# Design and Optimization of Radio-over-Fiber Links

**A Degree Thesis**

**Submitted to the Faculty of the**

**Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona**

**Universitat Politècnica de Catalunya**

**by**

**Arturo Aibar**

**In partial fulfilment**

**of the requirements for the degree in**

**TELECOMUNICATIONS TECHNOLOGIES AND SERVICES ENGINEERING**

**Advisor: MARÍA CONCEPCIÓN SANTOS BLANCO**

**Barcelona, June 2021**

## Abstract

In recent years, the use of optical links for data transmission has highly increased due their advantages as compared to electrical links, and with it, the number of software tools dedicated to simulate these optical links.

The main focus of this project is to create a software tool which can help in the study of this type of links, their components and their different behaviour. For that some blocks from VPIphotonics software tool have been converted into Matlab's functions to be able to create simple simulations.

# Resum

En els últims anys l'ús de enllaços òptics han sigut altament incrementats degut als avantatges que aquests suposen sobre els enllaços elèctrics, junt a això, també han incrementat els programes dedicats a la simulació d'aquest enllaços òptics.

L'objectiu principal d'aquest projecte es crear programes els quals puguin server per l'estudi d'aquest tipus d'enllaços i el seus components a més del diferents comportaments d'aquest. Per això s'han convertit en funcions de Matlab diversos blocs del programa VPIphotonics per fer diverses simulacions simples.

# Resumen

En los últimos el uso de enlaces ópticos han sido altamente incrementado debido a las ventajas que supone este sobre los enlaces eléctricos, junto con esto, también han aumentado los programas dedicados a la simulación de estos enlaces ópticos.

El objetivo principal de este proyecto es crear programas lo cuales puedan servir para el estudio de este tipo de enlaces y sus componentes además del diferente comportamiento de estos. Para ello se han convertido en funciones de Matlab diferentes bloques del programa VPIphotonics para poder hacer varias simulaciones simples.

## Acknowledgements

First of all, I would like to thank my supervisor Maria Santos for giving me the opportunity to do this thesis, and thank her for doing this a great experience due her kindness, comprehension and his implication on this project.

Also I would like to thanks to the person who I was hand by hand doing this project and working so hard, my mate Albert. I am really grateful to have had this opportunity to do this thesis with you. Without any doubt, doing this project wouldn’t have been the same if we hadn’t been together helping each other and evolving at the same time. Thanks mate.

Finally, I would like to thanks to the people who, although they have not participated directly in the project, have been helping and supporting me this whole time.

# Revision history and approval record

| Revision | Date | Purpose |
|---|---|---|
| 0 | 15/03/2021 | Document creation |
| 1 | 13/04/2021 | Document update |
| 2 | 08/05/2021 | Document update |
| 3 | 06/06/2021 | Document update |
| 4 | 12/06/2021 | Document revision |
| 5 | 17/06/2021 | Document correction |
| 6 | 18/06/2021 | Document update |
| 7 | 19/06/2021 | Document update |

DOCUMENT DISTRIBUTION LIST

| Name | e-mail |
|---|---|
| Arturo Aibar | arturo.aibar@estudiantat.upc.edu |
| Maria Santos | santos@tsc.upc.edu |
| Antoni Barlabé | barlabe@tsc.upc.edu |

| Written by: | | Reviewed and approved by: | |
|---|---|---|---|
| Date | 15/03/2021 | Date | 16/06/2021 |
| Name | Arturo Aibar | Name | Maria Santos |
| Position | Project Author | Position | Project Supervisor |

# Table of contents

# 1. Introduction

## 1.1. Project objectives

The main objective of this project is to recreate the simulation tool, VPIphotonics, in Matlab by converting its blocks in functions. Then by using those functions, create a link transmission simulation.

To do this, there are other objectives to accomplish:

-Study and understand how the different elements of a link (Mach-Zehnder, Amplifier, Photodetector…) work mathematically and conceptually.

-Study and understand the different phases of the data processing in transmission and reception of the information.

-Mix both parts mentioned before and create a software tool to simulate them.

## 1.2. Thesis overview

In this thesis, in Chapter 2, the Radio-over-fiber link is going to be explained, along with its main components such as Mach-Zehnder modulator (MZM), Optical fiber, detector, IQ modulator and demodulator.

In the following chapter, Chapter 3, will be a detailed description of the two software tool (MZM Transfer function & Rof Design System) and the functions used in each of them with some examples of their use.

Finally, in Chapter 4, there will be a conclusion and which topics would be interesting to explore for future work.

Also, at the end of the thesis there are two appendices to see the whole Live Script of the software tool created.

# 2. Radio-over-fiber

The development of this project follows the basic radio-over-fiber system scheme represented in figure 2.1

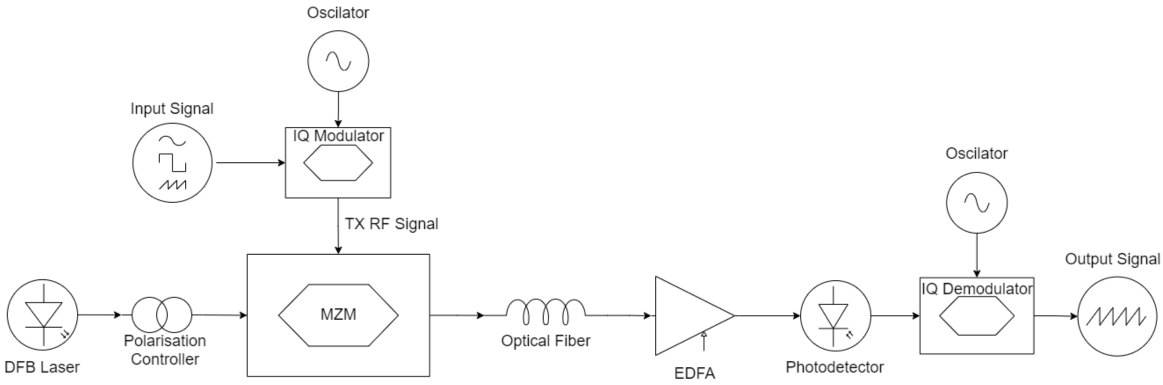



Figure 2.1: Radio-over-Fiber basic scheme

The following sections will explain the components that appear in the figure 2.1. Since there are two simulation tools (MZM Transfer function & RoF Desing System), the first elements to be explained are the ones that does not have any interaction with the data and then the ones that work with data.

## 2.1. Mach-Zehnder Modulator (MZM)

The Mach-Zehnder is the modulator used in this project. It is the most typical modulators based in the electrical-optic phenomenon that appears in crystals like Lithium Niobate ($LiNbO_3$). [1-3]

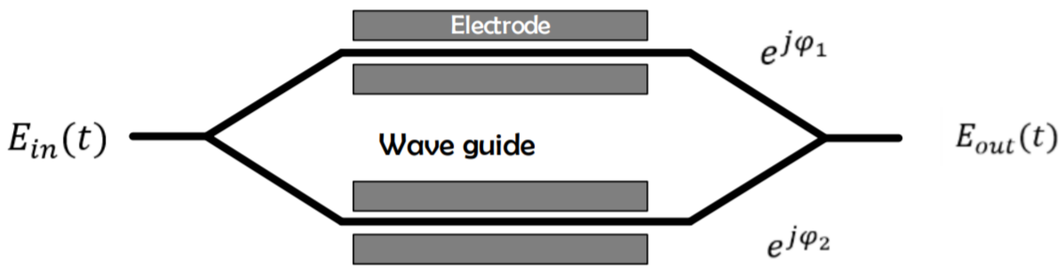



Figure 2.2 Diagram of an amplitude modulator based on the Mach-Zehnder interferometer

As we can see in figure 2.2 the optical input ($E_{in}$) is divided into two paths and introduced in both branches of the MZM, which will apply a different phase depending on the values of the voltages $V_1$ and $V_2$ applied to the branches. The output of the MZM can be written as:

$$E_{out} = \frac{E_{in}}{2\sqrt{L_{MZM}}} \left[e^{j\frac{\pi}{V_\pi}V_1} + e^{j\frac{\pi}{V_\pi}V_2}\right] \qquad \text{Expression 2.1}$$

Where $L_{MZM}$ are the losses of the modulator and $V_\pi$ is a voltage needed to produce a phase shift of 180º in each branch of the modulator.

