Tartu Ülikool

Loodus- ja täppisteaduste valdkond

Tehnoloogiainstituut

Jüri Gramann

Samaaegses droonisegamises ja -tuvastuses enesehäirete eemaldamiseks kasutatava analoogmooduli juhtimine

Magistritöö (30 EAP)
Arvutitehnika ja robootika eriala

Juhendajad:

MSc Jaanus Kalde
MSc Karel Pärlin

Tartu 2020

Resümee/Abstract

Samaaegses droonisegamises ja -tuvastuses enesehäirete eemaldamiseks kasutatava analoogmooduli juhtimine

Droonid on kiirelt arenev ja kasvav turg. Kuigi nende kasutuselevõtt on toonud kaasa palju võimalusi, on antud tehnoloogiaareng tekitanud ka teatud probleeme. Droone võib kasutada pahatahtlikult ja kuritegelikel eemärkidel. Selleks, et keelatud tegevusi droonidega piirata, on vaja neid tuvastada ja tõrjuda. Droone on võimalik tuvastada monitoorides nende raadiosidet. Sama raadiosidet on võimalik ka segada ja sellega neid tõrjuda. Selleks, et mõlemat samaaegselt teha, on vaja rakendada ühes sagedusvahemikus töötav täisdupleks (*in-band full-duplex* IBFD) raadiot. Täisdupleks raadio rakendamiseks on vaja lahendada sellega kaasnevaid probleeme. Antud töös tutvustatakse täisdupleks raadio põhimõtet, selle puudused ja kuidas neid lahendada. Samuti antakse ülevaade täisdupleksit võimaldavast analoogmoodulist ning luuakse selle juhtsüsteem. Seejärel katsetatakse kogu süsteemi efektiivsust erinevates situatsioonides. Viimaks näidatakse, et tänu arendatud süsteemile, on võimalik droone samaaegselt segada ja tuvastada.

CERCS: T121 Signaalitöötlus; T170 Elektroonika; T191 Kõrgsagedustehnoloogia, mikrolained;

Märksõnad: droonid, täisdupleks, FPGA

Control of an Analog Self-Interference Canceler for Simultanious Detection and Distruption of Drones

The market for drones is growing fast and with that comes problems. They are being used in malicious intent and for criminal activity. To combat that, you need to detect and distrupt them. For detection you can analyze radio frequencies and to distrupt you can jam their signals. But you can not do them at the same time. One way to do them at the same time is to implement an in-band full-duplex (IBFD) radio system. In this work we explain what an IBFD radio is and what are its problems. The main focus is to develop a control system for an analog self-interference canceler. The whole system is tested in various setups. Finally the system is used to jam and detect drones at the same time.

CERCS: T121 Signal processing; T170 Electronics; T191 High frequency technology, microwaves;

Keywords: drones, full-duplex, FPGA

Sisukord

Joonised

Tabelid

Programmide loetelu

Lühendid, konstandid, mõisted

ADC - *Analog-to-Digital Converter* - Analoog-digitaalmuundur

DAC - *Digital-to-Analog Converter* - Digitaal-analoogmuundur

DDR - *Double Data Rate* - Topeltkiirusega

FMC - *FPGA Mezzanine Card* - FPGA laienduskaart

FPGA - *Field-Programmable Gate Array* - Programmeeritav ventiilmaatriks

HLS - *High-Level Synthesis* - Kõrgetasemeline süntees

I2C - *Inter-Integrated Circuit* - Kahejuhtmeliides

IBFD - *In-Band Full-Duplex* - Ühes sagedusvahemikus töötav täisdupleks

ISM - *Industrial, Scientific and Medical* - Tööstus, teadus ja meditsiin

LO - *Local Oscillator* - Kohalik ostsillaator

LVDS - *Low Voltage Differential Signaling* - Madalpingeline diferentsiaal-signaaliedastus

MSB - *Most Significant Bit* - Suurima kaaluga bitt

MSPS - *Mega Samples Per Second* - Miljon proovi sekundis

RX - *Receive* - Vastu võtma

TX - *Transmit* - Saatma

UART - *Universal Asynchronous Receiver/Transmitter* - Universaalne asünkroontransiiver

VHDL - *Very High Speed Integrated Circuit Hardware Description Language* - Väga kiirete integraallülituste riistvara kirjeldamise keel

WiFi - *Wireless Fidelity* - Raadiokohtvõrk

1 Sissejuhatus

Droonid on kiirelt arenev ja kasvav turg. 2016. aastal oli nende müügist saadud käive 8,5 miljardit dollarit ja 2021. aastaks eeldatakse käibeks üle 12 miljardi dollari [1]. Suureneva kasutamisega kaasneb ka rohkem probleeme. On olnud juhtumeid, kus vanglatesse on üritatatud droonidega viia narkootikume, mobiiltelefone ja muid keelatud esemeid [2]. 2018. aasta detsembris segati ka Inglismaa Gatwick lennujaama tööd, mille tulemusel katkestati 760 lendu ja segati 110 000 inimese reisiplaane. 2018. aastal tuvastati Inglismaa lennujaamades kokku üle 100 juhtumi droonidega [3]. Samuti segatakse suurüritusi, näiteks 2019. aasta laulupeol üritasid inimesed lennata oma isiklike droonidega. On ka suurenev oht, et droone kasutatakse terrorismis [2]. Selleks, et keelatuid tegevusi droonidega piirata, on vaja neid tuvastada ja tõrjuda.

Droonide juhtimiseks ja droonilt videovoo tagastamiseks kasutatakse raadiosignaale. Antud raadiosignaale on võimalik tuvastada ja seeläbi droone avastada. Samuti on võimalik raadioeetrit segada droonide tõrjumise eesmärgil. Tänapäevased raadioside tuvastamise ja tõrjumise süsteemid ei tööta samaaegselt, sest tõrjumise ajal varjutavad segamise signaalid kõik ülejäänud signaalid ehk segaja segab ka tuvastajat. Selle tagajärjel kaob ülevaade olukorrast: pole võimalik enam drooni asukohta jälgida ja saada teada, kas segamine oli efektiivne. Juhtimissignaali kadudes naaseb droon üldjuhul õhkutõusmispaika ja selle kaudu oleks võimalik leida drooni operaator. Lisaks võimaldaks samaaegne segamine ja tuvastamine kasutada ka efektiivsemat suunatud segamist [4]. Selleks, et panna tuvastamine ja segaja samaaegselt tööle, on vaja eemaldada segamise signaal tuvastaja vastuvõetud signaalist. Antud põhimõte kehtib ka ühes sagedusvahemikus töötaval täisdupleks (*In-band full-duplex* - IBFD) raadiol [5].

Antud töö eesmärk on luua süsteem, mis võimaldab droone tuvastada ja tõrjuda samaaegselt. Süsteemi keskseks osaks on Rantelonis arendatud analoogmoodul. See võimaldab eemaldada enesehäireid vastuvõetud signaalist enne analoogsignaali digitaalseks muundamist. Antud mooduli juhtimiseks oli vaja valida riistvara ja arendada juhtimise tarkvara. Moodulit ja juhtimise tarkvara katsetatati kolmel erineval meetodil, et leida selle süsteemi efektiivsus IBFD raadiona. Viimaks ühendati loodud süsteemi abil droonide tuvastamine ja tõrjumine. Antud tööga kirjeldatakse IBFD raadiot, selle eeliseid ja puuduseid ning antakse ülevaade droonide tuvastamise ja tõrjumise tehnoloogiatest.

2 Ülevaade probleemist

Selles peatükis kirjeldatakse probleemi olemust ja seletatakse, kuidas seda lahendada üritati. Peatükk **2.1** annab ülevaate, millised droonide tuvastamise tehnoloogiad on olemas, ja seletab töös kasutatud meetodit. Järgnevalt **2.2** tutvustakse droonide tõrjumise vahendeid ja kirjeldatakse raadiosidet segavat signaali. Peatükis **2.3** antakse ülevaade IBFD raadiost, selle probleemidest ja kuidas neid lahendada.

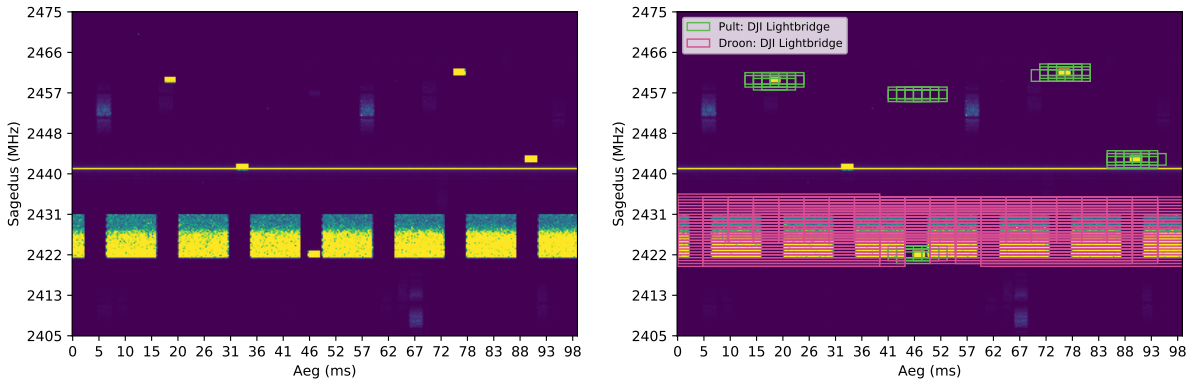
2.1 Droonide tuvastamine

Droonide tuvastamise süsteemid jagunevad kaheks: aktiivsed, mis saadavad signaali välja ja analüüsivad tagasi tulevat signaali, ja passiivsed, mis kuulavad või vaatavad ja analüüsivad saadud informatsiooni. Peamine aktiivne tuvastamise tehnoloogia on radar. Lennunduses on radarid juba aktiivselt kasutuses, aga neil süsteemidel on palju puuduseid, mis ei võimalda nendega tuvastada droone. Esiteks pole lennunduses kasutatavad radarid tihti kalibreeritud tuvastama väikeseid objekte, mille tõttu tuleb tihti kasutada eraldiseisvaid süsteeme. Teiseks on droonid ja linnud samas suurusjärgus, seega ei saa neil lihtsalt mõõtmete järgi vahet teha. Lindudest eristamiseks tuleb kasutada näiteks mikro-Dopplerit, mis lisab süsteemidele keerukust [6].

Passiivsed tuvastajad kasutavad palju erinevaid tehnoloogiaid. Peamine neist on raadioside monitooring. Levinud on ka teised vähemefektiivsed tehnoloogiad, näiteks kaamerad taevast droonide leidmiseks ja mikrofonid droonide poolt tekitatud heli kuulamiseks. Üldjuhul on nende töökaugus madal ja rohkem kasutatakse neid olukorra salvestamiseks või veendumaks, et lendav objekt on droon [7].

Antud töö keskendub raadioside monitooringule, mille puhul kuulatakse pealt drooni ja juhtpuldi vahelist suhtlust. Monitooringu seadmel on tavaliselt mitu suundantenni, tänu millele saab tuvastada kiirgaja suuna. Üldjuhul kasutavad droonid tööstuse, teaduse ja meditsiini (*industrial, scientific and medical* - ISM) jaoks jäetud sagedusi 2,45 GHz (ribalaius 100 MHz) või 5,8 GHz (ribalaius 150 MHz). Samu sagedusi kasutavad ka paljud teised levinud süsteemid nagu WiFi ja Bluetooth [8]–[10]. Kuna antud sagedusi kasutavad ka teised süsteemid, on vaja drooni signaali eristada muudest signaalidest.

Käesolevas töös on kasutusel Ranteloni droonituvastus süsteemi DTS-2458, mis kasutab raadioeetri monitooringuks tehisnärvivõrku. Süsteemi kaheksa suundantenni asemel kasutatakse üht antenni, mille tulemusel kaob suunamääramise funktsionaalsus. Joonisel **2.1a** on kujutatud droonituvastaja sisendit - 70MHz laiune spekter, 100 millisekundi kestvusega. Joonisel **2.1b** on näha, kuidas tehisnärvivõrk on eri spektriosasid liigitanud - roheliste kastikestega on tuvastud drooni puldi signaal ja roosade kastikestega drooni saadetava videovoo signaal [11], [12].



(a) Droonituvastaja sisend.

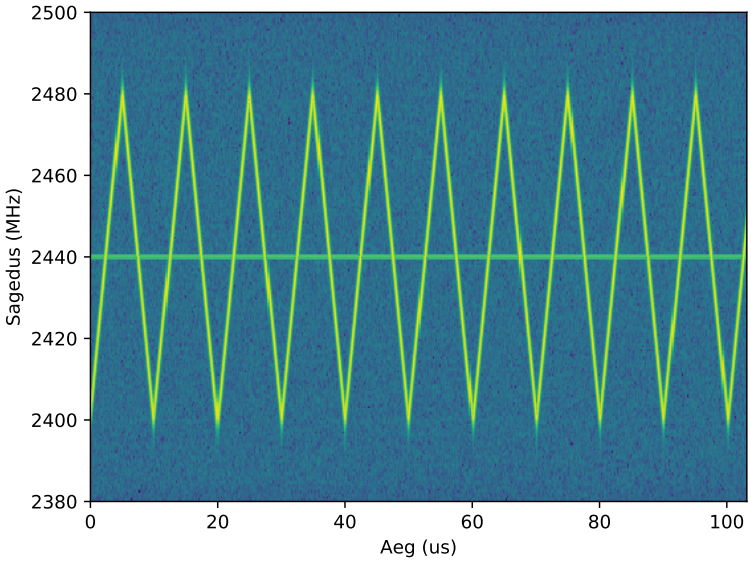
(b) Droonituvastaja väljund.

Joonis 2.1: Droonituvastaja DTS-2458 tööpõhimõte illustratsioon, kus vasakul on tehisnärvivõrku sisestatud spektrogramm ja paremal väljund, kus näha tuvastatud DJI drooni ja pulti. Droonina tuvastatud signaal on droonilt saadetud videosignaal ja puldi tuvastatud signaal drooni juhtmissignaal.

2.2 Droonide tõrjumine

Droonide tõrjumiseks on mitu erinevat võimalust. On olemas drooni füüsiliselt lõhkuvad meetodid, näiteks võrgud või tulirelvad. Kasutatakse ka mittekineetilisi lahendusi, nagu laserid ja elektromagnetrelvad. Samuti on olemas süsteemid, nagu raadioside segajad, mis neutraliseerivad droone. Eksisteerivad ka drooni juhtimise ülevõtvad süsteemid. Üheks selliseks tehnoloogiaks on *GPS spoofing*, mis saadab droonile valet asukohta ning seekaudu on võimalik drooni juhtida [7].

