Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Jarkko Aho

LoRaWAN osana IoT-talotekniikkaa

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintätekniikan tutkinto-ohjelma

Insinöörityö

01.11.2019

Tekijä Otsikko Sivumäärä Aika	Jarkko Aho LoRaWAN osana IoT-talotekniikkaa 44 sivua + 1 liite 01.11.2019
Tutkinto	Insinööri (AMK)
Tutkinto-ohjelma	Tieto- ja viestintätekniikka
Ammatillinen pääaine	Smart Systems
Ohjaajat	Lehtori Kimmo Sauren Lehtori Keijo Länsikunnas

IoT-laitteet ja vähävirtaiset langattoman tiedonsiirtotekniikat ovat kehittyneet kovaa vauhtia viimeisen 10 vuoden aikana, mutta mikä tekniikka on paras? Insinöörityön tavoitteena oli tutkia LoRaWAN-teknologian toimintaa ja sitä, miten sitä voidaan soveltaa IoT-laitteisiin osana IoT-talotekniikkaa. Insinöörityön vaatimuksena oli tutkia ja ymmärtää LoRa-teknologia, siihen liittyvät modulaatiot ja protokollat sekä luoda prototyyppi, joka lähettäisi antureista kerättyä tietoa pilveen.

Insinöörityön taustatutkimus keskittyi syvällisesti LoRa- ja LoRaWAN-tekniikkaan, niiden arkkitehtuuriin, vähävirtaisuuteen, kapasiteettiin ja turvallisuuteen. Taustatutkimuksen yhteydessä tehtiin laajakirjoinen vertailu tunnetuimpien langattomien LPWAN-tekniikoiden välillä sekä verrattiin niiden käyttötarkoituksia ja -kohteita LoRaWAN-tekniikan kanssa.

Prototyypin toteutus aloitettiin vaatimusten mukaisten osien hankinnalla, joista luotiin neljä itsenäistä laitetta: kolme lähetintä ja yksi vastaanotin. Lähetyslaitteet kasattiin LPC1549-kehitysalustan ympärille, joka keräsi tietoa antureilta ja lähetti niiden tiedot SX1272 LoRa -moduulin avulla vastaanottimelle, joka toimi lähettimien keskittimenä. Vastaanotin koostui LPC1549-kehitysalustasta ja SX1272-moduulista, jotka yhdistettiin Raspberry Pi 3 -tietokoneeseen vastaanotetun tiedon esittämiseksi ja toteamiseksi.

Insinöörityön lopputuloksena luotiin toimiva paikallinen LoRa-verkko neljän itsenäisen laitteen välillä, mutta laiteiden lähettämä tieto ei ollut salattu, eikä tietoa saatu pilveen verkkoprotokollan puutteen vuoksi. Digitan LoRaWAN IoT -verkkoon yhdistämiseksi tehtiin toinen LoRaWAN-laite Arduino Uno -kehitysalustan ja SX1272-moduulin avulla, mikä saatiin yhdistettyä pilveen.

Avainsanat	LoRa, LoRaWAN, langaton, tiedonsiirto, IoT, LPC

Author Title Number of Pages Date	Jarkko Aho LoRaWAN as a Part of IoT-Home Technology 44 pages + 1 appendix 1 November 2019
Degree	Bachelor of Engineering
Degree Programme	Information and Communication Technology
Professional Major	Smart Systems
Instructors	Kimmo Sauren, Senior Lecturer Keijo Länsikunnas, Senior Lecturer

IoT devices and low-power wireless communication technologies have developed rapidly over the last 10 years, but which technology is the best? The purpose of this thesis was to study the operation of LoRaWAN technology and how it can be applied to IoT devices as part of IoT-home technology. The requirement for this thesis was to study and understand the LoRa technology, its associated modulation and protocols, and to create a prototype that would send sensor data to the cloud.

The background research for this thesis focused on the LoRa and LoRaWAN technologies, their architecture, low power, capacity and security. In the background study, a wide-ranging comparison was carried out between the most prominent LPWAN wireless technologies and their uses and applications were compared with LoRaWAN.

The implementation of the prototype began with the purchase of compliant components that created four independent devices: Three transmitters and one receiver. The transmitters were connected to a LPC1549 development platform, which collected data from the sensors and sent their data via the SX1272 LoRa module to a receiver that served as a concentrator. The receiver consisted of a LPC1549 development platform and a SX1272 module, which were connected to a Raspberry Pi 3 computer to display and verify the received information.

As a result of the thesis, a working local LoRa network was created between the four independent devices, but the data transmitted by the devices was not encrypted and the data was not sent to the cloud due to lack of network protocol. To connect Digita's LoRaWAN IoT network, another LoRaWAN device was made using the Arduino Uno development platform and the SX1272 module, which was connected to the cloud.

Keywords	LoRa, LoRaWAN, wireless, communication, IoT, LPC

Sisällys

Lyhenteet

.cpp / .h	C++ ja header -tiedostolaajennukset.
ABP	*Activation by Personalisation.* Persoonallinen AES-avaimen aktivointi.
ALOHA	*Additive Links On-line Hawaii Area.* Hyvin yleinen MAC-kerrosprotokolla.
CSMA	*Carrier Sense Multiple Access.* Ethernetin MAC-protokolla.
CSS	*Chirp Spread Spectum.* LoRa-tekniikan viserryshajaspektri-taajuusmodulaatio.
D7A	*DASH7 Alliance Protocol.* DASH7-tiedonsiirtotekniikan virallinen lyhenne.
DSSS	*Direct-Sequence Spread Spectrum.* Pseudo-Noise-sekvenssejä käyttävä hajaspektri-taajuusmodulaatiotekniikka.
FIFO	*First-In First-Out.* Puskurista tai varastosta lähtevän tiedon tai tavaran poistumisidea.
FSK	*Frequency Shift Keying.* Oskillaattorilla toimiva taajuusmodulaatiotekniikka, jonka taajuus muuttuu lähetettävän tiedon perusteella.
GPIO	*General-Purpose Input Output.* Yleiskäyttöinen vastaanotin-lähetinportti mikroprosessoreissa.
I^2C	*Inter-Integrated Circuit.* Kaksisuuntainen ohjaus- ja tiedonsiirtoväylä mikrokontrollerin ja komponentin välillä.
IoT	*Internet of Things.* Esineiden internetillä tarkoitetaan internetverkon laajentumista laitteisiin ja koneisiin.
LMiC	*IBM LoRaWAN C-library.* Sovelluspohjainen MAC-protokolla Semtech SX1272 -radiomoduulille.

LoRa	*Long Range Radio.* Pitkän kantaman digitaalinen langattoman tiedonsiirtoteknologia.
LoRaWAN	*Long Range Wide Area Network.* LoRa-teknologian verkko, missä useampi LoRa-laite on kytkettynä samaan verkkoon.
LPWAN	*Low Power, Wide-Area Network.* Yleinen nimitys pienivirtaisille laajaverkoille.
LTE	*Long-Term Evolution.* Tiedonsiirtotekniikka, joka on suunniteltu laajakaistaisen internetyhteyden käyttöön.
MAC	*Medium Access Control.* Verkon varaamisen ja liikennöinnin hoitava osajärjestelmä.
MCU	*Microcontroller Unit*. Mikrokontrolleri yksikkö, joka on osa kehitysalustaa.
NB-IoT	*Narrowband-IoT.* LTE-tekniikkaan perustuva langaton tiedonsiirtoprotokolla.
OFDM	*Orthogonal Frequency Division Multiplexing.* Monikantoaaltojen modulaatiomuoto, joka koostuu joukosta lähekkäin sijoitettuja kantoaaltoja.
OFDMA	*Orthogonal Frequency Division Multiple Access.* OFDM-modulaation monikäyttäjämuoto, joka jakaa alikantoaallot ryhmiin.
OTAA	*Over-the-Air Activation.* Liittymispohjainen AES-avaimen aktivointi.
PSK	*Phase Shift Keying.* Vaiheavainnus modulaatiomenetelmä, missä signaali muuttaa kantoaallon vaihetta ja vaihe kertoo viestin arvon.
RTOS	*Real-Time Operating System.* Reaaliaikainen käyttöjärjestelmä.
SPI	*Serial Peripheral Interface.* Synkronoitu sarjaliikenneväylä.
UART	*Universal Asynchronous Receiver-Transmitter.* Asynkroninen sarjaliikenneväylä mikrokontrollerin ja komponentin välillä.

1 Johdanto

Esineiden internet, eli Internet of Things (IoT) on kasvanut ja kehittynyt nopeammin kuin mikään muu tekniikan ala viime vuosien aikana. Yhä useampi asia ja esine on yhdistetty internetiin, suuri osa jopa meidän tiedostamatta. Jokaiselle laitteelle ei ole mahdollista kytkeä langallista internetyhteyttä tai langatonta puhelinverkkoa. Syynä tähän voivat olla IoT-laitteen sijainti, virrankulutus tai laitteen tuotantokustannukset. Siksi meidän on kehitettävä ja löydettävä uusia, halvempia, tehokkaampia ja turvallisempia keinoja siirtää tietoa laitteelta toiselle.

Insinöörityön tavoite oli tehdä tutkimustyö LoRaWAN (Long Range Wide Area Network) langattomasta verkkoteknologiasta osana IoT-talotekniikkaa, ja toteuttaa prototyyppi SeniorSoft-startup yritykselle käyttäen kyseistä teknologiaa hyväksi. SeniorSoft on vuonna 2017 perustettu startup-yritys, joka tarjoaa IT-konsultointia ja -palveluita Suomessa. Tässä insinöörityöraportissa käydään läpi LoRaWAN-teknologian edut ja haitat osana IoT-talotekniikkaa ja verrataan sitä muihin vastaavanlaisiin langattomiin verkkoteknologioihin.

Prototyyppi tehtiin alusta asti ohjelmoimalla, eikä työssä käytetty valmiiksi ohjelmoituja kaupallisia LoRa-laitteita. Prototyyppi toteutettiin NXP Semiconductor LPCXpresso 1549 -kehitysalustalla ja Semtech SX1272 LoRa Transceiver -moduulilla, ja ohjelmoitiin NXP MCUXpresso -kehitysympäristössä C/C++ -ohjelmointikielellä. LoRaWAN-tutkimustyö aloitettiin ennen SeniorSoftin liittymistä projektiin. Työssä valmistunutta prototyyppiä oli tarkoitus käyttää pohjana SeniorSoftin tuotteeseen.

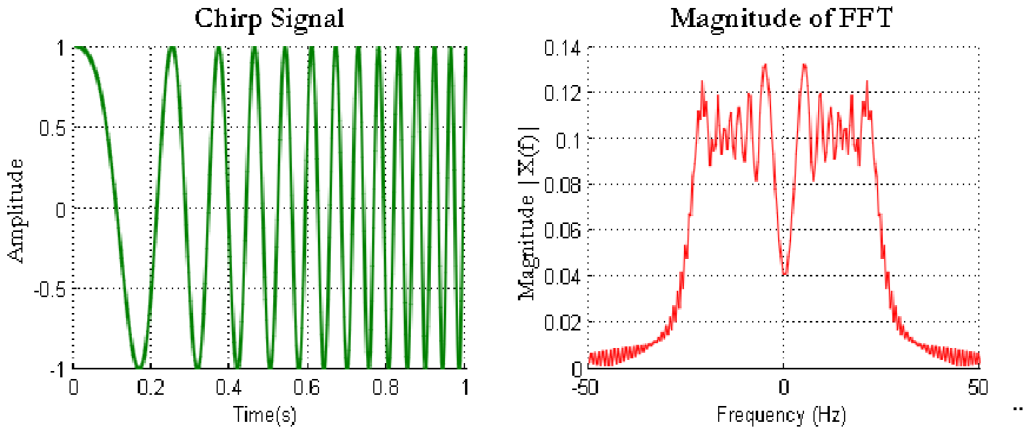
2 LoRa

LoRa eli Long Range on fyysisen tai langattoman kerroksen modulaatioratkaisu, jota hyödynnetään pitkän matkan viestintään. Useimmat langattoman tiedonsiirron järjestelmät käyttävät taajuusmodulointimenetelmää FSK (Frequency Shift Keying) fyysisenä kerroksena, koska se on erittäin tehokas vähävirtainen (Low Power, LP) modulaatioratkaisu. LoRa käyttää Chirp Spread Spectrum (CSS) hajaspektritekniikkaa, joka ylläpitää samoja vähävirtaisia ominaisuuksia kuin FSK-modulaatio, mutta mahdollistaa huomattavasti suuremman viestintäalueen. [1, s. 4.]

2.1 Chirp Spread Spectrum

Hajaspektritekniikoita käytetään yleisesti tele- ja radioviestinnässä. Hajaspektritekniikat ovat metodeja, joilla tietyllä kaistanleveydellä luotu signaali levitetään tarkoituksellisesti koko taajuusalueelle. Näin signaali luodaan entistä laajemmalla taajuusalueella ja nopeammalla tiedonsiirrolla. Nämä tekniikat ovat erityisen hyödyllisiä, kun halutaan tietoturvallinen viestintätekniikka, jolla on myös suuri vastustuskyky luonnollista häiriötä, radiomelua ja häirintää vastaan.

Chirp Spread Spectrum, eli viserryshajaspektritekniikka, käyttää laajakaistaisia, lineaarisia taajuusmoduloituja viserryspulsseja muuntamaan tietoa. Viserrys on siniaaltoinen taajuussignaali, joka nousee ja laskee ajan myötä (useimmiten polynomisesti ajan ja taajuuden suhteen). [2, s. 1–2.] Kuvassa 1 on viserryspulssi, jonka taajuus kasvaa lineaarisesti ajan myötä.



Kuva 1. Lineaarinen taajuusmoduloitu viserryspulssi aika- (vasen) ja taajuus tasossa (oikea) [3].

Kuten kaikki hajaspektritekniikat, CSS käyttää kokonaan sille myönnettyä kaistanleveyttä lähettämään signaalia, mikä tekee siitä kestävän kanavamelua kohtaan. Koska jokainen viserryspulssi hyödyntää laajaa taajuusaluetta, CSS on myös vastustuskykyinen monitie-etenemisen (Multipath ropagation) aiheuttamalle häiriölle, jopa vähävirtaisissa sovelluksissa. Toisin kuin DSSS tai FHSS-tekniikoissa (Direct-Sequence- ja Frequency-Hopping Spread Spectrum), CSS ei lisää mitään näennäissatunnaisia (pseudo random) elementtejä signaaliin, jotka auttaisivat erottamaan sen kanavanmelusta, vaan sen sijaan luottaa viserryspulssin suoraviivaiseen luonteeseen. [2, s. 2.]

Lisäksi, CSS on hyvin vastustuskykyinen Doppler-ilmiötä vastaan, mikä on tyypillinen ongelma liikkuville radiosovelluksille [4]. Tämän takia CSS-tekniikkaa saatetaan käyttää myös tulevaisuuden sotilaallisessa käytössä, koska pienellä virralla toimiessaan se on erittäin vaikea havaita esimerkiksi FMCW -tutkalla (Frequency-modulated Continuous-Wave). FMCW -tutka mittaa kohteen etäisyyttä Doppler-ilmiön avulla [5].

2.2 ALOHA-protokolla

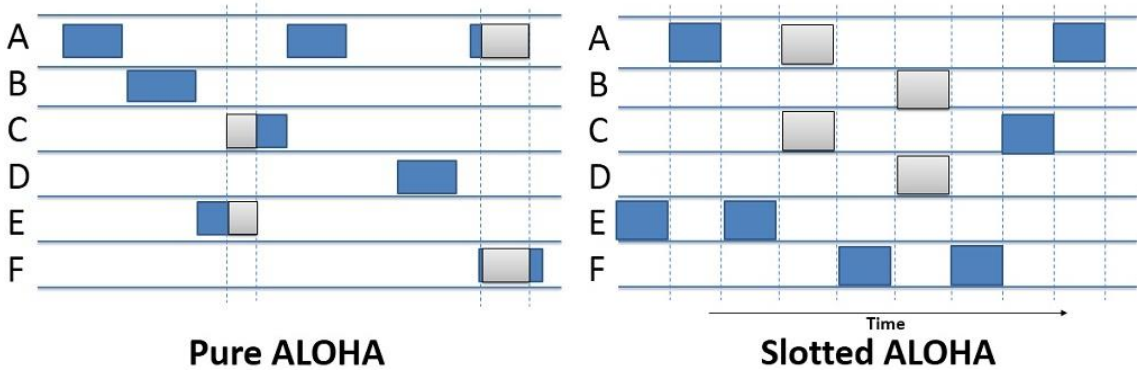
ALOHA tai ALOHAnet on nykyään hyvin yleinen MAC-kerros (Medium Access Control), ja se oli ensimmäinen random/multiple access -protokolla (satunnais-/monipääsy), jota on sovellettu laajempiin langattomiin verkkoihin. ALOHA, jonka alkuperäinen lyhenne oli Additive Links On-line Hawaii Area, kehitettiin nimensä mukaisesti Havaijin yliopistossa vuonna 1970. [6, s. 1–2.] 1980-luvun alussa mobiiliverkkojen ja langattomien verkkojen taajuudet avautuivat kuluttajakäyttöön, ja tämä mahdollisti ALOHA:n satunnaispääsytekniikan kehityksen molempiin verkkotekniikoihin.

