Faculty of Sciences and Technology

Department of Informatics Engineering

Physical-Layer Security with Interference Generation by the Receiver

André Horácio Carvalho Silva

Dissertation in the context of the Master in Informatics Security, advised by Professor João Vilela and Professor Marco Gomes. Presented to the
Faculty of Sciences and Technology / Department of Informatics Engineering.

June 2020

UNIVERSIDADE Ð
COIMBRA

This page is intentionally left blank.

Acknowledgements

I would like to firstly thank to my advisors Prof. João Vilela and Prof. Marco Gomes for the opportunity to work in their project and their guidance. Thanks to the Instituto de Telecomunicações and to the Centre for Informatics and Systems of the University of Coimbra for facilitating the installations and the used hardware which was crucial for developing this thesis.

I also have to deeply thank my parents, brother and girlfriend, who always encouraged me in the worst days, and have always been there for me. They will always be in my heart.

This work was funded by the European Regional Development Fund (FEDER), through the Regional Operational Programme of Lisbon under Grant POR LISBOA 2020, by the Competitiveness and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 framework [Project 5G with Nr. 024539 under Grant POCI-01-0247-FEDER-024539], and by FCT/MCTES through national funds and when applicable co-funded EU funds under the projects UIDB/EEA/50008/2020, PES3N (SAICT-45-2017-POCI-01-0145-FEDER-030629), SWING2 (PTDC/EEI-TEL/3684/2014).

This page is intentionally left blank.

Abstract

Secrecy against illegitimate actors always been a concern mainly with the growth of wireless devices, since it is not possible to direct a transmission to a given single receptor. Though modern cryptography helps to ensure secrecy in a communication it settles on the premise of "computer power infeasible" to rely on the impossibility of breaking some cryptographic scheme. The information-theoretically secure schemes are unbreakable, however, are unpractical due its constraints. That said, in the past few years there have been efforts on physical layer security to act as a complement of the already existing and implemented modern cryptographic schemes in order to improve security of a communication.

Recently it was proposed a coding for secrecy scheme called as Scrambled Coding for Secrecy with a Hidden Key (SCS-HK) which relies on scrambling the information with a random key, followed by encoding them together with an error correction code. Some of the bits of the resulted codeword are erased before the transmission to guarantee that only the legitimate receiver can correctly recover the message, and therefore, recover the key to descrambler into the original information. This scheme takes into account there is a wiretap channel, which assumes a better signal-to-noise ratio of the legitimate receiver's channel than the eavesdropper's channel, which enables the legitimate receiver recover the erased bits.

This advantage of the legitimate receiver over illegitimate one is difficult to assure all times, leading to the emergence of cooperative jamming schemes, where another transmitter helps to degrade the eavesdropper's channel. However, this type of schemes has some drawbacks like synchronization, willingness of cooperation by the helper since he has to spend his own energy to make others secure and also can degrade the legitimate receiver's channel. In this work, we address these drawbacks by proposing the use of the receiver as interference generator and consequently increasing security on a communication. Additionally, this removes a third-party dependency (resolves the willingness for cooperation), helps to solve the synchronization problem and it facilitates the removal of interference, because the receiving device knows the characteristics of the interference signal being generated. In this work is used GNU Radio (GR) and the Universal Software Radio Peripherals (USRP) B210 to develop the envisioned scheme.

In this thesis it is presented: 1) Design and implementation of an adapted SCS-HK scheme; 2) Implementation of an interference generation scheme by the legitimate receiver, and of the synchronization and Self-Interference (SI) cancellation mechanisms/algorithms (the Least Mean Squares (LMS), Normalized Least Mean Squares (NLMS), Recursive Least Squares (RLS) and QR Decomposition Recursive Least Squares (QRD-RLS) are studied); 3) Experimentation and evaluation of the SCS-HK scheme and the noise generation by the receiver, independently and combined, for a set of scenarios (one fair, one advantageous to the legitimate receiver and another advantageous to the illegitimate receiver).

Keywords

Software defined radio; GNU-Radio; Transceiver; Physical layer security; Wiretap channel; Coding for secrecy; Forward Error Correction Codes; Scrambler; Jamming; Cooperative Jamming; Self-interference; Full-duplex; Self-interference cancellation; LMS Filter, NLMS Filter, RLS Filter, QRD-RLS Filter.

This page is intentionally left blank.

Resumo

O sigilo contra recetores ilegítimos sempre foi uma preocupação principalmente com o crescimento de dispositivos sem fio, dado que não é possível direcionar uma determinada comunicação para um recetor específico. Embora a criptografia ajude a garantir o sigilo numa comunicação, baseia-se na premissa de "poder computacional inviável" para quebrar a segurança de um determinado esquema criptográfico. Os esquemas que de facto são inquebráveis são na realidade impraticáveis devido às suas restrições. Dito isto, nos últimos anos, foram feitos esforços na segurança a nível da camada física para atuar como um complemento aos esquemas criptográficos já existentes e implementados no intuito de melhorar a segurança de uma comunicação.

Recentemente, foi proposto um esquema de codificação para sigilo denominado Scrambled Coding for Secrecy with a Hidden Key (SCS-HK) que se baseia na mistura de uma mensagem utilizando uma chave e codificando os dois utilizando um código de correção de erros. Alguns dos bits do código resultante são apagados antes da transmissão para garantir que apenas o recetor legítimo possa recuperar corretamente a informação, e portanto, recuperar a chave para descodificar a informação original. Esse esquema leva em consideração a existência de um canal de escuta, isto é, assume uma melhor relação sinal-ruído no canal para o recetor legítimo do que o canal para o recetor ilegítimo, o que permite o recetor legítimo recuperar os bits apagados.

Essa vantagem do recetor legítimo sobre o ilegítimo é difícil de garantir todas as vezes, pelo que emergiram esquemas chamados de interferência cooperativa, onde outro transmissor ajuda a degradar o canal do recetor ilegítimo. No entanto, esse tipo de esquemas apresentam algumas desvantagens como a sincronização, a disposição de cooperação pela parte do ajudante dado que gasta a sua energia para deixar uma comunicação de outrem segura, e ainda, pode degradar o canal do recetor legítimo. Neste trabalho, abordamos esses problemas propondo a utilização do recetor como gerador de interferência e consequentemente aumentando a segurança na comunicação legítima. Além disso, isto elimina a dependência de terceiros (resolve a disposição para a cooperação), ajuda a resolver o problema da sincronização e facilita a remoção da interferência porque o recetor sabe as características do sinal de interferência gerado. Neste trabalho é utilizado o GNU Radio (GR) e as Universal Software Radio Peripherals (USRP) B210 para desenvolver o esquema visionado.

Nesta tese é apresentado: 1) Design e implementação de uma adaptação do esquema SCS-HK; 2) Implementação de um esquema de geração de interferência por parte do recetor legítimo, a sincronização e mecanismos/algoritmos de cancelamento da interferência própria (são estudados o Least Mean Squares (LMS), o Normalized Least Mean Squares (NLMS), o Recursive Least Squares (RLS) e o QR Decomposition Recursive Least Squares (QRD-RLS)); 3) Experimentação e avaliação do SCS-HK e da geração de ruído pelo recetor, de forma independente e combinada, para um conjunto de cenários (um justo, um vantajoso para o recetor legítimo e outro vantajoso para o recetor ilegítimo).

Palavras-Chave

Rádio definido por software; GNU-Radio; Transmissor; Segurança na Camada Física, Canal de Escuta; Codificação para segurança; Códigos de Correção de Erros; Scrambler; Interferência; Interferência Cooperativa; Interferência própria; Full-duplex; Cancelação de Interferência própria; Filtro LMS, Filtro NLMS, Filtro RLS,Filtro QRD-RLS.

This page is intentionally left blank.

Contents

Acronyms

ADC Analog to Digital Converter. 2, 9, 15, 44, 48

AES Advanced Encryption Standard. 29

AP Access Point. 40

AWGN Additive White Gaussian Noise. 13, 54, 56, 60

BCH Bose–Chaudhuri–Hocquenghem. 22, 23

BER Bit Error Rate. 1, 3, 20, 31–33, 54, 63, 69–71, 73–84

BlER Block Error Rate. 33

BPSK Binary Phase-Shift Keying. 9, 12, 13, 54

CI Confidence Interval. 70, 71, 74, 76

CPU Central Process Unit. 10, 49, 53, 69

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance. 40

DAC Digital to Analog Converter. 8

DES Data Encryption Standard. 29

DSP Digital Signal Processors. 8, 58

DVB Digital Video Broadcasting. 26

EVM Error Vector Magnitude. 1, 3, 32, 33, 69, 70, 72–74, 76–81, 84

FD Full-Duplex. 2, 4, 11, 38–41, 43, 44, 49, 69

FEC Forward Error Correction. 20, 51–53, 55, 56

FG FlowGraph. 9, 10, 49, 53, 56–65, 69, 71, 72, 75, 84

FIR Finite Impulse Filter. 45, 54

FLL Frequency-locked loop. 16

FPGA Field-Programmable Gate Array. 8–10

GF Galois Field. 23, 24

GPP General Purpose Processors. 8

GPS Global Positioning System. 7

GR GNU Radio. v, vii, 3, 4, 9–12, 14, 15, 22, 27, 44, 49, 51, 52, 55, 58, 59, 61, 63–65, 70, 71, 83

GSM Global System for Mobile communication. 7

GUI Graphical User Interface. 9

HD Half-Duplex. 39–41, 43

ICS Interleaved Coding Secrecy. 34

ICS-HK Interleaving Coding for Secrecy with a Hidden Key. 34

IDE Integrated Development Environment. 4

IEEE Institute of Electrical and Electronics Engineers. 7

ISI Inter-Symbol Interference. 14, 41, 54

ITU-T International Telecommunication Union-Telecommunication Standardization Sector. 25

LDPC Low-Density Parity-Check Code. 1, 21, 22, 33, 35, 51, 52, 56, 63, 64, 69, 75, 80, 84

LFSR Linear-Feedback Shift Register. 26, 51, 56, 64, 65

LMS Least Mean Squares. v, vii, 4, 44–48, 59–61, 72, 75–77, 83

LPF Low Pass Filter. 18

MCR Master Clock Rate. 58

MIMO Multiple-Input and Multiple-Output. 11, 39, 69

MMSE Minimum Mean Squared Error. 2, 44–46

MSB Most Significant Bit. 51

NLMS Normalized Least Mean Squares. v, vii, 4, 46, 48, 59, 75, 76, 83

OoT Out-of-Tree. 10, 56, 57, 64, 65

OSI Open System Interconnection. 36, 40

PL Packet Loss. 70

PLL Phase-locked loop. 15, 17

PLR Packet Loss Ratio. 1, 3, 32, 33, 40, 41, 69, 71, 73–84

PSK Phase-Shift Keying. 12, 18, 54

QPSK Quadrature Phase-Shift Keying. 9, 12, 16, 35, 54, 55, 58

QRD-RLS QR Decomposition Recursive Least Squares. v, vii, 4, 48, 59, 75–77, 83

RC Raised Cosine. 14

RLS Recursive Least Squares. v, vii, 4, 44, 46–48, 59, 75–77, 83

RRC Root Raised Cosine. 14–17, 35, 52, 54, 55, 64

RS Reed-Solomon. 22–24, 64, 65, 73

RSA Rivest-Shamir-Adleman. 3, 30

RX Receiver. 11, 19, 20, 25, 29, 35, 39–43, 49–51, 53, 54, 56–59, 63, 70, 72

SCS Scrambled Coding for Secrecy. 34

SCS-HK Scrambled Coding for Secrecy with a Hidden Key. v, vii, 2–5, 34, 35, 38, 49–51, 57, 60, 61, 64, 65, 67, 69, 72, 80–84

SDR Software Defined Radio. 2–4, 7–12, 35, 44, 67, 69, 70, 72, 77, 84

SI Self-Interference. v, 2–4, 11, 38–45, 48, 49, 59–61, 67, 68, 70, 72, 73, 75–79, 83

SINR Signal to Interference and Noise Ratio. 39, 41, 42

SNR Signal to Noise Ratio. 14, 31–33, 36, 40, 47, 55, 68

SoC System on Chip. 8

SPC Single Parity Check. 21

SPS Samples Per Symbol. 14, 16, 17, 54, 55, 61

SSL Secure Socket Layer. 30

TLS Transport Layer Security. 30

TX Transmitter. 11, 14, 18, 19, 25, 29, 39–43, 49, 50, 52, 55–61, 63, 70, 72

UHD USRP Hardware Driver. 9, 59

USRP Universal Software Radio Peripheral. 3, 4, 9–11, 15, 38, 49, 57–59, 69, 72, 83

VCO Voltage-Controlled Oscillator. 18

List of Figures

List of Tables

This page is intentionally left blank.

Chapter 1

Introduction

Since the primordial times secrecy was always taken into concern for some more important subjects, for example on the letters exchanged between generals on war in roman's time using Caesar's cipher. In the past few decades more and more people communicate through the internet for different needs like messaging, voice calls, emails, among others. This communication called for secrecy mechanisms that evolved to the modern cryptography which is implemented on upper protocol layers, by using symmetric and asymmetric cryptography [1] which settles on the premise of "computer power infeasible" for a given scheme; however, with the computers power growth over the past years what was previously infeasible can be now breakable.

On an alternative perspective, the notion of "perfect secrecy" was laid out by Shannon [2]. However, due to the restrict requirements of that notion, Wyner [3] postulated the relaxed version of "weak secrecy" where he was able to secure a communication against an eavesdropper by using only wiretap codes considering a channel model named as "wiretap channel" that assumes a worst eavesdropper's channel than the Bob's channel. He also sought to maximize the transmission rate, while keeping some controlled information leaked to the eavesdropper, guaranteeing both reliability and information-theoretic security against the eavesdropper, called as "secrecy capacity" of the wiretap channel.

In many cases this notion of secrecy was insufficient, thus Maurer [4] strengthened Wyner notion by introducing the "strong secrecy" that requires the message to be random and uniform distributed over the alphabet. However, in a realistic world this rarely happens, thus, leading to the proposal of an even stronger notion of secrecy that takes into account the distribution, the "mutual information security" [5]. These notions also work as metrics to the secrecy of a communication, along with others like Bit Error Rate (BER) [6], Error Vector Magnitude (EVM) [7], Packet Loss Ratio (PLR), and others [5].

Considering the wiretap channel model some work was developed in order to meet the restricted requirements by either having a noiseless main channel or a wiretap channel with less quality than the main one. So, more realistic and achievable coding schemes were developed like using Low-Density Parity-Check Code (LDPC) characteristics along with puncturing [6]. Latter, taking secrecy into account scrambling [8] or interleaving [9] was used in order to not directly expose the sent information.

These schemes require an eavesdropper to have a degraded channel with respect to the main/legitimate channel which is not controllable. Thus, other schemes emerged by using a third actor (Charlie) acting as an helper along with jamming [10] as a way to degrade the wiretap channel allowing advantage of legitimate actors (Alice as transmitter and Bob as receiver) over illegitimate ones (Eve as receiver). This is called as cooperative jamming which can be subdivided in three main categories:

- The cooperative jamming by Gaussian noise [10] uses Charlie closer to Eve than to Bob and generates Gaussian noise degrading more Eve's channel.
- The cooperative jamming by structured codes [11] in which the jam signal between Alice and Bob has a defined known code structure where Bob takes advantage of the knowledge of that code to retrieve the information sent by Alice, while Eve cannot do it since the code is mixed with the information.
- The cooperative jamming by alignment [12] uses more than one helper and takes as advantage the jamming signal being at the same space as the sub-message on Eve but not on Bob.

All these schemes have several problems namely in terms of synchronization, the willingness of the helper and his availability to spend his own energy to help a communication that has nothing to do with him, and even the problem of the mobility of that helper which directly affects the performance of the scheme. These disadvantages lead to the idea of setting the legitimate receiver as the helper [13, 14], this is, the legitimate receiver, at the same time it is receiving information from the Alice, emits a jamming signal to degrade Eve's channel, and cancels the Self-Interference (SI) it suffers to retrieve the information signal sent by Alice.

In order to jam and cancel it is necessary to send and receive the signal at the same time at the same frequency on the same device, this is, using Full-Duplex (FD) on wireless. The feasibility of the usage of FD was shown in [15], however introduces some drawbacks [16] where the main one already referred, is SI; the receiver antenna besides receiving the interesting signal also receives (with higher power [17]) its own generated jamming signal. Thus, the SI cancellation is the main role to successfully get to a FD scheme working which can be done in three main categories [16]:

- The passive suppression is based on attenuating the SI signal by physically separate/change somehow the transmitter antenna and the receiver antenna of the device [18].
- The analog SI cancellation it is performed before the signal goes into Analog to Digital Converter (ADC) and can be done with some known noise canceling integrated circuits, like the QHx220 [15].
- The digital SI cancellation is performed after ADC and takes the signal as knowledge to cancel it by using algorithms like least-square and Minimum Mean Squared Error (MMSE)-based techniques [16].

1.1 Objectives

Security is a requirement that is present on every technological interaction. In order to improve secrecy at physical layer of a communication within the presence of an eavesdropper, this thesis focuses on design and implementation in a Software Defined Radio (SDR) device of a FD transceiver that uses an already known coding scheme called Scrambled Coding for Secrecy with a Hidden Key (SCS-HK) combined with noise generation by the legitimate receiver. This, cancels the SI based on the knowledge of the jamming signal created by him and degrades the illegitimate receiver's channel in order to achieve both reliability of the communication between Alice and Bob and at the same time secrecy against an eavesdropper.

The interference signal generated by the legitimate receiver creates a "bubble" where, within it, the illegitimate device receives jamming signal with higher power than the legitimate information. Additionally, the interference generation mechanism consumes a lot of energy, thus, in a real-world scenario, the legitimate receiver should be connected to a power supply. A motivating real-world scenario possible is a payment with a contactless card (acting as transmitter) in a payment terminal (acting as legitimate receiver) connected to power supply and there is a illegitimate receiver trying to retrieve the information sent by the card. Note that the "bubble" created by the terminal protects the information transmitted by the legitimate card by generating interference on the illegitimate one. This situation is already protected with modern cryptography, in particular, the Rivest-Shamir-Adleman (RSA) algorithm. The mechanism of this thesis aims to be a complement layer of security at physical level, increasing the security with jamming interference; therefore the illegitimate card will have two layers of security to break (one software and one physical).

Furthermore, for the SDR implementation it will be used the Universal Software Radio Peripheral (USRP) B210 hardware from Ettus Research together with GNU Radio (GR) (as software component). From this goal also come the purpose of test and evaluate different real-world scenarios changing actor's positions and transmission power to seek the best possible scenario and the best algorithms/mechanisms taking into account different metrics (EVM, PLR and BER).

A Gantt diagram illustrates a project schedule relating several tasks to fulfill with the respective dates. Both planned and actual diagrams for the first semester are depicted in Fig. 1(a) and Fig. 1(b) in Appendix A. In the same way, the planned and actual diagrams for the second semester are depicted in Fig. 2(a) and Fig. 2(b) in Appendix A. The main differences are in the time to fulfill the tasks and the new unforeseen ones such as writing the tutorial, REC paper and CSNDSP paper, as will be mentioned next.

1.2 Contributions

It is important to mention that all the work of this thesis was developed firstly on the Centre for Informatics and Systems of the University of Coimbra (CISUC) within the project Securing Wireless Networks with Coding and Jamming (SWING2) and then in the Instituto de Telecomunicações (polo of Coimbra), within the project Power-Efficient Solutions for Secure wireless Sensor Networks (PES3N).

In line with the objectives above mentioned the main contributions are:

- It was developed a tutorial explaining the GR utilization and the setup of a basic communication between two USRPs for a beginner on this field. Presented in Appendix D.
- It was proposed an adaptation on the SCS-HK scheme.
- The design and implementation of the adapted SCS-HK on GR both in simulation and real-world prototype by using USRPs.

- Creation of GNU Radio Out-Of-Tree Blocks that can be used for other works (Insert Vector, Scramble Packetize/Descrambler Packetize, Additive Scrambler With Key/Additive Descrambler With Key, Multiplicative Scrambler with Key/Multiplicative Descrambler with Key, Synchronization and Detect Access Code), that were made available to the community GitHub [19].

- Design and implementation of a full-duplex scheme with interference generation by the legitimate receiver along with the usage of adaptive algorithms to perform SI cancellation both in simulation and in real-world prototype by using USRPs.

This work contributed also with the following publications:

- A. Silva, M. Gomes, J. Vilela, "SDR Implementation of Full-Duplex Jamming for Secrecy", REC 2020, February, 2020 - presented in Appendix C;

- A. Silva, M. Gomes, J. Vilela, "SDR Testbed of Full-Duplex Jamming for Secrecy", CSNDSP 2020, June, 2020 - presented in Appendix B.

1.3 Organization

This document is organized in 6 chapters. In the chapter 1 it has been presented the scope of the thesis, the challenges addressed and a resume of the main outcomes. In the second chapter we make an introduction to SDR along with the available Integrated Development Environment (IDE) tools for designing the radio signal managing in the software component of the SDR transceiver and our reasons for choosing GR. Then we will specify all the setup that has been used in the implementation of all the work. The communication fundamentals of a basic transmitter will be presented, explaining how the data is encoded, modulated, transmitted and recovered from the distortion effects added by the channel. It is also addressed mechanisms of recovering from errors in the transmission of data, and finally how scrambler works and why it is used for.

After getting all the background need then we are ready to debate about the beginning of the modern cryptography and to understand the differences and its disadvantages related to physical layer security, which is done in Chapter 3. In this chapter is introduced the wiretap channel which is the basis of several already known security schemes (some of them also here discussed) along with the metrics used for the evaluation of these schemes. Then we will introduce the role of jamming in a way to enhance security which is the main focus in this thesis and it will be addressed what is full-duplex along with its advantages and disadvantages. Yet related to jamming, cooperative jamming using a third actor is presented as a way to improve secrecy against an eavesdropper as well as its main drawbacks which leads to self-interference generation and its cancellation on the receiver using full-duplex.

The SCS-HK coding scheme GR implementation will be discussed in Chapter 4 both in simulation as well as in a real-world prototype leveraged by USRPs. Thereupon this scheme, it is presented and described the implementation details of the proposed FD jamming where the receiver acts as jammer and cancel the SI by using one of several adaptive algorithms that are here analyzed (Least Mean Squares (LMS), Normalized Least Mean Squares (NLMS), Recursive Least Squares (RLS) and QR Decomposition Recursive Least Squares (QRD-RLS)).

Finally, the results are presented in Chapter 5, where it will be first evaluated the impact of using passive suppression and header codification. Then, an analysis of the adaptive algorithms will be assessed considering three different setups where the actors positions are changed. Concluding with its enhancement by adding the SCS-HK scheme for different values of bits to be punctured. The Chapter 6 concludes this thesis reviewing the work done, the most important results achieved and future work.

This page is intentionally left blank.

Chapter 2

Background

2.1 Software Defined Radio

Everyday, more and more people communicate with each other in different ways for different needs, for instance, communications with phone or internet calls, messaging, socializing in social networks, emergency calls, among several other ways and needs. When people are interacting with a phone or a laptop either by Wi-Fi or Global System for Mobile communication (GSM), in reality, they are transmitting and receiving data which is created by them or another terminal. A radio is a device that can transmit/receive signals in electromagnetic waves with information encoded in it, making all these interactions possible.

Software Defined Radio (SDR) can be defined in many ways. One of them was introduced in a work of the Wireless Innovation Forum [20] in collaboration with the Institute of Electrical and Electronics Engineers (IEEE) where it was defined as "Radio in which some or all of the physical layer functions are software defined" [21]. "Software defined" means, in short, that the mechanisms previously statically placed on hardware are now implemented in software. Thus, SDR is a system for radio communication where several components (like modulators/demodulators, filters, amplifiers, equalizers, among others) are now implemented in software in place of hardware, hence, much more configurable and useful for many reasons, like research, personal prototypes among others.

Unlike conventional communication systems, such as common phones, which are limited in functionality and can only be modified through physical intervention, which results in low flexibility and low support for multiple waveform standards, the SDR is easily reconfigurable. For example, if we run a specific "software component" it can be used as Global Positioning System (GPS), however if it is changed to another "software component" can in seconds become a GSM receiver, and seconds later can become a Wi-Fi transceiver. Concluding, this bring us high flexibility to research, keeping a low cost and a portable device without using too much power.

In a more practical perspective, SDR have been deeply placed in military defense industry on several countries (e.g. Harris Corporation in US, Thales in France, Leonardo in Italy, Elbit Systems in Israel, among others [22]) with limited market for commercial applications. However, in the past few years, SDR usage on other commercial market areas emerged as a way to design, develop and validate systems that thereafter are expensive to produce, save them a lot of money in these phases. For instance, ZTE developed a SDR base station in order to "support multiple wireless standards with minimal software upgrade". The platform helps operators significantly reduce their Total Cost of Ownership, as mentioned on their website [23]. Recently, Huawei launched a solution to tackle the 5G challenges, called SingleRAN PRO which derives of the SingleRAN. This solution allows mobile communications operators to support multiple standards (2G, 3G, 4G and 5G) on a single network by using SDR [24]. Another example is on the automotive industry using the SDR on the concept "Vehicle-to-X" (communication between a vehicle with another vehicle or infrastructure) [25]. These devices usually includes several components for different needs and specialization, such as a Field-Programmable Gate Array (FPGA) for more specific processing, General Purpose Processors (GPP), a Digital Signal Processors (DSP), and/or a System on Chip (SoC) [21]. All these components allow new future standards and features to be implemented without requiring an hardware intervention, i.e. without the need to modify, change or add a new hardware component.

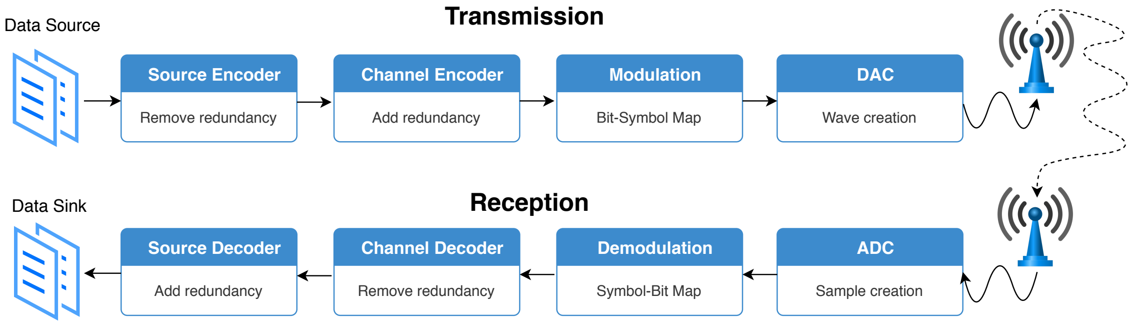


Figure 2.1: Basic block scheme of a digital communication.

The basic building blocks of a digital communication are represented in Fig. 2.1. This shows how binary data is treated and transformed until being transmitted through the antenna and then the reverse operation to retrieve the sent data [26]. A communication is represented by:

- Data Source/Sink: This is the desirable data to transmit, being for example speech, music, text files, and even video. The data is in a binary form or converted to binary by an analog-to-digital transducer, being necessary to previously agree in its representation.
- Source Encoder/Decoder: This is used for removing redundancy in the pure data optimizing the transmission. In the receiver side this redundancy may be reintroduced if needed, as for example after digital-to-analog conversion by a transducer.
- Channel Encoder/Decoder: Here is introduced controlled redundancy in the stream. This enables the detection and/or correction of errors occurred during the transmission through the physical channel making the transmitted data more resilient to noise. In the decoder, it picks the redundancy and with mathematical operations verify/recover erroneous bits of data. There are a several known codes and algorithms for this cause [27, 28].
- Modulation/Demodulation: In the modulator all the bits of data previously transformed are now mapped into symbols adapted for transmission to the characteristics of the physical channel, and then in the demodulator these symbols are mapped back into the original bits.
- DAC/ADC: This is the border between what is software and hardware. The Digital to Analog Converter (DAC) converts the discrete samples received into analog wave-

forms converting this way in analog data. In the Analog to Digital Converter (ADC) the received analog signal is then sampled into digital samples.

- Antenna: Here it is the border between the physical device and the air. The antenna receives the analog waveform previously created (a frequency up-conversion may be needed) and transmits it over-the-air. On the receiver, other antenna gets the waveform and sends the signal to the ADC.

2.1.1 Available Tools

There are different tools to work with the software component in a SDR device such as:

MATLAB & Simulink: The MATLAB [29] is a paid software developed by MathWorks which combines iterative analysis with a programming language based on matrix and arrays calculus, being a quick learning and fast running language, thus being widely used by engineers. Includes a live editor where it is used for writing the desired script. The Simulink [30] is a plugin that makes possible designing and simulating the system quickly and intuitively by enabling the creation of the design by blocks and a set of connections between these blocks. This plugin picks the block design and automatically generates the code into `C/HDL` deploying directly in the FPGA processor. This tool has the big advantage of having a detailed documentation with practical examples and has a dedicated support team.

LabView: The LabView [31] is another paid software created by National Instruments which is owner of Ettus Research who dominates the SDR market. Although it is more directed to automation, it is flexible and can be used for this purpose. Here the logic is represented on a diagram creating the system design. It has the advantage to allow directly program on FPGA, thus, being faster at signal processing level.

GNU Radio: The GNU Radio (GR) [32] is a free and open-source software to work with signal processing, where it is possible to simulate all steps by creating a FlowGraph (FG) by positioning blocks and connecting them. There exists some stock blocks to do predefined algorithms, and if a specific algorithm has not yet been implemented, it is possible to create new blocks using `Python` or `C++` (preferable `C++` for being the fastest language) and use `XML` to link the source code to the Graphical User Interface (GUI). This is the main problem for some users because it requires a lot more effort to code comparing with other tools.

In [33] a comparison between LabView and GNU Radio is fulfilled regarding the stock blocks implemented and its advantages and disadvantages. In short, it states that the GR allows plotting and sliding features in real time, comparing with LabView, which is extremely useful for setup a communication from scratch. When considering the FG's differences between these two tools, in [34] is depicted an implementation of Binary Phase-Shift Keying (BPSK)/Quadrature Phase-Shift Keying (QPSK) transmission on both software mentioned. Clearly, the GR tool is more user friendly in the GUI, both in the generated plots and in the FG. To deploy the project using the SDRs it is mandatory to have a proper driver considering the hardware that is being used, for instance, for the Universal Software Radio Peripheral (USRP) it is required use the USRP Hardware Driver (UHD) driver [35]. In this subject, the GR has the better driver support on USRP than other solutions, as there are a specific team from Ettus Research to help both the GR's development and the problems faced by users.

As the GR uses `C++` for the core processing it has the advantage of being a fastest tool, thus, it is the main choice in SDR community, leading to the existence of big conferences where different aspects and projects using this software are discussed. There are two main problems about this software, namely the lack of documentation to help the user starting and creating his own FG (information about the functioning of these blocks is dispersed). Also, the continuously improving and changing of the stock blocks causes some blocks to be deprecated and hence an outdated FG, needing a constant update by the FG's author of the used blocks.

2.1.2 GNU Radio Choice and Internal Working

Our choice about the tool will have two main requirements: The possibility of creating our own code for algorithms not yet implemented and getting the fastest throughput possible leading to the possibility of streaming a video. As already discussed in Section 2.1.1 the GR has in the core the `C++` as programming language being a fast language that will be used in complex algorithms. Besides that, it is also possible to create Out-of-Tree (OoT) Modules where we can program our own code in `Python` or `C++`. Taking this into account and also our previous experience in `C`, we have selected the GR for creating this project.

In the GR internal's working, the data is treated as a stream which has a sample rate defined in a "Source" block. The data flows through a connection and is feed into one block that processes the data. Then the data is output which flows to another block using connections, and so on until get to a "Sink" block.

When we do data streaming we are implicitly using buffers to hold the data between blocks. This buffers are of finite size, and as so, before the block does anything, it checks the amount of available data in input buffer and space to write in the output buffer. If there is not enough data or space available in such buffers the block does not do anything. This is called "Back-pressure" because it is pressured back the data that is incoming, slowing down the flow. This is how GR controls the data flow, giving data to the block when it is necessary and holding it when block is busy working.

There are source/sink blocks that produces/consumes data at a given fixed sample rate, as for example the "Audio" and "USRP" blocks. These blocks are called "clock" blocks, and a FG has typically only one source or one sink block, and not both because that usually leads to a problem called as "2 Clock Problem". This problem happens when these two blocks have different fixed sample rates leading to an asynchronous clock sources which causes "Underruns"/"Overflows" depending on the difference between the production and consumption rates. When none of this "clock" blocks is used then it is required to use the "Throttle" block to act as a clock. Without it, the FG will run as fast as the Central Process Unit (CPU) can process resulting in non-responsive program and other problems like hiccups. This means that while it is in simulating mode it is necessary to use the "Throttle" block to control the flow, although when we go to a real transmission this block is no longer required.

2.1.3 Setup Specification

In this project it will be used the SDRs USRP B210 [36] with VERT 2450 [37] antennas attached, both from Ettus Research. This USRP was designed to low-cost experimentation and it can be used in two ways, i.e. either all the processing is done in the computer and sent the data through high speed USB 3.0 (using the tools mentioned above), or it can be directly programmed in its FPGA (a Spartan 6) which enables higher speeds. Since all the work will be used the GNU Radio, the first possibility is chosen.

The main advantages of this hardware are the large frequency coverage from 70 MHz to 6 GHz and support Full-Duplex (FD) and Multiple-Input and Multiple-Output (MIMO) transmission. More specifically, it contains two inputs and two outputs for the antennas which enables the implementation upon a single USRP platform the work we proposed to: set the legitimate receiver acting as jammer and being able to cancel its Self-Interference (SI).

2.2 Basic Communication Concepts

A communication always assumes the existence of two sides: the Transmitter (TX) that manages to adapt the information to be sent to the characteristics of the transmission channel, making the transmission possible; and one Receiver (RX) where the sent data is received (wireless or not) in analog/digital information and recovers the original information from there. The main block components of these for SDR implementation are depicted in the Fig. 2.2.

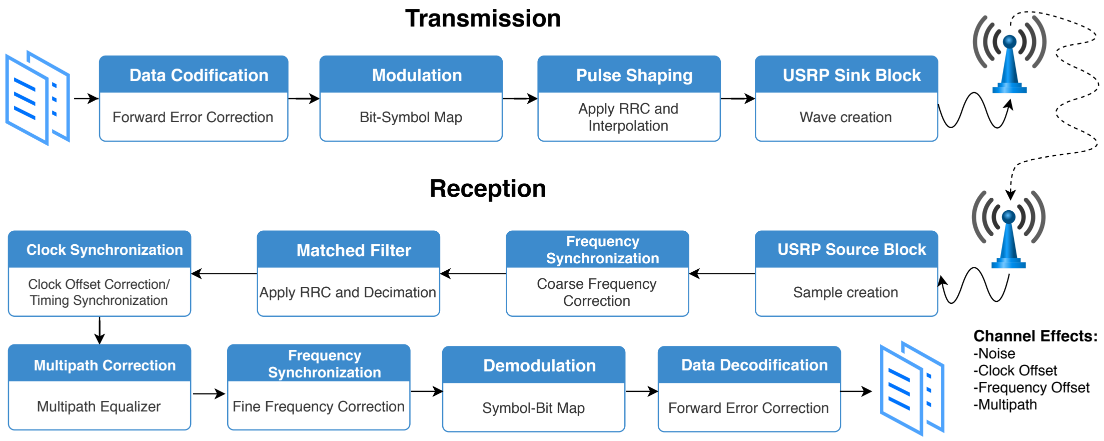


Figure 2.2: General concepts of a wireless transceiver.

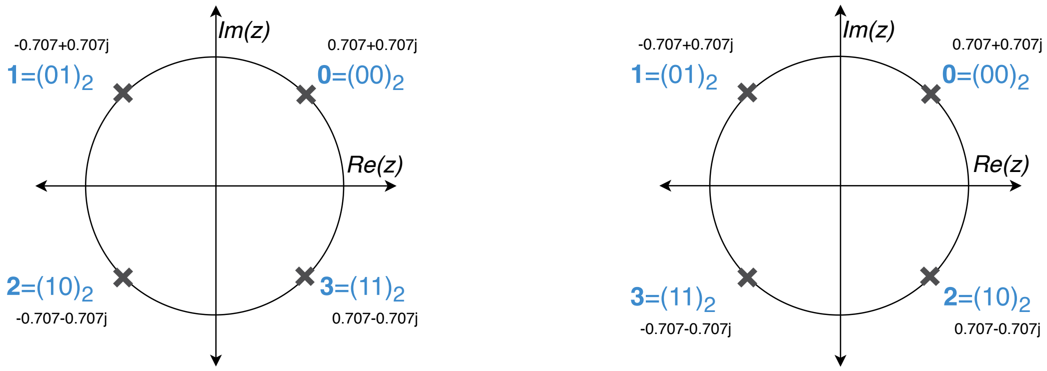
In TX side the goal is to pick the desired binary data (after source encoding) and make the transmission possible, thus requiring four main operations: First, the data is encoded in order to add controlled redundancy that enables the detection and/or correction of errors that may occur during transmission. Then, it is modulated where basically the binary information is mapped (using some mapping rule, usually called as constellation) into some characteristics (amplitude, phase or frequency) of the transmitting wave (called carrier) adapted to the transmission over the medium. Since the available spectrum for transmission is a scarce and expensive resource, the bandwidth of the transmission signal is usually narrowed by using a pulse shaping filter. Finally, the digital information is ready to be converted on analog one and up-converted to high frequency and sent them to the antenna by using the USRP block available in GR.

To retrieve the original information from the received signal it is imperative to successfully demodulate this. The RX side is substantially more complicated than the TX side because the channel causes distortions and added noise to the signal that it is crucial to undo them. First step is to convert the analog signal that came from the antenna to digital signal with the USRP block in GR. Then, it is necessary to take care aspects related to synchronization, firstly by grossly correcting the frequency offset, next applying a matched pulse shaping filter, then correcting the clock difference of the SDRs, followed by correcting the multipath problem and finally by performing fine frequency correction. Finally, the signal is ready for being successfully demodulated and decoded for recovering the transmitted bits.

2.3 Modulation and Pulse Shaping

The modulation is the act of convert/map bits into an electromagnetic wave (called carrier) by varying one or more of its properties in a manner adapted to the medium of transmission (wireless, optic, copper, among others). One popular form of modulation is called as Phase-Shift Keying (PSK), where the data is encoded by varying the phase of the wave keeping the amplitude and frequency unaltered. This type of modulation uses both the sine and the cosine wave together resulting in two orthogonal carriers making possible the representation of the tuple with their phase as a symbol in a complex plane, resulting in a constellation plot (on GR) with a trigonometric circle base.

The stream of bits thus mapped into symbols. The number of bits carried in each symbol depends of the modulation order M, i.e. the number of symbols of the constellation, and it is $\log_2 M$ bits. For example, with an order of $M = 2$, this is, BPSK then the phase is only changing one time (resulting in total of two phases), being encoded only one bit for each phase. In this case a symbol carries 1 bit and the constellation plot has 2 points because there are only 2 symbols possible (the symbol 0 and the symbol 1). Going for QPSK ($M = 4$), the phase changes in four possible ways, resulting in four constellation points, each carrying 2 bits. In the same way, 8PSK will result in 8 constellation points carrying 3 bits length per symbol, and 16PSK has 16 constellation points with 4 bits length per symbol.



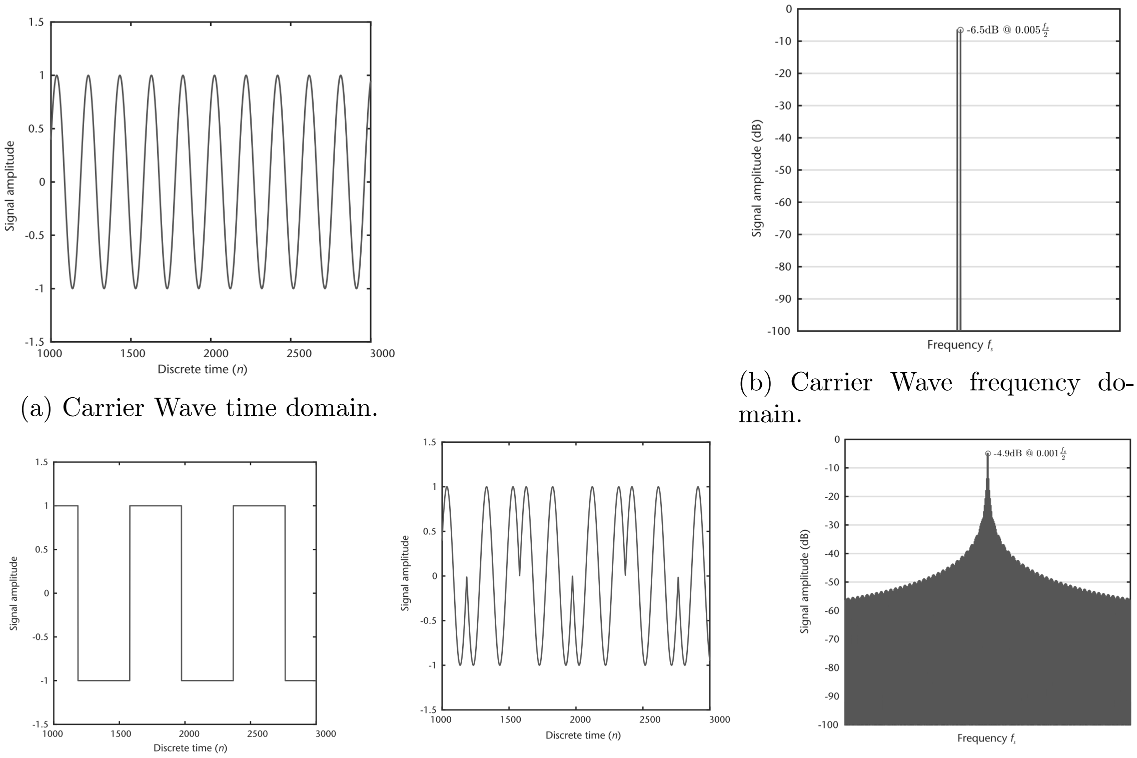
(a) Natural Mapping.

(b) Gray Code Mapping.

Figure 2.3: Difference between Natural Mapping 2.3(a) and Gray Code Mapping 2.3(b) on a QPSK constellation.

Regarding the mapping between symbol and constellation points, we can do it with "Natural Mapping" represented in Fig. 2.3(a) or use one concept called "Gray Code Mapping" represented in Fig. 2.3(b). The gray code has the advantage that neighbor symbols only differ one bit, reducing the amount of wrong bits if occur one error in the transmission. This happens because when occur one error, in the demodulator, instead mapping the right constellation point maps one of the neighbors with high probability, hence, resulting in maximum, only one-bit error instead of two or more bits (if using the natural mapping).

The step that follows modulation is pulse shaping. To understand why this is needed [26]



(a) Carrier Wave time domain.

(b) Carrier Wave frequency domain.

(c) Bit Stream.

(d) Information carrier wave in the time domain.

(e) Information carrier wave in the frequency domain.