Antud töös kasutatakse droonide tõrjumiseks raadioside segajat. See tehnoloogia kasutab suurel



Joonis 2.2: Segaja signaali illustratsioon spektrogramil.

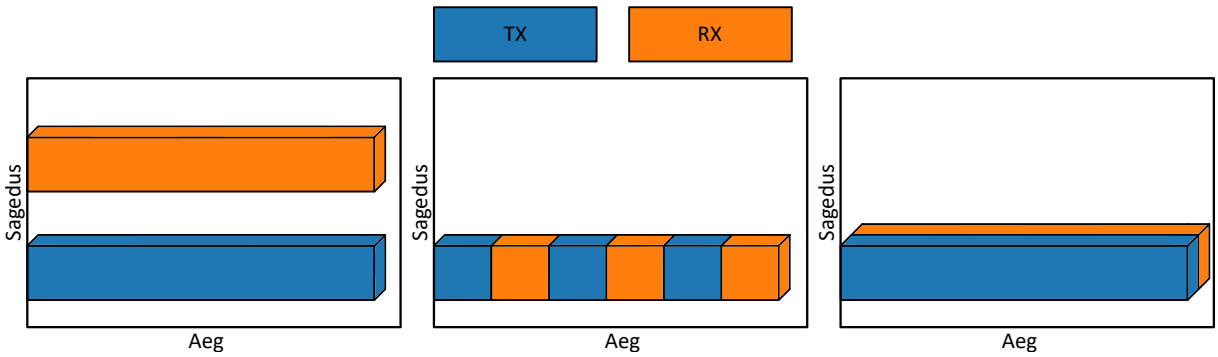
võimsusel signaali saatmist, mis varjutab droonide kasutatavat raadiosidet. Segaja on levinud lahendus troonide tõrjumiseks, sest füüsiliselt midagi ei lõhuta. Samas samaaegselt ei tööta teised süsteemid, mis kasutavad sama sagedusvahemikku. Selleks, et segajaga katta kogu sagedusvahemik, saadetakse kiirelt muutuva sagedusega signaali ehk *sweep* signaali. Drooni segamiseks peab kogu sagedusvahemiku katmise kiirus olema vähemalt 10 kHz. Lisaks on tähtis ka segaja võimsus. Kui saata ISM raadioside vahemiku lubatud maksimum võimsusel 20 dBm, ei suudeta drooni segada väga kaugelt [13]. Kasutatud segaja signaali illustratsiooni spektrogrammil on näha joonisel **2.2**

2.3 Täisdupleks raadio

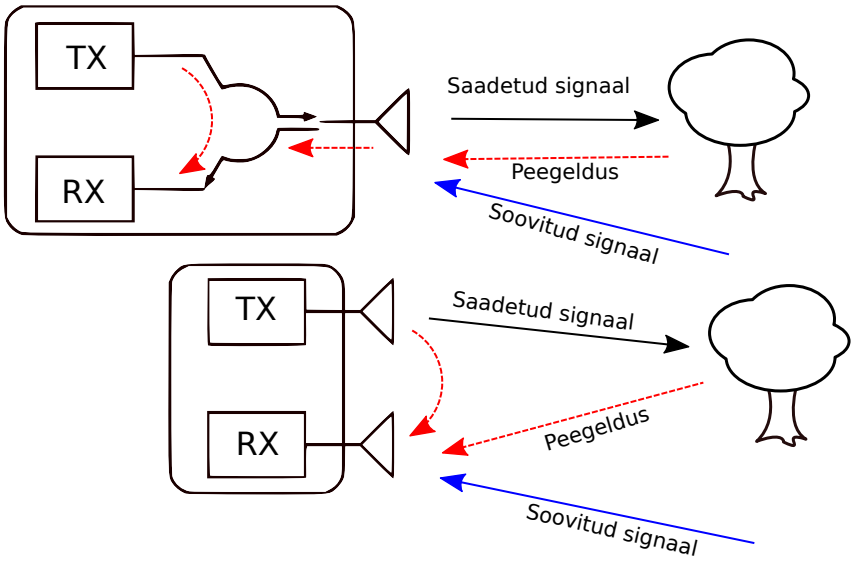
Kui üritada droone segamise ajal tuvastada ja segaja asub tuvastaja läheduses, on raadioeetris droonisignaalid varjutatud segava signaali poolt. Selleks, et droone raadioside abil tuvastada ning samaaegselt tõrjuda, on vaja vastuvõetud signaalist eemaldada segaja signaal. Sama põhimõte, kus vastuvõetud signaalist tuleb eemaldada saadetud signaal, kehtib ka IBFD raadiol. IBFD raadio on kiirelt arenev raadioside valdkond, kuna on tekkinud suurem vajadus kiirema juhtmevaba infovahetuse järele. Andmeside mahu tõstmise üks võimalusi on kasutada suuremaid sagedusvahemike, aga need on piiratud. Praegu on kahepoolse suhtluse korral kasutusel kas sagedusjaotusega duplekseerimine või ajajaotusega duplekseerimine. IBFD võimaldaks saata sama palju andmeid kui sagedusjaotusega, aga kasutades sama palju sagedusvahemikku nagu ajajaotusega (Vaata joonist **2.3**) [5].

IBFD raadio kasutamine pole siiani olnud levinud, sest sellel on probleeme enesehäiretega (*self-interference*). Enesehäirete all mõeldakse häireid, mis IBFDd rakendav seade tekitab saadetava signaaliga iseendale, segades soovitud vastuvõetavat signaali. See tähendab, et saadetud signaal jõuab ka vastuvõetud signaali, varjutades soovitud signaali. Enesehäired jõuavad vastuvõetud signaali erinevates allikatest. Ühe antenniga süsteemidel tekib enesehäired antennist tagasipeegelduva signaali või otse saatjast vastuvõtjasse mineva signaali mõjul. Kahe antenniga süsteemi puhul kostub saadetav signaal vastuvõtvasse antenni. Lisaks jõuab mõlemasse süsteemi enesehäired ka välismaailma peegeldustest. Joonisel **2.4** on näha enesehäirete tekkimise kohad [5].

On palju meetodeid, kuidas IBFD raadios enesehäireid eemaldada. Süsteemis, kus on ühendatud droonituvastus ja -segamine, on võimalik kasutatud segaja signaali ennustamine. Antud teh-



Joonis 2.3: Erinevad raadioside dupleksimise süsteemid. Vasakul sagedusjaotusega dupleksimine, keskel ajajaotusega dupleksimine ja paremal IBFD.



Joonis 2.4: Enesehäirete tekkimise kohad. Üleval ühe antenni süsteem ja all Kahe antenniga süsteem. Punasega märgitud enesehäired.

noloogia on võimalik, kuna kasutatav segaja signaal on ettearvatav ja lihtsalt taastatav. Segaval signaalil on ainult kolm põhilist parameetrit: sagedusvahemik, sagedusvahemiku katmise kiirus ja võimsus. Antud tehnoloogiat on võimalik kasutada ka vastase segamise mõju leevendamiseks oma seadmetes [14]. On olemas ka palju erinevaid digitaalseid süsteeme, mis võttes sisse osa saadetud signaalist, suudab selle eemaldada enesehäireid vastuvõetud signaalist. Ideaalis on kõik enesehäired võimalik eemaldada digitaaldomeenis. Probleem on ainult selles, et kui saadetud signaalitugevus on 20 dBm, vastuvõtjasse jõuab see 15 dB nõrgemana ja vastuvõtja müratase on -100 dBm, siis peab eemaldama 105 dB enesehäireid. Aga näiteks 14-bitise analoog-digitaalmuunduri (ADC) efektiivne dünaamiline vahemik on umbes 54 dB. Selle tulemusel jääb 51 dB enesehäireid alles vastuvõetud signaali. Kuna tänapäeva ADC tehnoloogia on piiratud tuleb enne digitaliseerimist suurem osa enesehäireid juba eemaldada. Esiteks saab selleks kasutada passiivseid meetmeid. Teisek on vaja vähendada enesehäirete tugevust analoogdomeenis, mis on antud töö põhifookuses.[5].

Analoogdomeenis enesehäirete eemaldamiseks on palju erinevaid meetodeid. Üks võimalus on võtta osa saadetavast signaalist, muuta selle viivist, faasi ja võimsust ning seejärel eemaldada see vastuvõetud signaalist [15]. Teine võimalus on muuta osa saadetud signaalist digitaalseks, muuta selle faasi, võimsust, viidet ja seejärel muuta see tagasi analoogsignaaliks ning need eemaldatakse vastuvõetud analoogsignaalist. Vahepeal digitaalseks muutmise eelise on see, et varem mainitud kolme parameetri muutmine on lihtsam [5]. Käesolevas töös kasutatakse analoogsignaalidega töötavat moodulit enesehäirete eemaldamiseks. Selle kirjeldus on peatükis **3.1.1**.

Passiivselt on ühe antenniga süsteemides on võimalik enesehäirete tekkimist vähendada, kui kasutada tsirkulaatorit. Tsirkulaator on passiivne raadioside seade, kus ühe sisendi signaal suunatakse järgmisesse väljundisse ringikujulises järjekorras. Sellisel juhul suunatakse põhiline osa saadetavast signaalist antenni ja vastuvõetud signaali jõuab seda vähem. Kahe antenniga süsteemides on võimalik vähendada enesehäireid, kui optimiseeride antennide vahelist sumbuvust, polarisatsiooni ja suunavust [5].

Üldjuhul rakendatakse IBFD raadios kõiki kolme meetodit enesehäirete eemaldamiseks. Antud töös keskendutakse enesehäirete eemaldamisele analoogsignaalidega.

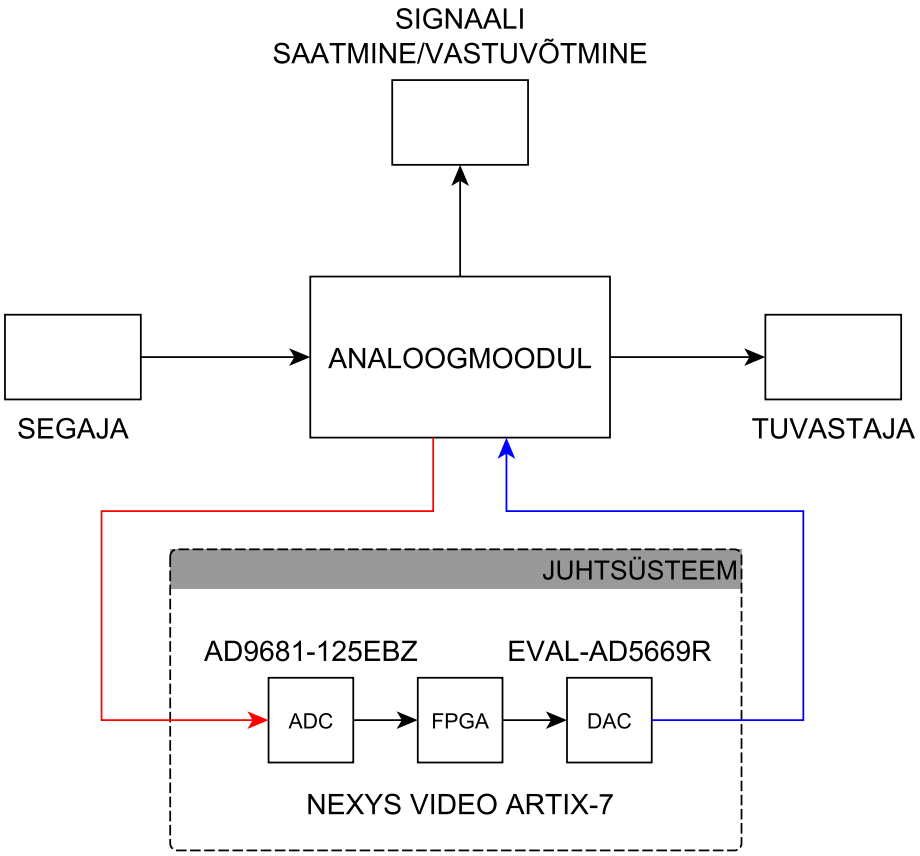
2.4 Varasemad tööd

IBFD raadiot on uuritud juba aastast 1940 [5] ning viimasel ajal on teema populaarsus tõusnud. Stanford ülikoolis rakendati seda 2013. aastal Wifi protokollile ja saavutati 1,87 kordne andmeside kiiruse tõus [16]. Kahe antenniga süsteeme on vähem katsetatud, aga üks hea artikkel on [17], kus luuakse spetsiaalselt IBFD tarbeks antenni. Hea ülevaate täisdupleks raadiost ja selle teemal avalikustatud artiklitest annab [5].

IBFD raadio juhtimissüsteeme on varasemaltki kirjeldatud. Tampere Tehnikaülikoolis kaitstud Joose Tammineni magistritöös [18]. Selles töös on kasutatud analoogne lahutaja arhitektuur, mis sarnaneb käesolevas töös kasutatuga.

Täiendavalt on uuritud IBFD raadiote kasutamist sise- ja riigikaitselistes rakendustes. Täisdupleks raadiot saaks kasutada nii taktikaliseks suhtluseks kui ka elektroonilise sõja pidamiseks. On välja pakutud ka antud töös kasutatud põhimõte, kus raadioeetri segamise ajal samaaegselt seda monitooritakse. Täiendavalt võimaldab IBFD raadio enda suhtlust varjata, kui vstuvõtmise ajal samaaegselt segada selles sagedusvahemikus. Antud tehnoloogiaid on vähe uuritud ja veel vähem rakendatud [19].

3 IBFD süsteem



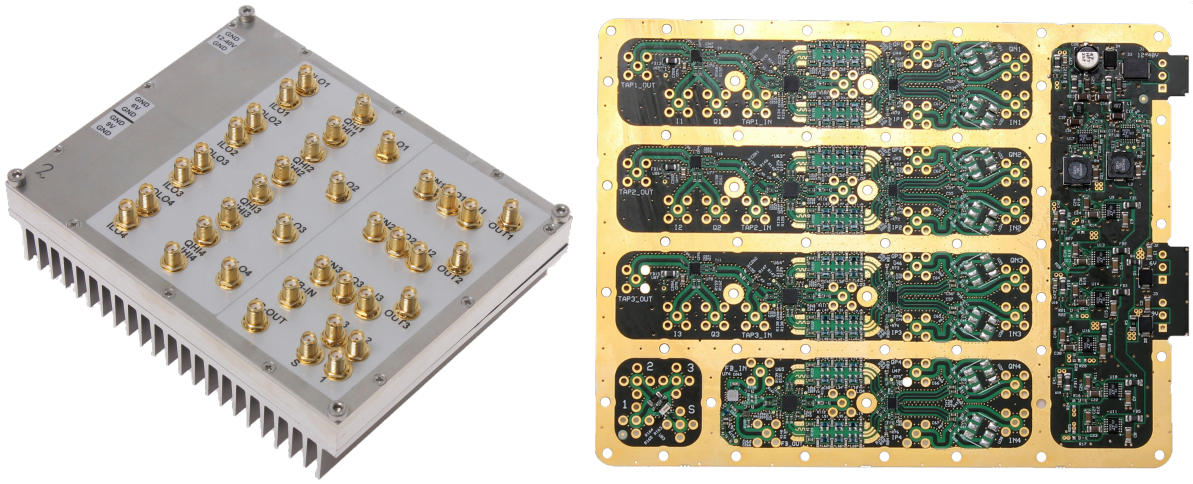
Joonis 3.1: Samaaegse segamise ja tuvastamise süsteemi skeem. Eraldi on välja toodud juhtsüsteemi komponendid. Sinisega on juhtsüsteemi väljundpinged ja punasega sisendsignaalid.

