Mjukvaruradiobaserad SIMO-demonstrator

MARIA ASPLUND

FOI
Totalförsvarets forskningsinstitut
Ledningssystem
Box 1165
581 11 Linköping

Tel: 013-37 80 00
Fax: 013-37 81 00

www.foi.se

FOI-R--2355--SE
ISSN 1650-1942

Teknisk rapport
November 2007

Ledningssystem

Maria Asplund

Mjukvaruradiobaserad SIMO-demonstrator

Titel	Mjukvaruradiobaserad SIMO-demonstrator
Title	Software Defined Radio: Single-Input Multiple-Output Communication Demonstrator
Rapportnr/Report no	FOI-R--2355--SE
Rapporttyp Report Type	Teknisk rapport Technical report
Månad/Month	November
Utgivningsår/Year	2007
Antal sidor/Pages	49 p
ISSN	ISSN 1650-1942
Kund/Customer	Försvarsmakten
Forskningsområde Programme area	7. Ledning med MSI 7. C4I
Delområde Subcategory	71 Ledning 71 Command, Control, Communications, Computers, Intelligence
Projektnr/Project no	E7107
Godkänd av/Approved by	Hugo Tullberg

FOI, Totalförsvarets Forskningsinstitut
Avdelningen för Ledningssystem
Box 1165
581 11 Linköping

FOI, Swedish Defence Research Agency
Command and Control Systems
SE-581 11 Linköping

Sammanfattning

Detta examensarbete har genomförts på uppdrag av Totalförsvarets Forskningsinstitut. Examensarbetet är ett av fyra delprojekt i projektet Telekommunikationer för strid i urban miljö. Projektet syftar till att ge förslag på metoder och utrustning som kan underlätta för soldaten på taktiskt nivå, i Försvarsmaktens insatsförband, genom att förbättra informationsinhämtning och -överföring. Syftet med examensarbetet är att i en praktisk demonstration visa nyttan med MIMO-system antingen genom ökad kapacitet eller genom ökad tillgänglighet.

Examensarbetet är uppdelat i tre olika delar: förstudie, utveckling av teknikdemonstrator i hårdvara och mjukvara samt demonstrationer och presentationer. Demonstratorn är byggd med GNU Radio mjukvara, som körs på två Linux-maskiner, med Universal Software Radio Peripheral som RF hårdvara.

För att visa fördelarna med MIMO har spatiell diversitet implementerats i en mottagare. Implementationen visar att bitfelshalten reduceras till en tredjedel jämfört med en SISO-mottagare.

Nyckelord: Mjukvaruradio, MIMO, USRP, GNU Radio

Summary

This master thesis has been commissioned by the Swedish Defence Research Agency and is one of four parts in the project Tactical Communications for Urban Warfare. The project aims at suggesting methods and equipment suitable for soldiers in international operations on a tactical level by improving information gathering and information transmission. The purpose of the master thesis is to by practical means demonstrate the benefits with MIMO systems, either by increasing the transmission capacity or by increasing the availability.

The master thesis is divided into three different parts: (1) feasibility study, (2) the development of a demonstrator in hardware and software, and (3) demonstrations and presentations. The demonstrator is built with GNU Radio software running on two Linux machines with a Universal Software Radio Peripheral as RF front end.

MIMO spatial diversity has been implemented in a receiver in order to demonstrate the benefits with MIMO. The experiments show that the bit error is reduced by two thirds compared to an ordinary SISO receiver.

Keywords: Software Defined Radio, MIMO, USRP, GNU Radio

Innehållsförteckning

Akronymer

Förkortning	Beskrivning
AD-omvandling	Analog till Digital omvandling
CORBA	Common Object Request Broker Architecture
DA-omvandling	Digital till Analog-omvandling
DBPSK	Differential Binary Phase Shift Keying
DDC	Digital Down Converter
DoD	Department of Defense
DQPSK	Differential Quadrature Phase Shift Keying
DSP	Digital Signal Processor/Digital Signal Processing
DUC	Digital Up Converter
EGC	Equal Gain Combining
FFT	Fast Fourier Transform
FHSS	Frequency Hopping Spread Spectrum
FM	Försvarsmakten
FOI	Totalförsvarets Forskningsinstitut
FPGA	Field Programmable Gate Array
GMSK	Gaussian Minimum Shift Keying
GNU	GNU Not Unix
GTRS	Gemensamt Taktiskt RadioSystem
GUI	Graphical User Interface
IDL	Interface Definition Language
IEEE	Institute of Electrical and Electronics Engineers
JTRS	Joint Tactical Radio System
LNA	Low Noise Amplifier
LPF	Low Pass Filter
MIMO	Multiple-Input-Multiple-Output
MISO	Multiple-Input-Single-Output
MRRC	Maximum Ratio Receive Combining
ORB	Object Request Broker

Förkortning	Beskrivning
OSSIE	Open Source SCA Implementation::Embedded
RF	Radiofrekvens
RX	Receiver
SCA	Software Communication Architecture
SDR	Software Defined Radio
SIMO	Single-Input-Multiple-Output
SISO	Single-Input-Single-Output
STBC	Space-Time Block Coding
SWIG	Simplified Wrapper and Interface Generator
TCP/IP	Transmission Control Protocol/Internet Protocol
TURBAN	Telekommunikationer för strid i urban miljö
TX	Transmitter
USB	Universal Serial Bus
USRP	Universal Software Radio Peripheral
VCO	Voltage Controlled Oscillator

1 Inledning

1.1 Bakgrund

Examensarbetet genomfördes på uppdrag av FOI (Totalförsvarets Forskningsinstitut) som är en civil myndighet under Försvarsdepartementet. Arbetet ingår som ett av fyra delprojekt i projektet TURBAN (Telekommunikationer för strid i urban miljö) under 2006-2008. Projektet syftar till att ge förslag på metoder och utrustning som kan underlätta för soldaten på taktiskt nivå, i FM:s (Försvarsmakten) insatsförband, genom att förbättra informationsinhämtning och -överföring. Fokus i projektet ligger på forskning mot kommunikationssystem för mobila och dynamiska scenarier i urban miljö. De fyra delprojekten är följande:

- Simuleringar av adaptiva kommunikationssystem i stadsmiljö.
- Studie av avancerade kommunikationsteknikers användbarhet för militära tillämpningar i stadsmiljö.
- MIMO-kanal (Multiple-Input-Multiple-Output) samt analys av MIMO-mätningar.
- Utveckling av teknikdemonstrator i hårdvara.

Den fjärde delen är den del som examensarbetet utgör och består av att utveckla en teknikdemonstrator för att visa praktisk nytta med MIMO-system. Med teknikdemonstratorutveckling menas att med mjukvara och hårdvara bygga/utveckla en mjukvaruradio som kan demonstrera de fördelar MIMO-tekniken erbjuder. Examensarbetet är alltså av praktisk karaktär.

1.2 Problemformulering

MIMO-tekniken har stora fördelar för kommunikationssystem på grund av den höga kapacitet och tillgänglighet tekniken kan erbjuda. Eftersom MIMO är ett relativt brett begrepp så uppstår frågan vad som här menas med en MIMO-demonstrator. Tre olika områden vad gäller MIMO-teknik har identifierats; diversitet, spatiell multiplexing samt lobformning (se kapitel 4). Varje område har olika fördelar och nackdelar vad gäller implementation och demonstrationslämplighet. Vilken typ av MIMO lämpar sig bäst att implementera i demonstratorn?

För att utveckla demonstratorn används en dator som sändare och en annan som mottagare. Dessa datorer kör GNU Radio (GNU Not Unix) programvara och använder USRP (Universal Software Radio Peripheral) hårdvara som RF- (radiofrekvens) hårdvara (se kapitel 3). För att kunna utveckla demonstratorn krävs kännedom om GNU Radio och USRP. Hur är programpaketet uppbyggt och vilka moduler finns att använda?

För att få svar på den frågan och för att verifiera att all utrustning fungerar är första steget att implementera en SISO-länk (Single-Input-Single-Output) som överför någon typ av information från den ena datorn till den andra. SISO-länken är den del som senare kommer att användas som jämförelse med MIMO-länken för att påvisa de fördelar MIMO har. Ett av kraven från projektledningen är att överföringen ska ske med någon form av digital modulation. Finns det stöd för digital modulation i GNU Radio?

Om det inte finns stöd för digital modulation i GNU Radio måste denna del först utvecklas innan nästa steg kan tas. Om så är fallet kommer arbetet först utgå från någon form av analog modulation för att sedan ta steget till digital modulation.

Om tidsåtgången för ovanstående uppgifter visar sig vara mindre än planerat finns alternativ till vidare arbete. En möjlighet vore till exempel att undersöka om GNU Radio är kompatibel med SCA (Software Communication Architecture) och vad som egentligen menas med SCA-kompabilitet. Troligt är att GNU Radion inte är det varför det vore intressant att undersöka vad som krävs för att kombinera de båda.

1.3 Syfte och frågeställningar

Syftet med examensarbetet är att undersöka om det är möjligt, och i så fall hur, att med GNU Radio bygga ett MIMO-system och sedan i en, eller flera, praktiska demonstrationer visa nyttan med systemet i form av ökad kapacitet eller minskad störkänslighet. Syftet är också att undersöka modulariteten hos GNU Radio och dokumentera de lärdomar som fås genom projektet. Därför ställs följande frågor:

- Hur ser GNU Radio-strukturen ut, det vill säga hur är GNU Radio uppbyggt?
- Är det möjligt att implementera ett MIMO-system som använder digital modulation med GNU Radio, och i så fall, vilken MIMO-teknik lämpar sig bäst att implementera?

- Om ja, vilka vinster kan man göra med MIMO och mjukvarudefinierad radio jämfört med SISO under ett visst givet scenario och hur stora är de?

1.4 Avgränsningar

En av de avgränsningar som gjorts var att välja vilken typ av MIMO som skulle implementeras: spatiell multiplexing eller rumsdiversitet. Här har rumsdiversitet valts eftersom att de algoritmer som finns är relativt enkla och bäst motsvarar tidsåtgången för ett examensarbete.

En annan avgränsning är att inte gå vidare och titta på SCA-strukturen och hur GNU Radio eventuellt skulle kunna SCA-anpassas.

Utöver de valda avgränsningar som gjorts finns en del mer implementationsnära avgränsningar. Dessa redogörs i tabell 1.

Tabell 1. Avgränsningar.

