa yksinkertainen ääni

riittää [38]. Ovikello halutaan tuottamaan ääntä muuttuvalla äänentaajuudella miellyttävämmän äänen tuottamiseksi, joten valmista ohjauspiiriä ei voida siinä käyttää.

Pietsosummerit eivät muutenkaan yleensä toista matalataajuuksisia ääniä hyvin [39]. Kuvassa 16 on esitetty erään pietsosummerin taajuusvaste.

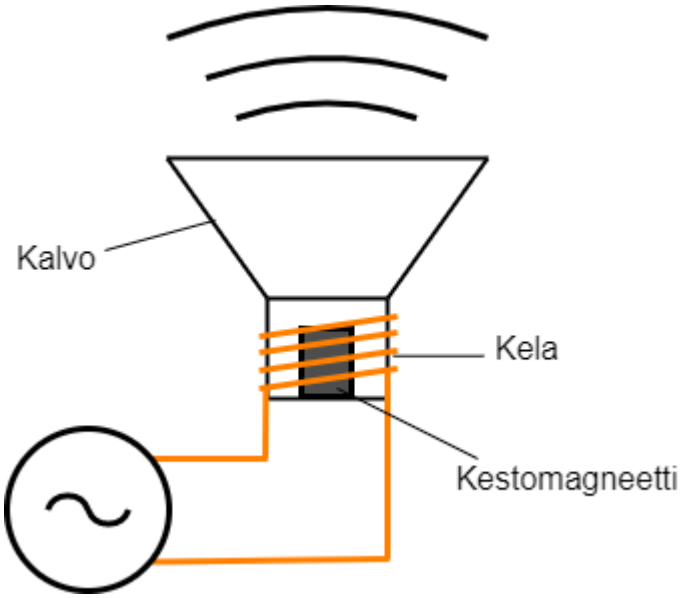


Kuva 16. Pietsosummerin CPT-3055-90PM taajuusvaste [40].

Taajuusvasteen mukaan esimerkiksi 80 dB:n äänenvoimakkuus saavutetaan suurin piirtein taajuusalueella 2,5–6 kHz. Alemmilla taajuuksilla äänenvoimakkuus laskee huomattavasti, jolloin ääni voi kuulostaa vääristyneeltä tai hiljaiselta.

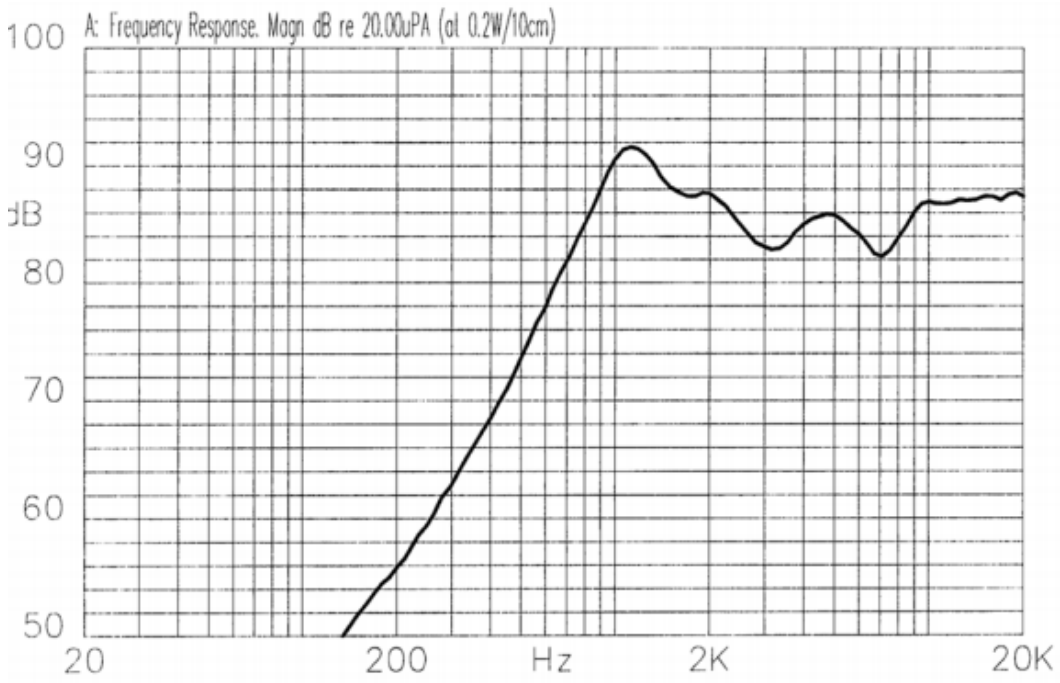
3.5.2 Kaiutin

Kaiuttimissa äänen tuotto perustuu sähkömagnetismiin. Yksinkertainen malli kaiuttimesta on esitetty kuvassa 17.



Kuva 17. Yksinkertainen malli kaiuttimesta.

Kaiuttimissa on kestomagneetti sekä sähkömagneettina toimiva kela, joka on kiinnitetty joustavaan materiaaliin. Kelassa vaihtuva sähkökenttä saa aikaan muuttuvan magneettikentän, jolloin kestomagneetti vaihtelevasti vetää kelaa puoleensa ja hylkii sitä. Tällöin myös kelaan kiinnitetty kalvo liikkuu, mikä aiheuttaa ääntä [41]. Kaiuttimet kuluttavat tyypillisesti enemmän sähköä kuin pietsosummerit, mutta niiden taajuusalue on myös suurempi [42].



Kuva 18. Kaiuttimen CDMG13008L-02 taajuusvaste [43].

Esimerkiksi kuvassa 18 näkyvän kaiuttimen taajuusvasteesta näkee, että 80 dB:n äänenvoimakkuus saavutetaan suurin piirtein taajuudessa 650 Hz, ja pysyy sen

yläpuolella vähintään 20 kHz:iin asti. Korkeammilla taajuuksilla kyseisen kaiuttimen taajuusvaste on myös suhteellisen tasainen, jolloin ääni toistuu tarkemmin kuin pietsosummerilla. Näistä syistä toteutukseen valitaan kaiutin.

Kaiuttimen tulee olla riittävän kovaääninen, jotta se tulee kuulluksi erilaisissa ympäristöissä. Toteutukseen valitaan kaiutin SM450208-1, jonka ilmoitettu taajuusalue on 450–5000 Hz. Se kykenee tuottamaan 92 dBA:n äänen 10 senttimetrin etäisyydelle. [44]

3.5.3 Mikrokontrolleri äänilähteenä

Kaiuttimelle tulevan äänisignaalin tuottamiseen käytetään mikrokontrollerin PWM-ulostuloa (Pulse-width modulation, pulssinleveysmodulaatio). PWM-signaalissa ulostulo vaihtelee tietyllä taajuudella ykkösen ja nollan välillä.

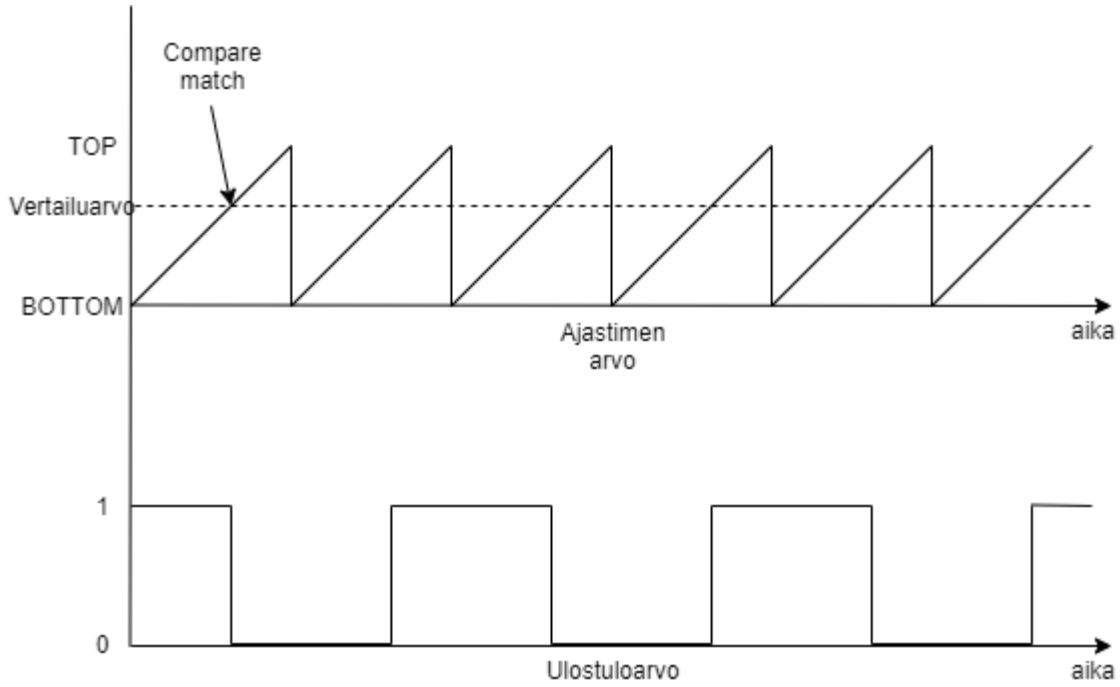
Mikrokontrollerin PWM-signaalin tuottaminen perustuu siinä oleviin sisäänrakennettuihin ajastin/laskurilohkoihin. Ajastin/laskurissa on vertailuyksikkö, joka vertaa ajastimen arvoa vertailuarvoon. Kuvassa 19 on esitelty vertailuyksikön eri toimintatilat.

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected WGM02 = 1: Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match, set OC0A at BOTTOM (non-inverting mode)
1	1	Set OC0A on Compare Match, clear OC0A at BOTTOM (inverting mode)

Kuva 19. Vertailuyksikön eri toimintatilat [35].

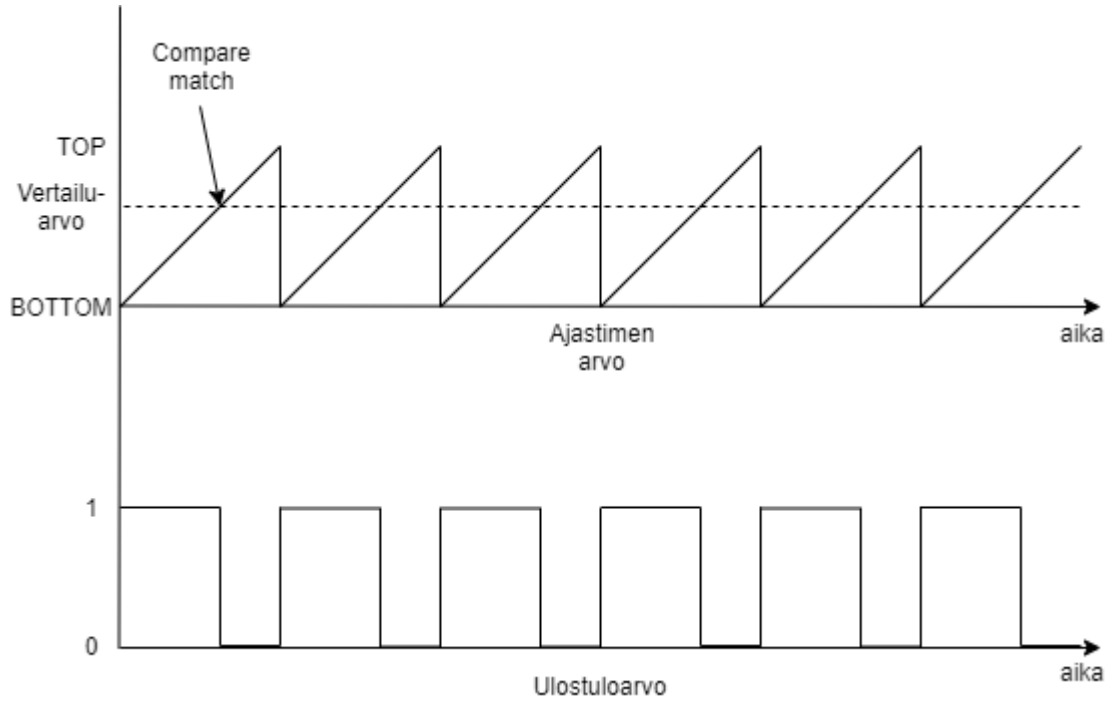
Kun vertailuyksikköä ohjaavien rekisteribitit ovat asetettu arvoon 00, vertailuyksikköön liitetyn ulostulopinnin toiminta on normaali, eli se ei käytä ajastin/laskurilohkon tuottamaa tulosta.

Arvolla 01, kun ajastimen toiminta on päällä, ulostulopinnin arvo vaihdetaan aina, kun ajastimen arvo täsmää vertailuarvoon. Tätä on havainnollistettu kuvassa 20.



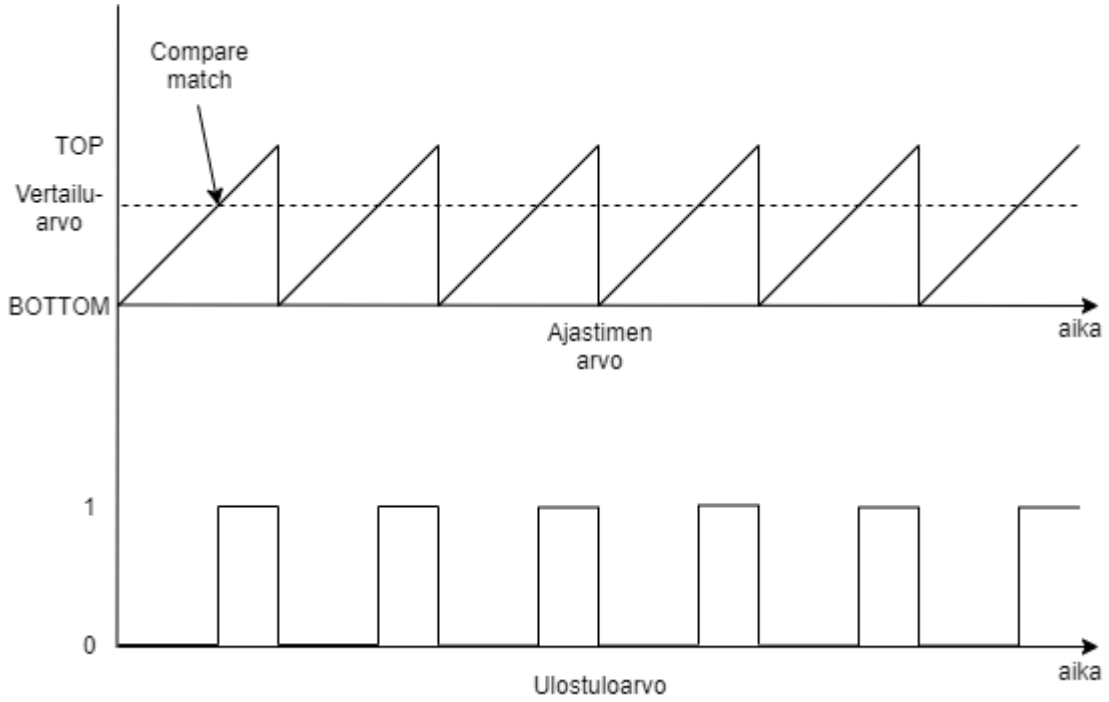
Kuva 20. Ajastimen toiminta COM0A[1:0] arvolla 01.

Arvolla 10 toiminta on niin kutsutusti ei-invertoiva, jolloin vertailuyksikön ulostulopinnin arvo asetetaan nollaksi, kun vertailu täsmää. Kun ajastin ylittää maksimiarvon ja palautuu takaisin nollaan, ulostulopinnin arvo asetetaan ykköseksi. Tätä on havainnollistettu kuvassa 21.