See peatükk annab ülevaate kasutatud riistvarast ning kirjeldab juhtimise tarkvara. Esimesena kirjeldatakse analoogmoodulit, mida kasutatakse enesehäirete eemaldamiseks, seejärel tutvustatakse juhtsüsteemiks vajalike komponente. Tarkvara peatükis kirjeldatakse juhtsüsteemi algoritmi ja antakse ülevaade kogu juhtsüsteemi loogikale. Joonisel **3.1** on näha samaaegse segamise ja tuvastamise süsteemi.

3.1 Riistvara

Süsteemi keskseks osaks oli Rantelonis arendatud analoogmoodul enesehäirete eemaldamiseks. Moodul üritab leida enesehäire signaale, et neid vastuvõetud signaalist eemaldada. Selleks suunatakse moodulile osa saadetud signaalist. Antud signaale viivistatakse ja muudetakse moodulis. Kasutatud moodul võimaldab muuta korraga kolme erinevat signaali, mis sõltuvalt valitud viivistest vastavad erinevatele enesehäiretele. Sisendsignaale viivistatakse enne moodulisse jõudmist erinevate pikkusega kaablitega ja pärast moodulit lahutataks need signaalid vastuvõetud signaalist maha. Neid kolme sisendit nimetatakse järkudeks (*tap*). Kogu moodul on ettenähtud töötama 2,4 GHz juures. Moodulit korpuses ja selle trükkplaati on näha jooniselt **3.2**. Antud mooduli kuvand pärineb artiklist [15]. Samas võib seda pidada töös [18] kasutatud mooduli edasiarendusena.

3.1.1 Analoogmoodul enesehäirete eemaldamiseks



(a) Analoogmoodul korpuses.

(b) Analoogmooduli trükkplaat.

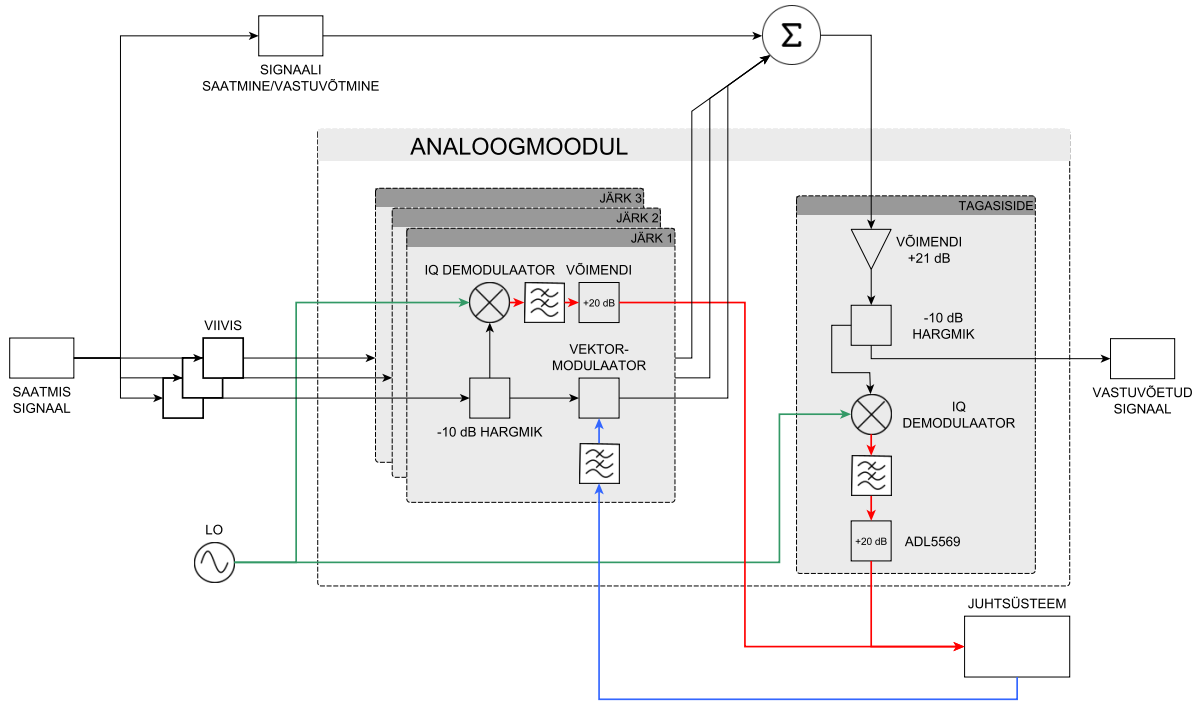
Joonis 3.2: Rantelonis arendatud analoogmoodul enesehäirete eemaldamiseks.

Mooduli kõige tähtsamad komponendid on kolm vektormodulaatorit HMC631 Analog Devicesilt. Ühte vektormodulaatorit kasutatakse ühe filtri järgu kohta. Vektormodulaatoriga on võimalik muuta analoogsignaali faasi ja võimsust. Vektormodulaatorit juhitakse I ja Q sisendtitele antud pingetega. Soovitud võimenduse G ja faasi θ kohta sisendpinged arvutatakse valemiga **3.1**. Seal V_{mi} ja V_{mq} viitavad pingetele, kus vektormodulaator pakub kõige rohkem isolatsiooni (nominaalselt 1,5V). Soovitud võimendus normaliseeritakse maksimaalse võimendusega G_{max} (nominaalselt 0,316) [20].

$$\begin{aligned} I(G,\theta) &= V_{mi} + 1.0V\frac{G}{G_{max}}Cos(\theta) \\ Q(G,\theta) &= V_{mq} + 1.0V\frac{G}{G_{max}}Sin(\theta) \end{aligned} \tag{3.1}$$

Iga järgu kohta on enne vektormodulaatorit -10 dB hargmik, mis saadab osa signaali demodulaatorisse. Demodulaator jagab raadiosignaali kaheks signaaliks, kus üks faas on teise suhtes 90° nihkes. Lisaks alandavad nad signaali sagedust sõltuvalt neile antud kohaliku ostsillaatori (*local oscillator* - LO) signaali sagedusest. Selleks korrutavad nad sisendsignaali LO signaaliga. Väljundsignaale nimetatakse IQ signaalideks. Need IQ signaalid läbivad madalpääsfiltri ja 20 dB võimendi. Seejärel suunatakse nad moodulist välja ja kasutatakse juhtimissüsteemis. Edaspidi viidatakse antud IQ signaalidele kui I_n ja Q_n, kus n on järk.

Vastuvõetud signaalist lahutatakse analoogmooduli järkude väljundid ja suunatakse tagasi moo-



Joonis 3.3: Analoogmooduli skeem. Rohelised jooned näitavad LO signaali, punased jooned moodulist juhtsüsteemi saadetavaid I ja Q signaale ning sinised jooned juhtsüsteemist saadetavaid juhtimispingeid.

dulisse. Seal võimendatakse seda 21 dB. Seejärel suunatakse signaal -10 dB hargmiku, kust põhiline väljund on mooduli lõplik väljund. Hargmiku -10 dB väljund suunatakse samase ahelasse nagu järkude sisendid, kus on demodulaator, filter ja võimendi. Tekitatud IQ signaalid suunatakse moodulist välja ja kasutatakse juhtsüsteemis tagasisidena. Edaspidi viidatakse antud IQ signaalidele kui I_e ja Q_e. Analoogmooduli skeemi on näha jooniselt **3.3**.

3.1.2 Analoog-digitaalmuundur

Kontrollsüsteemi kõige tähtsam osa on ADC, mille valimisel oli mitu olulist parameetrit. Esiteks peab korraga saama lugeda kaheksat kanalilt. Kaheksa kanali nõue tuleb analoogmoodulist, kust saadetakse välja neli erinevat IQ signaali. Nendeks signaalideks on kolm signaali järkudest ja tagasiside signaal. Teiseks oli tähtis muunduri kiirus, et saaks analüüsida võimalikult laia ribalaiust. Droonid töötavad sagedusvahemikus 2,40 GHz kuni 2,48 Ghz ehk 80 MHz ribalaiusel [9], mille tulemusel on ka segav signaal 80 MHz ribalaiusega [13]. Seega ADC kiirus

peab olema vähemalt 80 MSPS, et analüüsida sama ribalaiust. ADCl peab olema ka efektivne dünaamiline vahemik võimalikult suur, et arvutused oleksid piisavalt täpsed. Kolmandks oli tähtis, et ADCl oleks arendusplaat ja seda saaks ühendada juhtplaadiga.

Kõige paremini sobis Analog Devices AD9681 ADC ja selle AC9681-125EBZ arendusplaat. AD9681 on kaheksa kanaliga, 14-bitine ja 125 MSPS ADC. See kasutab madalapingelist diferentsiaalset signaaliedastusmeetodit (LVDS), et saata muundatud digitaalsed andmed juhtsüsteemile. ADC töö juhtimiseks kasutatakse SPI liidest [21]. Arnendusplaadi liidestamiseks kontrollplaadiga on sellel FMC ühendus[22], mis on levinud standard kontrollerite arendusplaatidel väliste moodulite lisamiseks.

3.1.3 Digitaal-analoogmuundur

Vektormodulaatorite juhtimiseks kasutatakse analoogsignaale, mille tarbeks oli vajalik juhtsüsteemi digitaalsed väljundid muundada. Selleks kasutatakse digitaal-analoogmuundurit (DAC). DACi peamine parameeter, mida vaja jälgida, oli juhitavate kanalite arv. Korraga peab juhtida saama kolme vektormodulaatori, mis tahavad kahte analoogjuhtimissignaali [20]. Lisaks on tähtis DACi täpsus ja juhtimisprotokoll. Kuna enamikel juhtplaatidel on ainult üks kiire ja paljude ühendustega liides, mis oli ADC poolt juba kasutatud, pidi DAC kasutama vähemate juhtmetega ja seetõttu ka aeglasemat portokolli.

Valitud Analog Devices AD5669R DAC on kaheksa kanaliga ja 16-bitise täpsusega ning juhtitav üle I2C liidese [23]. Lisaks on sellel olemas arendusplaat EVAL-AD5669RSDZ. Antud arendusplaadil on vaikimisi peal 5V väline referents [24]. Vektormodulaatorid vajavad tööks pingeid vahemikus 0 kuni 3V [20]. Sellest lähtuvalt sai arendusplaadil olev kivi välja vahetatud 3 V variandi vastu, et ei kaotaks DACi täpsust.

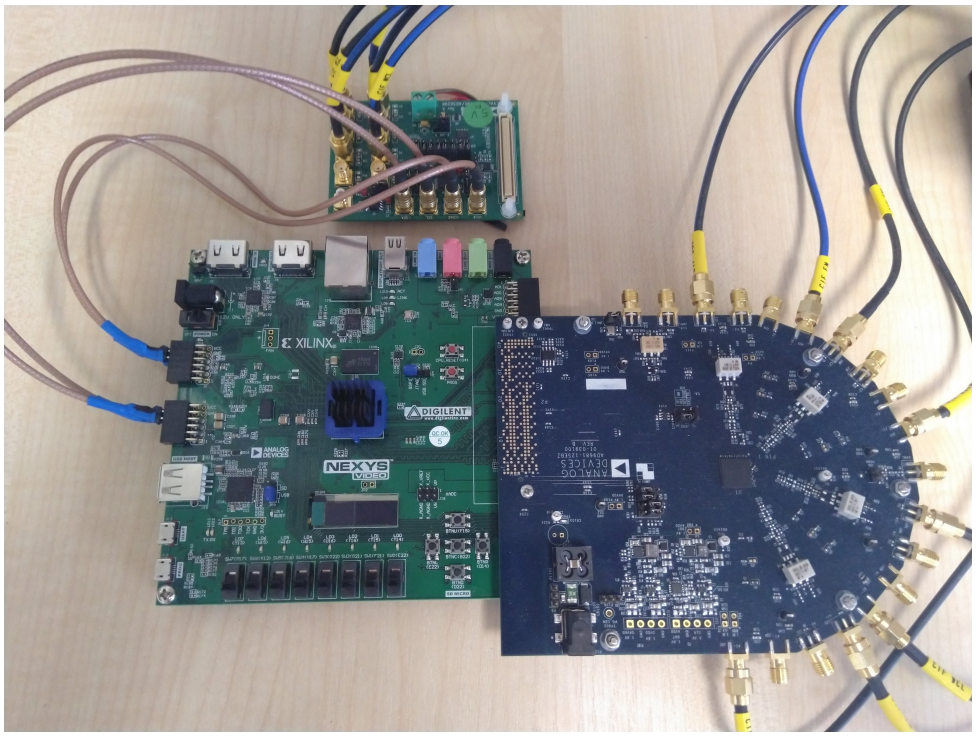
3.1.4 Juhtplaat

Juhtplaadi valimisel oli paar väga olulist kriteeriumit. Esiteks peab olema võimalik sellega suurel kiirusel ja paralleelselt töödelda andmeid ning teha arvutusi. ADC saadab kiirusel 500 MHz ning see peab olema minimaalne kiirus, mida juhtplaadil olev kontroller võimaldab. Selliseks probleemiks sobib kõige paremini FPGA. Lisaks peab olema võimalik liidestada DACi ja ADCd. Suurema piirangu paneb ADC, kuna see kasutab FMC ühendust.

Antud kriteeriumile vastas kõige paremini Digilent Nexys Video arendusplaat. Sellel on Xilinx Artix-7 XC7A200T FPGA 33650 loogikaplokkiga. Arendusplaadil on välja toodud palju FPGA jalgu, et ühendada erinevaid väliseid seadmeid. Ühtede nende külge on ühendatud DACi I2C ja muud kontrollühendused. Lisaks on sellel FMC ühendus, mis liidestub kasutatud ADCga. Arendusplaadil on ka UART liides, mida kasutatakse info saatmiseks arvutile [25]. Lisaks võimaldavad Xilinxi FPGAd kasutada täiendavaid silumise vahendeid, millega on võimalik muuta ja vaadata FPGA siseseid väärtuseid.