Depending on the voltage applied to the electrodes we can find two basic configurations:

-Push-pull, where the two branches are fed with antipodal voltage, producing a $\pm\pi V/2V_\pi$ phase shifting. In this case since both branches are fed with the same voltage, $V_\pi$ is the voltage needed to move from a maximum of the transmission function to a minimum or vice-versa

-Dual drive, where different and independent voltages are applied to the branches. In this case the $V_\pi$ remains with its previous definition: $V_\pi$ is a voltage needed to produce a phase shift of 180º in each branch of the modulator.

Also let's define some variables that will be used for the rest of the explanation:

$$m = \frac{V_{RF}\,\pi}{V_\pi} \qquad \theta_B = \frac{V_B\,\pi}{V_\pi}$$ Expression 2.2

Where $V_{RF}$ and $V_B$ are the amplitude of the RF signal and the bias voltage respectively.

Now we will explain the different configurations for a Radio-over-Frequency Links.

### 2.1.1. MZM Push-Pull (MZM-PP)

The first configuration is the MZM Push-Pull, which has the same input in both branches with different sign. This input is the sum of the Bias Voltage ($V_B$) and RF signal ($V_{RF}$ $\cos(\omega_{RF}\, t)$) (figure 2.3)

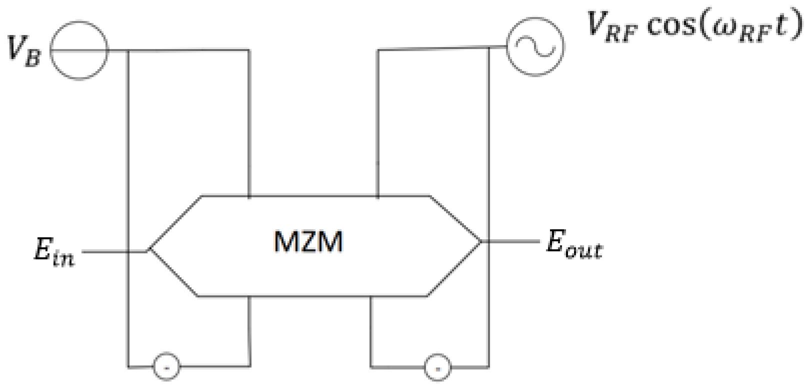



Figure 2.3 MZM-PP Configuration

Using the expression 2.1 and knowing that in the MZM-PP configuration we are interested in letting:

$V_1 = V_B + V_{RF}\cos(\omega_{RF}\, t)$

$V_2 = -V_1$

The field at the output of the modulator has the following expression:

$$E_{out} = \frac{E_{in}}{2}\left[e^{j\frac{\pi}{2V_\pi}(V_B+V_{RF}\cos(\omega_{RF}\, t))} + e^{-j\frac{\pi}{2V_\pi}(V_B+V_{RF}\cos(\omega_{RF}\, t))}\right] \quad \text{Expression 2.3}$$

$$E_{out} = \frac{E_{in}}{\sqrt{L_{MZM}}}\cos\left(\frac{\pi}{2V_\pi}(V_B + V_{RF}\cos(\omega_{RF}\, t))\right) \quad \text{Expression 2.4}$$

$$E_{out} = \frac{E_{in}}{\sqrt{L_{MZM}}}\cos\left(\frac{\theta_B + m\cos(\omega_{RF}\, t)}{2}\right) \quad \text{Expression 2.5}$$

Applying the corresponding small-signal approximation m<<1:

$$E_{out} \approx \frac{E_{in}}{\sqrt{L_{MZM}}}\left[\cos\left(\frac{\theta_B}{2}\right) - \sin\left(\frac{\theta_B}{2}\right)\frac{m}{2}\cos(\omega_{RF}\, t)\right] \quad \text{Expression 2.6}$$

## 2.1.2. MZM Dual-Drive (MZM-DD)

The second configuration is the MZM Dual-Drive. This configuration in the upper branch has the RF signal and the lower branch has the bias voltage as an input. (figure 2.4)

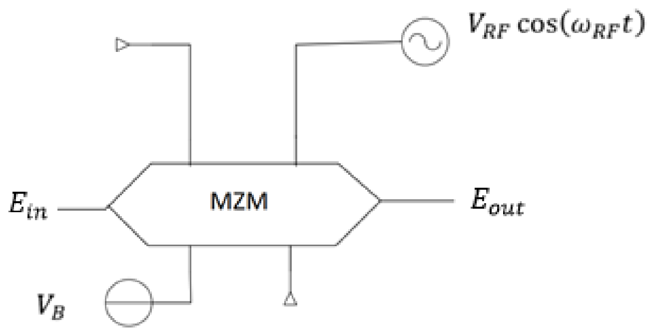



Figure 2.4 MZM-DD Configuration

Using the expression 2.1 and knowing that in the MZM-DD configuration we are interested in letting:

$V_1 = V_{RF}\cos(\omega_{RF}\, t)$

$V_2 = V_B$

The field at the output of the modulator has the following expression:

$$E_{out} = \frac{E_{in}}{2}\left[e^{j\frac{\pi}{V_\pi}V_{RF}\cos(\omega_{RF}\, t)} + e^{j\frac{\pi}{V_\pi}V_B}\right] \quad \text{Expression 2.7}$$

$$E_{out} = \frac{E_{in}}{2}\left[e^{jm\cos(\omega_{RF}\, t)} + e^{j\theta_B}\right] \quad \text{Expression 2.8}$$

### 2.1.3. MZM Dual-Parallel (MZM-DP)

The last configuration is the MZM Dual-Parallel. This configuration is more complex than the other ones due to the fact that it is composed by two MZM-PP in each branch. (figure 2.5)

The RF signal and the first bias voltage ($V_{B1}$) are applied to the upper MZM.

The second bias voltage ($V_{B2}$) is applied to the lower MZM.

Finally, there is a third bias voltage ($V_{B3}$) applied to the output of the lower MZM to apply a phase shift between the outputs of the two MZM.

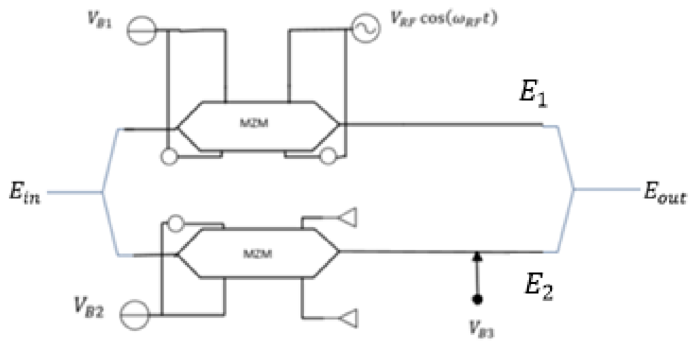



Figure 2.5 MZM-DP Configuration

The optical field at the output is a sum of two outputs from a different MZM with a phase between them.

$$E_{out} = \frac{1}{2}\left[E_1 + e^{jV_{B3}\frac{\pi}{V_\pi}}E_2\right] \quad \text{Expression 2.9}$$

$$E_1 = \frac{E_{in}}{2}\left[e^{j\frac{\pi}{2V_\pi}(V_{B1}+V_{RF}\cos(\omega_{RF}t))} + e^{-j\frac{\pi}{2V_\pi}(V_{B1}+V_{RF}\cos(\omega_{RF}t))}\right] =$$

$$E_{in}\cos(\frac{\pi}{2V_\pi}(V_{B1} + V_{RF}\cos(\omega_{RF}t))) \quad \text{Expression 2.10}$$

$$E_2 = \frac{E_{in}}{2}\left[e^{j\frac{\pi}{2V_\pi}V_{B2}} + e^{-j\frac{\pi}{2V_\pi}V_{B2}}\right] = E_{in}\cos(\frac{\pi}{2V_\pi}V_{B2}) \quad \text{Expression 2.11}$$

By using basic trigonometric identity and some approximations for small amplitude signals the equation can be rewritten as:

$$E_{out} \approx \frac{E_{in}}{2}[\cos\left(\frac{\theta_{B1}}{2}\right) - \sin\left(\frac{\theta_{B1}}{2}\right)\frac{m}{2}\cos(\omega_{RF}\ t) + e^{j\theta_{B3}}\ \cos\left(\frac{\theta_{B2}}{2}\right)] \quad \text{Expression 2.12}$$

## 2.2. Optic Fiber

The optical fiber is the environment where our optical field will be transferred. The phenomenon that our fiber is going to introduce is the chromatic dispersion.