Figure 2.4: At a simple carrier wave 2.4(a) with a narrow frequency band 2.4(b) is multiplied the bit stream 2.4(c) resulting in a ready to transmit carrier wave 2.4(d) where has an infinite frequency band 2.4(e). All images are from [26].

let's consider the case of a BPSK transmission. Take a look to the time domain plot of the carrier wave without the added bit stream in Fig. 2.4(a) where a perfect sinusoidal wave is shown and then taking a look to the frequency domain Fig. 2.4(b) we observe that a single strike appears at the frequency of the carrier. However, the spectrum of the modulated signal shown in Fig. 2.4(e) occupies a large bandwidth since the modulated signal results from the multiplication of the carrier by the bit stream shown in Fig. 2.4(c) resulting in Fig. 2.4(d) (Note that polar encoding is used with bits 0/1 being mapped to voltages $\pm 1V$). In fact the spectrum of the modulated signal is theoretically infinite due to the rectangular shape of Fig. 2.4(c). This large bandwidth brings several problems, such as:

- Costs of operation. Spectrum is a scarce resource and there is the need to buy it to a regulation entity;
- Difficult to transmit because of the large bandwidth and the physical channels (e.g. wireless) being selectively in the frequency;
- Problems of inter channel interference if transmission bandwidth exceeds the assigned one causing distortion in neighbour channels;
- More noise upon reception, since thermal noise, usually moduled as Additive White Gaussian Noise (AWGN) is proportional to the channel bandwidth.

Hopefully, according to the Nyquist criteria, the minimum bandwidth required to transmit at a symbol rate of R_s (Baud) is $\frac{R_s}{2}[Hz]$. The function of the pulse shaping is to limit the bandwidth of the channel toward the minimum Nyquist Bandwidth. The problem with narrowing the bandwidth is that the symbols become wide in the time domain and their tail will interfere with the beginning of the neighbouring symbol inducing Inter-Symbol Interference (ISI). Therefore the goal is narrow the bandwidth while keeping the lowest ISI possible. Hopefully, Nyquist has also defined a set of requirements on pulse shaping filters to avoid ISI. These are known as Nyquist filters.

A commonly used pulse shaping filter is Raised Cosine (RC) filter with frequency response [26]:

$$H_{RC}(f) = \begin{cases} 1, & \text{if } |f| \leq \frac{1-\beta}{2T_s} \\ \frac{1}{2}[1+\cos(\frac{\pi T_s}{\beta}[|f| - \frac{1-\beta}{2T_s}])], & \text{if } \frac{1-\beta}{2T_s} < |f| \leq \frac{1+\beta}{2T_s} \\ 0, & \text{if } otherwise \end{cases} \tag{2.1}$$

Where $Ts = \frac{1}{R_s}$ is the symbol period, and β is the roll-off factor that measures the excess of bandwidth with respect to the minimum Nyquist bandwidth $\frac{R_s}{2}$, thus, $\beta \geq \frac{1}{2T_s}$.

In order to maximize the Signal to Noise Ratio (SNR) upon reception, this RC filter is usually divided in two square Root Raised Cosine (RRC) ones, with frequency response $H_{RRC}(f) = \sqrt{H_{RC}(f)}$, where one is used upon transmission and other upon reception filtering out the receiving noise. This is the matched filter concept, where specifically in GR, the received signal (mixed with channel's noise) is correlated with a known delayed signal (called as template, which is the RRC filter in TX side) to detect the presence of the template recovering the signal while minimizing the ISI (with some due to the channel's distortions and discrete operations) and reducing the noise.

The digital implementation of this filters is usually done by windowing the filter impulse response considering a minimum duration of 11 symbols (center of the filter, plus the five close neighbors for each side), and by sampling the resulted signal at least at 2 Samples Per Symbol (SPS), although a higher sampling rate is typically used to grant better time resolution.

The Fig. 2.5 present a spectrum of a digital implementation on GR of RRC filters with different roll-offs.

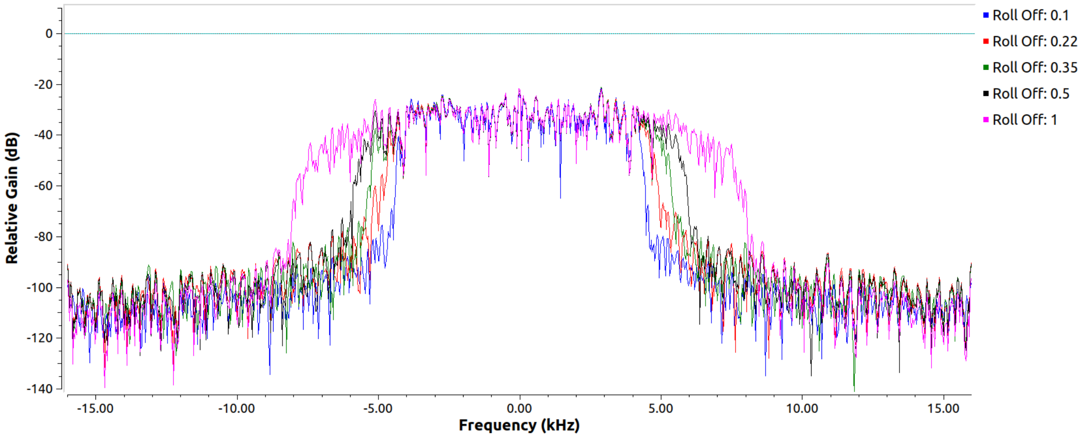


Figure 2.5: Different roll-off values in RRC filter on a GR simulation 4 SPS and 11 set as window for a random sequence of information.

2.4 Signal Synchronization

2.4.1 Clock/Timing Offset Synchronization

The first type of synchronization that it will be analyzed is timing offset, also called as clock offset. Typical communications systems (e.g. wireless) usually require time synchronization between multiple devices in order to work correctly. Remember that the analog signal is got from the antenna and sampled into digital one with the ADC inside of the USRP. This offset arises from sampling not happening in the ideal instant, creating timing offsets, resulting different samples than the ideal ones as explicitly shown in Fig. 2.6. This impacts ultimately on more disperse constellation points possible to be visualized in eye diagrams where the "eye" become more "closed" and a wrong symbol decoding may occur.

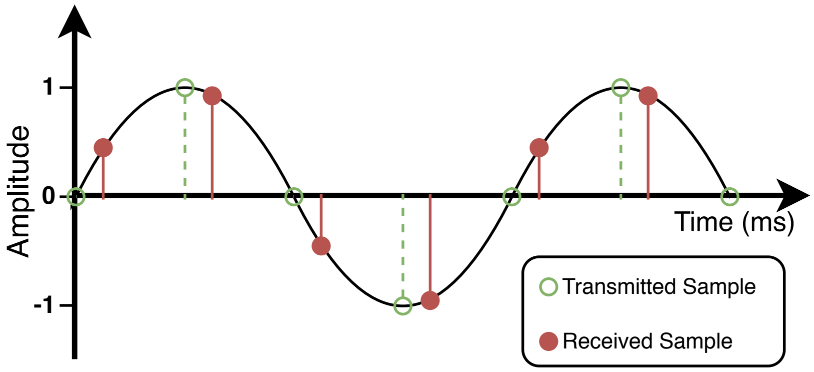


Figure 2.6: Representation on erroneous sampling due to clock offset between the transmitter and the receiver.

The timing synchronizer implemented on GR [38] uses two polyphase filterbanks. One contains the matched filter (the RRC) with different phases (i.e. delays for better resolution) and another one contains the derivatives of the first filterbank. Note that the first filterbank is the RRC "template" previously described and the peaks are known, hence, we have the knowledge of the ideal sampling points. This is conjugated with the property of the derivative of these peaks being zero, and the main objective is to align the output signal of this block to be sampled at exactly the peak of the first filterbank. Besides that, the region around the peak is relatively linear. All these facts are used to generate the error signal $\mathbf{E}[n]$. Assuming the received signal (with timing offset) is applied to the RRC filter, resulting in $X_i[n]$, and with the derivatives (of the second filterbank), $D_i[n]$, for the i-th filter, then the error $\mathbf{E}[n]$ is calculated as:

$$\mathbf{E}[n] = \frac{\Re(X_i[n]) * \Re(D_i[n]) + \Im(X_i[n]) * \Im(D_i[n])}{2} \tag{2.2}$$

$\mathbf{E}[n]$ results in a value of how far away a sample is from the zero in the derivative signal. Hence, we want push this value to zero seeking for more approximation of the ideal sample. For this, it is necessary a second order loop similar with Phase-locked loop (PLL), where it is defined two variables, the number of filters on the filterbank and how far we want to traverse the path filters to keep the receiver locked (forward or backward). When this value is increased then the loop goes further for looking the ideal sample spot, thus, increasing the amount of power needed for this operation.

2.4.2 Carrier Synchronization

Due to the transmitter and the receiver being two distinct and spatially separated devices the transmission over the channel induce a frequency or phase offset in the transmitted signal. In literature the carrier recovery is defined as carrier phase recover or carrier frequency recover, and has the same goal, which is attain a stable constellation on the output of the synchronizer, so the demodulator can correctly decode the symbols. The frequency and phase are related, so recover the frequency it is indeed equivalent to recover the phase, because the angular frequency ω (equivalent of $2\pi f$) is only a measure of a changing phase σ over time: $\omega = \frac{d\sigma}{dt} = 2\pi f$ [26].

2.4.2.1 Frequency Offset Synchronization

Taking into consideration the spectrum of a random sequence of bits with QPSK modulation and pulse shaping by RRC filter along operated 4 SPS of interpolation. If some frequency offset is added by the channel then it is possible to visualize the spectrum curve moving to right or left depending the value introduced. In Fig. 2.7 is shown the difference between before and after the frequency is added by the channel, which results in a rotation and dispersion on the points of the constellation of the received symbols that appear placed like a circle. What it is desired in the end is the constellation points well positioned and aggregated, which requires for the estimation and compensation of the frequency offset. Since this deviation can be large this procedure is usually carried in two-stages frequency recover [26] namely coarse and fine frequency recover.

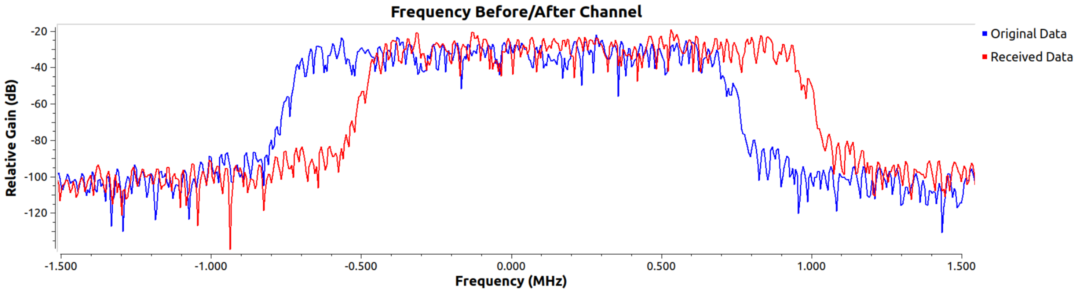


Figure 2.7: Frequency domain of the carrier wave before (in blue) and after (in red) added 0.25 MHz of frequency offset by the channel. The RRC is applied with 4 SPS and set 11 as window.

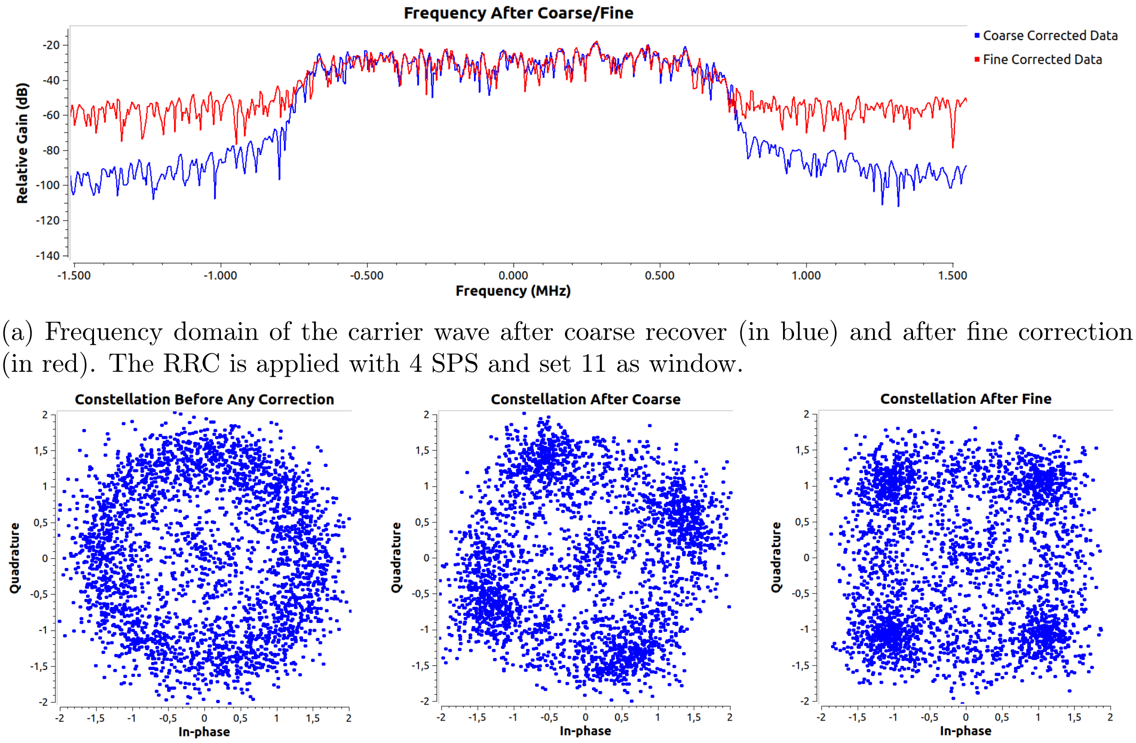
2.4.2.2 Coarse Frequency Correction

In the first stage it is possible grossly recover the frequency offset by using for example the Frequency-locked loop (FLL) technique which derives of band-edge filtering [39]. A band-edge filter covers the upper and lower bandwidths of a modulated signal taking into account the range of the roll-off factors and the interpolation already talked in Section 2.3 determining the frequency placement of the band-edges. Going to the working of the FLL itself, it filters the upper and lower band-edges into $X_u(t)$ and $X_l(t)$ respectively. Then it is calculated the error by [40]:

$$e(t) = \Re[(X_u(t) + X_l(t)) * \overline{(X_u(t) - X_l(t))}] \tag{2.3}$$

This error is directly proportional to the carrier frequency, thus, it is possible recover in a grossly form the frequency offset by using a second order loop.

2.4.2.3 Fine Frequency Correction



(a) Frequency domain of the carrier wave after coarse recover (in blue) and after fine correction (in red). The RRC is applied with 4 SPS and set 11 as window.

(b) Constellation points before any frequency correction.

(c) Constellation points after coarse frequency correction.

(d) Constellation points after fine frequency correction.

Figure 2.8: Frequency domain of the carrier wave after being applied the coarse frequency correction (at blue) and after being applied the fine frequency correction (at red) 2.8(a), and the constellation plots 2.8(c) and 2.8(d), respectively. In 2.8(b) is shown the constellation plot without any frequency correction.

After we got closer to the real frequency, although it is possible to visualize in Fig. 2.8(c) an aggregation of values to form 4 constellation points these aggregation is spinning due to a minor frequency deviation. The goal here is get a constellation where the points are right placed, steady, and as aggregate as possible, which is done by using fine frequency compensation techniques. In Fig. 2.8(a) is shown the difference between after coarse and fine frequency correction. Although in frequency domain plot, it is not possible to evidence big differences with the naked eye, when you look to the respective constellation points shown in Fig. 2.8(c) and in Fig. 2.8(d) the differences are substantially large and evident.

For solving this problem, it is possible the usage of algorithms like PLL or Square Difference. However, both requires the received signal to be processed (with some other algorithms) to get a recovered version of the carrier. The Costas Loop is a method that works directly over signal [41] which relies on a feedback concepts similar to PLL. This method has the ability to find the right phase and self-correct in order to keep the carrier correctly recovered being used a loop in order to achieve it. Because of this operation mode it has the only disadvantage of needing some prior time to settle the loop, although after it is settled, it feeds itself to keep track the correct frequency.

Before getting to the core of the Costas loop we need to understand some components and useful principles. An important component of the Costas loop is the Voltage-Controlled Oscillator (VCO) which is an electronic oscillator that produces a sinusoidal signal whose frequency depends on the input voltage on the oscillator. Also, Costas loop operation relies in two important principles: (1) the coherent principle which is basically the creation of a carrier that is identical in frequency and phase to the data signal carrier; (2) and the orthogonality principle which says that for a signal it is possible to recover both sine and cosine components (orthogonal carriers) by multiplying it upon reception with the correspondent carrier followed by low-pass filtering.

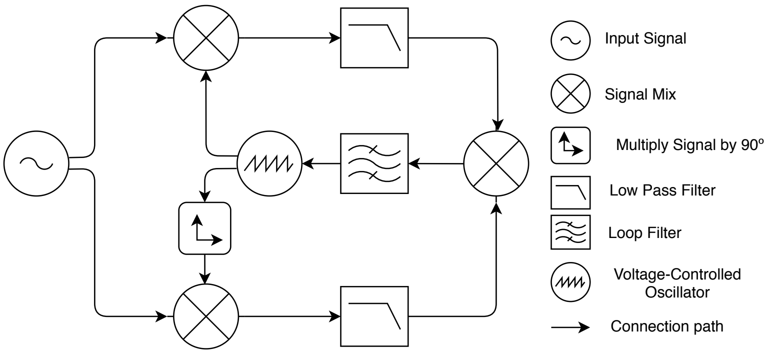


Figure 2.9: The Costas Loop circuit architecture.

That said let's go to the core of the Costas loop itself by explaining its architecture shown in Fig. 2.9 presented in [42] and [43]. Basically, it works by iterate over its internal generated carrier with the VCO into the correct phase and frequency (in this iteration is used the two previously discussed principles). After the signal is input, there are two paths possible: In the upper path, the signal is multiplied by the generated carrier (in VCO) and then at the result is applied a Low Pass Filter (LPF). In the lower path the signal is multiplied by the orthogonal carrier generated (by the VCO), and then applied a LPF. The result of the two LPFs is then multiplied to get the signal error and then, this error is filtered with a loop filter which controls the voltage to the VCO. There is two important notes to pointed out here:

- When the VCO's voltage is changed, in reality it is changing the frequency and phase of the carrier and hence making use of both principles mentioned above. When the right voltage is found the Costas loop "locks" and keeps track for frequency and phase changes in the original signal by feeding itself in VCO;
- Since this loop is "locked" then the Sine component of signal already corrected is the output of the LPF in the upper path and the cosine it is the output of the LPF of the lower path (because the orthogonality principle).

2.4.2.4 Phase ambiguity

Due to the presence of phase distortion in the communication a fine frequency recover is used as previously discussed. The Costas loop is a blind synchronizer of the transmitted signal thus, not having the true orientation arising the problem of phase ambiguity. This problem consists in the possibility of the constellation points being rotated some amount of degrees depending the type of the modulation, generically, for MPSK it is possible to have $\frac{360}{M} * n, n \in [1, ..., M-1])$ degrees of rotation comparing with the generated constellation on TX side.

One solution for solving this problem is the usage of a codeword known as a sync word, which is a set of bits known by both by TX and RX sides. When the TX sent this, the RX is capable of identify this sequence, and hence correct the phase, therefore receiving the remaining information without any ambiguity. The identification method used is called correlation where we correlate the signal received against this word. When the signal contains the sync word the correlation procedure results in a peak of energy that can be used for phase estimate, thus, being possible to set the proper phase removing ambiguity. The sequence has a low auto-correlation in all the points except in the central one, being in this manner strong against noise (it can change some bits). There is already good known sequences for this sync word, for instance, the Barker Codes [44].

The other solution is using differential encoding, where in TX side, instead transmitting the direct mapping of symbols into the constellation points, it is encoded the difference between these symbols and then mapped the resulting symbol into the constellation point. Here, a symbol depends not only of the transmitted symbol but also the previous one, being represented as $y[0] = (x[0] - x[-1])\%M$, where M is modulus of code's alphabet) [45]. At the RX side it is necessary do the reverse operation removing this way the phase ambiguity.

As everything else we have pros and cons in the use of one or another. The advantage of differential encoder is that there is no need to train the receiver, hence it is faster because we are not introducing overhead, however, has two main disadvantages:

- If one error in the transmission occurs, it can cause two symbols wrongly recovered (because the dependency of the last one).

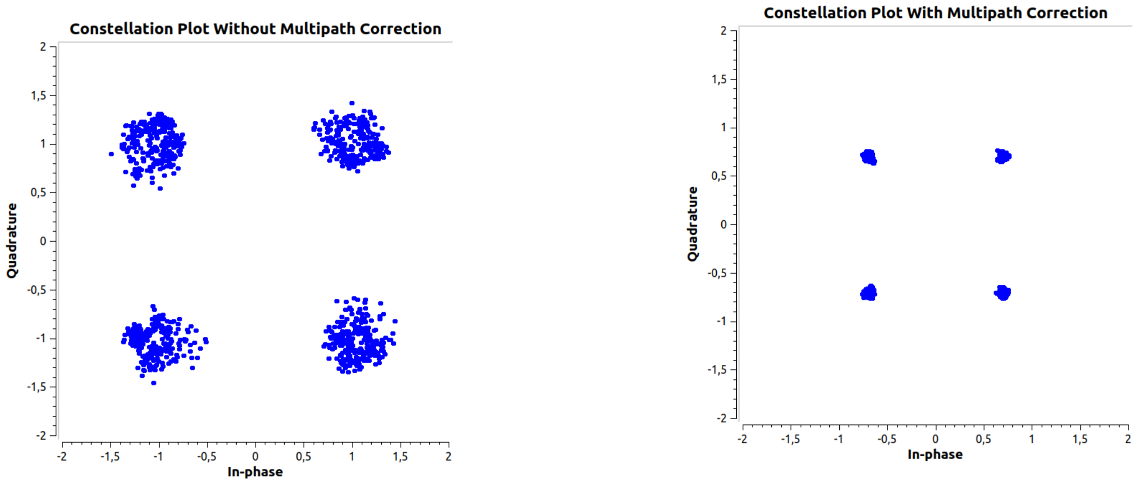
- It is not possible to use gray code mapping (if one bit is wrongly received the decoded symbol may have more than 1 bit error).

On other hand, with the sync word it is possible using the gray code map, so if 1 bit is wrongly received than the decoded symbol is the neighbor of the real constellation point, thus, being at maximum, one-bit error.

2.4.3 Multipath Problem

Multipath is in short, having different paths to the receptor. In a wireless communication the signal reflects on environment objects, for instance, walls, and the signal arrives to the receptor a little after of the direct path signal, thereafter the receptor receives the same signal various times with slight timing differences. This, obviously affects the transmission causing the dispersion of the points of the constellation as shown in Fig. 2.10(a), hence, increasing the probability of receiving a wrong symbol.

In order to tackle this problem it is need an equalizer, whose performance usually depends on the number of taps, where more taps mean better final results at the expense of an increase in computational overhead. Thus, this number must be small but enough for correcting the channel which is experimentally found. Comparing a constellation plot after being applied the equalizer it is possible to observe a constellation significantly more converged and cleaner like in Fig. 2.10(b).



(a) Multipath effect in a constellation.

(b) Multipath correction with a 15 taps equalizer.

Figure 2.10: Multipath effect in a constellation plot 2.10(a) and its correction after being applied an equalizer with 15 taps 2.10(b).

2.5 Forward Error Correction Codes

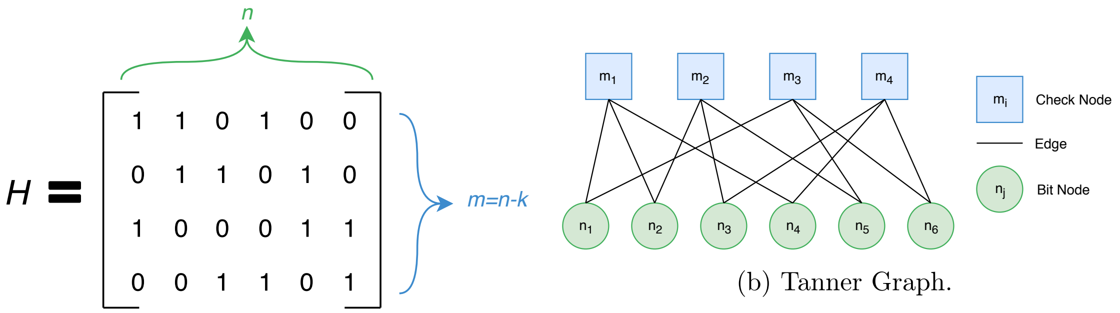
In a communication system the probability of having errors is high, so, it is necessary a mechanism which enable recovering the wrong bits. Forward Error Correction (FEC) is usually employed, where the messages to be sent are coded by introducing redundancy (added extra bits) that allows to detect and/or correct wrong bits upon reception. These check bits are usually linear combinations of several bits of the message. It is this knowledge and relation between them, that enables the message being correctly recovered in RX side, even with some corrupted bits of the codeword.

Errors that occur during transmission are typically of two types: a single-bit error where some randomly spaced bit is wrongly received; or a burst error where some amount of continuous bits are wrongly received. The error correction codes are mainly better prepared for the first type, despite the second being the worst type but common.

2.5.1 Repetition Code

One of the simplest error detection/correction code is the repetition code, in which, each bit is repeated a number of M times. Detection and correction of error upon reception is made by majority counting, i.e. if $M = 3$ and bit b_0 has been sent, and receiver gets b_0 b_0 $\overline{b_0}$, the receiver will know that an error have occurred during transmission and will decode the bit as b_0, because it has received b_0 twice and only one time $\overline{b_0}$. This simplest code enables a very simple receiver. However, it inducts a large overhead to the transmission since is transmitted M times the desired data.

Another important thing is that this code is extremely high sensitive to burst errors because there is no correlation with other bits unless the M neighbors. For all this, conjugated with comparison of other errors correction codes, in terms of Bit Error Rate (BER) results, is the less used in a communication system.



$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

(b) Tanner Graph.

(a) Parity-Check matrix.

Figure 2.11: For a given (6,2) LPDC code, its created a parity-check code matrix H represented in 2.11(a) with the respective Tanner graph illustrated in 2.11(b). All images adapted from [46].

2.5.2 Low-Density Parity-Check Code

Low-Density Parity-Check Code (LDPC) are linear codes. A Single Parity Check (SPC) code [46] is basically a code that adds a single bit to a message, with that bit dependent of the message (usually a linear combination of some bits of the message). An example is the Even Parity code where is added a parity bit that is the $sum(modulo2)$ of all the bits of the message resulting in a codeword that has always an even number of 1's. With this in mind it is possible to detect and recover a small error in the message.

A linear code is basically the combination of several SPC codes, defined by a parity check matrix H, with each row defining the SPC restriction that the codeword must satisfy, i.e. $Hc^t = 0$; e.g. considering the even parity code, the H matrix is a line of 1's, meaning that all the message bits plus the parity bit must sum 0.

The LDPC code was proposed by Robert G. Gallager in 1963 [27], as the name suggest has a parity matrix H which has a lot of 0's and a few 1's (low density of 1's). This sparseness grantees that the decoding complexity and minimum distance increases with the code length.

However, due to decoder requirements being to big at the time, LDPC codes were neglected, until 1993 when they were re-discovered [47] and large research was devoted to improve their design and decoding algorithms, therefore increasing their performance which make widely used in today communication systems due to their powerful error correction capacity.

More specifically, a (n, k) LDPC code means that for each k bits of the message will result in a n codeword length (redundancy added) using H as parity-check matrix. The matrix H has $m = (n - k)$ rows with each one corresponding to a parity check equation and has n columns with each one corresponding a bit of the codeword. Furthermore, it is a (W_c, W_r)-regular if for each column there are exactly W_c 1's and exactly W_r 1's for each row, which totals $X = n * W_c = m * W_r$ number of bits at 1 in the matrix H[46]. The matrix in Fig. 2.11(a) represents a (6,2) LDPC code and is (2,3)-regular, thus having 12 1's. A valid string y is a valid codeword for the code with matrix H if and only if $Hy^T = 0$.

The generator matrix of the code represented as G is used to the encoding process, for a u vector which holds k message bits, then the corresponding codeword can be found using the equation $c = uG$, where G is $k * n$ with k/n ratio. Thus, for a code with H as parity-check matrix, G can be found by performing Gauss-Jordan elimination resulting in the end $GH^T = 0$ as described in [46].

Finding G is not practical, thus, generally the LDPC code is restricted to be systematic, where the codeword is composed by the message unchanged appended to the parity bits, which correspond to H matrix with a well defined structure regarding the parity bits [48]. This structure allows the computation of the parity bits directly from the H matrix and the message m, precluding the need for finding G. This matrix can be randomly generated as long as the requirements are meet, or it can be generated using known algorithms like Gallager parity-check matrix and the MacKay Neal's algorithm proposed in [48].

The LDPC codes are represented mainly by a Tanner graph which consists in two vertices: n bits of the codeword called as *Bit Nodes* (circles), and m vertices to the parity-check equations called as *Check Nodes* (squares). Since the LDPC consists in parity-check equations, then a bit node j connects to a check node i if the bit j is included in the equation i, this is, if $h_{i,j}$ element of H is set as 1, so the number of edges (X) is equal to the number of 1's in the H matrix. The Tanner graph of the matrix in Fig. 2.11(a) is shown in Fig. 2.11(b), consequently has $n = 6$ bit nodes, and $m = (n - k) = 4$ check nodes with $X = 12$ edges.

The decoding process is usually iterative of type *message-passing algorithms* since the operation can be explained by exchanging information between bit nodes and check nodes by passing of messages in edges of a Tanner graph. There are several types of algorithms that differ on the type of messages exchange. Some of these messages are hard-decision (binary) like *bit-flipping decoding*, others are soft-decision (probabilistic) like *belief propagation decoding*, which represents the level of belief about the value of the codeword bits. When probabilities are represented as *log* likelihood ratios, then the algorithm is called *sum-product decoding* since this allow calculations with sum and product operations on bit and check nodes [46].

Regarding the *bit-flipping decoding* algorithm which is implemented on GR, like it was said before, the messages are represented as binary (hard-decision) and passed through the Tanner graph. A bit node sends a message declaring if the received bit is a 0 or a 1, then the check node replies a message to each connected bit node (through edges) declaring what value it should be, taking in account the check node restriction and the information it has received from the other bit nodes connected to it. Basically the check node determines the respective parity-check equation and verify if it is satisfied (by doing the modulo-2 sum of that bit, then the result must be 0). If it is not satisfied than the bit is changed (flipped), otherwise goes no next bit. This process is repeated until it recovers all the message correctly or until ends a previously defined maximum number of iterations.

2.5.3 Reed-Solomon Code

Another known error correction code is the Reed-Solomon (RS) introduced in 1960 by Irving S. Reed and Gustave Solomon [28] which is a subset of a higher-level group of binary cyclic codes called as Bose–Chaudhuri–Hocquenghem (BCH) codes. Besides being a classic error correcting code, it is still widely used in modern world in consumer technologies (e.g. CDs, DVDs, Blu-ray, QR Codes), in data transmission technologies (e.g. DSL, WiMAX) and storage systems (RAID 6) showing the great capability of correcting errors.

The RS code works with blocks with m bits length (called as symbols), where a (n, k) RS code produce n symbols for each k symbols. First, the bit data stream is split into k blocks with m bits length and it is chosen t number of symbols that can be corrected taking into account that $k = n - 2t$ and $n = 2^m - 1$. A symbol error occurs when at least 1 bit on that symbol is wrong. This means that a code that is capable of correcting $t = 16$ symbols with 8 bits for each symbol. In the worst case, if 16 bits errors with one on each symbol, then the decoder can only correct 16 bits in total. On the best case, if it occurs error on 16 complete symbols (8 bits), then the code can correct $16 * 8 = 128$ bits, thus, being a code highly resistant to burst errors.

On the encoder process itself, first it is necessary create the Galois Field (GF) to further compute the RS generator polynomial which is used on a feedback shift register in order to compute the parity check bits.

To understand the encoder process it is necessary understand the concept of GF. The GF(p^n) has p^n elements where p is a prime number (primitive element) and n a positive integer, and the elements of field obey to a set of restrictions and properties such as [49]:

- It is possible to apply at the elements operations like summation, subtractions, multiplications and divisions without getting out of the field;
- For each element a on the field, $a \times 0 = 0 \times a = 0$;
- For any element not null a and b on the field, $a \times b \neq 0$;
- $a \times b = 0$ and $a \neq 0$ imply that $b = 0$;
- For any two elements a and b on the field, $-(a \times b) = (-a) \times b = a \times (-b)$;
- For $a \neq 0$, $a \times b = a \times c$ imply that $b = c$.

A particular case of GF is when $p = 2$, thus, an addition is the same of a XOR operation, and a multiplication is the same of an AND operation, therefore being practical when managing data for encoding and decoding information in error correcting codes. The elements of the GF may be represented as polynomials with a degree lesser than n, hence, the entire GF can be constructed when given an irreducible polynomial.

After having the GF constructed it is necessary to compute the generator polynomial for the RS characteristics in cause. The RS code is a special case of nonbinary BCH code due to the length of the code n is 1 less than the field size of the GF. This means that, the sequences generated will have length $p^n - 1$, whose roots include $2t$ consecutive powers of p. Consequently, there are only $2t$ parity checks which leads to a t-error correcting code (corrects at maximum t errors - Singleton bound). In short, after choosing the roots, α^i to α^{i+2t-1}, the RS generator polynomial of a t-error correcting code is given by [50]:

$$g(X) = (X + \alpha^i) * (X + \alpha^{i+2}) * ... * (X + \alpha^{i+2t-2}) * (X + \alpha^{i+2t-1}) \tag{2.4}$$

Finally, it is used the values of the computed generated polynomial along with a feedback shift register in order to produce the redundant symbols to be appended at the original input data. The feedback shift register architecture is represented on Fig. 2.12.

For better understanding of the complete encoder process, let's suppose that we want to $(7, 3)$ RS encode the bit stream 111001111 with the possibility to correct 2 wrong symbols.

First, the data stream is mapped into symbols with $m = 3$ bits each, then its calculated the GF(2^3) with an irreducible polynomial $X^3 + X + 1$ represented in Fig. 2.13(a). Next, it is required to compute the generator polynomial taking into account $t = 2$, thus, the roots will be from α^0 to $\alpha^{2t-1} = \alpha^3$, which after applying the formula above mentioned results in the generator polynomial $g(X) = (X + \alpha^0)(X + \alpha^1)(X + \alpha^2)(X + \alpha^3) = X^4 +$

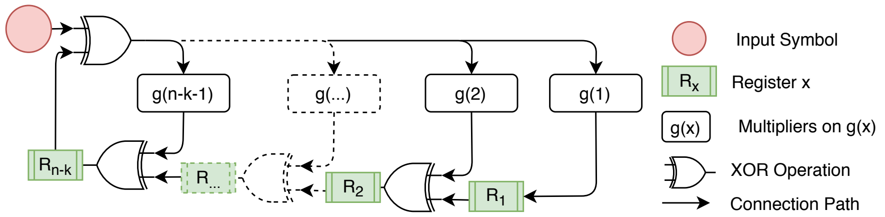
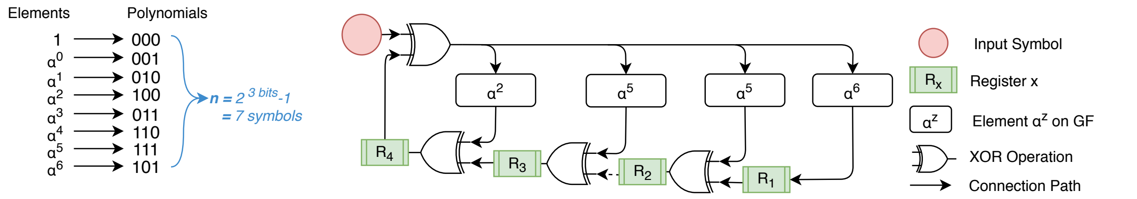


Figure 2.12: Feedback Shift Register architecture a Reed-Solomon Code. Adapted from [50]

$\alpha^2X^3 + \alpha^5X^2 + \alpha^5X + \alpha^6$. Each connection of the feedback shift register has a multiplier of the respective value of the generator polynomial previously computed (namely α^2, α^5, α^5 and α^6) as represented on Fig. 2.13(b). After all stream (previously mapped following the Fig. 2.13(a)) is input it will result α^6, α^5, α^0 and α^1 values on the registers. These values are mapped into the polynomial (mapped following the Fig. 2.13(a)) resulting in the bit stream 101111001010 to be appended into the original data stream which results in total $n = 7$ symbols for each $k = 3$ symbols input.



(a) Example of a GF(8) with irreducible polynomial $X^3 + X + 1$.

(b) Example of a RS feedback shift register using the $X^4 + \alpha^2X^3 + \alpha^5X^2 + \alpha^5X + \alpha^6$ generator polynomial.

Figure 2.13: In 2.13(a) is represented a RS feedback shift register example with symbols obtained using GF(8) (represented in 2.13(b)) and the $X^4 + \alpha^2X^3 + \alpha^5X^2 + \alpha^5X + \alpha^6$ generator polynomial. Adapted from [50]

The RS decoding process is a lot more complicated than the encoder as it is necessary determine both location and values of the symbol errors. Gorenstein and Zierler discovered the first algorithm for decoding that was further improved by Chien and Forney. However, the first efficient algorithm was proposed in 1968 and named Berlekamp's iterative algorithm, that is the one that will be addressed here.

This process consists in three major steps [51]: (a) Syndrome calculation; (b) Symbol error-location determination, and; (c) Symbol error correction.

The syndrome calculation is only dependent on errors $e(X)$ of the received sequence $R(X)$ (not of the transmitted codeword $v(X)$). These syndromes $S = (S_1, S_2, ...S_2t)$ can be calculated by substituting on $R(X)$ by the element of the respective GF(2^m), thus $S_i = r(\alpha^i)$.

Finding the symbol error-location polynomial $\sigma(X)$ involves solving several equations from syndromes previously calculated S. It is important to note that, if the degree of $\sigma(X)$ is $> t$ than it is not possible to correct all the errors on $R(X)$. If degree of $\sigma(X)$ is $\leq t$ then α^i, with $i = 1, ..., n$, is replaced on $\sigma(X)$ in order to get the roots

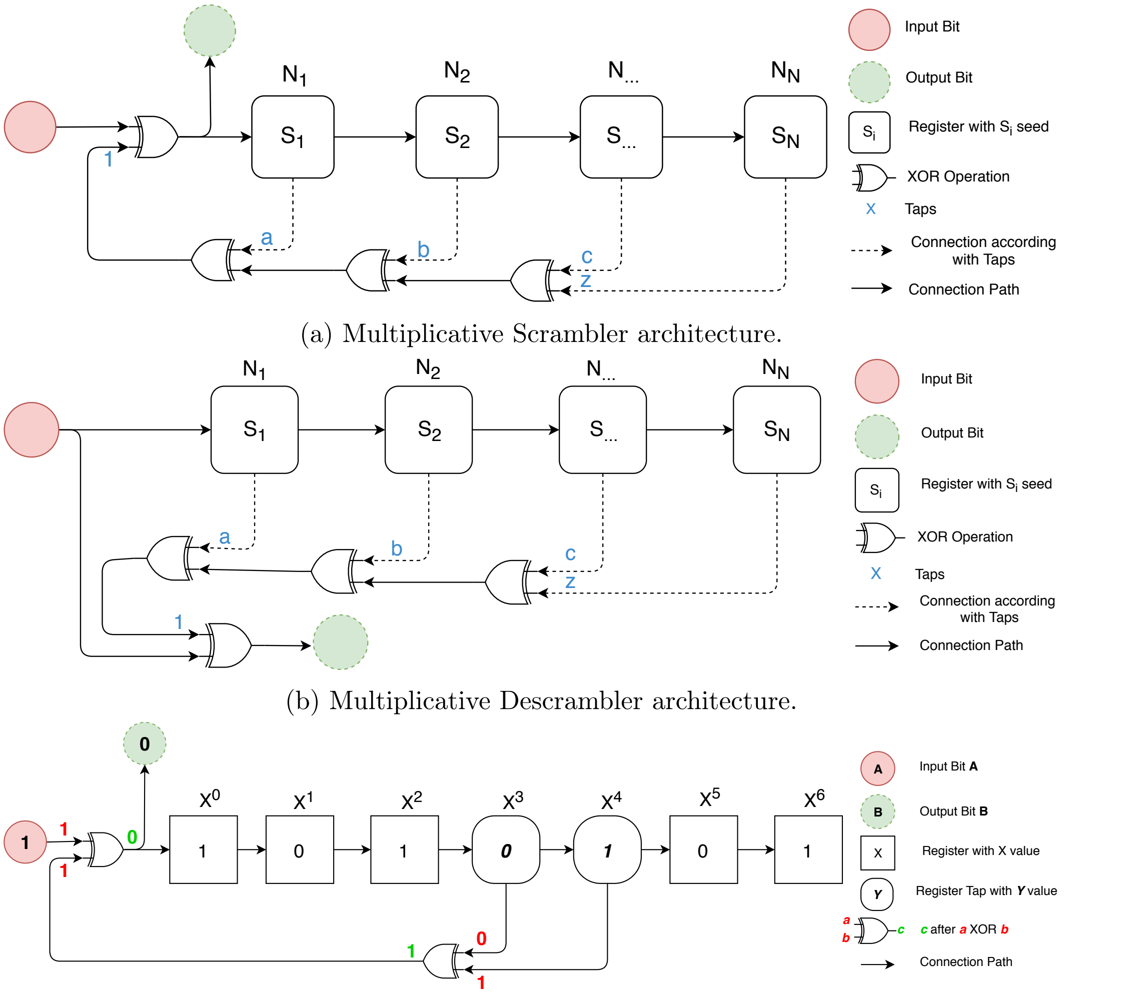
Finally, the goal is to find the error-location numbers $\beta_1, \beta_2, ..., \beta_v$. The roots of the previously calculated $\sigma(X)$ are $\beta_1^{-1}, \beta_2^{-1}, ..., \beta_v^{-1}$, thus, the coefficients of $\sigma(X)$ and the β_X are related. After found the error-location numbers it is solved for the error value at the error location and correct the error of $R(X)$.

In this section we presented three of many existing error correction codes, which are the most used in communication systems, and we use in this thesis.

2.6 Scrambling

As discussed, a received signal goes through many algorithms to successful recover the transmitted signal, for the purpose of both TX and RX sides being synchronized. This is possible because it is receiving different bits, however, if it is sent many zeros in a row, then, the receptor loses the symbol synchronization. The scramble is often used to improve security in a transmission, however, at this view only aims to eliminate long sequence of same bits, called as whitening sequences, consequently reducing problems with synchronization at the receiver [52]. There are two types of scrambler, the multiplicative scrambler and the addictive scrambler as it will be discussed next.

2.6.1 Multiplicative Scrambler



(a) Multiplicative Scrambler architecture.

(b) Multiplicative Descrambler architecture.

(c) Scrambler example for an input 1, with 1010101 as seed value and the x^4+x^3+1 polynom.

Figure 2.14: Multiplicative scrambler architecture represented in 2.14(a) and its respective descrambler in 2.14(b). A specific example is shown in 2.14(c).

A multiplicative scrambler [53] is recommended in V.34 by International Telecommunication Union-Telecommunication Standardization Sector (ITU-T) [54]. It works by using a Linear-Feedback Shift Register (LFSR), i.e. a shift register whose is a linear function of its previous state and new received bit. It is necessary to define three initial parameters: the number of registers N; the seed which is the initial value placed in theses inputs $S_1, ..., S_N$; and a polynomial of type $x^0 + x^a + x^b + ... + x^z$ where the exponent number $0, a, b, ..., z$ are the index taps. These taps are the registers that influences the output bit used for each iteration while the others registers are not considered (it does not influence the output bit). Note that x^0 is always 1 (always does one XOR ($\oplus$) operation between the input bit and the tap's resulted bit).

For better understanding, consider the example of the architecture presented in Fig. 2.14(a). An iteration is when a data bit is input and XORed with the bit that results from XOR operation of all of the tap's content (equivalent to $sum(modulo2)$ of all bits) on the current state resulting in the output bit already scrambled. After this, all the registers are updated shifting to the right register and the output bit is set in the first register discarding the last one.