3.2 Tarkvara

Süsteemi tarkvara kirjeldus on jagatud kaheks. Peatükis **3.2.1** kirjeldatakse juhtimise algoritmi, programmi ülesehitust ja selle kirjutamiseks kasutatud vahendeid. Ülejäänud süsteemi tarkvara

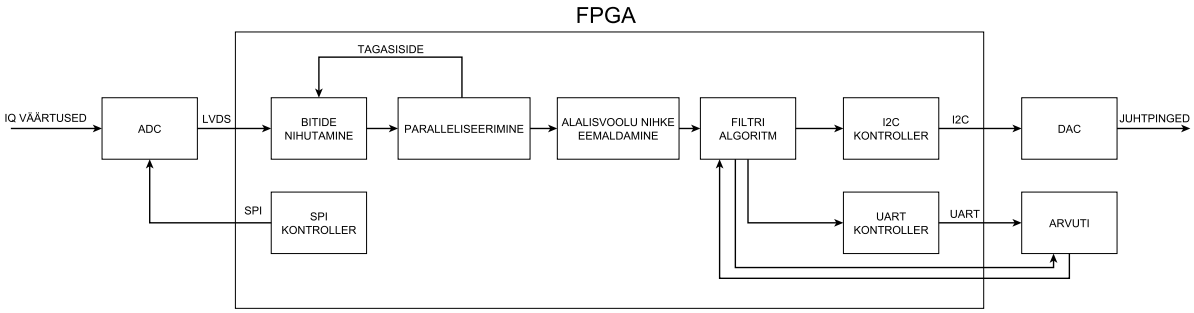


Joonis 3.4: Analoogmooduli juhtimissüsteem. Üleval on DAC arendusplaat, Vasakul FPGA arendusplaat ja paremal ADC arendusplaat.

kirjeldatakse peatükis **3.2.2**. Kogu süsteemi tarkvara plokkid ja andmete liikumine on kuvatud joonisel **3.5**

3.2.1 Juhtimise algoritm

On palju süsteeme, kus soovitud signaalist on vaja eemaldada ennustatavat müra. Näiteks on selliseid probleeme elektrokardiograafias ja kõnes. Sellist müra eemaldamist nimetatakse adaptiivseks müra summutamiseks ja selleks kasutatakse adaptiivseid filtreid [26]. On olemas palju erineviad adaptiivseid filtreid, nagu diferentiaalne järseima vähenemise (*differential steepest-descent*) ja lineaarne juhusliku otsimise (*linear random search*) algoritm. Antud töös kasutatakse väikseima ruutkeskmise (*least mean squares* - LMS) algoritmi, mis on varasemalt kasutatud IBFD raadiotes [18]. Samuti on leitud, et see on mainitud algoritmidega võrreldes kõige



Joonis 3.5: FPGAl jooksva süsteemi loogika skeem.

efektiivsem [27]. Töö tegemise raames katsetati ka normaliseeritud LMS algoritmi, kuid selle saavutatud tulemused olid halvemad ja jäeti tööst välja.

LMS algoritmi iseloomustab valem **3.2**. Antud valemis μ on filtri samm, millega määratakse väljundväärtuse muutumise kiirus. x_n on n järgu analoogmooduli väljundsignaalid kujul $I_n + Q_n \cdot j$ ning $x_n^*(t)$ on selle kompleks konjugatsioon. $e(t)$ on tagasisidesignaal kujul $I_e + Q_e \cdot j$. t on diskreetne aeg. Valemi tulemust nimetatakse filtri kaaluks w ja see jagatakse reaal- ja imaginaarosaks ning suunatakse tagasi analoogmoodulisse. Lisaks hoitakse seda väärtust meeles järgmise ajahetke arvutusteks [28].

$$w_n(t+1) = w_n(t) + \mu x_n^*(t) e(t) \tag{3.2}$$

LMS algoritm muudab iga ajahetk t natuke oma väljundi w väärtust selles suunas, et antud ajahetkel väheneks tagasiside. Filtri väljundi muutumise kiirus sõltub nii sammu μ suurusest kui ka sisendite x_n ja e võimsusest. Kui tagasiside signaal on null, siis pole võimalik seda vähendada ja filtri väljund ei muutu. Kui x_n on null, siis ka väljund ei muutu, sest selle muutmine ei anna mingit efekti tagasisidele. Kuna käesolevas süsteemis on vastuvõetud signaal tagasisideks ja saadetud signaal sisendiks, üritab LMS eemaldada saadetud signaali vastuvõetud signaalist ehk eemaldada enesehäireid.

FPGAl kasutati LMS algoritmi realiseerimiseks Vivado High-Level Synthesis (HLS) tarkvara. HLS võimaldab kirjutada FPGAde jaoks tarkvara C, C++ ja System C programmeerimiskeeltes. See teeb lihtsamaks keerulisemate funktsioonide kirjutamise, näiteks võimaldab see lihtsalt realiseerida trigonomeetrilisi ja lineaaralgebra valemeid. See võimaldab kergemini arendada algoritme, arvestamata riistvara piirangutega. Lisaks saab HLSiga koheselt kontrollida algoritmi tööd C keeles [29]. Iga mooduli sisendi kohta käivitatakse üks filtri funktsiooni üksus, mis jookseb pidevalt ja ei oota, kuni arvutatud väärtused DACile saadetakse.

Kirjutatud LMS filtri funktsiooni tööd saab jagada kolmeks etapiks: sisendite muutmine ujukoma kompleksarvudeks, uue kaalu väärtuse arvutamine ja väjundite määramine. LMS algoritmi programmi on näha **3.1**. C programmeerimiskeele kasutamine filtri algoritmi kirjutamiseks teeb programmeerijale töö palju lihtsamaks. On olemas kompleksarvude andmestruktuurid ja nendega seotud funktsioonid. Kui VHDLis oleks pidanud eraldi mooduleid hakkama looma kompleksarvude korrutamiseks, saab C keeles kasutada lihtsalt korrutamise operaatorit.

Programm 3.1: LMS filtri programm.

```
#include "lms_filter.h"

void lms_filter_top(
                in_data_t& in_i, in_data_t& in_q,
                in_data_t& e_i, in_data_t& e_q,
                in_data_t& mu,
                out_data_t& out_w_i, out_data_t& out_w_q
                )
{
        // Staatiline kaalu muutuja
        static std::complex<float> w(0.0, 0.0);
```

```
        // Ütleb, et kaalu muutuja taastataks algväärtustele
        // lähtestamis signaali saabumisel
        #pragma HLS reset variable=w
        // Muudab sisendväärtused ujuvkoma kompleksarvudeks
        std::complex<float> in(in_i.to_float(), in_q.to_float());
        std::complex<float> e(e_i.to_float(), e_q.to_float());
        // Uuendab kaalu väärtust
        w += mu.to_float() * std::conj(in) * e;
        // Väljasta I ja Q väärtused hetkesest kaalust
        out_w_i = w.real();
        out_w_q = w.imag();
}
```

Samas peab HLSi kasutades arvestama siiski mõnede riistvara piirangutega. Staatiliste muutujate puhul peab kirjeldama, mis nendega tehakse, kui riistvarast tuleb lähtestamise signaal. Lisaks peab arvestama, milline on sisend- ja väljundväärtuste andmestruktuur. Antud juhul kasutatakse 16-bitiseid püsikoma arve, kus üks bit on märgile ja 15 biti murdarvudele. Andmetüüpide defineeringud on näha päisefailis **3.2**.

Antud funktsiooni jooksutamine võtab kasutatud FPGA peal aega 39 taktitsüklit. Kasutatud takti kiiruseks on 125 MHz, mis on sama ADC kiirusega. Funktsiooni saaks küll paralleliseerida, et teha selle tööd kiiremaks. Ühe osana saaks teha sisendväärtuste muutmise. Teise osana uuendada kaalu arvutust. Kolmanda osana muuta väljundeid. Sellisel juhul saaks funktsiooni jooksutada tihemini kui iga 39 taktsitsükli tagant ning alustada uue funktsiooni täitmist, kui eelmine osa on tehtud. Antud võimalust ei rakendatud kahel põhjusel. Esiteks on vaja veenduda, et staatiline kaalu muutuja oleks sama igas paralleelselt jooksvas funktsioonis. Teiseks ei anna filtri algoritmi töö kiirendamine palju juurde, sest I2C liides DACi juhtimiseks on kordades aeglasem.

Programm 3.2: LMS filtri programmi päis.

```
#ifndef LMS_FILTER_H
#define LMS_FILTER_H

#include <complex>
#include <ap_fixed.h>

#define IN_BW   16
#define IN_IW    1
#define OUT_BW  16
#define OUT_IW   1

// Tüübidefineering sisend- ja väljundmuutujatele
// Kasutatud 16 (1:15) bitist märgiga püsikoma arvu saturatsiooniga
typedef ap_fixed<IN_BW,IN_IW, AP_TRN, AP_SAT>   in_data_t;
typedef ap_fixed<OUT_BW,OUT_IW, AP_TRN, AP_SAT> out_data_t;

// Filtri algoritmi funktsioon
void lms_filter_top(
                in_data_t& in_i, in_data_t& in_q,
                in_data_t& e_i, in_data_t& e_q,
                in_data_t& mu,
                out_data_t& out_w_i, out_data_t& out_w_q
                );
```

```
#endif
```

Filtri funktsiooni üksuse juhtimiseks FPGAl kirjutati VHDL keeles ümbris. Antud ümbris suunab õiged ADC muundatud signaalid õigete funktsiooni sisendite külge. Lisaks töötleb ümbris funktsiooni väljundid I2C kontrolleri jaoks arusaadavaks. Näiteks on vaja filtri *two's complement* väljundväärtused muuta *offset binary* arvutüüpideks. Kuna funktsioonide väljundid muutuvad kiiremini kui I2C suudab DACile käske saata, saadetakse kindlas järjekorras igale DAC väljundile sellel hetkel kõige uuemad väärtused. Sellise meetodiga uuendatakse igat DAC väljundit võrdse intervalli tagant. Veel rakendatakse ümbrises virtuaalne sisend-väljund (VIO). See on tuum Xilinxi FPGAdele, mis võimaldab muuta ja lugeda signaale FPGA sees. Seda moodulit kasutatakse filtri järkude sisse ja välja lülitamiseks, LMS filtri sammu μ muutmiseks ja filtri alguväärtuste taastamiseks.

3.2.2 Kogu süsteem

Ülejäänud tarkvara kirjutamiseks FPGAle kasutati Xilinx Vivado Webpacki ja VHDL riistvarakirjelduskeelt. Üks tähtis osa tööst on ADClt korrektsete andmete kätte saamine. Nagu varem mainitud, kasutab ADC digitaalsete andmete edastamiseks kontrollplaadile LVDS standardit. Täpsemalt saadetakse digitaalsed andmed üle kahe diferentsiaalse andmeraja paari. Kontrollivateks signaalideks kasutatakse biti taktsignaali ja paketi taktsignaali. Ühe paketi suuruseks on 8 biti ehk ühe kaadriga saadedakse kontrollerile 16 biti andmeid. ADC 125 MSPS kiiruse juures on paketi taktsignaali kiirus 125 MHz ja biti taktsignaali kiirus 500 MHz. Taktsignaali kiiruste neljakordne vahe tuleneb bitide topeltkiirusega (DDR) saatmisest, kus uus biti väärtus loetakse nii tõusval kui ka langeval taktil. Kuna suurel kiirusel saadetavatel andmetel võib tekkida bitinihe ja viivis, siis neid on vaja korrigeerida [21]. ADClt andmete lugemisprogrammi on näha lisas 1.

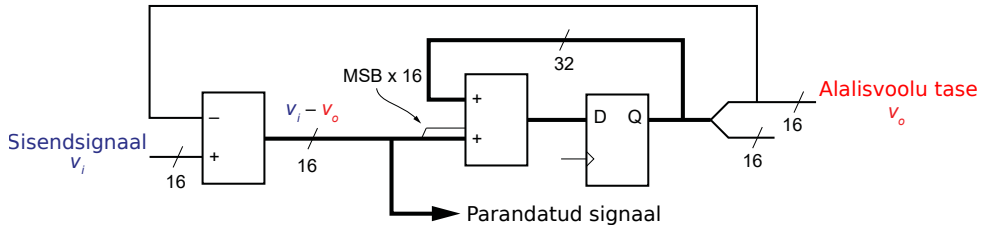
ADCd juhtimiseks oli vaja SPI kontrollerit, milleks kasutati olemasolevat moodulit [30]. SPI liidesega oli võimalik kirjutada ADC registrisse selle tööd muutvaid väärtuseid. Täiendavalt oli võimalik lähtestada seda. ADCd lähtestatakse peale igat käivitamist, et vältida vanade seadistuste kasutamist. Seejärel pannakse ADC saatma kindlat mustrit, mille järgi on võimalik bitinihe määrata. Alles peale neid protseduure alustab ADC tööd. Lisaks on võimalik ADC panna saatma pseudojuhuarvujadu PN9 ja PN23 [21]. Antud arvujadadega kontrolliti ADC tööd muutuvate väärtustega. Selleks pandi ADC saatma ühte neist arvujadadest, mille saadetud väärtused salvestati kasutades sisseehitatud silumisvahendeid. Sama arvujada genereeriti Pythoni programmi abil. Kahte genereeritud arvujada võrreldi omavahel ja tulemuseks saadi, et arvujadad ei erinenud üksteisest. Sellega veenduti, et ADCs muundatud väärtused jõuavad korrektselt FPGAsse. PN9 arvujada genereerimise ja selle ADC saadud väärtuste võrdlemise programm on lisas 2.

ADC saadetud väärtuste arvutisse salvestamiseks kasutatakse integreeritud loogikaanalüsaatorit (ILA). Sellega on võimalik korraga salvestada kuni 131072 järjestikust väärtust FPGA sisesesse mällu. Väärtused on vaja salvestada seadmesse, sest ADC töökiirusele vastavat liidest arvutiga suhtlemiseks arendusplaadil ei ole. Lisaks on sisseehitatud loogikaanalüsaatoril võimalus salvestada väärtusi automaatselt peale mingit trigerit [31]. Näiteks saab automaatselt alustada salvestamist, kui tuvastatakse genereeritud arvujada algus ja seejärel salvestada see arvutisse.

Analoogmooduli tööks oli vajalik juhtida DACi. Selleks oli vaja realiseerida I2C kontroller, milleks kasutati olemasolevat koodi kohandades seda vastavalt vajadustele [32]. Peamiselt tehti

andmepaketi suuruse muutmist lihtsamaks. Selleks lisati muutuja, mille väärtust muutes sünteseeriti kontroller vastavaks. I2C kontrolleri ümber tehti pakett, mis juhtis õigete väärtuste saatmist õiges järjekorras.