### 2.2.1. Chromatic dispersion

The Chromatic dispersion in optic fiber is the result of the group delay (different wavelengths arriving to their destination at different times). This basically produces a temporal spreading of the signal through the fiber.

The dispersion is characterised by the D[ps/nm·km] parameter. This parameter describes the time delay between two components separated in wavelength 1 nm when the signal has travelled 1km. The most common value for this parameter is 17 ps/nm·km for single-mode fibers in the C-Band.

Mathematically we can describe the dispersion as the relation between the constant propagation (β) and frequency. Assuming that the carrier frequency is much higher that the RF frequency ($\omega_0 >> \omega_{RF}$), we can develop the propagation constant with its Taylor series:

$$\beta(\omega_0 + \omega) = \beta(\omega_0) + \frac{\partial \beta}{\partial \omega}\omega + \frac{1}{2}\frac{\partial^2 \omega}{\partial \omega^2}\omega^2 = \beta_0 + \beta_1 + \frac{1}{2}\beta_2\omega^2$$ Expression 2.13

Where:

-$\beta_0$ is the phase velocity, the speed with which the phase of a wave propagates in space.

-$\beta_1$ is the group delay at a certain $\omega_0$.

-$\beta_2$ is related to the dispersion parameter.

Given a pulse with a spectral width of Δf, the widening after travelling through a fiber of length L, can be expressed as:

$$\Delta T = \frac{\partial T}{\partial \omega}\Delta\omega + \frac{\partial}{\partial \omega}\left(\frac{L}{vg}\right)\Delta\omega = L\frac{\partial^2 \omega}{\partial \omega^2}\Delta\omega^2 = L\beta_2\Delta\omega$$ Expression 2.14

Where vg is the group velocity, defined as $vg = (\frac{\partial \beta}{\partial \omega})^{-1}$

The previous equation (Expression 2.14) can be rewritten in terms of wavelength using $\omega = \frac{2\pi c}{\lambda}$ and $\Delta\omega = -\frac{2\pi c}{\lambda^2}\Delta\lambda$

$$\Delta T = \frac{\partial}{\partial \lambda}\left(\frac{L}{vg}\right)\Delta\lambda = DL\Delta\lambda$$ Expression 2.15

And finally we can get the expression 2.16 that relates the dispersion parameter D with $\beta_2$:

$$\Delta T = \frac{\partial}{\partial \lambda}\left(\frac{L}{vg}\right) = \left(-\frac{2\pi c}{\lambda^2}\right)\beta_2$$ Expression 2.16

### 2.2.2. Group delay

Our system has an RF signal modulated over carrier, that implies that both components will suffer delay: a phase delay which affects the carrier and a group delay which affects the envelope of the transmitted signal.

Taking into account the optical carrier frequency ($\omega_0$) and the envelop modulated with a frequency ($\omega_{RF}$), the low-pass equivalent of the optical field at the output of the modulator can be expressed:

$$E_{out} = 1 + \frac{m_0}{2}(e^{j\omega_{RF}t} + e^{-j\omega_{RF}t})$$ Expression 2.17

In the reception, at the end of the optical fiber, once the signal has travelled a distance L and is affected by the propagation constant β, the optical field with the Taylor series can be expressed as:

$$E_{out} = 1 + \frac{m_0}{2}(e^{j\omega_{RF}t}e^{-j(\beta_0\omega_0+\beta_1\omega_{RF}+\frac{\beta_2}{2}\omega_{RF}^2)L} + e^{-j\omega_{RF}t}e^{-j(\beta_0\omega_0-\beta_1\omega_{RF}+\frac{\beta_2}{2}\omega_{RF}^2)L})$$

Expression 2.18

Since we are detecting amplitude in reception, the phase delay ($\tau f = \frac{\beta_0 L}{\omega_0}$) that the carrier suffers can be ignored. Knowing this, the expression can be described as:

$$E_{out} = 1 + \frac{m_0}{2}(e^{j\omega_{RF}t} + e^{-j\omega_{RF}t})e^{-j(\beta_1\omega_{RF}+\frac{1}{2}\beta_2\omega_{RF}^2)L}$$ Expression 2.19

And finally defining the parameter $\phi = \frac{\beta_2}{2}\omega_{RF}^2 L = \frac{\pi D \lambda_0^2 f_{RF}^2 L}{c}$ the equation can be rewritten as:

$$E_{out} = 1 + m_0 e^{j\phi}\cos(\omega_{RF}(t - \beta_1 L))$$ Expression 2.20

The following image (figure 2.6) shows a theoretical $S_{21}$ for two different distances (L=25km and L=50km).

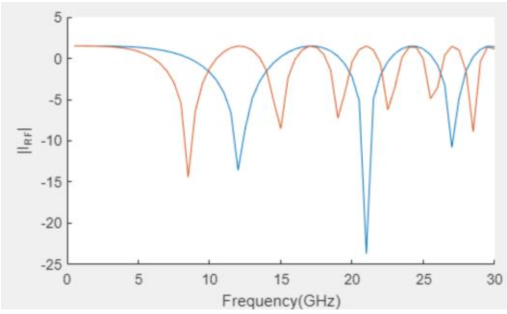



Figure 2.6: Theoretical $S_{21}$ of a SMF fibre at distances L=25km (blue) and L=50km (orange)

## 2.3. Erbium Doped Fiber Amplifier (EDFA)

The function of an Erbium Doped Fiber Amplifier (EDFA) is to get the same optical power at the output of the photodetector to compare the BER results between the MZM configurations.

The physical functioning of the EDFA is the following one: The EDFAs are made of $Er^{3+}$ ions which can have 2 states; Fundamental state or Excited state. The Excited ions, close to an input signal photon, have a probability of releasing another identical photon due the stimulated emission. From one input signal photon we can obtain an average of G (gain) photons at the fiber output. [2] (figure 2.7)

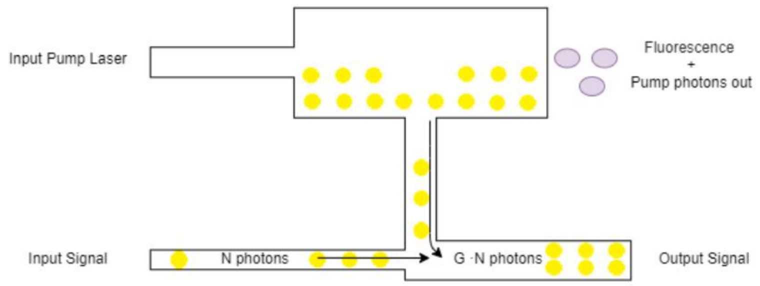



Figure 2.7: EDFA Principle Scheme

## 2.4. Photodetector

The photodetector is an optical receiver that converts optical signals into electrical signals.

One of the most common photodetectors is the PIN photodiode that has a three-layer structure. The middle layer is an intrinsic semiconductor and the outer layers are P-type and N-type. (figure 2.8)

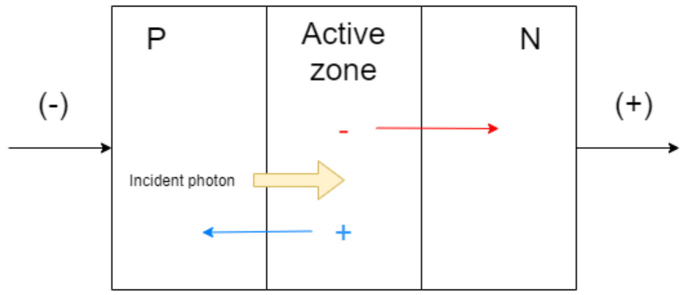



Figure 2.8: PIN Principle Scheme

The process to convert optical signals into electrical signals is via optical absorption. This process is based in the generation of an electron-hole pair in the active zones due the energy of the photon. Those electrons and holes are accelerated in opposite directions due to the electric field through which the diode is polarized, creating a current flow proportional to the incident optical power.

The output of the photodetector is a photocurrent and is expressed by the following expression:

$$I_{PD} = RP_{in} \approx |E_{in}|^2 \qquad \text{Expression 2.21}$$

Where R [A/w] is the Responsivity and $P_{in}$ the input power of the photodetector.