Avgränsning	Förklaring
Frekvens	Det frekvensband som används är 2,40 – 2,48 GHz. Anledningen är att frekvensbandet är fritt att sända på och att det har stora likheter med WLAN.
Modulation	Tillgängliga modulationsmetoder är GMSK, DQPSK samt DBPSK. Författaren har valt att använda DQPSK, se kapitel 5 för vidare förklaring.
Laborationsmiljö	Datorerna står uppställda med cirka en meters avstånd. Den USRP som används för mottagning har en tre meter lång USB kabel vilket gör det möjligt att flytta mottagaren, 4 till 5 meter från sändaren, för att minska effekten från den direkta signalen. Laborationsmiljön i övrigt kan antas ge upphov till stor flervägsutbredning eftersom en mängd annan utrustning finns i samma rum.
Datahastighet	500 kb/s
Bandbredd	226 kHz (gäller för DQPSK vid -3 dB)[1]

[1] Fler mätresultat finns i [14].

1.5 Metodbeskrivning

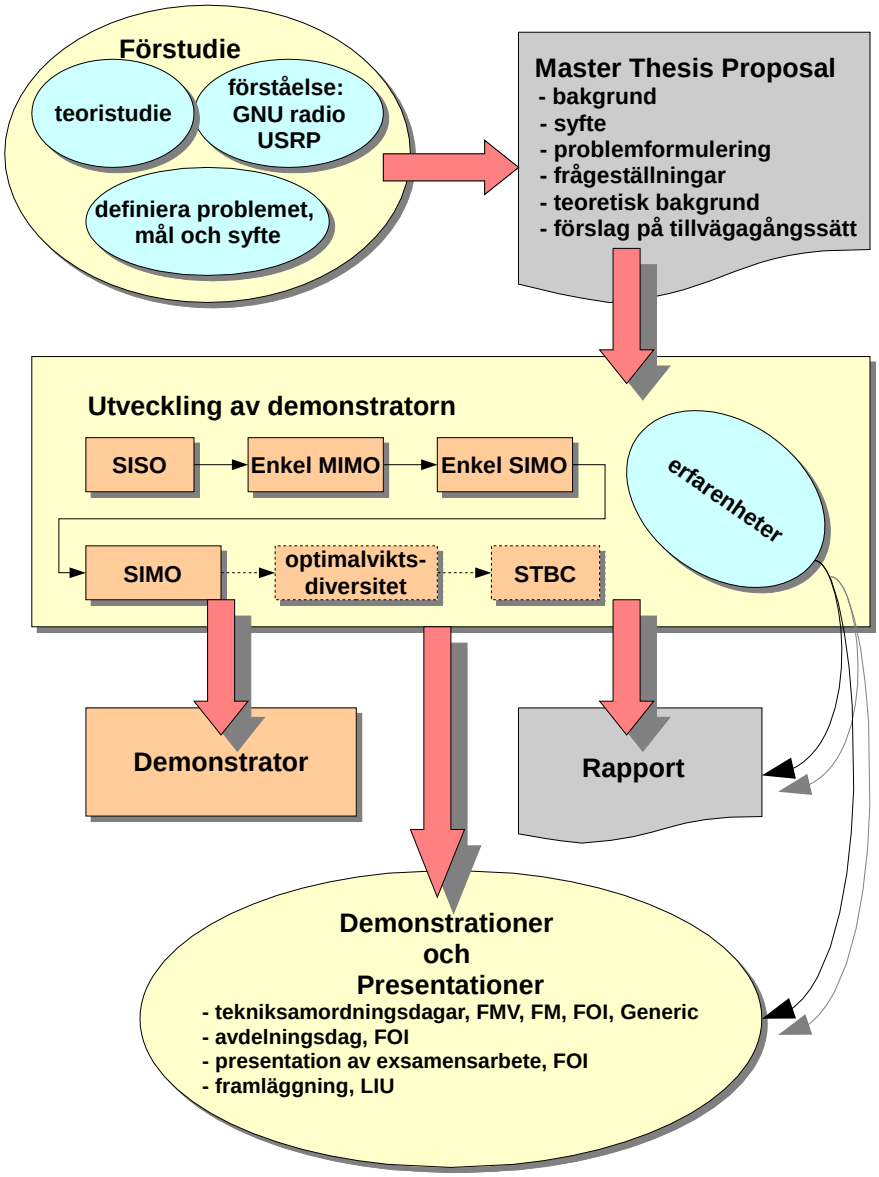
Arbetet kan delas in i tre stora delar som illustreras i **Figur 1**:

- Förstudie.
- Utveckling.
- Demonstrationer och presentationer.

Förstudien resulterade i en så kallad Master Thesis Proposal som presenterades innan halva tiden av examensarbetet gått. Detta dokument är grunden till arbetet och förutom bakgrund, syfte och frågeställningar med mera presenterades dessutom en tidsplan och ett förslag på tillvägagångssätt.

Utvecklingen av demonstratorn skedde i olika delmoment. Ett naturligt första steg var SISO-länken som hela tiden användes som jämförelse under arbetets gång. MIMO i form av Optimalviktsdiversitet och STBC (Space-Time Block Coding) är de delar som ännu inte blivit implementerade. Utvecklingsdelen resulterade i en SIMO-demonstrator men också i denna rapport. Rapporten innehåller samlade erfarenheter från projektet, både teoretiska och praktiska.

Demonstratorn har också visats vid en rad olika tillfällen då även erfarenheterna från utvecklingen har presenterats.



Figur 1. Metodbeskrivning.

1.6 Kunskapsområden som examensarbetet berör

Nedan ges en kort beskrivning över vilka kunskapsområden som krävs för att kunna genomföra examensarbetet.

Tabell 2. Kunskapsområden.

Område	Beskrivning
Digital kommunikation	Generell förståelse för hur ett digitalt kommunikationssystem är uppbyggt och hur trådlösa kommunikationssystem fungerar.
Mjukvaruradio - generellt	Generell förståelse över begreppet mjukvaruradio och vilka möjligheter som ges.
GNU Radio - specifikt	GNU Radios uppbyggnad och struktur samt möjligheter och begränsningar över vad som är praktiskt genomförbart.
MIMO Diversitet Spatiell multiplexing Lobformning	Begreppen SISO, SIMO, MISO samt MIMO och skillnader däremellan. Grundläggande förståelse för begreppen diversitet, spatiell multiplexing samt lobformning. Detaljerad kunskap om några specifika kända algoritmer (här ingår kännedom om Alamoutis STBC samt olika diversitetskombineringsmetoder)
Radio och RF elektronik	Kunskap om hårdvara och hur den fungerar tillsammans med mjukvaran.
Digital signalbehandling	Digitala filter, AD/DA-omvandling, FFT (Fast Fourier Transform), med mera.
Programmering C++ Python	Kunskap om programmering och objekt-orienterad programmering i C++ och Python.

2 Introduktion till mjukvaruradio

2.1 Definition

Termen mjukvarudefinierad radio (*eng. Software Defined Radio*) myntades 1991 av Joseph Mitola och området har sedan dess vuxit. Själva termen har blivit lite av ett modeord och det är inte alltid helt lätt att förstå vad som menas med begreppet. Jeffrey H. Reed som har skrivit boken "Software radio – A modern approach to radio engineering" definierar mjukvaruradio på följande sätt:

"A good working definition of a software radio is a radio that is substantially defined in software and whose physical layer behavior can be significantly altered through changes to its software [1]."

För att en radio ska vara en mjukvaruradio krävs alltså att största delen av radion är definierad i mjukvara och att de delar av radion som tillhör det fysiska lagret till stor del kan ändras genom mjukvara. Idén med mjukvarudefinierad radio är att maximera flexibiliteten genom att flytta mjukvaran så nära antennen som möjligt, det vill säga genom att flytta AD-omvandlaren så nära antennen som möjligt. Vilket medför att stora delar av radions egenskaper bestäms i mjukvaran. Det optimala skulle vara att digitalisera signalen redan i antennen, hos mottagaren, innan signalen behandlas i mjukvaran, så att all typ av signalbehandling sker i mjukvaran. Då det idag inte är möjligt måste en viss del av signalbehandlingen ske i RF-hårdvara innan signalen kan bearbetas i mjukvaran. Enkelt beskrivet strävar utvecklingen av mjukvaruradio till att förvandla hårdvaruproblem till mjukvaruproblem [2].

2.2 Varför mjukvaruradio?

Det intressanta med mjukvaruradio är inte enbart att det är möjligt att få samma funktionalitet med mjukvara som med hårdvara, utan även vad som utöver det går att göra med mjukvara som inte varit möjligt med traditionella radior.

En av dessa möjligheter är att radion lätt blir rekonfigurerbar som en följd av att radions egenskaper (vågform) definieras i mjukvaran. Istället för att byta ut hårdvarudelar, när funktionaliteten ska ändras eller när uppgraderingar behövs, kan man byta ut mjukvaran. Det blir också möjligt att göra det direkt, till exempel via ett luftgränssnitt.

Att definiera vågformen i mjukvara för också med sig fördelen att det går att bygga så kallade kognitiva radior som efter att ha lyssnat på det radiospektrum som ligger i radions område kan ställa in sig själv för bästa prestanda.

En annan fördel med mjukvaruradio är att det blir mycket enklare, jämfört med hårdvaruradio, att lyssna på flera olika kanaler samtidigt vilket betyder att det går att skapa system som kan kommunicera med varandra trots olika vågformer [2]. Detta passar väl in med den utveckling som FM går mot med ökad samverkan mellan olika nationer, men också mellan olika organisationer inom Sverige, som till exempel militär och polis.

För att en mjukvaruradio ska kunna hantera flera olika bredbandiga vågformer krävs dock en generell RF-hårdvara och flera olika antenner vilket inte är helt lätt att realisera. Desto mer generell hårdvaran blir desto klumpigare blir radion. Detta gäller också för mjukvaran: en generell hårdvara riskerar att resulterar i en ”klumpig” mjukvara. Dessutom kostar generalitet eftersom hårdvaran då blir mer komplex [1].

2.3 Områdets status

JTRS (Joint Tactical Radio System) är system från ett pågående projekt i USA på uppdrag av DoD (Department of Defense). Projektet syftar till att ta fram en familj av mjukvaruradior som ska förse soldaten med tal-, data- och videokommunikation från 2 MHz till 2 GHz. En viktig del i JTRS är SCA (Software Communication Architecture) som är en öppen standard och är grunden för vågformen i JTRS [3].