ADC väljundväärtuste keskmine amplituud ei ole null. Antud nähtust nimetatakse alalisvoo-

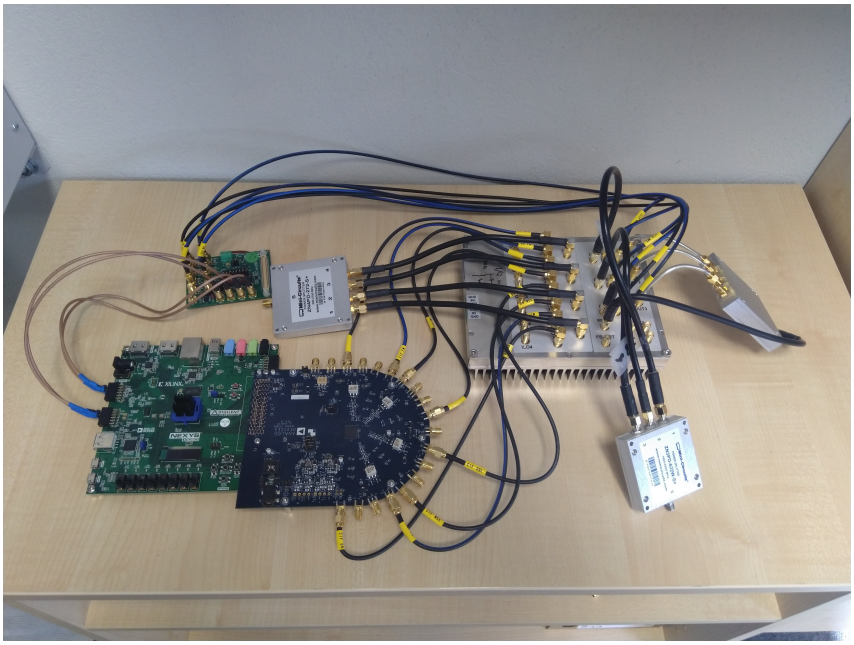


Joonis 3.6: Alalisvoolu nihke eemaldava loogika plokkskeem [33].

lu niheks ning seda on vaja eemaldada. Selleks rakendati digitaalselt lihtsustatud loogikaga kõrgpääsfiltrit, mis immiteeris takisti ja kondensaatori filtrit. Loogika alguses eemaldatakse sisendsignaalist arvutatud alalisvoolu nihke väärtus. Antud väärtus on algväärtustatud nulliks. Saadud lahutus on süsteemi väljundiks. Samas laiendatakse lahutuse tulemuse arvu kuju 16-bitist 32-bitiseks. Saadud väärtus suunatakse akumulaatorisse, mille väljundväärtuse ülemised 16 biti võetakse alalisvoolu nihke väärtuseks. Loogika skeemi on näha jooniselt **3.6** [33]. Alalisvoolu nihke eemaldamise programmi on näha lisas 3.

Täiendavalt rakendati UART kontroller info saatmiseks arvutile. UARTiga saadeti reaalajas filtri järkude väljundväärtuseid. UART kontroller saatis samu väärtusi arvutile, mis saadeti DAC väljunditesse. Arvutis kuvati reaalajas väärtused graafikule kasutades Pythonis kirjutatud programmi. Antud graafikult oli võimalik kiirelt lugeda, kas LMS filter töötas korrektselt.

4 Tulemused ja analüüs



Joonis 4.1: Katsetamiseks kasutatud süsteem, kus on näha analoogmoodul, selle tööks kasutatud jagurid ja liitja ning mooduli juhtimiseks kasutatud FPGA, ADC ja DAC arendusplaadid.

Analoogmooduli ja selle juhtimissüsteemi tööd testiti kolmel meetodil: terminaatoriga, ühe antenniga ja kahe antenniga. Peatükis **4.1** kirjeldatakse ja analüüsitakse katset terminaatoriga. Antud katse eesmärgiks oli veenduda süsteemi töös ja saada esialgsed tulemused. Peatükis **4.2** kirjeldatakse ja analüüsitakse katset süsteemiga, kus kasutati üht antenni. Selle katsega tõestati süsteemi töökindlust ka reaalses keskkonnas, kus on rohkem enesehäire tekkekohti. Peatükis **4.3** kirjeldatakse ja analüüsitakse kahe antenniga süsteemi katset, kus segaja väljundvõimsus

Tabel 4.1: Kõik katsetatud süsteemid ja signaalide võimsused erinevates punktides. Saavutatud lahutus eelmisest etapist on sulgudes. Kokku saavutatud lahutus on viimases tulbas.

Süsteem	Saatmis võimsus	Vastuvõtmis võimsus (lahutus)	Mooduli väljundvõimsus (lahutus)	Kokku lahutus
Terminaator	5 dBm	-32 dBm (-37 dB)	-80 dBm (-48 dB)	-85 dB
Üks antenn	-3 dBm	-30 dBm (-27 dB)	-62 dBm (-32 dB)	-59 dB
Kaks antenni	17 dBm	-32 dBm (-49 dB)	-54 dBm (-22 dB)	-71 dB

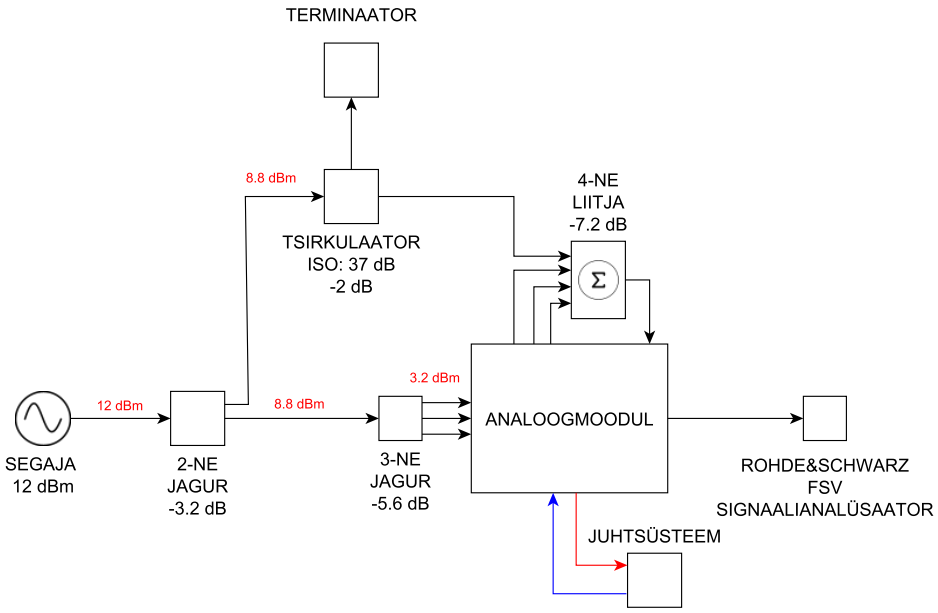
oli piisav droonide tõrjumiseks. Igas katses kasutatav segamissignaal töötas vahemikus 2,40 kuni 2,48 GHz ehk ribalaiusel 80 MHz. Segaja kattis antud sagedusvahemiku 10 kHz. Kogu kasutatud süsteemi on näha jooniselt **4.1**. Kõikide testide koondtulemusi on näha tabelist **4.1**. Vastuvõetud signaali võimsuse arvutustel pole arvestatud liitja kaoga, mille tulemusel on saadud tulemused 7,2 dB madalamad.

4.1 Terminaator

4.1.1 Süsteem

Analoogmooduli kontrolleri esimeseks testimiseks on süsteem, kus välismaailma midagi ei kiirata ja antenni asemel on terminaator. Antud meetod on hea katsetamaks kogu süsteemi, sest välised mõjutajad on elimineeritud. Ainsad enesehäired, mis sellise süsteemiga vastuvõetavasse signaali jõuavad, on peegeldus terminaatorist ja tsirkulaatori läbilase. Testimiseks kasutatav segaja signaal suunati kahesesse jagajasse, mille võrdsed väljundid suunati läbi viivskaablite moodulisse ja tsirkulaatorisse. Tsirkulaatori sisendid olid ühendatud järgmises järjekorras: segaja signaal, atenuaator ning vastuvõetav signaal. Tsirkulaatori väljundis oli signaalitugevus 5 dBm. Agilent N5181A signaaligeneraatorit kasutati LO signaali genereerimiseks. Kogu testitud süsteemi skeemi on näha jooniselt **4.2**.

Antud süsteemile vastavad viiviskaablid arvutati välja kasutades Rohde & Schwarz ZNB 8 vek-

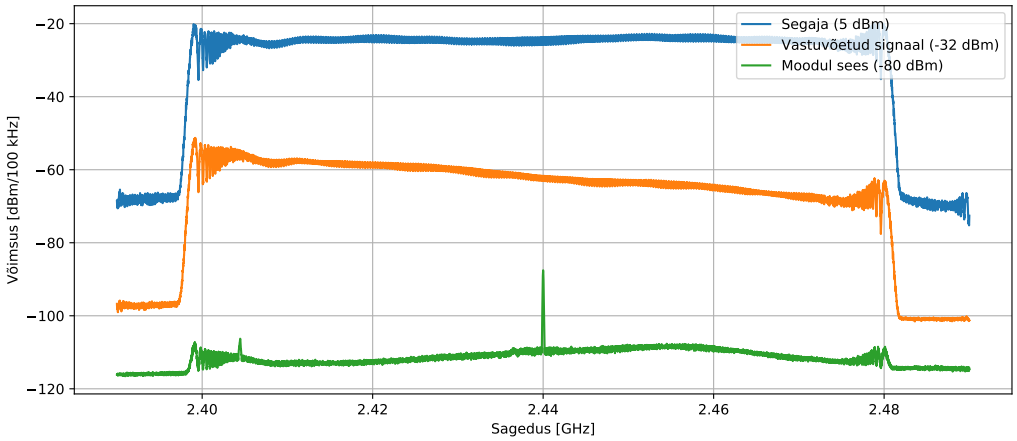


Joonis 4.2: Katsetamise süsteem, kus antenn on asendatud terminaatoriga. Joonisel on näha saadetud signaali kaod igas komponendis. Lisaks on punaselt välja tootud saadetud signaali võimsused peale igat komponenti. Punase noolega on ADCle saadetavad iga järgu IQ signaalid. Sinise noolega on DACi väljundiks olevad vektormodulaatori kontrollsignaalid.

tor võrguanalüsaatorit, millega mõõdeti sageduskostet. Saadud mõõtmistulemused salvestati ja arvutiga leiti selle Fourier' teisenduse vastand. Saavutatud tulemus on impulsskoste. Impulsskoste analüüsimisel leiti tekivatele enesehäiretele vastavad kaablipikkused. Täiendavalt kasutati Agilendi N5181A signaaligeneraatori, millega tekitati moodulile 2,44 GHz LO signaali

võimsusel 6 dBm. Spektrogrammi vaatamiseks ja salvestamiseks kasutati Rohde & Schwarz FSV signaalianalüsaatorit. Lisaks arvutati antud signaalianalüsaatoriga signaali võimsus. Antud võimsus erineb spektrogrammiga nähtavast võimusest, sest segaja signaal on laial ribal ja kiirel sagedusel liikuv signaal.

4.1.2 Tulemused

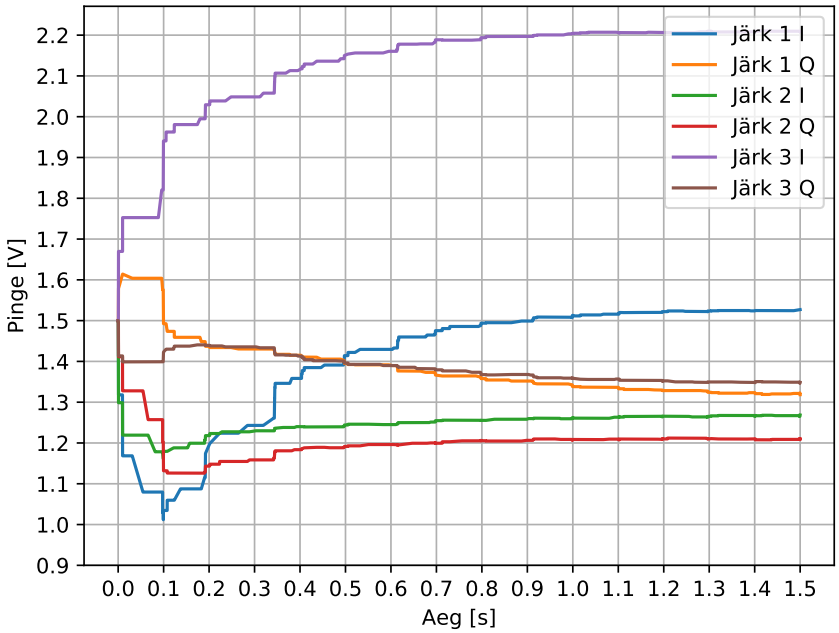


Joonis 4.3: Atenuaatoriga süsteemi tulemus, kus kuvatud saadetud signaal, vastuvõetud signaal, kui analoogmoodul on väljas ja vastuvõetud signaal kui analoogmoodul sisse lülitatakse. Joonise legendis on sulgudes võimsuse mõõtmise tulemus.

Süsteemi saadetud signaal, vastuvõetud signaal ja signaal, kui analoogmoodul on sees on näha jooniselt **4.3**. Vastuvõetava signaali tugevus antud süsteemis väljalülitatud mooduli korral oli -32 dBm. Kuna saadetava signaali tugevus oli 5,0 dBm, siis tsirkulaator andis terminaatoriga 37 dB isolatsiooni. Täiendavalt langes vastuvõetud signaalitugevus pärast mooduli sisselülitamist 48 dB -80 dBm võimsuseni. Kokku saavutas süsteem 85 dB peegelduste eemaldamist. Analoogmooduli väljundsignaali kostus ka kasutatud 2,44 GHz sagedusega LO signaal, mis moodustas spektrogrammi keskele tipu.

Süsteemi efektiivsus sõltus kasutatud filtri sammu μ suurusest. Mida madalam on samm, seda rohkem suudab analoogmoodul enesehäireid eemaldada. Samas võtab ka madala sammuga õigete väärtuste leidmine kauem aega. Hea vahekord kiiruse ja efektiivsuse vahel leiti, kui $\mu = 0.00046$. Suuremate kaaludega hakkas efektiivus langema ja väiksematega võttis kauem aega. Kasutatud sammu juures väljundpingeid pärast mooduli sisselülitamist on näha jooniselt **4.4**.