### 2.4.1. Noise

The parameter that always affects in a link is the noise coming from various sources. The noise is a power added to the signal through the transmission that in some cases can affect negatively to it if a good Signal to Noise Ratio (SNR) is not assured.

In this thesis there are three types of noise that affect to an RoF link that we will study.

All noises are assumed white and we characterize them by their spectral densities meaning that the noise parameters are given in normalized form for a bandwidth of 1 Hz

#### 2.4.1.1. Thermal noise

Thermal noise is the one generated by the agitation of electrons. It is mainly caused by the resistive elements that form the electronic components of our system.

This noise can be characterised by the mean square voltage $\langle V_{th}^2 \rangle = 4KTR$ or its mean square intensity $\langle I_{th}^2 \rangle = \frac{4KT}{R}$ introduced to the system. Being K the Boltzmann factor, T the temperature and R the resistance [5-6] (figure 2.9)


Figure 2.9: Thermal noise equivalent sources

#### 2.4.1.2. Rin noise

Rin noise is the one that results from random fluctuations of intensity. It mainly depends on the intensity of laser used ($I_D$) and the continuous current of the photodetector.

Mathematically it can be expressed as: $\langle I_{RIN}^2 \rangle = RIN\ I_D^2$ [5-6]

Being RIN a parameter of the laser used, with a typical factor of -10dB/Hz.

#### 2.4.1.3. Shot noise

Shot noise emerges from the current fluctuations due to the quantic nature of the process of generating electrons from the absorption of photons in the photodetector.

Mathematically can be expressed as: $\langle I_{Sh}^2 \rangle = 2qI_D$, where q is the charge of the electron in Coulombs ($1.6 \cdot 10^{-19}$ C) and $I_D$ the DC current. [5-6]

If we analyse the mean square value, we will notice that this noise increases with the optical power present in the photodiode.

These components that just have been explained, are independent of any type of data and they are used for both software tools: MZM Transfer function and RoF Design System. From now on the following part is focused on data. That means that the next components to be explained are going to be used only for RoF Design System.

## 2.5. IQ Modulator

The IQ modulator is where the signal that is going to be sent is created [4]. To explain how it is done, we are going to follow the scheme in figure 2.10.

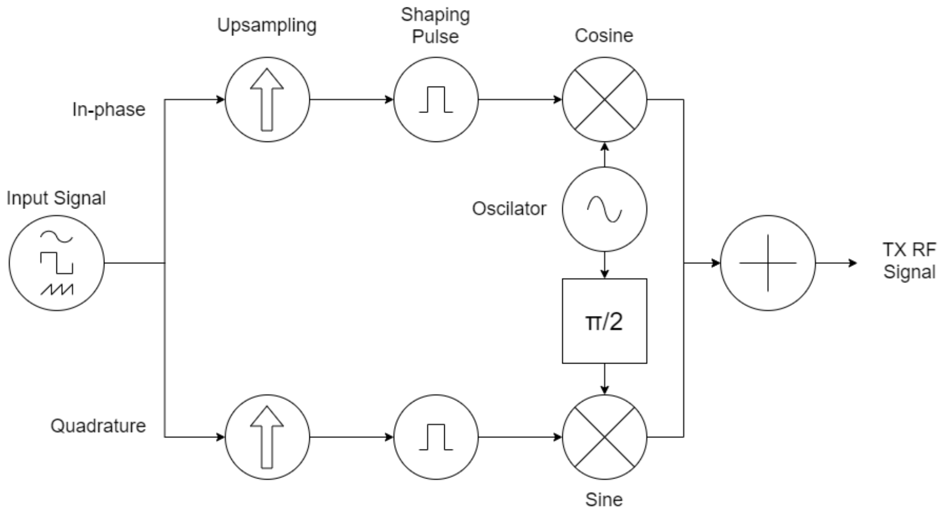



Figure 2.10: IQ Modulator Scheme

To start, the sequence of bits, that we want to transmit, is created and separated in the components in-phase and quadrature. That results in a two sequence of bits with half the length of the original one.

This process first consists in transforming every bit into a shaping pulse (in our case a square root raised cosine to reduce the bandwidth used and minimize the intersymbol interference) to give the bits a temporal meaning. To be able to do this, first it is necessary to do an up-sampling that consist in introducing 'zeros' samples in between the bits. The main function of this is to have space between the bits to introduce the shaping pulse. Then we convolve the result of the up-sampling with the shaping pulse to get the signal in time domain.

The following part consists in multiplying the signal that is in the in-phase branch with a cosine (that determines the RF frequency used in the system) and the same with the signal in quadrature branch with a sine. That part is what is going to let us extract both components in reception.

The last part to get the final signal that is going to be introduced into the Mach-Zehnder modulator, is to sum both components that are in in-phase and quadrature branch. The final expression has the following representation.

$$S_{out}(t) = I(t)\cos(2\pi f_{RF}t) + Q(t)\sin(2\pi f_{RF}t) \qquad \text{Expression 2.22}$$

Where I(t) and Q(t) are, respectively, the in-phase and quadrature component at the end of the shaping pulse.

## 2.6. IQ Demodulator

The IQ Demodulator is required to reobtain the original sequence of bits (the information) by doing the complementary process of the IQ Modulator [4]. To explain the process, we are going to follow the scheme of the figure 2.11

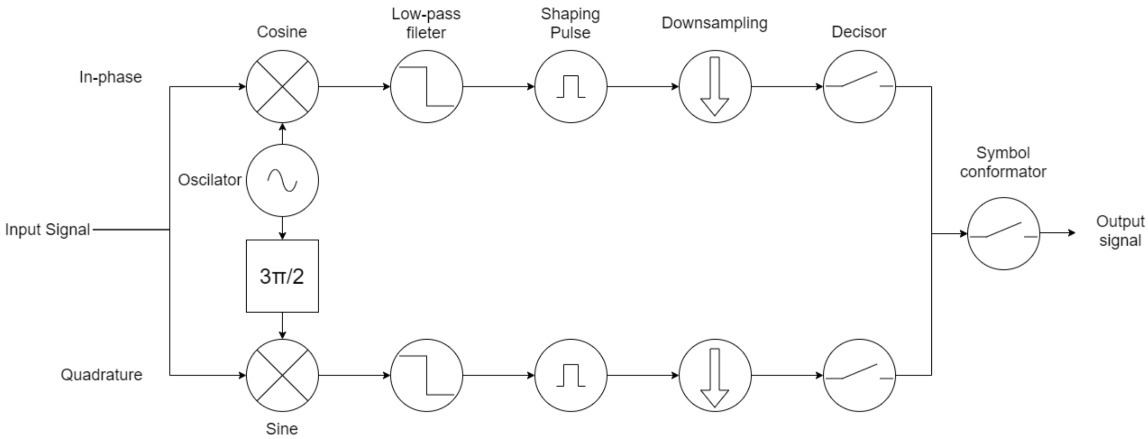



Figure 2.11: IQ Demodulator Scheme

The first step is to divide the path of the signal in two branches (so we have the same signal in both branches). As we want to get again the in-phase and quadrature components it is necessary to multiply again by a cosine and sine with the same $f_{RF}$ as the used in modulation. The expression of the signal once this step is done, for the in-phase case, is the following one:

$$S_I(t) = 2S_{out}(t)\cos(2\pi f_{RF}t) = 2I(t)\cos(2\pi f_{RF}t)\cos(2\pi f_{RF}t) + 2Q(t)\sin(2\pi f_{RF}t)\cos(2\pi f_{RF}t) =$$

$$2I(t)\frac{1}{2}[\cos(2\pi f_{RF}t + 2\pi f_{RF}t) + \cos(2\pi f_{RF}t - 2\pi f_{RF}t)] + 2Q(t)\sin(2\pi f_{RF}t)\cos(2\pi f_{RF}t) =$$

$$I(t) + I(t)\cos(2\cdot 2\pi f_{RF}t) + Q(t)\sin(2\cdot 2\pi f_{RF}t) \qquad \text{Expression 2.23}$$

For the quadrature case, the result is:

$$S_Q(t) = Q(t) + Q(t)\sin(2\cdot 2\pi f_{RF}t) + I(t)\,\mathrm{cosine}(2\cdot 2\pi f_{RF}t) \qquad \text{Expression 2.24}$$

Where the $S_I$ and $S_Q$ is the Output signal in the in-phase and quadrature case and the $S_{out}$ is the Input signal (figure 2.11)

Since we are only interested in the in-phase and quadrature components, we want to eliminate the part with the double frequency components with a low-pass filter.