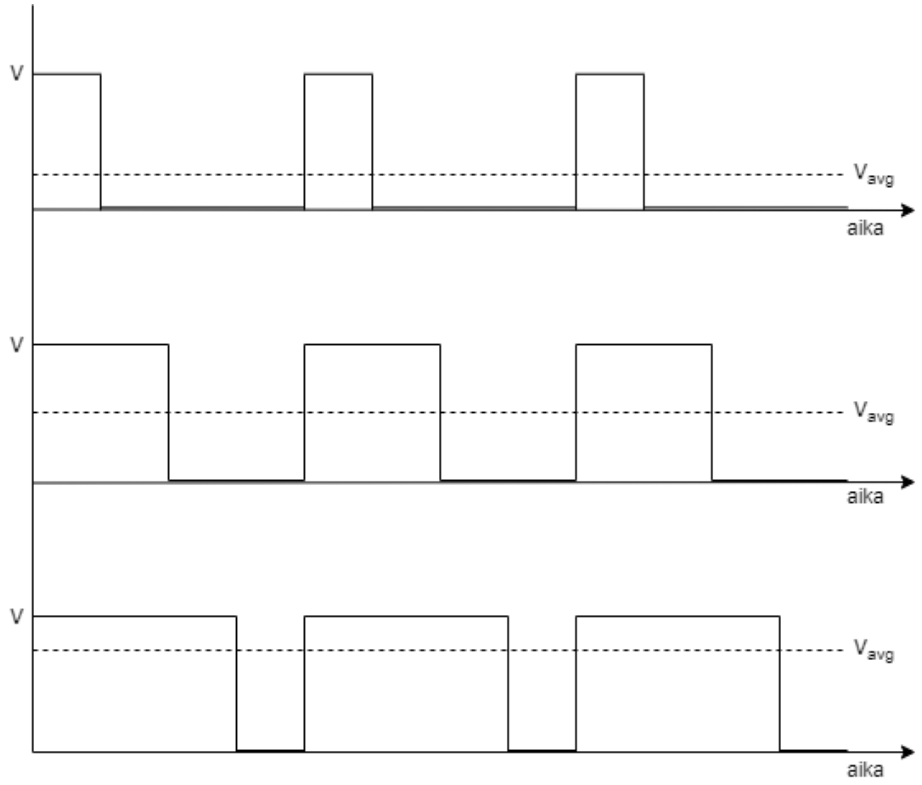
Kuva 21. Ajastimen toiminta ei-invertoivassa tilassa.

Arvolla 11 toiminta on invertoivaa, eli se toimii käänteisesti edelliseen toimintaan verrattuna. Tätä on havainnollistettu kuvassa 22.



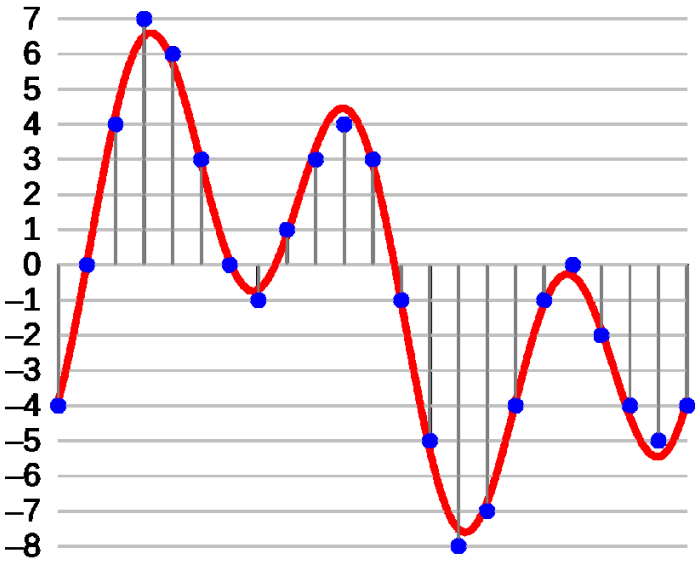
Kuva 22. Ajastimen toiminta invertoivassa tilassa.

Kun vertailuarvoa muuttaa, PWM-signaalin pulssisuhde muuttuu, jolloin myös ulostuleva keskijännite muuttuu. Tätä on havainnollistettu kuvassa 23.



Kuva 23. Pulssinleveysmodulaatio eri pulssisuhteilla.

Ääniaalto on useimmiten eritaajuisten äänikomponenttien summa. Ottamalla äänestä näytteitä näytteenottotaajuudella, saadaan äänisignaali muutettua kvantisoiduksi signaaliksi esimerkiksi kuvan 24 havainnollistamalla tavalla.



Kuva 24. Analoginen signaali ja sen 4-bittinen kvantisointi [45].

Kun PWM-signaalia ohjaavan laskurin vertailuarvoksi asetetaan näytteenottotaajuudella näytteen arvo, mikä on esitetty kuvassa sinisillä pisteillä, ja ulostulosignaali suodatetaan kondensaattorilla, saadaan alkuperäistä äänisignaalia, mikä on esitetty kuvassa punaisella, mukaileva signaali tuotettua ohjelmallisesti. Äänisignaalin tuottamisesta ohjelmallisesti kerrotaan enemmän luvussa 5.

ATmega328PB-mikrokontrollerin I/O-liittimen suurin sallittu virta on 40 mA [35]. Käytettäessä mikrokontrolleria 5 V jännitteellä, suurin sallittu ulostuloteho olisi

$$P = U * I = 40\ \text{mA} * 5\ \text{V} = 200\ \text{mW}.$$

Valitun kaiuttimen nimellisteho on 2 W, joten mikrokontrollerin pinnin ulostuloteho ei riitä kaiuttimelle. Mikrokontrollerin äänisignaali tulee viedä kaiuttimelle vahvistimen kautta.

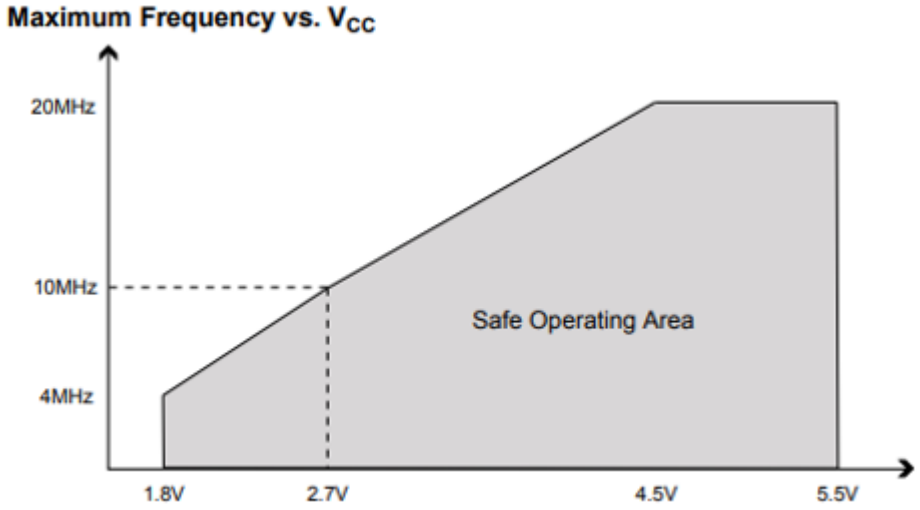
Laitteeseen valitaan PAM8303C-audiovahvistin, joka tuottaa enintään 3 W tehoa ja sen käyttöjännite on 2,8–5,5 V [46]. Audiovahvistimen sisäänmenon ja mikrokontrollerin ääniulostulon väliin kytketään vielä potentiometri, jolla voidaan säädellä äänenvoimakkuutta resistanssia muuttamalla.

3.6 Käyttöjännitteet

Power over Ethernet -kytkentä suunnitellaan tuottamaan 5 V ulostulojännite. Valitun Ethernet-kontrollerin suositeltu käyttöjännite on kuitenkin 3,3 V, joten PoE-kytkennän ulostulojännitettä ei voida suoraan käyttää Ethernet-kontrollerissa [20]. Mikrokontrollerin käyttöjännitealue sen sijaan on laaja, sitä voidaan käyttää alhaisimmillaan 1,8 V:lla ja korkeimmillaan 5,5 V:lla [35]. Mikrokontrollerin käyttöjännitteeksi voidaan valita myös

Ethernet-kontrollerin käyttämä 3,3 V, jolloin mikropiirien välillä olevaa SPI-väylää voidaan myös käyttää samalla käyttöjännitteellä.

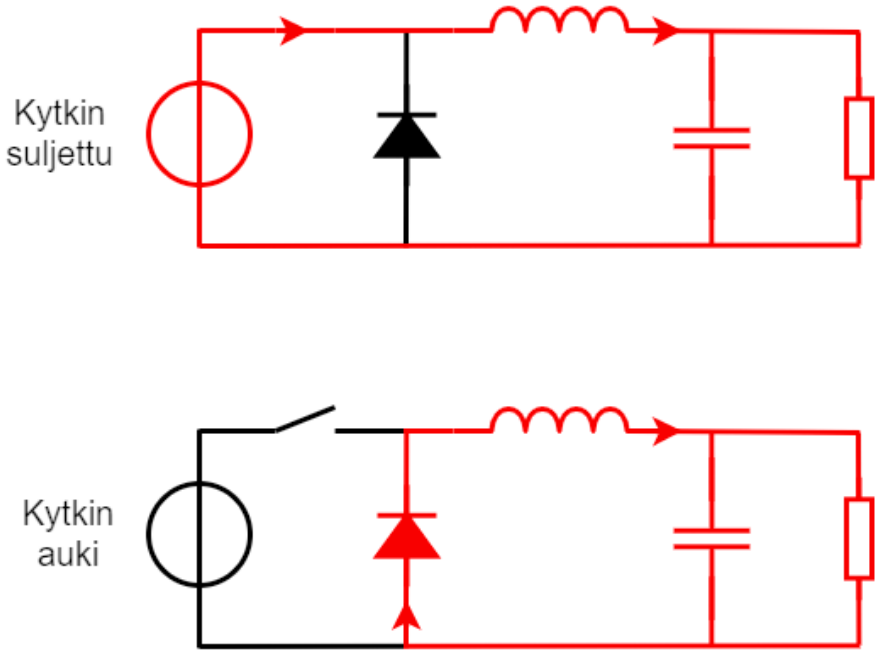
Mikrokontrollerin kellotaajuutta valittaessa on kuitenkin huomioitava, että mikrokontrollerin tukeman maksimitaajuutta rajoittaa sen käyttöjännite, mitä on havainnollistettu kuvassa 25.



Kuva 25. ATmega328PB-kontrollerin turvallinen taajuus suhteessa käyttöjännitteeseen [32].

Kuvasta huomataan, että mikrokontrollerin maksimitaajuus on 20 MHz, mutta 3,3 V:llä suurin turvallinen taajuus on noin 13 MHz. Audiovahvistimen datalehden [46] mukaan 3,2 V käyttöjännitteellä ja 8 ohmin impedanssilla sen suurin teho on 0,7 W, joka on huomattavasti pienempi kuin valitun kaiuttimen 2 W nimellisteho. 5 V käyttöjännitteellä suurin teho on 1,8 W, joka sopii valitulle kaiuttimelle paremmin. PoE-kytkennän 5 V jännitettä voidaankin käyttää suoraan audiovahvistimessa.

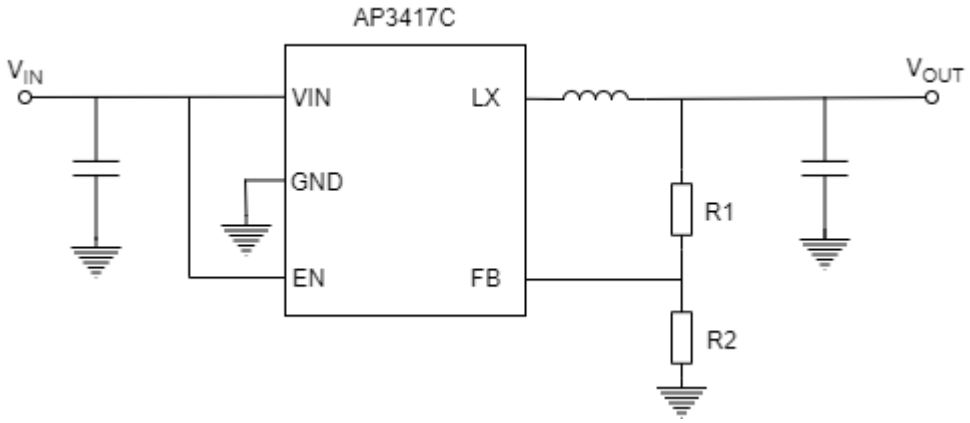
Mikrokontrollerin ja Ethernet-kontrollerin käyttöjännitteet saadaan 5 V:stä jännitettä pienentämällä. Tähän käytetään step-down buck-hakkuria. Buck-hakkurin toimintaperiaate on esitetty kuvassa 26.



Kuva 26. Buck-hakkurin toiminta.

Buck-hakkuri toimii käytännössä samalla tavalla kuin aiemmin esitelty flyback-muunnin, mutta siinä ei ole galvaanista eristystä. Kun kytkin on suljettu, jännitelähteestä varastoituu energiaa kelaan ja kuorma saa siitä suoraan virtansa. Tällöin diodi on estosuuntainen jännitelähteeseen nähden estosuunnassa. Kun kytkin avataan, alkaa kela purkamaan varastoitunutta energiaa tuottamalla virtaa kuormaan. Kytkennässä on kondensaattori suodattamaan ja tasoittamaan ulostulojännitettä. [47]

Buck-hakkuriksi valitaan AP3417C-mikropiiri, jonka esimerkkikytkentä on esitetty kuvassa 27.



Kuva 27. Buck-hakkurin esimerkkikytkentä.

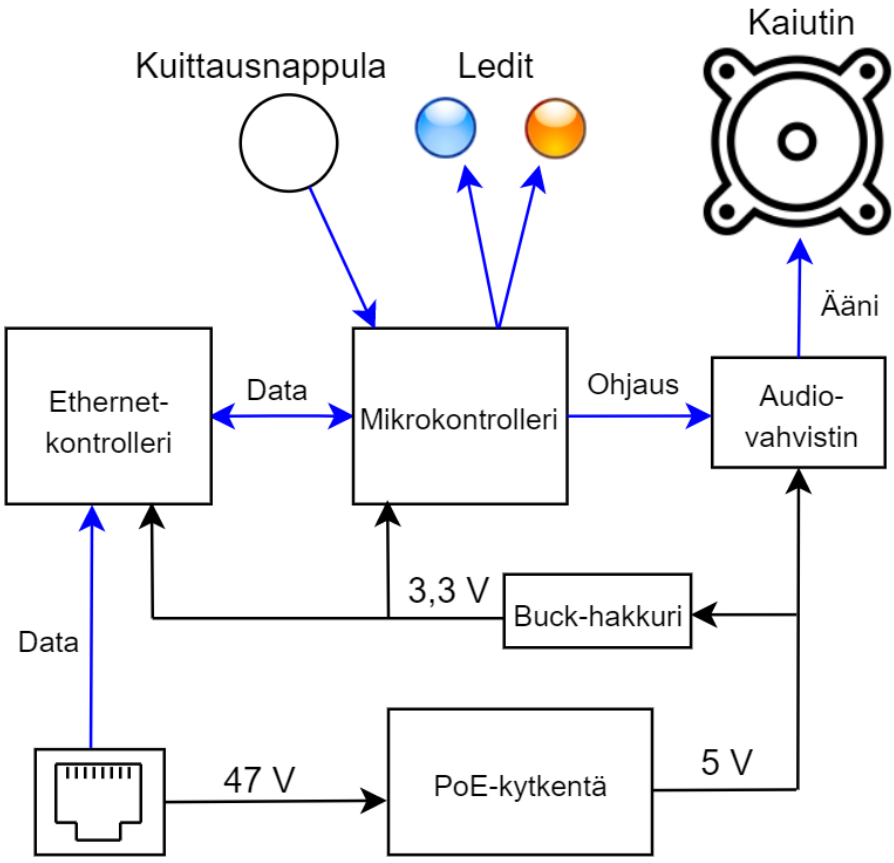
Mikropiirissä on EN-pinni (Enable), joka tulee asettaa aktiiviseksi, kun hakkuri halutaan pitää toiminnassa. Lisäksi siinä on takaisinkytkentäpinni FB, jolla se säätelee kytkimen

päälläoloaikaa pitääkseen ulostulojännitteen halutulla tasolla. Ulostulojännitteen määrää vastusten R1 ja R2 suhde.

Valittu Buck-hakkuripiiri pystyy datalehden mukaan tuottamaan enintään 1 A virtaa 2,5–5,5 V jännitteellä [48]. Ethernet-kontrolleri kuluttaa lähetystilassa virtaa tyypillisesti 132 mA ja mikrokontrolleri 8 MHz kellotaajuudella ja 5 V käyttöjännitteellä enintään 10 mA. Niiden lisäksi virtaa käyttää esimerkiksi 2 lediä, jotka 330 ohmin resistanssilla ja 3,3 V käyttöjännitteellä käyttävät päällä ollessaan noin 11 mA virtaa. Komponenttien yhteenlaskettu virrankulutus on noin 165 mA, joten valittu buck-hakkuri sopii toteutukseen.

4. LAITTEEN VALMISTUS

Tässä luvussa kerrotaan laitteen piirikaavion ja pohjapiirroksen suunnittelusta, sekä laitteen valmistuksesta. Kuvassa 28 on esitetty laitteen toimintaa kuvaava lohkokaavio, jonka perusteella laitteen piirikaavio ja layout on suunniteltu.



Kuva 28. Laitteen toimintaa kuvaava lohkokaavio.