Saavutatud tulemus oli hea indikatsioon, et analoogmooduli juhtimise süsteem toimib. Saavutatud tulemustega oli hea minna edasi järgmisele süsteemile, kus atenuaator oli asendatud antenniga.



Joonis 4.4: Juhtsüsteemi väljundpinged.

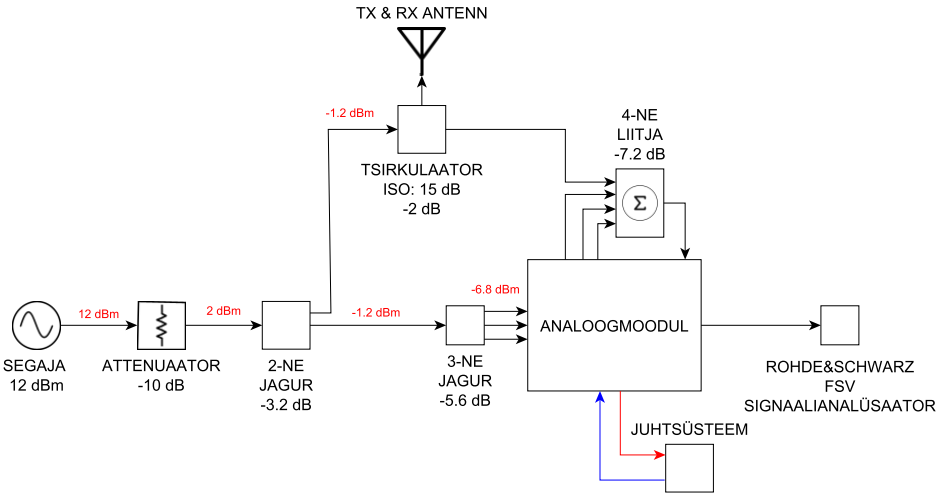
4.2 Üks antenn

4.2.1 Süsteem

Antenniga süsteemi ülesehitus oli peaaegu sama, mis terminaatoriga süsteemil. Esimeseks erinevuseks oli see, et terminaatori asemel kasutati paneelantenni. Kasutatud antenniks oli Rantelon SPA9-50N. Teiseks alandati saadetavat signaali attenuaatoriga. Antenni jõudva signaali võimsus mõõdetud Rohde & Schwarz FSV signaalianalüsaatoriga oli -3,0 dBm. Viiviskaablipikkused leiti samal meetodil, mida kirjeldati terminaatoriga süsteemi ülevaates. Kuna peegeldusi on reaalses maailmas rohkem, oli õigete viiviskaablite leidmine keerulisem. Kolm põhilist peegeldust antud süsteemis on järgmised: läbi tsirkulaatori, antennist ja esimene objekt, kus signaal tagasi peegeldub. Läbi tsirkulaatori peegeldus on püsiv, aga antennide puhul sõltub viivis kasutatud antennist. Lisaks on väga muutuv esimene peegeldus välismaailmast. See võib tulla maapinnalt, hoonetelt või puudelt. Antud mõõtmisi tehti õues, kus antenn oli suunatud paralleelselt maaga, seega esimene peegeldus võis tulla nii ees seisvatelt puudelt kui ka maapinnalt. Koos peegeldustega oli passiivne isoltasioon saadetud signaali ja vastuvõetud signaali vahel 15 dB. Kogu süsteemi skeem on näha jooniselt **4.5**.

Saadetava signaali väljundvõimsust piirati, et kaitsta ADC sisendeid. Kasutatud ADC maksimaalne sisendpinge on 2 V tipust tippu [21]. 2V tipust tippu pinge annab meile maksimaalse sisendsignaali võimsuse 10 dBm. Piirangu paneb vastuvõetava signaali võimusus, sest antud signaali võimendatakse moodulis 21 dB. See teeb vastuvõetud signaali maksimaalseks võimsuseks -21 dBm. Kuna enne katsetamist oli enesehäirete tugevus teadamata ja moodul ise võib vastuvõetavat signaali võimendada, siis kasutati turvalisuse mõttes umbes 0 dBm väljundsignaali.

Antud süsteemis sai asendatud ka LO signaali generaator. Varasem Agilent N5181A asenda-



Joonis 4.5: Ühe antenniga testimise süsteemi skeem. Joonisel on näha saadetud signaali kaod igas komponendis. Lisaks on punaselt välja tootud saadetud signaali võimsused peale igat komponenti. Punase noolega on ADCle saadetavad iga järgu IQ signaalid. Sinise noolega on DACi väljundiks olevad vektormodulaatori kontrollsignaalid.

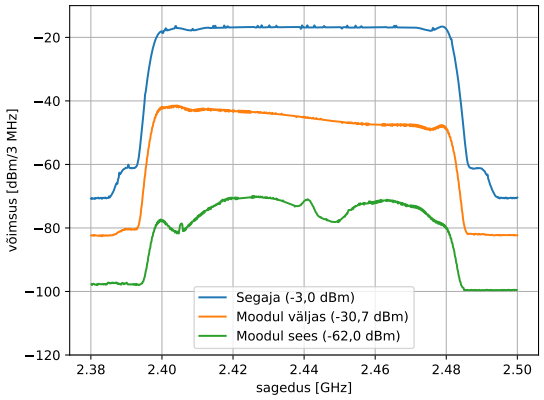
ti Analog Devicesi EVAL-ADF4351 arendusplaadiga, millel on ADF4351 signaaligeneraator. Antud plaadi väljundsignaali sagedust juhiti arvutiga. Täiendavalt ühendati EVAL-ADF4351 ja neljase jaguri vahele 2,40 kuni 2,48 GHz ribapääsfilter, et eemaldata signaaligeneraatori harmoonikud.

4.2.2 Tulemused

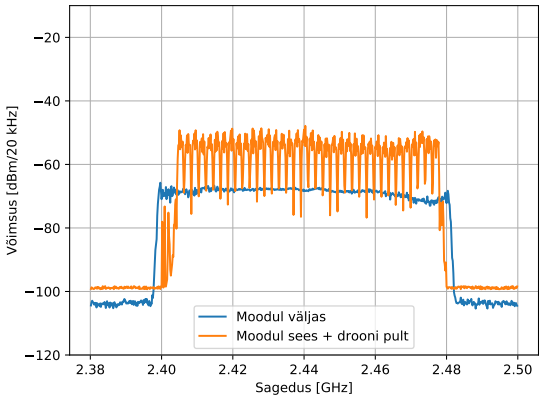
Antud süsteemis olid vastuvõetud signaali tugevus, kui analoogmoodul oli väljas -30,7 dBm. Kuna saadetav signaal oli -3,0 dBm, siis saavutatud isolatsioon saadetud ja vastuvõetud signaali vahel oli 27,7 dB. Mooduli sisselülitamisel langes vastuvõetud signaalitugevus võimsuseni -62,0 dBm. See tähendab, et moodul suutis täiendavalt eemaldada 31,3 dB peegeldusi vastuvõetud signaalist. Kokku saavutati 59,0 dB peegelduste eemaldamist. Tulemusi on näha joonisel **4.6a**.

Antud segamissignaali võimsuse ja tsirkulaatoriga saavutatud isolatsiooni juures oli drooni pulti juba ilma moodulita näha. Kui moodul sisse lülitada, olid drooni puldi tekitatud signaalid palju paremini loetavad. Peab arvestama, et katsel oli drooni pult antennist 5 meetri kaugusel otse selle ees. Jooniselt **4.6b** on näha signaali, kui moodul oli väljas ning kui moodul oli sees koos drooni puldi signaaliga.

Kasutatud segaja signaalitugevus ei ole effektiivne droonide tõrjumiseks. Selleks tuleb tõsta segaja väljundvõimsust umbes 20 dBm võimsuseni [13]. Segamissignaali tugevuse tõstmiseks on vaja tõsta isolatsiooni saadetava signaali ja vastuvõetava signaali vahel. Üheks võimaluseks on paremini isoleeritud tsirkulaator, aga see ei eemalda antennist tagasipeegeldavat signaali. Tekkis vajadus leida moodus isolatsiooni tõstmiseks.



(a) Saadetud signaal, vastuvõetud signaal, kui analoogmoodul on väljas ja vastuvõetud signaal kui moodul sisse lülitatakse. Joonise legendis on sulgudes võimsuse mõõtmise tulemus.

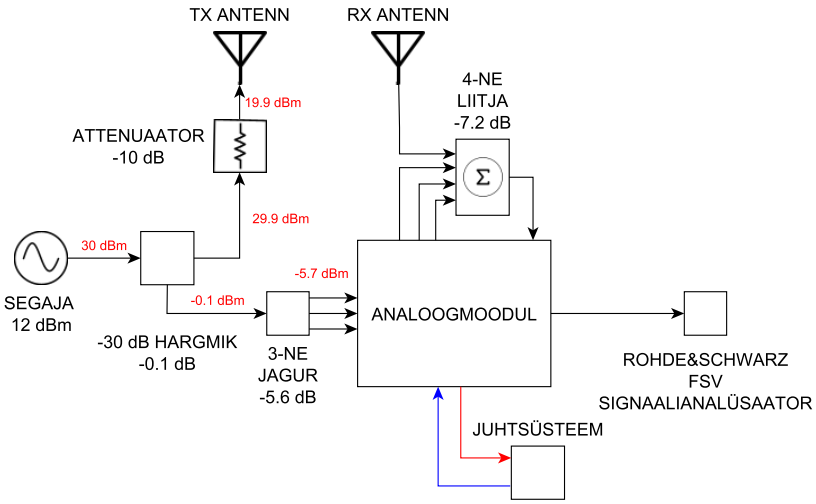


(b) Süsteemi väljund, kui moodul oli väljas ja kui analoogmoodul oli sees ning drooni pult oli sees

Joonis 4.6: Ühe antenniga süsteemi tulemused.

4.3 Kaks antenni

4.3.1 Süsteem

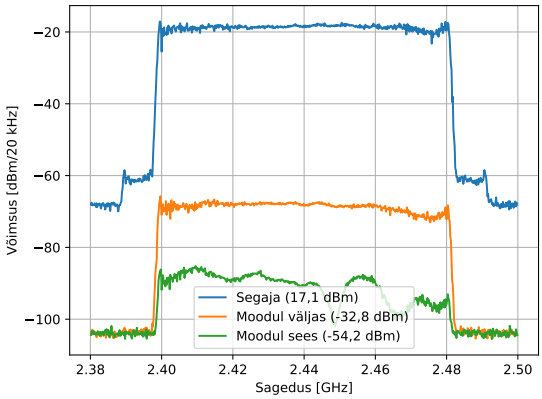


Joonis 4.7: Kahe antenniga süsteemi skeem.

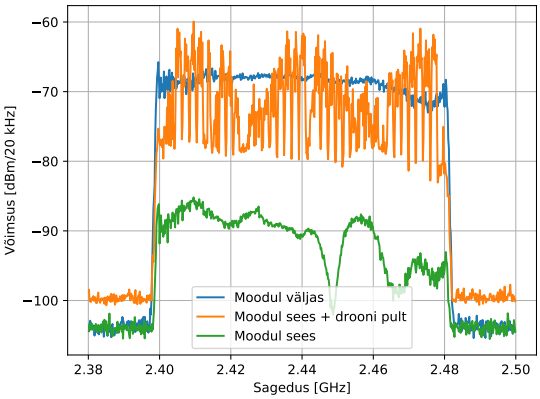
Üks viis, kuidas tõsta isolatsiooni saadetava ja vastuvõetud signaali vahel, on kasutada tsirkulaatori asemel kahte erinevat antenni. Sellisel juhul kaob ära tsirkulaatori läbilase ja antennist tagasi peegeldav signaal. Antud süsteemis on põhiliseks peegelduseks ühest antennist otse teise kostuv signaal. Järgmised peegeldused tulevad ümbritsevast keskkonnast. Testitud süsteemis suunati antennid taevasse, et vähendada keskkonnast tulevaid peegeldusi. Saatmiseks ja vastuvõtmiseks kasutati kahte Ranteloni SPA9-50N antenni.

Ka ülejäänud süsteemis tehti muudatusi, nagu võimsama segaja kasutamine. Kasutatava segaja väljundvõimsus oli umbes 30 dBm. Lisaks asendati kahene jagur, mis jagas signaali antenni ja mooduli vahel, -30 dB hargmikuga, et moodulisse minev signaalitugevus jääks alla mooduli ja ADC piiridele. Täiendavalt kasutati enne antenni 10 dB atenuaatori eesmärgiga võrdsustada saadetavat signaali droonide ja nende pultide kasutatava signaalitugevusega, mis on samaaegselt ka maksimaalne lubatud kiirgamisvõimsus antud sagedustel Euroopa Liidus [34]. Antenni jõudava signaali tugevus oli 17,1 dBm, mis mõõdeti Rohde & Schwarz FSV signaalianalüsaatoriga. Antud signaalitugevus on efektiivne droonide tõrjumisel [13].

4.3.2 Tulemused



(a) Saadetud signaal, vastuvõetud signaal, kui analoogmoodul on väljas ja vastuvõetud signaal kui moodul sisse lülitatakse. Joonise legendis on sulgudes võimsuse mõõtmise tulemus.



(b) Süsteemi väljund, kui analoogmoodul oli väljas, kui moodul oli sees ning drooni pult oli sees ja kui moodul oli sees.

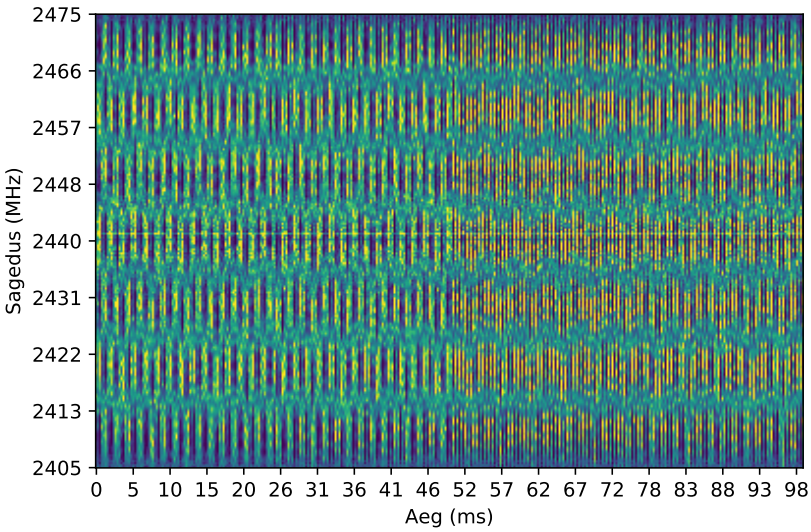
Joonis 4.8: Kahe antenniga süsteemi tulemused.