2.2.1 Pure ALOHA

LoRa käyttää Pure ALOHA -protokollaa (P-ALOHA tai Classic ALOHA), jonka periaate perustuu hyvin yksinkertaiseen ajatukseen:

- Jos pääteasemalla on tietoa lähetettävänä, pääteasema lähettää tietoa.
- Jos tietoa lähettäessä asema vastaanottaa tietoa toiselta asemalta, on tapahtunut viestiristiriita. Kaikkien lähettävien asemien on yritettävä uudelleenlähetystä myöhemmin.

Ensimmäinen kohta viittaa, että pure ALOHA ei varmista, onko kanava varattu ennen lähetystä. Koska ristiriidat ovat mahdollisia ja tietoa täytyy mahdollisesti lähettää uudestaan, ALOHA ei voi käyttää sataa prosenttia yhteyskanavan kapasiteetista. Se kuinka kauan pääteasema odottaa ennen uutta lähetystä ja todennäköisyys, että uusi lähetysristiriita tapahtuu, vaikuttaa, kuinka tehokkaasti kanavaa voidaan käyttää. Tämä tarkoittaa, että konsepti ”lähetä myöhemmin” on kriittinen näkökohta. Valitun järjestelmän laatu vaikuttaa merkittävästi protokollan tehokkuuteen, lopulliseen kanavakapasiteettiin ja sen käyttäytymisen ennustettavuuteen. [7, s. 262–264.]

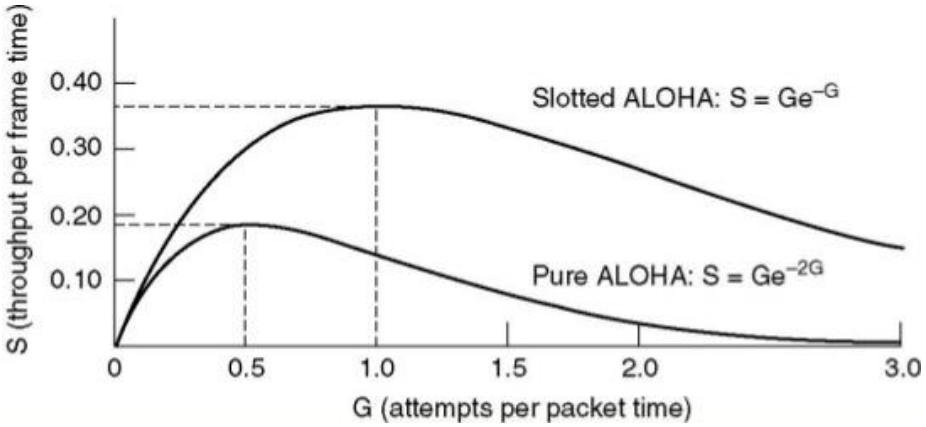


Kuva 2. Pure ALOHA ja Slotted ALOHA -protokollien eroavaisuus [8].

2.2.2 Slotted ALOHA

ALOHA-protokollasta on tehty paranneltu versio: Slotted ALOHA, eli Viipaloitu ALOHA -protokolla. Slotted ALOHA esitteli erilliset aikavälit, joka kasvatti onnistuneiden lähetysten määrää. Tällä protokollalla pääteasema voi aloittaa lähetyksen vain aikavälinsä alussa, mikä vähentää lähetysristiriitojen määrää. Tällöin kaksi tai useampi lähetysyritys voi olla ristiriidassa vain samalla aikavälillä, eikä ristiriidat enää tapahdu eri kohdassa lähetystä (kuva 2). Asemien synkronointi tapahtuu erillisen erityisaseman avulla, joka lähettää merkkiääntä jokaisen aikavälin alussa. Ikään kuin metronomi, mitä kaikki asemat kuuntelevat. [7, s. 264–265.] Slotted ALOHAa on ehdotettu LoRa-teknologian standardiksi, mutta sitä ei ole vielä otettu käyttöön. [9, s. 1.]

Slotted ALOHA -protokollaa käytetään myös sotilaallisten joukkojen matalan tiedonsiirtonopeuden satelliittitiedonsiirtoverkoissa, tilaajapohjaisissa satelliittiviestintäverkoissa, mobiilipuhelujen soittoasetuksissa, digisovittimien kommunikaatiossa ja kontaktittomissa RFID-teknologioissa (Radio-Frequency Identification). [7, s. 266.]



Kuva 3. Onnistuneen lähetyksen todennäköisyys aikavälissä verrattuna lähetysyritykseen aikavälissä ALOHA-protokollilla. [7, s. 265.]

Pure ALOHA -protokollan onnistuneiden lähetysten todennäköisyys voidaan laskea seuraavasti:

$$S_{pure} = Ge^{-2G} \quad (1)$$

S on onnistuneiden lähetysten todennäköisyys

G on keskiarvo Poissonin jakaumalla lähetettyjen lähetysyritysten määrä ajassa

e on Neperin luku

Ja Slotted ALOHA:

$$S_{slotted} = Ge^{-G} \quad (2)$$

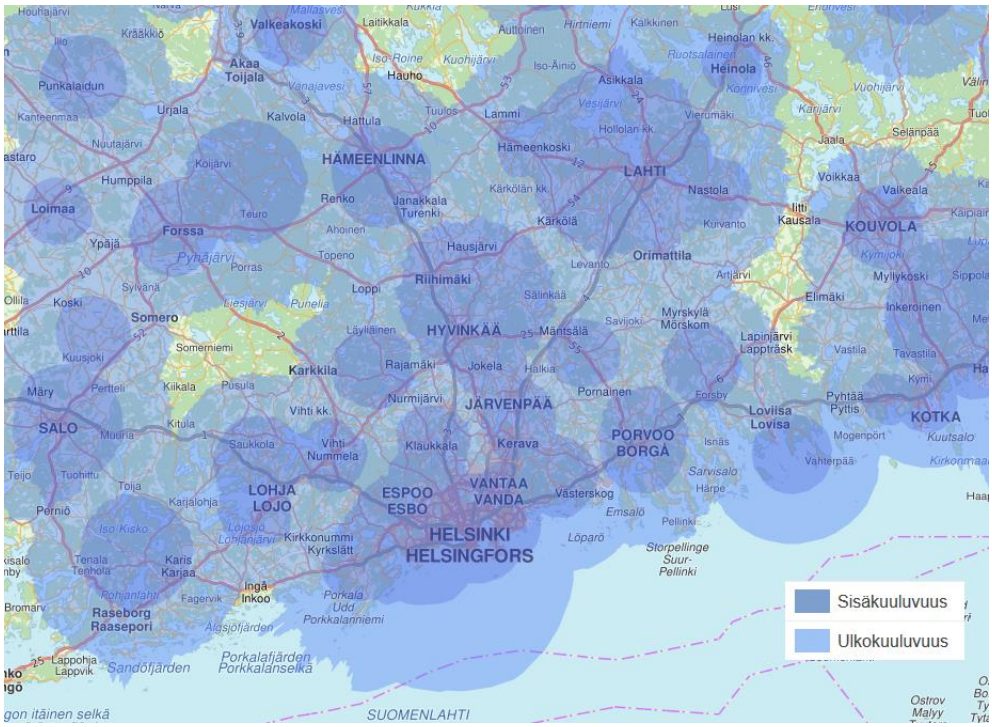
Pure- ja Slotted ALOHA -protokollien onnistuneen lähetyksen todennäköisyyserot voidaan nähdä kuvasta 3. Pure ALOHA:n korkein ”läpäisevyyden” todennäköisyys on 18,4 %, kun lähetysten määrä *G* = ½ ja Slotted ALOHA:n 36.8%, kun lähetysten määrä *G* = *1.* Tämä todistaa, että Slotted ALOHA -protokolla on jopa tuplasti todennäköisempi onnistumaan lähetyksessä kokonaisella aikavälillä. [7, s. 264–265.]

2.2.3 CSMA

ALOHAnetin satunnaispääsy kanava johti CSMA:n kehitykseen (Carrier Sense Multiple Access), jota voidaan kuvailla ”kuuntelee ennen lähetystä” MAC-protokollaksi. Tätä protokollaa voidaan käyttää, kun kaikki pääteasemat lähettävät ja vastaanottavat samalla kanavalla. CSMA:n ensimmäinen toteutus oli Ethernet. [7, s. 266–267.]

2.3 LoRaWANin kantavuus Suomessa

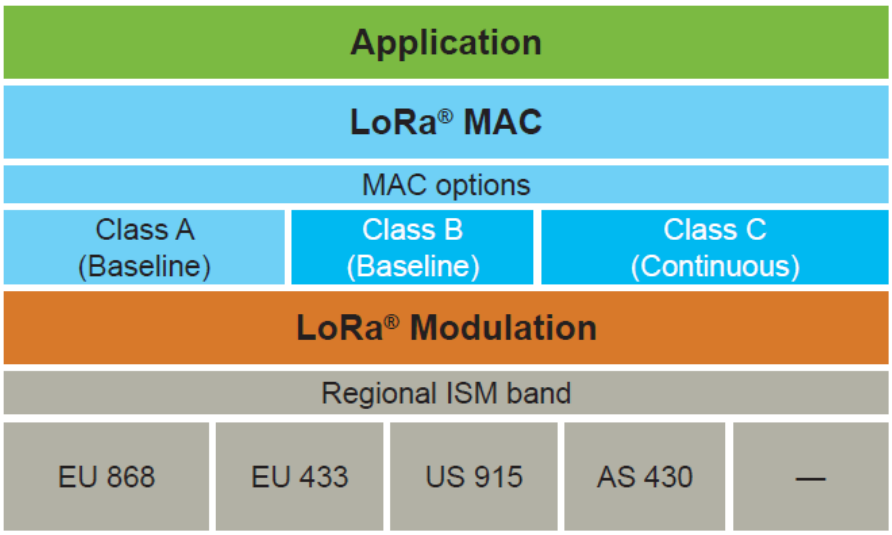
LoRa-teknologian etuna on pitkän kantaman valmiudet. Yksi yhdyskäytävä tai tukiasema voi kattaa kokonaisia kaupunkeja tai satoja neliökilometrejä. Alueen kantama riippuu ympäristöstä ja mahdollisista esteistä, mutta LoRa:n ja LoRaWAN:in Link Budget on suurempi kuin millään muulla standardoidulla viestintätekniikalla. Link Budget, eli vastaanottimen hyötysuhde, joka yleensä mitataan desibeleissä (dB), on ensisijainen tekijä kantaman määrittelyssä annetussa ympäristössä. Kuvassa 4 on Digitan IoT LoRaWAN -verkon peittokartta Etelä-Suomessa. Jokaisen tumman pallon keskellä on yksi tai useampi tukiasema.



Kuva 4. Digitan IoT LoRaWAN -verkon peittokartta Etelä-Suomessa. [10]

3 LoRaWAN

LoRaWAN määrittää tiedonsiirtoprotokollan ja järjestelmäarkkitehtuurin verkolle, kun LoRa-fyysinen kerros mahdollistaa pitkän matkan tiedonsiirtoyhteyden. Protokollalla ja verkkoarkkitehtuurilla on eniten vaikutusta määritettäessä IoT-laitteen akun käyttöikää, verkkokapasiteettia, Quality of Service (QoS), turvallisuutta ja muita verkon tarjoamia sovelluksia.



Kuva 5. LoRaWAN verkkoarkkitehtuuri. [1, s. 7.]

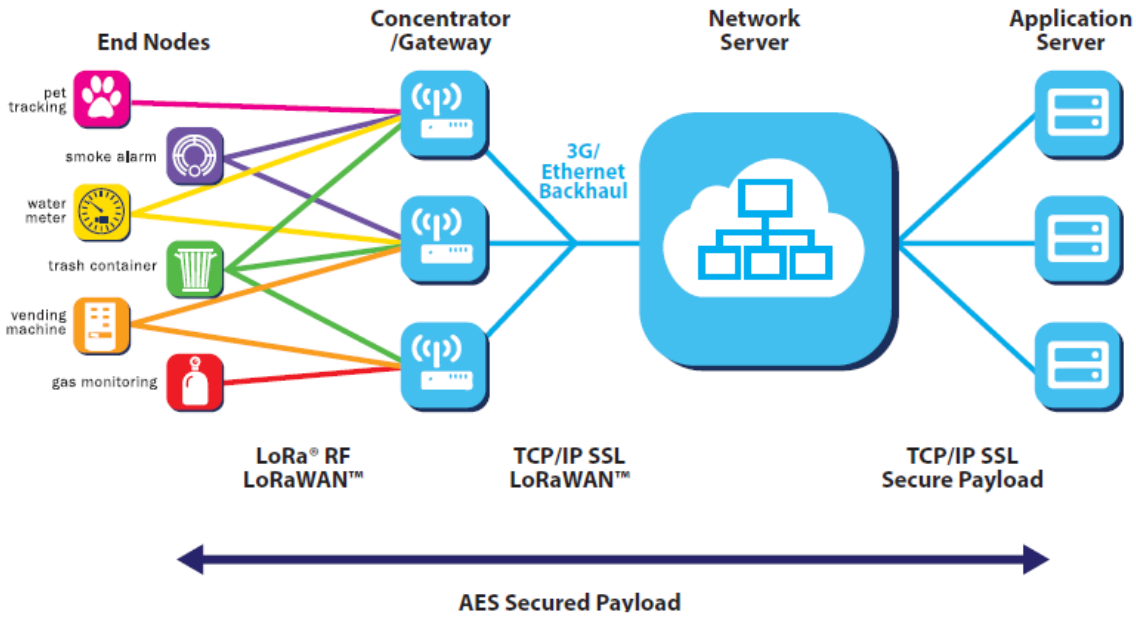
LoRaWAN on pienivirtainen laajaverkkoprotokolla (Low Power, Wide Area Network, tai LPWAN), joka on suunniteltu langattomasti liittämään akkukäyttöisiä IoT-laitteita internetiin, kohdentaen erityisesti kahdensuuntaiseen viestintään, end-to-end tietoturvaan ja lokalisointipalveluihin.

3.1 Verkkoarkkitehtuuri

Monet nykyään käytössä olevat verkot hyödyntävät reitittävää (Mesh) verkkoarkkitehtuuria. Mesh-verkossa yksittäiset päätteet (Node) siirtävät tietoa toisille päätteille, joka kasvattaa verkon suuruutta ja kantamaa. Tämän takia tietopaketit voivat saavuttaa päämääränsä useita eri reittejä ja yhden reitin katketessa käytetään vain toista reittiä. Jokaisen laitteen dynaamiset reititysalgoritmit auttavat tämän saavuttamisessa. Reititysprotokollien toteuttamiseksi jokaisen laitteen on kommunikoitava reititystiedot muille verkon lait-

teille. Kukin laite päättää, mitä vastaanotetulle tiedolle tehdään – joko siirretään se seuraavalle laitteelle tai säilytetään se protokollan mukaan. Käytetyn algoritmin tulisi aina varmistaa, että tieto menee sopivimman (nopeimman) reitin päämääräänsä.

Vaikka tämä parantaa verkon toiminnallisuutta, se myös tekee verkosta monimutkaisen, vähentää verkkokapasiteettia ja IoT-sovelluksissa vähentää laitteen akun käyttöikää, kun päätteet vastaanottavat ja lähettävät niille todennäköisesti turhaa tietoa. LoRa perustuu tähtiarkkitehtuuriin, joka mahdollistaa pitkän akun eliniän, sekä pitkän kantaman.



Kuva 6. LoRaWAN-verkon tähtiarkkitehtuuri (star-of-stars) esimerkki. [1, s. 8.]

LoRaWAN-verkon päätteitä ei ole liitetty yhteen tiettyyn yhdyskäytävään (gateway). Sen sijaan päätteen tiedot vastaanotetaan tyypillisesti useilla yhdyskäytävillä (kuva 6). Jokainen yhdyskäytävä välittää vastaanotetun paketin päätelaitteelta pilvipohjaiseen palvelimeen jonkin muun yhteyden avulla (esimerkiksi: ethernet, satelliitti, tai langaton). Monimutkainen tietojenkäsittely on jätetty palvelimelle, joka hallinnoi verkkoa ja suodattaa ylimääräiset vastaanotetut paketit. Samalla palvelin suorittaa turvatarkastukset, aikataulutunnisteet optimaalisen yhdyskäytävän kautta, mukautetun tiedonsiirronnopeuden, jne. Jos päätelaite on liikkuva tai liikkeessä, yhdyskäytävän siirto ei vaadi päätelaitteelta aktiivisia toimenpiteitä (kuten esimerkiksi 3G/4G-verkot tarvitsevat), mikä on kriittinen ominaisuus paikannukseen erikoistuville laitteille ja merkittävä kohdesovellutus IoT-laitteille.