The descrambler operation just reverts the operations mentioned above. For each received bit at input, the output bit is the XOR of that input bit with all taps XORed between them and with the input bit being fed to the shift register for the next operation (where the registers are right shifted). Note that it is required to start with the same seed value as in the scrambler or it will be lost some bits. The descrambler architecture is shown in Fig. 2.14(b).

The Fig. 2.14(c) presents a practical example where $N = 7$, as registers containing as initial seed 1010101 and the scrambler polynom is $(0X19)_{16} = (00011001)_2 = x^4 + x^3 + 1$, (hence only the content of 4^{th} and the 5^{th} tap registers influences the output). Let's say that these registers have as contents bit X and bit Y respectively. For a given input bit B, the result that will add to the first register is: $B \oplus X \oplus Y$, and finally all bits are shifted to the next register and B is set in the first register (so X and Y is now in the 5^{th} and 6^{th} register respectively). In particular, if the bit 1 is input, then $1 \oplus 0$ (4^{th} register) $\oplus$ 1 (5^{th} register) results B as 0. The content of the registers will be 0101010 and the output bit scrambled that will be transmitted is 0.

This type of scrambler has the property of being self-synchronizing, this means that even with different seed after a some amount of input bits the original content will be descrambled correctly, this is good for a normal situation but bad in security perspective, because a eavesdropper, even without the correct seed will be able to get the information after some fixed number of bits (number of taps). Another property is, when descrambling, if one error bit is input, than will cause the output of K wrongly descrambled bits, being K the number of taps.

2.6.2 Additive Scrambler

The addictive scrambler is a standard for Digital Video Broadcasting (DVB) [55], and is much alike as the previous except that it uses a sync-word as seed [53]. A sync-word is a bit pattern that is inserted in the input stream at equal intervals. The descrambler searches for that pattern and whenever it finds it, it resets the registers states by reloading the seed; in this manner the processing is frame based.

The architecture of the additive scrambler is presented in Fig. 2.15. As before, when a data bit is input, is XORed with all content taps producing the output scrambled bit. The registers content bits in the registers are shifted for the right register discarding the last

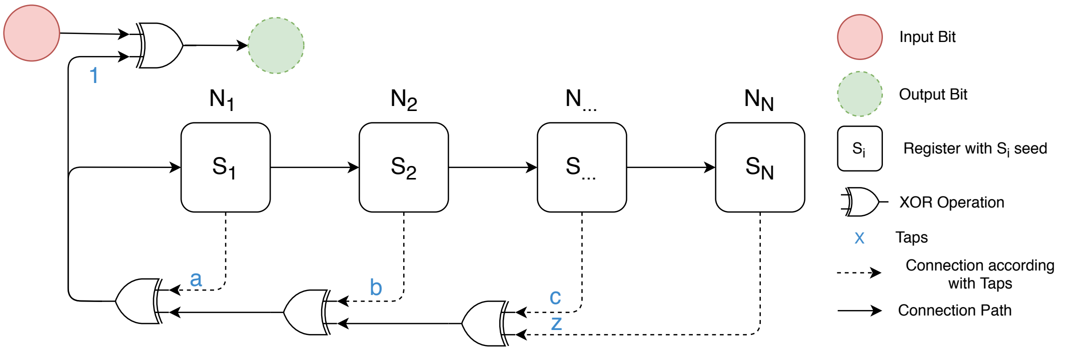


Figure 2.15: Addictive Scrambler/Descrambler architecture.

one and the first register is set the XOR operation between all taps. The difference here is for setting the first register now, it is not dependent of the input bit, but dependent only of the taps. This result in the property of the additive scrambler not being self-synchronize, and thus, if some error occurs when descrambling then all the rest of the frame is lost. The descrambler has in this case exactly the same architecture as the scrambler.

2.7 Overview

In this chapter firstly we have analyzed and chosen the software that will be used to carry this work and define its setup specifications.

It was also given an overview of how a basic communication is processed and a more detailed explanation of each component of that communication system. This components can be mapped into GR blocks such as codification using the **FEC Extended encoder** block, the modulation using the **Chunks to Symbols** block and the pulse shaping by applying the **Polyphase Arbitrary Resampler** block.

Then we addressed problems upon reception, such as the imperfection on the received signal due to poor channel's conditions that are needed to be taken care of by both performing clock and carrier synchronization in order to retrieve a clean signal ready to be demodulated using the **Polyphase Clock Sync**, the **Costas Loop** and the **Constellation Decoder** blocks, respectively.

The step that follows is decode the sent information to correct bit errors introduced in the channel (using **FEC Extended Decoder** block), and finally, descramble the information to recover the original information.

All these components are crucial to understand the proposed physical layer security scheme that will be presented in the next chapters. Note that the GR blocks here mentioned will be detailed in the implementation chapter.

This page is intentionally left blank.

Chapter 3

Physical Layer Security

Wireless communications are broadcast in nature. It is therefore difficult to restrict the communication to an intended receiver and more actors may take part on a communication such as an eavesdropper, a relay or a jammer. In this work, we will resort to the communication actors first introduced in [56] in which Alice will be used to represent a Transmitter (TX) who wants to communicate to the intended Receiver (RX) named as Bob, and there is also Eve which do passive eavesdropping trying to extract information without being detected and without changing the signal.

3.1 Modern Cryptography

Shannon [2] started the theoretical information about cryptography creating a model where he assumed that for a plain text P uses a non-reusable key K to generate a cipher text C, then in a noiseless transmission, Eve has the knowledge of the scheme used for the encryption, with no limit in computer power and with a copy of C. This model introduced a "perfect secrecy" notion whereby Eve does not increase the probability of a successful decryption, meaning that the mutual information between the message and the cipher text is 0, i.e. $I(M;C) = 0$.

The perfect secrecy is only guaranteed if the message and key have the same length and if the entropy of the key is greater or equal than the entropy of the message, i.e. $H(K) \geq H(M)$. One example of when this happens is on the *one-time pad* cipher [1]. However this cipher has some problems, for instance: (1) the same length requirement between key and message is unpractical for big length messages; (2) the key must be used once and only once to guarantee the algorithm security; (3) how the key is shared to Bob without Eve catches it; and finally (4) the algorithm is dependent on how good is the random key generated, i.e., the security of the algorithm is pseudo-random generation algorithm's performance dependent.

Modern cryptography algorithms are mainly two types, the symmetric and asymmetric cryptography. The symmetric cryptography [1] is used in algorithms like Advanced Encryption Standard (AES), Data Encryption Standard (DES), 3DES, among others. The same key is used to both encrypt and decrypt information leading to some problems, for instance, how the key is shared between the parties without Eve intercepting it. One way to solve this problem is personally deliver it but this is not practical, so with years emerged some solutions, but always with some problems, resulting in another type of cryptography, namely the asymmetric cryptography.

The asymmetric cryptography [1] used mainly by the Rivest-Shamir-Adleman (RSA) algorithm [56] exploits the properties of prime numbers in a way to create two keys for each partie, a private and a public one, using one of them to encrypt a message and another to decrypt it. Alice encrypts a message with Bob's public key, hence only Bob can decrypt it with his private key. The drawback on this scheme is that Eve can impersonate Alice. A workaround on this is Alice encrypt with Bob's public key and, on top of that, encrypt with her own private key. This means that Bob knows for sure that the message come from Alice since it can decrypt it using her public key. This was a great advance because solves the problem of Eve intercepting the key, however it keeps a problem how to share the public key to everybody in a manner that the others knows for sure that who publishes is who it says it is, leading to a system called Public Key Infrastructure. This system relies in a Trusted Third Party to make sure the link between a person and a specific public key using a scheme of certificates and digital signatures hierarchy being thus, a complex system.

Since the asymmetric cryptography uses prime numbers for encryption of data it is not a fast encryption like the symmetric one, so what usually is done is using symmetric cryptography to encrypt the message and then encrypt the symmetric key using asymmetric cryptography. There is an underlying assumption about the cryptography, all these operations/algorithms rely on mathematical functions that are computationally infeasible, but not impossible, to invert. With the growth of computer market the processing power increased to, and it is always increasing leading to, what it infeasible now, in a given period of time may be possible to break.

Concluding, all the security we are relying in for a communication is not impossible to break, but is infeasible to do it only at the present. Besides, new vulnerabilities are always being found on these schemes compromising the security. For instance, the "Heartbleed" bug [57] was a serious vulnerability in OpenSSL cryptographic software library compromising all communications using Secure Socket Layer (SSL)/Transport Layer Security (TLS) in the internet.

The Physical layer security exploits the characteristics of the medium, typically wireless (such noise, fading, interference), in a way to limit Eve from receiving information (at bit level) while maximizing the received information by Bob. Therefore, has the main advantage of not being computer power dependent as modern cryptography, thus, the security added does not need to be updated constantly. Finally, this level of security is placed as a complement and not a replacement of the modern cryptography.

3.2 Wiretap Channel and Security Metrics

In a communication with three parties where one of them is an eavesdropper there are two important channels to consider, the Alice to Bob channel and the Alice to Eve channel. Shannon [2] described in 1949 a special case of the wiretap channel (not known with this notation at the time) where there is no noise in the channels and relies only in a shared secret key between Alice and Bob for the achievement of a secret communication in a way that no information is leaked to Eve as previously discussed in Section 3.1. In this scheme, the source message M is coded by Alice represented as X^n, with n the resultant length. Here, when the mutual information between the message and the coded message is zero, $I(M; X^n) = 0$, the "perfect secrecy" notion is achieved. This is only possible when the codeword's entropy is not lower than message's entropy, $H(X^n) \geq H(M)$. In particular, $H(K) \geq H(M)$, where K is the shared key, thus, K must have the same length as M;

therefore, impractical.

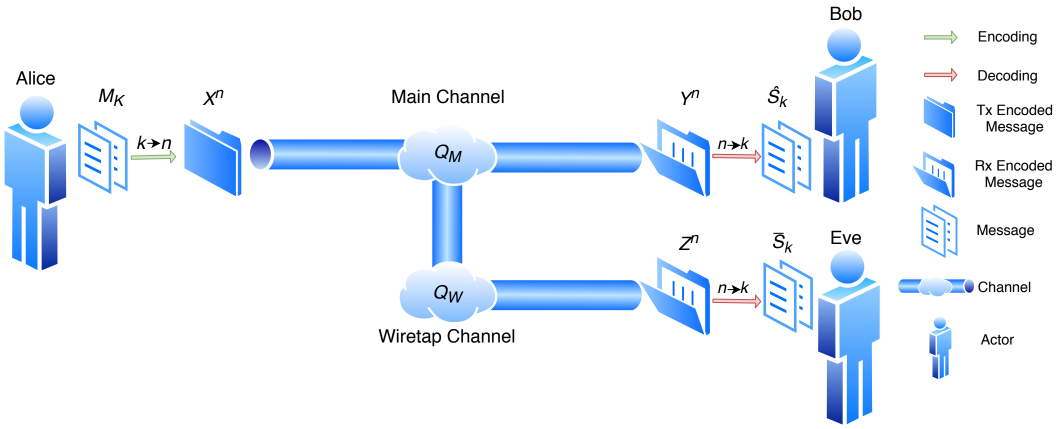


Figure 3.1: Wiretap channel general case model.

Wyner introduced in 1975 the wiretap channel [3] where he considered the imperfections introduced to the channel. In his model, M is an aggregation of binary data sequences, $S_1, S_2, S_3, ...$ with same probability of being 0 or 1 ($P_r\{S_i = 0\} = P_r\{S_i = 1\} = \frac{1}{2}$). Alice encodes the first K data sequences, $S_1, .., S_K$, resulting into a n binary vector, represented as X^n, then is transmitted to Bob through the "Main channel", Q_M, which is a noiseless discrete memoryless model, and then goes through a "Wiretap channel", Q_W, which is a degraded version relative to Q_M, until reaches Eve. Bob receives Y^n which after decoding it retrieves the data $\hat{S}_k$ with an error probability of $P_e = \frac{1}{K}\sum_{k=1}^{K} Pr\{S_k \neq \hat{S}_k\}$ and Eve receives Z^n since is added to the transmission the channel imperfections retrieving $\overline{S}_k$. This model is shown in Fig. 3.1.

Taking into account this model, Wyner postulated a relaxed version of the "perfect secrecy" called as "weak secrecy" defined as $\lim_{n\to\infty} \frac{1}{n} I(M; Z^n) = 0$. Instead of guaranteeing that there is no information leaked to Eve, this notion considers the leakage of only a small amount of information to Eve Z^n, where there is no secrecy compromise by driving $1/n$ factor to zero. With this new notion, Wyner sought to maximize the transmission rate $R = K/n$ bits, while keeping some controlled information leaked to Eve guaranteeing both reliability and information-theoretic security against Eve. This maximum rate R is called "secrecy capacity" of the wiretap channel and with this he was be able to show that a communication can simultaneously guarantee both reliability to Bob and secrecy against Eve by using some codes called "Wiretap codes" and taking advantage of the channel characteristics.

In many cases this notion of secrecy was insufficient, thus, Maurer [4] strengthened Wyner notion by introducing the "strong secrecy" defined as $\lim_{n\to\infty} I(M; Z^n) = 0$. However, even this notion may not be sufficient because it requires that the message be random and uniform distributed over the alphabet. In realistic messages this rarely happens leading to an even stronger notion of secrecy where the message distribution is considered, the "semantic security" [5] defined as $\lim_{n\to\infty} max_{pM}\{I(M; Z^n)\} = 0$, where pM are all possible message distributions.

In closer real-world channels, for instance the Gaussian wiretap channel or the fading channel, it raised some other more practical metrics like Bit Error Rate (BER) [6]. The BER metric takes into account only the bits on output of the decoder and compares them with the original ones of the transmitted message, for example in 10 bits if it was successfully recovered 7 bits, then $BER = \frac{WrongBits}{TotalBits} = \frac{3}{10} = 0.3$. The Signal to Noise

Ratio (SNR) is the ratio between the signal and the noise that is added on the channel.

With these notions, also in the same work [6] the authors set a BER target on Bob and another on Eve with the goal to find the SNR required to each channel to achieve the same goals (both reliability and secrecy), raising another important security metric, the "security gap". The security gap (S_G) [6] is the difference between the "Main Channel" and "Wiretap Channel" quality required for a desired level of physical security. To increase the security gap it was proposed [6] a coding scheme based on puncturing, this is, purposely erase some bits on Alice in a way that the error correction code can recover them in Bob but is not capable to do it on Eve. Therefore, security gap is the difference between maximum Eve's SNR where until that value she can not correctly decodes the information (granting secrecy) and minimum Bob's SNR in which, from there he is able to correctly decode the information (granting reliability), represented as $S_G = SNR_{min}^{Bob} - SNR_{max}^{Eve}$. To sum up, if the security gap is small but enough to Bob having advantage over Eve, then it is possible to secure a communication [58] using "Wiretap codes" and taking advantages of channel quality (in this case the lack of it).

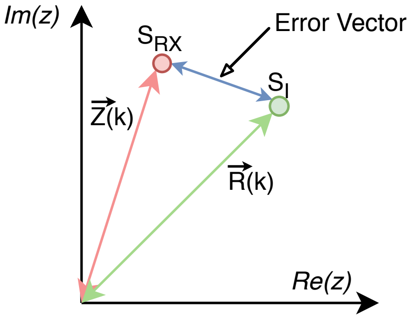


Figure 3.2: Representation of the received and ideal symbols with their respective vectors.

Another known practical metric is Error Vector Magnitude (EVM) which is an indicator of modulated signal quality by measuring the error in the distance between the received symbols, S_{RX}, and the ideal ones on the constellation plot, S_I, thus, containing intrinsically more information about amplitude and phase distortion compared to BER [7]. In Fig. 3.2 is represented the received symbol with its measured trajectory $\vec{Z}(k)$, and the ideal symbol with the reference vector $\vec{R}(k)$ along with the error vector as the difference between both vectors. With M symbols the EVM can be calculated as:

$$EVM(dB) = 10\log_{10}\left(\frac{\sum_{k=1}^{M}\left|\vec{Z}(k) - \vec{R}(k)\right|^2}{\sum_{k=1}^{M}\left|\vec{R}(k)\right|^2}\right) \tag{3.1}$$

Logically, a smaller EVM value means that there is a better approximation to the constellation points, thus, better possibilities to correctly decode and retrieve the information.

In packet-based transmissions, such as in GNU Radio, we can also use as metric the Packet Loss Ratio (PLR). A packet loss occurs when some packet fails to successfully reach its destination in a network communication, either caused by errors in the transmission channel or by network congestion. The latter is not taken into account in our work once it is a direct communication (without intermediate nodes) and we have full control of the transmitting sample rate to prevent it (detailed in Section 4.1.2.1). The PLR is a practical metric that measures the percentage of lost packets (i.e. difference between the number of received packets N_{Rp} and packets sent N_{Sp}), relatively the total packets sent, thus, it is given by:

$$PLR = \frac{N_{Sp} - N_{Rp}}{N_{Sp}} \tag{3.2}$$

Finally, the Block Error Rate (BlER) takes into account not only the number of lost packets, but also the wrong ones N_{Wp} (it is considered a wrong packet if it has at least 1 bit wrong), and it is given as:

$$BlER = \frac{(N_{Sp} - N_{Rp}) + N_{Wp}}{N_{Sp}} \tag{3.3}$$

In a similar way to the security gap definition, the EVM, PLR and BER can be used in practical scenarios to evaluate security by stipulating reliability (for Bob) and secrecy (for Eve) levels based on each of these metrics, and computing the difference (i.e. "gap") between the SNRs for which Bob and Eve achieves them.

3.3 Coding For Secrecy

Although it is theoretically possible to secure a communication with a wiretap channel using "Wiretap codes", in real-world it is hard to meet all needed conditions for that to correctly work, by either having a noiseless "Main Channel" or a "Wiretap Channel" with less quality than the "Main Channel"; thus, other schemes have been proposed. In this work it will be studied some of these schemes and it will be chosen one of them as basis; however, the main focus is the jamming generation and its cancellation.

Using Low-Density Parity-Check Code (LDPC) characteristics mentioned in Section 2.5.2 and puncturing [6], a coding scheme was created where the message is coded using LDPC and the transmitted codeword is punctured in a manner that taking into account the security gap, Bob is able to retrieve the original message but Eve cannot, because Eve has a lower SNR and consequently bigger BER. The puncturing was extended to a scheme in a way to create random puncturing with a pattern that is kept secret from Eve [59], thus at the same time not needing to re-design the error correction codes and achieving more secrecy in a communication even when the channel is degraded.

In order to not expose directly the transmitted information and at the same time reduce the security gap in a way to increase secrecy performance, some solutions emerged like using the scrambling technique as mentioned in Section 2.6 [8] or with a similar effect, by using interleaving [9].

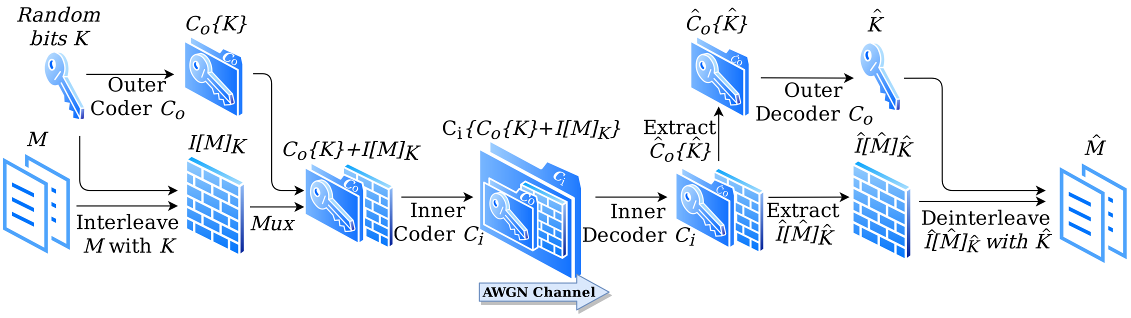


Figure 3.3: ICS security scheme architecture (figure adapted from [9]).

The scheme using interleaving named Interleaved Coding Secrecy (ICS) was first proposed on [9], and it is represented on Fig. 3.3. Alice wish to transmit a message M, so first she randomly generates a key K that is used for interleave M resulting in $I[M]_K$. An outer code C_o is applied to the key to protect it from transmission errors, resulting $C_o\{K\}$. Then the encoded key is mixed with the interleaved message, and to the formed block is applied an inner code C_i, thus enabling protecting from transmission errors both on the interleaved message, and an additional layer on the key. The final encoded information represented as $C_i\{C_o\{K\} + I[M]_K\}$ is then transmitted through a channel and at the receiver side an estimation of the transmitted message $\widehat{M}$, is obtained by reversing the operations carried in the transmitter. Thereafter it was developed a variant of these scheme by using only one encoder called as Interleaving Coding for Secrecy with a Hidden Key (ICS-HK) [60] that takes advantage of puncturing the key as it will be discussed next.

As an alternative to interleaving, it was proposed the use of scrambling [61], with this technique being called as Scrambled Coding for Secrecy (SCS). In the same way as before it was developed a variant called as Scrambled Coding for Secrecy with a Hidden Key (SCS-HK).

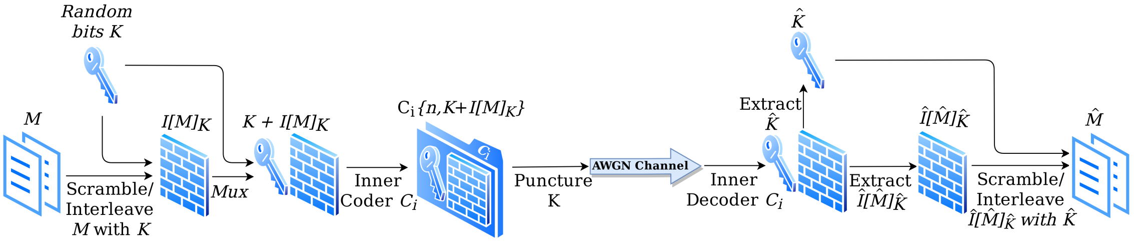


Figure 3.4: ICS-HK and SCS-HK Security schemes architecture (figure adapted from [61]).

Both schemes with the hidden key, namely SCS-HK and ICS-HK, are represented on Fig. 3.4. They rely on the generation of a key K that is used to scramble/interleave the desired message M creating $I[M]_K$. Thereafter the key is concatenated with the shuffled message, $K + I[M]_K$, and then the information is encoded with C_i which is a systematic encoder of size $(n, K + M)$, thus, K is punctured being only transmitted the last $n - K$ bits through the channel. Note that the key is hidden both from Bob and Eve but it is implicitly transmitted on the bits added by the encoder (the parity bits). The secrecy is obtained when Eve has a worse channel than Bob creating a small advantage on Bob which can retrieve the correct key due to the parity bits and hence recover the transmitted message while Eve cannot do it. Due to the Eve's worst channel the received information has more errors and the key is incorrectly decoded not enabling her of correctly descramble/deinterleave the information. Both these schemes were implemented and evaluated [61], [62] where was shown that the scrambler outperforms the interleaver.

3.3.1 Adapted SCS-HK Security Scheme

Considering the coding for secrecy schemes mentioned, the SCS-HK is the one that performs better as refereed in [62]. This scheme was implemented in MatLab and we have identified some problems that resulted in some adaptations to the original scheme, such as the type of scrambler used and how a packet is detected. As previously discussed in Section 2.6, the multiplicative scrambler has the self-synchronizer capability granting the information retrieval even without the correct seed/key which would be good for a normal communication but bad when security is a concern. Due to this characteristic, we have choose to use the additive scrambler rather than the multiplicative one (since this does not

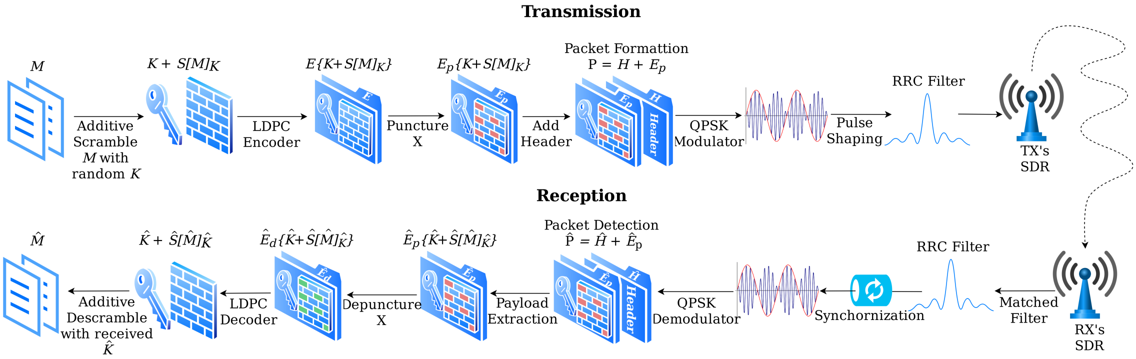


Figure 3.5: Adapted SCS-HK architecture.

have that ability). Also, in [62] it was used correlation to perform packet detection which is computational heavy, thus, in order to maximize the throughput we have changed the packet detection mechanism by using a known word (both by transmitter and receiver) and perform a bit level comparison as it will be discussed next in detail.

The SCS-HK adapted scheme architecture is represented in Fig. 3.5. Firstly, the desired binary data for transmission M is scrambled using the additive scrambling discussed in Section 2.6.2 (because it is not self-synchronizing, thus is required to have the correct key to descramble or all the frame is lost) with a random key K resulting in $S[M]_K$. This scrambled data is then appended to K resulting in $K + S[M]_K$ and then encoded using the LDPC encoder $E\{K + S[M]_K\}$ which results in the packet's payload E_p punctured with a pattern X. Now it is necessary to use an header H containing some known sequence to be recognized on RX side in a manner that it knows that is a new payload incoming resulting in a packet built by $P = H + E_p$. For the last, it is required to do modulate the data in a Quadrature Phase-Shift Keying (QPSK) and then do pulse shaping by applying the Root Raised Cosine (RRC) filter as presented in Section 2.3, and finally transmit over the Software Defined Radio (SDR).

In RX side is performed the reverse operation, where after the signal is received in the SDR first it is required do the matched pulse shaping by applying the RRC filter, then do synchronization, which includes timing offset recovery as in Section 2.4.1, carrier recover as in Section 2.4.2 and multipath correction as in Section 2.4.3. After the synchronization the binary packet $\widehat{P}$ is continuous, so the packet detection looks for the known sequence added in the header $\widehat{H}$ (considering that may occur a limit value of different bits - threshold) and then extracts the payload which is encoded and punctured $\widehat{E_p}$. The payload is then depunctured with the same pattern X where the "holes" are replaced with bit 1, and then the payload E_d is decoded. The recovered key $\widehat{K}$ will be used to descramble to get the message $\widehat{M}$ which can be exactly the same or with some differences depending on the channel interference.

3.4 Jamming and Cooperative Jamming

In a typical wireless broadcast communications supporting multiple users, when one user is transmitting the others remain silent. However, when security is considered the transmission becomes challenging making it difficult for two intended communication parties to keep their messages secure [63]. The two main reasons for this are:

- The broadcast nature of wireless makes it difficult to protect a communication because it is not possible to make it directed only to the intended receiver, thus, any in adversarial user range can receive the signal and use it somehow to retrieve private information without being detected (the direction of the communication is done on upper layers, e.g. Data Link Layer on the Open System Interconnection (OSI) model);

- The superposition characteristic of the wireless channel can lead to multiple overlapped signals making some unintended malicious signal on top of the wanted signal. Therefore, it is possible to have a malicious transmitter overlapping the information signal preventing a correct communication, this transmitter is called as jammer.

Jamming consists in the act of block/interfere the signal, and it is often used in a malicious way by blocking the communication between Bob and Alice. Previously, in Section 3.3.1, was discussed schemes that take advantages of the wiretap channel in which there is a better channel for Bob than Eve. Since this is not controllable and it is not always possible to ensure this advantage, then friendly jamming (for the intended communication parties) arises as a way [10] to help lowering the SNR of the wiretap channel.

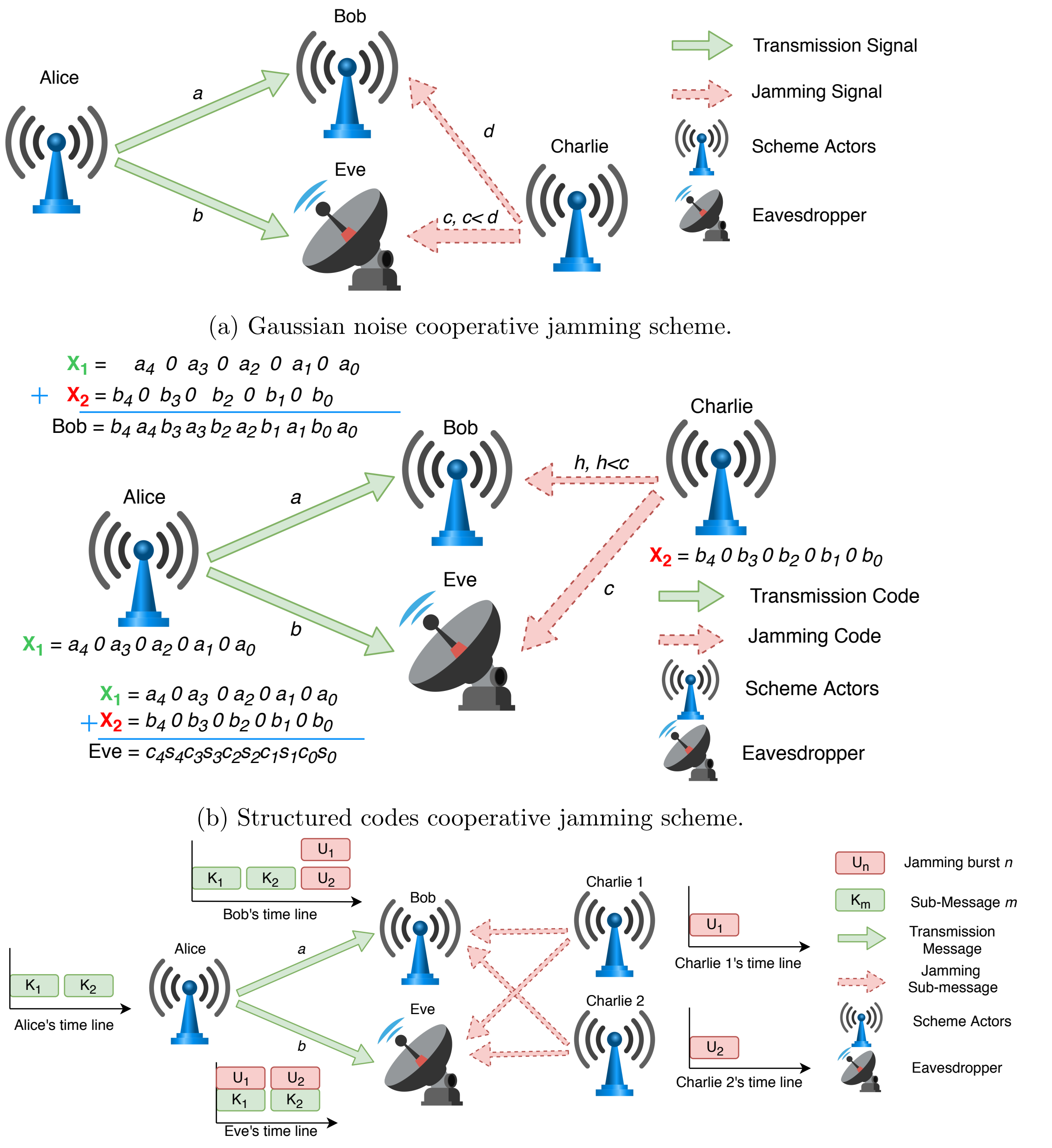
After its proposed the idea was developed by the same authors in artificial noise [64] and thereafter called as cooperative jamming [65]. Furthermore, it was also evaluated several possible scenarios of the jamming usage [66], namely, using one or more jammers, jamming regardless of the channel conditions, jamming when main channel is worse than the wiretap channel, and even use jamming when the quality channel is lower than a threshold.

Cooperative network techniques can be designed for enabling physical layer security when trusted relays are used, as for example in the generation of friendly jamming. Cooperative jamming consists in having an external jammer to help Alice to communicate with Bob, let's call him as Charlie. Depending on how the jamming is implemented, the cooperative jamming schemes can be divided in three main groups: by using Gaussian noise, by using structured codes and by using interference alignment.

Cooperative Jamming with Gaussian Noise proposed in [10, 64, 65] is the most intuitive where the jammer uses a Gaussian signal in order to degrade Eve's channel. For better understanding, take in the consideration the scheme of Fig. 3.6(a) where Alice wants to communicate with Bob, and Charlie is closer to Eve than to Bob, i.e. distance of c is lower than d. Thus, Charlie has a stronger channel to Eve, that he can jam using a Gaussian noise signal in such a way that the jamming affects Eve more than Bob (due to Eve having a stronger gain then Bob) [67]. Still inside of this group, instead the jamming be composed by Gaussian signal it can be formed by noise forward which consists in the jammer transmitting additional randomness in a form of random chosen codewords of a known codebook. In the last case, Bob can decode the confused signal (receiving a clean carrier with the knowledge of the codebook) while Eve remains jammed (because she does not know the codebook). This scheme has the advantage of being less harmful for Bob since in the Gaussian noise both Eve and Bob are jammed without any knowledge of the jammed signal while here Bob knows some codebook of the jamming signal.

In this type of jamming, depending in the channel conditions, Bob can also be completely jammed, thus not allowing it to receive the information. Since this is not desirable, it was proposed the usage of structured codes.

On the contrary of the last scheme, in *Cooperative Jamming with Structured Codes* [11, 68], a smaller distance is assumed from Charlie to Bob than to Eve. Therefore, a jam signal with a known structure is transmitted by Charlie in order to enable Bob exploiting the known code to retrieve and decode the signal from Alice. On the other hand, Eve receives a



(a) Gaussian noise cooperative jamming scheme.

(b) Structured codes cooperative jamming scheme.

(c) Alignment cooperative jamming scheme.

Figure 3.6: Scheme of cooperative jamming using Gaussian noise 3.6(a), by structured codes 3.6(b), and by alignment 3.6(c). Figures adapted from [67].

the signal that went through a bigger distance in the channel, hence, making it impossible of decode the sent signal. This strategy is represented in Fig. 3.6(b), where Alice has the code $X_1 = a_4 0 a_3 0 a_2 0 a_1 0 a_0$, Charlie has the code $X_2 = b_4 0 b_3 0 b_2 0 b_1 0 b_0$ and the distance from Charlie to Bob h is smaller than to Eve c, i.e. $h < c$. The chosen known codes have a structure in such a way that, they are aligned at Eve but separable on Bob. After the transmission, Bob receives $X_1 + X_2$ affected by the channel h, where both structured codes X_1, X_2 are not aligned (with respect to the positions of the zeros due to the shorter distance of h), hence, allowing Bob to recover data X_1 sent by Alice. Eve receives $X_1 + X_2$ affected by the channel c, with both codes aligned (in terms of the position of the zeros, due to a bigger distance channel), which results in a received sequence with the bits sent by Alice and the bits of the structured code added, i.e. each bit of the received sequence is correct (a_i) only if $a_i = b_i$, which occurs with probability of 0.5. This leads to the research of designing good structured codes to enable the increase of the achievable of the secrecy rate in a model where Bob is jammed too.

Cooperative Jamming with Interference Alignment assumes the existence of K jammers cooperating with each other to help securing the legitimate communication [12, 69]. This scheme is based on the last one adding interference alignment and using K-helpers. First Alice divide the message M into K sub-messages, each jammer sends the signal aligned at the same dimension to Bob occupying the smallest signal space possible to allow the maximum occupation possible of the message. Bob can then separate all K messages because they are in different orthogonal dimensions. Eve cannot recover because receives each jam signal aligned with the sub-messages guaranteeing that the information leak is upper bonded by a constant. In other words, each sub-message is protected by each aligned jamming signal of the helper. The Fig. 3.6(c) is demonstrated the working when $K = 2$.

All of these schemes share some disadvantages, such as:

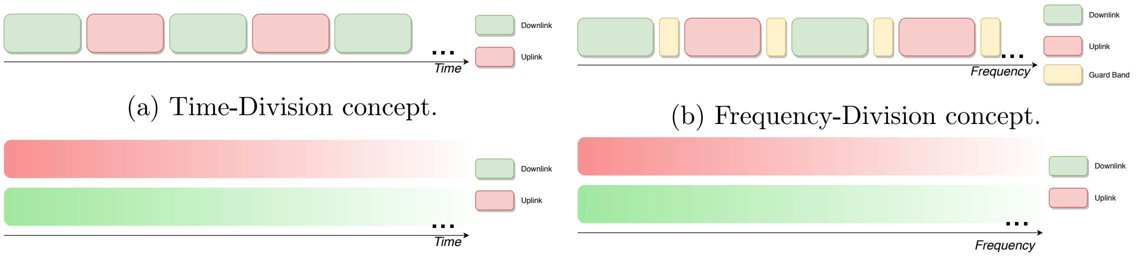
- Synchronization: the helper need to be in sync with Alice to correctly help Bob having the advantage and only spending the necessary amount of energy need to that effect;
- Charlie willingness: willingness of the helper to spend his own energy in order to emit the jam signal to make other communication secure;
- Trust relationship: there must be a trust relationship between the helper and Bob, because if he stops to jam signal, then Eve can retrieve all the information signal;
- Mobility: the mobility of the helper can be an issue since it may/will at some point lower the advantage added by the scheme.

Applying it closer to real-world, for instance, by using in smartphones, these schemes will drain battery faster and the helper cannot move while it is helping to not compromise the security of the other communication, thus reducing the willingness to cooperate.

As seen, these cooperative methods help to ensure the advantage of Bob over Eve by using external jammers, however have some drawbacks on their schemes. In this work, we address these drawbacks through a solution in which instead of using an external helper, the helper is the same device as the receiver. This helps to solve the synchronization, solves the issue of the need for cooperation, and facilitates the removal of interference, because the receiving device knows the characteristics of the interference as it will be discussed in Section 3.5. In short, in this thesis it will be explored the usage of the previously discussed security schemes on Section 3.3.1, specifically the SCS-HK scheme, along with interference generation by the receiver and its cancellation as presented in [58].

3.5 Full-Duplex and Self-Interference

The disadvantages referenced in Section 3.4 have led to the main contribution of this thesis which is use the legitimate receiver as a cooperative jammer, using the concept of Full-Duplex (FD) and Self-Interference (SI) cancellation. The legitimate receiver has the motivation to do it because he wants communicate in a secure way, therefore spends his own energy and has the responsibility to do the synchronization. To emit jam signal and at the same time receive both jam and the information signal it is only possibly by using full-duplex which is supported in Universal Software Radio Peripheral (USRP) B210.



(a) Time-Division concept.

(b) Frequency-Division concept.

(c) Full-Duplex concept at time view.

(d) Full-Duplex concept at frequency view.

Figure 3.7: The concept Time-Division is shown in 3.7(a), and the Frequency-Division concept in 3.7(b) and the time view of a Full-Duplex is shown in 3.7(c) and at frequency view in 3.7(d).

In order to increase data rates in a communication there are some advances already developed like Multiple-Input and Multiple-Output (MIMO) techniques (where a device has more than one antenna both in input and in output) [16] in a manner to enable the increase of the spectral efficiency. However, bidirectional transmission (usually designated by uplink and downlink transmission) requires the use of Time-division duplex or Frequency-Division duplex. In a Time-Division duplex (Fig. 3.7(a)) transmission is switched between uplink and downlink periodically, not enabling both at the same time while all the channel bandwidth is available during the communication. In Frequency-Division duplex it is possible to perform uplink and downlink communication, but where only part of the channel bandwidth is allocated for uplink, and the other part for downlink, with both separated by a "guard band" to avoid possible interference between them (Fig. 3.7(b)), and thus, working with lower rates. These schemes are named Half-Duplex (HD) and limit the theoretical maximum spectral efficiency of the channel. The FD promise to compensate the shortcomings of the HD system, by receiving and sending information continuously at the same time, as represented in Fig. 3.7(c), and at the same frequency band, as represented in Fig. 3.7(d), improving this way the spectral efficiency of wireless communications avoiding the utilization of two channels in an end-to-end transmission.

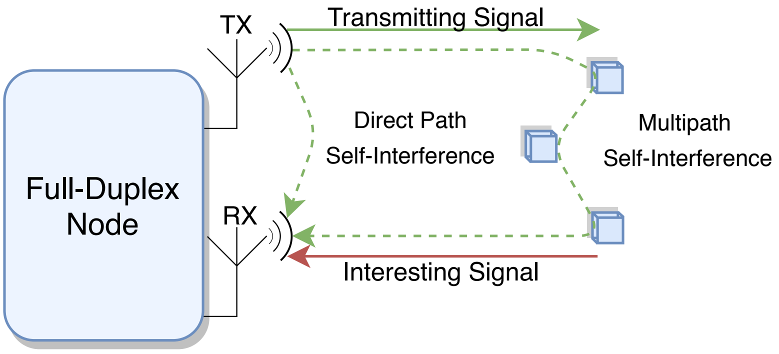


Figure 3.8: Self-Interference problem.

The feasibility of the usage of FD in wireless communications in practical systems was already shown in [15], but none was able to attain the theoretical double gain in terms of throughput on FD compared to HD, because it suffers of Signal to Interference and Noise Ratio (SINR) loss due to the SI caused by the TX antenna. The RX antenna is receiving, at the same time and frequency, both the information sent by another device to him (interesting signal) and the information he is sending on the TX antenna (SI signal). The problem of superposition of the two signals of the RX antenna arises because the power of the interesting signal is usually much lower than the power of the SI signal. Besides, the SI can arrive from a direct path or suffer multiple reflections on the ambient objects (multipath) as can be visualized on Fig. 3.8, making more difficult its cancellation.

It is a fact that when there is high SI, the FD presents a worse performance than the HD since SI leads to instability and oscillations of the system [70], with SI cancellation as the main critical and challenge problem to address in real-world deployment of FD systems [71, 72]. The FD has shown not only some advantages, but also some weakness compared to the HD mode as it will be discussed next in detail, and summarized in Table 3.1.

Starting with the advantages [16], since the communication is done at the same time and uses frequency band, then, in theory, the throughout can double compared to a single-hop in a wireless link, on a HD system.

In the Data Link Layer on the OSI model, there is a difference relatively to collision avoidance in the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol. In the HD it is required all nodes perform a quality channel sense before start a transmission, however in FD, only the first node that initiates the transmission performs channel sense necessary to collision avoidance in other FD nodes (that do not perform carrier sense).

There is a problem in HD called "hidden terminal problem", where is taking into consideration multiple nodes and one Access Point (AP). For instance, in an example of 2 nodes N_1, N_2 and 1 AP; all the nodes have a full transmission buffer and N_1 starts the transmission to AP and consequently the AP starts transmitting data back to N_1. Then other nodes, N_2, delay the transmission in order to avoid collision waiting until all data is transmitted between them. Even if the AP has no data to transmit, it is required to send acknowledgements to other nodes so they can know that he is unavailable, preventing this way to send him data. This problem is solved by the usage of FD techniques since the AP can at the same time receive data from N_1 and transmit data to N_2.

In the store-and-forward technique used by multi-hop networks with HD nodes (where a node receives several packets, stores them, and transmits them at the end of the transmission), cause the end-to-end delay to linearly increase as the number of hopes increases. When using FD, this technique is replaced by the ability to forward a partially received packet (while being received), this way reducing significantly the end-to-end delay in a network with multiple hops. This brought a reduction of the congestion because the nodes dispatches the received (or partially received) packet quicker.