The next step is to convolve again with the same shaping pulse and do a down-sampling to extract the zeros.

Now, we have almost the same signal as the original one, but due to the noise of the system we have soft 1s and 0s. By using the decisor we can transform those soft bits into hard 1s and 0s and get the original in-phase and quadrature components.

## 2.7. Summary

In the table 2.1 we can see a comparison of relevant characteristics of the MZM configurations explained before.

| | Push-Pull | Dual-drive | Dual-parallel |
|---|---|---|---|
| $E_{out}$ | $\cos\left(\frac{\theta_B}{2}\right) - \sin\left(\frac{\theta_B}{2}\right)\frac{m}{2}\cos(\omega_{RF}\ t)$ | $e^{jm\cos(w_{RF}\ t)} + e^{j\theta_B}$ | $e^{j\theta_{B3}}\ \cos\left(\frac{\theta_{B2}}{2}\right) - \frac{m}{2}\cos(\omega_{RF}\ t)$[1] |
| Number of bias voltages | 1 | 1 | 3 |
| Meaning of $\theta_B$ | Voltage applied to each branch | Difference of voltage between branches | Formed by two PP:<br>1º in null-point<br>2º in optimum<br>3º to control noch |
| Reaches maximum value | Yes | Yes | Yes |
| Control of fading nulls (with $V_{NOTCH}$) | No | Yes[2] | Yes[3] |

Table 2.1: Summary of the relevant equations of the MZM

1. The expression in this case is supposing $\theta_{B1} = \pi$

2. Given a frequency in which we want to optimize the transmission, we can find the voltage of the bias that we have to apply using the following equation:

$$V_{B\_NOTCH} = V_\pi[\pm(2n-1) + 2D\lambda_0^2 f_{RF}^2 L/c] \quad \text{Expression 2.25}$$

Then we find $\theta_B$ using the expression:

$$\theta_{B\_NOTCH} = \pi\frac{V_{B\_NOTCH}}{V_\pi} \quad \text{Expression 2.26}$$

3. This case works equals as the previous one, but with the following equation:

$$V_{B\_NOTCH} = 2V_\pi[\pm n + D\lambda_0^2 f_{RF}^2 L/c] \quad \text{Expression 2.27}$$

# 3. Simulation tools

In this chapter, the two software tools created, MZM Transfer functions & RoF Design System, are going to be explained.

First a detailed description of how the software tools works and which information will be obtained by using them is presented. Then a description of the functions of the software and some examples of how they can be used is given.

## 3.1. MZM Transfer function

### 3.1.1. Description

MZM Transfer function has the main objective to simulate the link that would be used to obtain the MZM transfer function. The result of this simulation tool is a visual representation of a MZM transfer function in function of the bias with different amplification factors.

This link is composed by a MZM whose configuration (Push-Pull, Dual Drive and Dual Parallel) can be chosen by the user, at whose output we have an optic fiber that introduces dispersion, an amplifier to adjust the amplitude of the signal and then the photodetector that converts the signal from the optical domain to the electric domain. The figure 3.1 shows a scheme of the link just explained.

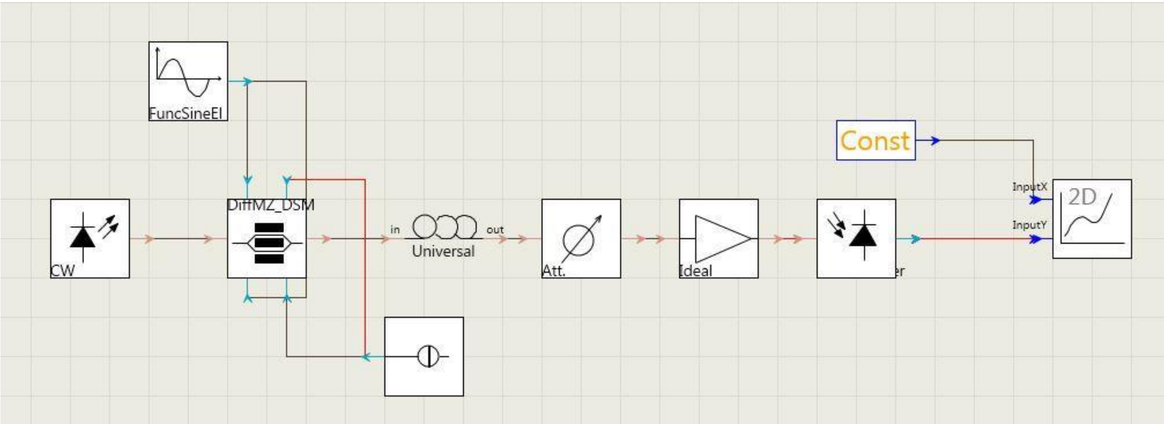



Figure 3.1: Schematic of MZM Transfer functions

The Live Script of this simulation tool consists in two different parts:

The first part (figure 3.2) is basically informative, there the user can read the objective of this simulation tool, a concise description of how it works and the schematic of the figure 3.1 so the user has a better idea of how it is going to work.

**Objective:**

The objective of this program is to recreate the fuctioning of a Mach-Zehnder modulator with three different configurations (Push-Pull, Dual Drive and Dual Parallel) and see their transfer functions depending on the bias voltage.

**Configuration:**

To fulfill the objective stated above, we need to introduce a pure tone as the input of the MZM, we are not using real data.

At the output we are connecting an optic fiber (which only effect considered is the dispersion), followed by an amplifier to control the level of power received by the photodetector.

Finally, we are plotting the signal received in function of the bias voltage of the MZM. Aditionally, this plot can be repeted with different levels of power at the photodetector.

**Schematic:**

In order to see graphically the configuration above and sum it up, we have created the following schematic using VPI:

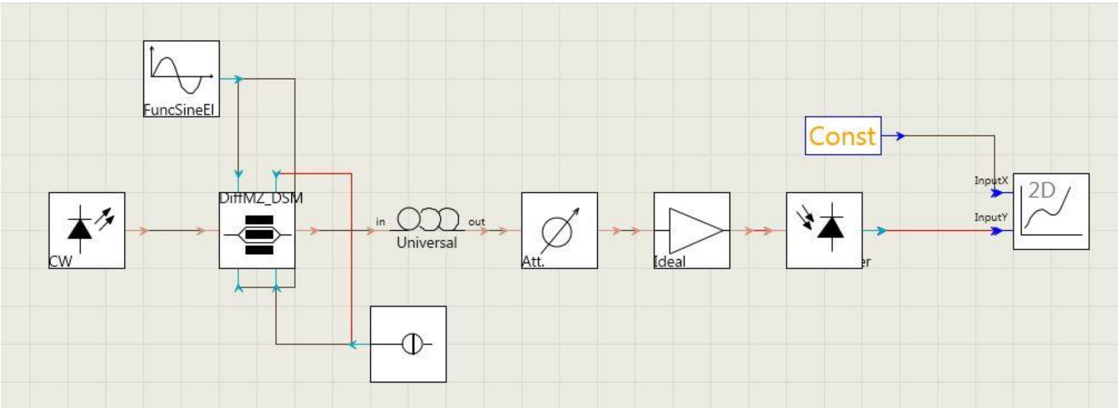



Figure 3.2: Objective, configuration and Schematic of the MZM Transfer function

The second part (figure 3.3) shows the code that is going to be executed, with a small description about the evolution of the signal through the functions.

The code starts initializing some variables needed (that can be changed depending of the preference of the user). After the user has chosen the configuration desired of the Mach-Zehnder modulator with a drop-down window, the code simulates the scenario explained before.