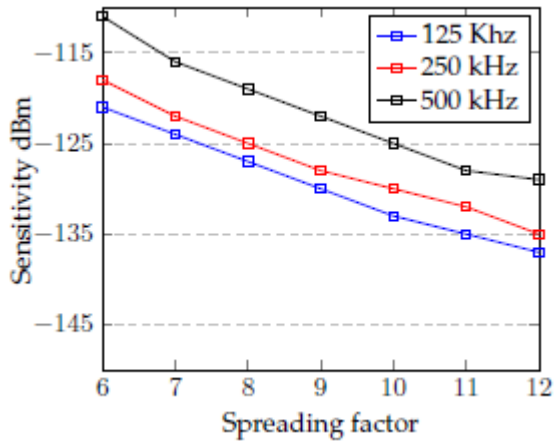
3.2 Akun elinikä

LoRaWAN-verkon päätelaitteet ovat asynkronisia ja lähettävät tietoa vain, kun niiden tietopaketti (payload) on valmis lähetettäväksi, joko tapahtumapohjaisesti tai aikataulullisesti (ALOHA-protokolla). Mesh- tai synkronisessa-verkossa, kuten puhelinverkossa, päätteet joudutaan usein herättämään verkon kanssa synkronointiin ja viestien lukemiseen. Tämä menetelmä käyttää huomattavasti energiaa ja on ensisijainen syy akun eliniän lyhenemiseen.

3.3 Verkkokapasiteetti

Jotta pitkän kantaman tähtiarkkitehtuurin verkko olisi toimiva, on yhdyskäytävällä oltava erittäin suuri kapasiteetti tai kyky vastaanottaa viestejä usealta päätteeltä yhtä aikaa. LoRaWAN-verkon suuri verkkokapasiteetti saavutetaan hyödyntämällä mukautumiskykyistä tiedonsiirtonopeutta (data rate) ja käyttämällä yhdyskäytävän monikanavaista vastaanotinta, jotta samanaikaisia viestejä voidaan vastaanottaa useilla kanavilla.

Kapasiteettiin vaikututtavat kriittiset tekijät ovat samanaikaisten kanavien lukumäärä, aika ilmassa (time on air), tietopaketin pituus, ja kuinka usein päätteet lähettävät. Koska LoRa on CSS-pohjainen modulaatio, signaalit ovat käytännössä päällekkäin (ortogonaalisia), eli alakanavien ylikuuluminen eliminoidaan, eikä välitaajuuksien suojakaistoja vaadita, kun käytetään erilaisia Spreading Factor (hajautuskerroin) arvoja. Spreading Factorin avulla määritetään, kuinka tiheään tieto on pakattu kantoaaltoon, ja tämän muuttuessa myös tiedonsiirtonopeus muuttuu. Kuva 7 esittää, kuinka työssä käytetyn Semtech SX1272 -moduulin kantavuus (Sensitivity) kasvaa annetulla kaistanleveydellä alentamalla tiedonsiirronnopeutta (nostamalla Spreading Factor -arvoa) tai vähentämällä signaalin kaistanleveyttä (vähentäen melua). Korkeampi kaistanleveys ei kasvata tiedonsiirronnopeutta, koska melun määrä kasvaa. [11, s. 25.]



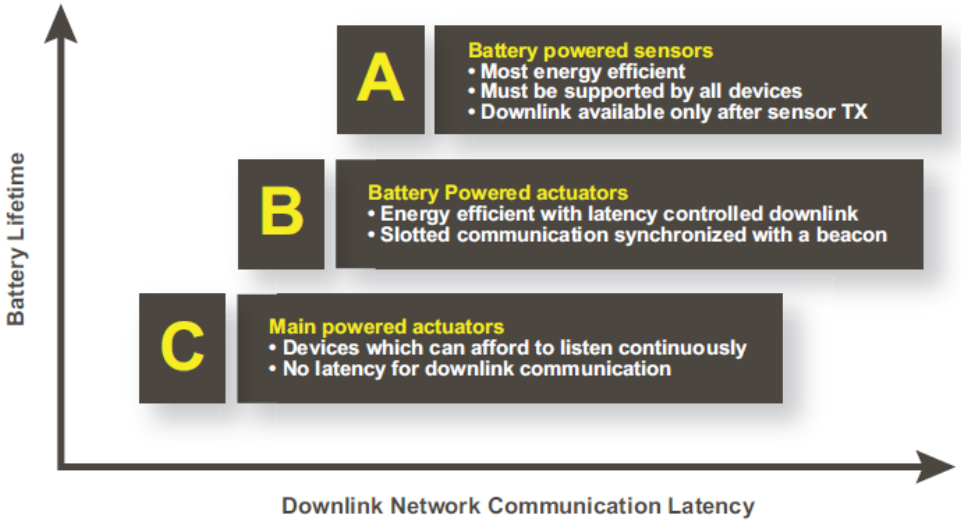
Kuva 7. Semtech SX1272 piirin Spreading Factor ja kaistanleveys verrattuna kantavuuteen (Sensitivity). [11, s. 25.]

Yhdyskäytävä hyödyntää tätä ominaisuutta vastaanottamalla useita eri tiedonsiirtonopeuksia samalla kanavalla samanaikaisesti. Jos päätteellä on hyvä yhteys ja se on lähellä yhdyskäytävää, sillä ei ole mitään syytä käyttää aina pienintä tiedonsiirronnopeutta ja täyttää käytettävissä olevat taajuudet kauemmin kuin sen tarvitsee. Muuttamalla tiedonsiirtonopeutta korkeammaksi, aika ilmassa lyhenee, mikä avaa enemmän mahdollista tilaa muille päätteille.

Mukautumiskykyinen tiedonsiirtonopeus parantaa myös päätteen akun elinikää. Jotta mukautumiskykyinen tiedonsiirtonopeus toimisi, tarvitaan riittävästi siirtotien kapasiteettia, joka saavutetaan symmetrisellä vastaanotto- ja lähetyslinkkillä (downlink ja uplink). Nämä ominaisuuksien ansiosta LoRaWAN-verkostossa on erittäin suuri kapasiteetti ja verkon laajennettavuusmahdollisuudet. Verkko voidaan asentaa hyvin vähäisellä infrastruktuurilla, mutta lisäämällä enemmän yhdyskäytäviä verkon kapasiteetti ja tiedonsiirtonopeus kasvavat, joka vähentää päällekkäisten lähetysten määrää muille yhdyskäytäville. Uuden yhdyskäytävän lisääminen kasvattaa kapasiteettia jopa 6-8-kertaisesti. [1, s. 10.] Muilla LPWAN-vaihtoehdoilla ei ole LoRaWAN-verkon laajennettavuusmahdollisuutta johtuen tekniikan kompromisseista, jotka rajoittavat vastaanottokapasiteettia tai tekevät vastaanottoalueen epäsymmetriseksi lähetysalueelle.

3.4 Laiteluokat

Päätelaitteita käytetään eri sovelluksiin ja niillä on erilaiset vaatimukset. Erilaisten päätelaitesovellusten optimoimiseksi LoRaWAN-verkon hyödyntävät erilaisia laiteluokkia. Laiteluokat eritellään latausnopeuden viiveen ja akun käyttöiän välillä. Ohjaus- tai toimilaitteen tyypillisessä sovelluksessa latausnopeuden viive on tärkeä tekijä.



Kuva 8. LoRaWAN-verkon laiteluokkien käyttötarkoitukset. [1, s. 10.]

LoRaWAN-verkolla on kolme erilaista päätelaiteluokkaa erilaisten sovellusten tarpeiden huomioonottamiseksi (kuva 8).

Luokan A päätelaitteet mahdollistavat kahdensuuntaisen tiedonsiirron, missä jokaisen päätelaitteen lähetystä (uplink) seuraa kaksi lyhyttä vastaanottoikkunaa (downlink). Päätelaitteen ajoitettu lähetyskohta perustuu sen omiin viestintätarpeisiinsa pienellä satunnaisajankohtaisella vaihtelulla (ALOHA-protokollan aikataulutus). A-luokan toimintaperiaate on vähävirtaisin päätelaitesovellus, joka tarvitsee vain vastaanottoyhteyden palvelimelta hetki sen jälkeen, kun päätelaite on lähettänyt. Päätelaite voi siirtyä virransäästötilaan niin pitkäksi aikaa kuin sen omalla sovelluksellaan on määritetty. Vastaanottoyhteys palvelimelta muulla aikavälillä on odotettava seuraavaa ajoitettua lähetystä.

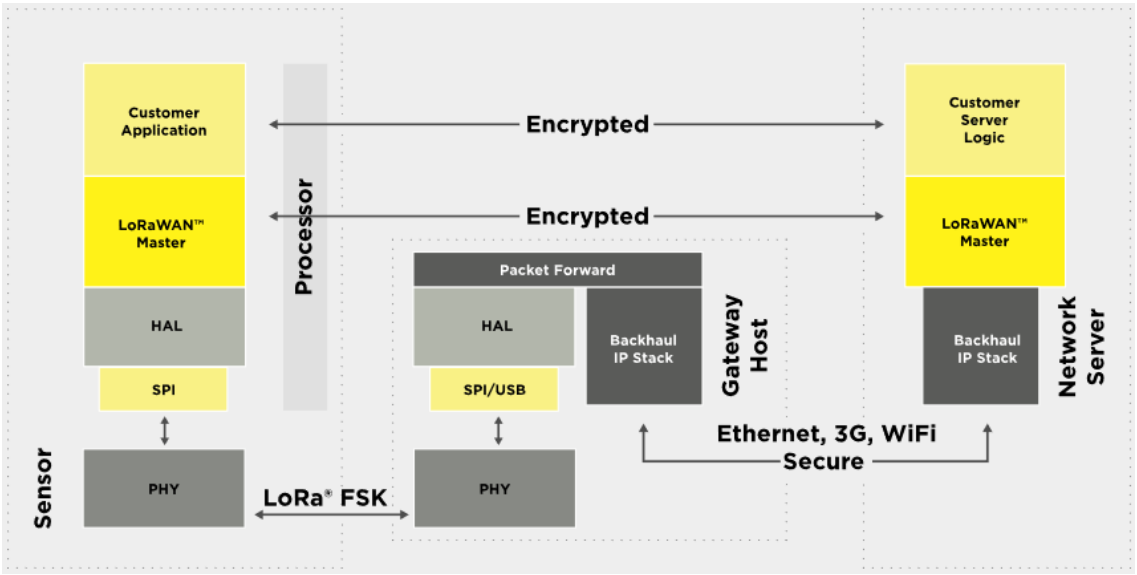
A-luokan satunnaisten vastaanottoikkunoiden lisäksi B-luokan laitteet avaavat ylimääräisiä vastaanottoikkunoita aikataulun mukaisesti. Jotta päätelaite pystyy avaamaan vastaanottoikkunansa aikataulussa, se vastaanottaa aikasynkronoidun signaalin yhdyskäytävältä. Tämä antaa palvelimelle mahdollisuuden tietää, milloin päätelaite kuuntelee, mutta vaatii päätelaitteelta ylimääräistä virrankulutusta. Ikkunan viive on mahdollista ohjelmoida 128 sekuntiin asti erilaisten sovellusten mukaiseksi ja ylimääräinen virrankulutus on tarpeeksi alhainen, jotta se olisi vielä käytettävissä akkukäyttöisille sovelluksille.

Luokka C vähentää viivettä vastaanotolle pitäen lähes jatkuvasti vastaanottoikkunoita auki, jotka suljetaan vain lähettäessä (half duplex). Tämän avulla palvelin voi aloittaa vastaanottolähetyksen milloin tahansa sillä oletuksella, että päätelaitteen vastaanotin on aina auki, joten viivettä ei juuri ole. C-luokka käyttää jopa ~50 mW virtaa ja sopii siksi vain sovelluksiin, joissa on jatkuvasti virtaa käytettävissä.

Akkukäyttöisillä laitteilla tilapäinen tilanvaihto A- ja C-luokkien välillä on mahdollista, ja se on hyödyllistä ajoittaisiin tehtäviin, kuten laiteohjelmiston päivityksiin.

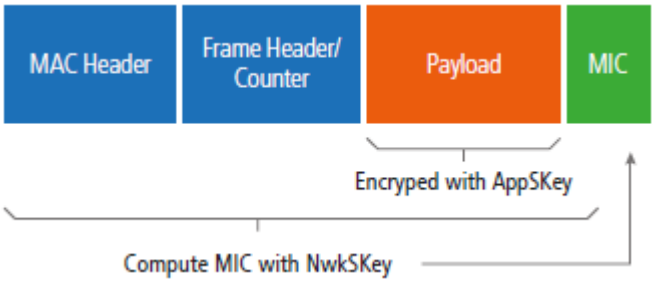
3.5 Tietoturva

Tietoturva on ensisijainen huolenaihe IoT- ja LPWAN-järjestelmien laitteissa. LoRaWAN käyttää kahta tietoturvatasoa; yhtä verkkoon ja yhtä sovelluksille. 128-bittinen verkkoistuntoavain (Network Session Key tai NwkSKey) takaa päätteen aitouden samalla kun 128-bittinen sovellusavain (Application Key, AppKey tai Application Session Key, AppSKey) varmistaa, että verkko-operaattorilla ei ole pääsyä loppukäyttäjän sovellustietoihin. AES-salausalgoritmia (Advanced Encryption Standard) käytetään avainvaihdossa. Sekä laitteella että sovelluksella on 64-bittinen IEEE EUI64-tunniste (DevEUI ja AppEUI), joita käytetään pakettien varmistukseen palvelimen ja päätteiden välillä. Näiden kahden tason avulla on mahdollista toteuttaa LoRaWAN-verkon tähtiarkkitehtuuri, ilman että verkko-operaattorien edes tarvitsee nähdä käyttäjien lähettämää tietoa.



Kuva 9. Tietoturva-anturin ja palvelimen välillä LoRaWAN ja LoRa FSK-tekniikalla [12].

Avaimet voidaan aktivoida päätelaitteen ja palvelimen kanssa kahdella eri tavalla: Liittymismenettely (Over-the-Air Activation, OTAA) edellyttää MIC-todistetta (Message Integrity Code) AppKeystä molemmilla sekä päätelaiteella että palvelimella. Tämä todiste luodaan AppKeyn avulla käyttäen AES-CMAC-algoritmia (Cipher-based Message Authentication Code) päätelaitteen lähettäessä liittymispyyntöä palvelimelle. Tämän jälkeen saadaan kaksi istuntoavainta: NwkSKey ja AppSKey. NwkSKeyllä salataan koko lähetys LoRaWAN MAC –komentoja ja tietopakettia myöten näyttäen vain MIC-todisteen, ja AppSKeyllä salataan tietopaketin sisältö (kuva 10).



Kuva 10. LoRaWAN-tietopaketin rakenne ja turva-avaimien sijainti paketissa. [13, s. 3.]

NwkSKey jaetaan koko LoRaWAN-verkolle, jotta pakettien aitous ja yhtenäisyys voidaan todistaa ja tarkistaa. AppSKey jaetaan sovelluksen palvelimelle, jotta tietopakettien sisältö voidaan salata ja purkaa. AppKey ja AppSKey voidaan piilottaa verkon operaattorilta, jotta se ei voi purkaa tietopaketin sisältöä, ennen kuin paketti on saavuttanut sovelluksen palvelimen. OTA-aktivointi soveltuu erityisen hyvin suuren tuotantolinjan LoRaWAN-laitteisiin, koska aktivointi voidaan suorittaa asennuksen jälkeen.

Persoonallinen avaimen aktivointi (Activation by Personalisation, ABP) toimii muuten samalla tavalla kuin OTA-aktivointi, mutta tässä tapauksessa sekä päätelaite että palvelin tietävät jo AppSKeyn ja NwkSKeyn. Nämä avaimet voidaan luoda myös etukäteen ennen laitteen asentamista, mutta tämä edellyttää, että avaimet kirjoitetaan suoraan laitteen ja palvelimen tietoihin. ABP-aktivointi soveltuu parhaiten yksilöllisiin LoRaWAN-laitteisiin.

3.6 Alueelliset muutokset

LoRaWAN-taajuudet vaihtelevat maailmanlaajuisesti eri alueellisten radio- ja sääntövaatimusten takia. LoRaWAN käyttää alhaista 433 – 928 MHz:n lisensoimatonta radiotaajuutta, minkä takia LoRaWAN:ia käytettäessä ei tarvitse maksaa lähetyskuluja. LoRaWAN:in radiotaajuutta voidaan sanoa alhaiseksi, koska esimerkiksi langaton paikallinen verkko WLAN (Wireless Local Area Network) käyttää 2.4 – 5 GHz radiotaajuutta.

Taulukko 1. LoRaWAN-taajuussuunnitelmat eri maanosissa ja valtioissa. [14, s. 15.]