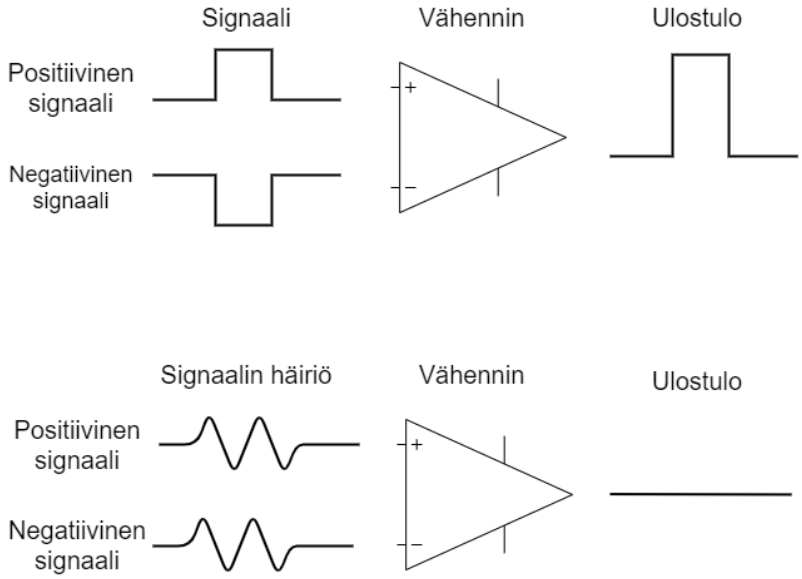
4.1 Fyysisen laitteen suunnittelu

Laitteen piirikaavio ja layout suunnitellaan PADS Logic -ja Layout -ohjelmilla. Laitteen koolle ei ole annettu vaatimuksia, mutta laite pyritään suunnittelemaan mahdollisimman pieneksi.

Ovikellolaitteessa on osia, jotka ovat herkkiä sähkömagneettisille häiriöille. On joitain suunnitteluperiaatteita, joilla voidaan minimoida laitteen sähkömagneettisia häiriöitä. Esimerkiksi mikropiireille kellotaajuuden tuottavat kideoskillaattorit ovat herkkiä häiriöille, joten niiden ympärillä tulisi olla maataso, eikä lähellä tulisi olla muita signaaleja. [49]

Ethernet-yhteys on myös suhteellisen herkkä häiriöille [50]. Ethernet käyttää differentiaalisia signaaleja, eli informaation siirtämiseen käytetään kahta,

vastakkaismerkkistä signaalia, millä voidaan parantaa signaalin laatua kuvan 29 havainnollistamalla tavalla.



Kuva 29. Häiriön poistaminen differentiaalisesta signaalista.

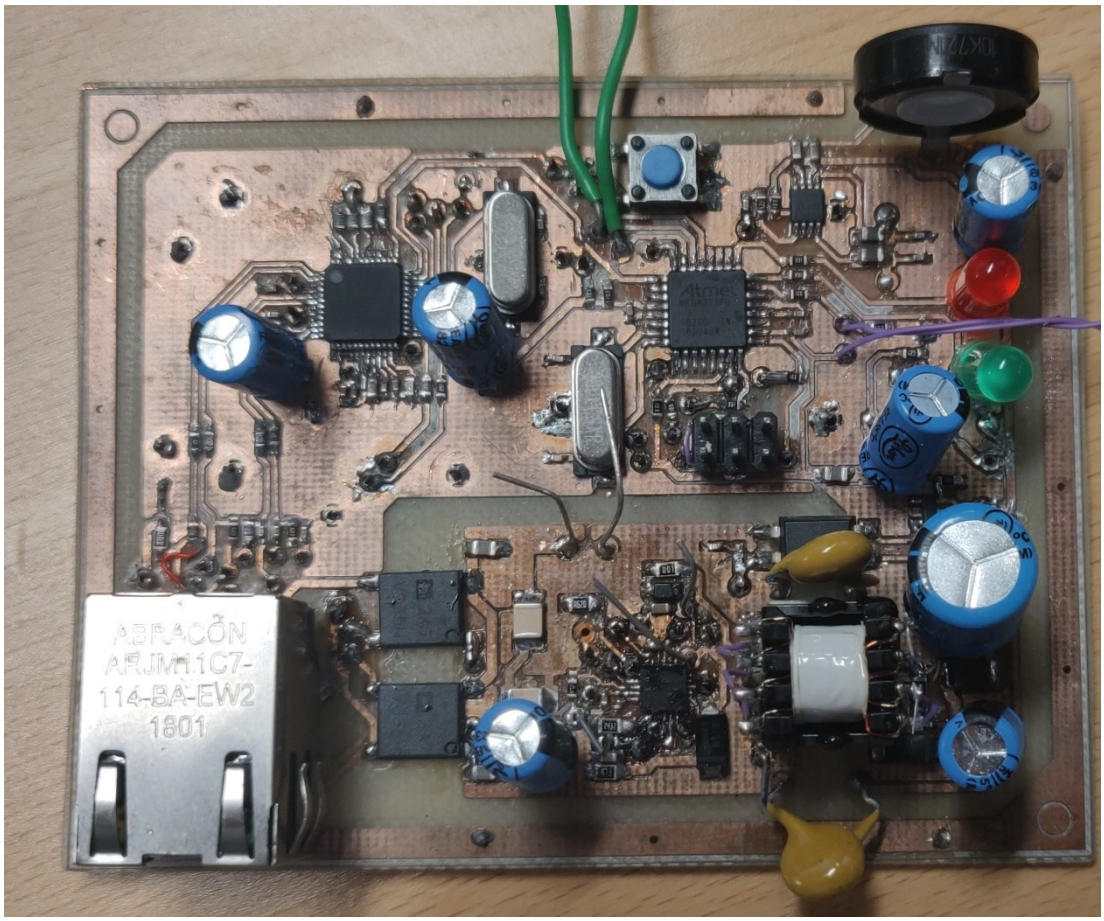
Signaalissa on informaatiota sekä esimerkiksi muiden lähellä olevien signaalien aiheuttamia häiriöitä. Ideaalisessa tilanteessa aiheutunut häiriö on kummassakin signaalissa samanlainen. Kun negatiivinen signaali vähennetään positiivisesta signaalista, informaation signaali voimistuu (kuvassa ylhäällä). Aiheutunut häiriö on kuitenkin suurin piirtein samanlaista kummassakin signaalissa, jolloin ne kumoutuvat, kun ne vähennetään toisistaan (kuvassa alhaalla).

Jotta häiriö saataisiin mahdollisimman samanlaiseksi kummassakin johtimessa, tulee johtimet sijoittaa mahdollisimman lähelle toisiaan. Johtimiin aiheutuvan häiriön määrää voidaan myös vähentää pitämällä johdinpituudet mahdollisimman lyhyinä. Saapuvien ja lähtevien Ethernet-signaalien johtimet kannattaa myös eristää maatasolla, jotta ne aiheuttaisivat toisiinsa mahdollisimman vähän häiriötä. [50]

PoE-kytkennän ja muun kytkennän väliin jätetään myös hieman suurempi eristeväli, jotta PoE-spesifikaation mukainen galvaaninen eristys toteutuisi.

4.2 Ensimmäinen prototyyppi

Ensimmäisen prototyypin piirilevy valmistettiin jyrsimällä kuparilevystä ylimääräinen kupari, jossa jäljelle jää toivotut kuparijohtimet ja alustat komponenteille. Koska PoE-kontrolleri Si3404:n pohjassa on padi, se juotettiin vielä kuumailmalla paikoilleen. Kuvassa 30 on laitteen ensimmäinen prototyyppi.



Kuva 30. Ensimmäinen prototyyppi.

Jyrsimällä tehdyssä piirilevyssä ei ole pinnoitetta ja läpivienneille tarkoitetut poratut reiät tuli juottaa käsin metallilankaa käyttämällä. Juottaminen oli hieman haasteellista pinnoittamattomalle levylle, etenkin PoE-kytkennässä, missä komponentit ja läpiviennit olivat aseteltu erittäin lähelle toisiaan.

Valmis prototyyppi saatiin toimimaan PoE-kytkentää lukuun ottamatta muutamalla korjauksella. Osa kytkennöistä korjattiin hyppylangoilla ja johtimia katkaisemalla, sekä juottamisesta aiheutuneita epätoivottuja oikosulkuja rapsutettiin piirilevystä pois veitsen ja mikroskoopin avulla. Yksi suurempi virhe tapahtui, kun viime hetkellä päätettiin, että flyback-muuntimeksi vaihdetaan suurempitehoinen malli. Tämän muuntimen fyysinen koko olikin alkuperäistä suurempi eikä se mahtunut piirilevylle oikein päin. Muunnin piti asettaa piirilevylle ylösalaisin ja juottaa hyppylangat piirilevyn padeista komponentin jalkoihin.

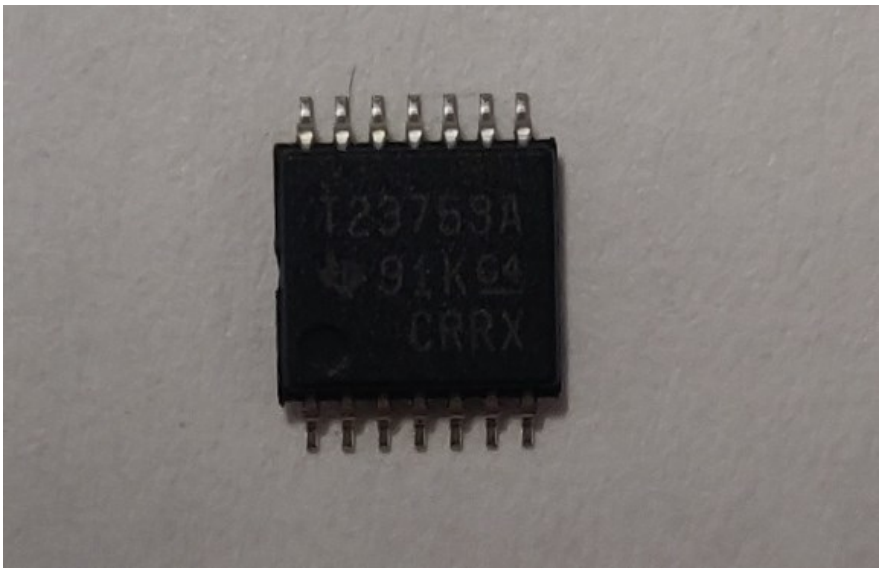
Power over Ethernetiä ei saatu huolellisen vianjäljityksenkään jälkeen toimimaan. Koska PoE-kontrollerin alla olevaan padi on piirilevyä vasten, sen kontaktia piirilevyyn ei voida tarkistaa. Komponentin muutkaan kontaktit eivät tule kotelosta ulos erillisinä jalkoina,

vaan ovat esillä vain sivulta, joten juotosten toimivuutta oli mikroskooppiakin käyttämällä vaikea tarkistaa.

Piirilevyjä valmistettiin kaksi kappaletta, ja myös toisen piirilevyn PoE-kytkentä yritettiin saada toimimaan, mutta se ei onnistunut. Koska valittu PoE-kontrolleri oli hyvin haastava, päätettiin laitteeseen kokeilla toista PoE-kontrolleria.

4.3 Toinen prototyyppi

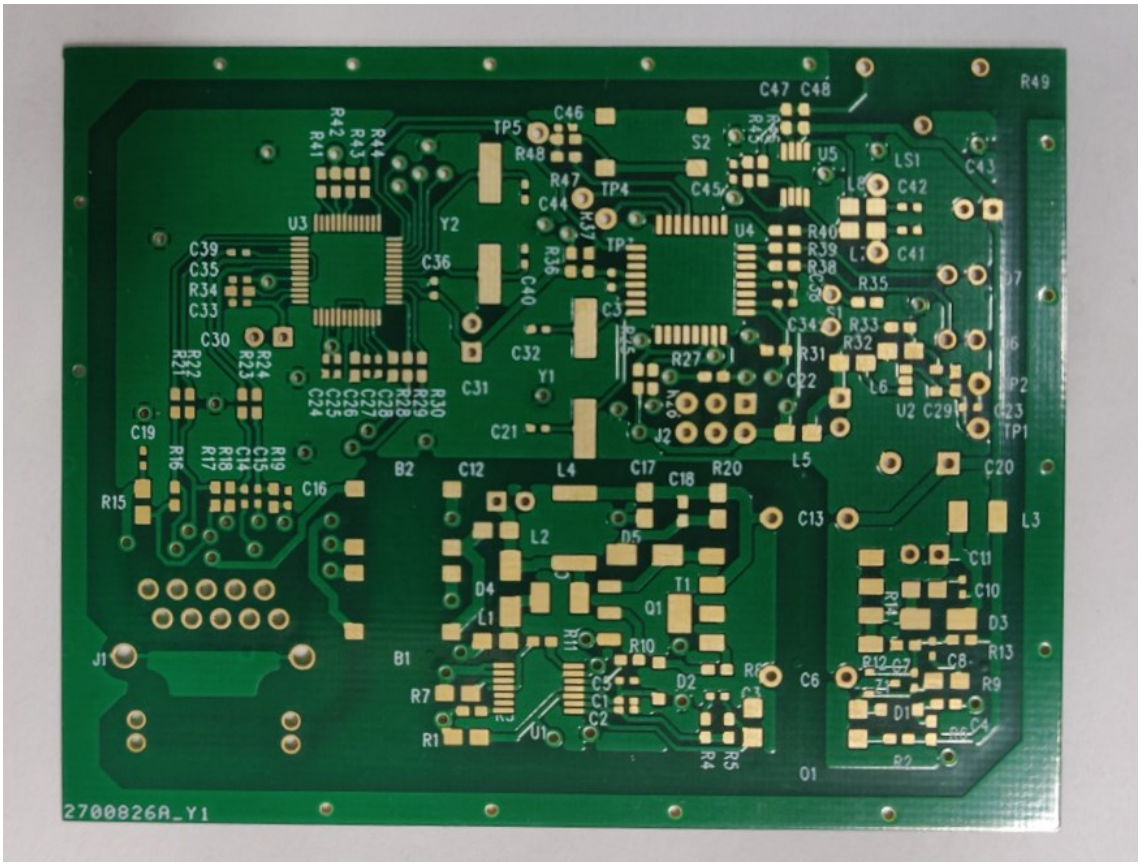
Toisen PoE-kontrollerin tärkeimpänä kriteerinä on helpompi juotettavuus. Kuvassa 31 näkyvässä PoE-kontrollerissa TPS23753A on helpommin juotettava kotelo, joten se valittiin laitteen uudeksi PoE-kontrolleriksi.



Kuva 31. TPS23753A-mikropiiri.

Valittu kontrolleri eroaa aiemmasta Si3404-kontrollerista jonkin verran. Si3404:ssa on 21 pinniä sekä esimerkiksi integroitu kytkin halutun jännitteen ylläpitämiseksi. TWPS23753A:ssä on 14 pinniä ja kytkimeksi tarvitaan erillinen transistori. Eroavaisuuksien takia PoE-kytkentä joudutaan suunnittelemaan uudelleen, mutta kytkennöissä on jonkin verran samankaltaisuuksia ja toimintaperiaate on samanlainen, joten uuden kytkennän suunnittelua ei tarvitse aloittaa alusta asti. Uusi PoE-kontrolleri on kuitenkin hieman isompi kuin vanha, sekä suuren tehon kestävä transistori on myös iso, joten kytkennän mahduttaminen samaan tilaan vaati hieman työtä.