**Code:**

We start of loading the constants needed and initialising the program. Any changes in the configuration of the program has to be done in this function:

```
clc;
Initialize();
```

Then, we choose which configuration of the MZM we want to be working with:

```
E=MZM_DP();
```

The next step is to introduce the fiber, the amplifier that controls the output at the receiver and the photodector:

```
F=Fiber(E);
R=Amplifier(F);
I=Photodetector(R);
```

Figure 3.3: Code of MZM Transfer function

Finally, it represents the MZM transfer function obtained for the different bias and amplification factors at the end of the script. (figure 3.4)

Finally we launch the plot:

```
Output2D(I)
```

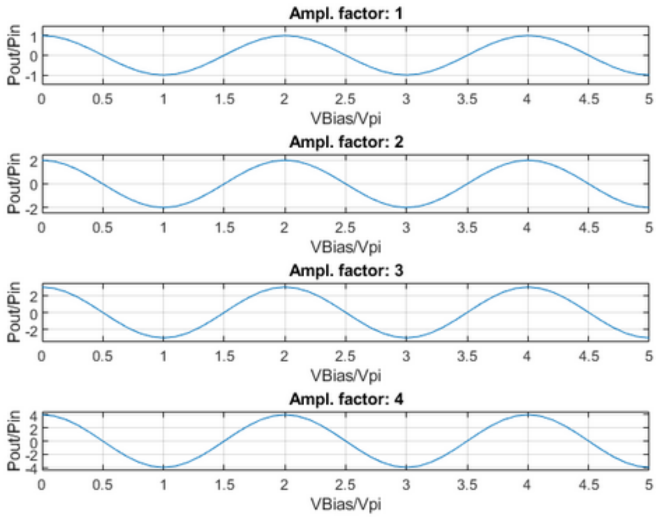



Figure 3.4: Results of the code: MZM Dual Parallel transfer functions with bias from 0 to 5 and 4 amplification factors from 1 to 4.

The whole Live Script can be seen in the Annex I.

### 3.1.2. Functions

This section explains the functions that appear in this software tool.

From the point of view of the user the most interesting part is the Initialize function. This function is the one used to change all the parameters that are going to interfere in the code. Some of the functions from Initialize, that the user could be interested in, are the following ones: vtheta_b (Voltage of the bias), ampli_factor (amplification factors).

Then there is the rest of the functions (MZM_PP, MZM_DD, MZM_DP, Fiber, Amplifier, Photodetector) that does the mathematical procedure of the respective component. It is not recommended to modify those functions.

The user can modify the Live Script, there he can choose which configuration of the MZM will be used and also can decide to add or remove the amplifier (not recommended).

### 3.1.3. Results

This section we are going to see how the change of some parameters, commented in the previous section 3.1.2, affects the code.

The default configuration that the user will have is the following one:

vtheta_b= [0:0.1:5]

ampli_factor= [1, 2]

E=MZM_DP()

With the result of the figure 3.5

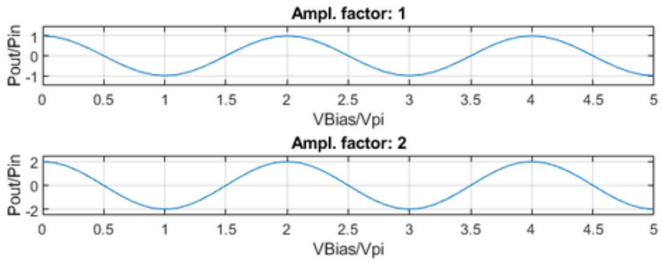



Figure 3.5: Result of vtheta_b=[0:0.1:5] and ampli_factor= [1, 2]

The principal parameter that can be changed is the configuration of the MZM in the Live Script using the drop down window. By doing this, the simulation tool will show the corresponding transfer function. (figure 3.6-3.8)

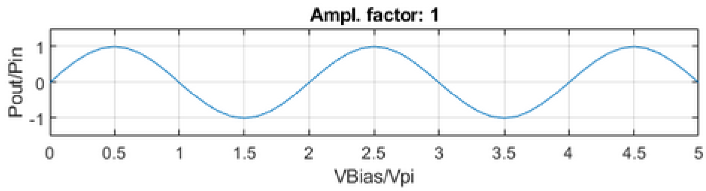



Figure 3.6: MZM_PP Transfer function

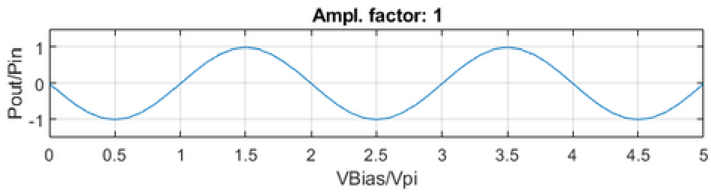



Figure 3.7: MZM_DD Transfer function

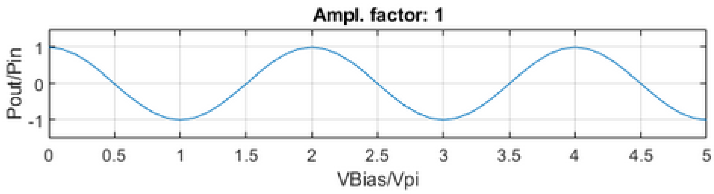



Figure 3.8: MZM_DP Transfer function

The first variable we are going to change is vtheta_b. This variable will affect the number of bias we are going to represent. For this parameter we can change the end of the results and the number of bias that are between them.

To start we are going to change the end of the result. Instead of going from 0 to 5 we will try going from 0 to 3 (figure 3.9)

vtheta_b= [0:0.1:3]

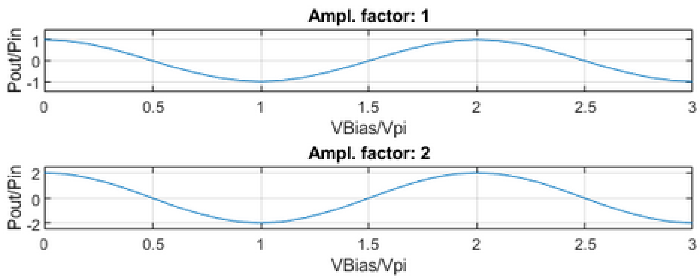



Figure 3.9: Result of vtheta_b= [0:0.1:3]

Now, instead of changing the ends we will see what happens (figure 3.10) when the number of bias is changed.

vtheta_b= [0:0.2:3]

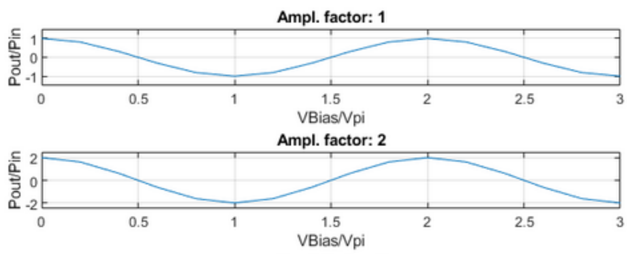



Figure 3.10: Result of vtheta_b= [0:0.2:3]

In this image, figure 3.10, we can see how the result is not as smooth as the one in figure 3.9, but it is still clear.

It is important that the first and last bias have a value distance of 2 in order to be able to see the whole period (figures 3.11 and 3.12).

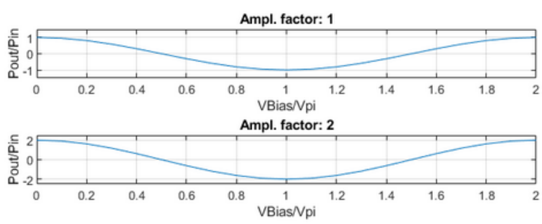



Figure 3.11: Result of vtheta_b= [0:0.1:2]

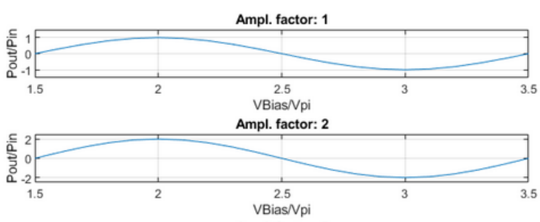



Figure 3.12: Result of vtheta_b= [1.5:0.1:3.5]

Also it is important to have a value between the bias lower than 0.3. Otherwise the result will lose its form. (figure 3.13)

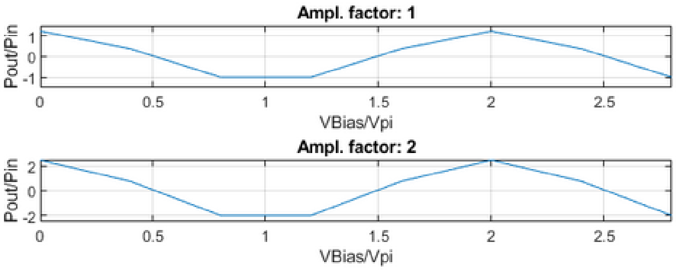



Figure 3.13: Result of vtheta_b= [0:0.4:3]