Den svenska motsvarigheten till JTRS är GTRS (Gemensamt Taktiskt RadioSystem) som är ett system under anskaffande inom FM. Systemet tillhör den kategori av radiosystem som har högst krav på robusthet och funktion [4].

GNU Radio är fri programvara och är den mjukvara som används i detta examensarbete (se kapitel 3.2).

3 Specifik mjuk- och hårdvara

3.1 Materiella förutsättningar

Följande utrustning tilldelades för genomförande av examensarbetet (se tabell 3).

Tabell 3. Specifik mjuk- och hårdvara.

Namn	Mjuk-/hårdvara	Komponenter	Kort beskrivning
GNU Radio	Mjukvara	Signalbehandlingsblock i C++ Flödesgraf i Python Gränssnitt mellan Python och C++ i SWIG	GNU Radio är fri programvara, vilket betyder att alla kan få tillgång till det helt gratis och alla som vill kan vara med och bidra till GNU Radios utveckling.
Två USRP	Hårdvara	2 antenner (per USRP) Moderkort med FPGA (per USRP): 4 AD-omvandlare 4 DA-omvandlare 4 DDC med programmerbar decimeringstakt 2 DUC med programmerbar interpolationstakt USB 2.0 gränssnitt 480 Mb/s 16 MHz momentan bandbredd	USRP är hårdvara speciellt utvecklad för att användas tillsammans med GNU Radio och tillhandahåller gränssnittet till luften.

Namn	Mjuk-/hårdvara	Komponenter	Kort beskrivning
		Två RFX2400 dotterkort (per USRP): Bandbredd 2,3-2,9 GHz Effekt 50 mW	
Två PC: Shuttle Barebone	Hårdvara och mjukvara	Operativsystem: Linux Distribution: Gentoo	Plattform för GNU Radio och kommunikation med USRP.

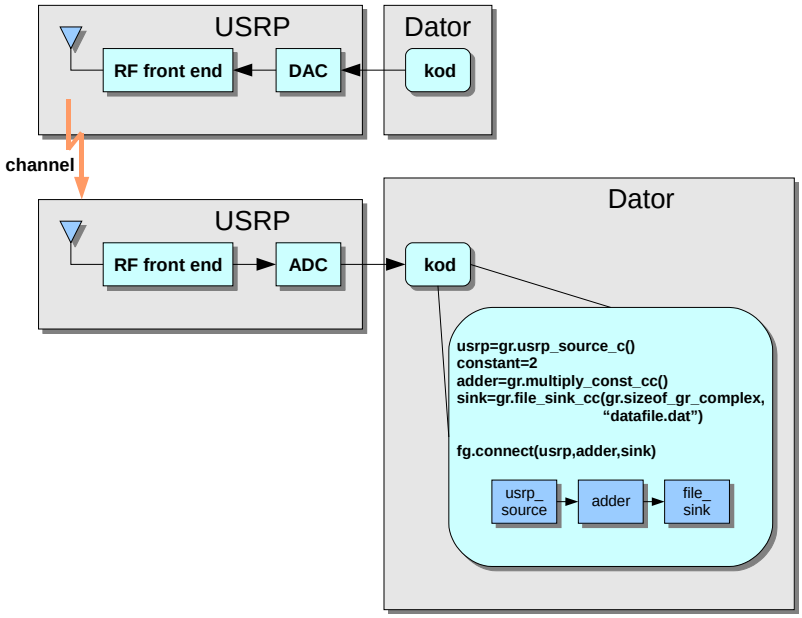
Figur 2a visar datorn och USRP:n. Två likadana utrustningsuppsättningar användes under examensarbetet. Figur 2b visar en närbild på hur USRP:n ser ut.

Figur 2 a) En av två utrustningsuppsättningar. b) USRP.

3.2 GNU Radio

GNU Radio är en samling mjukvara innehållande verktyg för att bygga en mjukvaruradio. Tanken med GNU Radio är att man ska kunna använda sig av en rad redan färdigdefinierade signalbehandlingsblock och sedan knyta ihop blocken till en flödesgraf vilket skapar vågformen. Signalbehandlingsblocken är skrivna i C++ och knyts ihop med Python som är ett språk på högre nivå än C++. Koppling mellan C++ och Python görs med hjälp av SWIG (Simplified Wrapper and Interface Generator) som är en så kallad gränssnittskompilator. Det är också möjligt att skriva sina egna signalbehandlingsblock [5].

I varje flödesgraf anger man en källa och en sänka. Figur 3 visar en förenklad bild över hur en flödesgraf hos mottagaren kan se ut. I koden anges USRP:n som källa och sänkan är en fil dit data som tas emot ska skrivas. Koden anger också att de data som ska tas emot från USRP:n är komplexa tal, genom suffixet _c. I figur 3 är det också angivet, som ett exempel, att varje tal i den komplexa dataströmmen ska adderas med konstanten 2.



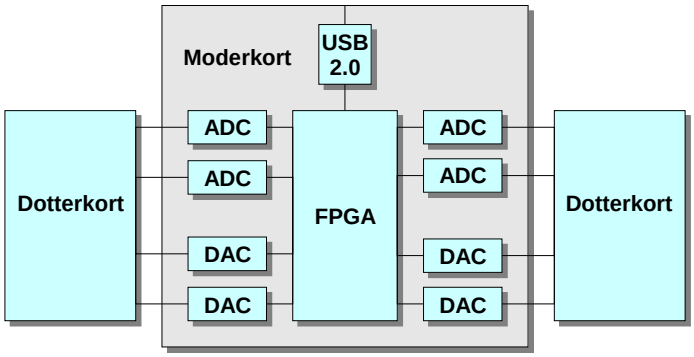
Figur 3. GNU Radio och USRP.

GNU Radio går att köra på både Linux och Windows-plattformar. Linux-plattformen tillsammans med GNU Radio är dock vanligast vilket i praktiken betyder att det finns mest tekniskt stöd för Linux.

3.3 Universal Software Radio Peripheral

USRP:n är hårdvara tillverkad för att köras tillsammans med GNU Radio-programvara och rekommenderas av de personer som leder GNU Radio-utvecklingen. Eftersom USRP:n är tillverkad enbart för detta ändamål finns det mest information för den här hårdvaran tillsammans med GNU Radio i diskussionsforum och på GNU Radios hemsida. Det är också möjligt att använda USRP:n tillsammans med LabView eller Matlab/Simulink eller att skapa sin egen mjukvaruradiomiljö för USRP:n [6].

Hur USRP:n är uppbyggd och vilka begränsningar den har är en viktig del i mjukvaruradioutveckling med GNU Radio och påverkar programmeringen till stor del. Därför är det viktigt att veta hur USRP:n fungerar och vilka begränsningar den har.

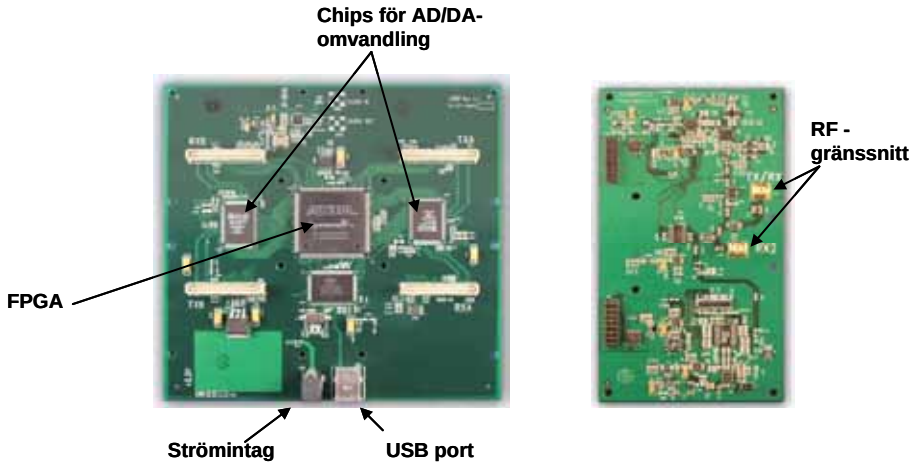


Figur 4. Schematisk bild av USRP:n.

USRP:n har ett moderkort som kan kombineras med upp till fyra små dotterkort eller två stora dotterkort. I examensarbetet användes två stora dotterkort som figur 3 visar. Dotterkorten är kopplade till en FPGA (Field Programmable Gate Array) via DA- och AD-omvandlare. Gränssnittet mellan USRP:n och datorn sker via USB 2.0 (USB 1.x finns det i dagsläget inget stöd för) [5].

3.3.1 Moderkort och dotterkort

Moderkortet är ett integrerat kort som tillsammans med ett chassi, strömkabel, USB-kabel och RF-kabel (som går mellan dotterkort och antennutgång) utgör USRP:ns grundpaketet. Till detta går det att beställa en rad olika dotterkort, som täcker olika frekvensområden. De dotterkort som används här kallas för RFX2400 och täcker 2,3-2,9 GHz men har blivit strypta och täcker därför endast 2,40-2,48 GHz-området. Anledningen till detta är att det sistnämnda frekvensbandet är fritt att sända på utan att speciella tillstånd behövs.

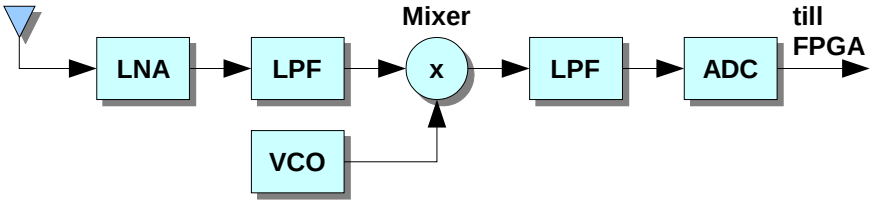


Figur 5. Moderkort och RFX2400 dotterkort.

På moderkortet sitter FPGA:n och DA/AD-omvandlarna. Varje AD-omvandlare har en sampeltakt på 64 MS/s och varje sampel har 12 bitars upplösning, det vill säga varje sampel beskrivs med 12 bitar. DA-omvandlarna har en sampeltakt på 128 MS/s och 14 bitars upplösning. Figur 5, som är hämtad från [7] och [8], visar hur moder- och dotterkort ser ut med chipsen för AD/DA-omvanlding, FPGA:n, USB-porten samt strömintag utmärkta på moderkortet och RF-gränssnittet (kabel ut- och ingång till antenner) utmärkta på dotterkortet [5].