Taajuussuunnitelma	Käytetty nimi
Eurooppa 433-434 MHz	EU433
Kiina 470-510 MHz	CN470
Kiina 779-787 MHz	CN779
Eurooppa 863-870 MHz	EU868
Venäjä 864-870 MHz	RU864
Intia 865-867 MHz	IN865
Pohjois-Amerikka 902-928 MHz	US915
Australia 915-928 MHz	AU915
Korea 920-923 MHz	KR920
Muu Aasia 920-928 MHz	AS923

Nämä lisensoimattomat radiotaajuudet vaihtelevat maanosien ja valtioiden välein. Esimerkiksi Euroopan LoRaWAN-taajuudet ovat 433 – 434 MHz ja 863 – 870 MHz, kun taas Pohjois-Amerikan taajuudet ovat 902 – 928 MHz. Taajuuksien erottamisen yksinkertaistamiseksi LoRa Alliance on keksinyt jokaiselle taajuussuunnitelmalle oman nimen (taulukko 1).

Euroopan 863 – 870 MHz:n taajuuden alataajuus on 863 – 865 MHz, eli kanava 70 (Channel 70) on käytössä hyvin monessa radiolaitteissa. Tämän takia Euroopassa saattaa ilmaantua paljon radiohäiriötä kyseisellä taajuudella, varsinkin alueilla missä on useita laitteita. Näitä laitteita ovat muun muassa langattomat kuulokkeet ja mikrofonit, huonokuuloisten kuulokojeet ja muut LPWAN-laitteet. 868 – 870 MHz taajuudella myös termostaattien, tulipalojärjestelmien, varkaudenestolaiteiden ja DIN-lähetin-vastaanottimien (Deutsches Institut für Normung -liitäntästandardi) voi olla vaikea kommunikoida, jos lähistöllä on 800 MHz taajuuden lähettimiä.

Euroopassa 863 – 870 MHz:n kaista on varattu lisenssitöntä käyttöä varten FHSS-, DSSS- tai analogisella modulaatiolla, lähetysteholla 0,1 %, 1 % tai 10 % taajuuden mukaan. Lähetysteho on rajoitettu käyttösyklin (duty cycle) rajoittamisella, eli laitteiden lähetysaikaa rajoitetaan prosenttien mukaisesti. LoRaWAN-tekniikka on rajoitettu Euroopassa 1 %:n lähetystehoon, joka hidastaa tiedonsiirtonopeutta huomattavasti [15].

Kaikissa teknologian valinnoissa on kompromisseja, mutta LoRaWAN-verkon ominaisuudet verkkoarkkitehtuurissa, laiteluokissa, tietoturvallisuudessa, kapasiteetin skaalautuvuudessa ja liikutettavuuden optimoinnissa antavat eniten potenttialisia mahdollisuuksia IoT-sovelluksiin. CSS-hajaspektritekniikan, usean kanavan ja vähäisen lähetettävän tiedonmäärän ansiosta LoRaWAN-tekniikan tietopakettien törmäysten määrä jää pienemmäksi kuin perinteisessä radioliikenteessä.

4 LoRaWAN-vertailu

LoRaWAN:ia vastaavia langattoman tiedonsiirtoteknologioita on useita, joista jokainen soveltuu eri käyttötarkoituksiin. Tiedonsiirtotekniikkaa valittaessa täytyy miettiä tietopaketin suuruutta, lähetysnopeutta, lähetyksen kantamaa ja akun virrankulutusta. Taulukossa 2 on vanhempia LoRaWAN-teknologiaa vastaavia vaihtoehtoja ja taulukossa 3 verrataan LoRaWAN-teknologiaa tunnetuimpiin telekommunikaatiomenetelmiin. Taulukon jälkeen verrataan näiden teknologioiden ominaisuuksia ja kerrotaan, miksi LoRaWAN on yleistymässä erityisesti IoT-talotekniikassa. Listasta on jätetty pois kaksi yleisintä langattoman tiedonsiirtoteknologiaa: Bluetooth ja WLAN. Nämä jätettiin pois vertailusta, koska Bluetooth ei ole verkko, se vaan on langatonyhteys kahden laitteen välillä, ja WLAN on nimensä mukaisesti vain paikallinen verkko, missä päätelaitteet keskustelevat yhden ennalta määrätyn yhdyskäytävän kanssa.

Taulukko 2. LPWAN tiedonsiirtoteknologioiden ominaisuuksia.

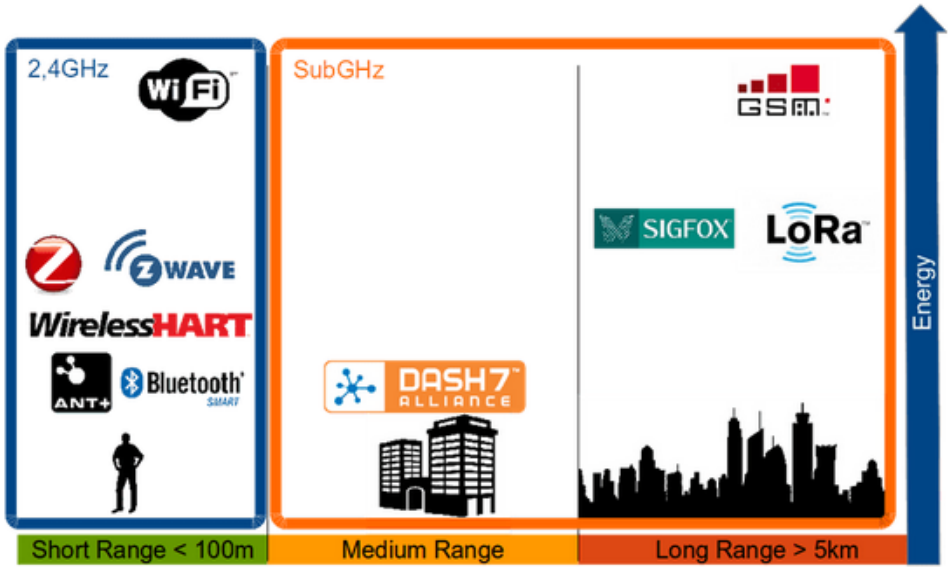
Ominaisuus	LoRaWAN	Sigfox	ZigBee	DASH7
Modulaatio	CSS	BPSK	QPSK	GFSK/FSK
Kaistanleveys	125–500 kHz	200 Hz	2 MHz	0.5 – 1.75 MHz
Tiedonsiirtonopeus	290 b/s– 50 Kb/s	10 b/s – 1 Kb/s	20 Kb/s – 250 Kb/s	13 Kbs – 100 Kbs
Max. # lähetyksiä/päivä	Rajaton	Max 140 lähetystä päivässä	Rajaton	Rajaton
Max lähtöteho	20 dBm	20 dBm	20 dBm	27 dBm
Link Budget	154 dB	151 dB	103 dB	140 dB
Akun elinikä – 2000 mAh	105 kuukautta	-	-	-
Tehon hyötysuhde	Korkea	Keskiverto	Korkea	Hyvin korkea
Häiriönsieto	Hyvin korkea	Matala	Keskiverto	Keskiverto
Tietoturva	Kyllä	Ei	Kyllä	Kyllä
Liikutettavuus /paikannus	Kyllä	Rajattu liikkuvuus, ei paikannusta	Rajattu liikkuvuus, ei paikannusta	Paikannus

Taulokossa 2 on IoT-laitteille tarkoitettuja langattoman tiedonsiirtoteknologioita, jotka vastaavat LoRaWAN:ia. Muut teknologiat eivät pärjää LoRaWAN:ille joko kantamassa tai energian kulutuksessa.

4.1 Sigfox

LoRaWAN:in ominaisuuksia lähimpänä on ranskalainen Sigfox-yritys, joka keskittyy ainoastaan IoT-laitteiden vähävirtaiseen langattomaan tiedonsiirtoon, kuten LoRa Alliance. Suomessa Sigfoxin yhteyttä tarjoaa Connected Finland, joka on osa Connected Baltics -järjestöä [16].

Sigfoxin langaton kommunikaatio perustuu binääriseen vaiheavainnus (Binary Phase Shift Keying tai BPSK) -modulaatiomenetelmään. BPSK on nimensä mukaisesti kaksivaihe-eroinen laji vaiheavainnus (Phase Shift Keying tai PSK) modulaatiomenetelmistä, missä moduloiva signaali muuttaa kantoaallon vaihetta ja hetkellinen vaihe kertoo viestin arvon. BPSK:ssa kantoaalto ilmaisee binäärisen viestin arvon 0 tai 1, missä 0 voi olla siniaallon nolla astekulma ja 1 olisi +180 astetta. BPSK-lähetin synkronoi vastaanottimen kanssa lähettämällä nollavaiheista kantoaaltoa tietyn ajanjakson ajan ja vastaanotin synkronoi itsensä tälle binääriselle nollavaiheelle.



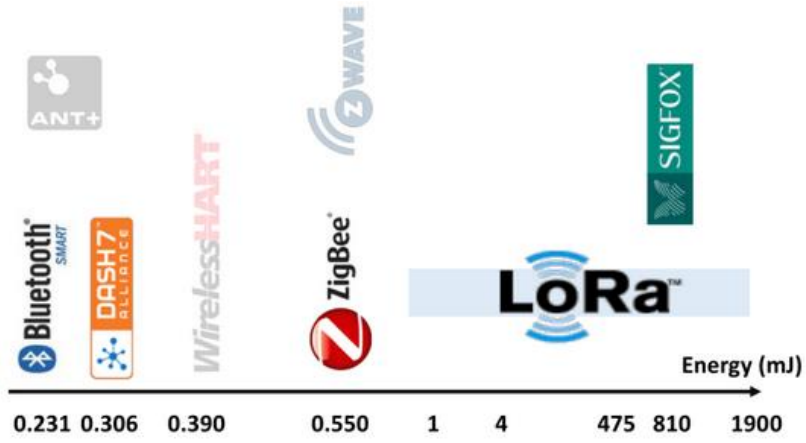
Kuva 11. Langattoman tiedonsiirtoteknologioiden kantavuuksia [17].

Sigfoxin kantama on yli 5 km, joka on silti pienempi kuin LoRaWAN:in luvattu 10 km (kuva 11). Sigfoxin laitteita ei voi käyttää ilman sopimusta paikallisen operaattorin kanssa, kun LoRa-verkon voi luoda ilman paikallisia operaattoreita ja LoRaWAN-verkon voi luoda omalla tukiasemalla. Sigfoxin tiedonsiirtoprotokolla ei sisällä omaa tietoturvallisuutta tai lähetyksen salausta. Sigfoxin tietoturva vaihtelee piirivalmistajan ja pilvipalvelun mukaan. [18, s. 6–8.]

4.2 ZigBee

Toisin kuin LoRaWAN ja Sigfox, ZigBee on tarkoitettu langattomaan persoonalliseen alueverkkoon (Wireless Personal Area Network, WPAN). WPAN-sovelluksessa laitteet keskustelevat keskenään luoden paikallisen mesh-verkon, joka ei tarvitse yhdyskäytävää. ZigBee-laitteille ei ole tarjolla operaattoreita, koska ZigBee on tarkoitettu persoonalliseen käyttöön.

ZigBee-kommunikaatiota voidaan käyttää kahdella eri modulaatiomenetelmällä. BPSK-menetelmää käytetään 868 ja 915 MHz:n taajuuksilla, ja nelivaiheista vaiheavainnusta (Quadrature Phase Shift Keying tai QPSK), jossa on neljä kantoaallon vaihetta, käytetään 2,4 GHz:n taajuudella [19]. QPSK toimii samalla periaatteella kuin BPSK, mutta se voi ilmaista numeroarvot 0-3, eli kaksi bittiä (00, 01, 10 ja 11). Tämä mahdollistaa joko kaksinkertaisen tiedonsiirtonopeuden samalla signaalin kaistanleveydellä, tai saman tiedonsiirtonopeuden puolet pienemmällä kaistanleveydellä kuin BPSK-modulaatiossa. Monet ZigBee-laitteiden radiomoduulit käyttävät DSSS-modulaatiota sen turvallisuuden, häiriönsietokyvyn ja pienivirtaisuuden vuoksi.



Kuva 12. Langattoman tiedonsiirtoteknologioiden energiankulutuksia [17].

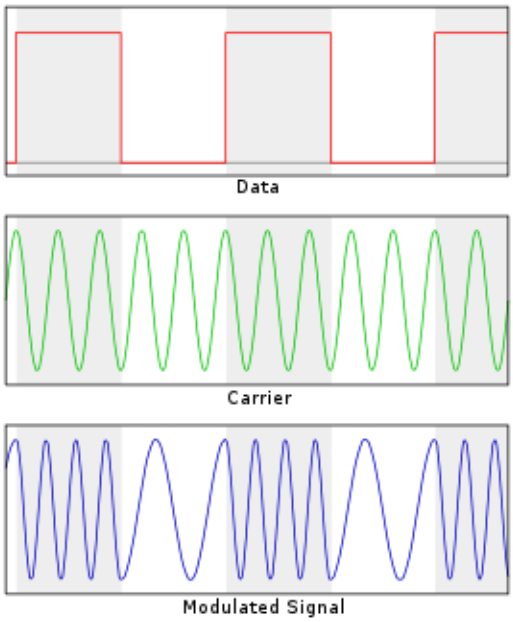
ZigBee-laitteet ovat halvempia ja vähävirtaisempia kuin LoRaWAN- ja Sigfox-laitteet (kuva 12). Pienempi teho vaikuttaa huomattavasti ZigBee-laitteiden langattoman tiedonsiirron kantamaan (kuva 8). ZigBee-laitteiden langaton tiedonsiirto kantaa vain 10-100 metriä, ja ne tarvitsevat näköyhteyden (line-in-sight) toisiinsa, jos 2,4 GHz taajuutta ei käytetä [20]. Tämän takia ZigBee-laitteita käytetään ensisijaisesti vain kodin älylaitteissa.

ZigBee-nimi vertaa hunajamehiläisten mehiläistanssiin (waggle dance), minkä mehiläiset tekevät pesään saavuttuaan. [21, s. 111.]

4.3 DASH7

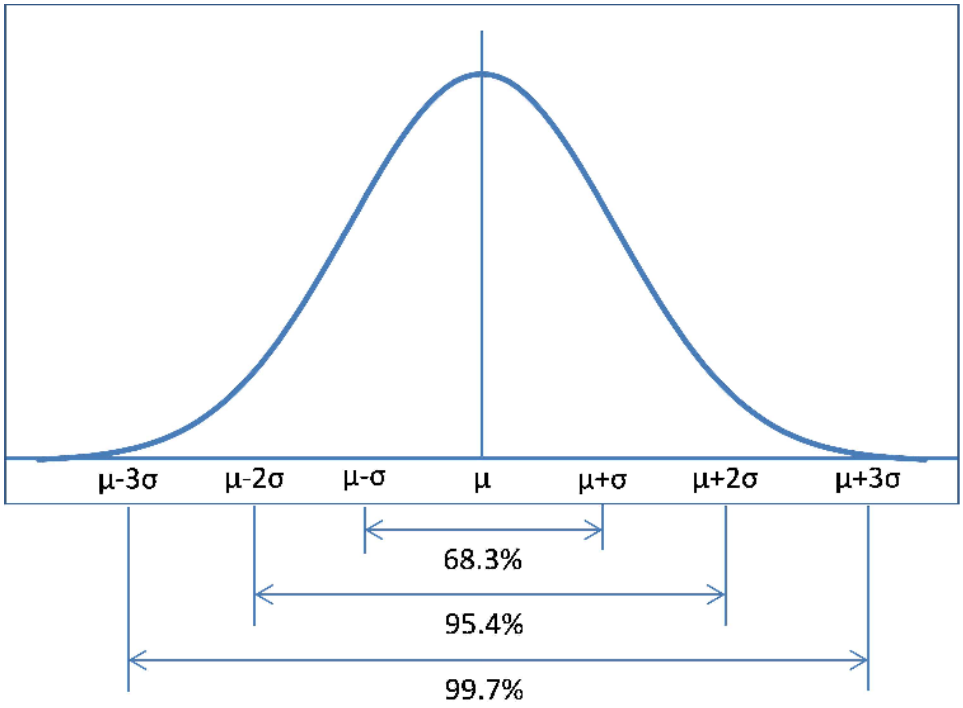
DASH7 Alliance Protocol, eli D7A putoaa kantavuudeltaan yllä olevien teknologioiden keskelle (kuva 11). D7A:n kantama on enintään 500 metriä, mutta se kuluttaa huomattavasti vähemmän virtaa kuin LoRaWAN (kuva 12). Tämän vuoksi D7A soveltuu erityisesti kerrostalojen toimilaiteiden, kuten hälytys-, anturi- ja tilatietojen keräämiseen.

D7A käyttää GFSK (Gaussain Frequency Shift Keying) -taajuusmodulointimenetelmää, jota käytetään myös Bluetooth laitteissa [22]. Yksinkertaisuudessaan FSK toimii oskillaattorilla, jonka ennalta määrätty taajuus muuttuu lähetettävän tiedon perusteella. Esimerkiksi Binääri FSK (BFSK) käyttää kahta toisista eroavaa taajuutta merkitsemään binäärejä (0 ja 1). BFSK:n tapaisessa järjestelmässä binääriä 1 kutsuttaan merkkitaajuudeksi, jonka taajuus on suurempi kuin kantotaajuuden ja binaaria 0 kutsutaan välitaajuudeksi, joka on kantotaajuutta huomattavasti hitaampi (kuva 13).