As mentioned, the main FD disadvantage [16] is the SI. The SI is added by the fact of the same device is receiving information from one antenna at the same time and frequency that is transmitting information from another antenna. As the antennas are in the same device, they are really close degrading the FD gains, namely the transmitted signal is also, in a feedback way, received again with higher power that the information signal by the other node. So, the performance is dependent on how good is done the SI cancellation. Even with or without taking into account the SNR of the link, the reliability is significantly degraded when compared with HD. It was also shown that it can only achieve 88% of reliability with digital SI cancellation, and 67% without it [15].

The FD needs to process twice of the number of packets at a given instant because it needs to process both the TX and the RX packets simultaneously. This fact plus SI leads to a consequence of higher PLR. In terms of internal buffers, since the PLR is higher and we have double number of packets, it requires a sufficiently larger buffer in a way that can forward the packets or they will be discarded due to queue overflow, requiring this way a bigger buffer to reduce the PLR; therefore, more memory is needed when operating in FD mode.

Content	**HD**	**FD**
Throughput	Lower	Higher - in theory $2xHD$
SI	Avoided	Not avoided
Hidden Terminal	Suffered	Avoided
Congestion	Higher	Lower
End-to-end Delay	Higher	Lower
Queue Size	Smaller	Larger
PLR	Lower	Higher

Table 3.1: Comparison between HD and FD modes.

As seen FD relay has been used and experimented for communication systems without considering an eavesdropper. The application of FD along with jamming in order to improve security in communications using trusted relays was researched in [73]. In this scheme the FD relay is receiving and sending messages constantly, thus, Eve receives both the signal from the source and the signal from the relay at the same time. As the relay is a delayed version of the source, Eve will receive the signal with high Inter-Symbol Interference (ISI), degrading the decoding capacity.

Yet considering both FD and jamming, was recently researched to improve security in communication without any relay in [14] as an improvement of [13], where the receiver is self-protecting by acting as jammer in a way to prevent others to decode the sent signal. The *iJam* scheme proposed in [13] is simple: Alice repeats its transmission and for each sample, Bob jams either the transmitted or the complementary sample. As Eve does not know which one is clean so she cannot decode it, while Bob can select the correct samples and its repetition to rearrange them to get a clean signal. The main problem of this scheme is that as was not using FD to achieve the self-protection; being thus necessary retransmission of the source signal which lowers the throughput. In [14] was improved this scheme by using FD and to cancel the SI they studied *"the optimal jamming covariance matrix at the destination (...) and power allocation"* between Alice and Bob.

3.6 Self-Interference Cancellation

As it was already discussed, the goal of FD is enabling transmission and reception at the same time while sharing the same frequency band, with the receiver acquiring not only the signal of interest but also the transmitting one, thus suffering from SI. Because TX signal has higher power relatively to RX interesting signal (may be 50-100 dB), then the noise plus interference will dominate over the interesting signal, hence resulting in lower SINR in the channel [17]. Furthermore, to successfully retrieve the signal of interest it is required to sufficiently reduce the SI signal to decode the interesting one correctly, so the goal of SI cancellation is *"predict and model the distortions in order to compensate for them at the receive antennas"* as indicated on [74].

The SI cancellation is not a simple linear operation due to possible distortions on the signal representation, namely, both linear and nonlinear distortions. Signal attenuations and reflections on the environment (multipath problem) are considered as linear distortions, and circuit power leakage, higher order signal harmonics, nonflat hardware frequency responses are considered nonlinear distortions [16]. These types of distortions will need to be taken care on the SI cancellation operation as will be discussed next.

There is considered three different categories of SI cancellation, namely, passive suppression, analog cancellation and finally digital cancellation as visualized in Fig. 3.9 [16].

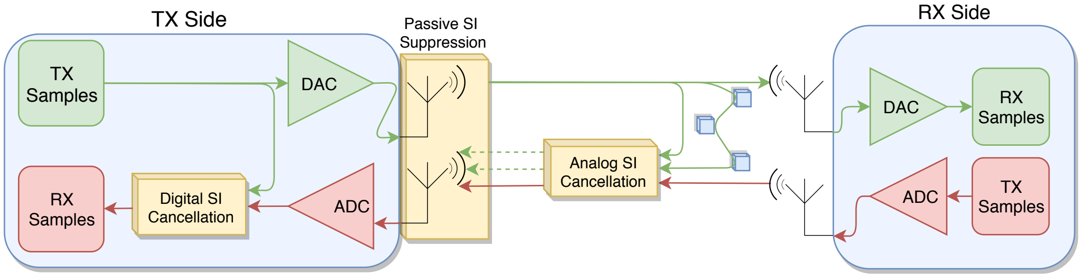


Figure 3.9: Active and Passive SI cancellation mechanisms.

3.6.1 Passive Self-Interference Suppression

The *Passive SI Suppression* is based on attenuating the SI signal by physically separate/isolate/change somehow the TX the RX antennas of the device as mentioned in [75]. If the electromagnetic coupling between both antennas is reduced, then, the SI signal power it will also be weaken, consequently obtaining a better SINR. Also, it may be used techniques for directing the lobes (beamforming) of the transmitters and receivers antennas in different directions [76].

The three possible ways to perform passive SI suppression are represented in Fig. 3.10 and are: *Antenna Separation*, *Antenna Cancellation*, and *Directional Passive Suppression*.

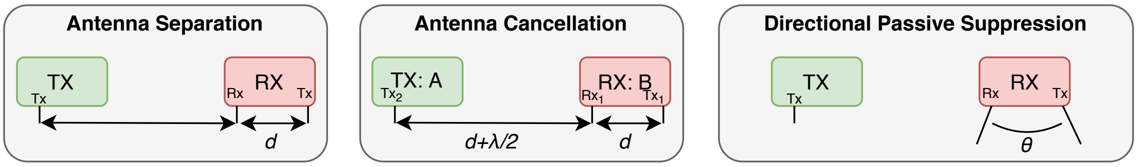


Figure 3.10: Representation of the difference between Passive Suppression mechanisms.

The most intuitive, is using the *Antenna Separation* where is physically increased the separation distance between the TX and the RX antennas of the same device. The path-loss effect measures the power attenuation of the electromagnetic wave as it propagates through space, so, since the path-loss effect between those antennas is increased then there is an attenuation on SI signal power enabling a better recover of the interesting signal. This mechanism is highly dependent of the physical constraints of the system as there is a need to physically separate the antennas. For instance, in transceivers having small dimension (e.g. mobile) may not be possible to apply this mechanism. However, in others (e.g. relay systems) may be possible to do it and achieve a significant amount of reduction.

When using big antenna separations distances ($4-6$ m), it was shown in [77] to be possible to achieve 80 dB of suppression of SI when combined with directional suppression (that will be discussed next). A more realistic scheme was studied in [78], considering communication at 2.6 GHz between an outdoor base station to indoor users, where it was achieved 48 dB of isolation for a compact relay with antennas attached in opposite sides, and 70 dB of isolation when using 5 meters of antenna separation between both antennas while guaranteeing the best orientation possible.

More practical lower separation distances, such as 20 cm and 40 cm were first explored in [75] achieving 39 dB and 45 dB, gains in SI reduction respectively. This concludes that antenna separation alone is not enough to reduce the SI signal to a level below of the signal of interest to correctly decode the information. To achieve this, it was also proposed in [75] alternative schemes that combine antenna separation with digital SI cancellation (ASDC), analog SI cancellation (ASAC) and both (ASADC), increasing the SI cancellation to 70, 72 and 78 dB for the 20 cm case and $76, 76$, and 80 dB for the 40 cm.

In conclusion, the bigger the distance between the antennas of the same device, the higher is the SI suppression achievable. However, this isolation values may not be enough for a FD system (specially the compact relay), but it is an help to increase SI cancellation when adding other techniques.

Another way is exploiting the distances between the antennas of both devices considering the wave's length, this is called as *Antenna Cancellation* technique proposed in [15]. For instance, considering a receptor device A which has 2 antennas $TX1$ and $RX1$ and another device B, which is only transmitting with the $TX2$, then the distance between both antennas on A would be d and then the distance between the $RX1$ and $TX2$ differ exactly $d + \lambda/2$, with λ being the wave length. Then d is set in such way to get the TX's distance being an odd multiple of $\lambda/2$, where the signal is destructively superimposed canceling one another. Also in the same paper, it was shown that antenna cancellation alone can provide about 30 dB of interference suppression, and when adding active and passive cancellation increases up to 60 dB of SI reduction.

The last method is exploiting the angle in the antennas in a way that the main radiation lobes in the antennas has minimal intersection. This is called *Directional Passive Suppression* presented and was experimented on [76]. In this paper this mechanism is combined with digital cancellation considering a scenario where the distance between the TX's and the RX's antennas of the same device was set to 10 meters, and the angle between them being $\geq 45^o$. With this configuration, the FD performs 60% better than the HD, while a maximum of 95% improvement was obtained when angle difference was set to 120^o angle and 10 meters of distance.

However, these passive SI suppression techniques alone are not sufficient to guarantee the successfully decoding the intended signal. In addition, although it is not infeasible to do it, it is no practical at all physically separate the antennas after some distance value.

These methods exploring antenna's distances would be a good option to implement if, the Eve's position is known (which usually is not) and it is allowed to have a big distance between them. In this case, it is possible to set the TX antenna closer to Eve creating more jamming power to her, and at the same time lower SI on Bob. This enables, by one side more throughput due to less computer power need on the cancellation operation, and by another side better performance results in terms of security against Eve. However, since our objective is to have a compact receptor, thus, it did not make sense to explore big distances between antennas. However, it is possible to explore the passive suppression based on the angle between antennas, allowing a compact node with an angle between them. This solution will be explored in this work.

3.6.2 Active Self-Interference Cancellation

As the passive method alone is insufficient, it is necessary the use of *Active SI Cancellation* techniques on top of the passive ones as an increment and not a replacement. Active SI cancellation techniques use the knowledge of SI signal to cancel it from the received signal.

They can be carried at both Analog or Digital domain. Since this work focuses on an SDR implementation using GNU Radio (GR) we will address digital SI cancellation techniques in more detail.

Regarding *Analog SI Cancellation* there is mainly two ways to do it: 1) Using a Balun cancellation (balanced-to-unbalanced converter) proposed in [79] to obtain the inverse of the self interfering signal, and using it to cancel the received signal; 2) Using a QHx220 noise canceling integrated circuit to remove the known self-interference as was used in [15]. This mechanism takes into account the multipath problem where the same signal comes from different routes as mentioned on Section 2.4.3. In particular, it is added another transmission chain in the FD node which is applied an array of variable attenuators to adaptively adjust a copy of the transmitted signal into the receiving chain. Since the signal feedback into the receiver antenna is imperfect (e.g. time/phase delay or amplitude attenuation) it is necessary the array mentioned to adjust these imperfections. This situation is described as $y(t) = \sum_i a_i(t)x(t - \theta_i(t))$ where $a_i(t)$ is the attenuation and $\theta_i(t)$ is the delay, both on i route as described in [80].

The *Digital SI Cancellation* is an active mechanism for SI cancellation performed after Analog to Digital Converter (ADC), taking as advantage the knowledge of the interfering signal for the purpose of canceling it. The most recent researches concentrate in antenna placement and in analog SI cancellation since the digital cancellation usually shows a limited performance; for instance in [15] it is reported a suppression of only around 10 dB of SI signal reduction, when adding digital SI cancellation.

The leaked SI from imperfect analog SI cancellation [1], can be divided in linear plus a nonlinear component. The former is where is located the main SI power and the estimating mechanisms are called as *Estimation of the Linear Component of the SI Leakage-Channel*; the latter results from the distortions caused by the hardware, and the estimating mechanisms are called as *Estimation of the Nonlinear Components of the SI Leakage-Channel.*

The second mechanism is the less important since the nonlinear distortions are relatively weak and it is really hard to estimate random happenings on the hardware. So we will focus in the first mechanism which it will be the one implemented in this thesis. There are some known algorithms to estimate and correct the SI channel distortions and can be divided in block-based and adaptive algorithms. The first one operates in an entire matrix A (which represents the known samples of the signal transmitted) at once, thus effectively computing $\vec{x}$ (compute a fixed array of attenuators) by doing the pseudo-inverse of A, notated as A^+ and multiplying by a vector $\vec{b}$ (the received signal values with SI), resulting in $\vec{x} = A^+\vec{b}$ [81]. Algorithms like "Singular Value Decomposition", "QR decomposition", "Cholesky decomposition", "Conjugate gradient method" are examples of this type. The second one operates in a row of A each time and iteratively adjust the values of $\vec{x}$ updating them and driving the predicted signal closer to the desired one (by find $\vec{x}$ that minimizes $A\vec{x} - \vec{b}$). Adaptive algorithms based on least-square (e.g. Recursive Least Squares (RLS)) and Minimum Mean Squared Error (MMSE) (e.g. Least Mean Squares (LMS)) are used, since in practice has better performance results in SI cancellation as shown in the GR conference of 2017 [81]. Furthermore in [82] was created an adaptive filter based on LMS filter where they was able to cancel 20 dB of SI which is superior to the 10 dB [15] previously mentioned.

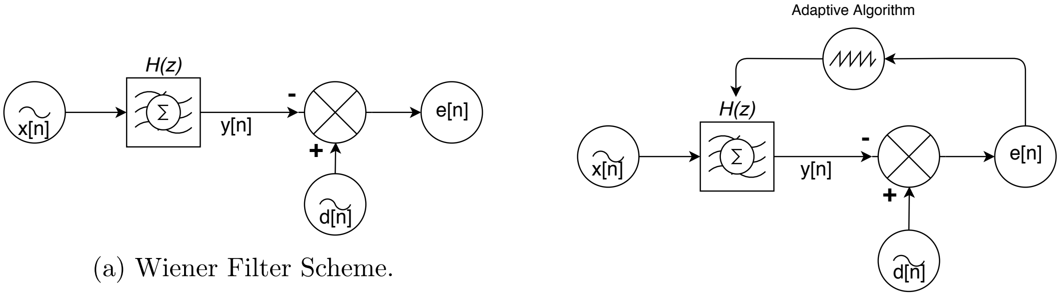
[1] Which corresponds to the total SI signal when no analog SI cancellation is used, as it is the case of this thesis.

3.6.2.1 Least Mean Squares Algorithm

The LMS algorithm is included within the digital SI cancellation techniques and has its roots on the concept of Finite Impulse Filter (FIR) Wiener filter.

The Wiener filter [83] filtering technique, in which given a target/reference signal $d(n)$ and with $x(n)$ and $y(n)$ being the input and output of the filter respectively, the filter transfer function $H(z)$ is defined (taking into account $x(n)$ and $d(n)$ statistics) in order to $y(n)$ approximate $d(n)$. This is, after $x(n)$ is filtered, is produced $y(n)$ which is an estimation of $d(n)$ with some error of $e(n)$. Particularly, it uses the MMSE which is an estimation method aimed to minimize the mean square error. The mean square error is used as measure of the quality estimator by averaging the square of the difference between the estimated and real values. The scheme is represented on Fig 3.11(a). It is important to note that the target signal needs to be stationary (i.e. it is a stochastic process where the probability distribution does not change along time).

The adaptive FIR Wiener filter has the same goal as the previous one, but where filter coefficients are continuously recalculated/adjusted by an adaptation process that aims to minimize a cost function that measures the difference $e(n)$ between the adaptive filter output $y(n)$ and the reference signal $d(n)$ [84]. The adaptive FIR Wiener filter estimates the target signal by using auto-correlation and cross-correlation operations, thus, for a given input matrix it produces a vector using both input and output signal statistics (also uses MMSE) of the filter in order to calculate the taps' weights granting adaptive capabilities. This input matrix is initially populated with estimates of auto-correlation of the input signal while the output vector is populated with estimates of the cross-correlation between the input and output signals.



(a) Wiener Filter Scheme.

(b) Adaptive FIR Wiener Filter Scheme.

Figure 3.11: The Wiener Filter scheme is represented in 3.11(a), and the adaptive FIR Wiener Filter in 3.11(b). Figures adapted from [83] and [85].

The adaptive FIR Wiener filter scheme is represented in Fig 3.11(b). Following [85], by denoting h_k the coefficients of the FIR filter with order p, this can be represented as $H(z) = \sum_{k=0}^{p} h_k z^{-k}$. The output of the filter having $x(n)$ as input, is given by $y(n) = \sum_{k=0}^{p} h_k x[n-k]$. Intuitively, the error $e(n)$ is the difference between the target signal and the estimated one, i.e. $e(n) = d(n) - y(n) = d(n) - \sum_{k=0}^{p} h_k x[n-k]$. Based on the auto-correlation on $x(n)$ and cross-correlation between $x(n)$ and $d(n)$, are used in calculating the $p+1$ taps $h_k, k = 0, ..., p$ by using mean square of the error as cost function $\mathscr{E}(n)$ in order to minimize it, $\mathscr{E}(n) = E\{|e(n)^2|\} = E\{|d(n) - y(n)|^2\}$.

The LMS algorithm [86] was invented by Bernard Widrow in 1960 together with his student Ted Hoff and it is based on the concept of the adaptive FIR Wiener Filter. Instead of using auto-correlation and cross-correlation operations to estimate the signal, the LMS uses a the steepest descent method where uses estimates of the gradient vector and constantly updates the filter taps' weights in a way to converge to the optimum filter weight vector while minimizing the mean square error (to converge to the MMSE). In practical view, this algorithm is simpler since it does not use correlation techniques neither require matrix inversions, which are computational heavy.

Taking into account the method of steepest descent, the taps' weight vector is given by $h(n+1) = h(n) + \frac{1}{2}\mu[-\nabla(E\{e^2(n)\})]$, where the $e^2(n)$ is the mean square error. In short, the LMS algorithm at each instant n needs to compute: 1) The error $e(n)$; and 2) The next tap's weight $h(n+1)$; with these being given by [86]:

1. $e(n) = d(n) - h^H(n)x(n)$
2. $h(n+1) = h(n) + \mu x(n)e^*(n)$

Where h^H denotes the Hermitian transpose of h and e^* the conjugate of e. Finally, this algorithm is considered a nonlinear feedback control system known as "double-edged sword" since it can work for us or against us, hence, the μ step-size parameter affects directly the control mechanism which plays a critical role on the convergence and stability of the algorithm (normally, is used the 0.0075 value) [87].

3.6.2.2 Normalized Least Mean Squares Algorithm

The Normalized Least Mean Squares (NLMS) is a version slightly modified of the LMS. In the LMS previously discussed we got a final equation do determinate the weight update operation of the tap given by $h(n+1) = h(n) + \mu x(n)e^*(n)$. This means that the weight's adjustment of the taps is dependent of three main terms: 1) The step-size parameter, μ; 2) The input vector $x(n)$; and 3) The estimation error $e(n)$.

As it is possible to visualize, the weight's adjustment is directly proportional to the input $x(n)$, therefore, when that input is large the algorithm may suffer of "gradient noise amplification" problem [87]. This means that the normal LMS algorithm is sensitive to the scaling of its input making it harder to choose an appropriate μ value (that guarantee stability).

To overcome this problem, the weight's adjustment on NLMS is normalized with squared Euclidean norm of the input vector. This results in a final equation of $h(n+1) = h(n) + \frac{\widetilde{\mu}}{\|\mu\|^2}x(n)e^*(n)$, with $\widetilde{\mu}$ as a positive real scaling factor. If we compare with the original equation, the big difference is that now the step-size parameter is time-varying rather than fixed what makes it converge faster [88].

While resolving the "gradient noise amplification" problem the NLMS introduce a new one, namely, when the input vector $x(n)$ is too small, it may raise numerical issues when trying to divide by a small value of $\|\mu\|^2$. This is easily fixed by adding δ, with $\delta > 0$ in the denominator part, resulting in: $h(n+1) = h(n) + \frac{\widetilde{\mu}}{\delta+\|\mu\|^2}x(n)e^*(n)$ [87].

3.6.2.3 Recursive Least Squares Algorithm

The RLS algorithm was initially discovered by Gauss in 1821, however it was not used until 1950, when Plackett rediscovered it. While the LMS aims to reduce the mean square error between the estimated signal and the desired one, the RLS tries to recursively find the taps' weights using the least squares method. As previously mentioned, the LMS input signal needs to be stochastic, while the RLS works better for a deterministic environment (i.e. there is no randomness on the input signal or has a slowly time-varying randomness).

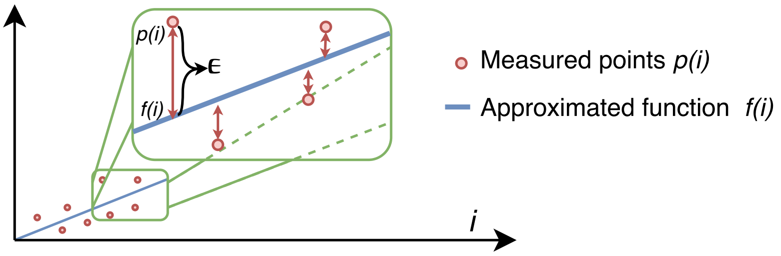


Figure 3.12: Linear Regression Representation using Least Squares Method.

The method of least squares aims to find the best-fitting curve through a set of points by minimizing the sum S of the squares of the difference ϵ between the approximation $f(i)$ and the measured point $p(i)$, with $i = 1, ..., N$, leading to: $S = \sum_{i=1}^{N} \epsilon_i^2$. It is the most commonly method applied in linear regression as represented in Fig. 3.12. This method is often applied to solve linear filtering problems, and it is viewed as an alternative of Wiener filtering due to its deterministic approach. The cost function when applied to the RLS filter can be reduced into weighted least-squares [87]: $\mathscr{E}(n) = \sum_{i=i_1}^{n} \lambda^{n-i}|e(i)|^2$, where $e(i)$ is the difference between $d(i)$ and $x(i)$, and λ^{n-i} is the weighting/forgetting factor used to ensure "old" data is forgotten as it will be explained ahead.

The RLS taps' weight updating mechanism is a lot more complex than the LMS. The idea of RLS's filter is minimize the cost function $\mathscr{E}(n)$ of least squares by tweaking the taps' weights (on the filter) in order to to model the input signal into a desired one. Therefore, taking into account $\mathscr{E}(n)$, after mathematical manipulations detailed in [87], the updating mechanism of the filter at each instant n, requires the computation of: 1) The gain vector $k(n)$; 2) The *priori* estimation error $\xi(n)$ (a "tentative" value of the *a posteriori* estimation error $e(n)$ mentioned above calculated before update the taps' weights); 3) The tap's weight $h(n)$; and finally, 4) The update of the inverse correlation matrix $P(n)$ of the input signal $x(n)$; with the following equations [87]:

1. $k(n) = \frac{\lambda^{-1}P(n-1)x(n)}{1+\lambda^{-1}x^H P(n-1)x(n)}$
2. $\xi(n) = d(n) - h^H(n-1)x(n)$
3. $h(n) = h(n-1) + \xi^*(n)k(n)$
4. $P(n) = \lambda^{-1}P(n-1) - \lambda^{-1}k(n)x^H(n)P(n-1)$

Note that δ, $(\delta > 0)$ is the regularization factor which is used for initialize $P(0)$, λ, $(0 < \lambda \leq 1)$, is the forgetting factor. Furthermore, for a high SNR of the input signal, a small value of δ should be used in order to achieve stability faster, and vice versa. However, if there is an abrupt change of the input's SNR the algorithm will take much time to stabilize, thus, in this situation the best option is restart it taking into consideration the new conditions. Finally, the forgetting factor λ indicates the contribution of the previous samples to the filter; hence, smaller λ makes the algorithm more sensitive to recent samples than the old ones. Normally it is chosen in $[0.98, 1]$ range [89].

The RLS algorithm uses the inverse correlation matrix $(P(n))$ of the input, raising its main feature: the rate of convergence is typically faster than that of the LMS algorithm. However, this performance increase is achieved at the expense of an increase in computational complexity, and therefore, it has lower throughput than LMS algorithm [87].

3.6.2.4 QR Decomposition Recursive Least Squares Algorithm

With the QR decomposition algorithm it is possible to decompose a matrix A into a product of an orthogonal matrix Q and an upper triangular matrix R, $A = QR$. This is often used to solve the linear least squares problem, and when combined with adaptive filters, it enables the exploration of good numerical properties, hence, achieving a numerically more stable algorithm. The good numerical properties are inherited from the QR-decomposition part of the algorithm; the numerical properties raise when the algorithms are numerically implemented, i.e. due to inaccuracy of the ADC.

The eigenvalue spread is defined by the ratio between the maximum and minimum eigenvalues of the input auto-correlation matrix. The RLS previously discussed, although having a fast convergence speed, it loses that property when the inverse correlation matrix ($P(n)$) loses its positive definiteness or Hermitian symmetry [89]. The QR Decomposition Recursive Least Squares (QRD-RLS) is similar of the RLS, however, instead using the $P(n)$, it uses the QR decomposition algorithm directly on the input matrix, thus, showing a lower dependence of the eigenvalue spread, being this its main advantage (at cost of more computational resources).

3.6.3 Overview

In this section we consider three different categories to perform SI cancellation, namely, passive suppression, analog cancellation and finally digital cancellation. As discussed, in this work it will not be implemented any analog SI cancellation due hardware/testbed constraints; however it will be tested some passive SI suppression techniques discussed, in particular, the *Directional Passive Suppression*.

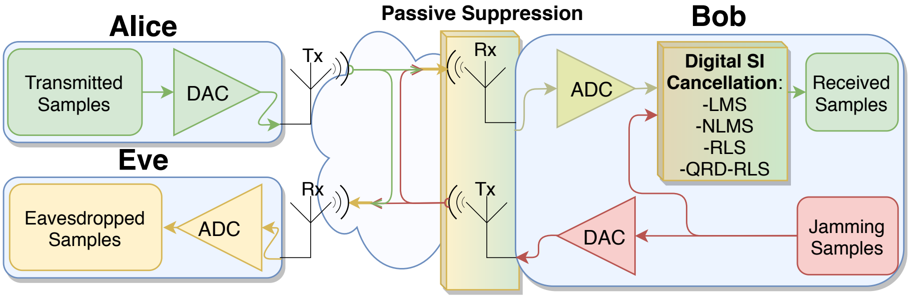


Figure 3.13: Conceptual scheme for implementing Self-Interference Cancellation.

Considering digital SI cancellation, we will evaluate the recovery perform of the LMS, NLMS, RLS and QRD-RLS algorithms that takes as knowledge the noise generated to estimate the transmitted signal. The conceptual scheme for the combination of SI cancellation mechanisms that will be implemented is represented on Fig. 3.13.

Chapter 4

Implementation

This chapter address all the implementation component of this work. This includes first the implementation of the adapted Scrambled Coding for Secrecy with a Hidden Key (SCS-HK) scheme discussed in Section 3.3 and further improvement of physical layer security by the use of Full-Duplex (FD) jamming in the legitimate receiver along with Self-Interference (SI) cancellation mechanisms mentioned in Section 3.6.

4.1 SCS-HK Implementation

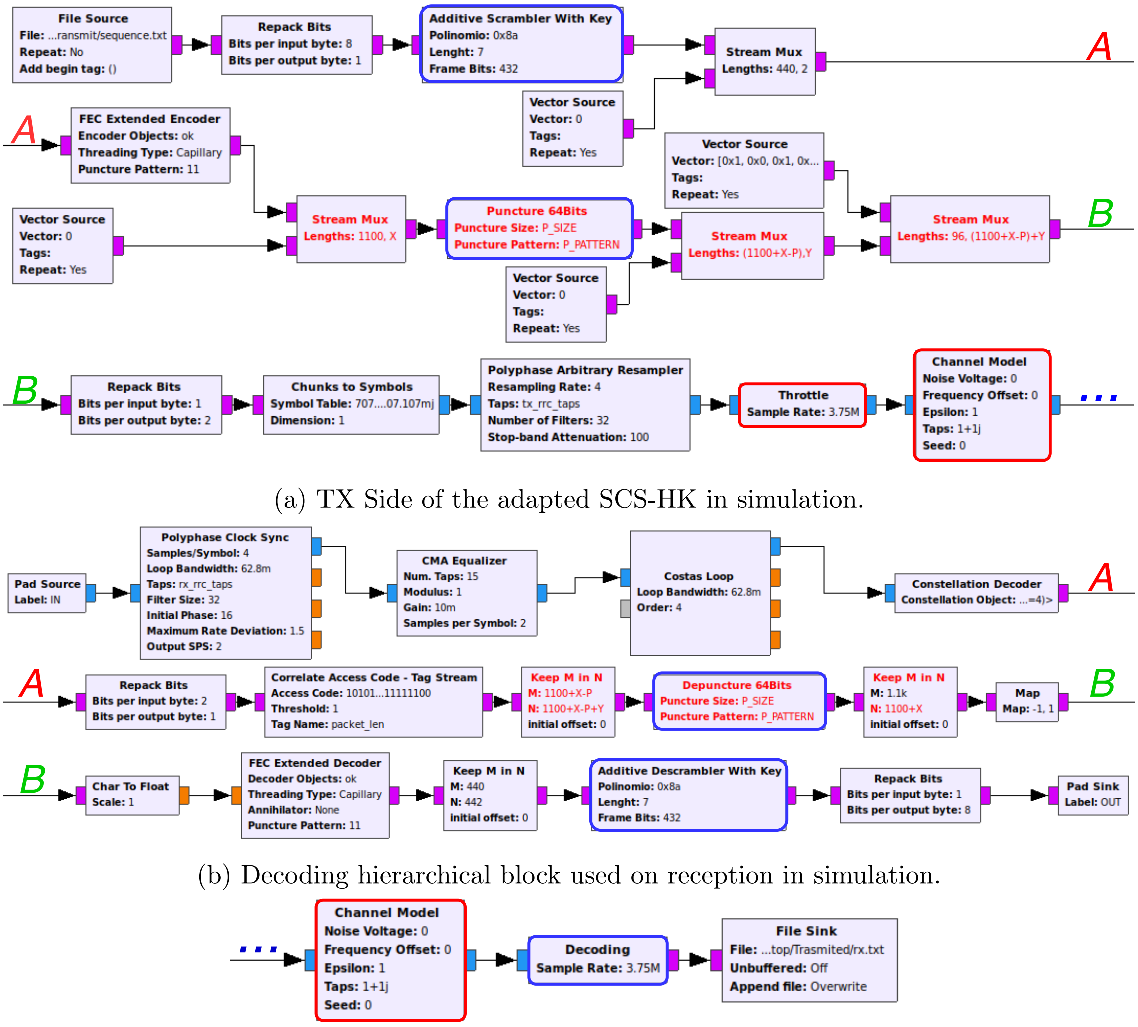
The implementation will follow the SCS-HK scheme presented and adapted in Section 3.3, represented in Fig. 3.5. Before going to real wireless transmission by using the Universal Software Radio Peripheral (USRP), the scheme was simulated in GNU Radio (GR) in order to find what were the main challenges and how much was required to change the FlowGraph (FG) when moving from a simulation into a real wireless transmission of the desired data (i.e. text, images and even video streaming).

4.1.1 Simulation

As mentioned in Section 2.1.2, when working in simulating mode on GR, the sampling rate is as high as the Central Process Unit (CPU) can process the samples, which may lead to a non-responsive program and other problems, like hiccups on the stream of data. Therefore, it is required the usage of a block called **Throttle** to limit and control the sample rate flow, thus, not allowing the FG running exhaustively.

Also, note that this block is not necessary when using wireless since the samples production/consumption rate is defined in the USRP source/sink blocks at a fixed given number. In this case, the samples are dropped when the income sample rate is higher than the fixed one, and on the other hand, the USRP starves if the income samples arrive with a rate lower than the fixed sample rate, as it will be discussed in the next section.

In simulation, the Transmitter (TX) and the Receiver (RX) components must be all together in the same FG in order to the **Throttle** block properly manage the sample rate flow. However, for explanation purposes it was divided the FG into three main ones. In the Fig. 4.1(a) is represented the TX side, and in the Fig. 4.1(c) is represented the RX side (which can be either Bob or Eve). Note that the last has a block called **Decoding** which is an hierarchical block containing all the core work of the reception side as it is represented on Fig. 4.1(b). Furthermore, all the packet's components and the respective lengths can be visualized on Fig. 4.2. Note that the blocks circled in red are only needed in simulation, being removed when moving to a real wireless transmission, and the blocks circled in blue were the ones implemented from scratch within the scope of this thesis.



File Source
File: ...ransmit/sequence.txt
Repeat: No
Add begin tag: ()

Repack Bits
Bits per input byte: 8
Bits per output byte: 1

Additive Scrambler With Key
Polinomio: 0x8a
Lenght: 7
Frame Bits: 432

Stream Mux
Lengths: 440, 2

Vector Source
Vector: 0
Tags:
Repeat: Yes

FEC Extended Encoder
Encoder Objects: ok
Threading Type: Capillary
Puncture Pattern: 11

Vector Source
Vector: [0x1, 0x0, 0x1, 0x...
Tags:
Repeat: Yes

Vector Source
Vector: 0
Tags:
Repeat: Yes

Stream Mux
Lengths: 1100, X

Puncture 64Bits
Puncture Size: P_SIZE
Puncture Pattern: P_PATTERN

Stream Mux
Lengths: (1100+X-P),Y

Stream Mux
Lengths: 96, (1100+X-P)+Y

Vector Source
Vector: 0
Tags:
Repeat: Yes

Repack Bits
Bits per input byte: 1
Bits per output byte: 2

Chunks to Symbols
Symbol Table: 707....07.107mj
Dimension: 1

Polyphase Arbitrary Resampler
Resampling Rate: 4
Taps: tx_rrc_taps
Number of Filters: 32
Stop-band Attenuation: 100

Throttle
Sample Rate: 3.75M

Channel Model
Noise Voltage: 0
Frequency Offset: 0
Epsilon: 1
Taps: 1+1j
Seed: 0

(a) TX Side of the adapted SCS-HK in simulation.

Pad Source
Label: IN

Polyphase Clock Sync
Samples/Symbol: 4
Loop Bandwidth: 62.8m
Taps: rx_rrc_taps
Filter Size: 32
Initial Phase: 16
Maximum Rate Deviation: 1.5
Output SPS: 2

CMA Equalizer
Num. Taps: 15
Modulus: 1
Gain: 10m
Samples per Symbol: 2

Costas Loop
Loop Bandwidth: 62.8m
Order: 4

Constellation Decoder
Constellation Object: ...=4)>

Repack Bits
Bits per input byte: 2
Bits per output byte: 1

Correlate Access Code - Tag Stream
Access Code: 10101...11111100
Threshold: 1
Tag Name: packet_len

Keep M in N
M: 1100+X-P
N: 1100+X-P+Y
initial offset: 0

Depuncture 64Bits
Puncture Size: P_SIZE
Puncture Pattern: P_PATTERN

Keep M in N
M: 1.1k
N: 1100+X
initial offset: 0

Map
Map: -1, 1

Char To Float
Scale: 1

FEC Extended Decoder
Decoder Objects: ok
Threading Type: Capillary
Annihilator: None
Puncture Pattern: 11

Keep M in N
M: 440
N: 442
initial offset: 0

Additive Descrambler With Key
Polinomio: 0x8a
Lenght: 7
Frame Bits: 432

Repack Bits
Bits per input byte: 1
Bits per output byte: 8

Pad Sink
Label: OUT

(b) Decoding hierarchical block used on reception in simulation.

Channel Model
Noise Voltage: 0
Frequency Offset: 0
Epsilon: 1
Taps: 1+1j
Seed: 0

Decoding
Sample Rate: 3.75M

File Sink
File: ...top/Trasmited/rx.txt
Unbuffered: Off
Append file: Overwrite

(c) RX Side of the adapted SCS-HK in simulation.

Figure 4.1: The simulation of the adapted SCS-HK divided in TX side 4.1(a) and in the RX side on 4.1(c). The last uses the Decoding block represented on 4.1(b).

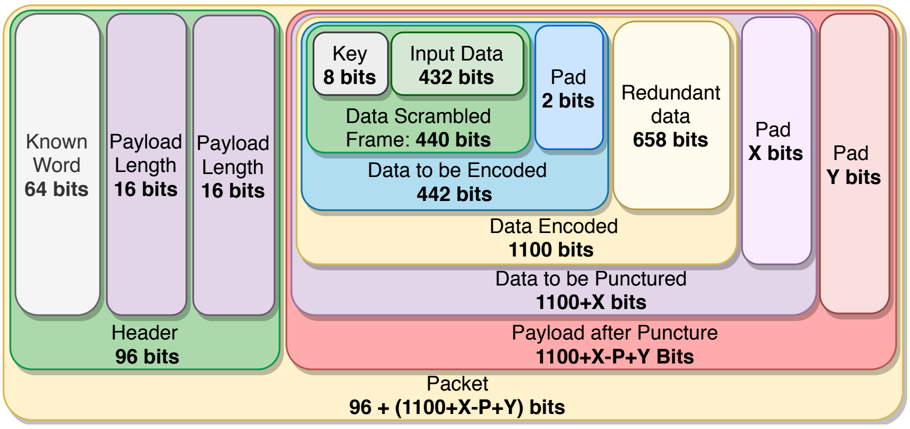


Figure 4.2: Representation of the packet's components and the respective lengths for SCS-HK scheme.

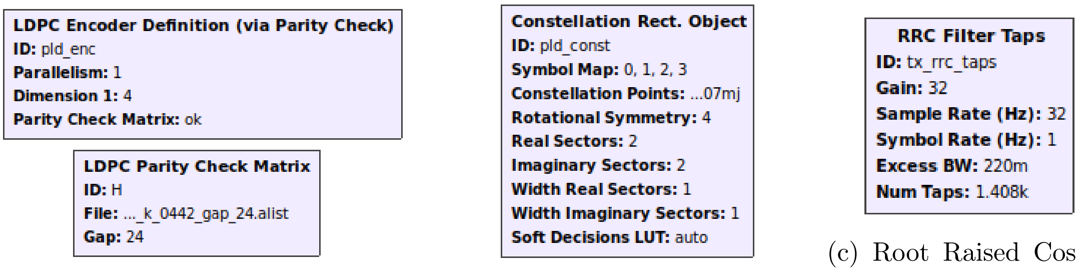
4.1.1.1 Transmitter Side

The first block **File Source** picks the binary data of the file and outputs samples where each sample is a byte with 8 bit of relevance, i.e. all 8 bits are information. However, to use the scrambler operation mentioned in Section 2.6, and the encoding mentioned in Section 2.5, it is required to input only 1 bit of relevance for each sample. The **Repack Bits** is a block that transforms some amount of relevance bits for each input byte into a desired number of relevance bits in the output byte while takes into account the endianness. We need to use this block to transform 8 bits of the input byte and output 8 bytes with only 1 relevant bit each while the remaining ones are set at 0's (with Most Significant Bit (MSB) of endianness), thus corresponding to an interpolation by 8.

Now the samples go through the scrambler (**Additive Scrambler with Key**) to both improve security (using the seed as key) and allowing the symbol synchronization on RX side (to solve the whitening sequences problem as discussed on Section 2.6). GR has an implementation of both multiplicative and additive scramblers stock blocks, but the input is treated as a continuous stream, i.e. as there is never a seed reset, which means that the output bit of the Linear-Feedback Shift Register (LFSR) is always dependent of the input bit (as can be visualized on the Fig. 2.14(a) and Fig. 2.15); therefore, one bit error leads to multiple bit errors on the remaining stream. As so, if a packet is lost in the transmission then some of the initial bits of the next packet (if using the multiplicative scrambler) or all the remaining packets (if using the additive scrambler) will be lost. To solve this problem it is required to packetize the output data of the scrambler, i.e. first take a pre-established seed and scramble a frame of bits, then reset the seed to the pre-established value and scramble another frame, and so on (using always the same seed to initiate the registers in packets creation). Additionally, the SCS-HK uses a random key as seed, as mentioned on Section 3.3.1. Since the key size must be the same size as the number of register of the LFSR, the seed will need to have 8 bits randomly generated (not a pre-established fixed as previously mentioned) and concatenate the frame (of $440 - 8 = 432$ bits) to the key.

Note that the **Additive Scrambler with Key** block was implemented in the scope of this thesis and can be found in [19]. A tricky detail that is faced on scrambler is that at first the registers contain the seed, thus, the first output byte will be junk scrambled. This problem it is easily fixed by dropping the first byte after the data is scrambled.

The **FEC Encoder** block defines which type of channel encoder is used. Since the SCS-HK security technique makes use of Low-Density Parity-Check Code (LDPC) (detailed in



LDPC Encoder Definition (via Parity Check)
ID: pld_enc
Parallelism: 1
Dimension 1: 4
Parity Check Matrix: ok

LDPC Parity Check Matrix
ID: H
File: ..._k_0442_gap_24.alist
Gap: 24

(a) Forward Error Correction (FEC) encoder definition objects.

Constellation Rect. Object
ID: pld_const
Symbol Map: 0, 1, 2, 3
Constellation Points: ...07mj
Rotational Symmetry: 4
Real Sectors: 2
Imaginary Sectors: 2
Width Real Sectors: 1
Width Imaginary Sectors: 1
Soft Decisions LUT: auto

(b) Constellation definition object.

RRC Filter Taps
ID: tx_rrc_taps
Gain: 32
Sample Rate (Hz): 32
Symbol Rate (Hz): 1
Excess BW: 220m
Num Taps: 1.408k

(c) Root Raised Cosine (RRC) definition object on TX side.

Figure 4.3: Definition objects of the TX side of the SCS-HK scheme.

the Section 2.5.2), we performed a search for which object blocks could be referenced in the **FEC Encoder** block in order to implement LDPC encoding. The GR has three objects devoted to LDPC encoding which are:

- **LDPC Encoder Definition**: use a parity check algorithm with parity chech matrix defined by an alist file.
- **LDPC Encoder Definition (via Parity Check)**: receives a prebuilt H matrix from the **LDPC Parity Check Matrix** object block that receives as input a given alist file. This is the same as before but with a better implementation and a cleaner look.
- **LDPC Encoder Definition (via Generator)**: receives a prebuilt G matrix from the **LDPC Generator Matrix** object block. This object constructs a generator matrix, G, from a given alist file.

The second object block mentioned above uses a reduced complexity algorithm compared to the last one, which requires heavier operations at each encoding step. This improvement is accomplished by performing a significant amount of the complex matrix manipulation (i.e. inverse, multiplication, and Gaussian elimination operations) during preprocessing. However, as disadvantage, this encoder requires a specially formatted matrix as input [90]. It is also important to note that a prebuilt of alist files (containing the necessary matrix) are distributed and installed with GR.

Taking into account the better performance of the **LDPC Encoder Definition (via Parity Check)** we choose to use it, as represented in Fig. 4.3(a). Also, taking into account the code's rate and the size of each packet, it will be used the $H(1100, 442)$ matrix (i.e. for each 442 bits, outputs 1100 bits), thus corresponding to a rate of 442/1100. As the required input is 442 bits on the encoder and we actually have 440 bits (scrambled frame 432 plus key 8), it is necessary to add 2 extra bits to allow the independence of bytes (useful when transmitting text for example), i.e. if one packet is lost, it does not affect the bytes alignment of the remaining packets. Taking into consideration that the **Stream Mux** block concatenates the first input stream (with size of the first value parameter) with the second input stream (with the size of the second value parameter), and the **Vector Source** produce the same bit with an option to repeat it, it is possible do padding of a stream. Therefore, we set $440, 2$ on the **Stream Mux** and 0 on the **Vector Source** with "Repeat" option selected.