Kui analoogmoodul oli väljas, saavutati kahe antenniga süsteemis vastuvõetud signaalitugevuseks -33 dBm. Kuna saadetud signaali võimsuseks mõõdeti 17 dBm, oli kahe antenni vahel isolatsioon 50 dB. Kui moodul lülitati sisse, langes signaal täiendavad 21 dB -54 dBm võimsuseni. Kokku suudeti peegeldusi eemaldada 71 dB. Signaalide spektreid erinevates etappides on näha jooniselt **4.8a**.

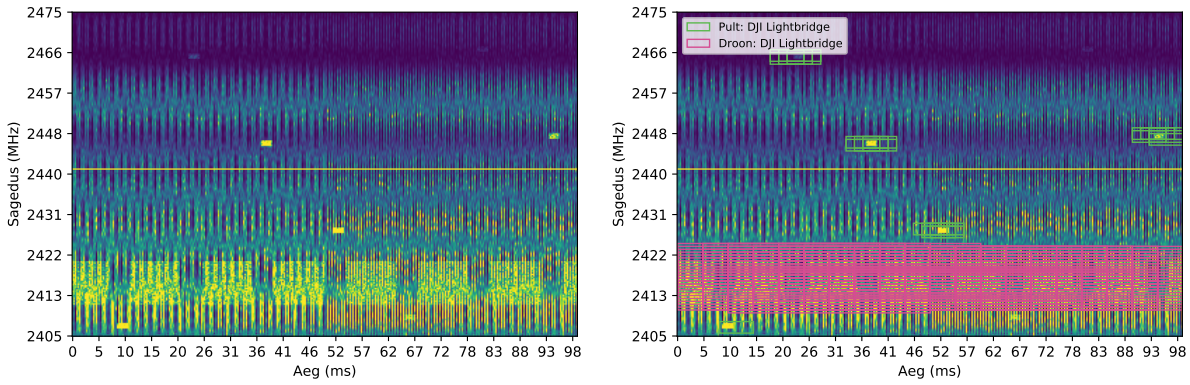
Lisaks katsetati, kas süsteemiga on spektril näha droonipuldi signaali. Kuna otse ette ei saanud pulti paigutada, sest antenn oli suunatud taevasse, asetati pult horisontaalselt antennist viie meetri kaugusele. Kuna vastuvõtva antenni SPAP9-50N efektiivne nurk oli 50°, langes antenni jõudva signaali tugevus. Kuigi ilma moodulit sisselülitamata oli puldi signaali spektrogrammil juba näha, siis peale mooduli sisselülitamist muutus puldi signaal selgelt nähtavaks. Drooni puldiga tehtud katsete spektrogramme on näha joonisel **4.8b**.

4.3.3 Droonituvastus

Kahe antenniga süsteemi saavutatud saadetava signaali tugevus oli piisav, et droone segada. Selle tulemusel ühendati analoogmooduli väljund Ranteloni droonituvastaja DTS-2458 külge. Droonituvastajaga salvestati spektrogrammi 100 ms lõikudena. Seejärel suunati salvestatud



Joonis 4.9: Segaja signaal spektrogrammil, kui analoogmoodul on väljalülitatud.



(a) Tuvastaja väljas.

(b) Tuvastaja sees.

Joonis 4.10: Drooni ja puldi signaal koos segaja signaaliga, kui analoogmoodul on sees.

andmed droonituvastaja tehisnärvivõrku. Kõigepealt katsetati süsteemi, kui moodul oli välja lülitatud. Joonis **4.9** näitab, et droonituvastaja ei näe antud spektrogrammis ühtegi äratuntavat signaali. Joonisel **4.10** on näha spektrogrammi pilti, kui moodul on sees ja droonituvastaja tehisnärvivõrgu väljundit. Jooniselt on inimsilmaga näha drooni ja puldi andmepakette. Ka tehisnärvivõrk suudab tuvastada vastavad mustrid ja näidata, et spektrogrammil on drooni ja puldi signaal. Spektrogrammi salvestamise hetkel oli droon 20m kõrgusel antenni kohal ja pult 5 meetri kaugusel antenni kõrval. Kasutatud drooniks oli DJI Phantom 4, mis kasutab suhtluseks DJI Lightbridge protokolli.

4.4 Analüüs

Varasemalt on tehtud palju süsteeme, mis kasutab ühte antenni. 2016. aastal Joose Tammineni Tampere Tehnikaülikoolis kaitsud magistritöös [18] katsetatakse analoog lahutajat paljude erinevate ribalaiustega signaalidega 2,46 GHz sageduse juures. 80 MHz ribalaiusega signaaliga tsirkulaatoriga saavutati isolatsiooni 22,2 dB ja analoogse lahutajaga täiendavad 41,0 dB lahu-

tamist. Kokku suudeti peegeldusi eemaldada 63,2 dB. Kahjuks pole öeldud, mis signaali saadeti testimiseks, sest analoogse lahutaja töö efektiivsus väga sõltub kasutatavast signaalist.

Kui võrrelda tulemusi teiste kahe antenni süsteemidega, siis 2017. aasta veebruaris ilmus artikkel [17], kus katsetati süsteemi kajavabas kambris. Antud süsteem kasutas erinevatel ribalaiustel 2,56 GHz sagedusel mobiilside signaali. 80 MHz ribalaiusel signaaliga saavutati kajavabas kambris antennidega 75,6 dB isolatsiooni. Analooglahutajaga saavutati täiendavad 7 dB lahutamist. Tubases keskonnas saavutati 80 MHz signaaliga kahe antenni vahel 63,5 dB isolatsiooni. Selles keskkonnas analoogset lahutajat kahjuks ei katsetatud. Õues mõõdeti ainult vektor võrguanalüsaatoriga, kuid kogu süsteemi ei katsetatud. Kajavabas kambris saavutati kokku 78,6 dB peegelduste eemaldust. Põhilised erinevused käesoleva töö süsteemiga on kasutatud sihtotstarbeline antenni süsteem ja kasutatud mobiilside signaal.

Saavutatud signaalitugevused iga süsteemiga, kui moodul oli sisselülitatud, jäid piisavalt madalale, et neid täiendavalt parandada digitaalsete enesehäireid eemaldavate süsteemidega. Näiteks kahe antenni süsteemi juures, kus saadetav signaal oli kõige tugevam, jäi mooduli väljund 45,8 dB kõrgemale -100 dBm müratasemest. Samas oli saavutatud ensesehäirete eemaldamine juba piisav, et vastuvõetud signaalist tuvastada droone ja nende pulte.

Kokkuvõte

Käesolevas töös ühendati droonide tuvastamine ja segamine ühte süsteemi. Selleks rakendati IBFD raadio põhimõtteid. IBFD raadio realiseerimiseks tutvustati selle eeliseid ja puuduseid. Lisaks kirjeldati Rantelon OÜs arendatud analoogmoodulit, mis teeb IBFD raadios enesehäireid eemaldada. Antud töö raames valiti selle mooduli kontrollimiseks riistvara ja realiseeriti juhtimistarkvara FPGAl. Seejärel katsetati kogu süsteemi kolmel erineval meetodil ja erinevate segava signaali tugevustega, et saada aimu selle efektiivsusest. Lõpuks ühendati segaja ja tuvastaja loodud süsteemi abil ning näidati, et antud süsteemi saab kasutada nende samaaegse töö võimaldamiseks.

Töös kasutatud süsteemil on väga palju edasiarenduse suundi. Esiteks oleks vaja parandada kasutajamugavust, sest loodud süsteem oli kobakas ning ülesseadmine tülikas. Selle tarbeks oleks vaja projekteerida FPGAd, DACi ja ADCd ühendav trükkplaat, et praegu kasutatavad arendusplaadid ära kaotada. Teiseks tuleks lihtsustada ja automatiseerida õigete viiviskaablite leidmine, sest sellele kulub palju aega. Kolmandaks võiks süsteemi juhtimine olla samas programmis, mis näitab reaalajas filtri väljundeid. See võimaldaks ka süsteemi ühendamist arvutiga vaid ühe kaabli abil. Neljandaks tuleks parandada ka süsteemi väljundväärtuse leidmise kiirust, et saaks kasutada DTS-2458 suunamääramise funktsionaalsust. Kogu süsteemile tuleks arendada ja liita ka digitaalne enesehäirete eemaldamise süsteem, et parandada saavutatavat lahutust veelgi.

Tänuavaldused

Tänan juhendajaid Jaanus Kaldet ja Karel Pärlinit antud nõuannete ja tarkuse eest.

Tänan Ranteloni ja selle meeskonda võimaluse eest teha oma magistritööd nende seltskonnas ja töö käigus saadud nõu eest.

Tänan Kaitseministeeriumit antud toetuse eest.

Viited

[1] B. I. Intelligence. (veebruar 2020). Commercial Unmanned Aerial Vehicle (UAV) Market Analysis – Industry trends, forecasts and companies, [Võrgumaterjal]. Saadaval: https://www.businessinsider.com/commercial-uav-market-analysisy. (vaadatud: 28.04.2020).

[2] M. Matiisen, "Mehitamata lennuvahenditega seotud siseturvalisuse alased ohud ja nende maandamine muutuvas keskkonnas", magistritöö, Sisekaitseakadeemia, 2019.

[3] BBC. (detsember 2018). Gatwick Airport: Drones ground flights, [Võrgumaterjal]. Saadaval: https://www.bbc.com/news/uk-england-sussex-46623754. (vaadatud: 28.04.2020).

[4] T. Multerer, A. Ganis, U. Prechtel, E. Miralles, A. Meusling, J. Mietzner, M. Vossiek, M. Loghi ja V. Ziegler, "Low-cost jamming system against small drones using a 3D MIMO radar based tracking", teoses *2017 European Radar Conference (EURAD)*, 2017, lk. 299–302.

[5] A. Sabharwal, P. Schniter, D. Guo, D. W. Bliss, S. Rangarajan ja R. Wichman, "In-Band Full-Duplex Wireless: Challenges and Opportunities", *IEEE Journal on Selected Areas in Communications*, köide 32, nr 9, lk. 1637–1652, 2014.

[6] S. Rahman ja D. A. Robertson, "Radar micro-Doppler signatures of drones and birds at K-band and W-band", *Scientific Reports*, köide 8, nr 1, november 2018.

[7] R. R. Systems. (märts 2019). 9 Counter-Drone Technologies To Detect And Stop Drones Today, [Võrgumaterjal]. Saadaval: https://www.robinradar.com/press/blog/9-counter-drone-technologies-to-detect-and-stop-drones-today. (vaadatud: 28.04.2020).

[8] *Frequency bands designated for Industrial, Scientific and Medical use (ISM)*, Ofcom, juuni 2017.

[9] DJI. (2019). Mavic 2, [Võrgumaterjal]. Saadaval: https://www.dji.com/ee/mavic-2/info. (vaadatud: 14.05.2020).

[10] *802.11ac: The Fifth Generation of Wi-Fi*, White paper, Cisco, 2018.

[11] K. Pärlin, T. Riihonen, G. Karm ja M. Turunen, "Jamming and Classification of Drones Using Full-Duplex Radios and Deep Learning", Avalikustamisel, 2020.

[12] *Drone Detection System*, DTS-2458, Rantelon, november 2019.

[13] K. Pärlin, M. M. Alam ja Y. Le Moullec, "Jamming of UAV remote control systems using software defined radio", teoses *2018 International Conference on Military Communications and Information Systems (ICMCIS)*, 2018.

[14] K. Pärlin, T. Riihonen ja M. Turunen, “Sweep Jamming Mitigation Using Adaptive Filtering for Detecting Frequency Agile Systems”, teoses *2019 International Conference on Military Communications and Information Systems (ICMCIS)*, 2019, lk. 1–6.

[15] Y. Choi ja H. Shirani-Mehr, “Simultaneous Transmission and Reception: Algorithm, Design and System Level Performance”, *IEEE Transactions on Wireless Communications*, köide 12, nr 12, lk. 5992–6010, 2013.

[16] D. Bharadia, E. McMilin ja S. Katti, “Full duplex radios”, teoses *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*, 2013, lk. 375–386.

[17] D. Korpi, M. Heino, C. Icheln, K. Haneda ja M. Valkama, “Compact Inband Full-Duplex Relays With Beyond 100 dB Self-Interference Suppression: Enabling Techniques and Field Measurements”, *IEEE Transactions on Antennas and Propagation*, köide 65, nr 2, lk. 960–965, 2017.

[18] J. Tamminen, “Digital Control of RF Self-Interference Canceller in Full-Duplex Radio”, magistritöö, Tampere University of Technology, 2016.

[19] M. Adrat, R. Keller, M. Tschauner, S. Wilden, V. Le Nir, T. Riihonen, M. Bowyer ja K. Pärlin, “Full-Duplex Radio – Increasing the Spectral Efficiency for Military Applications”, teoses *2019 International Conference on Military Communications and Information Systems (ICMCIS)*, 2019, lk. 1–5.

[20] *GaAs HBT VECTOR MODULATOR 1.8 - 2.7 GHz*, HMC631LP3 / 631LP3E, v00.1007, Analog Devices.

[21] *Octal, 14-Bit, 125 MSPS, Serial LVDS, 1.8 V Analog-to-Digital Converter*, AD9681, Rev. C, Analog Devices, 2015.

[22] Dougl. (oktoober 2018). EVALUATING THE AD9681 ANALOG-TO-DIGITAL CONVERTER, [Võrgumaterjal]. Saadaval: `https://wiki.analog.com/resources/eval/ad9681-125ebz`. (vaadatud: 16.04.2020).

[23] *Octal, 12-/16-Bit, I2C, denseDACs with 5 ppm/C On-Chip Reference*, AD5629R/AD5669R, Rev. F, Analog Devices, 2018.

[24] *EVAL-AD5629RSDZ/EVAL-AD5669RSDZ User Guide*, UG-867, Rev. 0, Analog Devices, 2015.

[25] Digilent. (juuli 2019). Nexys Video Reference Manual, [Võrgumaterjal]. Saadaval: `https://reference.digilentinc.com/reference/programmable-logic/nexys-video/reference-manual`. (vaadatud: 20.04.2020).

[26] B. Widrow, J. R. Glover, J. M. McCool, J. Kaunitz, C. S. Williams, R. H. Hearn, J. R. Zeidler, J. Eugene Dong ja R. C. Goodlin, “Adaptive noise cancelling: Principles and applications”, *Proceedings of the IEEE*, köide 63, nr 12, lk. 1692–1716, 1975.

[27] B. Widrow ja J. McCool, “A comparison of adaptive algorithms based on the methods of steepest descent and random search”, *IEEE Transactions on Antennas and Propagation*, köide 24, nr 5, lk. 615–637, 1976.