3.3.2 RF hårdvara

Den RF hårdvara (*eng. RF front end*) som är implementerad på dotterkorten kan för det mesta behandlas som en svart låda med en enda styrparameter; nämligen den centerfrekvens som är av intresse. Trots detta kan det ändå vara bra att förstå vad som händer i RF hårdvaran. En typisk topologi för en front end ges i figur 6. Det som händer här är att signalen först förstärks i en LNA (Low Noise Amplifier) för att sedan filtreras i ett lågpassfilter, LPF (Low Pass Filter). Filtret används för att välja ut den signal som ligger i ett visst frekvensområde. Signalen blandas sedan ner med hjälp av en mixer, som kan ses som en enkel multiplicerare, och en VCO (Voltage Controlled Oscillator) som genererar en sinussignal. Filtret som följer används för att filtrera bort de oönskade delar som uppstår av multiplikationen. I USRP-fallet skickas signalen efter RF hårdvara till FPGA:n som behandlas i kapitel 3.3.



Figur 6. RF-hårdvara.

Anledningen till att signalen blandas ner är för att minska samplingstakten. Enligt Nyquists kriterium så måste samplingstakten vara minst två gånger den högsta frekvensen. Eftersom den utsända signalen ligger på 2,40 -2,48 GHz skulle det ge en otroligt hög samplingstakt, vilket i sin tur skulle resultera i otroligt mycket datakraft för att hinna bearbeta alla sampel. När signalen istället blandas ned i RF hårdvaran flyttas samma signal från en högre frekvens till en lägre och kan på så sätt minska samplingstakten [9].

3.3.3 Field Programmable Gate Array

En viktig del när man arbetar med GNU Radio/USRP är att förstå vad som händer i FPGA:n. Väldigt enkelt beskrivet är FPGA:ns uppgift att utföra kraftfulla beräkningar och minska datatakten till något man kan överföra på USB 2.0 vidare in till mjukvaran i datorn [10].

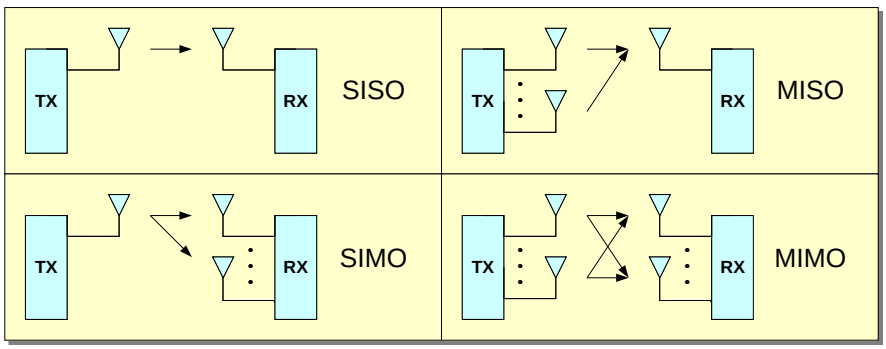
FPGA:n kan liknas vid en liten dator som går att programmera till att göra precis vad som önskas. För att programmera FPGA:n krävs en hel del kunskap och om den programmeras fel kan den hänga sig permanent. Som tur är finns det en standardkonfiguration som är användbar till det mesta. Standardkonfigurationen gör det möjligt att plocka ut de delar av det digitaliserade spektrum som är av intresse, blanda ner dem till basband och decimera dem efter behov. Detta är likvärdigt med det som händer i RF hårdvara, enda skillnaden är att detta görs av DDC som sitter i FPGA:n och att det görs på digitaliserade sampel. Fördelen med att göra detta i den digitala domänen är att det går att ändra centerfrekvensen omedelbart vilket är användbart vid FHSS-system (Frequency Hopping Spread Spectrum systems) [5].

I FPGA:n sitter en multiplexer samt flera DDC. Varje DDC har en I-ingång och en Q-ingång och kan kopplas samman med varje AD-omvanldare genom multiplexern (se appendix 9.1.2). Detta gör att man kan styra om man vill använda ett eller två dotterkort. Varje DDC är i sin tur kopplade till en interleaver-funktion som skickar data vidare till USB 2.0-gränssnittet [10].

4 Fördjupad teoretisk bakgrund för examensarbetet

4.1 Inledning Multiple-Input-Multiple-Output

I traditionella system används SISO (Single-Input-Single-Output) vilket betyder att *en* sändarantenn används och *en* mottagarantenn används. Namnet syftar på att det är en signal som går in i kanalen och en signal som kommer ut från kanalen. Om flera mottagarantenner används när det endast finns en sändarantenn kallas det för SIMO (Single-Input-Multiple-Output). Motsvarande för flera sändarantenner och en mottagarantenner kallas för MISO (Multiple-Input-Singel-Output). MIMO kallas det när det finns minst två sändarantenner och minst två mottagarantenner.



Figur 7. SISO, SIMO, MISO, MIMO.

MIMO-tekniken har en rad olika utformningar. Dessa kan delas in i tre delområden:

- Rumsdiversitet, även kallat spatiell diversitet
- Spatiell multiplexing
- Lobformning

Rumsdiversitet är en teknik för att minska de negativa effekter som fädning för med sig. Spatiell multiplexing används för att öka kapaciteten för ett system genom att utnyttja flervägsutbredning och lobformning används för att öka

signalbrusförhållandet vilket kan öka kapaciteten, räckvidden eller minska bitfelshalten [4].

I traditionella SISO-system är metoderna för att öka kapaciteten eller minska effekterna av fädning begränsade. Antingen så kan bandbredden ökas, vilket ökar antalet bitar per sekund och därmed gör det möjligt att lägga på mer redundant data för att rätta de fel som uppstår, eller så kan effekten ökas. Problemet med att öka bandbredden är att den är en knapp resurs som helst inte ska slösas med i onödan. Det kan också få en negativ effekt på närliggande kanaler, antingen spektralt närliggande eller geografiskt närliggande. Detta gäller även för effektökning. Därför finns det stränga regler för bandbredd och effekt, vilket begränsar kapaciteten och prestandan i systemet.

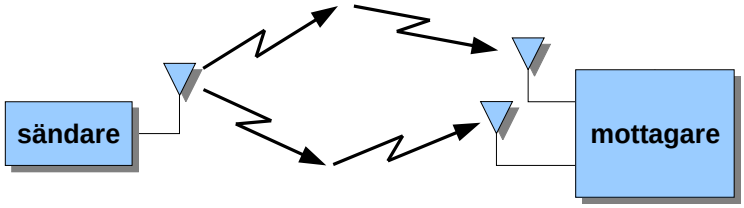
MIMO-system kommer runt dessa problem, som vanligen uppstår i urban miljö där flervägsutbredningen är stor, genom att använda rumslig diversitet och spatiell multiplexing. Systemen är dock kostsamma eftersom en ökad förmåga vad gäller databehandling behövs. Lösningen är att använda MIMO-tekniken tillsammans med mjukvaruradio [11].

IEEE (Institute of Electrical and Electronics Engineers) har tagit fram en standard som heter 802.11n som innefattar MIMO-teknik. Denna standard är inte helt klar ännu men det finns redan trådlösa routrar på marknaden som använder en betaversion av standarden.

4.2 Rumsdiversitet

Rumsdiversitet kan delas in i två olika delar: mottagningsdiversitet och sändningsdiversitet. Skillnaden mellan de båda är att vid mottagningsdiversitet används SIMO-teknik och vid sändningsdiversitet används MISO-teknik. Det går också att kombinera de båda varvid MIMO-tekniken erhålls.

Vid rumsdiversitet drar man nytta av att kanalerna mellan sändar- och mottagarantenner fädar olika. Detta under förutsättning att antennerna är tillräckligt separerade spatiellt för att endast begränsad korrelation ska uppstå. Vid mottagaren väljs den signal som är starkast, alternativt kombineras signalerna så att fädningen blir så liten som möjligt [4].

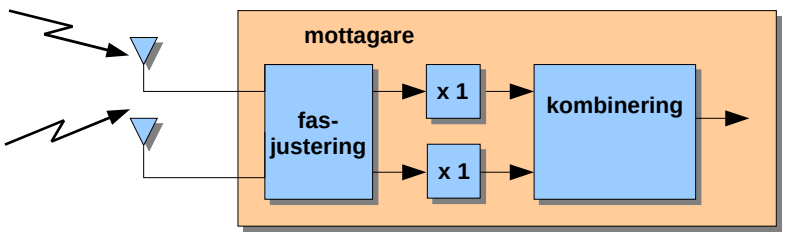


Figur 8. Rumsdiversitet.

Två olika metoder som används för att kombinera signalerna i mottagaren är likaviktsdiversitet och optimalviktsdiversitet. Det finns också ett mellanting mellan likaviktsdiversitet och optimalviktsdiversitet vilket är valdiversitet, då den starkaste mottagna signalen används. Valdiversitet tas dock inte upp i denna rapport eftersom tekniken i praktiken i princip blir lika komplicerad som optimalviktsdiversitet, eftersom kanalen måste estimeras, men ger en sämre prestanda än optimalviktsdiversitet [12].

4.2.1 Likaviktsdiversitet

Signaler som kombineras med likaviktsdiversitet (*eng. Equal Gain Combining*) kombineras genom att varje signal viktas med en förstärkningsfaktor som är lika stor för alla signaler, exempelvis med faktorn 1 som i figur 9. För att signalerna ska kunna kombineras krävs att de ligger ”rätt” i fas. Om så inte är fallet riskeras att signalerna kombineras destruktivt och i värsta fall helt tar ut varandra [1]. I fallet med två signaler måste därför den ena signalen ”skjutas” åt rätt håll så att de båda signalerna går att kombinera.



Figur 9. Likaviktsdiversitet.

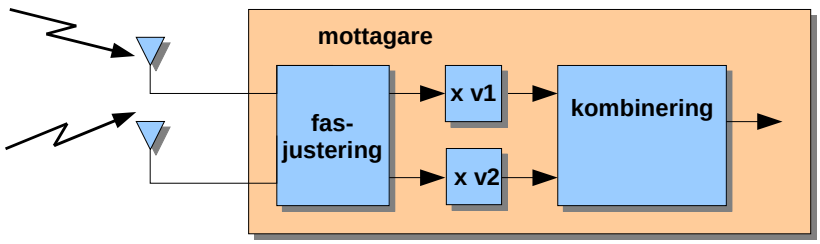
Den här metoden tar inte hänsyn till signalbrusförhållandet vid kombineringen, vilket är fallet i optimalviktsdiversitet, utan det antas att bruset är lika stort på de båda mottagna signalerna. Om bruset är väldigt stort i en av grenarna kan detta resultera i att signalerna kombineras på ett sätt som gör att signalen blir sämre än om en SISO-mottagare använts. För att tekniken ska ge en diversitetsvinst krävs

att de mottagna signalerna fädar kraftigt. Om fädningen är liten kommer den direkta signalen ge det största bidraget vid kombineringen vilket kommer att resultera i en prestanda liknande den vid en SISO-mottagare.