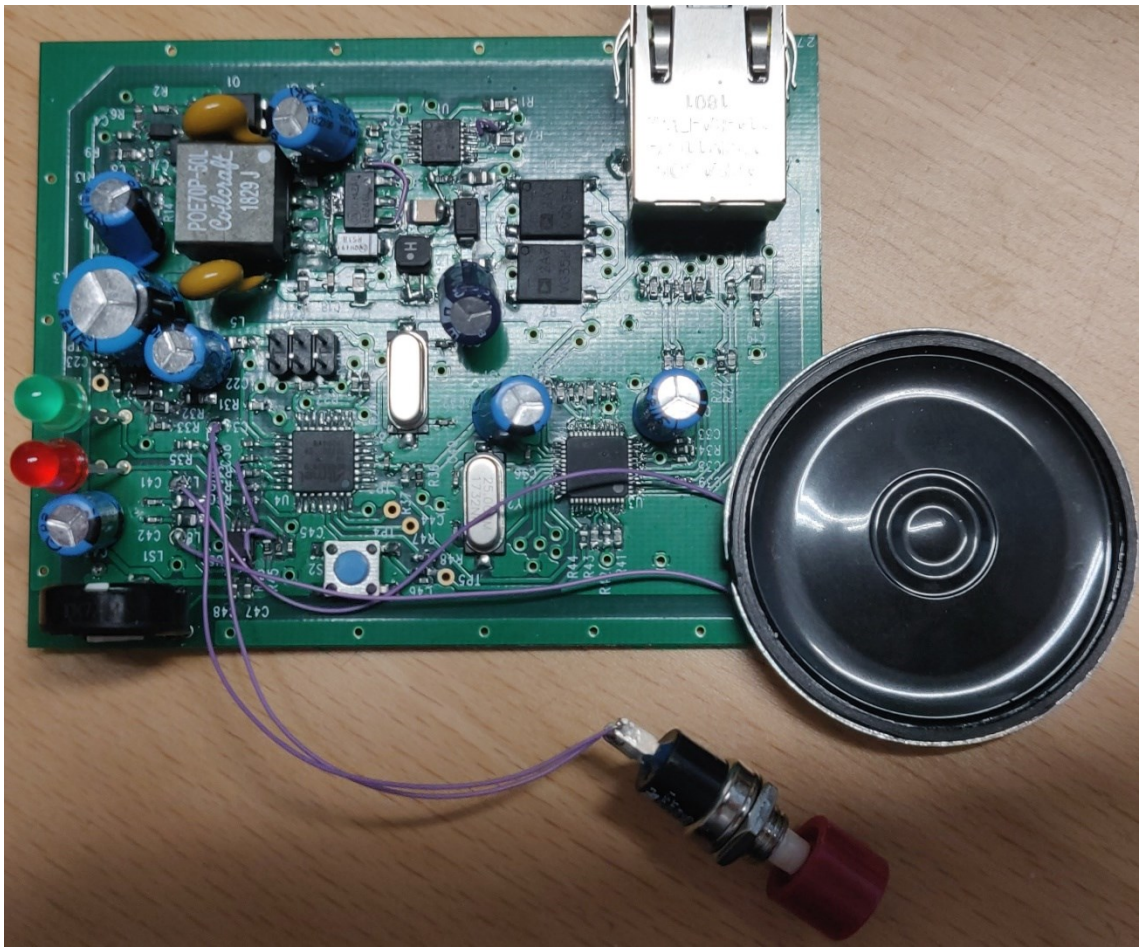
Toinen piirilevy tilattiin kaupallisesta piirilevytehtaasta ja piirilevylle saatiin piirilevyä suojaava juotosmaski. Juotosmaski peittää kaikki johtimet komponenttien padeja lukuun ottamatta, mikä estää johtimia hapettumasta sekä suojaa niitä tahattomilta oikosuluilta [51].



Kuva 32. ENIG-pinnoitettu piirilevy.

Komponenttien padeille voitiin myös valita pinnoite, mikä suojaa johtimia hapettumiselta ja helpottaa komponenttien juottamista [52]. Pinnoitevaihtoehtoja ovat esimerkiksi kuvassa 32 näkyvä ENIG-pinnoite (Electroless nickel immersion gold, kulta-nikkelipinnoite), jossa nikkelöity piirilevy upotetaan sulaan kultaan, sekä HASL-pinnoite (Hot air solder leveling), jossa piirilevy upotetaan sulaan tinaan ja puhalletaan kuumalla ilmalla, jolloin padeille jää kerros tinaa [53]. Pinnoitteeksi valitaan ENIG, koska sillä saadaan tasaisempi piirilevyn pinta kuin HASL-pinnoitteella [54].

Piirilevyyn saatiin myös tekstikerros, niin kutsuttu silkkipaino (silkscreen), jonka avulla komponentit voidaan piirilevyllä numeroida, mikä helpottaa komponenttien organisointia juotosvaiheessa. Valmis prototyyppi on näytetty kuvassa 33.



Kuva 33. Kuva toisesta prototyypistä.

Koska tehtaan piirilevynvalmistustekniikassa piirilevy kuparoidaan levyn poraamisen jälkeen, tuli myös piirilevyn läpiviennit automaattisesti tehtyinä, mikä helpotti toisen prototyypin juottamisessa. Pinnoitetulle levylle oli myös huomattavasti helpompi juottaa, eikä epätoivottuja oikosulkuja aiheutunut. Toinen prototyyppi saatiin toimimaan toivotusti.

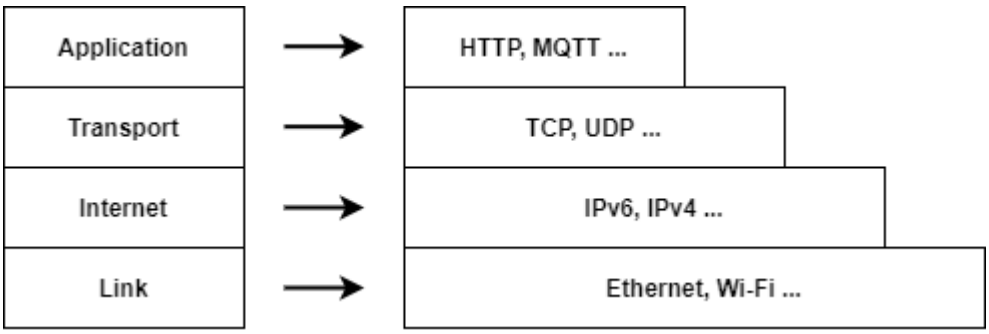
5. LAITTEEN OHJELMA

Tässä luvussa kerrotaan laitteen käyttämän ohjelman toteutuksesta, sen siirtämisestä mikrokontrollerille, sekä miten ohjelma hyödyntää mikrokontrollerin ominaisuuksia esimerkiksi kuittausnappulan tilan seuraamiseen sekä äänen tuottamiseen. Luvussa kerrotaan myös laitteen käyttämistä tiedonsiirtoprotokollista.

Laitteessa oleva ohjelma tehtiin C- ja C++-ohjelmointikielillä. Koska C:llä tehty ohjelma käyttää vähemmän muistia ja on C++:aa nopeampi, tehdään suurin osa ohjelmasta C:llä [55]. Osa laitteen käyttämistä valmiista ohjelmistokirjastoista on kuitenkin tehty C++:lla, joten C++:aa täytyy käyttää ohjelman niihin osiin, jotka käyttävät näitä kirjastoja. C++:lla myös liitetään ohjelman C-kieliset osat C++:lla tehtyyn pääohjelmaan.

5.1 TCP/IP

Jotta ovikellon saa keskustelemaan internetissä muiden laitteiden kanssa, tulee laitteiden ensin muodostaa toisiinsa yhteys. Ovikellolaite käyttää yhteyden muodostukseen TCP/IP-protokollapinoa. TCP/IP-protokollapino koostuu neljästä niin kutsutusta kerroksesta (Layer), jotka on esitetty kuvassa 34. [56]



Kuva 34. TCP/IP-protokollapino.

TCP/IP-pinon alimmaisena on siirtokerros (Link layer), joka vastaa fyysisestä yhteydestä. Ovikellolaitteen siirtokerrosta vastaa Ethernet-yhteys.

Verkkokerros (Internet layer) ohjaa verkkopaketit oikeaan osoitteeseen. Tärkein protokolla tässä kerroksessa on IP-protokolla. IP-protokollasta on yleisesti käytössä kaksi versiota, IPv6 ja IPv4. Suurin ero protokollien välillä on se, että IPv6-osoitteet ovat 128 bittiä (2^{128}) pitkiä, kun IPv4-osoitteet ovat vain 32 bittiä (2^{32}) pitkiä [57]. Tulevaisuudessa vapaat IPv4-osoitteet tulevat loppumaan, kun taas IPv6-osoiteavaruus tulee riittämään vielä pitkäksi aikaa [58]. Toteutukseen valitaan kuitenkin IPv4, sillä Ethernet-kontrolleri tukee vain IPv4:ää [20].

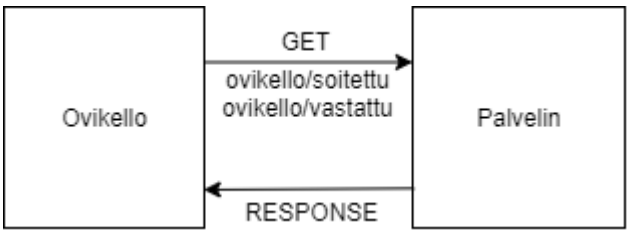
Kuljetuskerros (Transport Layer) vastaa siitä, miten verkkopaketit ohjataan perille. Yleisimmät protokollat tässä kerroksessa ovat TCP ja UDP (User Datagram Protocol). TCP mahdollistaa luotettavan tiedonsiirron, joka myös pitää huolen siitä, että siirrettävät paketit saapuvat perille oikeassa järjestyksessä. Protokolla mahdollistaa myös hävinneiden pakettien uudelleenlähettämisen. [56]

UDP on yhteydetön protokolla, jossa pakettien perillemenoa ei varmisteta. Siitä johtuen UDP-paketit ovat hieman pienempiä kuin TCP-paketit ja mahdollistavat pienemmän viiveen, mutta se sopii käyttöön vain sellaisissa sovelluksissa, missä jokaisen paketin perille saapuminen ei ole välttämätöntä. Tällaisia sovelluksia ovat esimerkiksi videon tai äänen välittäminen. [59]

Sovelluskerros (Application Layer) mahdollistaa käyttäjän, joko ihmisen tai laitteen, ottamaan yhteyden internetiin erilaisia protokollia käyttäen. Sovelluskerroksen protokollia ovat esimerkiksi Internet-selaamisessa käytetty HTTP (Hypertext Transfer Protocol) sekä MQTT (MQ Telemetry Transport). [56]

5.2 HTTP-protokolla

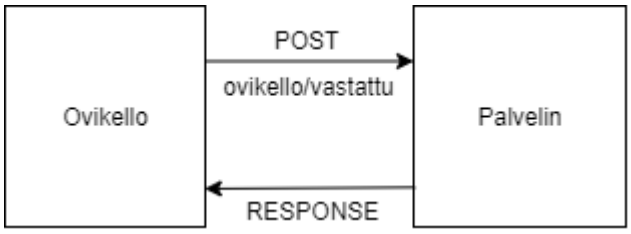
HTTP on asymmetrinen pyyntö-vastaus-protokolla (request-response) käyttäjän ja palvelimen välillä, jossa palvelin kuuntelee tietoliikennettä ja vastaa käyttäjän pyyntöihin, kun ne tulevat palvelimelle. Pyyntöjä ovat esimerkiksi GET, jolla voidaan pyytää jotain resurssia, kuten internet-sivun sisältöä, sekä POST, jolla voidaan lähettää tietoa palvelimelle. Esimerkiksi näitä pyyntöjä käyttämällä ovikellolaitteen tietoliikenne voitaisiin toteuttaa. Ovikello voi käyttää GET-komentoa ovikellon tilan kysymiseen palvelimelta kuvan 35 havainnollistamalla tavalla.



Kuva 35. Hälytyksen tilan saaminen HTTP:lla.

Huono puoli pyyntö-vastaus-protokollan käytössä on se, että palvelin ei automaattisesti lähetä tietoa ilmoittautumislaitteelta eteenpäin tilan muuttuessa, vaan sitä pitää erikseen pyytää. Tästä syystä ilmoittautumislaitteen tilaa tulee kysyä riittävän usein pienen viiveen mahdollistamiseksi, mistä myös aiheutuu suuri määrä ylimääräistä tiedonsiirtoa.

Ovikellon tilan lähettäminen on sen sijaan toimivampaa. Kuvassa 36 on esitetty esimerkki POST-komennon käytöstä ovikellon tilan lähettämiseen.



Kuva 36. Ovikellon tilan lähettäminen HTTP:lla.

POST-komennolla ovikellon tila voidaan lähettää juuri silloin kuin se muuttuu, eikä palvelimen tarvitse kysyä sitä erikseen.

5.3 MQTT-protokolla

Sovelluskerroksella sijaitsee myös MQTT-protokolla, joka on yksinkertainen ja kevyt tiedonsiirtoprotokolla. Sen periaatteina on minimoida verkon ja laitteen suorituskyvyn käyttö, yrittäen samalla taata yhteyden luotettavuus [60]. Tämän ansiosta protokolla soveltuu hyvin IoT-laitteiden käyttöön, missä pieni verkon käyttö ja sähkönkulutus ovat tärkeitä.

MQTT perustuu publish-subscribe-malliin, jossa laitteet välittävät viestejä toisilleen MQTT-välittäjänä toimivaa palvelinta käyttäen. Laitteet voivat julkaista (publish) viestejä tietyllä aiheella lähettämällä ne MQTT-välittäjälle, joka toimittaa viestit eteenpäin niille laitteille, jotka ovat tilanneet (subscribe) kyseisen aiheen. Toisin kuin HTTP:ssa, palvelin välittää viestit muille laitteille suoraan, ilman että laitteiden tarvitsee kysellä tilaa erikseen. MQTT-viestinnässä laitteen lähettämille viesteille lähetetään yleensä vastaukseksi hyväksyntäviesti (acknowledge). [61]

Yhteys muodostetaan CONNECT-paketilla, jolla voidaan määrittää myös esimerkiksi käyttäjänimi ja salasana, sekä mahdollisesti viesti, jonka MQTT-välittäjä lähettää, jos yhteys laitteeseen katkeaa. Palvelin vastaa CONNACK-paketilla, mikäli palvelin hyväksyy CONNECT-paketin. [62]

Yhteyden muodostamisen jälkeen laite voi tilata aiheita tai alkaa suoraan julkaisemaan viestejä. SUBSCRIBE-paketissa määritetään haluttu tai useampi haluttu aihe, sekä esimerkiksi suurin laitteen hyväksymä QoS. MQTT-välittäjä vastaa viestiin SUBACK-paketilla (Subscribe acknowledgement), jolla ilmoitetaan, että viesti on tullut perille ja se on hyväksytty.

Viestien lähettäminen tapahtuu PUBLISH-paketilla, johon voidaan määrittää muun muassa aihe, valinnainen viesti, sekä käytettävä QoS-tila. Viestiin voidaan myös määrittää retain-lippua käyttämällä, mikäli MQTT-välittäjän halutaan säilyttävän

viimeinen viesti, joka toimitetaan automaattisesti laitteille, jotka tilaavat aiheen ensimmäistä kertaa.

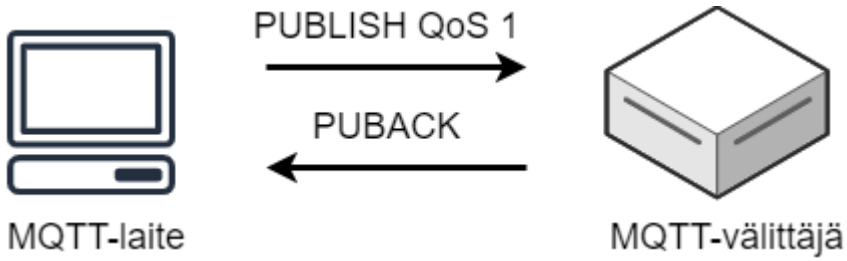
5.3.1 Quality of Service

MQTT-protokolla käyttää eritasoisia QoS-tiloja, joilla voidaan vaikuttaa siihen, millä varmistuksella viestejä lähetetään. MQTT:ssä on 3 eri QoS-tilaa:

- QoS 0: Toimita enintään kerran
- QoS 1: Toimita vähintään kerran
- QoS 2: Toimita täsmälleen kerran.

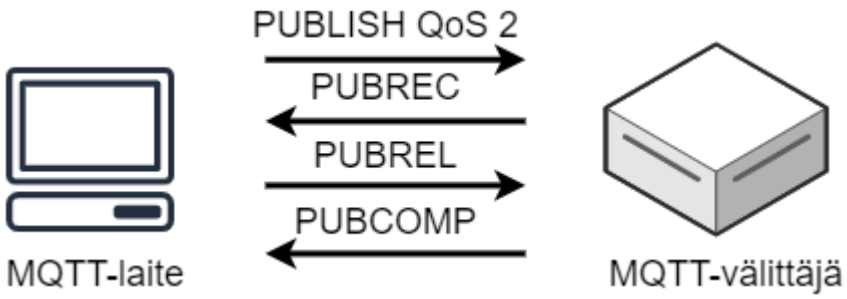
Kuva 37. Quality of Service arvolla 0.

Kuvassa 37 on esitetty lähetettävä viesti QoS arvolla 0. Tällöin viesti lähetetään vain kerran, riippumatta siitä, vastaanottaako MQTT-välittäjä sitä. Tilaa voidaan käyttää esimerkiksi silloin, kun joka viestiä ei ole välttämätöntä saada [62]. Tällainen voi olla esimerkiksi lämpötila-anturi.



Kuva 38. Quality of Service arvolla 1.