The GR supports multi-threading in the encoder process in order to improve the throughput. The "Parallelism" parameter creates multiple encoder variables with same settings, where choosing "1" it creates a list of variables with length defined in "Dimension 1" parameter, and choosing "2" it creates a list of lists of variables with length defined in "Dimension 1" and "Dimension 2". Still about this subject, the **"FEC Encoder"** block must know how to handle with these variables, thus, we can set one of the two options: "Capillary" or "Ordinary" in "Threading Type" parameter. In the first type, all the data stream is divided in N sub-streams, where each one is passed to one encoder variable to process it in a parallel way. After the processing is done, the resulted sub-stream is interleaved back together making again a single stream. The second type is similar; however, it creates a tree where each branch launches 2 more branches, thus N must be a factor of 2. This outperforms the former. With this in mind, we have set to 1 in "Parallelism" and 4 in "Dimension 1", leaving room for more variables in the decoder side (which is heavier) so the CPU can handle it.

The puncture component is basically erase some bits in a continuous stream with a pre-established pattern, thus, it is defined with: 1) a pattern ($P\_PATTERN$), which is a binary sequence, where the 0's defines which bit will be punctured, and 2): the size (P_SIZE) of the pattern (used because a pattern may start with a 0 bit). Let N be the number of 0's of a pattern and S the size of a given stream, the number of bits punctured for that stream is given by $P = \frac{S}{N*P\_SIZE}$.

As example, if we want puncture the stream $(101110111011)_2$ ($S = 12$) with the pattern $(01)_2$ ($P\_PATTERN = (1)_{10}$ with $N = 1$) of size $(2)_{10}$, will result in $P = \frac{12}{1*2} = 6$ bits punctured and the final stream will be $(010101)_2$. This to explain that the puncture component is a little tricky when applying it to the FG as there is a lot of further bit padding in order to make independent packets.

The **Puncture 64Bits** is a block implemented by us (that can be found in [19]) to enable the usage of patterns until 64 bits of size. As mentioned, we want independent packets, thus, we want the pattern's end bit to coincide with the stream's end bit, hence, the size of the stream to be punctured needs to be multiple of P_SIZE. Using the same technique previously discussed, we first pad the stream with the X value in the **Stream Mux**, which is the necessary amount of bits to add in order to make it multiple of P_SIZE. This results in a stream of $1100 + X$ bits (per packet's payload). After the puncture operation by P bits, the size of the stream will result in $1100 + X - P$ bits, which may be/not be divisible by 8.

To extract the payload from the packet, on the RX side the **Correlate Access Code - Tag Stream** block looks for a known sequence of 64 bits on a stream, then after finding it, the next 16 bits are the payload length (represented in bytes), repeated twice, resulting in 32 bits. Taking this into account, the header is composed by 12 bytes which are 64 bits of the known word plus 16 bits of the payload length that is repeated twice as shown in Fig. 4.2.

Back to the question of the stream being divisible by 8, as the payload length in the header is defined in bytes, it is impossible to represent it if it is not divisible by 8 (e.g. for 1020 bits of payload length, $1020/8 = 127.5$ bytes, which is impossible to represent it in binary in the header). Therefore, the Y variable in the **Stream Mux** is the number of bits needed to make the incoming stream divisible by 8 (for the previous example, there is need to add 4 bits, thus, $1024/8 = 128$ bytes possible to be represented in the header). The **Vector Source** outputs (i.e. produces) the binary sequence of the header (with 96 bits of size) repeatedly to be attached the payload (which has $1100 + X - P + Y$ bits of size) by using the **Stream Mux** block resulting in the packet being a continuous stream of $96 + 1100 + X - P + Y$ bits.

In Section 2.3, it was addressed Phase-Shift Keying (PSK) type of modulation with different orders. The Binary Phase-Shift Keying (BPSK) transmit only one bit for each symbol, which restrains us regarding throughput. On other side, when using an order ≥ 8, although allowing a higher throughput, the constellation points are too close, which in our case after adding the jamming signal and perform its cancellation will still result on high Bit Error Rate (BER) on Bob's side. Therefore Quadrature Phase-Shift Keying (QPSK) is chosen.

In QPSK modulation each symbol has 2 bits length. However, the stream to be modulated has only 1 relevant bit for each byte; therefore, it is used the **Repack Bits** to convert into 2 bits relevant for each byte creating the required symbol to be modulated. This symbol is modulated by using the **Chunks to Symbols** block, which takes a byte and converts it into a complex symbol taking into account the symbol table defined in the **Constellation Rect. Object** on the "Constellation Points" parameter represented in Fig. 4.3(b).

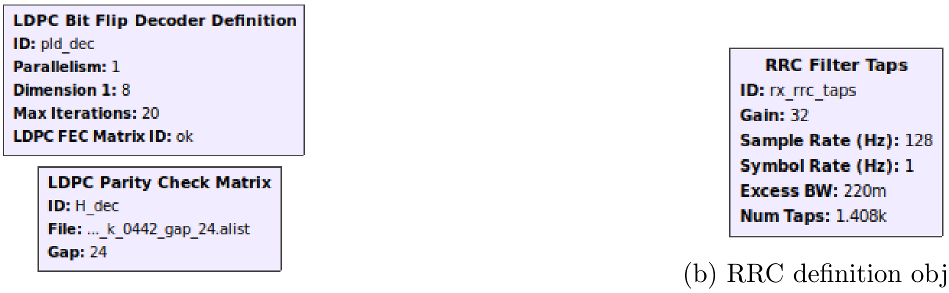
The final step before sending data through the channel, is interpolate and apply the RRC filter on the modulated symbols, both done using the **Polyphase Arbitrary Resampler** block. In this block is defined the Samples Per Symbol (SPS) to be interpolated in the "Resampling Rate" parameter (as mentioned in Section 2.3, it must be ≥ 2, however is typically used $SPS = 4$ for better time resolution, thus, this is our choice), and all the RRC parameters are defined in the "RRC Filter Taps" object block represented in Fig. 4.3(c). There is a tricky detail different from the theory mentioned in Section 2.3. In practice, instead of applying only one filter, several small size filters are used to accelerate processing, i.e. it creates multiple filters where each one has a different phase, resulting in a filterbank with N filters in order to increase resolution, hence, decrease Inter-Symbol Interference (ISI). Note that N can be as high as we desire while taking into account the processing power we have, and this is defined in the "Number of Filters" parameter (in my case $N = 32$, as recommended in [91]). On the **RRC Filter Taps** object block is defined the "Sample Rate"$=N * SPS = 32 * 1 = 32$ (with SPS before interpolation), the "Symbol Rate"$=1$, the roll-off factor on the "Excess BW" parameter $= 0.22$ (which is typically used), and finally the number of taps which is calculated as $11 * SPS * N = 11 * 4 * 32 = 1408$ (remember, 11 was discussed in Section 2.3, for being the center of the filter, plus the five close neighbors for each side).

Finally, the stream goes through a channel model block that emulates the characteristics and properties of a real channel whose distortion caused and noise added to the transmitted signal need to be handled by RX, such as:

- Noise - Additive White Gaussian Noise (AWGN) is mixed with the signal taking into account the "Noise Voltage" parameter, where it is set a level as voltage;
- Frequency Offset - The "Frequency_offset" parameter where 0 is no offset and 0.25 would be a digital modem (1/4 of the symbol rate);
- Timing Offset - The "Epsilon" parameter allows emulating different sample clocks between the transmitter e the receiver and 1.0 is no offset added;
- Multipath - The multipath delay is emulated by adding taps of a Finite Impulse Filter (FIR) filter in the "Taps" parameter.

4.1.1.2 Receiver side

In the RX side is necessary to take care of the synchronization problems as mentioned on Section 2.4 and apply match filtering (RRC filter) followed by decimation in order to



LDPC Bit Flip Decoder Definition
ID: pld_dec
Parallelism: 1
Dimension 1: 8
Max Iterations: 20
LDPC FEC Matrix ID: ok

LDPC Parity Check Matrix
ID: H_dec
File: ..._k_0442_gap_24.alist
Gap: 24

RRC Filter Taps
ID: rx_rrc_taps
Gain: 32
Sample Rate (Hz): 128
Symbol Rate (Hz): 1
Excess BW: 220m
Num Taps: 1.408k

(a) FEC decoder definition objects.

(b) RRC definition object.

Figure 4.4: Definition objects on RX side of the SCS-HK scheme.

maximize the Signal to Noise Ratio (SNR), as discussed on Section 2.3. Regarding clock synchronization, the GR has a stock block that exploits the usage of the multiple filters (N) in polyphase RRC filter implementation to outperform an independent one. The **Polyphase Clock Sync** performs:

1. Correction of the clock offset mentioned in the Section 2.4.1;
2. Application of the RRC filter as discussed on Section 2.3. The RRC filter is defined in "Taps" parameter referencing a "RRC Filter Taps" object block represented on Fig. 4.4(b);
3. Decimation from a specified SPS value on the "Samples/Symbol" parameter (4 as previously discussed) into a desired value on the "Output SPS" parameter.

Note that as the incoming sample rate is interpolated by the SPS value, then the "Sample Rate" parameter value on the "RRC Filter Taps" object block is different from the TX side, in this case it is: $N * SPS = 32 * 4 = 128$.

The multipath problem mentioned in Section 2.4.3 is taken care with an equalizer, particularly, the **CMA Equalizer** block which must have an input of 2 SPS. Therefore, the desired value to be set in the "Output SPS" parameter of the previous block must to be in accordance with the input SPS value of this (by using "Samples per Symbol" parameter). The "Gain" and the "Number of taps" are experimentally found, by visualizing the point's convergence in a constellation plot block and increase its value until the convergence is not affected anymore) and we have set 0.1 and 15, respectively. After equalization the signal outputs has 1 SPS and with more aggregated constellation points.

The final step for correcting the signal is perform fine frequency correction using the Costas loop algorithm in order to remove the rotation of the constellation points as explained in Section 2.4.2.3. This algorithm is implemented on **Costas Loop** block and has as parameter the order of the constellation (i.e. 4 for QPSK).

At this moment we got a clear, aggregated and steady constellation, being possible to correctly recover the symbols by demodulating the signal stream using the **Constellation Decoder** block. This block takes as parameter the **Constellation Rec. Object** object block represented on Fig. 4.3(b) to output the symbol in a binary form with 2 bits of relevance for each byte. Currently, the received data are continuous packets; therefore it is required to detect each header to extract the payload.

To detect the header in a continuous stream of data, it is first necessary to use the **Repack Bits** block to split them into 1 bit for each byte to allow the usage of the **Correlate Access**

Code - Tag Stream block that detects the header and extracts the payload, as already discussed. This block has the "Threshold" parameter which defines the maximum number of different bits that can be wrong on the received known word to accept it. To the accepted packet is extracted the payload considering the length information included in the header. The length defined on the header is repeated twice to prevent the extraction of a bigger/smaller payload which results on the corruption of the next packet. The length is verified, and if it does not coincide the payload is discarded and tries to find the next sequence of the known word.

If the payload is correctly extracted, it contains Y extra bits previously added to allow the byte representation on the header, thus, the **Keep M in N** block extracts only the necessary information to be depunctured.

The **Depuncture 64Bits** block performs in part the reverse operation of the puncture carried at TX. Hence, it uses the $P\_PATTERN$ to insert a bit set to 1 into the location of the bit that has been punctured, adding therefore P bits on the data stream. As before, this block was also implemented from scratch to enable the usage of patterns until 64 bits of size (found in [19]).

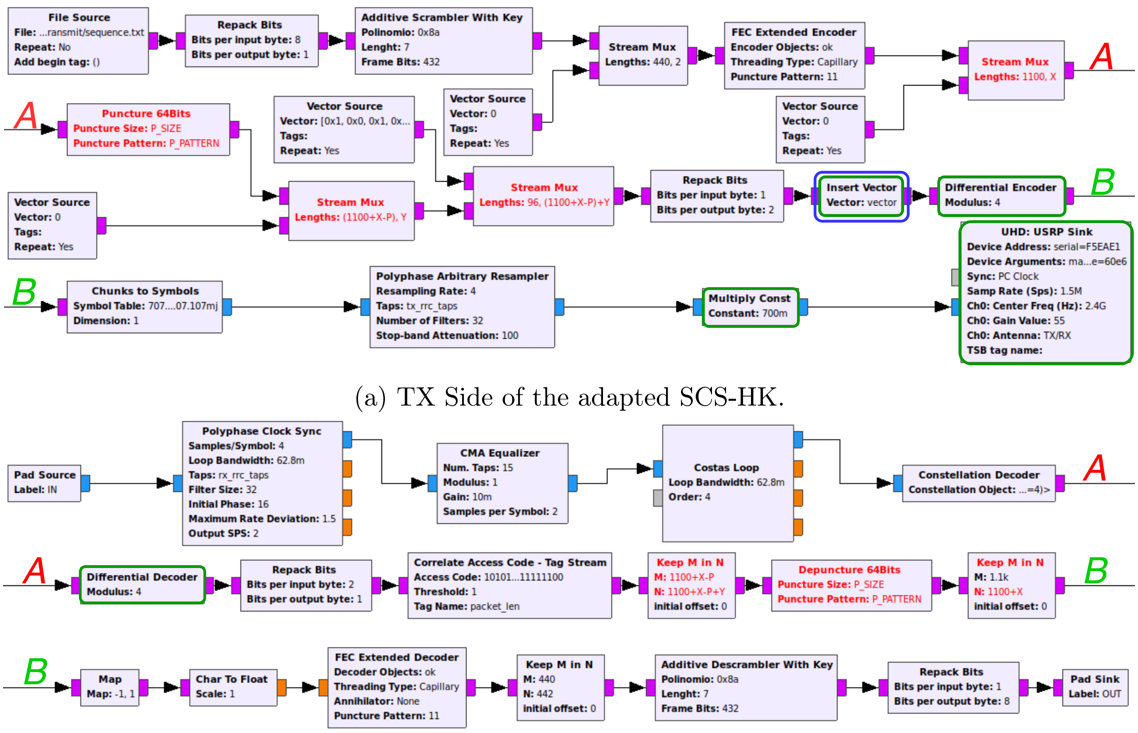
The payload is now ready to be decoded using the **FEC Extended Decoder** block. Here, it is possible to reference either the **LDPC Decoder Definition** or the **LDPC Bit Flip Decoder Definition**. The first uses a belief propagation decoding (i.e. it is a soft-decision decoder) with σ^2 noise variance of the AWGN channel. The latter is a hard decision decoding scheme which uses bit-flipping decoding, where the decoder seeks to find the codeword that was most likely sent. In this implementation it was chosen the **LDPC Bit Flip Decoder Definition** object block for enabling us to use the **Differential Decoder** (which is not possible if using the other one). Also, it is required to use the same H matrix as the TX side, and we have to set the maximum number of iterations of the decoding process, as shown in Fig. 4.4(a). This operation decodes the original scrambled information of 442 bits, of which only 440 are used.

Finally, for each decoded payload the data is descrambled in the **Additive Descrambler With Key** block, using as key the first 8 bits of the decoded word output from the LDPC decoder, and the remaining ones (432 bits) as the input information on descrambler's LFSR in order to recover the original information. Note that all this block was implemented in the scope of this thesis and then integrated on the Out-of-Tree (OoT) blocks, found in [19]. In the same way as before, on descrambler the last byte is not output because it stays on the registers. To fix this problem it is required to input one more byte on the LFSR to flush the interesting byte (so the last byte of the frame is output). The descrambled information is repacked again into a byte and flushed into a file.

4.1.2 Real-World Prototype

When moving to a real wireless transmission it is required to have one FG for each side of the communication in order to separate the sample processing. As mentioned, the **Throttle** block is no longer needed, as there are new blocks to control the TX and RX sample rate. Also, the channel will be the air, thus, the **Channel Model** block will not be used either.

The TX and RX FGs can be visualized in Fig. 4.5(a) and in Fig. 4.5(c), respectively. The latter uses a **Decoding** hierarchical block represented on Fig. 4.5(b). Note that the added blocks necessary to achieve a wireless transmission when compared with the simulation are marked in green and the blocks implemented in the scope of this thesis are marked in blue.



(a) TX Side of the adapted SCS-HK.

(b) Decoding hierarchical block used on reception.

(c) RX Side of the adapted SCS-HK.

Figure 4.5: The adapted SCS-HK divided in TX side 4.5(a) and in the RX side on 4.5(c). The last uses the Decoding block represented on 4.5(b).

4.1.2.1 Transmitter side

As previously described, the TX's FG encodes and modulates a data stream using the SCS-HK scheme. When compared with the FG of the TX's side using wireless, some blocks are required to be added, such as **Insert Vector**, **Differential Encoder**, **Multiply Const** and **USRP Sink**. Others are removed, such as **Throttle** and **Channel Model**. After assessing the main differences between the simulated and real-world scenarios, we now describe the main challenges faced and how they were overcome.

It is important to mention that the FG does not implement coarse frequency correction due to the closeness of the USRP's; therefore, having a small frequency offset. However, if there is need to implement it, the block **FLL Band-Edge** connected to the **Polyphase Clock Sync** corrects it.

A problem that we came across was that, after starting the transmission only after a certain amount of packets the information was correctly decoded, i.e. the first packets are not written on the file. Following what was explored in Section 2.4.2.3, the Costas loop algorithm needs some initial time to lock the correct frequency of the signal to correct it. Therefore, as long as the Costas Loop algorithm has not blocked, the signal was not correctly demodulated and nothing was written on the file losing all these packets. In order to solve this problem, the **Insert Vector** block was implemented and integrated on OoT blocks (found in [19]). This block takes a vector of integers as parameter and sends them before sending the actual data (only at the beginning of the communication). The length of the vector is experimentally found in such way that allows to the Costas Loop algorithm lock the correct frequency. This way, the receptor first receives "junk" added by us in order to make Costas loop lock in the right frequency, after that, the remaining transmission works correctly. Note that the "junk" vector inserted needs to have 2 bits for each byte in order to further modulate it, and this "junk" is not written on the file because it has no header associated with it.

Another interesting problem was that looking to the data written on the file, sometimes it was written, other times it was not. This is the phase ambiguity problem due to the presence of phase distortion in the communication channel with four possible constellation orientations (as it is QPSK modulation) as mentioned in Section 2.4.2.4. This is taken care by using the differential encoding where instead of transmitting the symbol, it transmits the difference between continuous symbols using the present and the previous one. This can be done by using the **Differential Encoder** block and set 4 as the modulus (QPSK's alphabet has 4 symbols). As mentioned in Section 2.4.2.4, it is required to use natural mapping represented in Fig. 2.3(a); therefore in the **Constellation Rect. Object** (Fig. 4.3(b)) is set $0, 1, 2, 3$ on the "Symbol Map" parameter and $0.707+0.707j, -0.707+0.707j, -0.707-0.707j, 0.707-0.707j$, on the "Constellation Points" parameter.

When we move from the simulation to real-world, the problem called Saturation Limits is presented. This occurs because the **USRP Sink** block must send float values in $[-1, 1]$ range, otherwise, the signal becomes saturated and nonlinear, which we do not desire. Using the **Time Sink GUI** block before the connection to the **USRP Sink** block, it is possible to analyze the values input into the USRP. The goal is found the best parameter value in the **Multiply Const** block to adjust these values to be as high as possible considering the range mentioned ($[-1, 1]$). The value to be multiplied is experimentally found by setting it at 0 and successively increased until achieve the goal mentioned. Using this process we set the value to 0.7 in the **Multiply Const** block parameter.

To maximize the throughput, it is required to adjust the sample rate in such way that can be the maximum as possible without causing any problems associated to Underrun or Overflow (these problems prints on GR console U's and O's, respectively). The Underrun problem occurs when the TX side is not providing samples fast enough to the **USRP Sink** block to send them. This can be caused by lack of processing power (note that the encoder block requires a lot of power). Another problem that may happen is Overrun, i.e. in RX side we are not able to consume samples quickly enough, probably because of the back-pressure caused by a computationally heavier block. Both problems may be solved with two solutions: 1) Or the processing power of the machine is changed; or 2) It is mandatory to decrease the sample rate of both FG (TX and RX sides). Taking this into account, the goal is to find the maximum value for the sample rate, where this problems may occur only on the beginning of the transmission (which is normal when the FG is initializing).

Another point to consider is setting the sample raste is the Clock Rate value in the USRP that feeds the radio frequency frontends and the Digital Signal Processors (DSP) chains. This value must be set in [5 MHz, 61.44 MHz] range on the Master Clock Rate (MCR) argument of the respective USRP block. The sample rate value must be chosen taking also into consideration that $\frac{ClockRate}{SampleRate}$ results in an integer value and preferably divisible by 4 (for better performance) as mentioned in [92]. In a transmission with no puncturing (i.e. $P\_SIZE = 2; P\_PATTERN = 3; X = 4; Y = 0$ and $P = 0$), the highest sample rate achieved without occurring any problem (only at the beginning as normal) was 1.5 Mhz. Taking these constraints into consideration, we choose 60 MHz for clock rate and 1.5 MHz for sample rate, thus, $\frac{ClockRate}{SampleRate} = \frac{60}{1.5} = 40$, which is integer and divisible by 4.

Another advantage for adding "junk" values in the **Insert Vector** block previously discussed is exactly these problems may occur only at the transmission initialization, while the "junk" data is been transmitted and, therefore, not losing any important information data.

To send this stream to the USRP it is used the **USRP Sink** block got from the USRP Hardware Driver (UHD) with the parameters mentioned before.

4.1.2.2 Receiver Side

The Saturation Limits issue mentioned above was solved only on the TX side, however, it has not been solved for the RX side yet. In order to handle it, it is necessary play around in the TX/RX gain of the **USRP Sink/Source** blocks respectively, until finding reception values inside of $[-1, 1]$ range. If the positions of the USRP's antennas are not changed, the chosen gain values stay the same, otherwise, it is necessary adjust them again. One way to work around the constant manual adjustment of the gain is to change them dynamically by using an automatic gain controller. We did not implement this block because: 1) increases the computational complexity of the FG and; 2) to be in exact control of the transmission values and parameters to get more realistic and aggregated results.

Finally, as in the TX the **Differential Encoder** block was added in order to handle the phase ambiguity problem, therefore it is necessary to implement on the RX side the reverse operation, with this being carried by the **Differential Decoder** block after the **Constellation Decoder** retrieving the information symbol as mentioned on Section 2.4.2.4.

4.2 Jamming and Self-Interference Cancellation Implementation

The idea to implement is presented in Fig. 3.13 and it is: Bob generates noise to be mixed with the transmitted signal in such way that Eve cannot recover the signal of interest, while Bob, using as an advantage the knowledge of the generated noise, can estimate the transmitted signal. As before, it was be implemented first in simulation to analyze the main challenges and then we have moved into the real-world prototype.

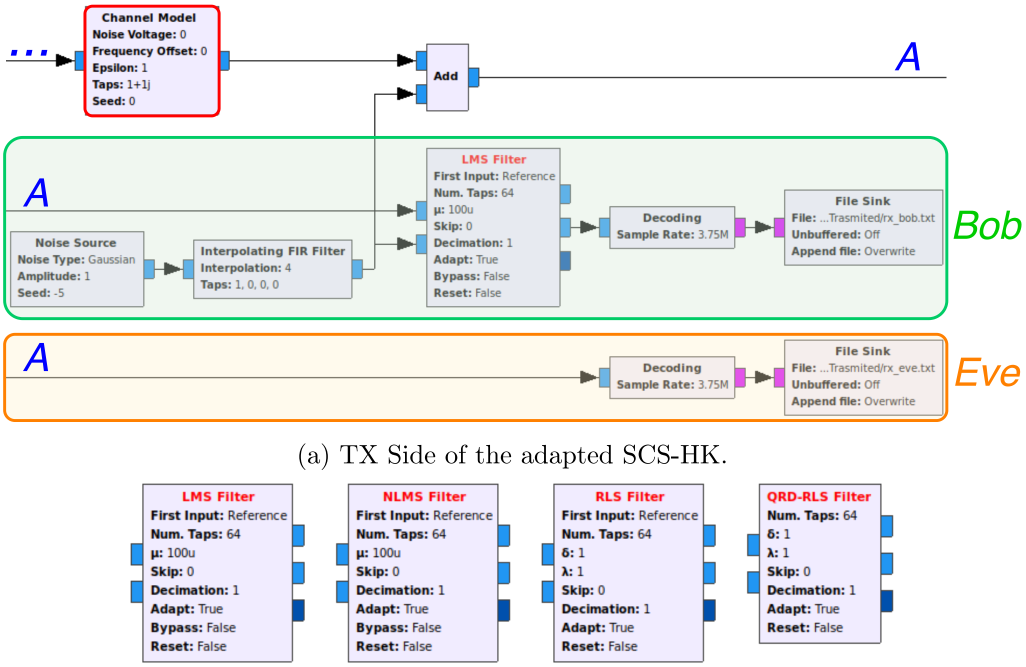
4.2.1 Simulation

The implementation of the jamming and its cancellation is only done on top of the RX's side, thus, the TX's FG is the same as before that was presented in Fig. 4.1(a) and discussed in Section 4.1.1.1. Regarding the RX's side, since it is a simulation both Bob and Eve are included in the same FG represented on Fig. 4.6(a) as green and orange, respectively. The block marked at red is removed when going to real-world wireless transmission.

The **Noise Source** is a GR stock block that generates a noise signal with a Gaussian distribution (defined in "Noise Type" parameter).

Considering the SI cancellation algorithms to be used i.e., Least Mean Squares (LMS), Normalized Least Mean Squares (NLMS), Recursive Least Squares (RLS) and QR Decomposition Recursive Least Squares (QRD-RLS) (mentioned in Section 3.6.2), in the FG of Fig. 4.6(b) can be found the GR blocks used to implement these techniques and which can be found in [93]. The only modification necessary to switch between algorithms is changing the respective block.

For better explanation, first let us link the notation defined in Section 3.6.2 with the



Channel Model
Noise Voltage: 0
Frequency Offset: 0
Epsilon: 1
Taps: 1+1j
Seed: 0

Add

LMS Filter
First Input: Reference
Num. Taps: 64
µ: 100u
Skip: 0
Decimation: 1
Adapt: True
Bypass: False
Reset: False

Noise Source
Noise Type: Gaussian
Amplitude: 1
Seed: -5

Interpolating FIR Filter
Interpolation: 4
Taps: 1, 0, 0, 0

Decoding
Sample Rate: 3.75M

File Sink
File: ...Trasmited/rx_bob.txt
Unbuffered: Off
Append file: Overwrite

Decoding
Sample Rate: 3.75M

File Sink
File: ...Trasmited/rx_eve.txt
Unbuffered: Off
Append file: Overwrite

(a) TX Side of the adapted SCS-HK.

LMS Filter
First Input: Reference
Num. Taps: 64
µ: 100u
Skip: 0
Decimation: 1
Adapt: True
Bypass: False
Reset: False

NLMS Filter
First Input: Reference
Num. Taps: 64
µ: 100u
Skip: 0
Decimation: 1
Adapt: True
Bypass: False
Reset: False

RLS Filter
First Input: Reference
Num. Taps: 64
δ: 1
λ: 1
Skip: 0
Decimation: 1
Adapt: True
Reset: False

QRD-RLS Filter
Num. Taps: 64
δ: 1
λ: 1
Skip: 0
Decimation: 1
Adapt: True
Reset: False

(b) Blocks of the used algorithms.

Figure 4.6: Receiver simulation FG on 4.6(a) with SI cancellation implemented by using one of the blocks of 4.6(b). Bob's side is separated from Eve's side, at green and red, respectively.

adaptive filter blocks. Taking as example the LMS filter and following Fig.4.7 it is easily explained how the transmitted signal is recovered. The goal is adapt the filter $\widehat{h}(n)$ to make it as close as possible of an unknown system $h(n)$, i.e. estimate $\widehat{y}(n)$ from $x(n)$ by minimizing the error $e(n)$ between $d(n)$ and $\widehat{y}(n)$. In the **LMS Filter** block, the first input is the reference signal $d(n)$, the second input is $x(n)$ and then after the filter is applied the first output is $\widehat{y}(n)$, and the second is the error $e(n)$. Let's denote S as the signal transmitted by Alice (the signal that we desire to recover), N_c the AWGN that is added by the channel and N_j the Gaussian noise generated by Bob. Therefore, the receiving signal from Bob's antenna is the reference signal, which is composed by $d(n) = \overline{N}_j + S + N_c$, where $\overline{N}_j$ is N_j modified by the channel and the error is the difference between the estimated noise by the filter $\widehat{y}(n) = \widehat{N}_j$ and the reference signal $d(n) = \overline{N}_j + N_c + S$, i.e. $e(n) = \widehat{y}(n) - d(n) = \ddot{N}_j + N_c + S$, where $\ddot{N}_j$ is the remaining noise that the filter was not able to cancel.

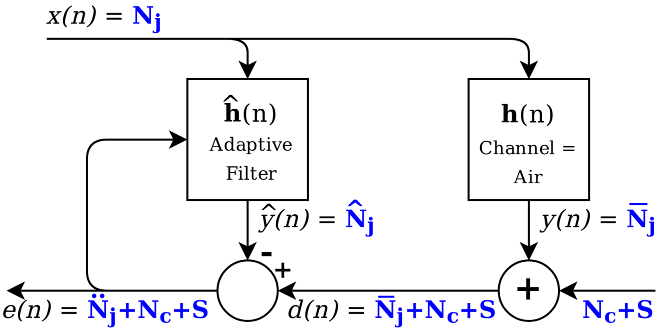
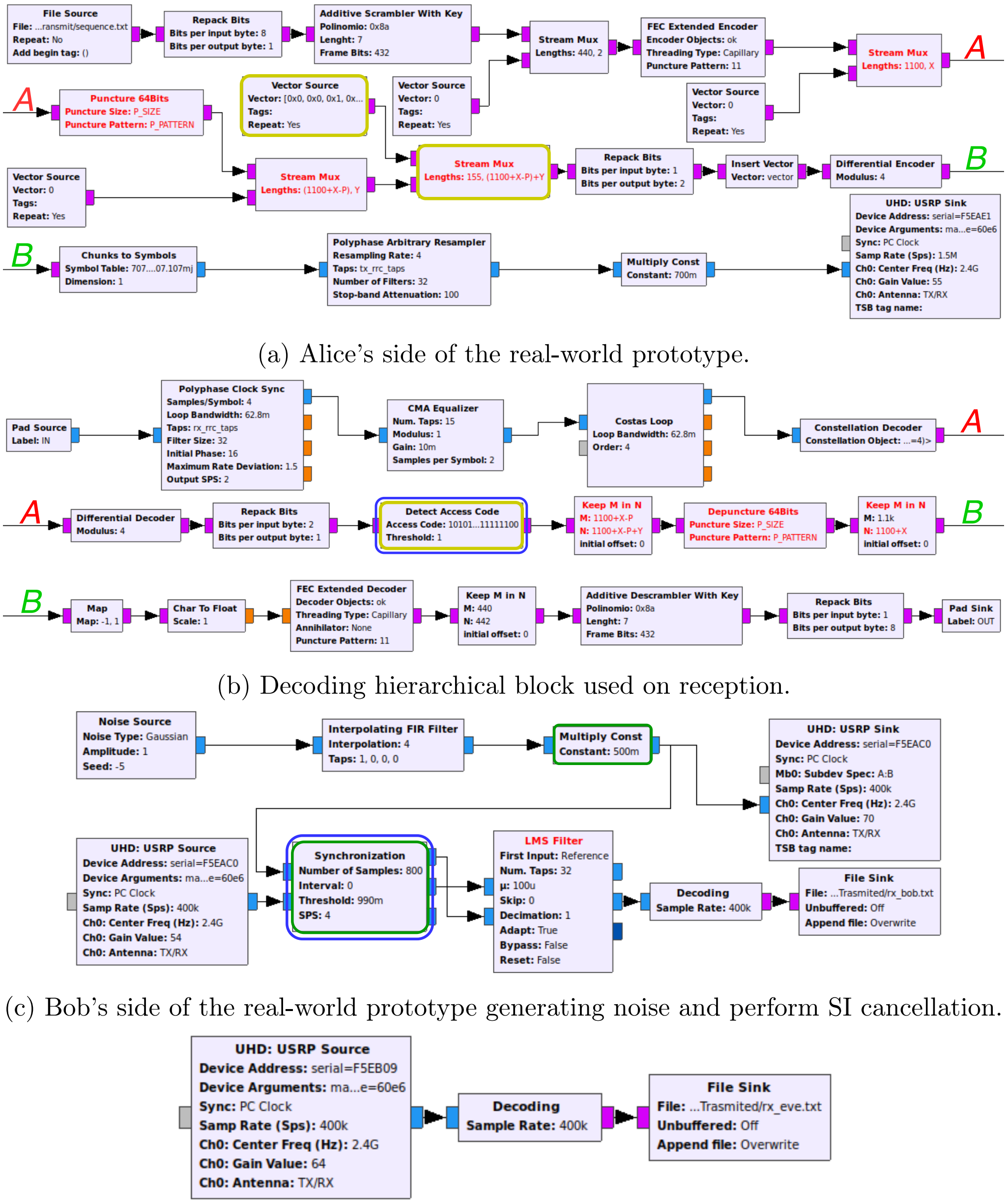

x(n) = $\mathbf{N_j}$
$\hat{\mathbf{h}}$(n)
Adaptive
Filter
h(n)
Channel =
Air
$\hat{y}$(n) = $\hat{\mathbf{N}}_\mathbf{j}$
y(n) = $\overline{\mathbf{N}}_\mathbf{j}$
-
+
+
e(n) = $\ddot{\mathbf{N}}_\mathbf{j}$+$\mathbf{N_c}$+$\mathbf{S}$
d(n) = $\overline{\mathbf{N}}_\mathbf{j}$+$\mathbf{N_c}$+$\mathbf{S}$
$\mathbf{N_c}$+$\mathbf{S}$

Figure 4.7: Diagram of the Alice's signal recovery using the adaptive filter on Bob's side.

The first problem faced is that the GR suddenly stops the data flow. This is caused because both streams must have the same rate at the LMS inputs, however, the signal originated on TX side comes interpolated by SPS= 4 and the signal generated by the **Noise Source** block only has 1 SPS. Therefore, the block **Interpolating FIR Filter** is used in order to interpolate the noise signal by 4 resulting in the same rate for both streams.

4.2.2 Real-World Prototype



File Source
File: ...ransmit/sequence.txt
Repeat: No
Add begin tag: ()

Repack Bits
Bits per input byte: 8
Bits per output byte: 1

Additive Scrambler With Key
Polinomio: 0x8a
Lenght: 7
Frame Bits: 432

Stream Mux
Lengths: 440, 2

FEC Extended Encoder
Encoder Objects: ok
Threading Type: Capillary
Puncture Pattern: 11

Stream Mux
Lengths: 1100, X

Puncture 64Bits
Puncture Size: P_SIZE
Puncture Pattern: P_PATTERN

Vector Source
Vector: [0x0, 0x0, 0x1, 0x...
Tags:
Repeat: Yes

Vector Source
Vector: 0
Tags:
Repeat: Yes

Vector Source
Vector: 0
Tags:
Repeat: Yes

Vector Source
Vector: 0
Tags:
Repeat: Yes

Stream Mux
Lengths: (1100+X-P), Y

Stream Mux
Lengths: 155, (1100+X-P)+Y

Repack Bits
Bits per input byte: 1
Bits per output byte: 2

Insert Vector
Vector: vector

Differential Encoder
Modulus: 4

Chunks to Symbols
Symbol Table: 707....07.107mj
Dimension: 1

Polyphase Arbitrary Resampler
Resampling Rate: 4
Taps: tx_rrc_taps
Number of Filters: 32
Stop-band Attenuation: 100

Multiply Const
Constant: 700m

UHD: USRP Sink
Device Address: serial=F5EAE1
Device Arguments: ma...e=60e6
Sync: PC Clock
Samp Rate (Sps): 1.5M
Ch0: Center Freq (Hz): 2.4G
Ch0: Gain Value: 55
Ch0: Antenna: TX/RX
TSB tag name:

(a) Alice's side of the real-world prototype.

Pad Source
Label: IN

Polyphase Clock Sync
Samples/Symbol: 4
Loop Bandwidth: 62.8m
Taps: rx_rrc_taps
Filter Size: 32
Initial Phase: 16
Maximum Rate Deviation: 1.5
Output SPS: 2

CMA Equalizer
Num. Taps: 15
Modulus: 1
Gain: 10m
Samples per Symbol: 2

Costas Loop
Loop Bandwidth: 62.8m
Order: 4

Constellation Decoder
Constellation Object: ...=4)>

Differential Decoder
Modulus: 4

Repack Bits
Bits per input byte: 2
Bits per output byte: 1

Detect Access Code
Access Code: 10101...11111100
Threshold: 1

Keep M in N
M: 1100+X-P
N: 1100+X-P+Y
initial offset: 0

Depuncture 64Bits
Puncture Size: P_SIZE
Puncture Pattern: P_PATTERN

Keep M in N
M: 1.1k
N: 1100+X
initial offset: 0

Map
Map: -1, 1

Char To Float
Scale: 1

FEC Extended Decoder
Decoder Objects: ok
Threading Type: Capillary
Annihilator: None
Puncture Pattern: 11

Keep M in N
M: 440
N: 442
initial offset: 0

Additive Descrambler With Key
Polinomio: 0x8a
Lenght: 7
Frame Bits: 432

Repack Bits
Bits per input byte: 1
Bits per output byte: 8

Pad Sink
Label: OUT

(b) Decoding hierarchical block used on reception.

Noise Source
Noise Type: Gaussian
Amplitude: 1
Seed: -5

Interpolating FIR Filter
Interpolation: 4
Taps: 1, 0, 0, 0

Multiply Const
Constant: 500m

UHD: USRP Sink
Device Address: serial=F5EAC0
Sync: PC Clock
Mb0: Subdev Spec: A:B
Samp Rate (Sps): 400k
Ch0: Center Freq (Hz): 2.4G
Ch0: Gain Value: 70
Ch0: Antenna: TX/RX
TSB tag name:

UHD: USRP Source
Device Address: serial=F5EAC0
Device Arguments: ma...e=60e6
Sync: PC Clock
Samp Rate (Sps): 400k
Ch0: Center Freq (Hz): 2.4G
Ch0: Gain Value: 54
Ch0: Antenna: TX/RX

Synchronization
Number of Samples: 800
Interval: 0
Threshold: 990m
SPS: 4

LMS Filter
First Input: Reference
Num. Taps: 32
µ: 100u
Skip: 0
Decimation: 1
Adapt: True
Bypass: False
Reset: False

Decoding
Sample Rate: 400k

File Sink
File: ...Trasmited/rx_bob.txt
Unbuffered: Off
Append file: Overwrite

(c) Bob's side of the real-world prototype generating noise and perform SI cancellation.

UHD: USRP Source
Device Address: serial=F5EB09
Device Arguments: ma...e=60e6
Sync: PC Clock
Samp Rate (Sps): 400k
Ch0: Center Freq (Hz): 2.4G
Ch0: Gain Value: 64
Ch0: Antenna: TX/RX

Decoding
Sample Rate: 400k

File Sink
File: ...Trasmited/rx_eve.txt
Unbuffered: Off
Append file: Overwrite

(d) Eve's side of the real-world prototype.

Figure 4.8: The final FG of Alice's side represented on 4.8(a), the Bob's side on 4.8(c), the Eve's side on 4.8(d). Both using the Decoding block on 4.8(b).

In Fig. 4.8 are represented the final FGs for the real-world testbed, i.e. the implementation of noise generation and SI cancellation by the receiver along with the SCS-HK scheme for a real-world wireless transmission. The modified blocks to improve packet detection on Fig. 4.8(a) and Fig. 4.8(b) are marked in yellow, and in Fig. 4.8(c) the blocks marked in green were added when moving from simulation to wireless. The blocks marked in blue are implemented in the scope of this thesis. Also, the structure of the transmitted packets is represented on Fig. 4.9 with the respective component's length.

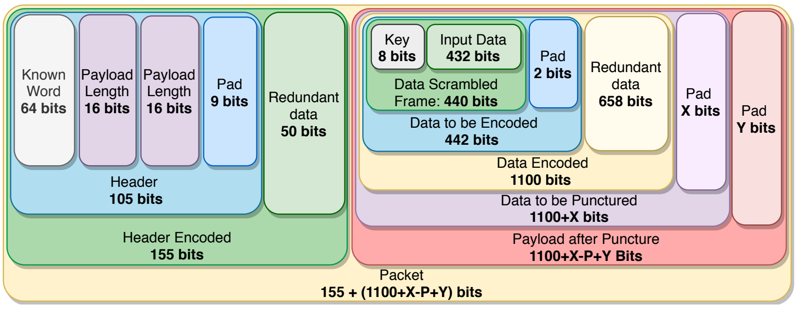


Figure 4.9: Representation of the final packet's components and the respective lengths for the Real-World prototype when combining the SCS-HK scheme and Self-Interference Cancellation.

These FGs were based in the simulation ones presented in Section 4.1.2.1, with some improvements and additional blocks that allow a wireless transmission. Therefore, next we will address specifically the description of these improved/added blocks only.

The first problem to solve was the Saturation Limits already discussed in Section 4.1.2.1; by using the method described in that section it was found the 0.5 value to the **Multiply Const** block.

The biggest challenge we faced was that the noise sample that is mixed with the signal to be input in the second entry of the **LMS Filter** block must be in accordance with its respective noise sample, i.e. both inputs of the adaptive filter need to be synchronized (regarding to the noise samples) in order to work properly. The problem is created due to the "path" through **USRP Sink** block, air and **USRP Source** block takes more time for a sample to travel compared with the direct path on the FG. The solution found was to create a block that, delays the noise stream to synchronize it with the receiving stream. Therefore, the block selects the first X noise samples and performs a correlation operation ([94]) against the mixed signal to find the value needed to delay the noise stream. The block works with a sliding window on top of the mixed stream with the size of X and performing a correlation operation on that window. If the result of the correlation is greater than a desired threshold, the delay is found, and the noise stream is delayed by the found value to align with the other stream. Otherwise, the window will slide and perform the correlation operation again, and so on.

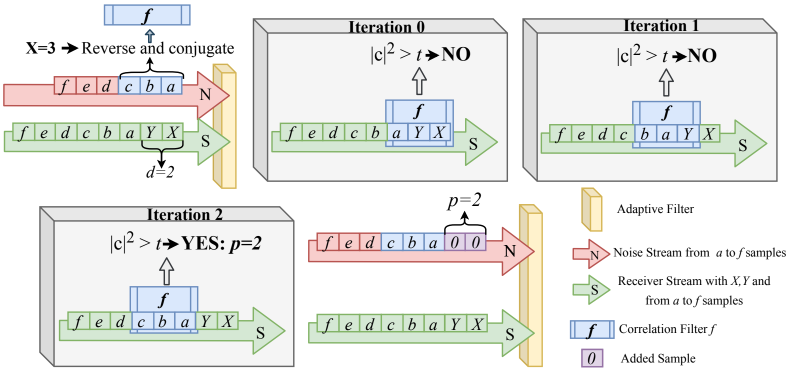


Figure 4.10: Example diagram to represent the operation of Synchronization block

Explaining the implementation in a more detailed way, let S be the receiver stream (of length S_s) containing noise mixed with the transmitted signal, N the noise stream and consider a correlation window of X samples and set threshold to t. The **Synchronization** block: 1) Selects the samples $N[1:X]$ to be perform the reverse of the conjugate in order to create the correlation filter f; 2) Applies f against a sliding window of size X on top of S (i.e. filter $S[p+1:p+X]$ with f where $p=[0,1,...,S_s-X]$). This results in a value c, for which its square magnitude $|c|^2$ is calculated. 3) If $|c|^2 \geq t$, the correlation is found with delay being indicated by p, otherwise, the sliding operation continues. 4) The block outputs the noise with a delay d (using a circular vector) and the stream N input goes directly to the output. An example with N=3 and d=2 is depicted in Fig. 4.10. Note that there is not a GR stock block to perform this procedure, thus, it was implemented by us and it is found in [19].