Kuva 13. Kantoaalto (carrier) muuttuu lähetettävän tiedon (data) mukaan moduloiduksi signaaliksi (modulated signal) BFSK-menetelmällä [23].

Sen sijaan, että taajuutta moduloidaan suoraan digitaalisilla datasymboleilla (binääreillä), muuttuu taajuus heti kunkin symbolijakson alussa, D7A:n käyttämä GFSK suodattaa tietopulssit Gaussin suodamessa (Gaussian filter), mikä tekee muutoksesta tasaisemman. Gaussin suodatus muotoilee pulssin Gaussin käyrän, eli normaalijakauman muotoiseksi, ennen modulointia. Tämä pienentää kaistanleveyden spektriä ja vähentää häiriötä vierekkäisten kanavien kanssa, mutta lisää symbolien välistä häiriötä (Intersymbol Interference). Gaussin suodatus on yleinen tapa pienentää kaistanleveyden spektriä taajuusmodulaatiossamenetelmissä. GFSK-sovelluksessa sitä kutsutaan pulssin muotoiluksi (pulse shaping), joka muokkaa pulssin Gaussin kellokäyrän muotoiseksi (kuva 14) [22].



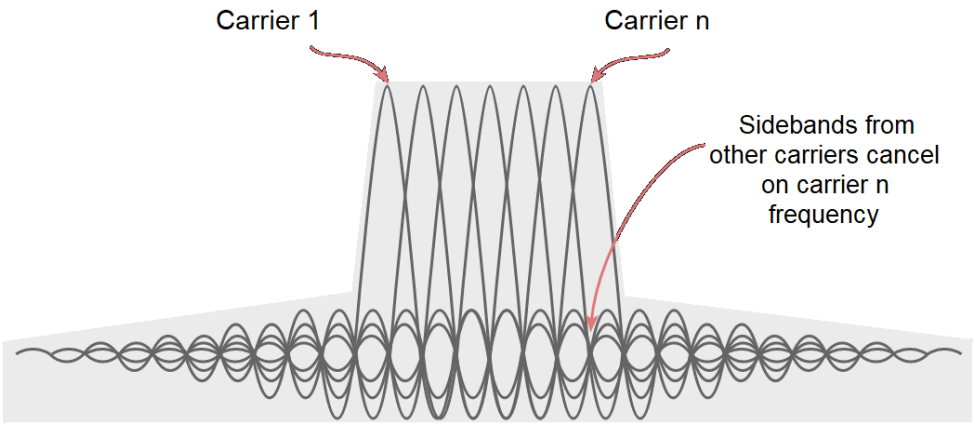
Kuva 14. Normaalijakauma, eli Gaussin kellokäyrä, jossa $\mu \in \mathbf{R}$ on jakauman oletusarvo (keskiarvo) ja $\sigma^2 > 0$ on jakauman varianssi.

D7A:n versiossa 1.2 on D7A-LoRa laajennus, joka käyttää LoRa:n fyysistä kerrosta. D7A-LoRa on integroitu tavalliseen D7A-pinoon. Tämä sallii saman pitkän kantaman kattavuuden kuin LoRaWAN, mutta pitää silti kaikki D7A:n ominaisuudet, mitä LoRaWAN ei tue [17].

4.4 LTE IoT

Long-Term Evolution, tai LTE on neljännen sukupolven (4G) langaton tiedonsiirtotekniikka, joka on suunniteltu laajakaistaisen internetyhteyden käyttöön [24]. LTE 4G tunnetaan parhaiten matkapuhelimien langattoman tiedonsiirron standardina. LTE perustuu OFDMA-taajuusmodulaatioon (Orthogonal Frequency Division Multiple Access), joka on monikäyttäjämuunnelma tunnetusta OFDM-modulaatiosta (Orthogonal Frequency Division Multiplexing).

OFDM on monikantoaaltojen modulaatiomuoto. OFDM-signaali koostuu joukosta lähekkäin sijoitettuja moduloituja kantoaaltoja (carrier), joiden sivukaistat (sideband) leviävät molemmille puolille kantoaaltoa. Kuten CSS-modulaatiossa, OFDM:n signaalit ovat ortogonaalisia, ja ne voidaan silti vastaanottaa ilman häiriötä, koska kantoaaltoväli on yhtä suuri kuin symbolijakso (kuva 15).



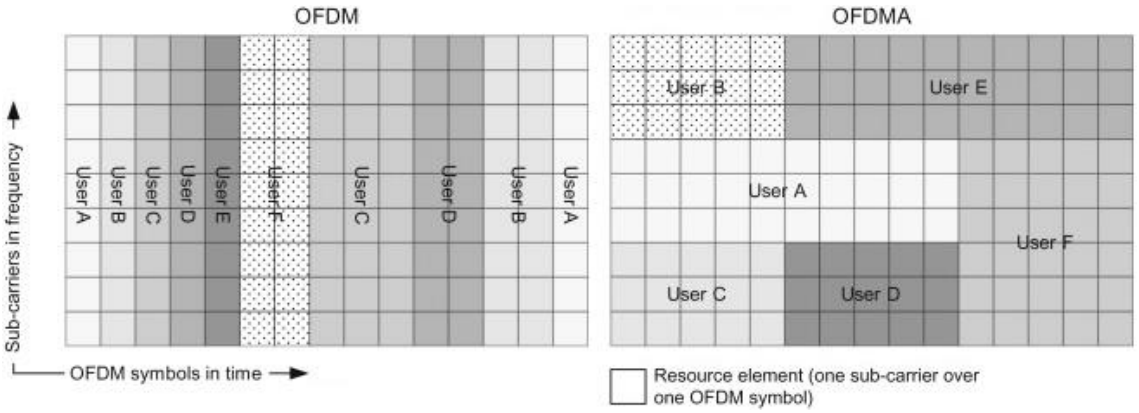
Kuva 15. OFDM-modulaatiosignaalin kantoaallot symbolijaksossa [25].

Yksi OFDM-lähetys- ja vastaanottojärjestelmien vaatimus on, että niiden on oltava lineaarisia. Mahdolliset epälineaarisuudet aiheuttavat häiriötä kantoaaltojen välillä moduloinnin vääristymisen seurauksena. Tämä tuo esiin ei-toivottuja signaaleja, jotka aiheuttavat häiriöitä ja heikentävät lähetyksen ortogonaalisuutta.

OFDM-signaalin sisällä tiedot siirretään rinnakkain eri kantoaaltojen välillä. Tieto jaetaan lukuisiin rinnakkaisiin alikantoaaltoihin matemaattisella IFFT-funktiolla (Inverse Fast Fourier Transform). Alikantoaaltojen tiedonsiirtonopeus on pienempi kuin kantoaallolla, ja symbolit ovat etäällä toisistaan aikajaksossa (kuva 16 vasemmalla). Tämä vähentää häiriötä helpottaa tarkkaa vastaanottamista samalla suorituskyvyllä [26]. Tämä tiedon

jakaminen antaa useita etuja. Monitiehäiriöt (Multi Path) ja muut taajuushäiriöt vaikuttavat vain pieneen osaan kantoaalloista ja muut vastaanotetaan oikein. Käyttämällä virhekoodaustekniikoita (error-coding technique), mikä tarkoittaa lisätiedon lisäämistä lähetettävään signaaliin, mahdollistetaan monien tai kaikkien vioittuneiden tietojen rekonstruoinnin vastaanottimessa.

OFDMA:n monikäyttö saavutetaan määrittämällä alikantoaaltojen alajoukot eri käyttäjille, mikä mahdollistaa samanaikaisen tiedonsiirron usealta käyttäjältä. OFDMA:ssa radioresurssit ovat kaksiulotteisia alueita aikajaksossa (OFDM-symboleiden kokonaislukuja) ja taajuudessa (useita jatkuvia ja jatkumattomia alikantoaaltoja) (kuva 16 oikealla).



Kuva 16. OFDM- ja OFDMA-radioresurssien eroavaisuus aikajaksossa ja taajuudessa [26].

Kuten OFDM, OFDMA käyttää useita lähekkäin sijaitsevia alikantoaaltoja, mutta alikantoaallot jaetaan ryhmiin, joissa kutakin ryhmää kutsutaan resurssilohkoksi tai alakanavaksi. Resurssilohkon muodostavien alikantoaaltojen ei tarvitse olla fyysisesti vierekkäisiä. Vastaanotossa (downlink) resurssilohko voidaan antaa eri käyttäjille, ja lähetyksessä (uplink) käyttäjä voidaan osoittaa yhteen tai useampaan resurssilohkoon.

IoT-laitteiden suosio ja kysyntä johti LPWAN vaihtoehtojen nousuun. Perinteiset LTE 4G -verkkotekniikat käyttävät liian paljon virtaa toimiakseen IoT-sovelluksissa, missä lähetetään harvoin ja vain pieniä määriä tietoa kerralla. Tätä varten kehitettiin useampi LTE IoT -protokolla, kuten LTE Cat-0, Cat-M1 ja Narrowband-IoT [27].

Taulukko 3. LoRaWAN- ja LTE IoT -tiedonsiirtoteknologioiden ominaisuuksia [27].

Ominaisuus	LoRaWAN	LTE Cat-0 2016(Rel12)	LTE Cat-M1 2018 (Rel13)	NB-IoT 2019 (Rel13+)
Modulaatio	CSS	OFDMA	OFDMA	DSSS
Kaistanleveys	125–500 kHz	20 MHz	1.4 MHz	180 kHz
Tiedonsiirtonopeus	290 b/s–50 kb/s	1 Mb/s	200 kb/s–1 Mb/s	~250 kb/s
Max. # lähetyk-siä/päivä	Rajaton	Rajaton	Rajaton	Rajaton
Max lähtöteho	20 dBm	23 dBm	20/23 dBm	20/23 dBm
Link Budget	154 dB	130 dB+	146 dB	150 dB
Akun elinikä – 2000 mAh	105 kuu-kautta	-	18 kuukautta	-
Tehon hyötysuhde	Korkea	Matala	Keskiverto	Korkea
Häiriönsieto	Hyvin korkea	Keskiverto	Keskiverto	Matala
Tietoturva	Kyllä	Kyllä	Kyllä	Kyllä
Liikutettavuus /paikannus	Kyllä	Kyllä	Kyllä	Rajattu liikkuvuus, ei paikannusta

Taulukossa 3 verrataan LoRaWAN-tekniikkaa erilaisiin LTE IoT -tekniikoihin. Taulukkoon on valittu vain kehitysaskeleet kohti Narrowband-IoT-protokollaa, joka on verrattavissa LoRaWAN-tekniikkaan ominaisuuksiltaan.

4.4.1 LTE Cat-0

LTE Cat-0, eli kategoria 0 oli ensimmäinen LTE-M-protokolla (LTE-M tai LTE-MTC, eli Machine Type Communication). Cat-0 kehitettiin aikaisemmasta Cat-1-protokollasta, eli 3G-verkosta. Cat-0 on paljon halvempi valmistaa kuin Cat-1, koska siitä otettiin pois ominaisuudet, jotka tukivat nopean tiedonsiirtonopeuden vaatimuksia. Vaikka Cat-0 oli kevyt versio edeltäjästään, se silti käytti paljon virtaa ja kykeni lähettämään IoT-sovelluksille turhan korkealla tiedonsiirtonopeudella (1 Mb/s) [27].

4.4.2 LTE Cat-M1

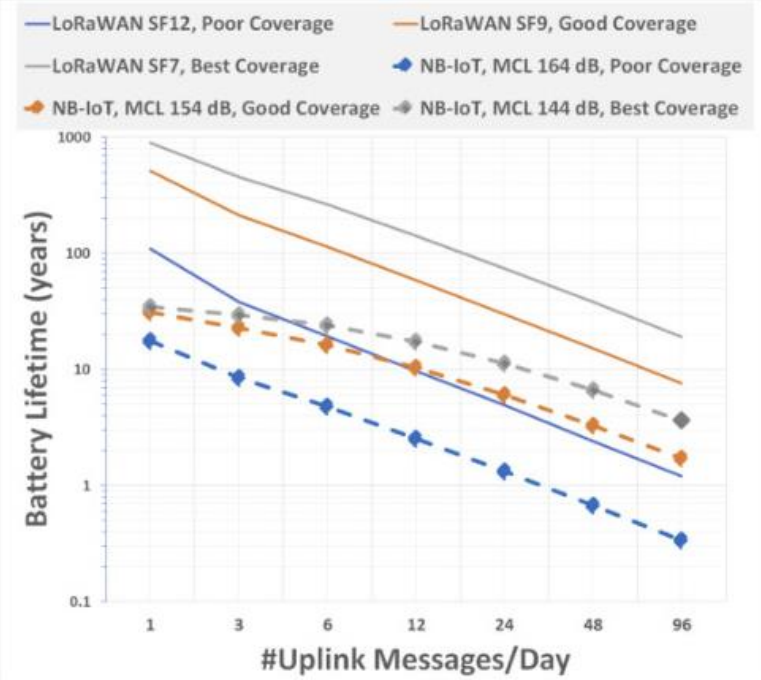
LTE Cat-M1 kuuluu samaan LTE-M-kategoriaan, mutta oli ensimmäinen eMTC (enhanced MTC) -standardi. Cat-M1 saavutti LPWAN-sovellukselle vaaditut virrankulutukset pienentämällä järjestelmän kaistanleveyttä 1,4 megahertsiin. Cat-M1:n suurin etu ylitse edeltäjien oli yhteensopivuus olemassa olleisiin LTE-verkkoihin. Teleoperaattorien ei tarvinnut rakentaa uusia antenneja Cat-M1:n käyttöönottoa varten, mutta LTE-verkot tarvitsivat silti ohjelmistopäivityksen [27]. Cat-M1-protokollaa käytetään edelleen kannettavissa IoT-laitteissa, kuten älykelloissa, potilasmonitoreissa ja liikuntarannekkeissa.

4.4.3 NB-IoT

Narrowband-IoT, eli NB-IoT eroaa huomattavasti muista LTE-tekniikoista. NB-IoT käyttää suorasekvenssihajaspektritaajuusmodulaatiota, eli DSSS-taajuusmodulaatiota (Direct-Sequence Spread Spectrum). DSSS, kuten LoRaWAN:in CSS, toimii hajaspektritekniikalla, joka mahdollistaa turvallisen, häiriösietoisen ja vähävirtaisen kommunikaation.

DSSS-tekniikassa lähetin monistaa databitit Pseudo-Noise-sekvenssillä ja hajautetaan koko kaistanleveydelle. Nämä hajautetut bitit lähetetään sitten suuremmalla taajuudella kantotaajuuden kanssa. Pseudo-Noise-sekvenssit ovat kohinalta (noise) kuulostavia deterministisiä, jaksollisia ja binaarisia sekvenssejä, jota kutsutaan myös näennäissatunnaislukukohinaksi (pseudo random noise), koska kohinasignaali on näennäissatunnaislukusekvenssi luvuista 1 ja -1. Vastaanotin sitten purkaa tämän signaalin käyttäen samaa Pseudo-Noise-sekvenssiä. Tämän ansiosta tieto voidaan tulkita oikein, vaikka kantotaajuus olisi 10 desibeliä pienempi kuin kohina, eikä saman taajuuden käyttö samalla alueella heikennä yhteyden laatua juuri yhtään. [2, s. 2–3.]

LoRaWAN ja NB-IoT ovat molemmat varteenotettavia vaihtoehtoja IoT-laitetta suunnitellessa, mutta molemmilla on omat etunsa eri käyttötarkoituksiin. NB-IoT pitää yhdistää paikallisen operaattorin verkkoon, joka käyttää 4G-yhteyttä. Tämän takia NB-IoT-verkkon kantavuus on paljon parempi tiheillä kaupunkialueilla, jossa 4G-yhteyden kuuluvuusalue on parhaimmillaan. NB-IoT:n noin viisi kertaa isompi tiedonsiirtonopeus ja pienempi vasteaika mahdollistaa paremman palvelun laadun.



Kuva 17. LoRaWAN- ja NB-IoT-akun virrankulutusvertailu [28].

LoRaWAN:illa on silti monta etua NB-IoT:n verrattuna. LoRaWAN:in virrankulutus on huomattavasti pienempi ja soveltuu näin paremmin akulla toimiviin IoT-laitteisiin. Kuvassa 17 on harmaalla viivalla kuvattu LoRaWAN:in ja NB-IoT:n virrankulutusta vuosissa lähetettyjen viestien päivittäiseen määrään verrattuna. LoRaWAN kuluttaa huomattavasti enemmän virtaa per lähetetty viesti, kun NB-IoT:n virrankulutus laskee tasaista tahtia suhteessa viestien määrään. Virrankulutukseen vaikuttaa kuuluvuusalueen voimakkuus, mutta vain huonolla kuuluvuusalueella LoRaWAN kuluttaa enemmän virtaa kuin NB-IoT hyvällä tai parhaalla alueella [28].