Finally, with the ampli_factor the user can choose different amplitudes for the MZM transfer functions. (figure 3.14)

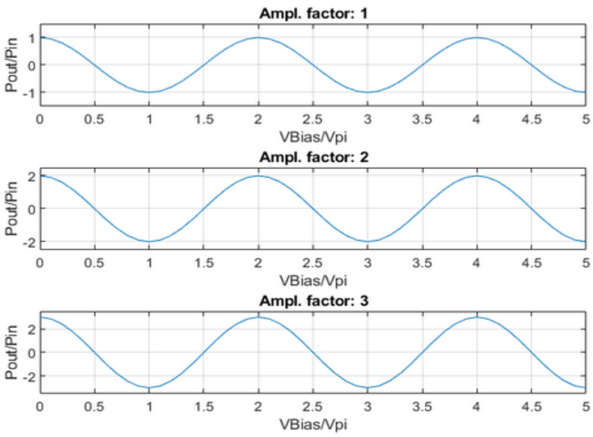



Figure 3.14: Results of ampli_factor= [1, 2, 3]

## 3.2. RoF Design System

### 3.2.1. Description

RoF Design System is an evolution of the MZM Transfer function, their objectives are not the same, while MZM Transfer function has as objective to study the functioning of a simple link, this software tool wants to study the evolution of the data through the transmission and the BER for different bias and values of amplification.

The execution of this software tool follows the scheme of the figure 3.15. It starts by generating what would be the sequence of bits (the information), then the process to transform those bits into a signal, separating them in in-phase and quadrature components. The next part is transmitting the signal through the link of the MZM Transfer function, to finally receive the signal with the noise, undo the transformation, get the original sequence of bits and then compare the result with the original one.

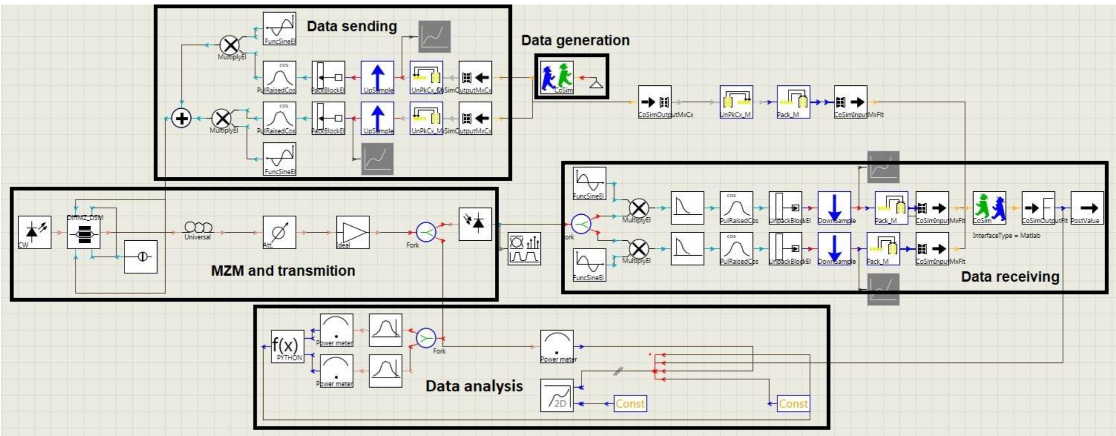



Figure 3.15: Scheme of RoF Design System

This Live Script also consist in two parts, the first one (figure 3.16) let the user read the objective, a concise description of how the different parts works and the schematic of the figure 3.15

**Objectives:**

The objective of this program is to recreate the functioning of an entire radio-over-fiber transmition system using a Mach-Zehnder modulator with three different configurations (Push-Pull, Dual Drive and Dual Parallel). Additionally, it allows the user to calculate the bit error rate in function of the bias used, and the input power of the detector.

**Configuration:**

**Data generation:**

With the objective of calculating the bit error rate, as stated above, we need to generate random bits and compare the originals with the ones received.

For this pourpose we are using a pseudoaletory bit generator which, once we have the vector of bits created, splits the bits in in-phase and quadrature in order to create a QPSK modulation.

**Digital demodulation:**

After reciving our electric signal, we need to demodulate the input in the in-phase and quadrature component. In order to do so, we will multiply the signal by a sine and a cosine, both with the same frequency that the one we used in the modulation process. After the multiplication, we will use a low-pass filter to eliminate the parasite frequencies generated in the process. One we have done this process, we have the I and Q signal again.

After reaching this point, we will reverse the process we did in the modulation: firstly, we will convolve the signal with the conformator pulse; then, we will downsample the vector, in order to convert it from a time vector to a bit one.

The last step is to use a decision to transform the "weak" 1's and 0's into "hard" ones, and -finally- we will reconstruct the PRBS signal sent with the I and Q vectors we have in reception.

**Data analysis:**

The final feature included in our system is to be able to compute and compare the BER using different bias voltage in our modulator and different photocurrents in the photodetector. This is done by itering the same program that the user can see in but in a particular function with the correspondet changes.

**Schemating:**

In order to see graphically the configuration above and sum it up, we have created the following schematic using VPI:

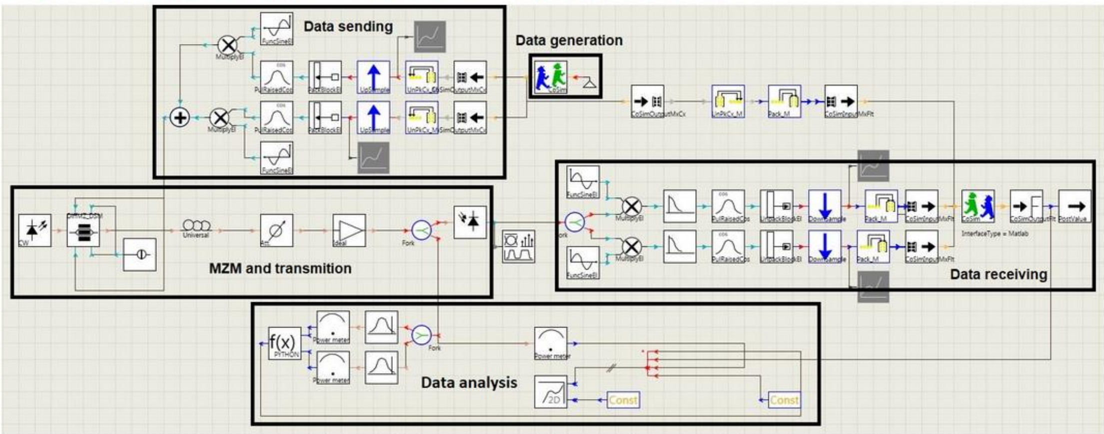



Figure 3.16: Objective, configuration and Schematic of the RoF Design System

The second part is the code where all the generation, transmission and reception of the data is made.