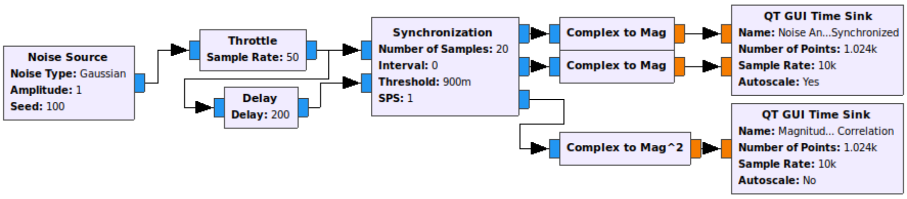
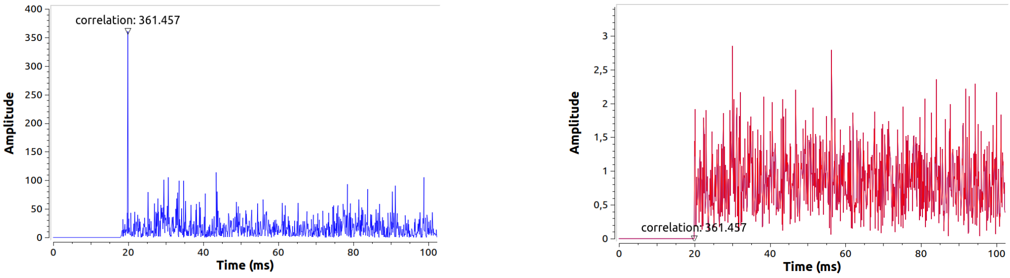


Figure 4.11: FG to evaluate Synchronization block.



(a) Correlation values output from the Synchronization block.

(b) Stream outputs of Synchronization block, where both streams are synchronized.

Figure 4.12: Running the FG on Fig. 4.11 to evaluate the Synchronization block, it outputs the correlation values on 4.12(a), and synchronizes both streams 4.12(b).

The Fig. 4.11 represents a FG example in simulation mode which has a **Delay** block applied on the second stream of data to evaluate the work of this block. The block implemented, correlates the first 20 samples (defined in first parameter) of the first input, with the second input (which has initially 400 samples that are 0's because of the delay). After the operation of the block, the first output is the first input delayed by 400 samples, the second output is the second input unchanged (both being visualized on Fig. 4.12(b)) and the third output presents the correlation values, as visualized in Fig. 4.12(a) (the correlation peak means that the correlation was found; therefore, synchronizing both streams).

After the TX and RX sides were working, because of the additional noise in the channel due to the jamming generated by Bob, we realized that even with BER on the payload being low (because some errors are corrected by the LDPC code) there were several packets lost during transmission. This was due to errors added in the header during transmission, so the known word was not caught and the payload was not extracted. The workaround founded was the encoding of the header with a fast error correcting code, thus, allowing on reception the correction of some errors of the known word introduced during transmission. Therefore, the **Correlate Access Code - Tag Stream** was replaced by the **Detect Access Code** block, as shown in Fig. 4.8(b). This block constantly decodes the input data in order to check if the known word is present and, if so, only the defined number of bits are output. This block was implemented in the scope of this thesis and is found in [19].

In the implementation of this block we made use of the Reed-Solomon (RS) code, as discussed in Section 2.5.3, particularly, the freely distributed source code found in [95] with some adaptations. Making use of the notation defined in Section 2.5.3, the goal was to find the best set of parameters that could comply with the requirement of encoding the 96 bits of the known word. If we chose $m = 4$ bits for each symbol, the codeword would result in $n = 2^4 - 1 = 15$ symbols, i.e. $15 * 4 = 60$ bits. As the codeword cannot be smaller than the information to be encoded, the $m = 4$ is excluded. With $m = 5$, it results in a codeword of $n = 31$ symbols, i.e. 155 bits leaving room for at most $155 - 96 = 59$ redundant bits. The k and t values needs to be in accordance with equation $k = n - 2t$. The k value selected is as low as possible without losing any bit of the sequence to be encoded, thus, if $k = 19$ and $t = 6$ the RS code will encode $19 * 5 = 95$ bits which is lower than the sequence's size that is to be encoded. If $k = 21$ and $t = 5$ are selected, the RS code will encode $21 * 5 = 105$ bits which fits in the presented situation (> 96 bits). After selecting the parameters, note that the sequence to be encoded has 105 and our sequence has 96 bits, hence, it is purposefully padded by adding $105 - 96 = 9$ bits at 0 to the preamble to be encoded.

As discussed, the header was directly attached to the payload in a binary form using the **Vector Source** block. With its codification, there is required to input the codeword in the block to allow the receiver to decode it. Additionally, by varying the number of bits to be punctured, the payload's length needs to be adjusted. To overcome this manual adjustment, we have created a script (also found in [19]), where the input is the payload's length (in bits) and outputs the codeword's binary sequence to be input on the **Vector Source** of the present FG.

Finally, the last change is the new sample rate value depending on the algorithm used and the conditions of the transmission (jamming power, LDPC iterations, adaptive filter parameters, etc.). Therefore, this value is as maximum as the host machine can handle, and it was found with the method previously discussed in Section 4.1.2.1.

4.3 Final Thoughts

This Chapter covered the implementation of: 1) the adapted SCS-HK scheme discussed in Section 3.3.1 and; 2) noise generation by the receiver along with its cancellation and synchronization in Section 3.6.

In the first, several concepts discussed in Chapter 2 were used to achieve a basic transmission considering several steps to correctly modulate, apply the RRC filter and correct the interference added by the channel. For this, we resorted in the already implemented blocks in GR and a new block we developed to allow better synchronization called **Multiplicative Scrambler/Multiplicative Descrambler** which solves the problem of whitening sequences (e.g. when transmitting video) and packetizes the data by resetting the LFSR seed. The most challenging work was done here, not by creating the OoT block, but by getting them all to work together in a wireless scenario without any prior experience in electronics, in particular, communications. Consequently, it was required to understand all the steps of a communication system and solve several faced problems, such as phase ambiguity, saturation limits, whitening sequences, the block's parameters, among others. The resulting FG of this work is not represented in this thesis; however a tutorial was written to help any beginner in this field to successfully develop a wireless transmission explained in a detailed way each block required and how to create OoT blocks.

Thereafter, in Section 3.3.1 we addressed the operation of the SCS-HK scheme, which required developing specific blocks to add on top of the normal transmission. The new **Additive Scrambler With Key**/**Additive Descrambler With Key** blocks were developed to replace the old multiplicative scrambler (without key), and the **Puncture 64Bits**/**Depuncture 64Bits** blocks were adapted from the GR stock blocks (where only 32 bits of puncture pattern were allowed). Specifically, in the `C++` implementation, all OoT blocks require a special attention regarding the consumption/production of samples considering the GR's internal buffers, i.e. in short, we could only produce the same number of samples as the length of the buffer *noutput_ items*, and the remaining samples to be sent should be stored and output in the next iteration (perform the back-pressure characteristic of GR). In particular, the multiplicative scrambler/descrambler blocks were the most challenging as there were a few details in the implementation to be considered, namely, the disposal of the first bits for each frame when scrambling, and the flush of the LFSR for each frame of data when descrambling.

For generation of noise and self-cancellation, we used in the FG adaptive algorithms found in [93]. The challenge was mainly to adapt them to our FG so as to correctly link the data streams to the input/output of the blocks and set the correct parameters. The development of the **Synchronization** block was crucial to enable a wireless communication. This block implements kernel/libraries functions of the GR, such as: *std::reverse* and *std::conj* to perform the reverse and conjugate of the desired samples to correlate with; the *kernel::fft_filter_ ccc* to create the correlation filter f; *volk::volk_ 32fc _magnitude_ squared_ 32f* to perform the magnitude squared of the correlation result; and the *gr::buffer_ sptr* to implement the circular buffer to direct the samples from input to the output with the found delay. The remaining code flow (pick the first X samples to correlate, detect the correlation considering the threshold, delay the stream, manage the circular buffer for writing and reading, etc.) was developed considering the consumption/production of samples discussed above. Finally, the header detection performed by the **Detect Access Code** block uses the RS freely distributed source code algorithm found in [95]; therefore the challenge was to link the *C* file to the OoT block, and create all the operations of constantly decoding the stream to extract the payload when the header is detected. Also, several changes were necessary in the RS source code to adapt it considering the discussed parameters in order to output the correct information to be interpreted by the block. Note that a script to create the binary sequence of the header considering the payload's length was also developed in *C*.

Concluding, the main challenge of the implementation component of this thesis was, not only the development of several OoT blocks, but also (with more time spent) understanding the right combination of the blocks (and its parameters) and formulation of the noise canceling problem on the adaptive filter blocks to correctly set the options and link the streams for each input/output.

This page is intentionally left blank.

Chapter 5

Results

The present chapter contains the evaluation and discussion of the experimental results of the real-world prototype implementation previously presented in Section 4. For this, firstly three setup configurations are described along with used hardware and Software Defined Radio (SDR) main settings. Then the most suitable metrics presented in Section 3.2 will be chosen, along with its explanation. After establishing the statistical analysis procedures, the results are presented and discussed in Section 5.3.

In Section 5.3 will be addressed first the influence of passive suppression in Self-Interference (SI) cancellation mentioned on Section 3.6.1, and the effect of header codification, described in Section 4.2.2. Then, SI cancellation algorithms will be evaluated for several jamming powers in the most fair scenario possible, as discussed on Section 3.6.2. Thereafter, the chosen adaptive algorithm will be used to evaluate the three setups described to understand the SI cancellation capability when changing SDRs' positions. Finally, using the best convergence algorithm the impact of implementing the Scrambled Coding for Secrecy with a Hidden Key (SCS-HK) scheme with different number of bits punctured will be evaluated, as explored in Section 4.2.2.

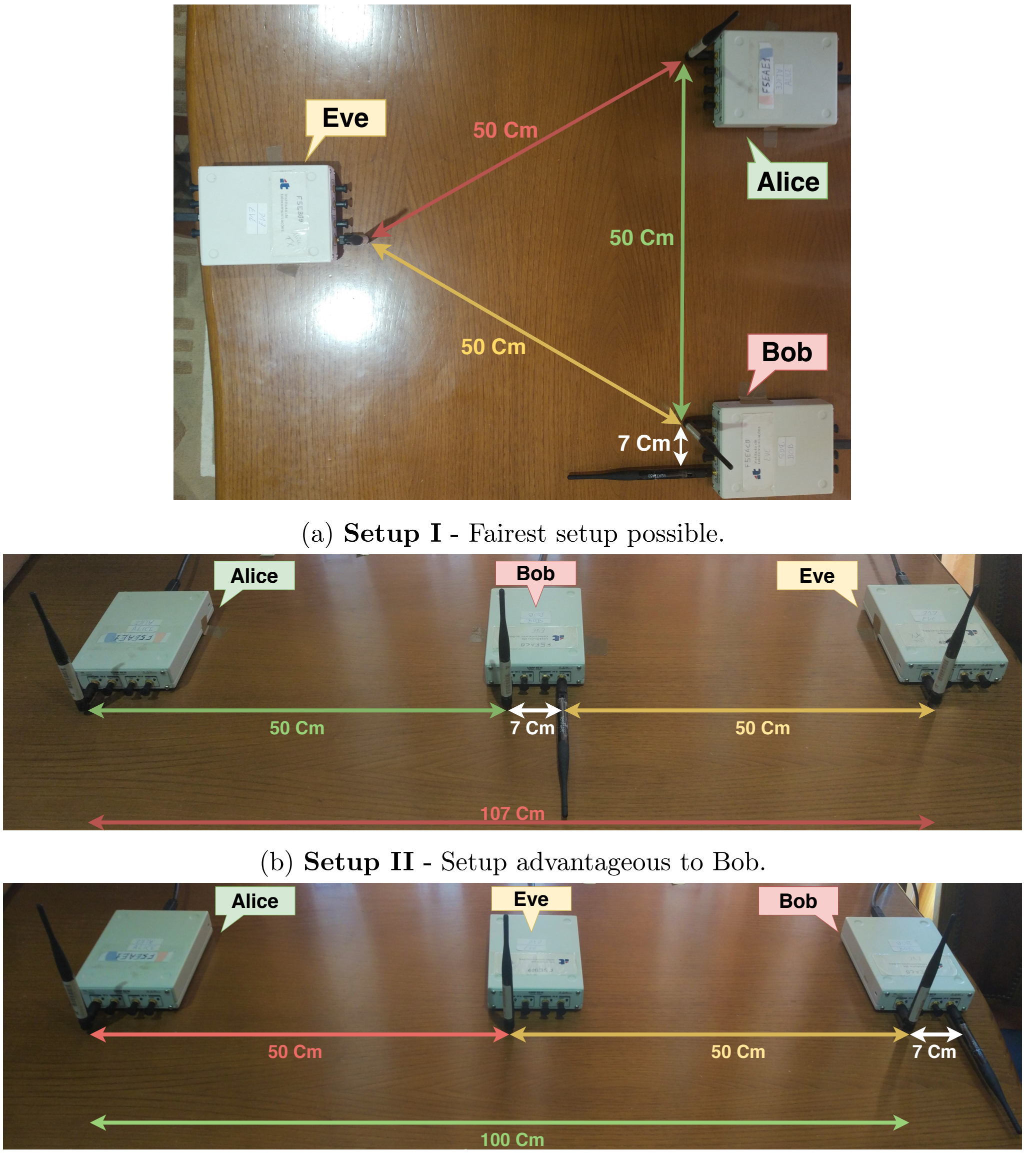
5.1 System Setup

5.1.1 Setup and Practical Metrics

Three different configurations were created to evaluate the SI cancellation capability of the adaptive algorithm to be chosen and the secrecy of information achieved. Each setup is chosen considering the pertinent real-world possibilities to compromise the security of this scheme, and these setups will be linked to the contactless example given in Section 1.1 (with Bob acting as a payment terminal, Alice as the person who wants to pay and Eve as the thief).

The first setup (**Setup I**) recreates the fairest scenario possible, where all SDRs are evenly separated as illustrated in Fig. 5.1(a). In this scenario, the signal power from Alice to Bob equals the signal power to Eve, thus, Bob needs to generate a noise signal which can superpose the information signal on Eve, and at the same time cancel the SI. This represents the most common situation of using a contacless card, where a person who desires to capture the information is side by side with the person with the legitimate card.

The Fig. 5.1(b) depicts the **Setup II**, where Bob is in advantage because the distance



(a) **Setup I** - Fairest setup possible.

(b) **Setup II** - Setup advantageous to Bob.

(c) **Setup III** - Setup advantageous to Eve.

Figure 5.1: Representation of the setup scenarios, where 5.1(a) is the fairest possible, 5.1(b) is advantageous to Bob, and 5.1(c) is advantageous to Eve.

between Alice and Bob is half than Alice to Eve. This allows for Bob to receive a better signal from Alice than Eve, hence, Eve has a Signal to Noise Ratio (SNR) significantly lower than Bob, which will allow to degrade more the information signal with the same jamming power, not affecting the SI capability of Bob. This situation represents a possible situation of the thief's position, where he is on the other side of the payment terminal (this scheme does not take into account any wall in the middle, thus, it is even more difficult to the thief if the terminal payment is fixed on a wall).

Finally, **Setup III** explores a disadvantageous scenario for Bob, where its distance to Alice is double than Eve's distance, as indicated on Fig. 5.1(c). Therefore, Bob is receiving lower signal power from Alice than Eve, which will pressure Bob to decrease its jamming signal, otherwise it will not be able to recover the signal, and consequently, Eve will receive the Alice's signal with higher power. In the context of the example given, this is the most difficult situation to occur as the thief needs to be in the middle of the communication, i.e. needs to be in the middle of the payment terminal and the person who is paying. However, this situation needs to be evaluated because this concept may be used for a lot of other situations.

In Section 3.2 several theoretical and practical security metrics were presented. Considering the practical security metrics, the ones chosen to evaluate the experiments were: Error Vector Magnitude (EVM), Packet Loss Ratio (PLR) and Bit Error Rate (BER) (at received packets only).

With EVM is possible to directly observe the convergence of the constellation after the signal goes through the adaptive algorithm, thus, the lower EVM values means better possibility of correctly demodulating the receiving signal. This is a metric that acts upon the signal itself being blind to the data actually received, i.e. even transmitting random symbols, the constellation will converge without actually receiving any interesting data. This problem is even worst when the SCS-HK scheme is implemented with any puncturing pattern, because, the decoded received data may not be correct (i.e. the Low-Density Parity-Check Code (LDPC) code may not correct the punctured bits) and the EVM value stays exactly the same. Therefore, the EVM metric alone is not enough to evaluate the scenarios mentioned.

The BER metric measures the probability of a bit error in the actual decoded received data, solving the problem that was previously mentioned. Therefore, this metric is only applicable to the received data, and does not take into consideration how many packets were lost.

Finally, the PLR measures the probability of packet loss, allowing us to evaluate the packet detection capability of the receiving actor. This is an important metric because it takes into consideration what BER does not, i.e. the information that was not received. Note that, Bob's jamming signal can disable Eve to detect a packet, hence, not trying to decode it. Even if the header codification decreases the PLR (and Eve tries to decode the payload), the effect of jamming makes Eve receive as many wrong bits as possible of payload, which is measured by the BER. Therefore, the final goal is keeping the Bob's PLR as small as possible (ideally zero), while achieving the maximum Eve's PLR and BER (when correctly detecting the packet).

5.1.2 Hardware and Settings

The ideal scenario would be to have access to three machines, one for each actor. However, only two machines were available at the moment of the experimentation, so it was selected the Central Process Unit (CPU) with better benchmark to act as Bob due to having the most computationally expensive algorithms (correction of channel's deformations and adaptive algorithm). Therefore, the other machine will act as Alice and Eve at the same time. This leads to a separation of the FlowGraph (FG) into one where the transmission is received and the data is output to a temporary file on the computer, and another that decodes the data extracting the information that was transmitted. Note that this solution does not affect any of the experiments results because the data to be decoded is exactly the same, there is only a decrease of the needed processing power since the most computational expensive algorithms are run sequentially rather than consecutively. The Table 5.1 describes the specification of both machines.

As previously mentioned in Section 2.1.3, the SDRs used are Universal Software Radio Peripheral (USRP) B210 from Ettus Research with VERT2450 antennas attached, both provided from the projects where this work is inserted and with specifications represented in the Table. 5.2. The USRP supports Full-Duplex (FD) using Multiple-Input and Multiple-

Host Machine 1 - Alice & Eve	Host Machine 2 - Bob
Ubuntu 14.0.4 LTS	Ubuntu 14.0.4 LTS
Intel Core i7-4710HQ - 4 x 2.5 GHz	Intel Core i7-6700HQ - 4 x 2.6 GHz
16 GB dual-channel 1600 MHz DDR3L SDRAM	8 GB single-channel 1600 MHz DDR4
GNU Radio Version 3.7	GNU Radio Version 3.7
USB 3.0	USB 3.0

Table 5.1: Specification of all host machines

Output (MIMO) (with 2 input and 2 output) which enable us to perform jamming in one of the Transmitter (TX) antenna and SI cancellation using the reception of one Receiver (RX) antenna. Furthermore, the antenna are optimized to work in the range specified, which will be taken into account when choosing the transmission frequency.

SDR	Antenna
USRP B210	VERT2450
Transmission from 70 Mhz to 6 GHz	2.4 GHz to 2.5 GHz and 4.9 GHz to 5.0
Up to 56 MHz of real-time bandwidth	Omnidirecional
Supports Full-Duplex with MIMO: 2TX & 2RX	
MCR from 5 MHz to 61.44 MHz	
FPGA Spartan 6 XC6SLX150	
Supports UHD in GR	
USB 2.0 and USB 3.0	

Table 5.2: Specifications of SDRs and attached antennas

Finally, taking into account the restrictions above mentioned, the main SDR's settings are defined on GNU Radio (GR) software as described in Table 5.3.

Parameter	Definition
Device Address	"serial= [Alice:F5EAE1, Bob: F5EAC0 & Eve:F5EB09]"
Device Arguments	"master_clock_rate=60e6" = 60 GHz
Sync	PC Clock (default)
Sample Rate (Sps)	375e3 = 375 KHz
Ch0: Center Freq (Hz)	2.4e6 = 2.4 GHz
Ch0: Gain Value	Alice: 55, Eve & Bob: Variable
Ch0: Antenna	TX/RX

Table 5.3: SDRs settings defined in GR

5.2 Statistical Analysis

As discussed, the experiments will be evaluated using EVM, Packet Loss (PL) and BER metrics, particularly, by extracting its mean $\overline{x}$. However, this value alone is not an indicator reliable enough, thus, it should be followed with standard deviation σ or Confidence Interval (CI). When using two times the standard deviation 2σ the probability that a value will be inside the interval $[\overline{x} \pm 2\sigma]$ is 95.45%. An even more reliable indicator is CI, which reflects how confident we are that the sample caught the true population mean μ, i.e. μ of the metric (which is unknown) is inside of the interval $[\overline{x} \pm me]$ with a confidence of $C\%$, where

$\overline{x}$ is the measured mean, me is the margin of error and C is confidence level, chosen as 90%, 95% or 99% [96].

Theoretically, for the CI work perfectly, the sample must be drawn from a Normal distribution, so the confidence level is exactly the selected $C\%$. In practice, it is also possible to use CI even with a sample that is not drawn from a Normal distribution, as long as the sample size is large enough. In a non-normal situation there is an approximation to the selected $C\%$ and the difference is devalued [96].

That said, using Student's t-distribution it is possible to construct CI for a sample of size n without knowing the real value of σ. What we desire is to use the estimated standard deviation s and estimated mean $\overline{x}$, to calculate an interval (CI) containing μ with a confidence level C. The confidence interval is given by the following equation:

$$CI = \overline{x} \pm me, \qquad me = t\frac{s}{\sqrt{n}} \tag{5.1}$$

Where t is the value for the $t(n-1)$ density curve with area C between $-t$ and t (in practice is the value in t-distribution table with C and $n-1$ degrees of freedom). In particular, for $C = 95\%$ and $n = 30$, then $t = 2.045$. Furthermore, this t procedure can be used for sample sizes of [96]:

- $n < 15$ if the sample data is close to Normal;
- $n \geq 15$ following a non-Normal, except if there are big outliers or a clearly skewness;
- $n \geq 40$ even with outliers and a skewed distribution.

In fact, an effort was made to perform normality tests on sample data, specifically, the KS Limiting Form algorithm, which returned positive, i.e. the data is normal, as suspected from transmission theory. Anyway, the first step for extracting the CI of a sample of the metrics discussed is to analyze for the presence of any outlier to remove them. Ultimately, we select a sample size of 30 and 95% of confidence level (which is the most used in practice). Therefore, for each experiment will be executed 30 transmissions, each 1-minute long each. As each transmission will be of 8000 packets with 55 bytes this results in a total of $30 * 8000 = 240000$ packets and $30 * 55 = 13.2$ MB transmitted for each experiment.

Finally, both PLR and BER metrics will be represented in a plot with y-axis in logarithmic scale because of the disparity of the measured values; thus, when a value is 0, it is not shown in the plot (it is not possible to represent 0 in a logarithmic scale). Also, to achieve a specific statistical significance of a metric, there is necessary to transmit one more order of magnitude. Particularly, for the PLR metric, as it was transmitted a total of $240000 = 2.4 * 10^5$ packets, thus, values below 10^{-4} on y-axis has no statistical significance. In the same way as before, a total of $240000 * 440 = 1.1 * 10^8$ bits are transmitted, hence, we only considered BER values above 10^{-7} on y-axis. In short, for non represented values, consider a 0 value, which may be in reality 0, or a not statistically significant value.

5.3 Experimental Results

After selecting the transmission size and the sample size of further experiments, there is a need to automatize these experiments, given that, running the FG manually 30 times for each experiment is unfeasible. In compilation operation of the FG, the GR generates a python file containing blocks' definitions and connections. Therefore, this file can be modified to receive arguments from the terminal in order to change the input/output files and a varying parameter (e.g. jamming power). It is also possible to split the FG into two components: The transmission component and the decoding operation as mentioned in Section 5.1.2. Therefore, five scripts are created for each experiment (i.e. one for Alice, two for Eve and two more for Bob). Nevertheless, each test of the experiments needs to be manually verified for errors that may occur, before (e.g. USB linking problem, non-detection of the USRP) or during the transmission (e.g. printing U/O's in the console, the correlation of the synchronization block was not detected).

As previously mentioned, first the influence of passive suppression and the effect of header codification will be evaluated. After selecting the best options, the adaptive algorithm will be evaluated to analyze their performance in SI cancellation with varying jamming powers in the most fair scenario possible. The chosen adaptive algorithm will be used to evaluate the three setups described to understand the SI cancellation capability when changing SDRs' positions. Finally, using the best algorithm and the fairest setup, the impact of implementing the SCS-HK scheme with different puncture patterns will be evaluated.

5.3.1 Evaluating Passive Suppression

In Section 3.6.1 passive suppression was discussed to help the SI cancellation process by physically changing the positions of Bob's antennas. Despite *Antenna Separation* would help to increase the path-loss effect between antennas increasing the SI cancellation, it is an option left out, because this work aims to create a compact node.

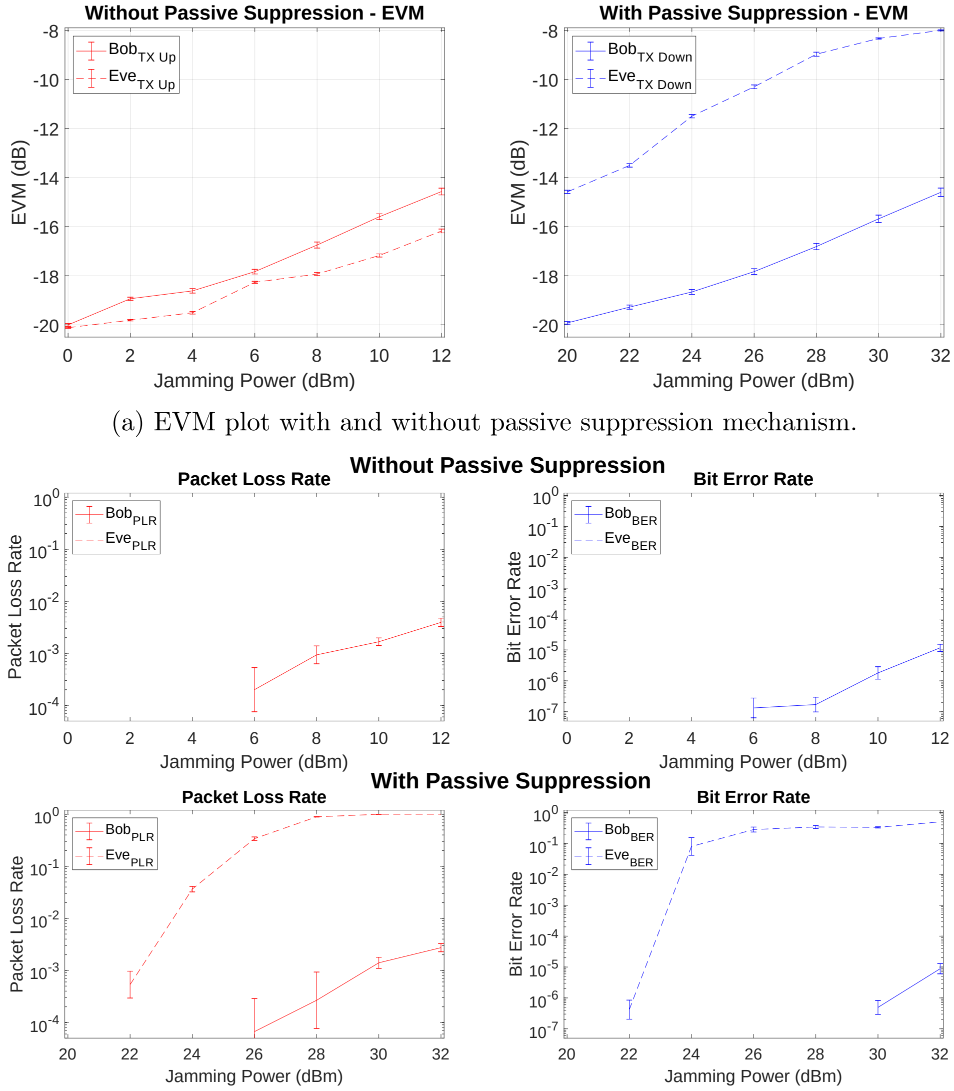
The *Directional Passive Suppression* explores the angle between antennas without separating them. While a transmission was running, the angle on Bob's antennas is manually changed to find the best angle considering the constellation recovery, resulting in a 90^{o} vertical angle as visualized in Fig. 5.1(a). To confirm, it was experimented using the Least Mean Squares (LMS) filter in the fairest setup (i.e. **setup I**) with and without passive suppression. The results are depicted in Fig. 5.2(a) and Fig. 5.2(b).

Although Bob's EVM on both options is similar, it is important to note that when using passive suppression the jamming power needs to be lower. This is caused by the closeness between Bob's TX and RX antenna. Therefore, the receiving signal (which includes the jamming signal) has high energy, consequently, it is necessary to lower the jamming power to manage the EVM values to be close as possible between the two options. As might be expected, this lowers the interference in the Alice's signal on Eve, allowing her to recover the constellation. Finally, without passive suppression, Eve does not have any packet loss, neither wrong bits in all jamming powers.

In another words, looking only to EVM, it is clear that the passive suppression, not only help the SI cancellation, but is crucial to achieve a secure communication against Eve, allowing us to achieve 20 dB of SI suppression (when combined with LMS filter) on Bob. For this reason, passive suppression will be used for further experiments.

5.3.2 Evaluating Header Codification

The "Correlate Access Code - Tag Stream" block discussed and implemented in Section 4.1.1.1 allows to extract a packet's payload by performing a bit level comparison with a known word (without any error correction). The threshold defines the maximum number of different bits that can be wrong on the received known word to accept it.



(a) EVM plot with and without passive suppression mechanism.

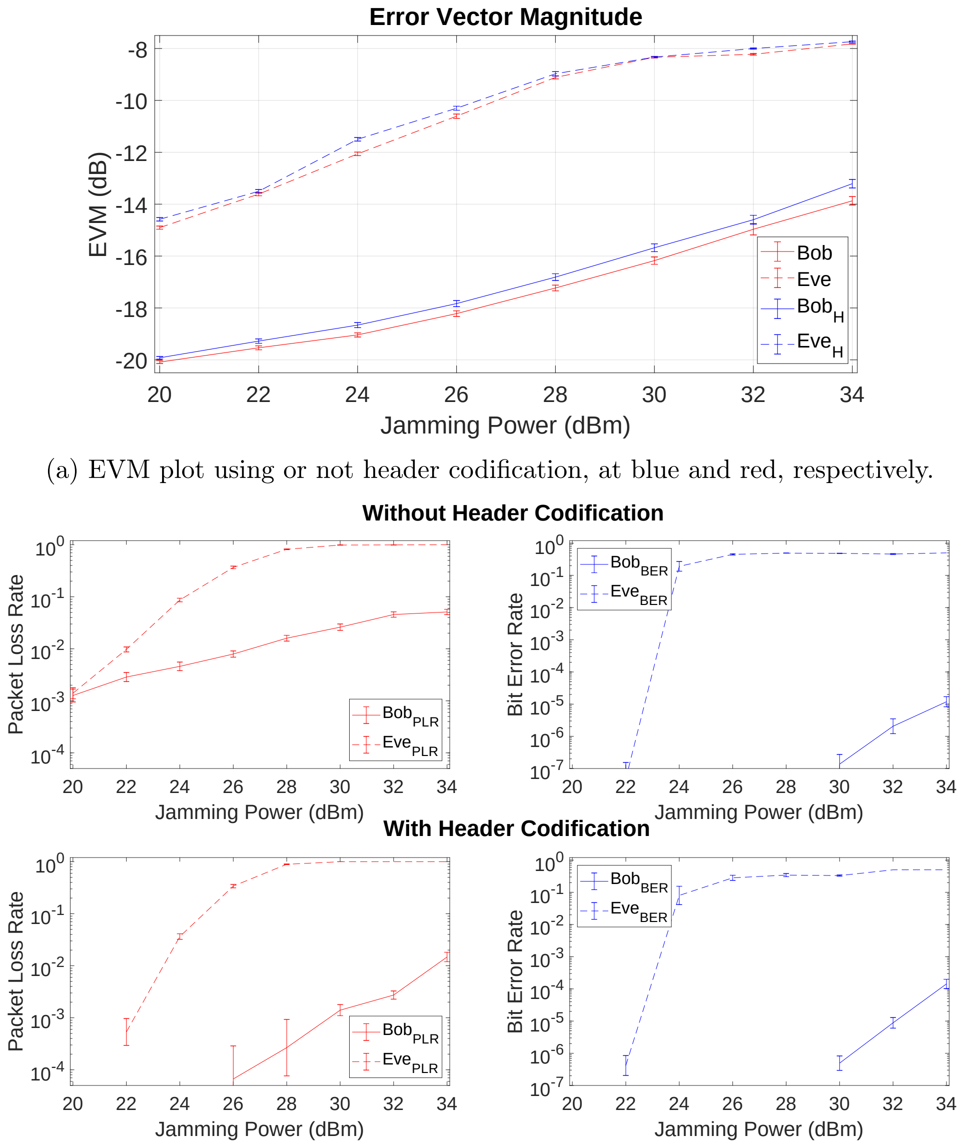
(b) PLR, and BER plots with and without passive suppression mechanism.

Figure 5.2: Influence of passive suppression in SI cancellation for the chosen metrics.

The Reed-Solomon (RS) algorithm discussed in Section 2.5.3 and implemented in Section 4.2.2 encodes the header to allow error correction of the symbols, which was defined (in the mentioned parameters) to be able to correct up to 10 symbols. Considering this error correction capability, the threshold parameter of the "Correlate Access Code - Tag Stream" block was defined to 10 (i.e. there may be up to 10 different bits from the known word and the header is still accepted), otherwise, the PLR on Eve would be significantly high for most interference powers.

With header codification, not only Bob will be able to recover errors in the transmission, but also Eve will be able to do it. Moreover, not being able to catch the header does not mean that the information did not reach Eve's antenna, and may use it *a posterior* in some way to decode the payload. Thereupon we may also decrease the PLR on Eve's side, however, for sure we increased our certainty about the security of the scheme. In short, the main motivation behind header codification is to decrease Bob's PLR.

To obtain reliable results, this experiment needed to be carried out in the fairest environment possible; therefore **Setup I** was chosen (Fig. 5.1(a)), using the passive suppression



(a) EVM plot using or not header codification, at blue and red, respectively.

(b) PLR and BER plots of using or not the header codification.

Figure 5.3: Influence of header codification on its detection for the chosen metrics.

as previously established. The Fig. 5.3(a) plots the EVM value and Fig. 5.3(b) plots the remaining metrics along with the correspondent CIs. It was evaluated for jamming power values from 20 dBm to 34 dBm with 2 dBm step. Below that, the difference between having/not having the header encoded is not significant considering security since Eve is able to detect all packets.

The EVM metric in this case does not give us any information regarding header codification because it measures the signal itself (all received symbols, being these correct or not) and not the data; thus, it is not a good indicator to achieve any conclusion. Instead, allow us to understand if both receivers are in the same channel conditions, in which case, the EVM values would be exactly the same. However, in real-world, due to environmental randomness (e.g. power of devices, multipath effects, etc.), these values have a small difference. In particular, both Eve and Bob performs slightly better without header codification (note that $\geq$ 28 dBm of jamming power the constellation is so disperse that the EVM does not significantly increase; thus, the difference between them is narrowing). That said, it is necessary to analyze other considered metrics, such as PLR and BER.

Considering the PLR metric it is notorious the difference caused by the header codification for Bob at all jamming values while the Eve is not much affected for values $\geq$ 24 dBm. In fact, by encoding the header, Bob's capability to detect packets significantly improves, as observed by the larger gap between Eve and Bob's values. In particular, header codification allows to detect all packets until 26 dBm of jamming power, which in this case, Bob's PLR is 6.67×10^{-5} and Eve's PLR is 3.39×10^{-1}, while without header codification is 7.93×10^{-3} and 3.68×10^{-1}, respectively for the same jamming power. Note that for the relatively same Eve's value (the small difference is due to channel conditions), there is an improvement in 0.79%. Although this value is not rather high, it may be important to recover as much data transmitted as possible, and even more important when transmitting data that is sensitive to packet loss (e.g. video or image).

The best security considering the maximum PLR possible on Eve and minimizing Bob's PLR is achieved when jamming power is 28 dBm. In this case, for the relatively same Eve's PLR (8.92×10^{-1} with header codification and 8.18×10^{-1} without it), the Bob's PLR using header codification is 2.67×10^{-3} and when it is not using is 1.60×10^{-2}, representing an improvement of 1.3%. This means that Eve loses 89% of the transmitted packets while Bob only loses 0.026% of them. Furthermore, the highest improvement for all of all jamming power values considered is achieved at 32 dBm of jamming power with an improvement of 4.29% due to the severe interference generated. This situation is useful when we desire Eve to be completely blind (as Eve's PLR is 1).

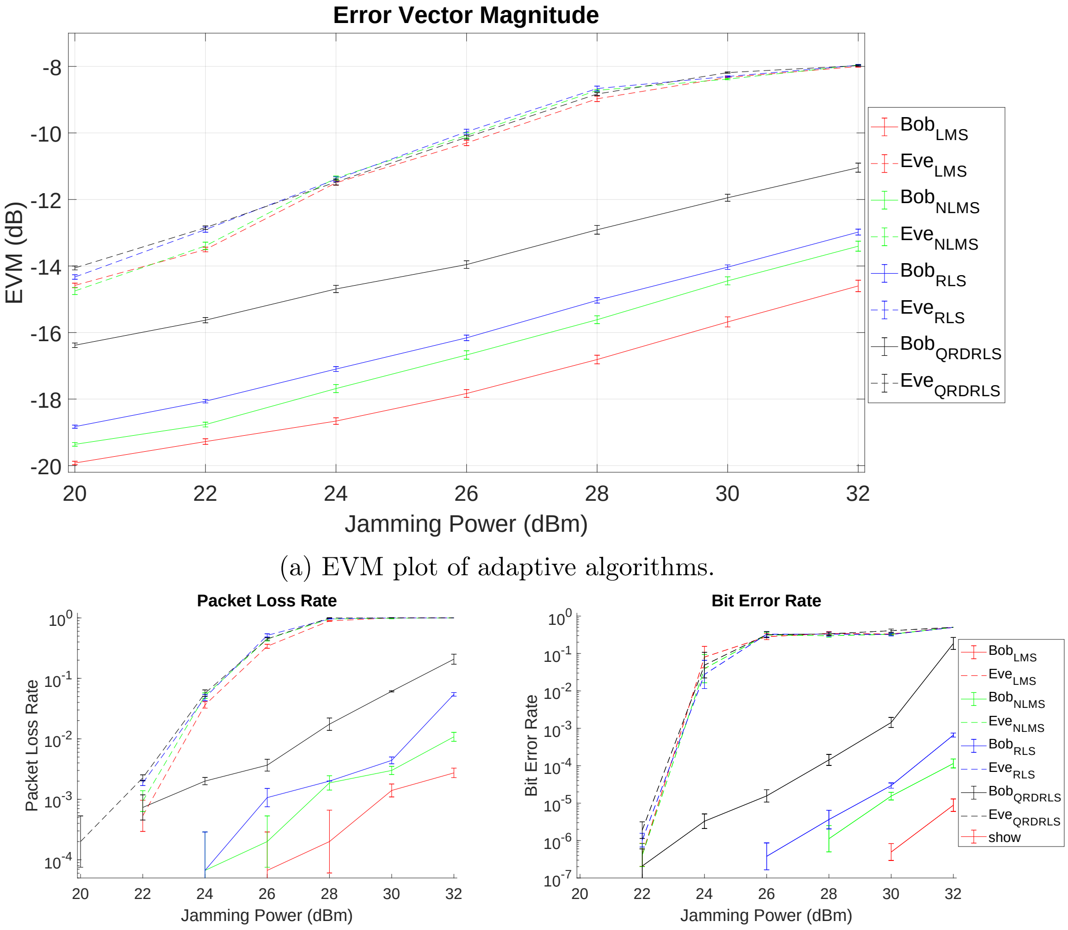
As mentioned, header codification also allows Eve to receive more packets as observed for 20 and 22 dBm of jamming power. In the first, allows her to decode all the packets and on the second has a PLR of 5.33×10^{-4}. The fact that only impacts in these values and not for higher ones (at least not significantly), evidence that the jamming effect is significantly working.

Finally, considering the BER metric, any differences may be due to the slight difference in channel's conditions discussed above or a packet being accepted without actually being one, i.e. a sequence with 10 differing bits from the known word is considered as packet (even without being one) and extracted, which results in all the bits of the supposed payload being counted as wrong bits. When using the same FG this may be a problem since it will require the LDPC code to try to decode the supposed packet. This is observable for jamming values greater or equal to 26 dBm, where Eve is slightly worse using header codification.

Unquestionably, header codification allows better performance in Bob (mainly when transmitting data highly sensitive to packet loss) while not helping Eve to recover more packets due to the interference caused by the jamming signal (for values $\geq$ 24 dBm of jamming power). That said, for the next experiments header codification will always be employed.

5.3.3 Evaluating Self-Interference Cancellation Adaptive Algorithms

Now that we have defined the best options to further experiments (i.e. using both passive suppression and header codification), the first goal is to evaluate the adaptive algorithms discussed in Section 3.6.2 regarding to SI cancellation capability, namely LMS, Normalized Least Mean Squares (NLMS), Recursive Least Squares (RLS) and QR Decomposition Recursive Least Squares (QRD-RLS). Ideally, we would evaluate all these algorithms for all setup scenarios mentioned with a set of jamming powers, however, that is infeasible for the amount of needed tests (i.e. 4 algorithms $\times 3$ setups $\times 7$ jamming powers ×30 tests = 2520 total tests). Moreover, if a specific algorithm A outperforms B in the fairest scenario, for sure B will never outperform A weather is in another scenario with an advantageous or a disadvantageous Bob's position. That said, it was chosen the fairest scenario possible (**Setup I** represented in Fig. 5.1(a)) to evaluate the SI cancellation capability of the adaptive algorithms by varying the jamming power from 20 dBm to 32 dBm with 2 dBm as step. Also, the parameters used for the algorithms are the recommended ones that were discussed in Section 3.6.2, with as much taps t as the processor can handle considering the maximum sample rate S possible. In particular, for LMS: $\mu = 0.0075, t = 64, S = 405.4054054$ KHz; for NLMS: $\mu = 0.0075, t = 32, S = 405.4054054$ KHz; for RLS: $\sigma = 1, \lambda = 1, t = 12, S = 200$ KHz and for QRD-RLS: $\sigma = 1, \lambda = 1, t = 12, S = 200$ KHz.



(a) EVM plot of adaptive algorithms.

(b) PLR and BER plots of adaptive algorithms.

Figure 5.4: Comparison between the LMS, NLMS, RLS and QRD-RLS algorithms.

The EVM values and the respective CIs for all algorithms are depicted in Fig. 5.4(a), and the remaining metrics in Fig. 5.4(b). In the first, as before, the Eve's EVM values allow us to understand if, with the same channel conditions, the receivers achieve the same EVM values, while varying the adaptive algorithm. It is possible to note that the difference in Eve's values over algorithms is insignificant, except when jamming power is 22 dBm. Here, the LMS and NLMS have a slightly better EVM value than the others algorithms. This suggests that some change has occurred in the environment, modifying the multipath effect and, consequently, allowing Eve to receive a slightly cleaner constellation. Either way, the difference is not significant, but explains the slightly lower Eve's PLR in these algorithms for that jamming power value.

It is important to remember that the EVM gap is the difference between Bob's EVM and Eve's EVM, consequently, larger EVM gap between them means a better security. Moreover, the worse BER value is 0.50, corresponding to same probability of having a bit 1 or 0.