NB-IoT ei sovellu liikutettaviin sovelluksiin, koska verkon ja yhteystornin vaihtaminen on hidasta ja vaikeaa. LoRaWAN kykenee vaihtamaan yhteystorniaan lennosta, kunhan operaattorin kuuluvuusalue yltää uudelle alueelle. LoRaWAN on myös mahdollista tehdä omaan suljettuun verkkoon, mikä tekee LoRaWAN:ista paljon halvemman vaihtoehdon pieniin paikallisiin sovelluksiin. Myös LoRaWAN:in symmetrisen linkin kaksisuuntainen yhteys helpottaa joustavan tähtitopologian luomisessa [29].

5 LoRa IoT -laitteen toteutus

SeniorSoft-startup-yritys liittyi projektiin asiakkaana heti tutkimustyöosuuden jälkeen. SeniorSoftin projekti tehtiin yrityksen vaatimusten ja pyyntöjen mukaisesti. Työssä tehtiin SeniorSoftille LoRa IoT -laiteprototyyppi, joka kerää tietoa erilaisista antureista ja lähettää ne toiselle LoRa-laitteelle. Laitteen oli tarkoitus toimia sänkyvahdin tavoin. Laite keräisi tietoa paineanturin avulla, onko sängyssä joku ja kauan hän on siellä ollut, sekä huoneen lämpötilaa lämpötila-anturilla.

Projektin aikana luotu prototyyppi koostui neljästä itsenäisestä laitteesta: kolmesta lähettimestä ja yhdestä vastaanottimesta. Lähetyslaitteet kasattiin kehitysalustan ympärille, joka keräsi tietoa antureilta ja lähetti anturien tiedot LoRa-moduulin avulla vastaanottimelle. Vastaanotin koostui kehitysalustasta ja LoRa-moduulista, jotka yhdistettiin Raspberry Pi 3 -tietokoneeseen, vastaanotetun tiedon esittämiseksi.

5.1 Laitteisto

5.1.1 NXP Semiconductors LPCXpresso 1549 -kehitysalusta

Jokaisen laitteen ohjaamiseen käytettiin NXP Semiconductorsin LPCXpresso 1549 -kehitysalustaa. LPC1549-kehitysalustan keskusyksikkönä toimii ARM Cortex-M3 -pohjainen, 72 MHz taajuinen mikrokontrolleri. Kehitysalusta sisältää 256 kB flash-, 32 kB ROM- (Read-Only Memory), 4 kB EEPROM- (Electrically Erasable Programmable Read-Only Memory) ja 36 kB SRAM-muistia (Static Random-Access Memory) [30, s. 1].

LPC1549-kehitysalustassa on 76 GPIO-pinniä (General-Purpose Input/Output), joissa jokaisessa on konfiguroitavat ylös- ja alasvetovastukset, invertteri ja ohjelmoitava digitaalinen virhesuodatin. GPIO-pinnien lisäksi LPC1549-kehitysalustassa on kolme säädettävää USART-liitäntää (Universal Synchronous and Asynchronous Receiver-Transmitter), kaksi SPI-liitäntää (Serial Peripheral Interface) ja yksi säädettävä I^2C-liitäntä (Inter-Integrated Circuit) [30, s. 2-3].

Nämä ominaisuudet mahdollistavat kehitysalustan yhteensopivuuden lähes kaikkien ulkopuolisten anturien kanssa. Kuvassa 6 on projektissa käytetty LPC1549-kehitysalusta.

Kuva 18. NXP LPCXpresso 1549 -kehitysalusta [31]

LPC1549-kehitysalusta valittiin täysin sen helppokäyttöisyyden, saatavuuden ja aikaisemman kokemuksen takia. Kehitysalusta soveltui hyvin myös valmiin tuotteen jatkokehitykseen.

5.1.2 RF Solutions RF-LoRa-868-SO Transceiver -moduuli

LoRa-moduuliksi valittiin RF Solutions RF-LoRa-868-SO Transceiver -moduuli, jonka yksikkönä toimii Semtech SX1272 LoRa -radiomoduuli. SX1272-moduuli on erityisen vähävirtainen pitkänkantaman LoRa-radiolähetin/-vastaanotin, minkä lähetysvirrankulutus on 10 mA ja rekisterin säilytysvirta 100 µA. SX1272 toimii säädettävällä 860-1020 MHz:n taajuudella, kattaen lähes koko maailman taajuussäädökset, Aasiaa lukuun ottamatta. Semtechin patentoidulla LoRa-modulaatiotekniikalla SX1272 pystyy saavuttamaan yli -137 dBm herkkyyden (Sensitivity). Suuri herkkyys yhdistettynä integroituun +20 dBm vahvistimeen tuottaa hyvin suuren hyötysuhteen (157 dB Link Budget) parantaen laitteen optimaalista kantavuutta. [11, s. 1] Moduulissa ei ole sisäänrakennettua antennia, joten työssä käytettiin noin 8,70 cm kokoista rautalankaa antennina, joka oli riittävä toiminnallisuus- ja seinänläpäisytesteihin.

SX1272 kykenee LoRa-modulaation lisäksi lähettämään tietoa erilaisilla taajuusavainnusmenetelmillä (Frequency Shift Keying), mutta niiden toimintaa ei testattu tässä projektissa. Kuvassa 7 on suurennettuna RF-LoRa-868-SO -moduuli (oikeat mitat 20mm x 23mm).

RF-LoRa-868-SO -moduuli liitettiin LPC1549-kehitysalustaan SPI-väylällä. SPI-väylä toimii Master-Slave-periaatteella, ja se koostuu neljästä pinnistä: sarjakellosta (SCLK), master output slave inputista (MOSI), master input slave outputista (MISO) ja Slave-Selectistä (SS tai CS). Näiden neljän pinnin lisäksi SPI-väylässä oleva laite tarvitsee myös jännitteen ja nollan. Master määrää kellotaajuuden SCLK-pinnin kautta ja lähettää tietoa Slavelle MOSI-pinnin kautta. Tiedon lähetettyään Master nostaa SS-pinnin ylös (tai alas joissakin laitteissa) ja odottaa Slavelta vastausta MISO-pinnin kautta. Tässä projektissa LPC1549-kehitysalusta oli Master, ja RF-LoRa-868-SO -moduuli oli Slave. [11, s. 11–12.]

Kuva 19. RF Solutions RF-LoRa-868-SO Transceiver [32]

SX1272-moduuli valittiin sen erityisten ominaisuuksien ja helppokäyttöisyyden vuoksi. SX1272 oli myös yksi halvimmista ja pienimmistä LoRa-moduuleista, joka soveltui erityisen hyvin myös laitteen jatko- ja tuotekehitykseen. kun laite kytketään Digitan LoRaWAN-verkkoon. RF-LoRa-868-SO -moduulissa ei itsessään ole MAC-osoitetta, joten toimiakseen operaattorin verkossa mikrokontrollerille on ohjelmoitava sovelluspohjainen MAC-protokolla. [11, s. 10.] Tähän valittiin valmis IBM LoRaWAN C-kirjasto (LMiC).

5.1.3 TC74-lämpötila-anturi

Antureina laitteessa toimi TC74-digitaalinen lämpötila-anturi ja Defender Security PM3/PK-paineanturimatto. TC74-lämpötila-anturi on pieni ja yksinkertainen TO-220-pakkauksessa oleva digitaalinen anturi, joka soveltuu erityisesti IoT-laitteisiin alhaisen hintansa, helppokäyttöisyytensä ja vähävirtaisuutensa vuoksi. TC74 mittasi laitteen ympäristön lämpötilaa, jonka avulla voidaan varmistaa laitteen oikea käyttölämpötila sekä seurata käyttäjän huoneen lämpötilaa.

TC74-lämpötila-anturi liitettiin LPC1549-kehitysalustaan I^2C-väylällä, joka on yksinkertainen kaksisuuntainen ohjaus- ja tiedonsiirtoväylä. I^2C-väylässä on neljä pinniä: synkronointikello (SCL), tiedonsiirtoväylä (SDA), jännite ja maa. [33, s. 5.] SCL-pinnin avulla varmistetaan, että tieto anturin ja prosessorin välillä lähetettään synkronoidusti. SDA-pinni toimii Master-Slave-periaatteella. Master generoi kellotaajuuden ja aloittaa kommunikaation Slaven kanssa, joka vastaa, jos Master sitä pyytää. I^2C-väylässä olevat laitteet voivat vaihdella näitä rooleja tarvittaessa, mutta tässä projektissa LPC1549-kehitysalusta oli Master, ja TC74-lämpötila-anturi oli Slave.

5.1.4 Defender Security PM3/PK -paineanturimatto

Defender Security PM3/PK-paineanturimatto on sarjansa isoin (720mm x 560mm) matto. Paineanturimattoja käytetään yleensä terveydenhuollossa, vartioinnissa, leluissa ja autojen istuimissa. Paineanturimaton ensisijainen tarkoitus on havaita makaava, istuva tai muuten maton kanssa vuorovaikutuksessa oleva henkilö. Tämä havaitseminen toimii yksinkertaisella kytkimellä, joka aktivoituu, kun maton päälle laitetaan tarpeeksi painetta, joka on noin 25 kg PM3/PK-maton kohdalla.

PM3/PK liitettiin LPC1549-kehitysalustaan GPIO-liitännällä, joka toimi painokytkimen tavoin. PM3/PK-matossa on myös kaksi ylimääräistä johtoa, jotka on kytketty kiinni toisiinsa maton sisällä. Päästämällä jatkuvaa jännitettä näiden johtojen läpi GPIO-liitännällä pystyttiin varmistamaan, että matto toimii eikä ole irti kehitysalustasta. [34, s. 2.] GPIO on yleiskäyttöinen pinni, joka löytyy useimmista mikropiireistä. GPIO-pinnillä ei välttämättä ole suurempaa käyttötarkoitusta, mutta sillä voidaan lähettää yksinkertainen digitaalinen signaali toiselle komponentille. GPIO-pinnit ohjelmoidaan ja ohjataan mikrokontrollerin ohjelmistolla.

GPIO-pinnien jännite on yleensä 3,3 V, mutta se voi olla mitä vain 2 – 5 V:n välissä. Tässä työssä GPIO-pinnit ohjelmoitiin kahteen tarkoitukseen PM3/PK-mattoa varten:

- Ensimmäinen pinni vahti, onko maton painekytkin sulkeutunut, eli onko maton päällä painoa.

- Toinen pinni varmisti suljetun virtapiirin avulla, että matto on kiinni mikrokontrollerissa ja ettei matto ole epäkunnossa.

5.1.5 Raspberry Pi 3 Model B+

Vastaanotetun tiedon tarkistamiseksi vastaanotin liitettiin Raspberry Pi 3 Model B+ -tietokoneeseen UART-liitännällä (Universal Asynchronous Receiver Transmitter). Raspberry Pi on hieman luottokorttia suurempi, yhden piirilevyn tietokone. Raspberry Pi 3 Model B+ sisältää Cortex-A53 1.4 GHz ARM -prosessorin, 1 Gb SDRAM-muistia (Synchronous Dynamic Random-Access Memory) ja microSD-korttipaikan (Secure Digital), joka toimii laitteen massamuistina käyttöjärjestelmää ja ohjelmistoa varten [35].

Vastaanottimen LPC1549-kehitysalusta liitettiin Raspberry Piihin UART-liitännällä. UART-kommunikaatiossa kaksi eri UART-piiriä, lähettävä ja vastaanottava, keskustelee keskenään Tx- (lähetys) ja Rx-pinnien (vastaanotto) avulla. Kommunikointi voidaan tehdä asynkronisesti, eli ajasta riippumattomasti tai ei-reaaliaikaisesti, koska piireissä ei ole kellosignaalia (CLK) synkronoimassa lähteviä bittejä. Tämä on mahdollista vain, koska UART käyttää omaa pinniä lähetykseen ja vastaanottamiseen, toisin kuin I^2C, joka lähettää ja vastaanottaa samasta pinnistä. Lähetyksen synkronointiin UART käyttää piirien yhteistä siirtonopeutta (baud rate), joka on asetettava molemmista laitteista samalle tasolle. Erityisesti antureissa on valmiiksi asetettu siirtonopeus, johon mikroprosessori on asetettava ohjelmoinnin yhteydessä.

Raspberry Pii toimi myös LPC1549-kehitysalustan virranlähteenä. Raspberry Pi sai virtansa USB-muuntajalla verkkovirrasta, kun LPC1549 sai virtansa Raspberry Piin 5 V GPIO-pinnistä.

5.2 Ohjelmisto

5.2.1 NXP MCUXpresso -ohjelmointiympäristö

Prototyypin LPC1549-kehitysalusta ohjelmointiin NXP MCUXpresso -ohjelmointiympäristössä C/C++-ohjelmointikielellä. MCUXpresso IDE -ohjelmointiympäristö on NXP Semiconductorsin Eclipse-pohjainen kehitysympäristö, joka on suunniteltu erityisesti NXP Semiconductorsin ARM Cortex-M -kehitysalustoille. MCUXpresso-ohjelmointiympäristö tarjoaa monipuoliset muokkaus-, kokoamis- ja virheenkorjausominaisuudet sekä monipuoliset työkalut koodinseurantaan [36, s. 8]. MCUXpresso IDE valittiin ohjelmointiympäristöksi, koska se tukee LPC1549-kehitysalustaa, se on ilmainen pienissä projekteissa ja tarjoaa monipuoliset virheenkorjausominaisuudet.

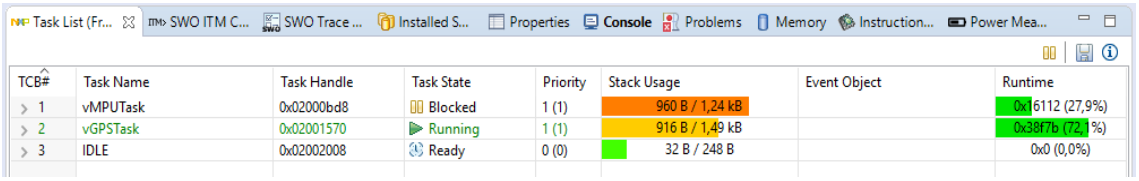
5.2.2 FreeRTOS Kernel

FreeRTOS Kernel (Real-Time Operating System) on Real Time Engineersin kehittämä reaaliaikainen käyttöjärjestelmäydin (kernel) mikrokontrollereille. FreeRTOS on suunniteltu tarpeeksi pieneksi, jotta se toimisi pienitehoisissa mikroprosessoreissa – vaikka sitä ei ole rajoitettu vain mikroprosessorisovelluksiin. FreeRTOS tarjoaa siis vain prosessori ytimen reaaliaikaisen aikataulutustoiminnon, tehtävien välisen viestinnän, ajoituksen ja perusalkioiden synkronoinnin. Lisätoiminnot, kuten komentokonsolikäyttöliittymä tai verkkopinot sisällytetään sitten lisäkomponentteihin.

Reaaliaikainen sovellus, joka käyttää FreeRTOS-järjestelmää, voidaan jäsentää itsenäisten tehtävien (task) joukkoihin. Jokainen tehtävä suoritetaan omassa kontekstissaan ilman satunnaista riippuvuutta muista järjestelmän sisällä olevista tehtävistä tai itse RTOS-aikataulusta. Vain yksi sovelluksen sisällä oleva tehtävä voidaan suorittaa milloin tahansa, ja RTOS-aikataulu on vastuussa siitä, minkä tehtävän tämän pitäisi olla. RTOS-aikataulu voi siksi toistuvasti käynnistää ja lopettaa jokaisen tehtävän (vaihtaa tehtävien välillä) sovelluksen suoritettuaan.

Koska tehtävällä ei ole tietoa RTOS-aikataulun toiminnasta, RTOS-aikataulun vastuulla on varmistaa, että prosessorin tila (rekisteriarvot, pinon sisältö jne.) tehtävän vaihtamisen yhteydessä on täsmälleen sama kuin silloin, kun sama tehtävä vaihdettiin. Tämän saavuttamiseksi jokaisella tehtävällä on oma pino. Kun tehtävä vaihdetaan pois, suoritus tallennetaan tehtävän pinoon, jotta se voidaan myös palauttaa tarkalleen samaan tilaan, kun tehtävä vaihdetaan myöhemmin takaisin [37].

Työssä hyödynnettiin myös FreeRTOS:in jonoja (queue), jotka ovat pääasiallinen muoto tehtävienväliseen viestintään. Niitä voidaan käyttää viestien lähettämiseen tehtävien välillä, ja keskeytyksien ja tehtävien välillä. Useimmissa tapauksissa niitä käytetään Thread Safe FIFO -puskureina (First In First Out), kun uutta tietoa lähetetään jonon etu tai takaosaan. Viestit lähetetään jonon läpi kopioimalla, eli tieto (joka voi olla osoitin suuremmille puskureille) itse kopioidaan jonoon sen sijaan, että jono aina tallentaisi vain viittauksen tiedon paikasta [38].