Kuvassa 38 on esitetty lähetettävät viestit käytettäessä QoS arvoa 1. Siinä viesti lähetetään niin monta kertaa, kunnes MQTT-välittäjä vastaa viestiin lähettämällä hyväksyntäviestin (PUBACK). Tätä tilaa käytettäessä viestin toimitus perille varmistetaan, mutta on myös mahdollista, että laite ehtii lähettää viestin uudelleen senkin jälkeen, kun palvelin on vastannut viestiin. Tästä seuraa duplikaattiviesti, joten tilaa ei tule käyttää silloin, kun duplikaateista voi olla haittaa. [62]



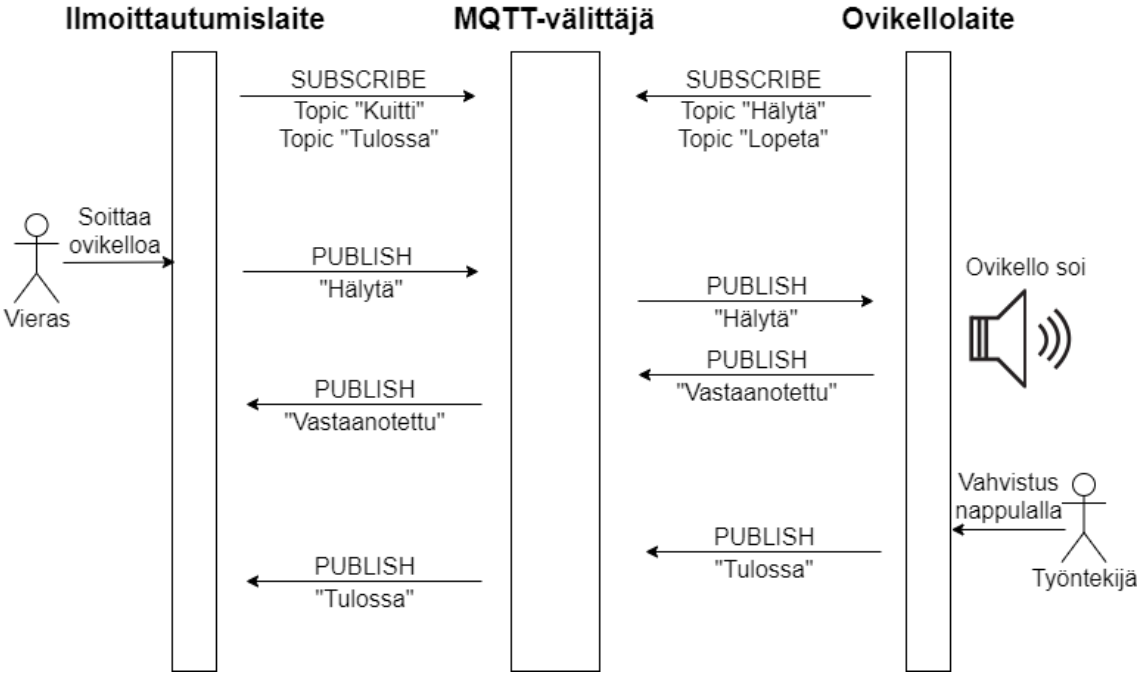
Kuva 39. Quality of Service arvolla 2.

Kuvassa 39 on esitetty lähetettävät viestit QoS arvolla 2. QoS 2:lla voidaan varmistaa, että viesti lähetetään täsmälleen kerran. Laite lähettää PUBLISH-paketteja niin kauan, kunnes MQTT-välittäjä vastaa PUBREC-paketilla (Publish received, julkaisu vastaanotettu). Tähän laite vastaa PUBREL-paketilla (Publish release). Vasta, kun MQTT-välittäjä vastaanottaa PUBREL-paketin, se voi olla varma, että viesti on saapunut tarkalleen yhden kerran. Tähän laite vastaa PUBCOMP-paketilla (Publish complete). Laite lähettää PUBREL-paketteja uudelleen niin kauan, kunnes se vastaanottaa PUBCOMP-paketin. [62]

5.3.2 MQTT ovikellolaitteessa

Ovikellolaitteen tarvitsee tietää, milloin vierailijat painavat ilmoittautumislaitteella ovikellonäppäintä ja sen tulee ilmoittaa, kun ovikellolaitteen hyväksymisnappia painetaan merkiksi siitä, että vierailijaa tullaan hakemaan.

Koska MQTT:ssa julkaistut viestit välitetään eteenpäin ilman vahvistusta siitä, kenelle viestit on toimitettu, ei edes QoS:ia käyttämällä voida varmistua siitä, että jokainen ilmoittautumislaitteelta lähetetty viesti kulkeutuisi ovikellolaitteeseen asti. Viestin vastaanottaminen voidaan varmistaa lähettämällä vahvistusviesti takaisin ilmoittautumislaitteelle, kun ovikellolaite saa ilmoituksen ovikellon soitosta. Esimerkki toiminnasta on esitetty kuvassa 40.



Kuva 40. MQTT:n käyttö ovikellossa.

Kun ovikellolaite vastaanottaa ”hälytä”-viestin, se vastaa ilmoittautumislaitteelle ”vastaanotettu”-viestillä, jolloin ilmoittautumislaite varmistuu siitä, että viesti on tullut perille. Ovikellolaite tilaa lisäksi ”Lopeta”-aiheen, jotta ovikellolaitteen voi sammuttaa myös MQTT-viestillä esimerkiksi silloin, kun käytössä on useampi ovikellolaite ja toinen ovikello vastaa vieraalle.

5.4 Valmiit ohjelmakirjastot

Laitteen ohjelmoimiseen on käytettävissä valmiita ohjelmakirjastoja. Esimerkiksi Arduinon Ethernet-kirjasto yhdessä Ethernet-kontrolleri W5500:n kanssa mahdollistaa yhteyden muodostamisen internetiin. Kirjastoa käyttämällä laite voi toimia sekä palvelimena että asiakkaana. Kirjasto mahdollistaa neljän samanaikaisen yhteyden käyttämisen [63]. Koska ovikellolaite käyttää MQTT:tä, sen tarvitsee muodostaa yhteys vain MQTT-välittäjään, joten laitteen ei tarvitse toimia palvelimena.

Ohjelmakirjasto on siis hieman monimutkaisempi kuin vaaditaan, jolloin kirjasto myös käyttää hieman enemmän muistia kuin on tarpeen. Pelkästään Ethernet-kirjaston käyttäminen internet-yhteyden muodostamiseen käyttää 10,6 kilotavua mikrokontrollerin noin 32 kilotavun ohjelmamuistin maksimimäärästä, sekä noin 300 tavua mikrokontrollerin 2048 tavun käyttömuistin maksimimäärästä.

Myös MQTT:lle on valmiita ohjelmakirjastoja, kuten Adafruitin tekemä Adafruit MQTT library [64]. Kirjaston kuvauksen mukaan kyseinen ohjelmakirjasto ei tue esimerkiksi QoS-tasoa kaksi, joten sitä on hieman kevennetty MQTT:n toiminnallisuutta karsimalla.

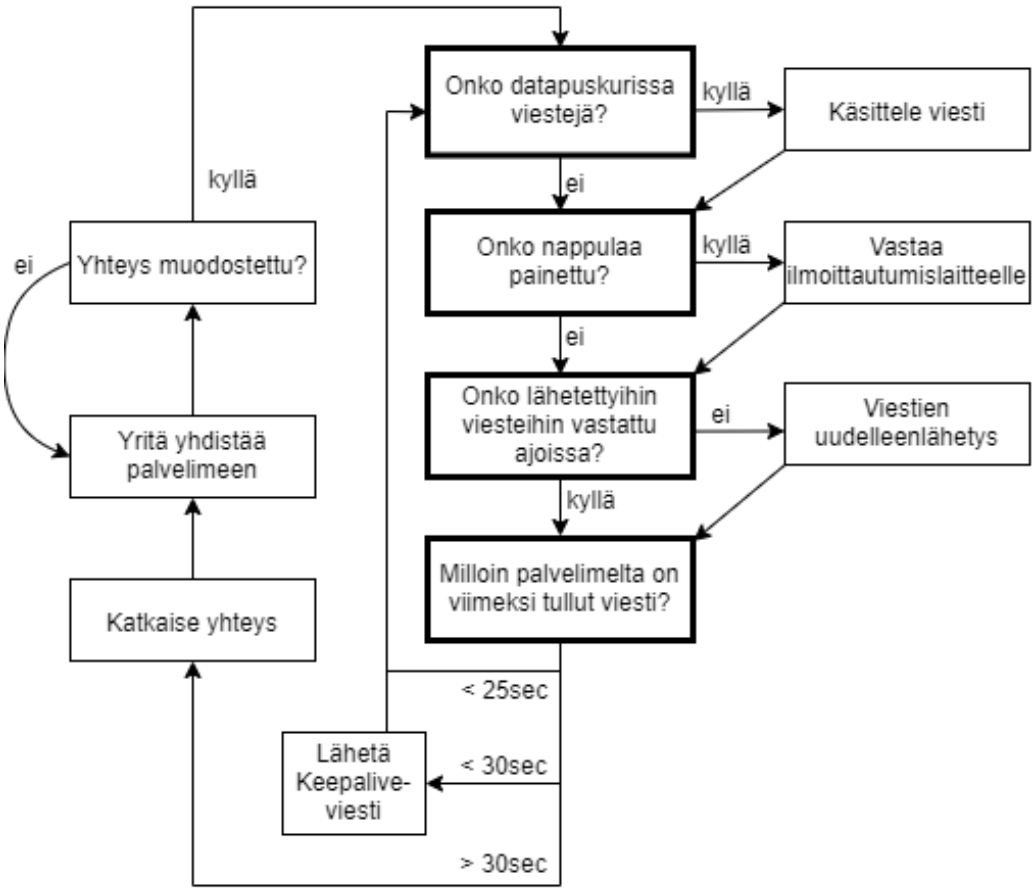
Kirjaston sisältämän esimerkin [65] perusteella MQTT-kirjasto käyttää noin 8,1 kilotavua ohjelmamuistia ja noin 930 tavua käyttömuistia. Koska kirjasto käyttää niin paljon muistia, päätetään MQTT-toiminnallisuus ohjelmoida itse.

Muistin säästämiseksi jätetään osa MQTT:n toiminnallisuudesta jättää pois. Esimerkiksi QoS 2:n vaatimia PUBREC, PUBREL ja PUBCOMP -viestejä, eikä myöskään virheellisistä viesteistä aiheutuvia virhekoodien lähettämistä toteuteta.

5.5 Ohjelmiston toiminta

Mikrokontrolleri on ohjelmoitu käynnistyessään ajamaan asetusfunktio, jonka jälkeen ohjelman suoritus siirtyy ikuisessa silmukassa olevaan pääohjelmaan.

Asetusfunktiossa asetetaan kuittausnappulaan kytketty pinni sisäänmenoksi, sallitaan napin painalluksesta aiheutuvat keskeytykset, sekä asetetaan ledejä sekä kaiutinta ohjaavat pinnit ulostuloiksi. Tämän jälkeen yhdistetään lähiverkkoon, sekä muodostetaan TCP- ja MQTT-yhteys MQTT-välittäjään. Lopuksi ohjelman suoritus siirtyy pääohjelmaan, jonka yksinkertaistettu toimintakaavio on esitetty kuvassa 41.

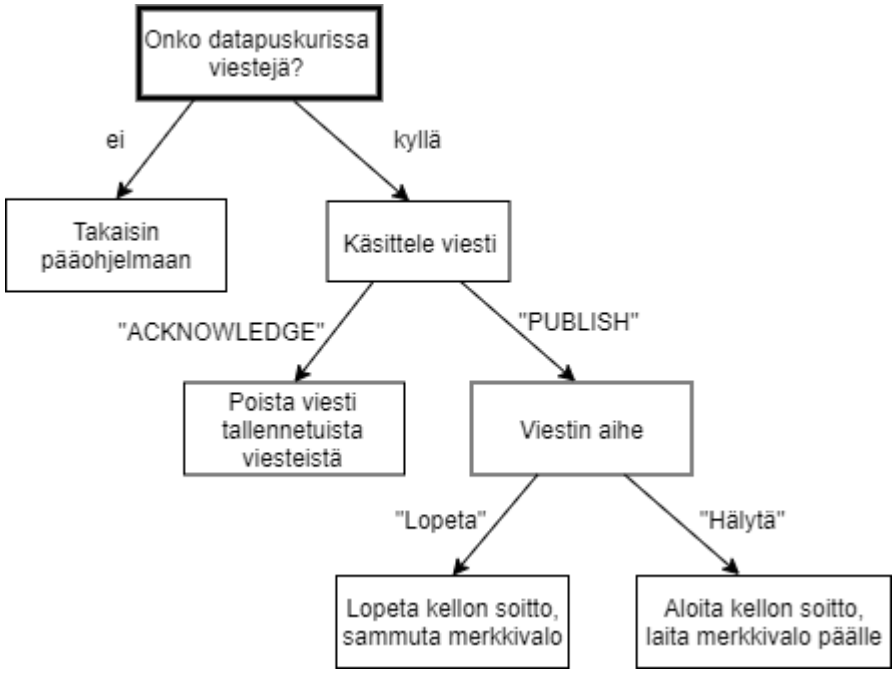


Kuva 41. Ohjelmiston yksinkertaistettu toimintakaavio.

Seuraavissa kappaleissa on kuvailtu ohjelman suoritusta tarkemmin.

5.5.1 Datapuskurin tarkistus

Datapuskurin tarkastus tapahtuu kuvan 42 osoittamalla tavalla.



Kuva 42. Datapuskurin tarkistus.

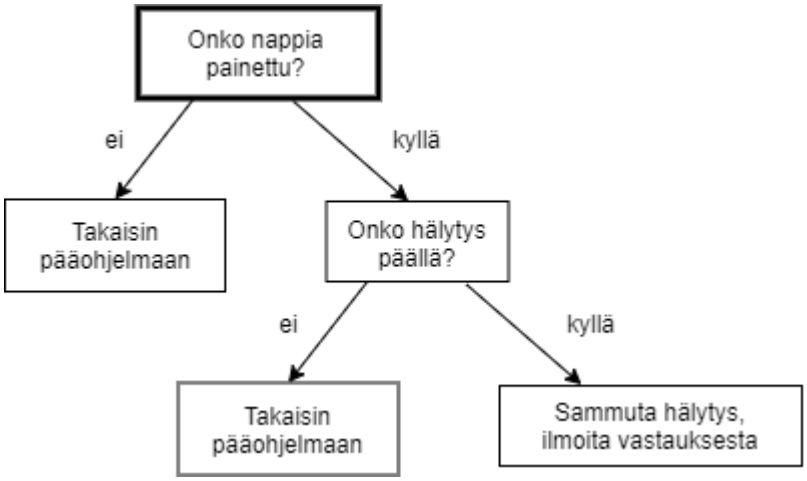
Mikäli datapuskurissa ei ole viestejä, jatketaan pääohjelmassa seuraavaan kohtaan. Muuten tarkistetaan, onko viesti ”ACKNOWLEDGE”-hyväksymisviesti aiemmin lähetetystä viestistä, vai onko se uusi julkaisu.

MQTT-välittäjälle lähetetyt viestit tallennetaan laitteen muistiin, jotta ne voidaan lähettää uudelleen, mikäli välittäjä ei vastaa niihin. Jos vastaanotettu viesti on tällaiseen viestiin vastattu hyväksymisviesti, voidaan pakettitunnisteen perusteella poistaa laitteen muistissa tallessa ollut viesti.

Mikäli viesti on uusi julkaisu, tarkistetaan viestin aihe, jonka perusteella hälytys, eli merkkivalo ja äänentuotto voidaan joko aloittaa tai lopettaa. Toiminnan jälkeen suoritus siirtyy takaisin pääohjelmaan.

5.5.2 Napin painalluksen tarkistus

Seuraavaksi ohjelman suoritus siirtyy napintarkastusfunktioon. Napin painallus tarkistetaan kuvan 43 osoittamalla tavalla.

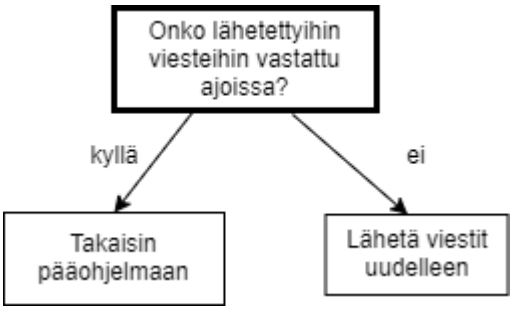


Kuva 43. Napin painalluksen tarkistus.

Mikäli nappia on painettu, tarkistetaan myös, onko hälytys jo päällä virhetilanteiden välttämiseksi. Laitteen on turha lähettää ilmoittautumislaitteelle vahvistusviestiä, jos ilmoittautumislaite ei odota sitä. Jos hälytys on päällä, se sammutetaan ja ilmoittautumislaitteelle lähetetään kuittauksesta viesti. Näin ilmoittautumislaite osaa kertoa vierailijalle, että häntä ollaan tulossa hakemaan, sekä muiden linkitettyjen ovikellojen hälytys voidaan sammuttaa.