Genom den här typen av kombinering får man en mottagare som är relativt enkel med en prestanda som bara är en aning sämre än en mottagare som använder optimalviktsdiversitet [12].

4.2.2 Optimalviktsdiversitet

Vid optimalviktsdiversitet (*eng. Maximum Ratio Receive Combining*) viktas varje signal med en förstärkningsfaktor som är proportionell mot signalens signalbrusförhållande. Den signal som har störst signalbrusförhållande får större vikt i kombineringen än den signal som har mer brus.



Figur 10. Optimalviktsdiversitet.

Metoden liknar likaviktsdiversitet men är en aning mer komplicerad eftersom kanalen måste estimeras (både fas och amplitud) i alla grenar vilket resulterar i en mer komplex mottagare [1].

4.2.3 Alamoutis Space-Time Block Coding

Alamoutis STBC (Space-Time Block Coding) är ett sätt att genomföra sändningsdiversitet som sedan kan kombineras med mottagningsdiversitet. I sin enklaste form består schemat av en MISO-länk med två sändarantenner och en mottagarantenn. Från sändaren skickas samma information från två olika antenner vid två olika tidpunkter (se tabell 4). Vid den första tidpunkten skickas originalsignalerna s_0 på antenn 1 och s_1 på antenn 2. Den andra gången skickas komplexkonjugatet av s_0 på antenn 2 och det negativa komplexkonjugatet av signal s_1 på antenn 1.

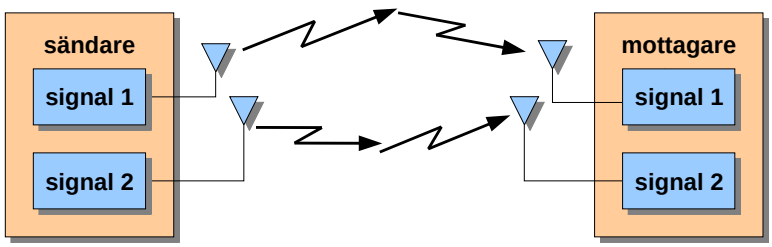
Tabell 4. Sekvens för sändningsdiversitet med två antenner.

tid	antenn 0	antenn 1
t	s_0	s_1
t+T	$-s_1^*$	s_0^*

Vid mottagaren estimeras kanalen och de fyra utskickade signalerna kombineras ihop till två nya. Signalerna skickas sedan till en maximum likelihood-avkodare som avgör vilka av signalerna som sänts [13].

4.3 Spatiell multiplexing

Spatiell multiplexing är en metod för att öka kapacitet genom att utnyttja flervägsutbredning. Genom att utnyttja att signaler som sänds från olika antenner tar olika vägar går det att identifiera olika så kallade spatiella signaturer (se figur 11). Dessa kan ses som olika kanaler. Ju större flervägsutbredning är desto lättare blir det att urskilja dessa ”kanaler”. Detta gör det möjligt att urskilja olika signaler som sänts från olika antenner, på samma frekvens, även om dessa antenner ligger relativt nära varandra [4].



Figur 11. Spatiell multiplexing.

5 Implementeringsbeskrivning

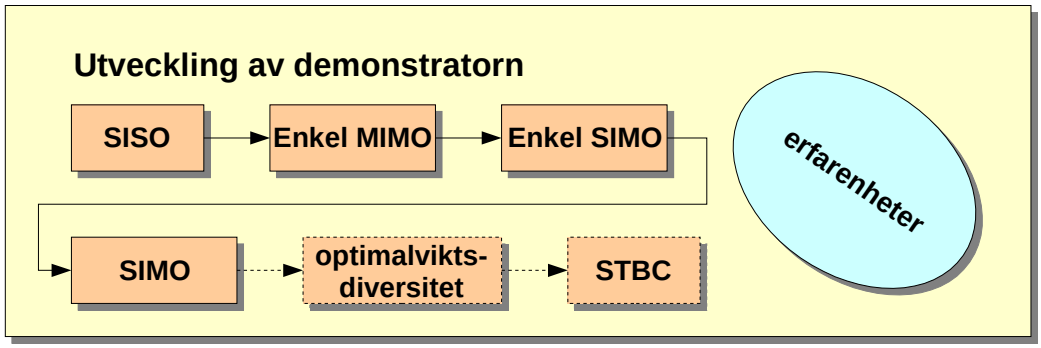
5.1 Inledning

Det första steget i implementationen var att få en SISO-länk att fungera (se figur 12). SISO-länken räknades som fungerande när det gick att överföra en datafil från den ena datorn till den andra med någon form av digital modulation. Detta gjordes för att verifiera att GNU Radio tillsammans med USRP fungerade som tänkt.

Enkel-MIMO gjordes för att testa att det går att sända och ta emot signaler på två olika antenner samtidigt men på olika bärvågsfrekvenser. Det var också en utmaning att kunna urskilja hur de olika dotterkorten och antennerna skulle styras för att båda antennerna, hos sändare såväl som hos mottagare, skulle kunna fungera samtidigt. En viktig del i detta var förståelse för vad som händer i USRP:n och då speciellt hur multiplexern styrs med mjukvara.

Steget efter var Enkel-SIMO. Denna del innehåller ingen diversitetsalgoritm utan gjordes som ett första steg mot mottagningsdiversitet. Till skillnad från Enkel-MIMO, som är två parallella instanser av en sändar- och en mottagarkedja är, Enkel-SIMO en modifierad mottagare. Enkel-SIMO är svårare än Enkel-MIMO eftersom det endast är en frekvens som används. Anledningen till att mottagningsdiversitet valdes som ett första steg är att det inte behövs en mer komplex sändare eftersom det endast är *en* signal som sänds och samma signal tas emot på två antenner. Att börja med sändningsdiversitet hade resulterat i en mer komplex mottagare än om SISO-teknik hade använts eftersom de båda mottagna signalerna måste separeras i mottagaren på något sätt. Detta betyder i praktiken att det blir svårare att testa enskilda delar av programmet om arbetet sker med både mottagare och sändare parallellt. I Enkel-SIMO används samma sändare som i SISO-länken.

Steget efter Enkel-SIMO var att implementera någon form av diversitetsteknik hos mottagaren. Likaviktsdiversitet valdes eftersom det är den enklaste kombineringsmetoden (se kapitel 5.2). Denna del av implementationen kallas för SIMO.

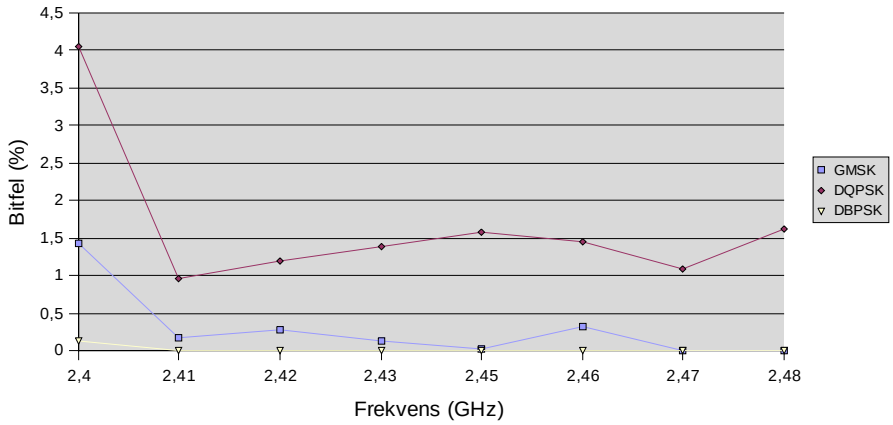


Figur 12. Metodbeskrivning över implementation.

I figur 12 visas också, förutom ovannämnda delar, ett block som innehåller optimalviktsdiversitet. Denna del har ännu inte implementerats men vore en intressant fortsättning på arbetet. Även Alamoutis STBC var inplanerat från början men eftersom tidsåtgången för implementationen av de första delarna tog längre tid än beräknat begränsades implementationen till SISO, Enkel-MIMO, Enkel-SIMO samt SIMO.

Nämnas bör också att blocket STBC, i figur 12, antagligen skulle behövas delas upp i ungefär lika många delar som är beskrivet innan detta block, det vill säga att implementera STBC skulle antagligen vara ett lika omfattande arbete som det arbete som redan är gjorts, om inte mer. För att implementera STBC-kodare och avkodare behövs nya signalbehandlingsblock skrivas i C++ och integrera dem i GNU Radio- strukturen.

Alla olika delar, från SISO-länken till likaviktsdiversitet, kan använda tre olika modulationstekniker: DQPSK, DBPSK, GMSK. Här har DQPSK valts eftersom den ger flest fel (se figur 13) vilket innebär att det blir lättare att se skillnader mellan SISO och SIMO. Resultatet av mätningarna visas i figur 13 där antalet bitfel i procent hos DQPSK, DBPSK och GMSK plottas för 8 olika mätningar.

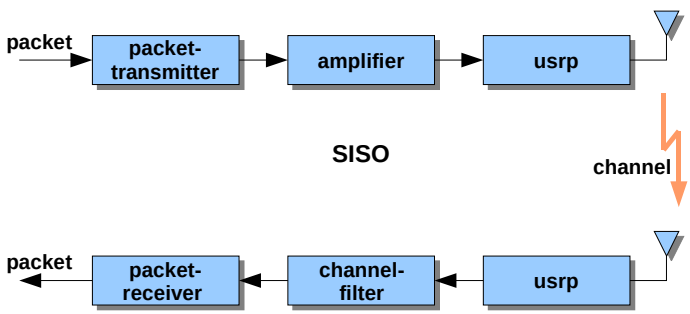


Figur 13. Bitfel.

Varje mätning har gjorts på olika frekvenser mellan 2,40 GHz och 2,48 GHz som representerar olika kanaler. Mätningen på frekvens 2,44 GHz har dock inte tagits med, vilket gäller även för figur 18 (se kapitel 6.3 för vidare förklaring).