In fact, even without considering the actual received data, by exclusively examining the EVM metric it is possible to jump directly to conclusions, simply because a better Bob's EVM means that constellation points are more aggregated, and consequently, better SI cancellation was performed (bigger EVM gap means better security). Therefore, clearly the LMS algorithm outperforms the others in a matter of constellation convergence, especially, when compared with the QRD-RLS algorithm and even the RLS. Nevertheless, with this comparative analysis between algorithms, although we can recognize which is the best, it does not mean that it properly works in the actual received data, so let's consider other metrics to understand how different, in practical received data values, the algorithms are.

Considering the other metrics, the LMS algorithm allows a better recovery of the information for all jamming values, specially for high ones. Assuming the jamming value of 32 dBm, with the LMS algorithm it is possible to achieve 2.73×10^{-3} of PLR against 1.08×10^{-2}, 5.44×10^{-2} and 2.8×10^{-1} for others. When considering the errors in the received packets, the QRD-RLS algorithm is by far the worse (with 1.85×10^{-1}), while the LMS achieve only 8.82×10^{-6}. Finally, as the LMS filter outperforms the others and considering its sample rate performance, it will be chosen to the next experiments.

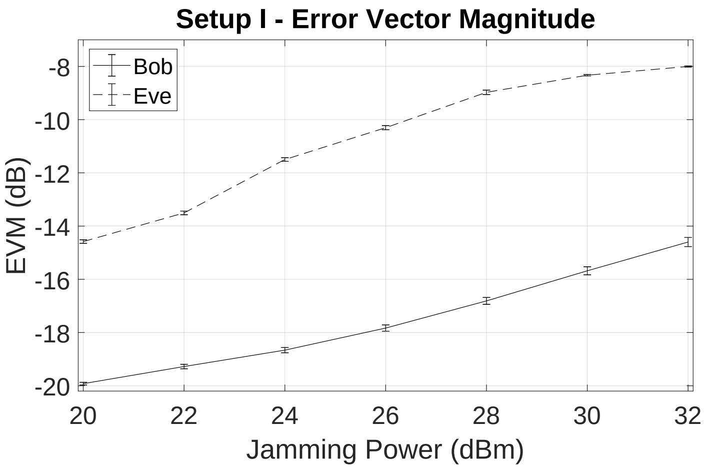
Finally, as these results are obtained from a real testbed that is affected by all the real world phenomena of communications (e.g. frequency offset, clock offset, multipath propagation), there is some expected variability in the curves as the jamming power varies. The confidence intervals for both Bob and Eve are presented and follows the same behavior for all jamming values.

5.3.4 Evaluating SDR Positions Setups

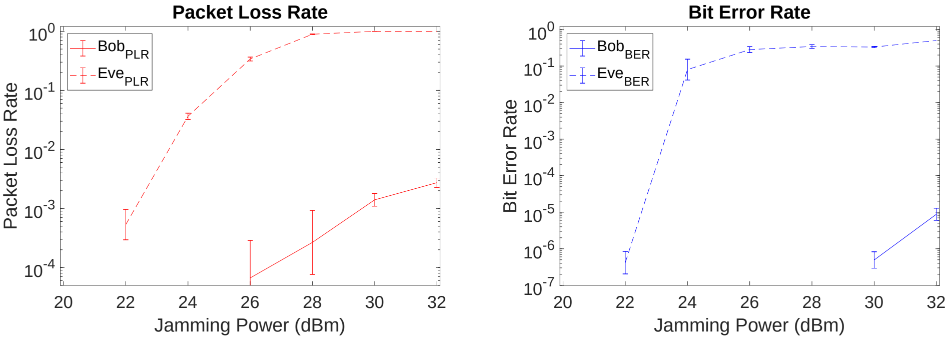
With the LMS filter established, the following fact to take into consideration will be the SDRs' positions by exploring its impact on both advantageous and disadvantageous setups for Bob. For this, the **setups I** (fair), **II** (advantageous for Bob) and **III** (disadvantageous for Bob) will be considered.

In fact, when considering only the **setup I**, the LMS algorithm is resistant to PLR up to 26 dBm of jamming (6.67×10^{-5}, at that point), while Eve loses little more than a third of the packets transmitted as represented on Fig. 5.5(b). Within the received packets, Bob has no errors (not represented in the graph because its logarithmic) while Eve has 2.82×10^{-1} of BER. In another words, this specific configuration has a considerable level of security since Bob is capable of recovering almost all packets with all bits correct while Eve recovers only 28% of the bits of one third of the received packets. However, to achieve even more secrecy the 28 dBm of jamming power is a better option, as 89.2% of the packets are not caught in Eve's side and has a BER of 3.43×10^{-1}, while Bob only loses 2.67×10^{-2}% of the packets providing the same BER as before (which is 0). Also, with higher jamming values the Eve's PLR is approximately 1 and Bob keeps lower values (1.4×10^{-3} and 2.73×10^{-3} of PLR) which is useful for situations where we desire Eve to be blind. The EVM plot of this setup is depicted in Fig. 5.5(a).

When comparing **setup I** with **setup II**, as the distance from Alice to Bob was not changed the SI cancellation capability on Bob remains the same, thus, the jamming values to be evaluated are untouched. As observable in Fig. 5.6(a), Bob's EVM values were slightly



(a) EVM plot of **setup I**.



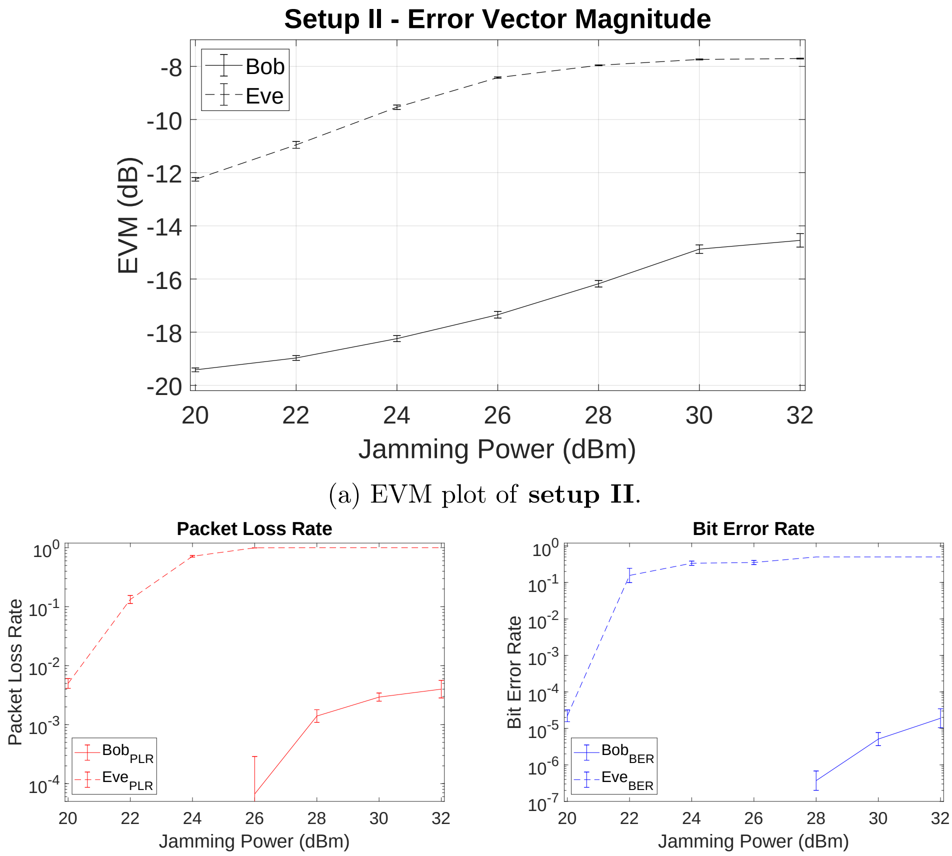
(b) PLR and BER plots of **setup I**.

Figure 5.5: Results of **setup I**.

varied due to modifications on the environment (e.g. multipath effect), however, these variations are not significant. With respect to Eve's EVM values, a general increase is noticeable, but particularly for lower values of jamming power. This is expected since Eve is now in a disadvantageous situation, placed further away from Alice. This difference allows to have an increase of the EVM gap resulting in a better security.

As might be expected, other metrics' plots reflects the EVM gap discussed above, which is reflected in the left shift of Eve's values in setup **II** (Fig. 5.6(b)) when compared to setup **I** (Fig. 5.5(b)), leading to a larger secrecy margin (i.e. for the same jamming power, Eve performs worst than before). Obviously, this setup increases the security of a communication, particularly, it is possible to secure a communication with 24 dBm of jamming power Eve's PLR is 7.16×10^{-1} and has 3.36×10^{-1} of BER while Bob is unaffected either in packet loss or in errors. In short, due to Bob's advantage, this setup outperforms the previous one for lower jamming power.

As discussed, the transmission power is affected due to path-loss effect when changing the antenna's distance. Therefore, the jamming powers need to be modified to be able to maintain the same SI cancellation capability, i.e. the Bob's EVM line on **setups I** and **II** is relatively the same as **setup III**, only differing the Eve's capability to recover the signal (Eve's EVM), as depicted in Fig. 5.7(a). In **setup III** the Eve's advantage over **setup I** is accomplished by keeping the same distance to Alice, while Bob's distance is doubled. The path-loss effect is given by Friis equation and states that attenuation is proportional to the distance quadrature, specifically, by the following equation: $FSL = \frac{(4\pi d)^2}{l^2}$, where d is distance and l is the wavelength. Therefore, as the wavelength is not modified and the



(a) EVM plot of **setup II**.

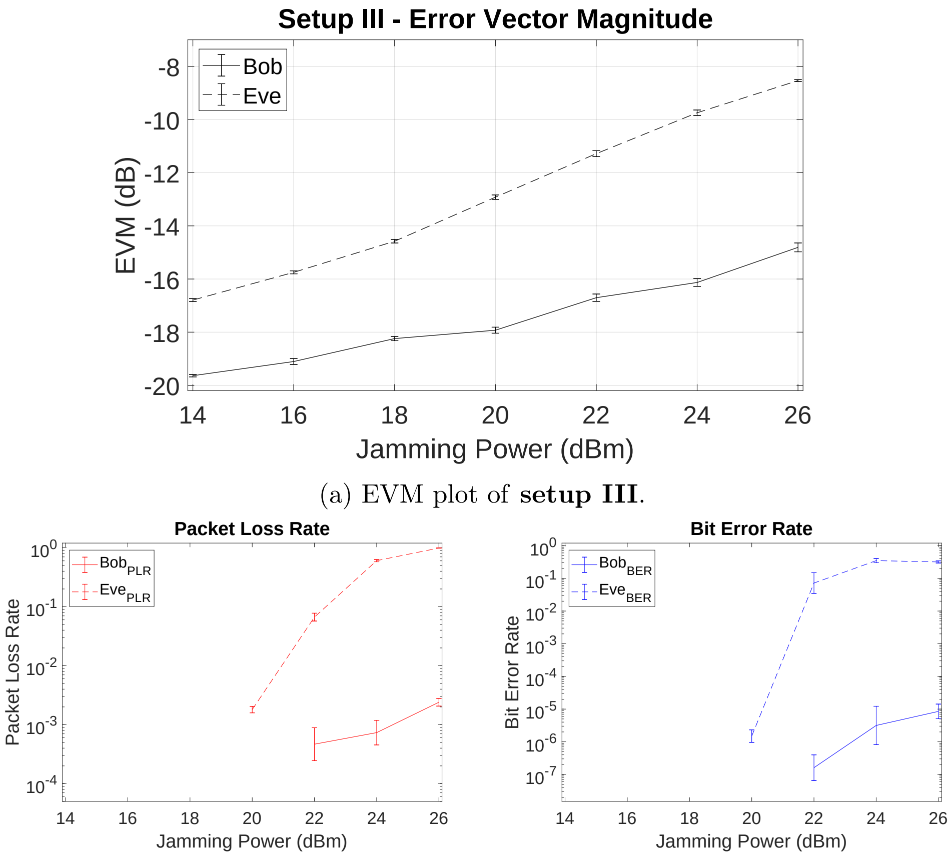
(b) PLR and BER plots of **setup II**.

Figure 5.6: Results of **setup II**.

distance is doubled, it is expected an attenuation of $10log_{10}2^2 = 6$ dBm. With this in mind, maintaining the same SI cancellation capability (i.e. the same Bob's EVM values) it was achieved the the jamming values from 14 dBm to 26 dBm resulting in a 6 dBm of path-loss attenuation, which is exactly what was theoretically calculated. Still considering the EVM metric, in contrast to **setup II**, the gap between Bob and Eve was undoubtedly reduced, hence, lowering the security level. Furthermore the remaining metrics corroborate what was mentioned, since the Eve's curves are shifted to right, thus requiring more jamming power to affect Eve as observable in Fig. 5.7(b). This is, Eve is no longer affected until 20 dBm of jamming power and at this point there is a low security level since its PLR is 1.8×10^{-3} and its BER is 1.49×10^{-6}.

Anyway, even with a lower security gap it is possible to achieve a secure communication by employing 24 dBm of jamming. In this case, Bob is not able to recover 7.33×10^{-2}% of the packets and 3.16×10^{-4}% of bits of these packets, while Eve does not recover 60.4% and has 34.8% of wrong bits. For higher jamming powers, Eve's PLR is nearly to 1, particularly, 26 dBm of jamming power is an option in a situation where we accept to lose 2.4% of the packets (while Eve is blind).

Concluding, the **setup I** allows us to provide a basis for comparison to other setups. In **setup II** the EVM gap was increased, thus, there was a higher security gain and on the contrary, in **setup III** the EVM gap was decreased narrowing the margin of security. Either way, in all setup scenarios it is possible to achieve a certain level of secure communication only by using noise generation and its cancellation on the legitimate receiver. Also note that there was not strictly defined a better/worse jamming power for a communication.



(a) EVM plot of **setup III**.

(b) PLR and BER plots of **setup III**.

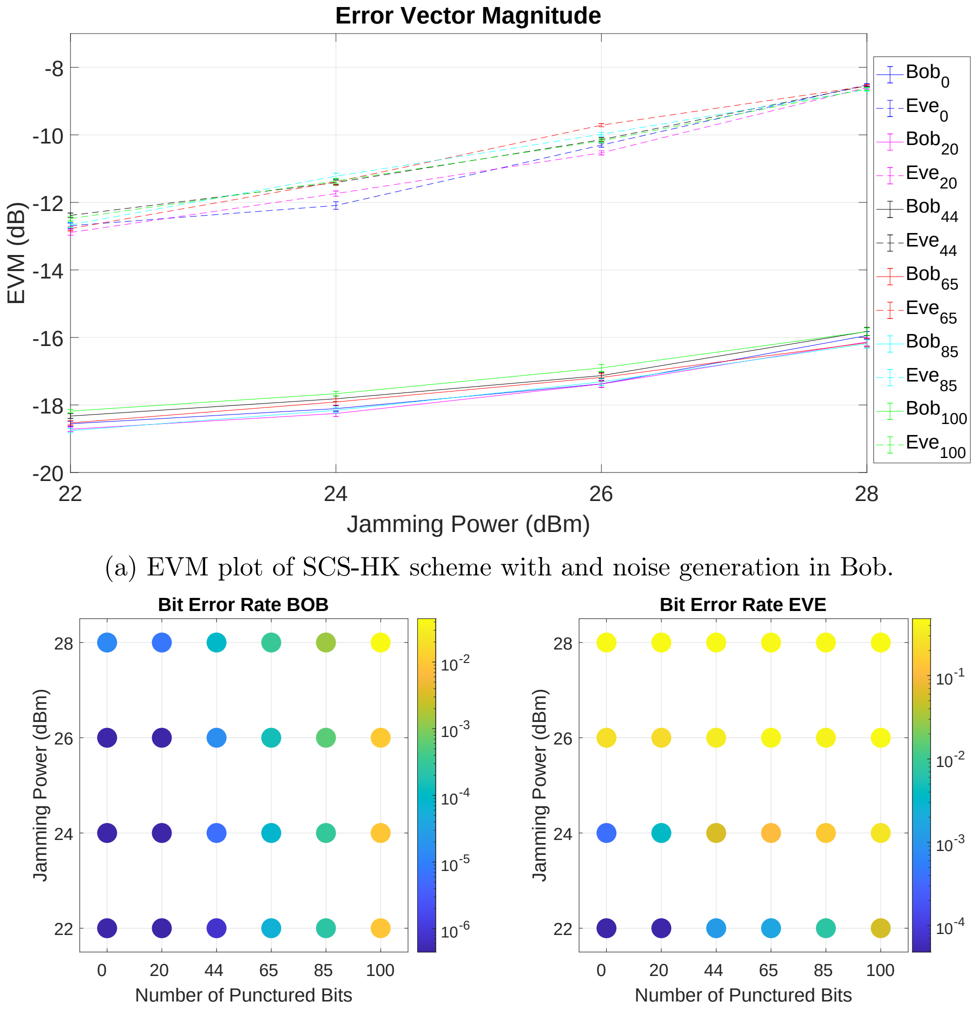
Figure 5.7: Results of **setup III**.

The jamming power to be selected to secure a specific communication is a trade-off with packet loss and data errors that we accept on Bob and the received packets and correct data that we allow Eve to collect.

5.3.5 Evaluating the adapted SCS-HK scheme

The SCS-HK scheme detailed in Section 3.3 exploits the usage of puncturing along with the LDPC code to increase secrecy in a communication. An evaluation of original SCS-HK scheme was performed in [62] where, in order to degrade the Alice to Eve channel, these actors were positioned in such a way Eve was behind a wall, unlike Alice to Bob for which there was a direct path. When combined the adapted SCS-HK with the jamming component of this work, the goal is to degrade the channel between Alice and Eve without the need for a physical artifact such as a wall in between.

Another point of view with the same objective is to implement the adapted SCS-HK detailed in Section 3.3.1 along with the jamming component of this work to increase the Eve's BER. Therefore, it will be evaluated the impact of several puncture patterns (i.e. number of bits punctured in the codeword) for selected values of jamming (chosen considering lower PLR and BER jamming). Specifically, it was evaluated to $0, 20, 44, 65, 85, 100$ bits punctured for $22, 24, 26, 28$ dBm jamming values. Note that 0 bits punctured means that it was only added the additive scrambler, this allow us to have a basis for comparison. Also, this experiment was carried out using the fairest setup possible (**setup I**) to evaluate in a fair manner the impact of SCS-HK not only on Eve but also on Bob.



(a) EVM plot of SCS-HK scheme with and noise generation in Bob.

(b) BER plots of SCS-HK scheme and noise generation in Bob.

Figure 5.8: Impact of the SCS-HK scheme with $0, 20, 44, 65, 85$ and 100 bits punctured when combined with noise generation mechanism.

Once again, as the SCS-HK does not affect in anything the point's aggregation in the constellation, the EVM values at this situation, represented on Fig. 5.8(a), are only meant to analyze if the experiment occurred in the same conditions (if there are any existing differences, these are related with environmental situations not from the SCS-HK scheme). As well as the PLR will not be affected, since the SCS-HK scheme only punctures the payload bits, hence, the most important metric here to consider is the BER, represented in Fig. 5.8(b).

Considering the BER metric, it is a fact that the SCS-HK has an impact, which is observed in Fig. 5.8(b), where the increase of the number of punctured bits also increases the BER for most situations, meaning that more errors are caused, particularly for Eve. The one situation in which increasing the number of puncturing bits produced no effect is when a high jamming power of 28 dBm is employed, which already causes a high BER for Eve. This means the SCS-HK scheme and self-interference/jamming are two complementary approaches to increase the security level for wireless networks. An important detail to remember is that the goal is looking for the combinations where the Eve's BER increases quicker than the Bob's BER allowing us to have a combination where puncturing interference is significantly heavier on Eve (because is combined with jamming and in Bob the jamming is canceled).

When looking only at Eve's side when jamming is 28 dBm, it is clear that there is no improvement when adding SCS-HK scheme on Eve's side, since her BER remains relatively the same value when varying the number of punctured bits (between 4.49×10^{-1} and 4.82×10^{-1}) and Bob's BER significantly increases, which is exactly the opposite of our intent. Therefore, there is no interest of using the SCS-HK scheme with this level of jamming power, so it is excluded from further analysis.

In the same way, when considering 22 dBm of jamming power, as the noise generation power is not strongly affecting Eve the SCS-HK scheme not only affects her, but also impacts Bob considerably. Thus, the SCS-HK scheme is not worth implementing in a transmission with this jamming power value to this scenario. Moreover, there are combinations with 24 dBm of jamming with relatively the same Eve's BER and considerable lower Bob's BER, particularly, when 22 dBm of jamming power and 100 bits punctured, the Eve's BER is 5.60×10^{-2} and Bob's BER is 9.61×10^{-3} while in 24 dBm of jamming power and 44 bits punctured, Eve's BER is 6.11×10^{-1} and Bob's BER is 6.07×10^{-6}, i.e. with relatively the same Eve's BER, the Bob's BER is significantly lower. In conclusion, this jamming value is also excluded for further analysis.

When employing both using 24 dBm and 26 dBm of jamming power it is notorious the impact of the SCS-HK scheme. However, once again, it is not possible to strictly say which combination is the best, since the choice to a specific transmission is a trade-off between the packet loss and data errors that we accept on Bob and the received packets and correct data that we allow Eve to collect. Instead, there are some combinations more interesting than others as it will be addressed.

Fixing the same order of magnitude on Bob's BER (10^{-5}), it is important to mention that the combination of 26 dBm of jamming power and 44 of bits punctured significantly outperforms the 24 dBm of jamming power with 65 bits punctured, achieving 3.54×10^{-1} and 1.10×10^{-1} on Eve's BER, respectively. Considering this situation and considering the PLR, in the first combination (26 dBm of jamming power and and 44 of bits punctured) we accomplish a total of BER of 3.54×10^{-1} and a PLR of 3.39×10^{-1} on Eve (note that BER is more 5.70×10^{-1} when compared with 2.82×10^{-1} mentioned in Section 5.3.4 without puncturing) and a total of BER of 1.63×10^{-5} and a PLR of 6.67×10^{-5} on Bob (note that BER increased 1.63×10^{-5} when compared to 0 mentioned in Section 5.3.4 without puncturing). Finally, When using 24 dBm of jamming, as the Eve's PLR is low (3.64×10^{-2}, discussed Section 5.3.4), even with BER of 1.10×10^{-1}, there is no interest in being a choice to any communication.

Furthermore, for higher values of puncturing (65, 85 and 100) when using 26 dBm of jamming power, there is no significant variation on Eve's side, thus, there is no much interest in these combinations.

In conclusion, it is possible to secure a communication only using the noise generation, however, when combined with the SCS-HK scheme, for a selected combination of parameters (i.e. number of bits punctured and jamming power) it is possible to increase Eve's BER with a minor interference on Bob, thus improving the security level.

Chapter 6

Conclusion

With the growth of the Internet and usage of wireless devices, security becomes a concern and this work addresses that on a physical level as an complement to the present upper layers mechanisms like cryptography.

To master the techniques required for this work, it was necessary to acquire background knowledge on communication systems. For that, a great effort was put into learning all the communication fundamentals and software needed to create a simple testbed communication scenarios. This resulted in a tutorial produced for that effect.

Yet in the research phase, was developed the state of art of physical layer security where was addressed among other things the classic wiretap channel, useful metrics for future work, and mainly the deepening working of the adapted coding for secrecy scheme implemented. Several other important concepts were addresses, such as jamming, cooperative jamming and the necessary theoretical information background to employ full-duplex jamming along with the necessary algorithms to cancel the self-interference.

In fact, this thesis adapts and implements a coding scheme for secrecy, in particular, the Scrambled Coding for Secrecy with a Hidden Key (SCS-HK) using as software the GNU Radio (GR) and as hardware the Universal Software Radio Peripheral (USRP)'s B210. With this scheme it was possible to successfully transmit and receive information over the air, being this information, text, image and even video. It also implements the noise generation along with the necessary synchronization to cancel the Self-Interference (SI) on the receiver side, and finally implements and evaluates these two mechanisms individually and combined.

Besides, it has been proven that passive suppression mechanism not only helps the SI cancellation, but is crucial to achieve a secure communication against Eve (up to 20 dB of SI suppression when combined with Least Mean Squares (LMS) filter on Bob), and the header codification is an important feature which in fact, allows a better packet detection to allow a transmission with information highly sensitive to packet loss. Furthermore, the LMS outperforms over all other algorithms considered (Normalized Least Mean Squares (NLMS), Recursive Least Squares (RLS) and QR Decomposition Recursive Least Squares (QRD-RLS)) being the best choice to use in this situation (SI cancellation).

It was shown that it is possible to secure a communication only considering noise generation by the legitimate receiver, in particular, considering a fairest scenario, for 26 dBm of jamming power, it is possible to achieve a Packet Loss Ratio (PLR) of 3.39×10^{-1} and a Bit Error Rate (BER) of 2.82×10^{-1} on Eve's side, affecting only 6.67×10^{-5} of Bob's PLR. For even more security, 28 dBm of jamming power can be considered which achieves a PLR on Eve's side of 8.92×10^{-1} with a BER of 3.43×10^{-1} affecting only the Bob's PLR of 2.67×10^{-4}. It was also shown that for both advantageous and disadvantageous scenarios for Bob the PLR/BER gap is increased and decreased, respectively; either way being possible to secure the communication in each case.

Finally, when noise generation is combined with the adapted SCS-HK scheme, it was observed a security improvement by increasing Eve's BER with minor interference on Bob, in particular, by using 26 dBm of jamming power and 44 bits of puncture combination in a fairest scenario, was achieved 3.39×10^{-1} of PLR and 3.54×10^{-1} of BER on Eve's side, while Bob's PLR is 6.67×10^{-5} and its BER is 1.63×10^{-5}.

In conclusion, it is possible to secure a communication only using the noise generation by legitimate receiver; however, when combined with the SCS-HK scheme information security is increased through a higher BER at Eve with a minor interference on Bob.

6.1 Future Work

Future directions of this work include the usage of directional antennas to achieve a higher Error Vector Magnitude (EVM) gap, specifically, when it is not a setup with the Software Defined Radio (SDR)s aligned, and thus, possible to direct the jamming signal to an eavesdropper (only when it is assumed that his position is known).

In order to improve the throughput it is also possible to improve the FlowGraph (FG) to work with Low-Density Parity-Check Code (LDPC) belief propagation decoding, which will recover better the wrong bits while increasing the throughput. Also, to achieve even lower PLR on Bob's side, it is possible to implement a number for each packet and implement re-transmission techniques in order for Bob to "ask" for a specific packet, recovering it.

Finally, it is possible to occur a decrease in processing power of the Bob's host computer which will back-pressure the "Synchronizer" block, consequently losing the synchronization between streams requiring to start over the transmission. Perform the synchronization repeatedly over a period of time could be implemented to prevent that from happening.

References

[1] W. Stallings, *Cryptography and Network Security.* Pearson Education, 2017.

[2] C. E. Shannon, "Communication theory of secrecy systems," *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.

[3] A. D. Wyner, "The wire-tap channel," *The Bell System Technical Journal*, vol. 54, no. 8, pp. 1355–1387, 1975.

[4] U. M. Maurer, *The Strong Secret Key Rate of Discrete Random Triples*, pp. 271–285. Boston, MA: Springer US, 1994.

[5] D. Sarmento, M. A. C. Gomes, J. P. Vilela, and W. K. Harrison, "Analysis of short blocklength codes for secrecy," *EURASIP Journal on Wireless Communications and Networking*, 2018.

[6] D. Klinc, J. Ha, S. W. McLaughlin, J. Barros, and B. Kwak, "LDPC codes for the gaussian wiretap channel," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 532–540, 2011.

[7] M. Vigilante, E. McCune, and P. Reynaert, "To evm or two evms?: An answer to the question," *IEEE Solid-State Circuits Magazine*, vol. 9, pp. 36–39, Summer 2017.

[8] M. Baldi, M. Bianchi, and F. Chiaraluce, "Coding with scrambling, concatenation, and HARQ for the AWGN wire-tap channel: A security gap analysis," *IEEE Transactions on Information Forensics and Security*, vol. 7, pp. 883–894, 06 2012.

[9] J. P. Vilela, M. Gomes, W. K. Harrison, D. Sarmento, and F. Dias, "Interleaved concatenated coding for secrecy in the finite blocklength regime," *IEEE Signal Processing Letters*, vol. 23, no. 3, pp. 356–360, 2016.

[10] E. Tekin and A. Yener, "Achievable rates for the general gaussian multiple access wire-tap channel with collective secrecy," *ArXiv*, vol. abs/cs/0612084, Dec. 2006.

[11] X. He and A. Yener, "Providing secrecy with structured codes: Tools and applications to two-user gaussian channels," *CoRR*, vol. abs/0907.5388, 2009.

[12] J. Xie and S. Ulukus, "Secure degrees of freedom of the gaussian wiretap channel with helpers," in *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 193–200, Oct 2012.

[13] S. Gollakota and D. Katabi, "Physical layer wireless security made fast and channel independent," in *2011 Proceedings IEEE INFOCOM*, pp. 1125–1133, April 2011.

[14] G. Zheng, I. Krikidis, J. Li, A. P. Petropulu, and B. Ottersten, "Improving physical layer secrecy using full-duplex jamming receivers," *IEEE Transactions on Signal Processing*, vol. 61, pp. 4962–4974, Oct 2013.

[15] J. I. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti, "Achieving single channel, full duplex wireless communication," in *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking*, MobiCom '10, (New York, NY, USA), pp. 1–12, ACM, 2010.

[16] Z. Zhang, K. Long, A. V. Vasilakos, and L. Hanzo, "Full-duplex wireless communications: Challenges, solutions, and future research directions," *Proceedings of the IEEE*, vol. 104, pp. 1369–1409, July 2016.

[17] A. Sahai, G. Patel, C. Dick, and A. Sabharwal, "On the impact of phase noise on active cancelation in wireless full-duplex," *IEEE Transactions on Vehicular Technology*, vol. 62, pp. 4494–4510, Nov 2013.

[18] M. Duarte, C. Dick, and A. Sabharwal, "Experiment-driven characterization of full-duplex wireless systems," *IEEE Transactions on Wireless Communications*, vol. 11, pp. 4296–4307, December 2012.

[19] André Silva, "Git repository of oot blocks." `https://github.com/andrehoracio97/investigacao/tree/master/OOT`. [Online; Accessed: 13/02/2020].

[20] W. I. Forum. `https://www.wirelessinnovation.org`, 2019. [Online; Accessed: 16/10/2019].

[21] W. I. Forum, "What is sdr?." `https://www.wirelessinnovation.org/Introduction_to_SDR`, 2019. [Online; Accessed: 16/10/2019].

[22] CISION PR Newswire, "Global software defined radio market 2018-2022: Increasing demand for sdr in homeland security, industrial, and commercial applications." `https://www.prnewswire.com/news-releases/global-software-defined-radio-market-2018-2022-increasing-demand-for-sdr-in-homeland-security-industrial-and-commercial-applications-300613180.html`, 2018. [Online; Accessed: 12/02/2020].

[23] ZTE, "Zte's sdr element management system." `https://www.zte.com.cn/global/about/magazine/zte-technologies/2010/7/en_488/187391.html`, 2010. [Online; Accessed: 12/02/2020].

[24] Huawei, "Huawei launches singleran pro to tackle 5g challenges and create an intelligent digital world." `https://www.huawei.com/en/press-events/news/2018/4/Huawei-SingleRAN-Pro-Solution`, 2018. [Online; Accessed: 12/02/2020].

[25] EENewsAutomotive, "Software-defined radio opens up new chances for the automobile industry." `https://www.eenewsautomotive.com/content/software-defined-radio-opens-new-chances-automobile-industry`, 2011. [Online; Accessed: 12/02/2020].

[26] T. F. Collins, R. Getz, D. Pu, and A. M. Wyglinski, *Software-Defined Radio for Engineers.* Artech House Publishers, 2018.

[27] R. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, pp. 21–28, 1962.

[28] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[29] MatWorks, "MATLAB." `https://www.mathworks.com/products/matlab.html?s_tid=hp_products_matlab`, 2019. [Online; Accessed: 16/10/2019].

[30] MatWorks, “Simulink.” https://www.mathworks.com/products/simulink.html, 2019. [Online; Accessed: 16/10/2019].

[31] N. Instruments, “LabView.” https://www.ni.com/en-us/shop/labview.html, 2019. [Online; Accessed: 16/10/2019].

[32] G. Companion, “GNURadio.” https://www.gnuradio.org/about/, 2019. [Online; Accessed: 16/10/2019].

[33] Saruch Rathore, “Graphical programming language labview & xcos, gnu radio or blockly as an open source alternative to labview for data acquisition and control - indian institute of technology - bombay.” https://pdfs.semanticscholar.org/fd4f/cb10e56426b93e063fad8443d92083fa487d.pdf. [Online; Accessed: 12/02/2020].

[34] Neelabhro, “Psk-modulation-of-audio-signal-in-matlab-gnu-radio-labview.” https://github.com/neelabhro/PSK-modulation-of-Audio-Signal-in-MATLAB-GNU-Radio-LabVIEW. [Online; Accessed: 12/02/2020].

[35] Ettus, “USRP HD.” https://files.ettus.com/manual/, 2019. [Online; Accessed: 16/10/2019].

[36] E. Research, “USRP B210 Kit.” https://www.ettus.com/all-products/ub210-kit/, 2019. [Online; Accessed: 16/10/2019].

[37] E. Research, “VERT 2450 Antenna.” https://www.ettus.com/all-products/vert2450/, 2019. [Online; Accessed: 16/10/2019].

[38] F. J. Harris, S. Member, M. Rice, and S. Member, “Multirate digital filters for symbol timing synchronization in software defined radios,” *IEEE Journal on Select Areas in Communications*, vol. 19, pp. 2346–2357, 2001.

[39] F. Harris, “Band edge filters: Characteristics and performance in carrier and symbol synchronization,” *The 13 th International Symposium on Wireless Personal Multimedia Communications*, 2010.

[40] G. Companion, “FLL Band-Edge.” https://wiki.gnuradio.org/index.php/FLL_Band-Edge, 2019. [Online; Accessed: 20/01/2020].

[41] J. C. Richard Johnson, W. A. Sethares, and A. G. Klein, *Software Receiver Design - Build Your Own Communication System in Five Easy Steps.* Cambridge, 2011.

[42] J. Feigin, “Practical costas loop design: Designing a simple and inexpensive BPSK costas loop carrier recovery circuit,” *RF signal processing*, pp. 22–36, 2002.

[43] R. T. R, N. R, S. Joy, D. A. J, and Alex.V, “Design and implementation of digital costas loop and bit synchronizer in FPGA for BPSK demodulation,” *International Conference on Control Communication and Computing*, 2013.

[44] P. Borwein and M. J. Mossinghoff, *Barker sequences and flat polynomials*, p. 71–88. London Mathematical Society Lecture Note Series, Cambridge University Press, 2008.

[45] GNU Radio Wiki, “Differential encoder.” https://wiki.gnuradio.org/index.php/Differential_Encoder, 2019. [Online; Accessed: 16/10/2019].

[46] S. Johnson, “Introducing low-density parity-check codes,” *School of Electrical Engineering and Computer Science - University of Newcastle, Australia*, 05 2010.

[47] D. J. C. MacKay and R. M. Neal, "Near shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 33, no. 6, pp. 457–458, 1997.

[48] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, pp. 399–431, March 1999.

[49] L. Lolis, "Universidade federal do paraná - campos finitos." `https://professorluislolis.weebly.com/uploads/1/3/2/7/13273601/2-campos_finitos.pdf`. [Online; Accessed: 19/05/2020].

[50] P. Sweeney, *Error control coding: from theory to practice.* Wiley, 2002.

[51] S. Lin and D. J. Costello, *Error Control Coding, Second Edition.* USA: Prentice-Hall, Inc., 2004.

[52] A. B. Carlson and P. B. Crilly, *Communications Systems - An Introduction to Signals and Noise in Electrical Communication.* McGraw-Hill, 5° ed., 2010.

[53] U. of British Columbia Course ELEX: Data Communications Lesson 13, "Pn sequences and scramblers." `http://www.ece.ubc.ca/~edc/3525.jan2014/lectures/lec13.pdf`, 2014. [Online; Accessed: 16/10/2019].

[54] I. T. U. T. S. Sector, *Series V: Data Communication Over the Telephone Network.* International Telecommunication Union, 1998.

[55] E. T. S. Institute, *Digital Video Broadcasting: Support for use of scrambling and Conditional Access (CA) within digital broadcasting systems.* European Telecommunications Standards Institute, Oct. 1996.

[56] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, 1978.

[57] I. Synopsys, "The heartbleed bug." `http://heartbleed.com/`, 2019. [Online; Accessed: 16/10/2019].

[58] Z. Dryer, A. Nickerl, M. A. C. Gomes, J. P. Vilela, and W. K. Harrison, "Full-duplex jamming for enhanced hidden-key secrecy," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pp. 1–7, May 2019.

[59] J. Almeida and J. Barros, "Random puncturing for secrecy," in *2013 Asilomar Conference on Signals, Systems and Computers*, pp. 303–307, Nov 2013.

[60] D. Sarmento, J. Vilela, W. K. Harrison, and M. Gomes, "Interleaved coding for secrecy with a hidden key," in *2015 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, 2015.

[61] C. Martins, T. Fernandes, M. Gomes, and J. Vilela, "Testbed implementation and evaluation of interleaved and scrambled coding for physical-layer security," in *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pp. 1–6, June 2018.

[62] C. Martins, "Implementção em plataformas SDR de esquemas de codificação para segurança na camada física baseados em códigos curtos e técnicas de interleaving," Master's thesis, University of Coimbra, 07 2017.

[63] A. Mukherjee, S. A. A. Fakoorian, J. Huang, and A. L. Swindlehurst, "Principles of physical layer security in multiuser wireless networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1550–1573, 2014.

[64] E. Tekin and A. Yener, “The gaussian multiple access wire-tap channel: wireless secrecy and cooperative jamming,” in *2007 Information Theory and Applications Workshop*, pp. 404–413, Jan 2007.

[65] E. Tekin and A. Yener, “The general gaussian multiple-access and two-way wiretap channels: Achievable rates and cooperative jamming,” *IEEE Transactions on Information Theory*, vol. 54, pp. 2735–2751, June 2008.

[66] J. P. Vilela, M. Bloch, J. Barros, and S. W. McLaughlin, “Wireless secrecy regions with friendly jamming,” *IEEE Transactions on Information Forensics and Security*, vol. 6, pp. 256–266, June 2011.

[67] R. Bassily, E. Ekrem, X. He, E. Tekin, J. Xie, M. R. Bloch, S. Ulukus, and A. Yener, “Cooperative security at the physical layer: A summary of recent advances,” *IEEE Signal Processing Magazine*, vol. 30, pp. 16–28, Sep. 2013.

[68] X. He and A. Yener, “Secure degrees of freedom for gaussian channels with interference: Structured codes outperform gaussian signaling,” in *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, pp. 1–6, Nov 2009.

[69] J. Xie and S. Ulukus, “Secure degrees of freedom of the gaussian wiretap channel with helpers and no eavesdropper CSI: blind cooperative jamming,” *CoRR*, vol. abs/1302.6570, 2013.

[70] T. Riihonen, S. Werner, and R. Wichman, “Mitigation of loopback self-interference in full-duplex MIMO relays,” *IEEE Transactions on Signal Processing*, vol. 59, pp. 5983–5993, Dec 2011.

[71] T. Riihonen, S. Werner, and R. Wichman, “Hybrid full-duplex/half-duplex relaying with transmit power adaptation,” *IEEE Transactions on Wireless Communications*, vol. 10, pp. 3074–3085, Sep. 2011.

[72] 3GP, “TSG RAN WG1 Meeting R1-100139 - Full duplex configuration of Un and Uu subframes for type I relay.” `https://www.3gpp.org/DynaReport/TDocExMtg--R1-59b--28028.html`, Jan. 2010. [Online; Accessed: 16/10/2019].

[73] G. Chen, Y. Gong, P. Xiao, and J. A. Chambers, “Physical layer network security in the full-duplex relay system,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 574–583, 2015.

[74] S. Hong, J. Brand, J. I. Choi, M. Jain, J. Mehlman, S. Katti, and P. Levis, “Applications of self-interference cancellation in 5g and beyond,” *IEEE Communications Magazine*, vol. 52, pp. 114–121, February 2014.

[75] M. Duarte and A. Sabharwal, “Full-duplex wireless communications using off-the-shelf radios: Feasibility and first results,” in *2010 Conference Record of the Forty Fourth Asilomar Conference on Signals, Systems and Computers*, pp. 1558–1562, Nov 2010.

[76] E. Everett, M. Duarte, C. Dick, and A. Sabharwal, “Empowering full-duplex wireless communication by exploiting directional diversity,” in *2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pp. 2002–2006, Nov 2011.

[77] C. R. Anderson, S. Krishnamoorthy, C. G. Ranson, T. J. Lemon, W. G. Newhall, T. Kummetz, and J. H. Reed, “Antenna isolation, wideband multipath propagation measurements, and interference mitigation for on-frequency repeaters,” in *IEEE SoutheastCon, 2004. Proceedings.*, pp. 110–114, 2004.

[78] K. Haneda, E. Kahra, S. Wyne, C. Icheln, and P. Vainikainen, "Measurement of loop-back interference channels for outdoor-to-indoor full-duplex radio relays," in *Proceedings of the Fourth European Conference on Antennas and Propagation*, pp. 1–5, April 2010.

[79] M. Jain, J. I. Choi, T. Kim, D. Bharadia, S. Seth, K. Srinivasan, P. Levis, S. Katti, and P. Sinha, "Practical, real-time, full duplex wireless," in *MobiCom '11*, 2011.

[80] S. Armas-Jimenez and J. Sanchez-Garcia, "An in band full duplex practical implementation applying passive self-interference cancellation," *Proceedings of the GNU Radio Conference*, vol. 2, no. 1, p. 5, 2017.

[81] A. P. by The Aerospace Corporation, "GNU Radio conference of 2017 - a GNU radio-based full duplex radio system," 2017.

[82] N. Li, W. Zhu, and H. Han, "Digital interference cancellation in single channel, full duplex wireless communication," in *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–4, Sep. 2012.

[83] G. Velarde, "Department of electrical engineering stanford university - wiener filtering (lecture 12)." `https://web.stanford.edu/class/archive/ee/ee264/ee264.1072/mylecture12.pdf`. [Online; Accessed: 08/01/2020].

[84] J. Apolinario, *QRD-RLS Adaptive Filtering*. Lecture notes in mathematics, Springer US, 2009.

[85] F. S. Perdigão, "Processos estocásticos." Departamento de Engenharia Electrotécnica e de Computadores da Universidade de Coimbra - Apontamentos de Processamento Digital de Sinal, 2007.

[86] U. of Wisconsin, "University of wisconsin - computer aided engineering: Least mean square algorithm." `http://homepages.cae.wisc.edu/~ece539/resources/tutorials/LMS.pdf`. [Online; Accessed: 08/01/2020].

[87] S. Haykin, *Adaptive Filter Theory*. Prentice-Hall information and system sciences series, Prentice Hall, 2002.

[88] N. Instruments, "Least mean squares (lms) algorithms (adaptive filter toolkit)." `http://zone.ni.com/reference/en-XX/help/372357B-01/lvaftconcepts/aft_lms_algorithms/`. [Online; Accessed: 21/05/2020].

[89] N. Instruments, "Recursive least squares (rls) algorithms (adaptive filter toolkit)." `http://zone.ni.com/reference/en-XX/help/372357B-01/lvaftconcepts/aft_rls_algorithms/`. [Online; Accessed: 21/05/2020].