5.5.3 Viestien tilanteen tarkistus

Viestien tilanteen tarkistus on esitetty kuvassa 44.

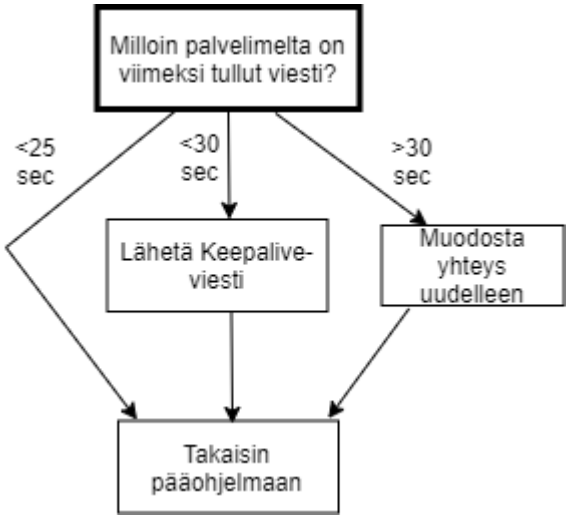


Kuva 44. Viestien tilanteen tarkistus.

Mikrokontrolleri tallentaa kaikki lähetetyt PUBLISH -ja SUBSCRIBE-viestit muistiin siltä varalta, että se ei vastaanota MQTT-välittäjältä ACK-viestejä niihin. Mikäli viesteihin ei ole vastattu tietyn ajan sisällä, laite yrittää lähettää ne uudelleen. MQTT-standardi ei määritä tarkkaa aikaa [62], jonka jälkeen paketit tulisi lähettää uudelleen. Koska ovikellolaite ei ole kovin aikakriittinen, voi uudelleenlähetysaika olla esimerkiksi yksi sekunti.

5.5.4 Yhteyden tarkistus

Jotta laite voi varmistua siitä, että sen yhteys MQTT-välittäjään on toiminnassa, tarkistetaan yhteys kuvan 45 osoittamalla tavalla.



Kuva 45. Yhteyden tarkistus.

Ensin ohjelma tarkistaa, milloin se on viimeksi saanut palvelimelta viestin. Mikäli viestistä on liian pitkä aika, on mahdollista, että yhteys on katkennut ilman että kumpikaan osapuoli on ilmoittanut siitä. Laitteelle riittää, jos se on saanut viestin palvelimelta alle 25 sekuntia sitten. Mikäli aikaa on kulunut enemmän, lähettää se varta vasten yhteyden ylläpitoon tarkoitetun, niin kutsutun Keepalive-viestin (Pidä hengissä) MQTT-välittäjälle.

Ohjelma lähettää Keepalive-viestejä niin kauan, kunnes palvelin vastaa viesteihin, tai aikaa on kulunut yli 30 sekuntia. Sen jälkeen laite toteaa, että yhteydessä on jotain vikaa, jolloin se yrittää muodostaa palvelimeen yhteyden uudelleen, ja aloittaa ohjelman suorittamisen alusta.

5.6 AVR-keskeytykset

Aiemmin esitetyssä napin painalluksen tarkistuksessa tarkistetaan, onko nappia painettu, mutta se ei ole niin suoraviivaista, että sen voisi vain tarkistaa yhdellä hetkellä pääohjelman suorituksessa. On mahdollista, että käyttäjä painaa nappulaa muuna aikana kuin juuri silloin, kun ohjelman suoritus on napintarkistusfunktiossa.

Ongelma voidaan ratkaista AVR-mikrokontrollereiden mahdollistamien keskeytysten käytöllä. Kun keskeytys laukaistaan, koodin suoritus siirtyy keskeytysohjelmaan tallentaen muistiin ohjelmakoodista kohdan, johon suoritus jäi. Kun keskeytysohjelma on suoritettu loppuun, siirtyy koodin suoritus takaisin normaaliin ohjelmakoodiin siihen kohtaan, mihin oltiin ennen keskeytystä jääty. [66]

Keskeytysohjelma voidaan määrittää jokaiselle erilaiselle keskeytykselle erikseen ja kukin keskeytys asetetaan erikseen päälle, jotta niistä aiheutuvia keskeytysohjelmia voidaan käyttää. ATmega328PB:ssä on useita erilaisia keskeytyksen lähteitä, joista tiettyjä kriteerejä voidaan muuttaa. Keskeytykset ovat ohjelmoitu tiettyyn tärkeysjärjestykseen, jota ei voi muuttaa. Useamman keskeytyksen tapahtuessa listassa tärkein keskeytysohjelma suoritetaan ensin. [35]

Keskeytysten lähteitä ovat esimerkiksi

- Pinnin muutos
- Sarjaliikenne
- Ajastin/laskuri.

Kun ohjelmoijan valitseman pinnin arvo muuttuu, aiheutuu PCINT-keskeytys (Pin change interrupt). Nappulan painallus voidaankin ohjelmoida tuottamaan tällainen keskeytys, ja keskeytysohjelmassa voidaan asettaa globaalin muuttujan ”nappi_painettu” arvoksi 1. Tarkistusfunktio voikin vain katsoa tämän muuttujan arvoa ja päätellä siitä, onko nappia painettu.

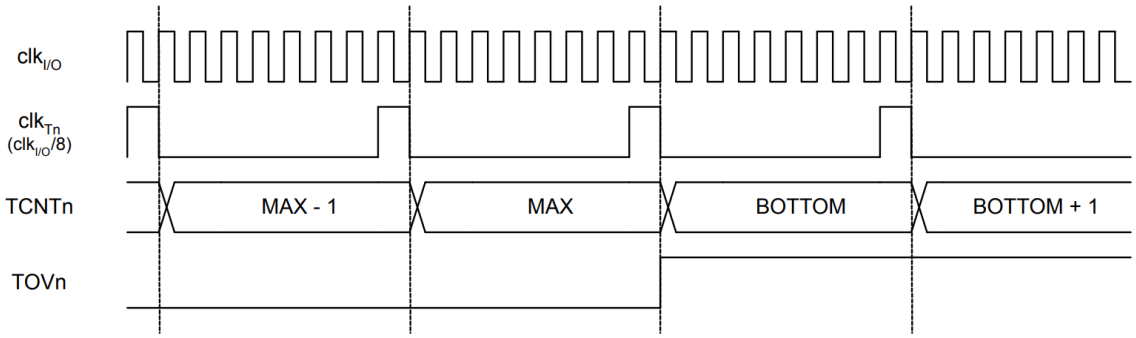
Esimerkiksi USART-sarjaliikenteessä voidaan tehdä keskeytys, kun mikrokontrollerin USART-sarjaliikennepuskurissa on dataa [35]. Tällä voidaan esimerkiksi välttää erillisen puskurin tarkastusohjelman käyttö. Ovikellolaitteessa keskeytystä ei voida kuitenkaan käyttää, sillä ovikellon käyttämässä SPI-sarjaliikenteessä ei tällaista keskeytystä ole.

5.6.1 Ajastin/laskurit

Keskeytyksiä voidaan myös aiheuttaa useissa eri tilanteissa ajastin/laskurien yhteydessä. ATmega328PB:ssä on kaksi 8-bittistä, sekä kolme 16-bittistä laskuria. [35]

Laskurien toimintaa voidaan muokata eri tavoin, kuten käyttämällä kellon esijakajaa, jolla laskurin toimintaa voidaan hidastaa suhteessa mikrokontrollerin kellotaajuuteen. Esijakajaksi voidaan asettaa 1, 8, 64, 512 tai 1024, millä järjestelmän kellojaksojen määrä jaetaan. Myös laskurin ala -tai yläarvoa voi muuttaa. [35]

Kuvassa 46 on havainnollistettu esijakajan käyttöä.



Kuva 46. Ajastin/laskurin ajoituskaavio esijakajaa käyttämällä [35].

Esijakajana clk_{Tn} on käytetty arvoa 8, joten joka kahdeksannella järjestelmän kellojaksolla laskurin TCNTn arvo kasvaa yhdellä. Kun laskuri kasvaa maksimiarvosta yhdellä, alkaa laskuri alkuarvosta BOTTOM, ja keskeytyslipun TOVn arvo muuttuu ykköseksi. [35]

Laskurille myös voidaan asettaa vertailuarvo, jonka vastatessa laskurin lukemaa voidaan aiheuttaa keskeytys. Kullekin ajastin/laskurilohkolle on oma pinni mikrokontrollerissa, minkä arvoa voidaan myös muuttaa vertailun toteutuessa.

Ajastimet voi asettaa toimimaan seuraavissa tiloissa:

- Tavallinen tila
- CTC-tila (Clear timer on Compare Match)
- Fast PWM -tila
- Phase Correct PWM.

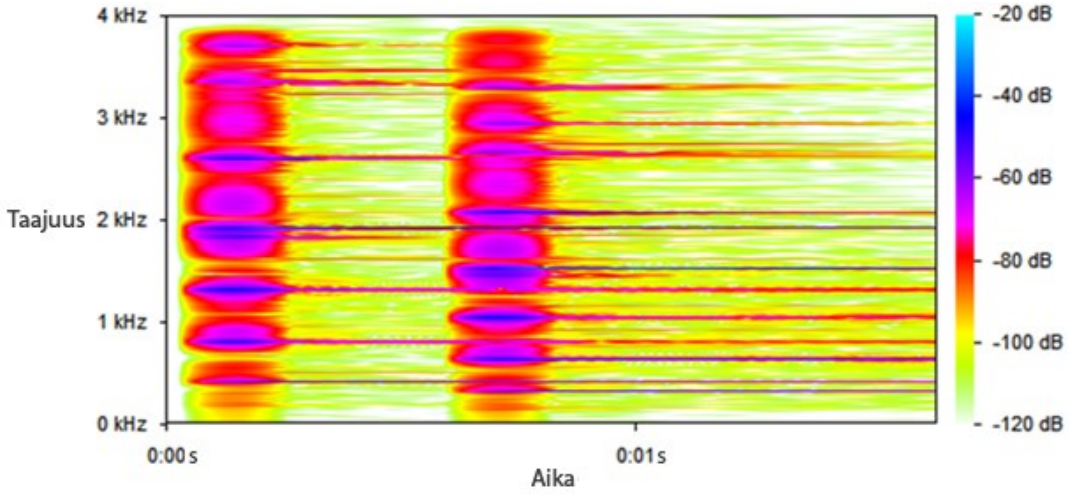
Tavallisessa tilassa laskuri toimii kuin yllä olevassa kuvassa 48. Phase Correct PWM:ssä laskuri laskee ensin ylöspäin, jonka jälkeen se laskeekin alaspäin. Koska sen toiminta on symmetristä, sitä suositellaan esimerkiksi moottorinhallintasovelluksissa [35].

CTC-tilassa laskuri nollataan, kun sen arvo täsmää vertailuarvoon. Tämä tila mahdollistaa ulostulotaajuuden helpon hallinnan, sillä yksinkertaisesti vertailuarvoa muuttamalla muuttuu myös ulostulotaajuus.

Fast PWM:ää voidaan käyttää korkeataajuuksisen PWM-signaalin muodostamiseen. Siinä laskurin arvoa vertaillaan vertailuarvoon, ja ulostulo asetetaan täsmäyksestä päälle, ja kun laskuri ohittaa maksimiarvon, asetetaan ulostulo nollaan.

5.6.2 Äänen tuotto

Käytettävän äänitiedoston spektrogrammi, joka sisältää äänen taajuudet sekä niiden äänenvoimakkuudet eri ajan hetkinä, on esitetty kuvassa 47.



Kuva 47. Käytettävän äänitiedoston spektrogrammi.

Äänitiedosto muunnetaan PCM-formaattiin (Pulse code modulation). PCM-formaatissa ääni on numerosarjana, jossa arvot kertovat signaalin amplitudin kunkin näytteen hetkellä [67].

PCM-muotoon tallennettava ääni on rajoitettu 8-bittiseen muotoon ja äänen näytteenottotaajuus on rajoitettu 8 kHz:iin, jotta äänitiedosto veisi mahdollisimman vähän tilaa. Suurempi näytteenottotaajuus tarvitsee myös enemmän suorituskykyä, joka on mikrokontrollerissa suhteellisen rajallinen. 8-bittisen äänen amplitudi saa siis arvoja väliltä 0–255, ja se päivittää amplitudin arvon 8000 kertaa sekunnissa.

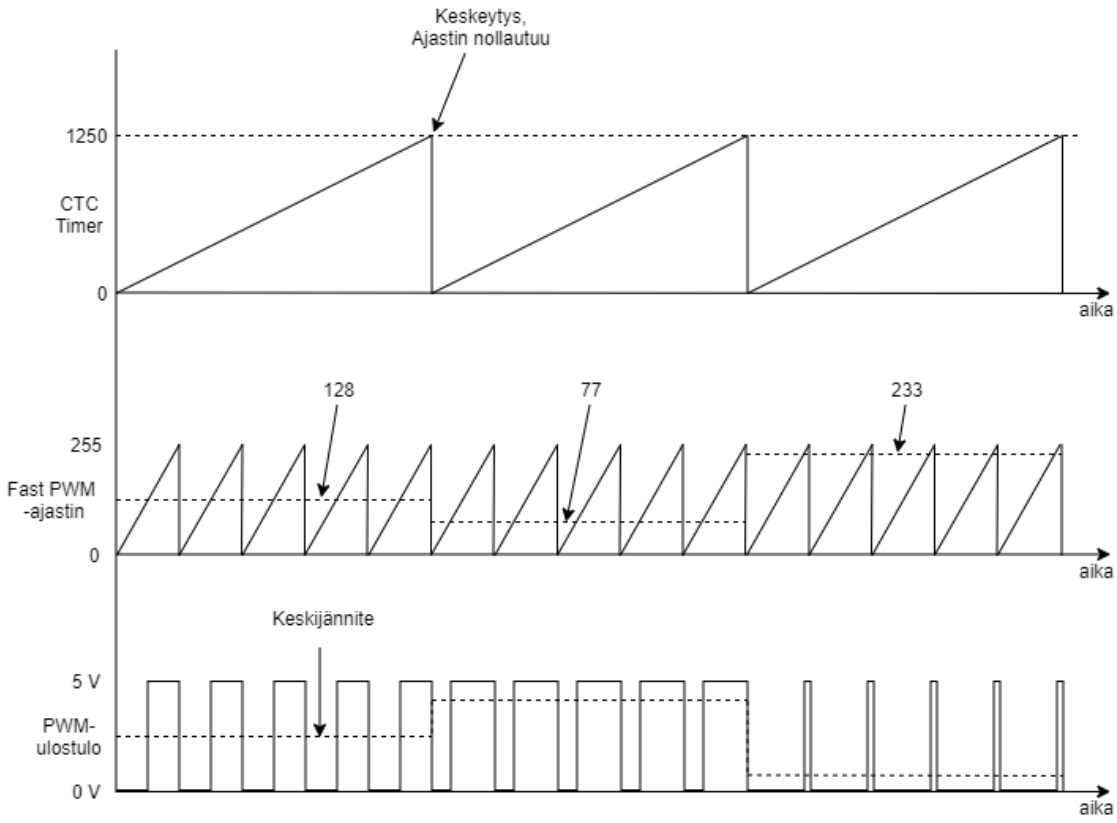
Koska mikrokontrolleri pystyy antamaan ulostuloon vain digitaaliset arvot 1 ja 0, tarvitaan äänen tuottamiseen kaksi ajastin/laskuria, joista toinen tuottaa 8 kHz näytteenottotaajuuden ja toinen sopivan jännitteen PWM-signaalilla. PWM-signaalin tuottamiseen käytetään 8-bittistä Fast PWM:ää ilman esijakajaa, jolloin sen taajuus on

$$f_{PWM} = \frac{F_{CPU}}{256} = \frac{10\ \text{MHz}}{256} \approx 39\ \text{kHz}.$$

Näytteenottotaajuuden tuottamisessa ajastin/laskuria käytetään aiemmin esitellyssä CTC-tilassa. Vertailuarvo asetetaan siten, että laskuri aiheuttaa keskeytyksiä samalla taajuudella, kuin äänisignaali on näytteistetty. Äänitiedoston näytteenottotaajuus on 8 kHz ja mikrokontrollerin kellotaajuus on 10 MHz. Koska laskurin arvo kasvaa yhdellä joka kellojakso, on vertailuarvon oltava

$$OCR0A = \frac{F_{CPU}}{F_{Sample}} = \frac{10\ \text{MHz}}{8\ \text{kHz}} = 1250.$$

Vertailuarvon täsmäämisestä aiheutetaan keskeytysohjelma, jossa Fast PWM:ää käyttävän laskurin vertailuarvoksi asetetaan PCM-muotoon tallennetun äänisignaalin näytteen amplitudin arvo. Kuvassa 48 on esitetty esimerkki ajastin/laskurien toiminnasta, kun ääninäytteiden arvot ovat 128, 77, 233.