Kuva 20. FreeRTOS Task List MCUXpresso -ohjelmointiympäristön konsoli-ikkunassa.

FreeRTOS mahdollistaa myös tehtäväkohtaisen virheenkorjauksen. FreeRTOS tarjoaa oman seuranta mekanisminsa (Task List), millä kyetään seuraamaan jokaista tehtävää yksilöllisesti ja paikantamaan mahdolliset virheet tehtäväkohtaisesti. Tämä tehtävä lista sisältää muun muassa tehtävien nimet, sijainnin muistissa, tilan, prioriteetit, muistipinon käytön ja tehtävän kuluttaman kokonaisajan (kuva 20).

5.2.3 Linux Raspbian

Raspbian on Raspberry Piille optimoitu ilmainen Debian-pohjainen Linux-käyttöjärjestelmä. Raspbian-käyttöjärjestelmä sisältää perusohjelmat ja apuohjelmat Raspberry Piin nopeaan käyttöönottoon. Se sisältää yli 35 000 pakettia, eli valmiiksi käännettyä ohjelmistoa, joihin kuuluvat esimerkiksi työssä käytetyt UART- ja Python-kirjastot. Tämä helpotti Raspberry Piin käyttöjärjestelmän valintaa, sillä Raspberry Piillä ei ollut isoa vaikutusta työn teossa.

6 Ohjelman toimintaperiaate

Työ ohjelmointiin C/C++-ohjelmointikielellä käyttäen olio-ohjelmointiparadigmaa ja RTOS-periaatetta. Ohjelma käynnistyy heti, kun virta kytketään päälle, eikä vaadi käyttäjältä mitään muuta toimenpidettä. Ohjelma alustaa itsensä ja aloittaa pakettien lähetyksen/vastaanottamisen automaattisesti. FreeRTOS Task -ominaisuudella ohjelma jaettiin tehtäviin, joista jokainen tehtävä vastasi erillisestä anturista ja LoRa-moduulista. Tämän kappaleen apuna voi käyttää liitteessä 1 olevia vuokaavioita lähettimen ja vastaanottimen toiminnasta.

6.1 Lähettimet

Työtä varten tehtiin kolme lähetintä. Lähettimien ohjelmat olivat identtiset radiokanavaa ja lähettimen numeroa lukuun ottamatta, jotka vaihdettiin käsin ohjelman latauksen yhteydessä.

Lähettimien ohjelmakoodi sisälsi pääohjelman (main) ja kolme luokkaa (class), joista jokaisella oli koodi- (.cpp -laajennus) ja otsikkotiedostot (header, .h -laajennus). Otsikkotiedoston ensisijainen tarkoitus on johtaa luokka- ja muuttuja-alustukset kooditiedostoihin.

```
int main(void) {
	prvSetupHardware();
	xBin = xSemaphoreCreateBinary();
	xQueue = xQueueCreate(5, 64);

	xTaskCreate(vTransmitterTask, "vTransmitterTask",
		configMINIMAL_STACK_SIZE + 256, NULL,
		(tskIDLE_PRIORITY + 1UL), (TaskHandle_t *) NULL);

	xTaskCreate(vSensorTask, "vSensorTask",
		configMINIMAL_STACK_SIZE + 256, NULL,
		(tskIDLE_PRIORITY + 1UL), (TaskHandle_t *) NULL);

	xTaskCreate(vButtonTask, "vButtonTask",
		configMINIMAL_STACK_SIZE + 128, NULL,
		(tskIDLE_PRIORITY + 1UL), (TaskHandle_t *) NULL);

	vTaskStartScheduler();

	return 0;
}
```

Esimerkkikoodi 1. Lähettimen main-ohjelmakoodin main-metodi.

Lähettimen main-ohjelmakoodi sisälsi kehitysalustan alustukseen vaadittavat asetukset, RTOS-tehtävien ja -jonojen alustuksen sekä tehtävämanagerin käynnistyksen. Esimerkkikoodin 1 main-metodista ja liitteen 1 vuokaaviosta nähdään, kuinka lähettimen ohjelma alustettiin käynnistyksen yhteydessä. Main-metodi ajettiin automaattisesti heti käynnistyksen yhteydessä, eikä vaatinut käyttäjältä minkäänlaista toimenpidettä. Main-metodin aikana luotiin kolme RTOS-tehtävää: Button Task, Sensor Task ja Transmitter Task, ja lopuksi aloitettiin tehtävien aikataulutuksesta huolehtiva Task Scheduler, joka saattaa main-metodin ikuiseen silmukkaan. Esimerkkikoodissa 1 näkyvä ’return 0;’ ei tule koskaan toteen, koska tehtävien teko ei lopu koskaan. Tämä vaadittiin vain metodin tyypin (integer) takia.

Button Task alustettiin pienemmällä muistilla kuin kaksi muuta tehtävää, koska sen sisältämät DigitalIo-luokat eivät tarvinneet paljoa muistia. DigitalIoPin.cpp-kooditiedosto loi hyvin yksinkertaisen GPIO-pinnien ohjausobjektin. Objektin tarkoitus oli kirjoittaa ja/tai lukea objektin alustuksen yhteydessä annettujen GPIO-pinnien tilaa. Objektilla oli yksi pääkäyttöinen metodi: read. Button Task meni objektien alustamisen jälkeen ikuiseen while-silmukkaan (while-silmukan ehto oli 1), jonka aikana se tarkisti (read), onko kehitysalustassa sijaitsevaa hätäpainiketta painettu ja onko paineanturimaton päällä painoa. Jos hätäpainiketta painettiin, tehtävä täytti tehtävien välisen jonon etuosan ’H’-kirjaimella hädän merkiksi. Jos painiketta ei painettu, tehtävä lähetti jonon takaosaan tiedon (true tai false), onko paineanturimaton päällä painoa.

Sensor Task alusti TempI2C-luokan lämpötila-anturin lukemiseen. TempI2C.cpp kooditiedosto alusti kehitysalustan I²C-liitännät lämpötila-anturin vaatimien asetuksien mukaisesti ja testasi anturin toiminnan. TempI2C-luokkalla oli vain yksi metodi: readTemp, joka kävi kysymässä anturilta lämpötilan ja palautti sen ohjelmalle. Kuten ButtonTask, SensorTask meni objektin alustamisen jälkeen ikuiseen while-silmukkaan. Jokaisen silmukan kierroksen aikana tehtävä luki sen hetken lämpötilan ja lähetti sen heti jonon taakse.

Transmitter Task oli vastuussa LoRa-moduulilla lähettämisestä LoRaSPI-luokan avulla. LoRaSPI.cpp-kooditiedostoa käytettiin molemmissa, lähettimen ja vastaanottimen ohjelmissa, koska ohjauksen lisäksi se piti huolen LoRa-moduulin alustamisesta SPI-liitännän kautta. LoRaSPI.cpp-kooditiedosto sisälsi useita medoteja, joista kolmea käytettiin main-kooditiedostossa ja muita luokan sisäisesti LoRa-moduulin alustamisessa.

```
LoRaSPI::LoRaSPI(transreceiver mode) {
      this->mode = mode;
      Init_SPI_PinMux();
      setupSpiMaster();
      configureModule();
}
```

Esimerkkikoodi 2. LoRaSPI-luokan objektin alustusmetodi.

LoRaSPI-objektia luodessa annettiin parametriksi vain moduulin tila (mode), jonka avulla määriteltiin, asetetaanko moduuli lähettävään vai vastaanottavaan tilaan. Tilan tallentamisen jälkeen alustettiin SPI-liitännän pinnit, kehitysalustan SPI asetukset kahdella sisäisellä metodilla: Init_SPI_PinMux() ja setupSpiMaster(). Objektin viimeinen sisäinen metodi, configureModule(), käynnisti ja valmisteli LoRa-moduulin lähetys- tai vastaanottotilaan riippuen annetusta mode-parametristä (esimerkkikoodi 2).

LoRaSPI-objektin luomisen jälkeen Transmitter Task antoi moduulille halutun kanavan. Tämä asetus lisättiin main-ohjelmakoodiin muokattavuuden helpottamiseksi. Kaikkien lähettimien kanava ja lähetinnumero määriteltiin (define) ohjelman alussa, mistä niitä oli helppo muokata avaamalla vain yhden tiedoston koko ohjelmasta.

Ennen kuin tehtävä meni ikuiseen while-silmukkaan, tehtävä loi tarvittavat puskurit tietojen lähettämistä varten, kertoi puskurin koon moduulille ja kuittasi writereg-metodilla moduulin flagit, eli liput, joiden avulla vahdittiin moduulin toimintaa ja virheitä. Ikuisen while-silmukan sisällä tehtävä keräsi jonosta lämpötila-anturin ja painikkeiden tiedot, ja kirjoitti ne LoRa-moduulin FIFO-puskuriin. Lopuksi FIFO-puskurin tiedot lähetettiin vastaanottimelle, odotettiin flagia onnistuneesta lähetyksestä, ja tietojen kerääminen aloitettiin alusta.

Nämä kolme tehtävää ajettiin tehtävä managerin avulla asynkronisesti vierekkäin, eikä millään tehtävällä ollut mitään tietoa toisen toiminnasta. Tehtävien välinen jono toimi tiedon siirtämisenä tehtävien välillä, mutta jos jonossa ei ollut mitään, tehtävät vain odottivat seuraavaa tietoa toisilta tehtäviltä.

6.2 Vastaanotin

Työssä oli yksi vastaanotin, joka toimi keskittimenä (concentrator) lähettimien välillä. Kaikkien lähettimien lähettämä tieto luettiin yksi kerrallaan, koska SX1272-moduuli ei kyennyt keskustelemaan usean lähettimen kanssa samanaikaisesti. Vastaanottimen ohjelmakoodin toimintaperiaate on sama kuin lähettimillä, mutta sisälsi vain kaksi tehtävää: Receiver Task ja Raspi Task (lyhenne Raspberry Piistä).

```
int main(void) {
	prvSetupHardware();
	xBin = xSemaphoreCreateBinary();
	xQueue = xQueueCreate(5, 64);

	xTaskCreate(vReceiverTask, "vReceiverTask",
		configMINIMAL_STACK_SIZE + 512, NULL,
		(tskIDLE_PRIORITY + 1UL), (TaskHandle_t *) NULL);

	xTaskCreate(vRaspiTask, "vRaspiTask",
		configMINIMAL_STACK_SIZE + 512, NULL,
		(tskIDLE_PRIORITY + 1UL), (TaskHandle_t *) NULL);

	vTaskStartScheduler();

	return 0;
}
```

Esimerkkikoodi 3. Vastaanottimen main-ohjelmakoodin main-metodi.

Esimerkkikoodin 3 main-metodista ja liitteen 1 vuokaaviosta nähdään, kuinka vastaanottimen ohjelma alustettiin käynnistyksen yhteydessä. Koska vastaanottimessa oli vain kaksi tehtävää, niille annettiin enemmän muistia käytettäväksi. Muistia jaettiin myös enemmän tehtävien laajuuden takia.

Receiver Task toimi samalla periaatteella kuin lähettimen Transmitter Task, mutta vastaanottotilassa. Receiver Task loi LoRaSPI-objektin lähettimen parametrilla, alusti puskurin vastaanotettua tietoa varten, kuittasi kaikki flagit ja asetti LoRa-moduulin jatkuvaan vastaanottotilaan (Continuous Receiver, RxCont) ennen ikuiseen while-silmukkaan menoa. Ikuisessa while-silmukassa Receiver Task odotti 3 sekuntia lähetystä, vastaanotti FIFO-puskurin ja purki lämpötila-anturin, lähetinnumeron ja painikkeiden tiedot omaan puskuriinsa. Puskuri laitettiin sitten jonoon, mistä Raspi Task sai kaiken tiedon. Lopuksi tehtävä vaihtoi vastaanotettavan tiedon kanavaa ja odotti seuraavan lähettimen lähetystä. Jos lähetystä odottaessa meni yli 3 sekuntia, vastaanotin ohitti tämän kaiken ja vaihtoi suoraan kanavaa.

Raspi Task käytti RaspiUart.cpp-kooditiedostoa RaspiUart-objektin luomiseksi. RaspiUart.cpp-kooditiedosto alusti kehitysalustan muokattavat USART-pinnit UART-kommunikaatiota varten ja käynnisti asynkronisen kommunikaation Raspberry Piin kanssa. UART-kommunikaatiota ohjattiin read- ja write-metodeilla, mutta tässä työssä käytettiin vain write-metodia, koska Raspberry Piille vain lähetettiin tietoa. Raspi Taskilla oli useampi sisäinen puskuri, mikä valittiin vastaanotetun tiedon perusteella. Objektin ja puskurien alustusten jälkeen tehtävä meni ikuiseen while-silmukkaan, missä se ensimmäisenä odotti jonoon ilmestyvää tietoa.

Tämä tieto tallennettiin omaan puskuriin, mistä sitä voitiin tarkastella tarkemmin. Jos vastaanotetun tiedon lämpötila oli numeron sijasta ’H’-kirjain, lähetettiin UART-liitännän kautta teksti Raspberry Piille, missä luki ”HELP transmitter #”, missä # oli lähettimen numero. Tämä tapahtui 2-3 kertaa, jos jostain lähettimestä painettiin hätäpainiketta. Jos lämpötila oli numero, tehtävä laski lämpötila-anturin mukana tulleen algoritmin avulla lämpötilan saadusta hex-luvusta. Laskettu lämpötila, lähettimen numero ja painematto-anturin arvo tallennettiin erilliseen puskuriin ja lähetettiin Raspberry Piille.

Raspberry Piille tehtiin Python-ohjelmointikielellä hyvin yksinkertainen UART-kommunikaatiota vahtiva ohjelma. Ohjelma käynnisti Raspberry Piin UART-liitännän ja alkoi odottamaan vastaanotettua tietoa ikuisessa while-silmukassa. Vastaanotettu tieto tulostettiin konsoli-ikkunaan, josta sitä voitiin tarkastella ja todeta LoRa-moduulien toimivuus.

7 Prototyypin jatkokehitys

Projektista valmistuneen prototyypin tarkoitus oli tutkimustyönä tehdyn LoRa-tekniikan ymmärtäminen. Prototyyppiä ei kytketty operaattorin verkkoon, eikä se sisältänyt verkkoprotokollaa tai salausavaimia. Kolme lähetintä lähetti puhtaalla LoRa-taajuudella viestiä, joka oli helposti kaapattavissa ja kaikkien näkyvissä. Vastaanotin ei tarkistanut vastaanotetun viestin salausta ja kuka vaan olisi voinut lähettää LoRa-taajuudella viestin, jonka vastaanotin olisi napannut ja mahdollisesti kirjannut ylös. Tämän takia prototyypistä tehtiin LoRaWAN-laiteversio, joka yhdistyisi Digitan IoT-verkkoon. Tätä varten Digitalta kysyttiin yhteistyötä, joka mielellään antoi parin ABP- ja OTAA-avaimia Metropolialle opetuskäyttöön.

LoRaWAN-laiteeseen valittiin valmis ohjelmakoodi GitHub-sivustolta. GitHub-käyttäjä nimeltään matthijskooijman [39] oli tehnyt valmiin SX1272-moduulille suunnatun ohjelmakoodin, joka käytti Arduino Uno -kehitysalustaa. Laite kasattiin Matthijskooijmanin arduino-lmic -ohjeiden mukaisesti ja rautalangasta tehty antenni vaihdettiin oikeaan 868 MHz:n antenniin. Ohjelmakoodin verkkoprotokollana toimi IBM LMiC -kirjasto, jonka avulla yhteys muodostettiin Digitan verkkoon. Ohjelmakoodista muutettiin NwkSKey, AppSKey (pakolliset avaimet ABP-aktivointiin), DevAddr (laitteen MAC-osoite) ja lähetettävä viesti, jotta onnistunut lähetys pystyttiin varmistamaan Digitan käyttöliittymän kautta. Lähetettävä viesti muutettiin sekunnin välein kasvavaksi numeroksi 0 – 99 väliltä, joka helpotti viiveen tunnistamisessa. LoRaWAN-laitteeseen ei kyketty lämpötila-anturia, eikä painiketietoja paineanturimatosta tai hätäpainikkeesta.

Arduino Uno -kehitysalustalla luotu LoRaWAN-laite saatiin toimimaan ongelmitta. Digitan verkon testaamiseksi laite käynnistettiin mahdollisimman useassa paikassa, yhteyden toiminnan toteamiseksi. Laite kasattiin ja testattiin ensimmäisenä Metropolian kampuksella Espoon Leppävaarassa, missä sitä kokeiltiin niin ulkona, sisällä ja rakennuksen kellarissa. Laite sai yhteyden Digitan verkkoon, jopa rakennuksen kellarista, mutta useita lähetettyjä viestejä katosi matkalla.

Kuva 21. Arduino Uno -kehitysalustalla testatun ohjelman Digita IoT -verkon testialueet.