[90] G. Companion, "FEC API." `https://www.gnuradio.org/doc/doxygen/page_fec.html`. [Online; Accessed: 26/05/2020].

[91] GNU Radio Wiki, "Polyphase clock sync." `https://wiki.gnuradio.org/index.php/Polyphase_Clock_Sync`, 2019. [Online; Accessed: 2/11/2019].

[92] B. S. E. Research, "GNU Radio Tutorials." `https://files.ettus.com/tutorials/labs/Lab_1-5.pdf`. [Online; Accessed: 27/05/2020].

[93] Karel, "GR-Adapt OOT Module." `https://github.com/karel/gr-adapt`. [Online; Accessed: 27/05/2020].

[94] OpenCourseWare, "Correlation." https://ocw.tudelft.nl/wp-content/uploads/Intro_reflection_seismics_Appendix_B._Correlation.pdf. [Online; Accessed: 18/06/2020].

[95] S. R. U. of Adelaide, "The Error Correcting Codes (ECC) Page." http://eccpage.com/. [Online; Accessed: 29/05/2020].

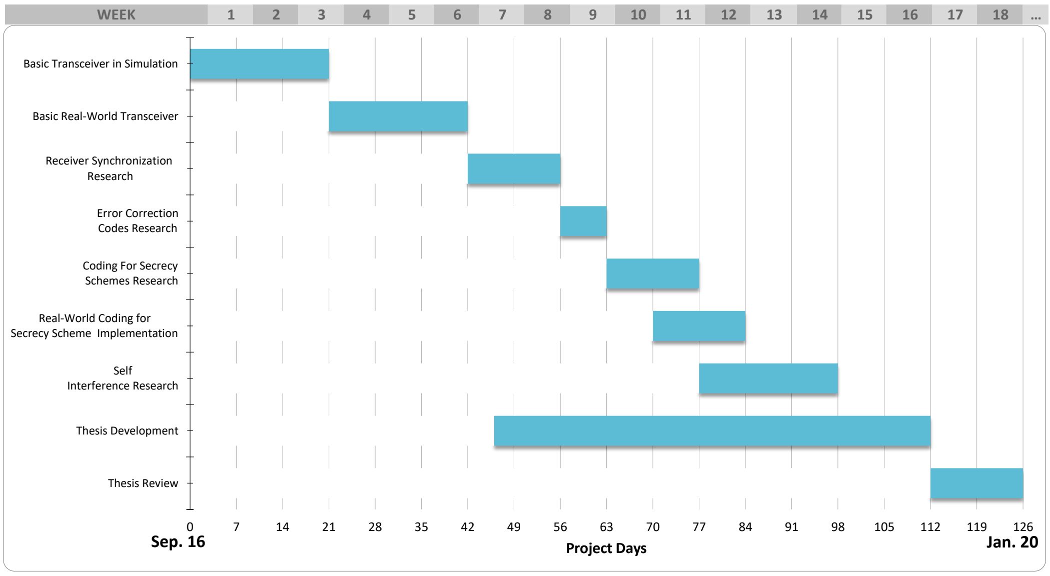
[96] D. Moore, G. McCabe, and B. Craig, *Introduction to the Practice of Statistics*. W.H. Freeman, 2009.

This page is intentionally left blank.

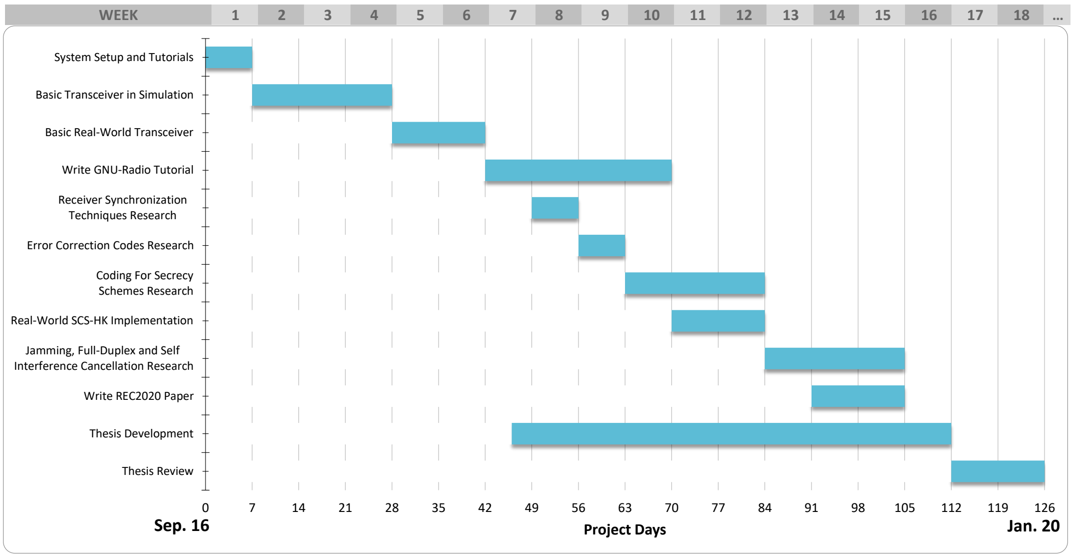
Appendices

This page is intentionally left blank.

Appendix A

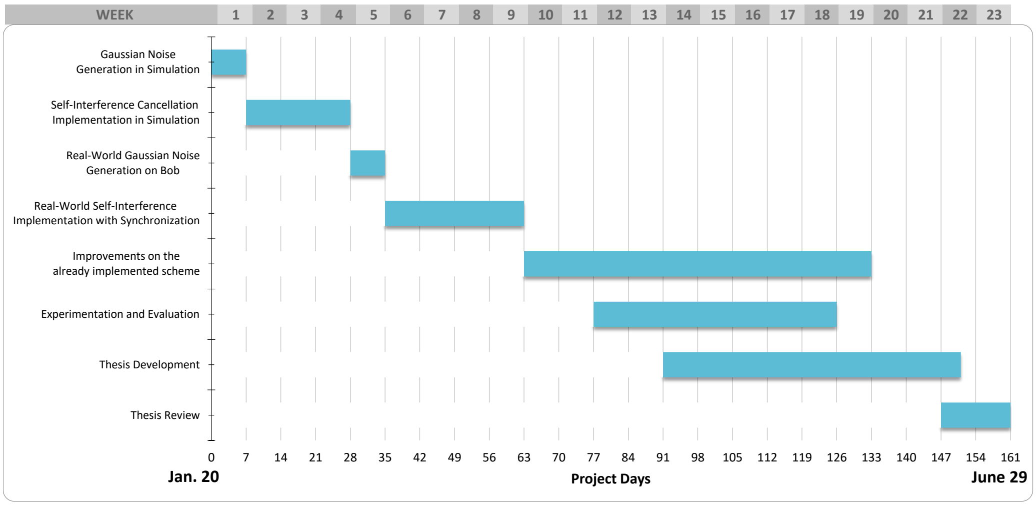


(a) Planned Gantt diagram for first semester.

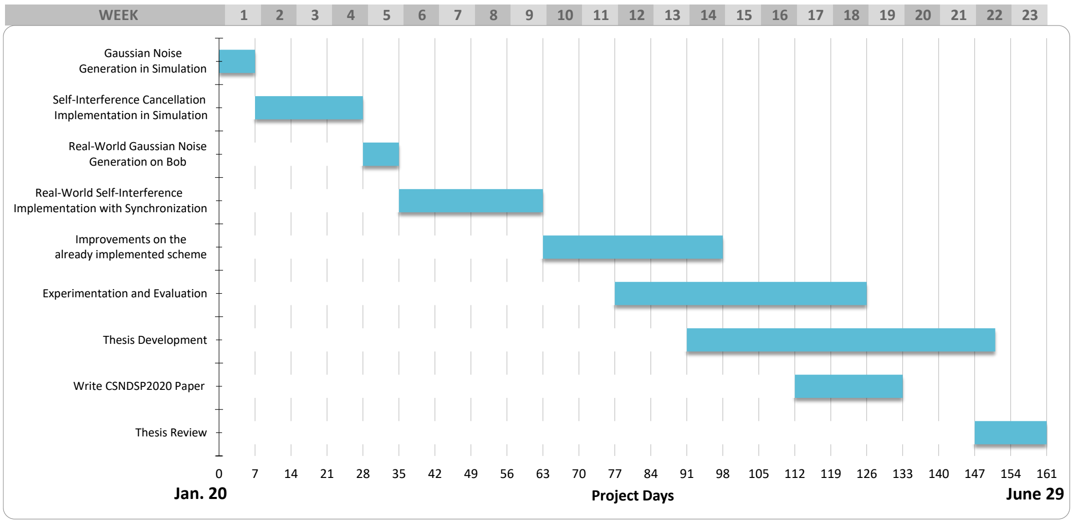


(b) Actual Gantt diagram for first semester.

Figure 1: A Gantt diagram showing the start and finish dates of the planned tasks for the first semester on Fig. 1(a) and the actual fulfilled on Fig. 1(b).



(a) Planned Gantt diagram for second semester.



(b) Actual Gantt diagram for second semester.

Figure 2: A Gantt diagram showing the start and finish dates of the planned tasks for the second semester on Fig. 2(a) and the actual fulfilled on Fig. 2(b).

Appendix B

SDR Testbed of Full-Duplex Jamming for Secrecy

André Silva[†‡], Marco A. C. Gomes[‡] João P. Vilela[*], Willie K. Harrison[§]
[†]University of Coimbra, CISUC, Department of Informatics Engineering, Portugal.
[‡]University of Coimbra, Instituto de Telecomunicações, Department of Electrical and Computer Engineering, Portugal.
[*]CRACS/INESCTEC, Departamento de Ciência de Computadores, Faculdade de Ciências, Universidade do Porto,
Rua do Campo Alegre s/n, 4169–007 Porto, Portugal.
[§]Department of Electrical and Computer Engineering, Brigham Young University, UT, USA.
Emails: uc2015228086@student.uc.pt, marco@co.it.pt, jvilela@fc.up.pt, willie.harrison@byu.edu

***Abstract*—In order to secure wireless communications, we consider the usage of physical-layer security (PLS) mechanisms (i.e. coding for secrecy mechanisms) combined with self-interference generation. We present a prototype implementation of a scrambled coding for secrecy mechanism with interference generation by the legitimate receiver and the cancellation of the effect of self-interference (SI). Regarding the SI cancellation, two algorithms were evaluated: least mean square and recursive least squares. The prototype implementation is performed in real-world software-defined radio (SDR) devices using GNU-Radio.**

***Index Terms*—Software-defined radio, GNU-Radio, Physical-layer security, Jamming, Full-duplex, Self-interference cancellation**

I. Introduction

Wireless communications are broadcast in nature. It is therefore difficult to restrict the communication to an intended receiver (Bob) while granting secrecy against an illegitimate one (Eve). Although modern cryptography helps to ensure secrecy in communication, it settles on the premise of the computational infeasibility of breaking cryptographic schemes. Information theoretically secure coding schemes, unfortunately, are currently unknown for real-wold channels.

That said, in the past few years there have been efforts for PLS to act as a complement of the modern cryptographic schemes. Wyner's wiretap channel [1] considers a three-party communication setup (Alice, Bob and Eve) where the wiretap channel (Alice and Eve channel) is degraded relative to the main channel (Alice and Bob channel). Also, Wyner showed that it is possible to build wiretap codes to achieve reliable communication to Bob and secrecy against Eve. More practical coding schemes have been proposed, for instance, Klinc [2] took advantage of the low-density parity-check (LDPC) code characteristics along with puncturing to create a coding scheme to transmit a coded message with punctured bits. In order to not directly expose the transmitted information, Baldi [3] proposed the use of scrambling/interleaving techniques [3], [4]. Following these works, a coding for secrecy scheme was recently proposed called interleaved/scrambled coding for secrecy with a hidden key (ICS-HK / SCS-HK) [5], [6].

Eve's channel disadvantage with respect to Bob is difficult to assure at all times, thus leading to the development of cooperative jamming schemes where there is one helper transmitting and degrading Eve's channel. However, this type of scheme has some drawbacks like synchronization, willingness of cooperation by the helper and the need of a trusted relationship. This work addresses these drawbacks through a solution in which Bob (other than a third party) acts himself as a jammer, while transmitting in the same frequency in which he is receiving (i.e. full-duplex), therefore suffering from self-interference (SI) upon reception [7]. A SDR testbed implementation, using Ettus Universal Software Radio Peripheral (USRP) B210 devices, is presented as a proof of concept where the adapted SCS-HK coding scheme is combined with full-duplex jamming. Furthermore, two algorithms of SI cancellation will be evaluated.

The remainder of the paper is organized as follows. In Section II, concepts of PLS are introduced along with the mechanisms used on the prototype implementation. In Section III the setup is explained and results presented, and Section IV concludes the paper.

II. Physical Layer Security

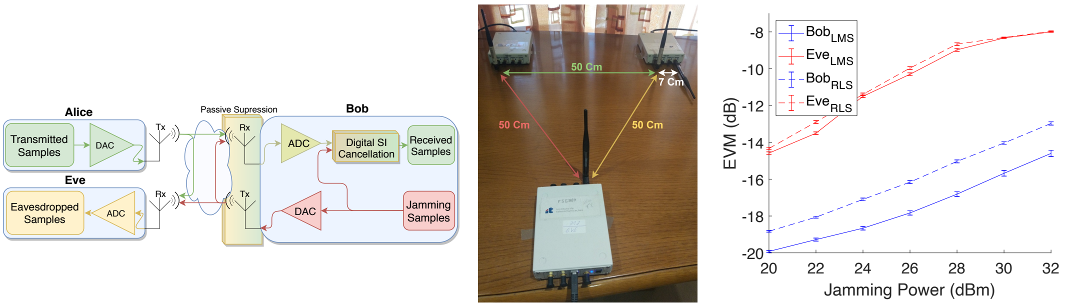
A. SCS-HK Adapted Scheme

The ICS-HK and SCS-HK coding schemes [5], [6] rely on scrambling/interleaving the information with a random key, followed by encoding them together with an LDPC code. Some of the bits of the resultant codeword are punctured before the transmission to guarantee that only the legitimate receiver having a better channel can correctly recover the message. The SCS-HK scheme [5] was adapted to use in this prototype. The modifications were mainly in the type of scrambler used (additive rather than multiplicative) and how a packet is found (a simple verification after the header being decoded through a Reed-Solomon code).

B. Jamming and Cooperative Jamming

In the coding for secrecy schemes it is necessary for Bob to have some advantage over Eve, often an uncontrollable factor. Cooperative jamming [8] was considered to overcome this challenge to help lower the signal to noise ratio (SNR) of the wiretap channel. It consists in having an external jammer to help Alice to communicate with Bob using several known schemes, namely cooperative jamming by Gaussian noise, by structured codes and by alignment.

This work was funded by the European Regional Development Fund (FEDER), through the Regional Operational Programme of Lisbon under Grant POR LISBOA 2020, by the Competitiveness and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 framework [Project 5G with Nr. 024539 under Grant POCI-01-0247-FEDER-024539], and by FCT/MCTES through national funds and when applicable co-funded EU funds under the projects UIDB/EEA/50008/2020, and PES3N (SAICT-45-2017-POCI-01-0145-FEDER-030629).



(a) (b) (c)

Fig. 1. (a) Conceptual Scheme of SI Cancellation of this work; (b) SDR setup; (c) Jamming Power and EVM comparison of Bob and Eve on LMS and RLS with 95% confidence intervals.

All these schemes have some disadvantages, for instance: 1) the synchronization, that is, the helper needs to be in sync with Alice to correctly help Bob to have the necessary advantage; 2) the willingness of the helper to spend his own energy in order to emit the jamming signal to make other communications secure; and 3) there has to be a trusted relationship between the helper and Bob, because if it stops jamming, then Eve can retrieve the signal. Using Bob as a jammer can overcome most of these issues since it helps to solve the synchronization problem, the issue of the need for cooperation and it facilitates the removal of interference (the receiving device knows the characteristics of the interference being generated).

C. Full-Duplex and Self-Interference

Full-duplex (FD) enables simultaneous uplink and downlink communication in the same frequency at the same time [9] which is useful in this context: Bob receives the transmitted signal and at the same time emits the jamming signal. The main drawback in FD is the self-interference that is generated due to the signal created on the transmitter antenna being fed back into the receiver antenna of the same device. On top of that, the SI signal has higher gain than the intended one. Thus, Bob will not only receive the information signal but also receive the jamming signal (with higher gain) that is generated resulting in a lower signal to interference and noise ratio (SINR) of the channel [10].

Furthermore, to successfully retrieve the signal of interest it is required to reduce the SI signal to decode it correctly, so the goal of SI cancellation is to predict and model the distortions and compensate for them at the receiver antennas.

D. Self-Interference Cancellation

There are three different categories of SI cancellation [9], namely passive suppression, analog cancellation and finally digital cancellation. *Passive suppression* is based on attenuating the SI signal by physically separating somehow the transmitter antenna and the receiver antenna of the device. The *analog SI cancellation* is performed before the signal goes into the analog-digital converter (ADC) (e.g. by using a noise canceling integrated circuit, which is not possible in our prototype). The *digital SI cancellation* mechanism works after the signal goes into the ADC and takes advantage of the knowledge of the interfering signal for the purpose of canceling it. There are adaptive algorithms like the least mean square (LMS) and recursive least squares (RLS) which iteratively adjust the coefficients of the signal in a way to predict the desired one. This will be the focus of this work: explore the usage of the adapted SCS-HK scheme (Section II-A) along with interference generation (Section II-B) by the receiver and its cancellation using methods of Section II-C (using either LMS or RLS). The conceptual scheme of the implemented prototype is represented in Fig. 1(a).

III. Setup and Results

To assess reception quality we will measure the error vector magnitude (EVM), i.e. the distance between the received constellation symbols and the ideal ones. A smaller EVM value, means better approximation to the constellation points, thus, better possibilities to correctly retrieve the information.

In this work, the usage of both LMS and RLS algorithms will be evaluated in order to cancel the SI signal. To be as fair as possible, the SDRs are evenly separated by 50 centimeters, as indicated in Fig. 1(b). It is important to note that Bob's transmit antenna is laying down for better SI cancellation (i.e. using passive suppression as mentioned in Section II-C). For better final results, 30 tests for each jamming power were performed and the mean and 95% confidence interval were calculated on the EVM data. In all these tests, Alice's power is fixed at 20 dBm and Bob's jamming power is varying between 20 dBm to 32 dBm (with a 2 dBm step).

All the corresponding EVM (of Bob and Eve with both algorithms) and jamming power values are presented in Fig. 1(c). As it is possible to analyze, the LMS algorithm has lower EVM values than the RLS, thus, the first outperforms the latter on the SI cancellation enabling better recovering of Alice's signal. Furthermore, with better recovery it is possible to increase the jamming power for the purpose of degrading as much as possible Eve's channel with minimal effect on Bob.

IV. Conclusion

In this paper, we presented an SDR prototype implementation in GNU Radio of the SCS-HK coding for secrecy scheme with interference generation by the receiver and its cancellation using full-duplex mechanisms. The results show a clear advantage for Bob with respect to an eavesdropper due to the ability to execute self interference cancellation. Also, it was shown that the LMS outperforms the RLS algorithm for the same environment.

References

[1] A. D. Wyner. The wire-tap channel. *The Bell System Technical Journal*, 54(8):1355–1387, 1975.

[2] D. Klinc, J. Ha, S. W. McLaughlin, J. Barros, and B. Kwak. LDPC codes for the gaussian wiretap channel. *IEEE Trans. on Inform. Forensics and Security*, 6(3):532–540, 2011.

[3] M. Baldi, M. Bianchi, and F. Chiaraluce. Coding with scrambling, concatenation, and HARQ for the AWGN wire-tap channel: A security gap analysis. *IEEE Trans. on Inf. Forens. and Secur.*, 7:883–894, 2012.

[4] J. Vilela, M. Gomes, W. K. Harrison, D. Sarmento, and F. Dias. Interleaved concatenated coding for secrecy in the finite blocklength regime. *IEEE Signal Processing Letters*, 23(3):356–360, 2016.

[5] D. Sarmento, J. Vilela, W. K. Harrison, and M. Gomes. Interleaved coding for secrecy with a hidden key. In *2015 IEEE Globecom Workshops*, pages 1–6, 2015.

[6] C. Martins, T. Fernandes, M. Gomes, and J. Vilela. Testbed implementation and evaluation of interleaved and scrambled coding for physical-layer security. In *2018 IEEE 87th VTC - Spring*, pages 1–6, June 2018.

[7] Z. Dryer, A. Nickerl, M. Gomes, J. Vilela, and W. K. Harrison. Full-duplex jamming for enhanced hidden-key secrecy. In *IEEE ICC*, pages 1–7, May 2019.

[8] R. Bassily, E. Ekrem, X. He, E. Tekin, J. Xie, M. R. Bloch, S. Ulukus, and A. Yener. Cooperative security at the physical layer: A summary of recent advances. *IEEE Signal Processing Magazine*, 30(5):16–28, 2013.

[9] Z. Zhang, K. Long, A. V. Vasilakos, and L. Hanzo. Full-duplex wireless communications: Challenges, solutions, and future research directions. *Proceedings of the IEEE*, 104(7):1369–1409, July 2016.

[10] J. Vilela, M. Bloch, J. Barros, and S. W. McLaughlin. Wireless secrecy regions with friendly jamming. *IEEE Transactions on Information Forensics and Security*, 6(2):256–266, 2011.

This page is intentionally left blank.

Appendix C

SDR Implementation of Full-Duplex Jamming for Secrecy

André Silva[‡], Marco A. C. Gomes[‡] João P. Vilela[*]
[‡]Instituto de Telecomunicações, Department of Electrical and Computer Engineering,
University of Coimbra, Coimbra, Portugal. Email: uc2015228086@student.uc.pt, marco@co.it.pt
[*]CISUC and Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal.
Email: jpvilela@dei.uc.pt

***Abstract*—We consider secure wireless communications via physical-layer security mechanisms, particularly coding for secrecy mechanisms, combined with self-interference generation. We present results of a prototype implementation of a scrambled coding for secrecy mechanism with self-interference generation by the legitimate receiver. The prototype implementation is performed in real-world software-defined radio devices through a GNU Radio implementation. Our preliminary results shows a better error vector magnitude value on the legitimate receiver due the self-interference cancellation while Eve gets a more disperse constellation point values, leading to a worst decoding results.**

I. Introduction

Secrecy against illegitimate actors always been a concern mainly with the growth of wireless devices, since it is not possible to direct a transmission to a given single receptor. Though modern cryptography [1] helps to ensure secrecy in a communication it settles on the premise of "computer power infeasible" to rely on the impossibility of breaking some cryptographic scheme. The information-theoretically secure schemes are unbreakable, however, are unpractical due its constraints.

That said, in the past few years there have been efforts on physical layer security to act as a complement of the already existing and implemented modern cryptographic schemes in order to improve security of a communication. This research effort started with Shannon's work that introduced the notion of perfect secrecy [2], stating that to achieve the maximum secrecy the mutual information between a message and the cipher text must be zero. Several works have followed, with most of them being developed for the Wyner's wiretap channel [3] that considers a three party communication scheme (Alice, Bob and Eve) where the wiretap channel is the channel between Alice and Eve, which is degraded comparatively to the main channel between Alice and Bob. Wyner showed that it is possible to build wiretap codes to achieve reliable communication to Bob and secrecy against Eve, and he also introduced a relaxed version of perfect secrecy called "weak secrecy", that takes into consideration the information leakage to Eve.

In face of the difficulty of using in practice Wyner's wiretap codes and information theoretic metrics, more practical coding schemes and applied metrics such as Security Gap (S_G) [4], have been recently proposed. Klinc's work [4] took advantage of the Low-Density Parity-check Code (LDPC) characteristics along with puncturing to create a coding scheme to transmit a coded message with punctured bits leading to a small S_G. In order not to directly expose the transmitted information and at the same time reduce the S_G Baldi [5] proposed the use of scrambling/interleaving techniques [6], [5]. Following this work, recently it was proposed a coding for secrecy scheme called Interleaved/Scrambled Coding for Secrecy with a Hidden Key (ICS-HK and SCS-HK) [6], [7], [8] which relies on scrambling/interleaving the information with a random key, followed by encoding them together with an error correction code. Some of the bits of the resulted codeword are erased before the transmission to guarantee that only the legitimate receiver having a better channel can correctly recover the message.

The advantage of the legitimate receiver over illegitimate one is difficult to assure all times, leading to the emergence of cooperative jamming schemes, where another transmitter helps to degrade the eavesdropper's channel. However, this type of schemes has some drawbacks like synchronization, willingness of cooperation by the helper since he has to spend his own energy to make others secure and also can degrade the legitimate receiver's channel. In this work, we address these drawbacks through a solution in which the receiving device (other than a third party) acts itself as a jammer, while transmitting at the same frequency as it is receiving (i.e. Full-Duplex), therefore suffering from self-interference upon reception. This helps to solve the synchronization problem and the issue of the need for cooperation. Besides, it also facilitates the removal of interference, because the receiving device knows the characteristics of the interference being generated. A (Software-Defined Radio) SDR testbed implementation is presented as proof of concept where the SCS-HK coding scheme is combined with the full-duplex jamming.

The remainder of the paper is organized as follows. In Section II, background concepts of physical-layer security are introduced. In Section III a prototype implementation of our proposed scheme is presented, along with preliminary results in Section IV. Section V concludes the paper.

II. Physical Layer Security

A. SCS-HK Adapted Scheme

The SCS-HK adapted scheme is an adaptation of the original scheme [8], with modifications on the type of scrambler used (additive rather than multiplicative), the forward error correcting code, and how a packet is found. The architecture is represented in Fig. 1. In the transmission side, firstly, the desired binary data for transmission M is scrambled using the additive scrambling (because it is not self-synchronizing, thus is required to have the correct key to descramble or all the frame will be lost) with a random key K resulting in $S[M]_K$. This scrambled data is then appended to K resulting in $K + S[M]_K$ and then encoded using some Forward Error Correction (Convolution Code or LDPC) $E\{K + S[M]_K\}$ which results in the payload's packet, so now it is necessary to use an header H containing some known sequence to be recognized on receptor side in a manner that it knows that is a new payload incoming resulting in a packet built by $P = H + E$. To finalize the transmitter side it is required to modulate the data in a Quadrature Phase-Shift Keying (QPSK) and then do pulse shaping by applying the Root Raised Cosine (RRC) filter, and finally transmit over the SDR.

In the reception side it is the reverse operation where after the signal is received in the SDR first it is required do pulse shaping by applying RRC filter, then do synchronization, which includes doing timing offset recovery, carrier recover and multipath correction. After the synchronization the binary packet $\widehat{P}$ is continuous, so the packet detection looks for the known sequence added in the header $\widehat{H}$ and then extracts the payload with is encoded $\widehat{E}$. The payload is then decoded, and it is recovered the key $\widehat{K}$ that will be used for descramble getting finally the message $\widehat{M}$ which can be exactly the same or with some differences depending on the channel interference.

B. Jamming and Cooperative Jamming

In the coding for secrecy schemes it is necessary for Bob to have some advantage over Eve. This advantage can come from the quality channel itself, hence not controllable. Jamming can be considered [9] to overcome this challenge to help lowering the SNR of the wiretap channel, being called cooperative jamming [10]. Cooperative jamming consists in having an external jammer to help Alice to communicate with Bob using several known schemes, namely cooperative jamming by Gaussian noise [9], [10], by structured codes [11] and by alignment [12].

All these schemes have some disadvantages, for instance the synchronization, this is, the helper needs to be in sync with Alice to correctly help Bob having the advantage and only spending the necessary amount of energy needed to that effect, leading us to another disadvantage which is the willingness of the helper to spend his own

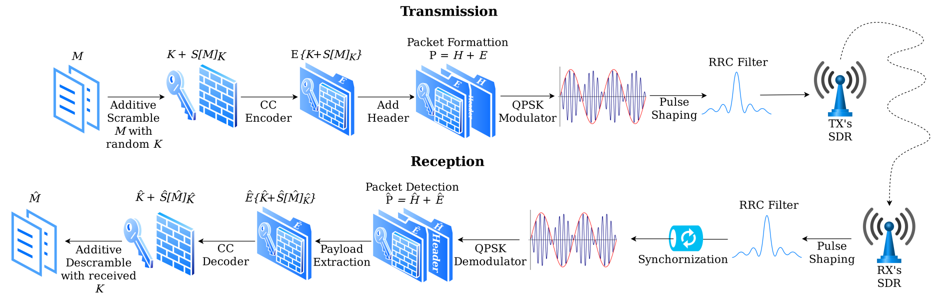


Fig. 1. Adapted SCS-HK architecture.

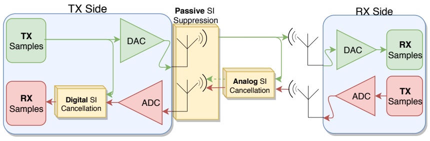


Fig. 2. Active and passive self-interference cancellation mechanisms.

energy in order to emit the jam signal to make other communications secure. Also there has to be a trust relationship between the helper and Bob, because if it stops to jam, then Eve can retrieve the signal.

Using the legitimate receiver as a jammer can overcome most of these issues. This will be the focus of this work: explore the usage of the previously discussed security schemes (more concretely the SCS-HK scheme), along with interference generation by the receiver and its cancellation.

C. *Full-Duplex and Self-Interference*

Full-duplex (FD) aims to enable simultaneous uplink and downlink communication in the same frequency [13]. This can be useful for Bob to received the transmitted signal and at the same time emit the jamming signal, being possible in the Universal Software Radio Peripheral (USRP) B210 which supports both MIMO and FD.

The application of FD along with jamming in order to improve security in communications was recently researched in [14], [15], where the receiver also acts as jammer in a way to prevent Eve to decode the interested signal.

FD carries some challenges [13] when compared to the HD mode, most notably the impossibility to attain the theoretical double gain compared with HD is the self interference (SI) that is generated on the transmission antenna and then received on the receiver antenna of the same device, thus, Bob will not only receive the information signal but also receive the jamming signal that is generating causing superposition on the information suffering of Signal-to-Interference-and-Noise-Ratio (SINR). On top of that, the USRP needs to consume values in $[-1, 1]$, but as the transmitter signal has higher gain relatively to receiver interesting signal, then the most values will be noise and the interesting signal will be smaller values, hence resulting in a lower SINR in the channel [16]. Furthermore, to successfully retrieve the signal of interest it is required to reduce the SI signal to decode it correctly, so the goal of SI cancellation is to predict and model the distortions and compensate for them at the receiver antennas, as indicated on [17].

D. *Self-Interference Cancellation*

There are three different categories of SI cancellation [13], namely passive suppression, analog cancellation and finally digital cancellation as visualized in Fig. 2. Passive suppression is based on attenuating the SI signal by physically separate/change somehow the transmitter antenna and the receiver antenna of the device. Inside of this group we have *antenna separation* where the separation distance between both antennas of the FD device is physically increased, the *antenna cancellation* where the distances between the antennas of both devices are exploited taking into account the wave length, and finally the *directional passive suppression* where the angle in the antennas is exploited in a way that the main radiation lobes in the antennas has minimal intersection.

The *Analog SI Cancellation* is performed before the signal goes into Analog-Digital Converter (ADC), e.g. by using the QHx220 noise cancelling integrated circuit. However, this work focuses only on SDR implementation using GNU Radio which allow us to work only after ADC. The *Digital SI Cancellation* mechanism works exactly on that place by taking as advantage the knowledge of the interfering signal for the purpose of canceling it. The SI can be subdivided in linear and nonlinear components previously discussed, the latter is the less important since it is really hard to estimate random happenings on the hardware, however in the first there is some known algorithms to SI estimating and correction like the block-based (Singular Value Decomposition, or QR decomposition) and the adaptive algorithms (Least Mean Square (LMS)). While the first one operates on an entire matrix the latter operates on each row and iteratively adjust the coefficients in a way to predict the signal performing better as shown on [18], thus, being the choice to this work.

III. Scrambled-Coding for Secrecy with Self-Interference Generation

The idea implemented in this paper is using both coding for secrecy, more concretely the adapted SCS-HK, along with interference generation by the legitimate receiver and its cancellation. Since the legitimate receiver knows the characteristics of the interference it is generating, he can use Digital SI Cancellation to cancel it, giving him an advantage over an eavesdropper. The conceptual scheme is represented on Fig. 3.

A. *Transmitter Side*

The transmitter is represented on Fig. 7(a). The information is picked from the file and divided into frames, with one bit for each byte. Then each frame is scrambled using an additive scrambler with a random key, which is concatenated in front of that frame. This frame goes through the forward error correction code encoder, and finally is properly padded to enable the creation of a payload. A packet is formed by adding to the payload an header constituted by a known sequence (for both transmitter and receptor) and the payload length. The generated packet sequence goes through the digital QPSK modulator, and pulse shaping (to limit the signals bandwidth), and is feed to the USRP transmitter. Furthermore, for adaptation purposes on the receptor, it is added random symbols with the "new_vec" block that are discarded on the receptor.

B. *Receptor Side*

Legitimate Receiver: Bob it is constantly generating Gaussian noise ("Noise source" block), this noise is interpolated (for rate needs) with

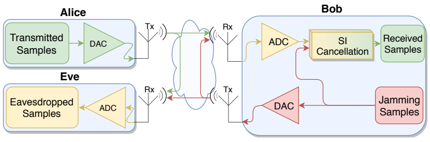


Fig. 3. Conceptual Scheme of SI Cancellation of this work.

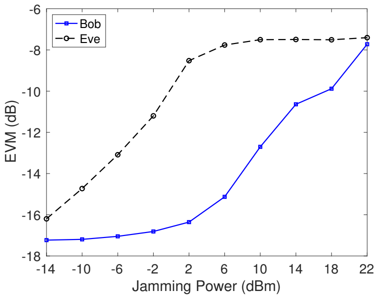


Fig. 4. Jamming Power and EVM comparison on both Bob and Eve.

the "Interpolating FIR Filter" block and outputted to one of the USRP's antenna. At the same time it is receiving both information and noise, however since it has the knowledge of the generated noise than uses the "LMS Filter" block to cancel it. Note that at the input of this block the noise samples need to be synchronized with the mixed ones (signal + noise), that is exactly what the "Synchronization" block does. After retrieving the transmitted signal it is treated and decoded in the "Decoding" block. The Bob's flow graph can be visualized on Fig. 7(d).

Illegitimate Receiver: Eve it is a normal receiver as can be visualized on Fig. 7(c) without any interference generation or cancellation, thus, receives the signal which in this case is both jam signal mixed with the interesting signal and tries to treat and decode it using the "Decoder" block.

SCS-HK Decoder: The interesting signal is inputted in the "Decoder" block which is the Fig. 7(b), on both legitimate and illegitimate receivers. This block receives the interesting signal where firstly it is required to be taken care of some aspects inducted by the channel. The clock offset is treated on the "Polyphase Clock Sync" block where also applies the RRC filter and decimate the signal, than goes though the "CMA Equalizer" to correct multipath aspects and finally to a "Costas loop" block to finely correct the frequency offset. With the treated signal it is possible to do the reverse what was done in the transmitter, namely it is QPSK demodulated, found the known word on the header using the "Correlate Access Code - Tag Stream" where is extracted the payload, it is discarded the padded bits and goes through a forward error correction decoder resulting in the information scrambled. Finally this information is descrambled using the key transmitted and outputted to a file.

IV. Results

On the receiver the signal should ideally have all constellation points in the ideal positions, however in practice this does not happen, thus, we will measure the error vector magnitude (EVM), i.e. the distance between the received symbols and the ideal ones. Less EVM value, in dB, means better approximation to the constellation points, thus, better possibilities to correctly retrieve the information.

The SDR's position configuration that was used to perform the tests is represented on Fig. 6. The Bob's Tx antenna is lying down not standing for better performance on jamming cancellation. All these tests the Alice's power was 8 dBm, the Bob's jamming power was varied between -14dBm to 22dBm. All the correspondent EVM (of both Alice and Eve) and Jamming Power values can be visualized on Fig. 4. Furthermore, Fig. 5 depicts the difference on Bob's and Eve's received signal and hence demodulation with 2 dBm of jamming. As it can be viewed Bob can successfully cancel the interference and demodulate the signal while Eve can not demodulate it correctly due to the added interference.

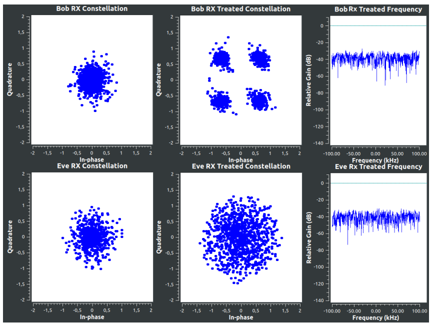


Fig. 5. Bob and Eve received signal comparison.

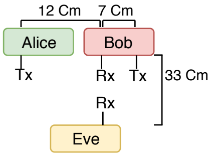


Fig. 6. SDR position scheme.

V. Conclusion

In this paper, we presented a prototype implementation using SDR along with GNU Radio of the SCS-HK coding secrecy scheme with interference generation by the receiver and its cancellation using full-duplex mechanisms. This results in an advantage over an eavesdropper that is not able to completely remove the generate interference when trying to decode the transmitted signal. Future work will address some improvements on the implementation like using enhanced error correction on top of the known words.

Acknowledgment

This work is funded by FCT/MCTES through national funds and when applicable co-funded EU funds under the projects UIDB/EEA/50008/2020, and PES3N (SAICT-45-2017-POCI-01-0145-FEDER-030588).

References

[1] William Stallings. *Cryptography and Network Security*. Pearson Education, 2017.

[2] C. E. Shannon. Communication theory of secrecy systems. *The Bell System Technical Journal*, 28(4):656–715, 1949.

[3] A. D. Wyner. The wire-tap channel. *The Bell System Technical Journal*, 54(8):1355–1387, 1975.

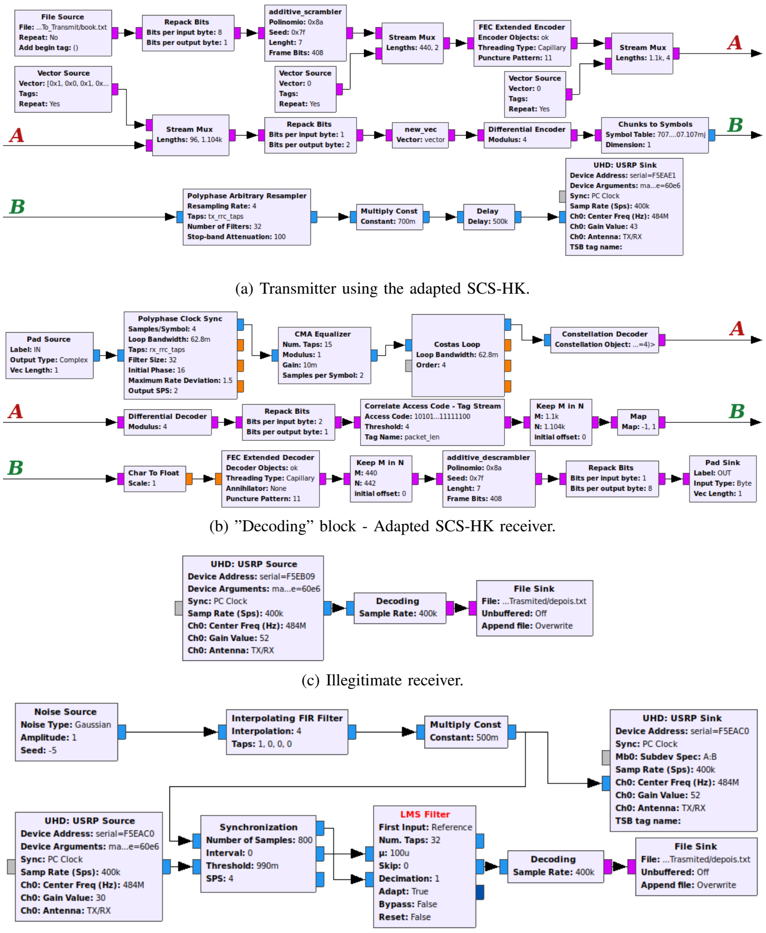
[4] D. Klinc, J. Ha, S. W. McLaughlin, J. Barros, and B. Kwak. LDPC codes for the gaussian wiretap channel. *IEEE Trans. on Inform. Forensics and Security*, 6(3):532–540, 2011.

[5] Marco Baldi, Marco Bianchi, and Franco Chiaraluce. Coding with scrambling, concatenation, and HARQ for the AWGN wire-tap channel: A security gap analysis. *IEEE Trans. on Inform. Forensics and Security*, 7:883–894, 06 2012.

[6] Joao Vilela, Marco Gomes, Willie Harrison, Dinis Sarmento, and Fabio Dias. Interleaved concatenated coding for secrecy in the finite blocklength regime. *IEEE Signal Proc. Letters*, 23:1–1, 01 2015.

[7] D. Sarmento, J. Vilela, W. K. Harrison, and M. Gomes. Interleaved coding for secrecy with a hidden key. In *2015 IEEE Globecom Workshops*, pages 1–6, 2015.

[8] C. Martins, T. Fernandes, M. Gomes, and J. Vilela. Testbed implementation and evaluation of interleaved and scrambled coding for physical-layer security. In *2018 IEEE 87th VTC - Spring*, pages 1–6, June 2018.



(a) Transmitter using the adapted SCS-HK.

(b) "Decoding" block - Adapted SCS-HK receiver.

(c) Illegitimate receiver.

(d) Interference generation an its cancellation on the legitimate receiver.

Fig. 7. The adapted SCS-HK transmitter side 7(a), the interference generation by in the legitimate receiver side 7(d), the illegitimate receptor 7(c) were both use the same SCS-HK receiver block 7(b).

[9] E. Tekin and A. Yener. Achievable rates for the general gaussian multiple access wire-tap channel with collective secrecy. 2, 01 2007.

[10] E. Tekin and A. Yener. The general gaussian multiple-access and two-way wiretap channels: Achievable rates and cooperative jamming. *IEEE Transactions on Information Theory*, 54(6):2735–2751, June 2008.

[11] X. He and A. Yener. Providing secrecy with structured codes: Tools and applications to two-user gaussian channels. *CoRR*, abs/0907.5388, 2009.

[12] J. Xie and S. Ulukus. Secure degrees of freedom of the gaussian wiretap channel with helpers. In *50th Annual Allerton Conference on Communication, Control, and Computing*, pages 193–200, Oct 2012.

[13] Z. Zhang, K. Long, A. V. Vasilakos, and L. Hanzo. Full-duplex wireless communications: Challenges, solutions, and future research directions. *Proceedings of the IEEE*, 104(7):1369–1409, July 2016.

[14] S. Gollakota and D. Katabi. Physical layer wireless security made fast and channel independent. In *2011 Proceedings IEEE INFOCOM*, pages 1125–1133, April 2011.

[15] G. Zheng, I. Krikidis, J. Li, A. P. Petropulu, and B. Ottersten. Improving physical layer secrecy using full-duplex jamming receivers. *IEEE Transactions on Signal Processing*, 61(20):4962–4974, Oct 2013.

[16] A. Sahai, G. Patel, C. Dick, and A. Sabharwal. On the impact of phase noise on active cancelation in wireless full-duplex. *IEEE Trans. on Vehicular Technology*, 62(9):4494–4510, Nov 2013.

[17] S. Hong, J. Brand, J. I. Choi, M. Jain, J. Mehlman, S. Katti, and P. Levis. Applications of self-interference cancellation in 5g and beyond. *IEEE Communications Magazine*, 52(2):114–121, February 2014.

[18] Adam Parower by The Aerospace Corporation. GNU Radio conference of 2017 - a GNU radio-based full duplex radio system, 2017.