Kuva 48. Esimerkki ajastin/laskurien toiminnasta.

Kuvan ylin signaali kuvastaa 16-bittisen CTC-tilassa olevan laskurin arvoa. Kun laskuri saavuttaa arvon 1250, laskurin arvo nollataan ja suoritetaan keskeytysfunktio. Keskeytysfunktiossa asetetaan Fast PWM -laskurille uusi arvo. Keskimmäisessä signaalissa on kuvattu 8-bittisen Fast PWM-tilassa olevan laskurin arvo, sekä merkattu laskurin vertailuarvo. Alin signaali kuvastaa mikrokontrollerin ulostulopinnin jännitettä.

Kun jännite suodatetaan kondensaattorilla, saadaan lopputulokseksi alkuperäistä ääntä vastaava signaali.

5.7 Mikrokontrollerin ohjelmointi

Kun ohjelmakoodi mikrokontrollerille on luotu, siirretään se tietokoneelta mikrokontrollerille. AVR-mikrokontrollereiden, jollainen myös valittu ATmega328PB on, on eri mahdollisuuksia siirtää ohjelma mikrokontrollerille.

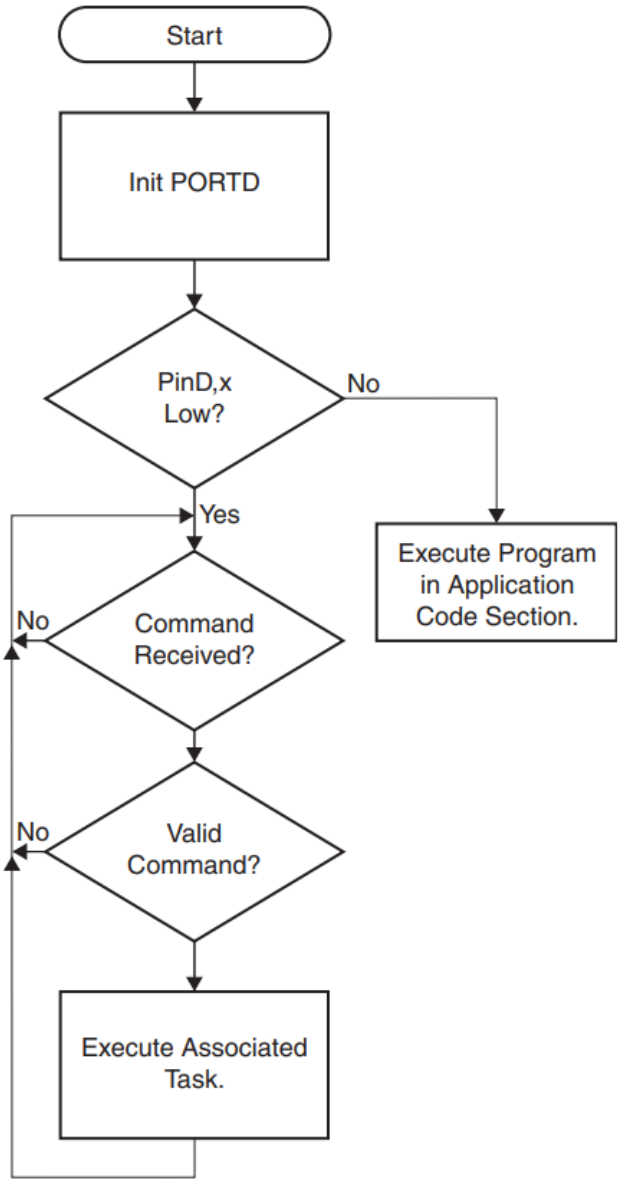
Tavallisesti mikrokontrollerit ohjelmoidaan ISP:llä (In-System Programming) tai niin kutsutun alkulatausohjelman (bootloader) avulla niiden helppoudesta johtuen. Turvallisimmat vaihtoehdot ovat kuitenkin HVPP (High voltage parallel programming, korkeajännitteinen rinnakkaisohjelmointi)- ja HVSP (High voltage serial programming, korkeajännitteinen sarjaohjelmointi) -ohjelmointi [68].

Mikrokontrollereissa on niin kutsuttuja sulakebittejä (fuse bit), joilla säädetään esimerkiksi käytettävän kellosignaalin lähdettä tai estetään SPI-väylän käyttö [69]. Käyttäjä voi esimerkiksi epähuomiossa ohjelmoida nämä bitit väärin, jolloin mikrokontrolleri voi mennä toimintakyvyttömäksi, eikä mikrokontrolleria voi enää ohjelmoida normaalia käyttöjännitettä käyttävillä ohjelmointimenetelmillä. Korkeajännitteinen ohjelmointi on aina mahdollista sulakebittien tilasta riippumatta.

Korkeajännitteisessä ohjelmoinnissa reset-pinniin tuodaan 12 voltin jännite, ja lisäksi sarjaohjelmoinnissa 7 muuta pinniä ja rinnakkasohjelmoinnissa 15, joten käytännössä nämä ohjelmointitavat eivät ole mahdollisia ilman mikrokontrollerin irrottamista piirilevyltä [68].

ISP-ohjelmoinnissa voidaan käyttää SPI-väylää, joka esiteltiin aiemmin luvussa 3. Sitä voidaan käyttää myös, kun mikropiiri on juotettu piirilevylle [68], kunhan muut SPI-väylää käyttävät piirit eivät häiritse SPI-väylän liikennettä.

Mikrokontrollerille on myös mahdollista ohjelmoida alkulatausohjelma, joka poltetaan muistiin siten, että se ajetaan ennen varsinaista ohjelmakoodia. Alkulatausohjelma käynnistyy aina ennen varsinaista ohjelmakoodia. [70]



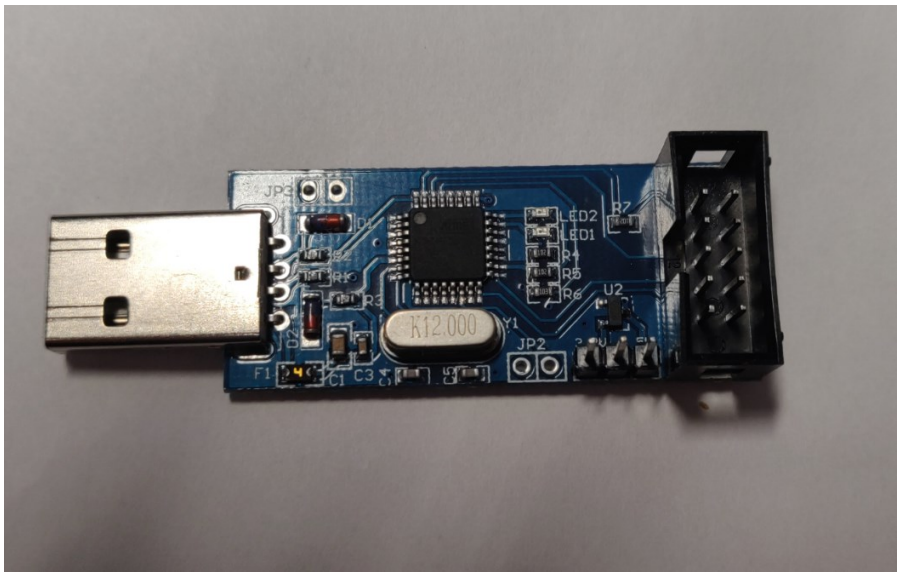
Kuva 49. Esimerkki alkulatausohjelman toiminnasta [70].

Esimerkiksi mikrokontrollerivalmistaja Atmelin esimerkin [70] mukaan, joka on näytetty kuvassa 49, ohjelma alustaa PORTD-pinniryhmän ja tarkistaa tietyn pinnin tilan. Mikäli pinnissä on jännite, ohjelmamuistissa alkulatausohjelman jälkeen oleva normaali ohjelma ajetaan. Jos pinni on jännitteetön, mikrokontrolleri alkaa kuuntelemaan sille tulevia komentoja ja toteuttaa niitä. Kun ohjelmointi on valmis, PORTD:n pinni vapautetaan ja mikrokontrolleri käynnistetään uudelleen ja normaalin ohjelman suoritus alkaa.

Alkulatausohjelman käyttö mahdollistaa esimerkiksi mikrokontrollerin ohjelmoinnin USB-väylän kautta. Tällöin USB-viestintä tulee vielä muuntaa mikrokontrollerin ymmärtämäksi USART-yhteydeksi (Universal serial asynchronous receiver/transmitter) esimerkiksi FT232R-mikropiirin avulla. Ennen alkulatausohjelman käyttämistä ohjelmoimiseen,

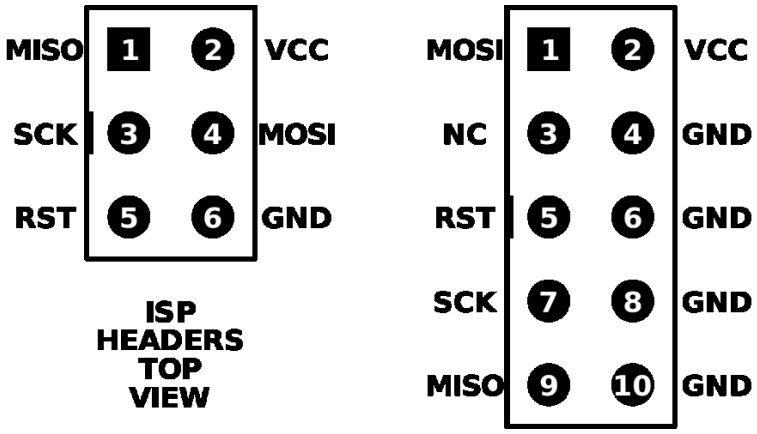
täytyy sellainen ohjelmoida laitteeseen ensimmäisen kerran jollain vaihtoehtoisella tavalla.

Koska ovikelloon ei tule USB-liitäntää ja laitteen tulee kuitenkin tukea jotain muuta ohjelmointitapaa, ohjelmoidaan mikrokontrolleri ISP:tä hyödyntämällä. Ohjelmointiin käytetään kuvassa 50 näkyvää USBASP-ohjelmoijaa.



Kuva 50. USBASP-ohjelmoija.

USBASP:ssa on ATmega8-mikrokontrolleri, sekä muutama passiivinen komponentti, joiden avulla sitä voidaan käyttää AVR-mikrokontrollerin ISP-ohjelmoijana. USBASP:n ohjelmointiliitin on 10-pinninen, mutta mikrokontrollerin liitin on 6-pinninen. Nämä liittimet ovat esitelty kuvassa 51.



Kuva 51. 6- ja 10-pinniset ISP-liittimet [71].

Liittimet ovat kuitenkin yhteensopivia adapterin avulla, sillä molemmissa liittimissä kulkee yhtä paljon informaatiota, 10-pinnisessä liittimessä on vain enemmän maajohtimia sekä yksi pinni on käyttämätön.

6. LAITTEEN TESTAUS

Tässä luvussa kerrotaan ohjelmoinnin aikana käytetyistä apuvälineistä -ja keinoista, sekä tarkastellaan laitteen toimivuutta. Lopuksi pohditaan myös testauksessa havaittuja laitteen mahdollisia puutteita.

6.1 Testausympäristö

Ohjelmakoodia kirjoittaessa voi tehdä virheitä, jotka estävät laitetta toimimasta niin kuin on tarkoitettu. Ohjelmointivaiheessa luodaankin pieni testausympäristö virhetilanteiden selvittämiseksi. Laitteessa ei ole USB-porttia, joten laitteen yhdistäminen suoraan tietokoneeseen ei ole mahdollista.

ATmega328PB tukee kuitenkin USART-sarjaliikennettä, joten laitteen voi kytkeä sarjaliikenneväylällä toiseen laitteeseen, joka kykenee kommunikoimaan tietokoneen USB-liitännän kautta. Tässä käytetään ATmega2560-mikrokontrolleriin perustuvaa alustaa. Alustassa on mikrokontrollerin lisäksi CH340-mikropiiri, joka muuntaa USB-liikenteen tietokoneelta mikrokontrollerille sopivaksi USART-sarjaliikenteeksi.

Avustavan mikrokontrollerialustan mikrokontrollerin normaali toiminta estetään kytkemällä sen reset-signaali aktiiviseksi, jolloin sen sarjaliikennepinnit ovat suoraan yhteydessä CH340-mikropiiriin ja sitä kautta tietokoneeseen. Ovikellolaitteessa olevan mikrokontrollerin sarjaliikennepinneihin juotetaan väliaikaisesti johtimet, jotka kytketään avustavan mikrokontrollerialustan sarjaliikennepinneihin. Kuvassa 52 on näytetty käytetty kehitysympäristö.

Kuva 52. Käytetty kehitysympäristö.

Ovikellolaitteen voi sitten ohjelmoida lähettämään USART-sarjaliikennettä pitkin viestejä, jotka esimerkiksi kertovat, että tiettyyn pisteeseen koodissa on päästy tai mikä jonkin muuttujan arvo on.

Ohjelmiston kehitysvaiheessa käytettiin tietokoneelle paikallisesti asennettua MQTT-välittäjää, ja ovikellolaite kytkettiin suoraan tietokoneeseen Ethernet-johdolla. Näin ovikellolaitteen ohjelmisto voidaan saada toimivaksi lähiverkossa ennen laitteen testaamista oikeassa ympäristössä.

6.2 Toiminnan testaaminen oikeassa ympäristössä

Kun ohjelmisto on saatu toimimaan testausympäristössä, voidaan ovikellolaite yhdistää internetin yli oikeaan MQTT-välittäjään. Tietokoneella suoritetaan erillistä ilmoittautumisjärjestelmän palvelinta, joka myös yhdistetään MQTT-välittäjään. Kaikki mahdolliset tilanteet testataan, ja pienten virheenkorjailujen jälkeen voitiin todeta, että laite toimii oikein normaaleissa olosuhteissa.

6.2.1 Virhetilanteet

Koska laitteeseen ei ole implementoitu kaikkia MQTT-spesifikaation määrittämiä ominaisuuksia, on joitain tilanteita, joissa laite alkaa käyttäytymään ei-toivotusti. Testeissä havaittiin, että esimerkiksi muun kuin QoS-arvon 1 käyttäminen tuottaa mikrokontrollerissa virhetilanteen.

Osa MQTT-paketeista voi myös sisältää erillisen viestin. Koska ovikellolaite ei käytä esimerkiksi PUBLISH-paketeissa erillistä viestiä, vaan toimii pelkästään paketin sisältämän aiheen perusteella, se tulkitsee viestillisen paketin virheellisenä, mikä myös aiheuttaa virhetilanteen.

Ennen kuin laitteen voi ottaa käyttöön, pitää esimerkiksi nämä tilanteet huomioida, mutta laitteelle tulee suorittaa myös perusteellisempia testejä, jotta ovikellolaite saadaan toimimaan mahdollisimman luotettavasti. Tällä hetkellä ovikellolaite käynnistyy virhetilanteista automaattisesti uudelleen, millä vältetään tarve käynnistää laite manuaalisesti uudelleen.

6.3 Laitteen puutteet

Laitteessa ei ole havaittu kriittisiä puutteita, mutta jotkin ominaisuudet voisivat olla parempia, etenkin tulevaisuudessa, mikäli laitetta kehitetään eteenpäin. Tällä hetkellä esimerkiksi ääni voi olla enimmillään vain muutaman sekunnin pituinen. Mikäli halutaan soittaa pidempiä ääniä, tarvitsisi laitteeseen lisätä ohjelmamuistin määrää. Tämän voi esimerkiksi toteuttaa muistikortilla ja muistikortinlukijalla. Tällöin laitteen muistiin voisi myös ohjelmoida useamman äänen.

Ovikellolaitteen tuottaman äänen laatukaan ei myöskään ole paras mahdollinen, joten sitä voisi parantaa paremman käyttökokemuksen saamiseksi. Äänenlaatua voi mahdollisesti parantaa ohjelmallisesti, mutta rajoitteena voi myös olla mikrokontrollerin suorituskyky, jolloin äänen parantamiseksi voisi joutua tekemään suuriakin muutoksia.

YHTEENVETO

Diplomityössä esiteltiin ovikellolaitteen suunnittelu ja toteutus, sekä laitteen ohjelmakoodin rakennetta. Työssä tutustuttiin myös joihinkin mikrokontrollerin ominaisuuksiin sekä erilaisiin tekniikoihin ja protokolliin. Laitteen piirikaavio sekä pohjapiirros suunniteltiin PADS Logic -ja Layoutilla. Ohjelmointi tehtiin C/C++:lla. Laitteessa käytettiin valmiina ohjelmakirjastoina Ethernet -ja TCP/IP-kirjastoja, mutta MQTT:n toiminnallisuus toteutettiin itse.