5.2 SISO

SISO-länken är utgångspunkten för de delar som senare utvecklas i demonstratorn. SISO-länken består av ett program som sänder och tar emot digital data. Enkelt beskrivet fungerar den så att en fil sänds genom att först packas in i ett paket. Paketet skickas sedan till packet-transmitter, som är en metod i Python, där viss signalbehandling sker (se figur 14). Det är också där paketet moduleras. Signalströmmen skickas sedan vidare till en förstärkare. Signalen representeras här av en ström av komplexa tal. Dessa komplexa tal skickas sedan vidare till USRP:n där de görs om till en analog signal och skickas ut via antennen.

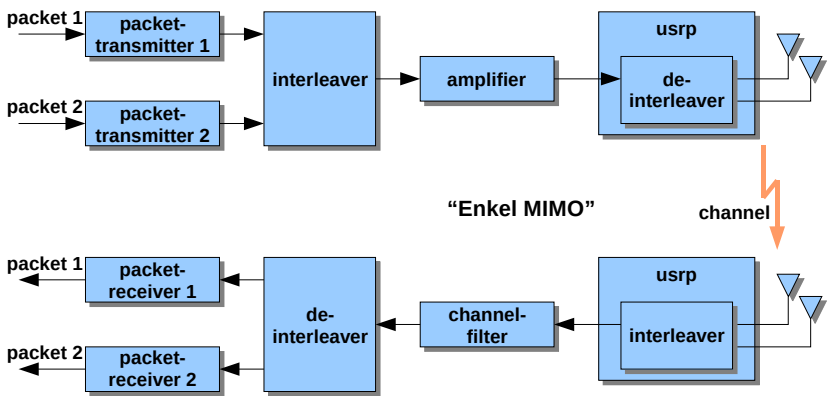


Figur 14. SISO-länk.

Mottagaren liknar sändaren så när som på förstärkaren och i omvänd ordning. Istället för en förstärkare har mottagaren ett kanal-filter som är ett FFT-filter och fungerar som ett lågpassfilter. Paketet demoduleras sedan och ut kommer den ursprungliga filen.

5.3 Enkel-MIMO

I Enkel-MIMO används två antenner för sändning, på olika bärvågsfrekvenser, och två antenner för mottagning, på motsvarande frekvenser. Ett enkelt blockschema över sändar- och mottagardelen visas i figur 15. I sändaren skapas först två olika paket av två olika dataströmmar. Paketen skickas sedan vidare till modulationsblocket som följs av ett block som kallas interleaver och en förstärkare. Interleaver-blocket slår ihop två dataströmmar till en. Detta görs eftersom *en* ström måste överföras via USB-kabeln (det finns bara en kabel mellan datorn och USRP:n). I USRP:n delas sedan strömmarna upp och skickas ut på två olika antennelementen. I USRP:n finns det mer än bara interleaver men de är utelämnade för tydlighetsskull (se kapitel 3.3).

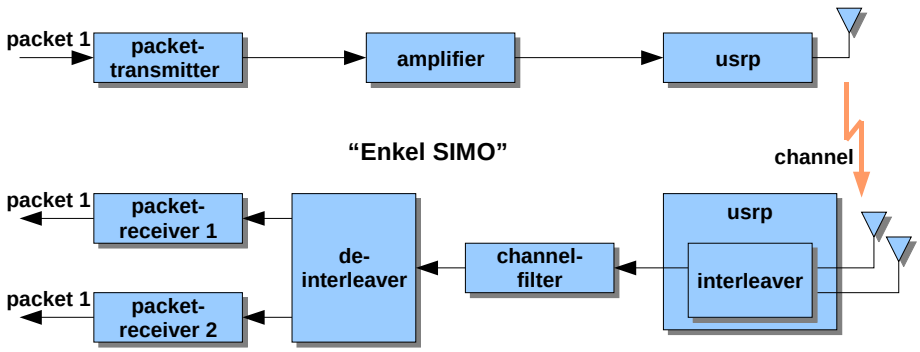


Figur 15. Enkel MIMO.

Mottagaren fungerar på liknande sätt fast i omvänd ordning. Paketen tas emot på två antenner, blandas ner och går igenom interleavern innan de skickas från USRP:n vidare in till datorn. Där filtreras signalströmmen och skickas till deinterleaver-blocket. I deinterleaver-blocket separeras de två olika dataströmmarna och skickas till varsin packet-receiver som demodulerar dataströmmarna till de två ursprungliga paketen.

5.4 Enkel-SIMO

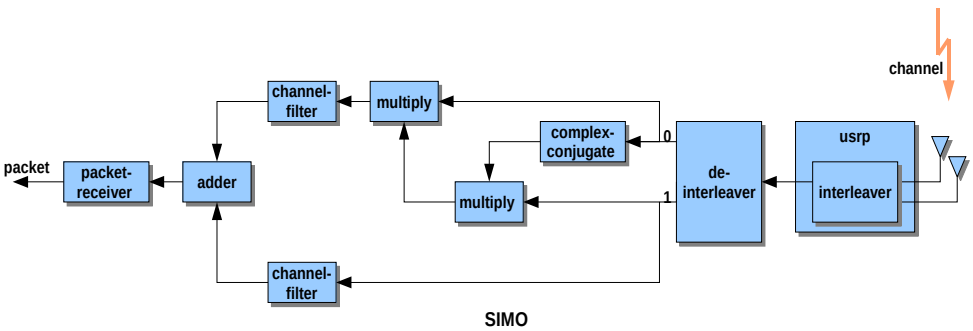
Sändardelen i Enkel-SIMO är densamma som vid SISO medan mottagardelen har samma struktur som hos Enkel-MIMO. Skillnaden mellan mottagaren i Enkel-MIMO och mottagaren i Enkel-SIMO är att i Enkel-SIMO tas samma signalström emot på två olika antenner. De båda mottagarantennerna är alltså inställda på samma frekvens och man får således ut två ”kopior” av samma paket genom två separata dataströmmar.



Figur 16. Enkel SIMO.

5.5 SIMO

Nästa steg i examensarbetet var att implementera en algoritm för enkel mottagningsdiversitet, vilket resulterade i SIMO-delen. Skillnaden mellan SIMO och Enkel-SIMO är att i SIMO kombineras de båda signalerna ihop, i mottagaren, till en enda signal. Blocken är kopplade så som visas i figur 17.



Figur 17. Implementering av likaviktsdiversitet.

Signalflödet som beskrivs i figur 17 går att beskriva matematiskt. De båda signalerna som tas emot är komplexa tal och kan beskrivas enligt ekvation 1.

$$r_0 = \alpha_0 e^{j\theta_0}$$
$$r_1 = \alpha_1 e^{j\theta_1}$$

Ekvation 1. Mottagna signaler.

Det som händer i mottagaren är att den ena signalen, r_0, förskjuts i fas. Detta görs genom att först ta komplexkonjugatet av signalen och multiplicera det med signal r_1. Det som fås fram är skillnaden i fas mellan de båda signalerna. Denna skillnad multipliceras med r_0 vilket resulterar i att r_0 får samma fas som r_1 (se ekvation 2).

$$conj(r_0) = \alpha_0 e^{-j\theta_0}$$
$$r_1 conj(r_0) = \alpha_0 \alpha_1 e^{j(\theta_1 - \theta_0)}$$
$$r_0 r_1 conj(r_0) = \alpha_0{}^2 \alpha_1 e^{j\theta_1}$$

Ekvation 2. Fasförskjutning av signal r_0.

De båda signalerna filtreras sedan och adderas vilket ger resultatet som visas i ekvation 3.

$$r_1 + r_0 r_1 conj(r_0) = \alpha_1(\alpha_0{}^2 + 1)e^{j\theta_1}$$

Ekvation 3. Addering av signalerna.

Genom att förskjuta den ena signalen i fas och sedan lägga ihop de båda signalerna fås en kombinerad signal som innehåller information från de båda ursprungssignalerna. Detta gör att det går att utnyttja att de båda signalerna fädar olika för att få en sammansatt signal som är starkare och innehåller mindre fel än vad en enskild signal gör.

6 Jämförelse mellan SISO-länk och SIMO-länk

6.1 Experiment: Överföring av grafisk bild

För att kunna jämföra SISO-länken med SIMO-länken har en grafisk bild överförts från den ena datorn till den andra. Fördelen med att överföra en bild att det även går att titta direkt på bilden för att få en uppfattning av hur stora fel som uppstått i överföringen. Hur pass bra bilden är när den kommer fram beror naturligtvis också på vilka bitar som är fel men det ger ändå en fingervisning av vad som är en acceptabel nivå på bitfel för att det ska vara möjligt att avgöra vad bilden föreställer.

För att få en bättre uppfattning av hur bra mottagningsalgoritmen är har en algoritm som räknar bitfel implementerats. Resultatet av överföringen går att mäta i bitfel på två olika sätt. Antingen så räknas antalet bitfel för varje paket som skickas eller så går det att få en utskrift på det ackumulerade medelantalet bitfel. När mätningarna gjordes ändrades inga av de övriga förutsättningarna annat än mottagningstekniken.

6.2 Plottning av resultat

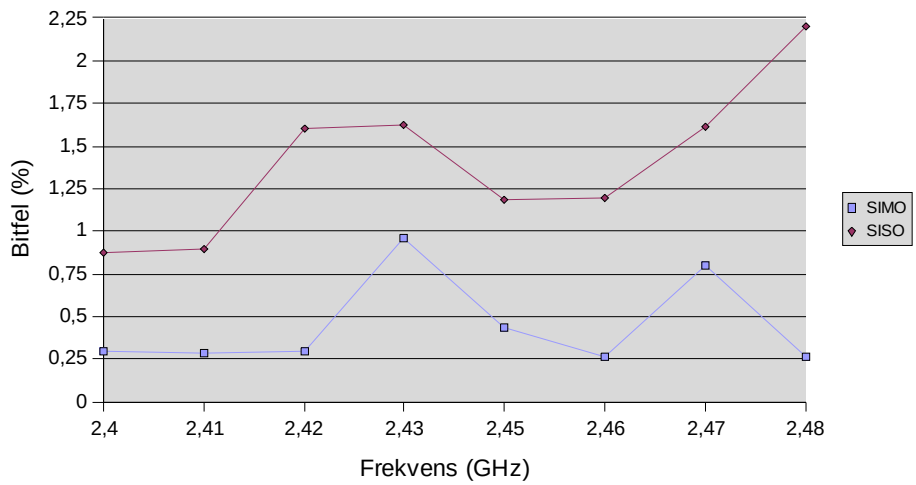
I tabell 5 visas resultatet av mätningarna som skett på frekvenser mellan 2,40 GHz och 2,48 GHz med ett mellanrum på 10 MHz. Varje frekvens representerar en kanal. Det är också möjligt att istället för att mäta på olika frekvenser flytta mottagaren i rummet för att få samma fädningseffekt. Att istället byta frekvens är en aning mer systematiskt och underlättar eftersom det är lättare att byta frekvens än att flytta runt mottagaren. Varje mätning som gjorts är ett medeltal av 100 paket och resultatet av mätningarna visas i tabell 5 och är även i figur 18.