Leppävaaran lisäksi laitetta testattiin ulkona Helsingin Pitäjänmäellä ja Espoon Kivenlahdessa ongelmitta. Laite vietiin myöhemmin Metropolian kampukselle Vantaan Myyrmäkeen, missä sen toiminta testattiin uudelleen sekä sisällä että ulkona, onnistuneesti (kuva 21). Kyseinen LoRaWAN-laite luovutettiin sitten jatkokehitykseen verkkotekniikan opiskelijoille.

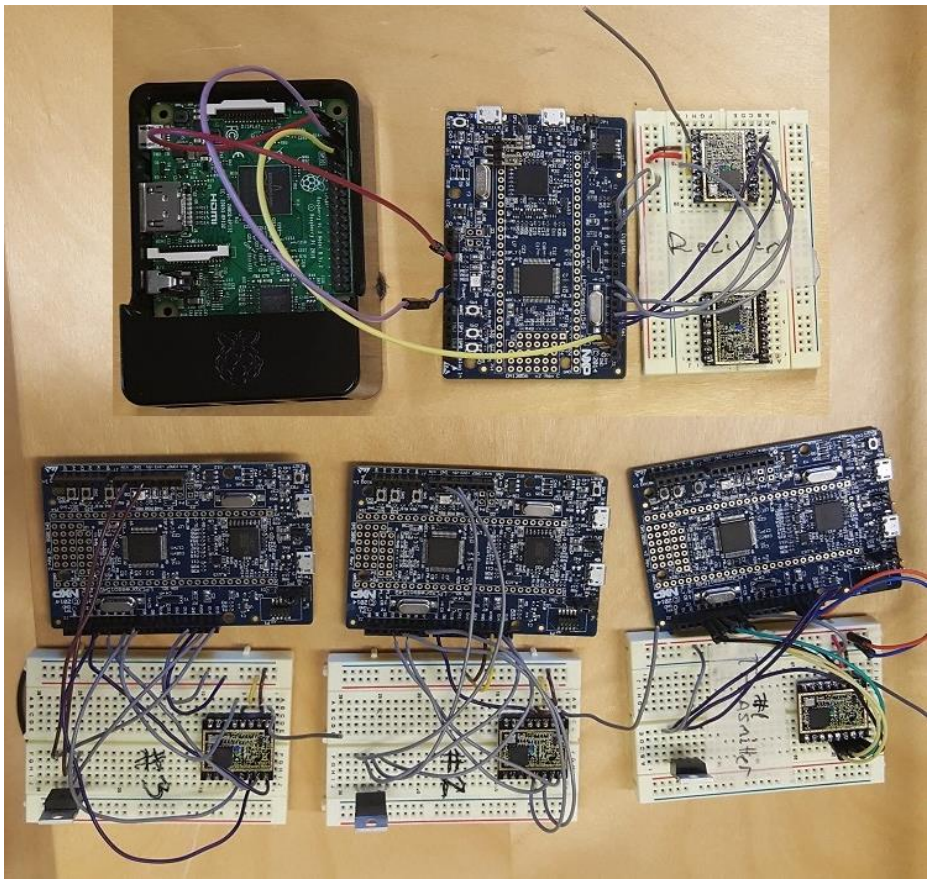
8 Yhteenveto

Opinnäytetyön tavoitteena oli suunnitella ja ohjelmoida SeniorSoft-yritykselle sänkyvahti-prototyyppi osana IoT-talotekniikkaa, joka lähettäisi tietoa turvallisesti LoRaWAN-teknologian avulla. Alkuperäisen prototyypin yhdistäminen Digitan LoRaWAN-verkkoon jäi tekemättä ajanpuutteen ja verkkoprotokollan puuttumisen vuoksi, mutta jatkokehityksen yhteydessä valmiin Arduino-ohjelmakoodin avulla yhdistäminen IoT-verkkoon onnistui.

Tutkimustyö osuus vei suurimman osan projektiin käytetystä ajasta, koska LoRaWAN-teknologia ei ollut ennestään tuttu. Tutkimustyön aikana tutkitut kilpailevat langattoman tiedonsiirtotekniikat auttoivat ymmärtämään tekniikoiden käyttötarkoituksia ja -kohteita, sekä erottamaan taajuusmodulaatiotekniikat toisistaan. Opinnäytetyössä luotu prototyyppi antoi hyvän käsityksen LoRa-tekniikan eduista ja rajoitteista, jotka otetaan huomioon lopputuotteen jatkokehitystä varten.

IoT-laiteiden kehittyessä, tämä opinnäytetyö antoi hyvät valmiudet tulevaisuuden IoT-laiteiden kehitykseen ja langattoman tiedonsiirtotekniikoiden valitsemiseen. Prototyypin vaatimukset saavutettiin ja tuloksena saatiin toimiva paikallinen LoRa-verkko sekä LoRaWAN-verkkoon yhdistetty LoRaWAN-laite.

Huolimatta siitä, että työstä olisi saatu vain yksi LoRaWAN-laite, joka ajaisi molempien testilaiteiden tarkoituksen, olen tyytyväinen työn tulokseen. Molemmat laitteet yhdessä saavuttivat työn alkuperäiset vaatimukset. Laitteet ja ohjelmakoodit siirtyivät jatkokehitykseen seuraaville Metropolian opiskelijoille.



Kuva 22. Insinöörityön aikana tehty LoRa-prototyyppi. Kuvassa yllä Raspberry Piihin kytketty vastaanotin ja alla kolme lähetintä. Kuvassa ei ole kytketty paineanturimattoja.

Lähteet

1 LoRa Alliance, Inc.. 2015. LoRaWAN™ What it is? A technical overview of LoRa® and LoRaWAN™. E-kirja. LoRa Alliance.

2 Abbas, Awais & Ul Haq. 2018. Comparative Analysis of Wideband Communication Techniques: Chirp Spread Spectrum and Direct Sequence Spread Spectrum. Pakistan: iCoMET.

3 Jacky8162002. Senyal amb modulació de freqüència Chirp : domini temporal i spectral. Verkkoaineisto. Wikipedia. <https://commons.wikimedia.org/w/index.php?curid=37957561> Luettu 24.10.2019.

4 Berni & Gregg. 1973. On the Utility of Chirp Modulation for Digital Signaling. IEEE Transactions on Communications. Vol. 21, s. 748–751.

5 Christian Wolff. Frequency-Modulated Continuous-Wave Radar (FMCW Radar). Verkkoaineisto. Radartutorial.eu. <https://www.radartutorial.eu/02.basics/Frequency%20Modulated%20Continuous%20Wave%20Radar.en.html>. Luettu 24.10.2019.

6 Binder, Abramson, Kiio, Okinaka & Wax. 1975. ALOHA Packet Broadcasting – A Retrospect. Honolulu: University of Hawaii.

7 Tanenbaum, Andrew & Wetherall, David. 2011. Computer Networks, uudistettu painos. Boston: Pearson Education, Inc.

8 Difference Between Pure ALOHA and Slotted ALOHA. 2016. Verkkoaineisto. Tech Differences. <https://techdifferences.com/difference-between-pure-aloha-and-slotted-aloha.html>. Luettu 24.10.2019.

9 Polonelli, Tommaso; Brunelli, Davide & Benini, Luca. 2018. Slotted ALOHA Overlay on LoRaWAN – a Distributed Synchronization Approach. E-kirja. IEEE.

10 Digitan IoT LoRaWAN-verkon peittokartta. Verkkoaineisto. Digita. <https://www.digita.fi/yrityksille/iot/iot_lorawan-verkon_peittokartta> Luettu 24.10.2019.

11 Semtech Corporation. 2019. SX1272/73 - 860 MHz to 1020 MHz Low Power Long Range Transceiver Rev. 4. E-kirja. Semtech.

12 What is the LoRaWAN® Specification. Verkkoaineisto. LoRa Alliance. <https://www.lora-alliance.org/about-lorawan>. Luettu 24.10.2019.

13 Gemalto, Actility & Semtech. 2017. LoRaWAN™ Security Full End-to-End Encryption for IoT Application Providers. E-kirja. LoRa Alliance.

14 LoRa Alliance Technical Committee Regional Parameters Workgroup. 2018. LoRaWAN™ 1.1 Regional Parameters Revision B. E-kirja. LoRa Alliance.

15 Commission Implementing Decision (EU) 2017/1483 – Amending Decision 2006/771/EC on harmonisation of the radio spectrum for use by short-range devices and repealing Decision 2006/804/EC. 2017. Verkkoaineisto. European Commission. <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32017D1483>. Luettu 28.10.2019.

16 Connected Finland kotisivu. Verkkoaineisto. Connected Finland. <https://www.connectedfinland.fi/>. Luettu 24.10.2019.

17 What is DASH7?. Verkkoaineisto. WizziLab. <http://wizzilab.com/dash7-technology> Luettu 24.10.2019.

18 Sigfox White Paper Security. 2017. Verkkoaineisto. Sigfox Connect. <https://www.sigfox.com/sites/default/files/1701-SIGFOX-White_Paper_Security.pdf>. Luettu 24.10.2019.

19 Zigbee Specification FAQ. 2013. Verkkoaineisto. Zigbee Alliance. <https://web.archive.org/web/20130627172453/http://www.zigbee.org/SpecificatiSpe/ZigBee/FAQ.aspx>. Luettu 24.10.2019

20 Egan, David. Zigbee Propagation for Smart Metering Networks, Issue 12, Vol 17. Verkkoaineisto. Silicon Laboratories. <https://www.power-grid.com/2012/12/01/zigbee-propagation-for-smart-metering/>. Luettu 24.10.2019.

21 Gislason, Drew. 2007. Zigbee Wireless Networking. Boston: Newnes

22 Sweeney. 2002. An Introduction to Bluetooth – A Standard for Short Range Wireless Networking. 15th Annual IEEE International ASIC/SOC Conference. Vol. 21, s. 748–751.

23 Ktims. An example of binary FSK. Verkkoainesto. Wikipedia. <https://commons.wikimedia.org/w/index.php?curid=635074>. Luettu 24.10.2019.

24 LTE. Verkkoaineisto. 3GPP. <https://www.3gpp.org/technologies/keywords-acronyms/98-lte>. Luettu 24.10.2019.

25 What is OFDM: Orthogonal Frequency Division Multiplexing. Verkkoaineisto. Electronics Notes. <https://www.electronics-notes.com/articles/radio/multicarrier-modulation/ofdm-orthogonal-frequency-division-multiplexing-what-is-tutorial-basics.php>. Luettu 24.10.2019.

26 Campalans & Pérez-Neira. 2009. Cross-Layer Resource Allocation in Wireless Communications – Techniques and Models from PHY and MAC Layer Interaction, s. 151-162. Academic Press.

27 Hwang, Yitaek. Cellular IoT Explained – NB-IoT vs. LTE-M vs. 5G and More. Verkkoaineisto. IoT for all. <https://www.iotforall.com/cellular-iot-explained-nb-iot-vs-lte-m/>. Luettu 24.10.2019.

28 Aufranc, Jean-Luc. 2018. A Look at LoRaWAN and NB-IoT Power Consumption. Verkkoaineisto. CNX Software. <https://www.cnx-software.com/2018/03/29/a-look-at-lorawan-and-nb-iot-power-consumption/>. Luettu 24.10.2019.

29 Ray, Brian. 2018. NB-IoT vs. LoRa vs. Sigfox. Verkkoaineisto. Link Labs. <https://www.link-labs.com/blog/nb-iot-vs-lora-vs-sigfox>. Luettu 24.10.2019.

30 NXP. 2015. LPC15xx Product datasheet revision 1.1. E-kirja. NXP Semiconductors.

31 NXP. OM13056: LPCXpresso™ Board for LPC1549. Verkkoaineisto. NXP Semiconductors. <https://www.nxp.com/design/microcontrollers-developer-resources/lpcxpresso-boards/lpcxpresso-board-for-lpc1549:OM13056>. Luettu 24.10.2019.

32 RF Solutions. RF-LoRa-868-SO Transceiver Module. Verkkoaineisto. <https://www.rfsolutions.co.uk/radio-modules-c10/frequency-c57/fm-lora-transceiver-module-pre-set-to-868mhz-p468> Luettu 24.10.2019.

33 Microchip Technology Inc. 2012. TC74 – Tiny Serial Digital Thermal Sensor. E-kirja. Microchip Technology Inc.

34 Pressure Mat Sensors. 2015. Verkkoaineisto. Defender Security. <http://www.farnell.com/datasheets/1895123.pdf> Luettu 24.10.2019.

35 Raspberry Pi 3 Model B+. Verkkoaineisto. Raspberry Pi Foundation. <https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/>. Luettu 24.10.2019

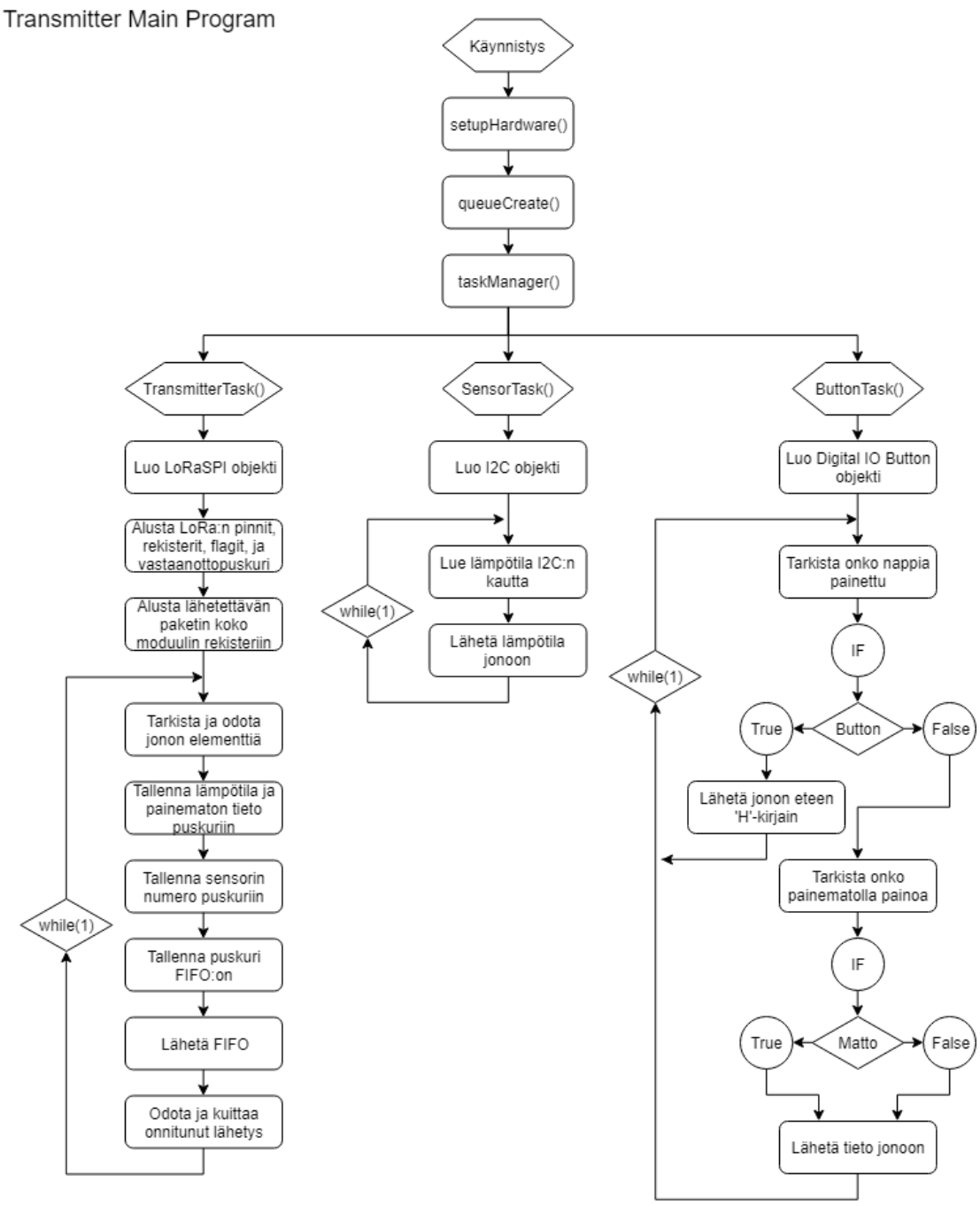
36 NXP. 2017. MCUXpresso IDE User Guide revision 10.1.0. E-kirja. NXP Semiconductors.

37 Tasks and Co-routines. Verkkoaineisto. FreeRTOS. <https://www.freertos.org/taskandcr.html>. Luettu 24.10.2019.

38 Queues, Mutexes, Semaphores… Verkkoaineisto. FreeRTOS. <https://www.freertos.org/Embedded-RTOS-Queues.html>. Luettu 24.10.2019.

39 Kooijman, Matthijs. Verkkoaineisto. GitHub. <https://github.com/matthijskooijman>. Luettu 24.10.2019.

Lähettimen ja vastaanottimen vuokaaviot



Receiver Main Program