Valmis ovikellolaite täytti sille asetetut tavoitteet. Ovikellolaitteen suunnittelu onnistui, vaikka PoE-kontrolleri jouduttiinkin vaihtamaan ja kytkentä siltä osin suunnittelemaan uudelleen. Ovikellon fyysiset ominaisuudet riittivät mahdollistamaan ohjelmiston toiminta, sekä ohjelmalla laite saatiin toimimaan halutulla tavalla. Taulukossa 5 on esitelty laitteen keskeisimmät ominaisuudet.

Taulukko 5. Ovikellolaitteen keskeisimmät ominaisuudet.

Aihe	Ovikellolaite
Mikrokontrolleri	ATmega328PB, 10 MHz, 32 kt ROM, 2 kt RAM
Liitännät	RJ45-liitin, 2 lediä, nappi, kaiutin, potentiometri
Ohjelman koko	30 kt ROM, 1 kt RAM
Ethernet-kontrolleri	W5500, 25 MHz, Integroitu TCP/IP, 32 kt muistia.
Kommunikaatiomenetelmät	MQTT, SPI
Virransyöttö	Power over Ethernet
Ääni	2 sekuntia, 8-bittinen, 8 kHz näytteenottotaajuus.
Kaiuttimen taajuusalue	450 – 5000 Hz
Kaiuttimen teho	2 W

Laitteessa on eniten kehitettävää äänentoiston osalta. Tällä hetkellä laite kykenee vain muutaman sekunnin mittaiseen ääneen ja myös äänenlaadussa olisi parannettavaa. Se kuitenkin riittää toistaiseksi. Lopulliseen tuotteeseen tulee myös vielä valmistaa kotelo.

Laitteen teko on ollut hyvin opettavaista. Laitetta suunnitellessa on perehdytty erilaisiin sähkökytkentöihin varsinkin PoE-kytkennässä, sekä ohjelmointivaiheessa mikrokontrollerin ominaisuuksia ja MQTT-protokollaa on tutkittu perusteellisesti.

Suurimmat ongelmat ilmenivät fyysisen laitteen toteutuksessa, sillä kytkentöjen suunnittelussa oli muutama virhe, sekä ensimmäisen prototyypin pinnoittamattomalle kuparilevylle komponenttien juottamisessa aiheutui muutama epätoivottu oikosulku. Vaikeasti käsin juotettavan PoE-kontrollerin juottaminen ei myöskään onnistunut, joten toiseen prototyyppiin oli vaihdettava PoE-kontrolleri. Toiseen prototyyppiin myös käytettiin pinnoitettua piirilevyä, jolloin komponenttien juottaminen onnistui ilman virheitä.

LÄHTEET

[1] A. Etahi, A. Gschwender, Introduction to the ZigBee Wireless Sensor and Control Network, 2009. Saatavissa (viitattu 9.10.2019): http://www.informit.com/articles/article.aspx?p=1409785&seqNum=2

[2] A. Stachowicz, ZigBee Wireless Networks, 2010. Saatavissa (viitattu 9.10.2019): http://zigbee.pbworks.com/w/page/25465049/ZigBee

[3] M. Rouse, Advanced Encryption Standard (AES), 2017. Saatavissa (viitattu 9.10.2019): https://searchsecurity.techtarget.com/definition/Advanced-Encryption-Standard

[4] Zigbee 3.0, *Zigbee Alliance*, 2019. Saatavissa https://zigbee.org/zigbee-for-developers/zigbee-3-0/

[5] A. Dementyev, S. Hodges, S. Taylor, J. Smith, Power consumption analysis of Bluetooth Low Energy, ZigBee and ANT sensor nodes in a cyclic sleep scenario, teoksessa *2013 IEEE International Wireless Symposium, IWS 2013*, 2013.

[6] Energizer CR2032, *Energizer*. Saatavissa https://data.energizer.com/pdfs/cr2032.pdf

[7] M. Turner, Wi-Fi 6 Explained: The Next Generation of Wi-Fi, 2019. Saatavissa (viitattu 9.10.2019): https://www.techspot.com/article/1769-wi-fi-6-explained/

[8] P. Panda, ESP8266 TCP server speed test, 2017. Saatavissa (viitattu 9.10.2019): http://iot-bits.com/esp8266-tcp-server-speed-test/

[9] N. Tengyuen, 5 Wireless Wifi 802.11 a, b, g, n, ac, ad, ah, aj, ax, ay Router Range and Distance Comparison, 2019. Saatavissa (viitattu 9.10.2019): https://www.geckoandfly.com/10041/wireless-wifi-802-11-abgn-router-range-and-distance-comparison/

[10] ESP8266EX Datasheet, *Espressif Systems*, 2015. Saatavissa (viitattu 9.10.2019): https://cdn-shop.adafruit.com/product-files/2471/0A-ESP8266__Datasheet__EN_v4.3.pdf

[11] What Is Ethernet?, *Linksys*. Saatavissa (viitattu 9.10.2019): https://www.linksys.com/us/r/resource-center/basics/whats-ethernet/

[12] G. Torres, How Gigabit Ethernet Works, 2005. Saatavissa (viitattu 9.10.2019): https://www.hardwaresecrets.com/how-gigabit-ethernet-works/

[13] Type 3 and Type 4 are Here!, *Fluke Networks*, 2018. Saatavissa (viitattu 9.10.2019): https://www.flukenetworks.com/blog/cabling-chronicles/type-3-and-type-4-are-here

[14] What is PoE?, *Shure*, 2014. Saatavissa (viitattu 9.10.2019): https://www.shure.com/en-US/support/find-an-answer/what-is-poe

[15] Si3404 Data Sheet, *Silicon Labs*, 2018. Saatavissa (viitattu 9.10.2019): https://www.silabs.com/documents/public/data-sheets/si3404-datasheet.pdf

[16] AN1130: Si3404/06x PoE-PD Controller Design Guide, *Silicon Labs*. Saatavissa (viitattu 9.10.2019): https://www.silabs.com/documents/public/application-notes/an1130-si3404-06x-dg.pdf

[17] Power over Ethernet (POE) Explained, *Veracity UK*, 2016. Saatavissa (viitattu 25.11.2019): https://www.veracityglobal.com/resources/articles-and-white-papers/poe-explained-part-2.aspx

[18] X. Lleixa, Active or Passive PoE, That is the Question, 2017. Saatavissa (viitattu 14.12.2019): https://blog.netgear.com/blog/active-or-passive-poe-that-is-the-question/

[19] PoE Injector, *Intellinet Network Solutions*. Saatavissa (viitattu 9.10.2019): https://intellinetnetwork.eu/glossary/what-is-a-poe-injector/

[20] W5500 Datasheet, *WIZnet*. Saatavissa (viitattu 9.10.2019): https://www.mouser.fi/datasheet/2/443/Wiznet_W5500_DS_1 0 5-1212109.pdf

[21] IEEE 802.3bt-2018 - IEEE Standard for Ethernet Amendment 2: Physical Layer and Management Parameters for Power over Ethernet over 4 pairs, 2018. Saatavissa (viitattu 9.10.2019): https://standards.ieee.org/standard/802_3bt-2018.html

[22] S. Chyo, V. Singh, Flyback transformer tutorial: function and design, 2006. Saatavissa (viitattu 9.10.2019): https://www.eetimes.com/document.asp?doc_id=1273089

[23] UNDERSTANDING VOLTAGE-REFERENCE TOPOLOGIES AND SPECIFICATIONS, *Maxim Integrated Products Inc*, 2013. Saatavissa (viitattu 9.10.2019): https://www.maximintegrated.com/en/design/technical-documents/tutorials/7/719.html

[24] Optocoupler tutorial, *Electronics Tutorials*. Saatavissa (viitattu 9.10.2019): https://www.electronics-tutorials.ws/blog/optocoupler.html

[25] TLV431x Low-Voltage Adjustable Precision Shunt Regulator, *Texas Instruments*, 2018. Saatavissa (viitattu 9.10.2019): http://www.ti.com/lit/ds/symlink/tlv431a.pdf

[26] K. Blåsol, How to Connect the ENC28J60 to an Arduino, 2016. Saatavissa (viitattu 9.10.2019): https://create.arduino.cc/projecthub/Sourcery/how-to-connect-the-enc28j60-to-an-arduino-efd0dd

[27] Arduino Ethernet Shield V2, *Arduino*. Saatavissa (viitattu 9.10.2019): https://www.arduino.cc/en/Main/ArduinoEthernetShieldV2

[28] ENC28J60-I/SO. Saatavissa (viitattu 21.11.2019): https://in.rsdelivers.com/product/microchip/enc28j60-i-so/microchip-enc28j60-i-

so-ethernet-controller-10/1784863

[29] WIZnet, W5500. Saatavissa (viitattu 21.11.2019): https://www.wiznet.io/product-item/w5500/

[30] ENC28J60 Data Sheet, *Microchip Technology Inc.*, 2008. Saatavissa (viitattu 9.10.2019): http://ww1.microchip.com/downloads/en/devicedoc/39662c.pdf

[31] M. Grusin, Serial Peripheral Interface (SPI). Saatavissa (viitattu 9.10.2019): https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all

[32] Atmega328PB, *Microchip Technology Inc.* Saatavissa (viitattu 9.10.2019): https://www.microchip.com/wwwproducts/en/ATmega328PB

[33] EFM8BB31F32I-B-QSOP24, *Farnell*. Saatavissa (viitattu 9.10.2019): https://ie.farnell.com/silicon-labs/efm8bb31f32i-b-qsop24/mcu-8bit-8051-50mhz-qsop-24/dp/2614239

[34] ESP8266EX, *Edwin Robotics*. Saatavissa (viitattu 9.10.2019): https://shop.edwinrobotics.com/ic-s/518-esp8266ex-tiny-wireless-80211-bgn-chip.html

[35] ATmega328PB Datasheet, *Microchip*, 2017. Saatavissa (viitattu 9.10.2019): https://www.mouser.fi/datasheet/2/268/40001906A-1222909.pdf

[36] EFM8BB3 Data Sheet, *Silicon Labs*. Saatavissa (viitattu 9.10.2019): https://www.mouser.fi/datasheet/2/368/EFM8BB3-DataSheet-938710.pdf

[37] Arduino UNO Rev3, *Arduino*. Saatavissa (viitattu 9.10.2019): https://store.arduino.cc/arduino-uno-rev3

[38] Product spotlight: Piezo and magnetic buzzers, *CUI Devices*. Saatavissa (viitattu 9.10.2019): https://www.cuidevices.com/product-spotlight/piezo-and-magnetic-buzzers

[39] Piezo buzzer definition, *Alan Butcher Components LTD*, 2019. Saatavissa (viitattu 9.10.2019): https://www.abcomponents.co.uk/piezo-buzzer-definition/

[40] CPT-3055-90PM Datasheet, *CUI Inc*, 2019. Saatavissa (viitattu 9.10.2019): https://www.mouser.fi/datasheet/2/670/cpt-3055-90pm-1627918.pdf

[41] How do speakers work?, *Physics.org*. Saatavissa (viitattu 9.10.2019): http://www.physics.org/article-questions.asp?id=54

[42] Choosing a Buzzer/Speaker/Transducer, *Source Research Inc*. Saatavissa (viitattu 9.10.2019): http://www.sourceresearch.com/newsletter/ChoosingSpeakers.cfm?emART

[43] CDMG13008L-02 Datasheet, *CUI Inc*, 2006. Saatavissa (viitattu 9.10.2019): https://www.mouser.fi/datasheet/2/670/cdmg13008l-02-1310164.pdf

[44] SM450208-1 Datasheet, *DB Unlimited*, 2013. Saatavissa (viitattu 10.10.2019): https://www.mouser.fi/datasheet/2/683/SM450208-1-469427.pdf

[45] 4-bit-linear-PCM, 2013. Saatavissa (viitattu 19.11.2019): https://commons.wikimedia.org/wiki/File:4-bit-linear-PCM.svg

[46] PAM8303C Datasheet, *Diodes Incorporated*, 2018. Saatavissa (viitattu 10.10.2019): https://www.diodes.com/assets/Datasheets/PAM8303C.pdf

[47] E. Coates, Buck Converters. Saatavissa (viitattu 14.12.2019): http://www.learnabout-electronics.org/PSU/psu31.php

[48] AP3417C Datasheet, *Diodes Incorporated*, 2019. Saatavissa (viitattu 10.10.2019): https://www.diodes.com/assets/Datasheets/products_inactive_data/AP3417C.pdf

[49] C. Cheung, M. Kjar, Using the 16 MHz Crystal Oscillator, 2005. Saatavissa (viitattu 30.11.2019): https://www.nxp.com/docs/en/application-note/AN2500.pdf

[50] PCB Layout for the Ethernet PHY Interface, *Digi International*. Saatavissa http://ftp1.digi.com/support/documentation/022-0137_F.pdf

[51] D. Cook, PCB Solder Mask and Silkscreen. Saatavissa (viitattu 10.10.2019): http://www.robotroom.com/Silkscreen-and-Solder-Mask-1.html

[52] A. Wright, PRINTED CIRCUIT BOARD SURFACE FINISHES: ADVANTAGES AND DISADVANTAGES. Saatavissa (viitattu 10.10.2019): https://www.epectec.com/articles/pcb-surface-finish-advantages-and-disadvantages.html

[53] hasl-vs-enig, *db Electronics*, 2017. Saatavissa (viitattu 17.12.2019): https://db-electronics.ca/2017/05/31/surface-finish-on-pcbs-and-why-it-matters-for-reproduction-carts/hasl-vs-enig/

[54] Hot Air Solder Levelling, *Silga*. Saatavissa (viitattu 10.10.2019): http://www.silga.com/en/pcb-division/products/surface-finishings/products/hot-air-solder-levelling]

[55] M. Kumar, Advantages of C over C++, 2016. Saatavissa (viitattu 10.12.2019): https://thebittheories.com/advantages-of-c-over-c-f31f61873997

[56] The TCP/IP Protocol Stack, *TechnologyUK*. Saatavissa (viitattu 28.10.2019): http://www.technologyuk.net/telecommunications/internet/tcp-ip-stack.shtml

[57] M. Rouse, IPv6 (Internet Protocol Version 6), 2019. Saatavissa (viitattu 28.10.2019): https://searchnetworking.techtarget.com/definition/IPv6-Internet-Protocol-Version-6

[58] P. Sclafani, IPv6 and the transition from IPv4 explained, 2019. Saatavissa (viitattu 28.10.2019): https://www.6connect.com/resources/ipv6-and-the-transition-from-ipv4-explained/

[59] UDP – USER DATAGRAM PROTOCOL, *IPv6.com*, 2019. Saatavissa (viitattu

28.10.2019): https://www.ipv6.com/general/udp-user-datagram-protocol/

[60] P. Tracy, MQTT protocol minimizes network bandwidth for the internet of things, 2016. Saatavissa (viitattu 29.10.2019): https://www.rcrwireless.com/20161129/fundamentals/mqtt-internet-of-things-tag31-tag99

[61] M. Yuan, Getting to know MQTT, 2017. Saatavissa (viitattu 30.10.2019): https://developer.ibm.com/articles/iot-mqtt-why-good-for-iot/

[62] A. Banks, MQTT Version 5.0, 2019. Saatavissa (viitattu 30.10.2019): https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html#_Toc3901103

[63] Ethernet library, *Arduino*. Saatavissa (viitattu 6.11.2019): https://www.arduino.cc/en/Reference/Ethernet

[64] Arduino library for MQTT support. Adafruit, 2019.

[65] A. Moore, Adafruit MQTT Library Ethernet Example. 2016.

[66] M. Silva, Introduction to Microcontrollers - Interrupts, 2013. Saatavissa (viitattu 18.11.2019): https://www.embeddedrelated.com/showarticle/469.php

[67] S. Brunhorn, PCM - Pulse Code Modulation. Saatavissa (viitattu 19.11.2019): http://einstein.informatik.uni-oldenburg.de/rechnernetze/pcm.htm

[68] AVR® Programming Interfaces, *Microchip Technology Inc.* Saatavissa (viitattu 21.10.2019): https://microchipdeveloper.com/8avr:programminginterfaces

[69] AVR® Fuses, *Microchip Technology Inc.* Saatavissa (viitattu 21.10.2019): https://microchipdeveloper.com/8avr:avrfuses

[70] AVR109: Self Programming, *Atmel Corporation*, 2004. Saatavissa (viitattu 21.10.2019): http://ww1.microchip.com/downloads/en/Appnotes/doc1644.pdf

[71] ISP HEADERS TOP VIEW. Saatavissa (viitattu 15.1.2019): https://www.pngfuel.com/free-png/lvszy