Tabell 5. Bitfel på olika kanaler.

Frekvens (GHz)	Bitfel SISO (%)	Bitfel SIMO (%)
2,40	0,87	0,29
2,41	0,90	0,28

Frekvens (GHz)	Bitfel SISO (%)	Bitfel SIMO (%)
2,42	1,60	0,30
2,43	1,62	0,96
(2,44	7,54	6,17)
2,45	1,18	0,43
2,46	1,19	0,26
2,47	1,61	0,8
2,48	2,2	0,26
Bitfel medel:	1,39	0,44

En av mätningarna har inte tagits med vid uträkning av medeltalet av bitfelen i tabell 5 och i figur 18 och det är mätningen på 2,44 GHz. Anledningen till det är att den mätningen inte är representativ för alla mätningar i stort. Som det går att se i tabell 5 så är antalet bitfel väldigt mycket högre på den frekvensen än de övriga. Detta gäller både för SISO och för SIMO (se kapitel 6.3 för vidare förklaring).



Figur 18. SIMO jämfört med SISO.

De båda signalerna i figur 18 följer varandra åt utom på frekvensen 2,48 GHz. Anledningen till avvikelsen bedömdes bero på en tillfällighet som kan ha försvunnit om antalet mätningar ökats.

6.3 Analys av resultat

Skillnaden mellan SIMO och SISO är att SIMO är ungefär tre gånger så effektiv i detta scenario, även om man generellt kan anse att bitfelshalten är lite väl hög både för SISO och för SIMO. Anledningen är att det för det första inte finns någon felrättande kod och för det andra dämpas signaler på 2,4 GHz kraftigt varför de används vid kommunikation på korta avstånd (i experimentet ovan var avståndet cirka 3 meter mellan sändare och mottagare). Det intressanta är dock inte bitfelshalten i absoluta tal utan att antalet bitfel klart minskar vid SIMO jämfört med SISO när övriga förutsättningar är oförändrade.

På frekvensen 2,44 GHz blev bitfelshalten betydligt högre jämfört med de andra frekvenserna. Anledningen till det antogs först bero på att någon annan sändare låg just i det frekvensområde vilket kunde orsaka en störning. Frekvensbandet undersöktes därför med en signalanalysator men ingen annan sändare eller störkälla kunde upptäckas på den frekvensen. Störningarna skulle mycket väl kunna bero på hårdvaran, till exempel att resonans uppstått i kretsarna eller att antennerna inte exakt matchar 50 ohm. Vid ett tillfälle har utrustningen inte visat upp sämre resultat vid 2,44 GHz. Om detta berodde på ovanstående, på mjukvaran eller på att antennerna inte var ordenligt åtskruvade är endast spekulationer. Orsaken kunde således inte fastställas med tillgängliga resurser och medel.

Några mätningar på de interna störningar som finns i USRP:n har också gjorts. Under dessa mätningar hittades ingen störkälla som skulle kunna resultera i den höga bitfelshalten vid 2,44 GHz. Huvudsyftet med dessa mätningar var dock inte att hitta en intern störkälla utan att mäta bandbredd för olika kommunikationsmetoder för att kunna jämföra prestanda hos USRP:n med teoretiska värden. Resultaten, som presenteras i [14], visar på att de uppmätta bandbreddvärdena skiljer sig något från de teoretiska men att USRP:n är tillräckligt bra för att användas i kommunikationssyfte.

7 Slutsats

7.1 Strukturen hos GNU Radio

GNU Radio är uppbyggt med ett antal signalbehandlingsblock skrivna i C++. Genom att knyta ihop blocken med Python skapas en signalflödesgraf som används för att definiera vågformen. Förutom själva flödesgrafen krävs också en hel del annan typ av kod för att få en fungerande vågform, till exempel inställningar för USRP:n som har används i det här examensarbetet.

Själva dokumentationen är knapphändig. Det bästa sättet att lära GNU Radio, efter att man läst och förstått [5] samt Dawein Shens tutorials [9,10], är att läsa källkoden.

7.2 MIMO-system i GNU Radio

Det är möjligt att implementera ett MIMO-system i GNU Radio med digital modulation. Det finns i GNU Radio stöd för DQPSK, DBPSK samt GMSK. I det här examensarbetet har det redovisats hur man hos mottagaren kan implementera rumsdiverstitet. Anledningen till att rumsdiversitet med likaviktsdiversitet har valts är för att det är en av de lättare teknikerna att implementera eftersom ingen estimering av kanalen krävs.

Vinsterna som kan göras med den här typen av implementation är en minskning av bitfelshalten med en storleksordning av en tredjedel jämfört med ett SISO-system.

8 Diskussion

8.1 MIMO i urban miljö

Eftersom FM har ändrat inriktning till ett insatsförsvar sker mer av insatserna i urban miljö. Med en ändrad uppgift och ändrad miljö behövs också annan typ av utrustning, detta gäller även utrustning för kommunikation. Eftersom miljön i staden ser annorlunda ut kommer även radiovågorna att bete sig annorlunda än på landsbygd, på grund av till exempel reflektioner, flervägsutbredning och fädning. För att förbättra kommunikationen går det att använda sig av MIMO-teknik på olika sätt, några har tagits upp i denna rapport.

Det är också viktigt att inte glömma bort att även om insatserna till största delen sker i urban miljö så måste styrkorna även ta sig till och från staden.

8.2 Mjukvaruradio kräver hårdvara

När man talar om de fördelar mjukvaruradio erbjuder är det lätt att glömma bort hårdvaran bakom. Hårdvara krävs naturligtvis för att köra mjukvara men den behövs också i RF hårdvaran. Den hårdvara som sitter närmast antennen går inte att kompromissa bort utan den krävs för att bland annat filtrera, blanda ner signalen och digitalisera signalen. Detta medför att det behövs flera olika RF hårdvaror om flera olika bredbandiga frekvensband ska kunna användas i en och samma radio.

Mer mjukvara kräver också processorer som klarar av större datamängder. Tidigare nämndes att en del av mjukvaruradiokonceptet är att flytta mjukvaran så nära antennen som möjligt. Detta medför att det kommer att bli en större mängd mjukvara och kan också medföra en högre samplingstakt vilket i sin tur kräver mer kraftfulla processorer. I längden kan detta också medföra att tunga beräkningsdelar läggs ut på flera olika FPGA:er eller DSP:er. Trots detta har mjukvaruradio sina fördelar just för att mjukvaran är lättare att handskas med.

8.3 Generalitet

Något som måste tas hänsyn till vid mjukvaruradioutveckling är också vilken typ av radio som behövs. En generell radio kanske är kompatibel med många andra system, har en längre livslängd och en mängd olika funktioner. Samtidigt kommer denna radio behöva en mycket mer generell programvara och hårdvara

vilket är kostsamt, både mjukvaru- och hårdvarumässigt. En generell radio blir dessutom klumpigare att hantera från ett utvecklings- och vidmakthållandeperspektiv. Därför måste radioutvecklingen anpassas efter de behov som användarna har.

8.4 GNU Radio lätt?

I många forum talas det om hur lätt och snabbt det är att bygga sin egen mjukvaruradio med GNU programvara. Det stämmer till viss del. Som nämnts tidigare krävs kunskap inom en rad olika områden för att använda de byggstenar som GNU Radio tillhandahåller. Behärskar man dessa kunskapsområden krävs fortfarande en del arbete för att sätta sig in i GNU Radios uppbyggnad. De forum som menar att GNU Radio är lätt att använda och sätta sig in i menar att det är ett *relativt* lätt verktyg jämfört med andra typer av mjukvaruradior. Att bygga en mjukvaruradio efter SCA kräver till exempel en hel del mer arbete, både att sätta sig in i SCA-strukturen och att implementera den typ av teknik som är av intresse.

Det finns inte heller mycket dokumentation när det gäller GNU Radio. Även om det finns en hel del kod att läsa händer det mycket i det dolda som ibland kan vara nödvändigt att förstå för att använda GNU Radio. Detta skapar naturligtvis en hel del problem och mycket detektivarbete.

9 Förslag till fortsatt inriktning

9.1 Demonstratorn

I arbetet med demonstratorn har en del begränsningar varit tvungna att göras. Här beskrivs några av de områden som skulle ha varit lämpliga fortsättningar på arbetet.

9.1.1 Multiple-Input-Multiple-Output

En naturlig fortsättning på examensarbetet är att bygga ut demonstratorn till en komplett MIMO-länk. Det vore intressant att implementera Alamoutis schema för sändningsdiversitet kombinerat med mottagningsdiversitet och sedan göra en jämförelse mellan SISO-länken och MIMO-länken liknande den som gjorts i detta arbete mellan SISO och SIMO.

9.1.2 Graphical User Interface

Ur demonstrationssynvinkel skulle det också vara intressant att ha ett mer användarvänligt program som har ett grafiskt användargränssnitt. I det grafiska gränssnittet skulle det kunna vara möjligt att välja modulationsmetod, på vilket sätt man vill räkna bitfel, om man vill ta emot signaler med SISO- eller SIMO-teknik eller vilken fil man vill sända över.

Detta skulle göra det lättare vid demonstrationer att visa vilka val som kan eller måste göras för att köra programmet samt fördelarna med SIMO-teknik.

9.1.3 Tunnel

I GNU Radio finns det en färdig modul som skapar en tunnel mellan de båda datorerna. Denna modul skulle kunna kopplas ihop med den SIMO-implementation som gjorts i arbetet. Sett ur TCP/IP-modellens perspektiv så skulle denna tunnel-modul ligga på applikationslagret medan den SIMO- och MIMO-teknik som beskrivits tidigare i examensarbetet ligger på datalänklagret. Eftersom tunnel-applikationen ligger på så pass hög nivå jämfört med SIMO-programmet så vore det relativt lätt att koppla de olika modulerna.