[28] B. Widrow, J. M. McCool, M. G. Larimore ja C. R. Johnson, “Stationary and nonstationary learning characteristics of the LMS adaptive filter”, *Proceedings of the IEEE*, köide 64, nr 8, lk. 1151–1162, 1976.

[29] *Vivado Design Suite User Guide - High-Level Synthesis*, UG902, v2018.3, Xilinx, 2018.

[30] S. Larson. (mai 2019). Serial Peripheral Interface (SPI) Master (VHDL), [Võrgumaterjal]. Saadaval: https://www.digikey.com/eewiki/pages/viewpage.action?pageId=4096096. (vaadatud: 26.08.2019).

[31] *Integrated Logic Analyzer - LogiCORE IP Product Guide*, PG172, v6.2, Xilinx, 2016.

[32] S. Larson. (juuli 2019). I2C Master (VHDL), [Võrgumaterjal]. Saadaval: https://www.digikey.com/eewiki/pages/viewpage.action?pageId=10125324. (vaadatud: 26.08.2019).

[33] *Digitally Removing a DC Offset: DSP Without Mathematics*, WP279, v1.0, Xilinx, 2008.

[34] E. Treiman, "Mehitamata õhusõidukite juhtsignaalide analüüs", Eesti Lennuakadeemia, 2018.

Lisad

Lisa 1: ADC saadetava väärtuste paralleliseerimise VHDL kood

```
library IEEE;
    use IEEE.STD_LOGIC_1164.ALL;
    use IEEE.NUMERIC_STD.ALL;
library UNISIM;
    use UNISIM.VComponents.all;

entity Adc_Channel_Serdes is
    Generic (
        C_Wires         : integer := 2;
        C_DataWidth     : integer := 16;
        C_IoDelayGroup  : string := "idelay_group"
    );
    Port (
        clk         : in std_logic; --200MHz
        test_mode   : in std_logic;
        test_done   : out std_logic;

        adc_fc      : in std_logic;
        adc_dc      : in std_logic;
        reset       : in std_logic;
        serial      : in std_logic_vector (1 downto 0);

        data        : out std_logic_vector (C_DataWidth-1 downto 0)
    );
end Adc_Channel_Serdes;

architecture Adc_Channel_Serdes_Struct of Adc_Channel_Serdes is

function bits_set(v : std_logic_vector) return natural is
  variable n : natural := 0;
begin
    for i in v'range loop
        if v(i) = '1' then
            n := n + 1;
        end if;
    end loop;
    return n;
end function bits_set;

signal clkb     : std_logic;
signal adc_data : std_logic_vector (C_DataWidth-1 downto 0);

signal delayed          : std_logic_vector (C_Wires-1 downto 0);
signal delay_count_0    : std_logic_vector(4 downto 0);
```

```
signal delay_count_1    : std_logic_vector(4 downto 0);
signal delay_ce         : std_logic_vector (C_Wires-1 downto 0);
signal bitslip          : std_logic_vector (C_Wires-1 downto 0);
signal proc_bitslip     : std_logic_vector (C_Wires-1 downto 0);
signal error_seen       : std_logic_vector (C_Wires-1 downto 0) := (
   others => '0');

signal nib_done         : std_logic_vector (C_Wires-1 downto 0);

signal holdoff_0        : unsigned(15 downto 0)  := (others => '0');
signal holdoff_1        : unsigned(15 downto 0)  := (others => '0');

signal debug_bitslip_inc    : std_logic_vector (C_Wires-1 downto 0);
signal debug_bitslip_inc_d  : std_logic_vector (C_Wires-1 downto 0);

attribute IODELAY_GROUP : STRING;
attribute IODELAY_GROUP of Adc_Channel_Idelaye2_01: label is
   C_IoDelayGroup;
attribute IODELAY_GROUP of Adc_Channel_Idelaye2_02: label is
   C_IoDelayGroup;

begin
process (adc_fc)
begin
    if rising_edge(adc_fc) then
        bitslip <= proc_bitslip;
    end if;
end process;
test_done <= and nib_done;

clkb <= not adc_dc;
Adc_Channel_Vio : vio_0
    PORT MAP (
        clk => clk,
        probe_in0 => nib_done,
        probe_out0 => delay_count_0,
        probe_out1 => delay_count_1,
        probe_out2 => debug_bitslip_inc(0 downto 0),
        probe_out3 => debug_bitslip_inc(1 downto 1)
    );

Adc_Channel_Idelaye2_01 : IDELAYE2
    generic map (
        CINVCTRL_SEL          => "FALSE",
        DELAY_SRC             => "IDATAIN",
        HIGH_PERFORMANCE_MODE => "TRUE",
        IDELAY_TYPE           => "VAR_LOAD",
        IDELAY_VALUE          => 0,
        PIPE_SEL              => "FALSE",
        REFCLK_FREQUENCY      => 200.0,
        SIGNAL_PATTERN        => "DATA"
    )
    port map (
        DATAIN      => '0',
        IDATAIN     => serial(0),
        DATAOUT     => delayed(0),
        --
        CNTVALUEOUT => open,
        C           => adc_dc,
```

```
        CE          => delay_ce(0),
        CINVCTRL    => '0',
        CNTVALUEIN  => delay_count_0,
        INC         => '0',
        LD          => '1',
        LDPIPEEN    => '0',
        REGRST      => '0'
    );

Adc_Channel_Idelaye2_02 : IDELAYE2
    generic map (
        CINVCTRL_SEL          => "FALSE",
        DELAY_SRC             => "IDATAIN",
        HIGH_PERFORMANCE_MODE => "TRUE",
        IDELAY_TYPE           => "VAR_LOAD",
        IDELAY_VALUE          => 0,
        PIPE_SEL              => "FALSE",
        REFCLK_FREQUENCY      => 200.0,
        SIGNAL_PATTERN        => "DATA"
    )
    port map (
        DATAIN      => '0',
        IDATAIN     => serial(1),
        DATAOUT     => delayed(1),
        --
        CNTVALUEOUT => open,
        C           => adc_dc,
        CE          => delay_ce(1),
        CINVCTRL    => '0',
        CNTVALUEIN  => delay_count_1,
        INC         => '0',
        LD          => '1',
        LDPIPEEN    => '0',
        REGRST      => '0'
    );

Adc_Channel_Iserdes_01 : ISERDESE2
    generic map (
        DATA_RATE         => "DDR",
        DATA_WIDTH        => 8,
        DYN_CLKDIV_INV_EN => "FALSE",
        DYN_CLK_INV_EN    => "FALSE",
        INIT_Q1 => '0', INIT_Q2 => '0', INIT_Q3 => '0', INIT_Q4 => '0',
        INTERFACE_TYPE    => "NETWORKING",
        IOBDELAY          => "BOTH",
        NUM_CE            => 1,
        OFB_USED          => "FALSE",
        SERDES_MODE       => "MASTER",
        SRVAL_Q1 => '0', SRVAL_Q2 => '0', SRVAL_Q3 => '0', SRVAL_Q4 =>
            '0'
   )
    port map (
        O => open,
        Q1 => adc_data(0), Q2 => adc_data(1), Q3 => adc_data(2), Q4 =>
            adc_data(3),
        Q5 => adc_data(4), Q6 => adc_data(5), Q7 => adc_data(6), Q8 =>
            adc_data(7),
        SHIFTOUT1 => open,
        SHIFTOUT2 => open,
```

```
        BITSLIP   => bitslip(0),
        CE1 => '1',
        CE2 => '1',
        CLKDIVP      => '0',
        CLK          => adc_dc,
        CLKB         => clkb,
        CLKDIV       => adc_fc,
        OCLK         => '0',
        DYNCLKDIVSEL => '0',
        DYNCLKSEL    => '0',
        D            => '0',
        DDLY         => delayed(0),
        OFB          => '0',
        OCLKB        => '0',
        RST          => reset,
        SHIFTIN1     => '0',
        SHIFTIN2     => '0'
    );

Adc_Channel_Iserdes_02 : ISERDESE2
    generic map (
        DATA_RATE         => "DDR",
        DATA_WIDTH        => 8,
        DYN_CLKDIV_INV_EN => "FALSE",
        DYN_CLK_INV_EN    => "FALSE",
        INIT_Q1 => '0', INIT_Q2 => '0', INIT_Q3 => '0', INIT_Q4 => '0',
        INTERFACE_TYPE    => "NETWORKING",
        IOBDELAY          => "BOTH",
        NUM_CE            => 1,
        OFB_USED          => "FALSE",
        SERDES_MODE       => "MASTER",
        SRVAL_Q1 => '0', SRVAL_Q2 => '0', SRVAL_Q3 => '0', SRVAL_Q4 =>
            '0'
    )
    port map (
        O => open,
        Q1 => adc_data(8), Q2 => adc_data(9), Q3 => adc_data(10), Q4 =>
            adc_data(11),
        Q5 => adc_data(12), Q6 => adc_data(13), Q7 => adc_data(14), Q8
            => adc_data(15),
        SHIFTOUT1 => open,
        SHIFTOUT2 => open,
        BITSLIP   => bitslip(1),
        CE1 => '1',
        CE2 => '1',
        CLKDIVP      => '0',
        CLK          => adc_dc,
        CLKB         => clkb,
        CLKDIV       => adc_fc,
        OCLK         => '0',
        DYNCLKDIVSEL => '0',
        DYNCLKSEL    => '0',
        D            => '0',
        DDLY         => delayed(1),
        OFB          => '0',
        OCLKB        => '0',
        RST          => reset,
        SHIFTIN1     => '0',
        SHIFTIN2     => '0'
```

```
   );
data <= adc_data; -- Output adc data

-------------------------------------------------------
-- Bitslip detect
-------------------------------------------------------
detect_delay_process: process(adc_fc, reset)
    begin
        if reset = '1' then
            holdoff_0 <= (others => '1');
            delay_ce(1) <= '0';
            proc_bitslip <= (others => '0');
            nib_done <= (others => '0');
        elsif rising_edge(adc_fc) then
            ----------------------------------
            -- See if an error has been seen.
            -- Holdoff gives a few cycles for bitslips and
            -- delay changes to take effect.
            ----------------------------------
            if test_mode = '1' then
                if holdoff_1 = 0 and nib_done(1) = '0' then
                    if adc_data(15 downto 8) /= "10110000" then
                        error_seen(1) <= '1';
                    else
                        nib_done(1) <= '1';
                    end if;
                else
                    holdoff_1 <= holdoff_1-1;
                end if;
                if holdoff_0 = 0 and nib_done(0) = '0' then
                    if adc_data(7 downto 0) /= "10110000" then
                        error_seen(0) <= '1';
                    else
                        nib_done(0) <= '1';
                    end if;
                else
                    holdoff_0 <= holdoff_0-1;
                end if;
            end if;
            proc_bitslip <= (others => '0');
            ------------------------------------
            -- Bitslip if required in a adc input
            ------------------------------------
            if error_seen(0) = '1' then
                error_seen(0) <= '0';
                holdoff_0 <= (others => '1');

                proc_bitslip(0) <= '1';
            end if;

            if error_seen(1) = '1' then
                error_seen(1) <= '0';
                holdoff_1 <= (others => '1');
                proc_bitslip(1) <= '1';
            end if;
        end if;
    end process;

end Adc_Channel_Serdes_Struct;
```

Lisa 2: PN9 arvujada genereerimise ja võrdlemise programm

```
import csv

start = 0x1fe0 # algusväärtus
a = start

pn = []
j = 0
while 1:
    j += 1
    # Arvutatakse uus bit ja listakase see eelmisele väärtusele
    # x^9 + x^5 + 1
    newbit = ((a >> 8) ^ (a >> 4)) & 1
    a = ((a << 1) + newbit) & 0x3fff
    # Võetakse iga 14 väärtus, sest ADC on 14-bitine
    if j%14 == 0:
        pn.append((a<<2)^0x8000)
        # Lõpetatakse, kui jada hakkab korduma
        if (a<<2)^0x8000 == 0x7f80:
            break

# Loetakse sisse ADClt salvestatud tulemused
adc_data = []
with open('adc_data.csv', 'r') as csvFile:
    reader = csv.reader(csvFile)
    for row in reader:
        adc_data.append(row[3])
csvFile.close()

# Kontrollitakse, kas on erinevuseid
# genereeritud ja ADC saadetud väärtustel
errors = []
for i, v in enumerate(ila):
    h = "{0:#0{1}x}".format(pn[i%len(pn)],6)
    if adc_data[i] not in h:
        errors.append((i, adc_data[i], h))

# Trükitakse välja vea suhe ja esimesed kümme viga
print(len(errors)/len(adc_data))
print(errors[:10])
```

Lisa 3: Alalisvoolu nihke eemaldamise VHDL kood

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity Dc_Removal is
    Generic (
        C_DataWidth : integer := 16
    );
    Port (
        clk     : in std_logic;
        rst     : in std_logic;
        enabled : in std_logic;

        i_data  : in  std_logic_vector(C_DataWidth-1 downto 0);
        o_data  : out std_logic_vector(C_DataWidth-1 downto 0)
    );
end Dc_Removal;

architecture Dc_Removal_Struct of Dc_Removal is

signal data_in  : signed(C_DataWidth downto 0);
signal offset_x2 : signed(2*C_DataWidth downto 0) := (others => '0');

begin
data_in <= RESIZE(signed(i_data), C_DataWidth+1);

process(rst, clk)
    variable offset : signed(C_DataWidth downto 0);

    variable cor_data_x2    : signed(2*C_DataWidth-1 downto 0) := (
       others => '0');
    variable cor_data       : signed(C_DataWidth downto 0);
begin
    if rst = '1' then
        offset_x2 <= (others => '0');
        o_data <= i_data;
    elsif rising_edge(clk) then
        if enabled = '0' then
            o_data <= i_data;
        else
            offset := offset_x2(2*C_DataWidth downto C_DataWidth);
            cor_data := data_in - offset;
            if cor_data(cor_data'high) /= cor_data(cor_data'high-1) then
                if data_in < 0 then
                    o_data <= (o_data'high => '1', others => '0');
                else
                    o_data <= (o_data'high => '0', others => '1');
                end if;
            else
                o_data <= std_logic_vector(resize(cor_data, C_DataWidth)
                   );
            end if;
            cor_data_x2 := RESIZE(cor_data, cor_data_x2'length);
            offset_x2 <= cor_data_x2 + offset_x2;
        end if;
    end if;
```

```
end process;

end Dc_Removal_Struct;
```

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Jüri Gramann

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

   "Samaaegses droonisegamises ja -tuvastuses enesehäirete eemaldamiseks kasutatava analoogmooduli juhtimine"

   mille juhendajad on Jaanus Kalde ja Karel Pärlin

   (a) reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

   (b) üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace'i kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autorõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile;

3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **20.05.2020**