Tunnelprogrammet gör det möjligt att till exempel föra över filer från den ena datorn till den andra eller att logga in på den ena datorn från den andra. Detta gör

det möjligt att skicka över större filer än den bildfil som använts samt att visa funktionaliteten på en högre nivå.

9.2 SCA-kompatibilitet

En annan intressant frågeställning som skulle kunna undersökas vid fortsatt arbete med GNU Radio är vad som menas med SCA-kompatibilitet och vad som skulle behövas för att göra GNU Radio SCA-kompatibel.

För att göra GNU Radio SCA-kompatibel krävs två olika delar: CORBA ORB med IDL- (Interface Definition Language) kompilator samt Core Framework. Den förstnämnda delen finns det många olika fria mjukvarualternativ till. Som Core Framework finns det inte lika många alternativ att välja mellan, OSSIE (Open Source SCA Implementation::Embedded) är ett alternativ som finns fritt tillgängligt [15].

OSSIE är en implementation av SCA med öppen källkod, som har skapats på Virginia Tech, ett universitet som arbetar mycket med mjukvaruradio. OSSIE går dessutom att köra tillsammans med USRP hårdvara [16].

En jämförelse mellan vågformsutveckling med GNU Radio och vågformsutveckling med SCA har gjorts i ett tidigare examensarbete [17]. Ett alternativ till fortsatt arbete skulle kunna vara att med detta examensarbete som utgångspunkt undersöka möjligheterna till att göra GNU Radio SCA-kompatibelt.

9.3 Crossbanding

En annan intressant möjlighet till fortsatt arbete med GNU Radio vore att implementera en crossbanding-funktion. Med det menas att en signal tas emot på en viss frekvens, till exempel 2,4 GHz, för att sedan blandas ner eller upp till en annan frekvens, exempelvis 400 MHz.

Appendix GNU Radio tips och trix

A.1 Erfarenheter

Ett av de första stegen i examensarbetet var att få en SISO-länk att fungera och med fungerade menas att överföra en datafil med digital modulation. Att få själva överföringen att fungera var inte helt lätt. I början strävades det efter att på ett enkelt sätt, med minimalt antal rader kod och på kort tid, få överföringen att fungera. Snabbt upptäcktes det att det inte finns något enkelt sätt att göra detta på. Det är nödvändigt att använda flera block i C++ och använda flera metoder och filer i Python för att få det att fungera, vilket ökar demonstratorns komplexitet.

För att kunna använda sig av alla de Python-filer och block som finns krävs förståelse för vad de gör och hur de gör det, vilket inte är helt enkelt. Men det är inte det enda. Det krävs också en hel del Linux-vana för att använda GNU Radio. Det vill säga om inte ett annat operativsystem används, vilket inte är att rekommendera. Det är inte helt lätt att använda Gentoo om man är van vid Windows. Bara en sådan enkel sak som att ladda hem ett program och installera detta kan ta tid om man aldrig gjort det förut.

En sak som har hjälpt vid arbetet med GNU Radio är att utnyttja ”block-tänket” som det står att läsa om på flera olika forum. Genom att rita upp varje konfiguration av block, i form av en flödesgraf, går det lättare att förstå med hjälp av papper och penna istället för att tänka ut olika konfigurationer från ett kod-perspektiv. Detta är precis på det sätt som de olika delstegen i examensarbetet har presenterats på. Även om denna typ av visualisering för de olika moduler som finns färdiga i GNU Radio inte är vanlig på forum och liknande, så är det att rekommendera. Denna typ av visualisering underlättar också när man ska gå tillbaka och titta på de steg som testats tidigare.

Det viktigaste paketet i GNU Radio är gnuradio-core. Denna kod bör inte ändras. Det kan vara värt att tänka igenom vad det egentligen är man vill göra innan man börjar ändra i den kod som ligger här. Antagligen är det så att Python-koden behöver skrivas om eller så behövs ett nytt signalbehandlingsblock om man känner att man vill skriva om koden i gnuradio-core. Det kan också vara så att just den funktion du söker redan finns i GNU Radio och att du behöver ta en extra titt i de många filer med kod som redan finns.

I GNU Radio finns det en fil som heter sdt_2rxhb_tx.rbf. Detta är standardkonfigurationen för FPGA:n, som laddas automatiskt när man sätter upp en USRP-källa eller USRP-sänka. Att lägga till kommandot fpga_filename=”

sdt_2rxhb_tx.rbf” i Pythonkoden kommer således inte att förändra något. Denna fil används om sändning och mottagning sker med en eller flera antenner. Ska man använda fler antenner hos mottagaren så måste man däremot lägga till kommandot fpga_filename=”sdt_4rx_0tx.rbf”. Detta räcker dock inte för att använda flera antenner. Samma inställningar måste göras på båda dotterkorten om inte antennerna är kopplade till samma dotterkort. Dessutom så måste antalet kanaler anges med kommandot nchan=”antal_kanaler”.

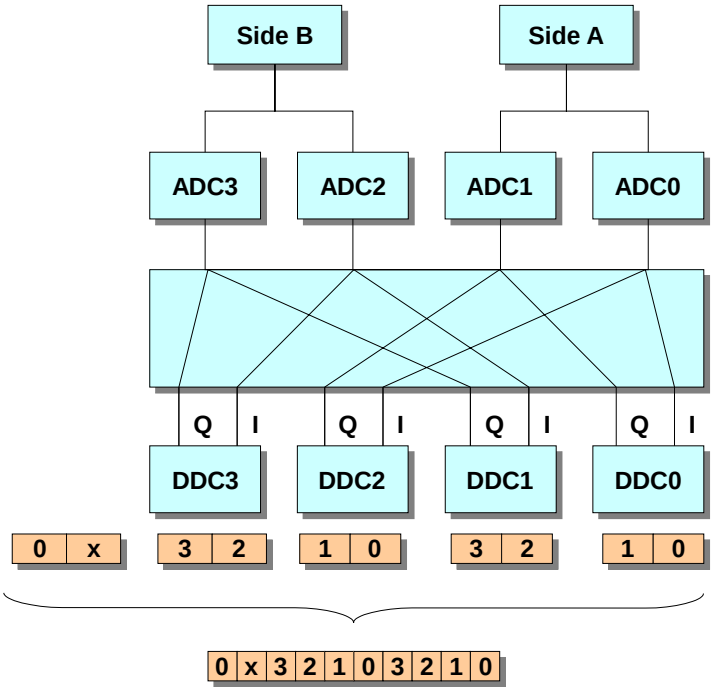
Om det inte fungerar att köra vissa färdiga exempel så är ett tips att testa att ändra på de inställningar som går att göra och leka sig fram till något som fungerar. Oftast finns ingen ultimat inställning som fungerar för alla alltid. Dessa kommandon kan man få fram i de flesta filer om man skriver in kommandot --help i skalet.

Det är också viktigt att använda den senaste släppta versionen av GNU Radio. Det finns en anledning till varför gammal kod inte längre används och det är för att den inte fungerade på det sätt som var tänkt. Det är också bra att hänga med på de diskussionsforum som finns eller att skriva upp sig på mejl-listan till GNU Radios diskussionsforum.

Sist men inte minst är det bra att ha i åtanke att även om det går att göra matematiskt betyder det inte att det kommer att bli något vettigt i GNU radio.

A.2 Multiplexer

Genom multiplexern definieras vilken AD-omvandlare som ska kopplas ihop med vilken DDC. Detta görs genom kommandot set.mux(*hexatal*), där hexatal är det hexadecimala tal som anger kopplingen. Varje hexadecimalt tal ska tolkas som vilken AD-omvandlare informationen ska koppa ifrån och ordningen i vilken talen står har betydelsen vilken DDC som kopplingen sker till. Det tal som står längst till höger är alltså I ingången på DDC 0.



Figur 19. Multiplexerinställning.

Referenser

[1] Jeffrey H. Reed, Software Radio – A Modern Approach to Radio Engineering, 2002, ISBN 0-13-081158-0

[2] Eric Blossom , Software Radio, jjuni 2007, http://comsec.com/software-radio.html

[3] John Pike, Joint Tactical Radio Systems Programmable, Modular Communications System, juni 2007, http://www.globalsecurity.org/military/systems/ground/jtrs.htm

[4] Åsa Waern, Bengt Lundborg, Elisabeth Löfsved, Gunnar Eriksson, Johan Öhgren, Jouni Rantakokko, Magnus Pettersson, Per Sakari, Slutrapport for projektet KOMET, december 2005, ISSN 1650-1942

[5] Eric Blossom, Exploring GNU Radio, juni 2007, http://www.gnu.org/software/gnuradio/doc/exploring-gnuradio.html

[6] Matt Ettus, USRP Brochure, juni 2007, http://www.ettus.com/Download.html

[7] Matt Ettus, USRP Datasheet, juni 2007, http://www.ettus.com/Download.html

[8] Matt Ettus, RFX-series Datasheet, juni 2007, http://www.ettus.com/Download.html

[9] Dawein Shen, Tutorial 3: Entering the world of GNU Software Radio, juni 2007, http://www.nd.edu/~jnl/sdr/docs/tutorials/3.html

[10] Dawein Shen, Tutorial 4: The USRP Board, Dawein Shen, juni 2007, http://www.nd.edu/~jnl/sdr/docs/tutorials/4.html

[11] Lee Pucker, Tutorial: SDR meets MIMO or, all you need to know about designing MIMO with a software-defined radio, juni 2007, http://www.wirelessnetdesignline.com/howto/184400718

[12] Lars Ahlin, Jens Zander, Principles of Wireless Communications, 1998, ISBN 91-44-00762-0

[13] S.M. Alamouti, A Simple Transmit Diversity Technique for Wireless Communications, oktober 1998, IEEE Journal on Selected Areas in Communications, Vol. 16, Issue. 8

[14] Hugo M. Tullberg, Maria Asplund, Mikael Alexandersson, Spectral Properties of an SDR using Universal Software Radio Peripheral, september 2007, Conference on RF Measurement Technology for State of the Art Production and Design, Gävle, Sweden

[15] Eric Blossom, Status Report #44, december 2004, http://comsec.com/gnuradio-status/status-2004-12-05

[16] Philip Balister, Jeffrey H. Reed, USRP hardware and software decription, juni 2007, http://www.ece.vt.edu/swe/chamrad/crdocs/CRTM09_060727_USRP.pdf

[17] Thomas Sundquist, Waveform Development using Software Defined Radio, 2006, LITH-ITN-ED-EX--06/005—SE