It starts by initializing the parameters necessary for the code. (figure 3.17)

```
Initialize();
load('variables.mat')
```

Figure 3.17: Initialize

### 3.2.1.1. Data generation

This part starts by creating the whole sequence of bits and then splitting them into in-phase (I) and quadrature (Q) following the 4 QAM codification. (figure 3.18)

```
[I,Q,PRBS]=IQcode();
Generator_plot(I,Q,PRBS);
```

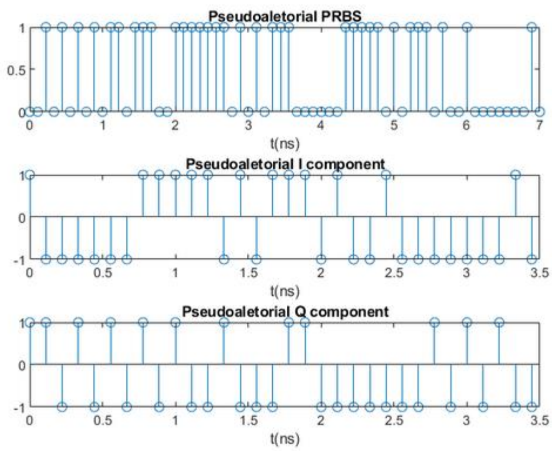



Figure 3.18: Generated sequence of bits

### 3.2.1.2. Digital modulation

Both sequences are up-sampled, this is done to be able to convolve with the shaping pulse that in our case is the squared root raised cosine set as default with common values. (figure 3.19 – 3.21)

```
I_up=upsample(I,sps);
Q_up=upsample(Q,sps);
IQ_plot(I_up,Q_up,1);
```

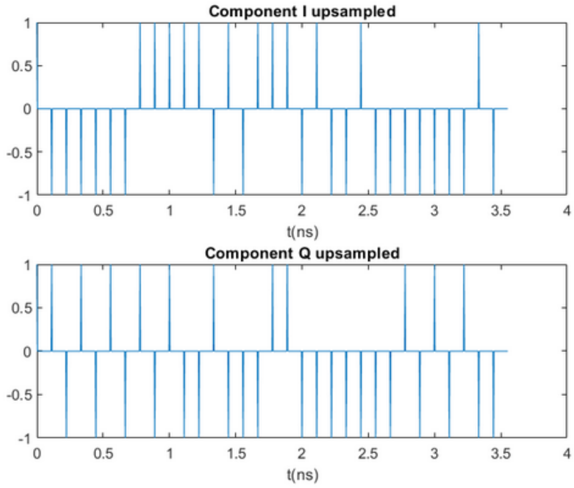



Figure 3.19: Up-sampled components

```
h=5*rcosdesign(rolloff,span,sps);
figure;
plot((0:length(h)-1)*Tsampling*10^9, h); xlabel("t (ns)"); ylabel("Amplitude (V)")
```

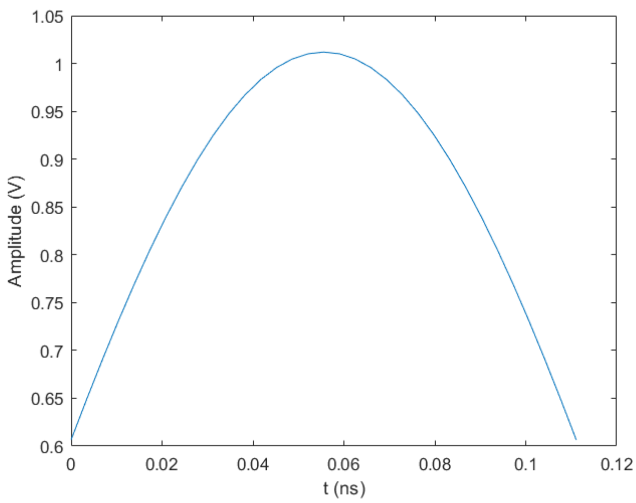



Figure 3.20: Squared root raised cosine

```
I_conv=conv(I_up,h);
Q_conv=conv(Q_up,h);
IQ_plot(I_conv,Q_conv,2);
```

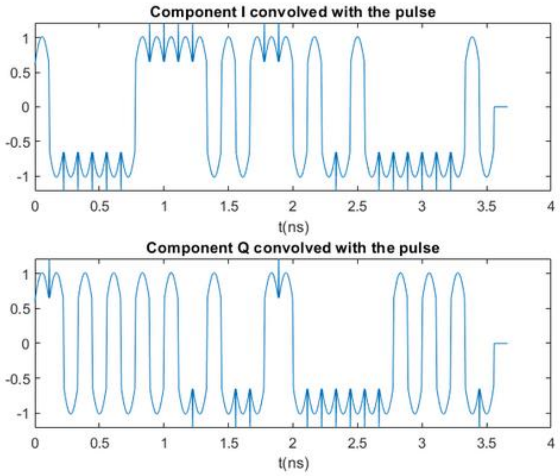



Figure 3.21: Convolution of squared root raised cosine and the up-sampled components

Finally, both signals are multiplied by a cosine and a sine (respectively) at $f_{RF}$ frequency. (figure 3.22)

```
t_conv=(0:paso:tiempo_total-paso);

I_in=I_conv.*cos(2*pi*f_rf*t_conv);
Q_in=-Q_conv.*sin(2*pi*f_rf*t_conv);
S_in=I_in+Q_in;
```

Figure 3.22: Convolution of squared root raised cosine and the up-sampled components

### 3.2.1.3. MZM and transmition

Once the signal is created, it is going to be introduced in the Mach-Zehnder modulator. The user can choose its configuration with a drop down window

```
I_mzm=MZM_PP_(S_in);
MZM_plot(I_mzm,1);
```

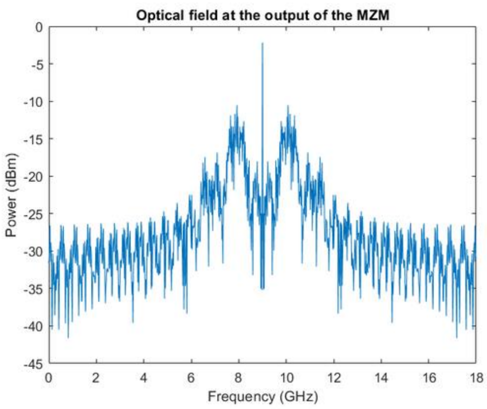



Figure 3.23: Optical field at the output of the MZM

The optical field obtained from the modulator (figure 3.23) is transmitted through the fiber (adding dispersion), amplified and then transformed to the electrical field by the photodetector (figure 3.24). At this point the user can decide if the system will have shot noise.

```
F=Fiber_(I_mzm);
A=Amplifier_(F);
```

```
S_out=Photodetector_(A);
S_out=Shot_Noise(S_out);
MZM_plot(S_out,2)
```

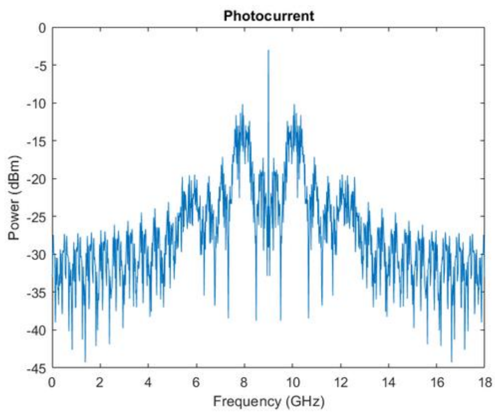



Figure 3.24: Photocurrent

### 3.2.1.4. Digital demodulation

Now with the actual signal, it is needed to do the digital demodulation. That is done by multiplying the signal again by the sine and the cosine and using a Chebyshev low-pass filter of 6$^{th}$ order to get the in-phase and quadrature components. (figure 3.25)

```
I_out=2.*S_out.*cos(2*pi*f_rf*t_conv);
Q_out=-2.*S_out.*sin(2*pi*f_rf*t_conv);
```

```
[b,a] = cheby1(6,10,pass_band);
I_fil = filter(b,a,I_out);
Q_fil = filter(b,a,Q_out);
Espect_plot(I_fil,Q_fil,I_out,Q_out);
```

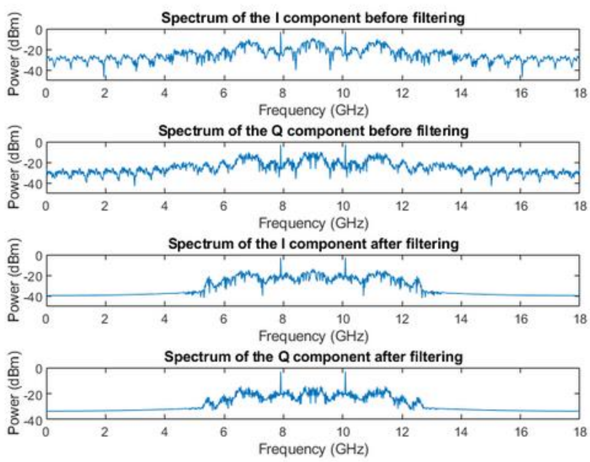



Figure 3.25: Spectrum of the components before and after filtering

Once the signal of interest in both components is obtained, the signal is convolved with the pulse and down-sampled to get the two original components. (figure 3.26 and 3.27)

```
I_des=conv(I_fil,h);
Q_des=conv(Q_fil,h);
IQ_plot(I_des,Q_des,3);
```

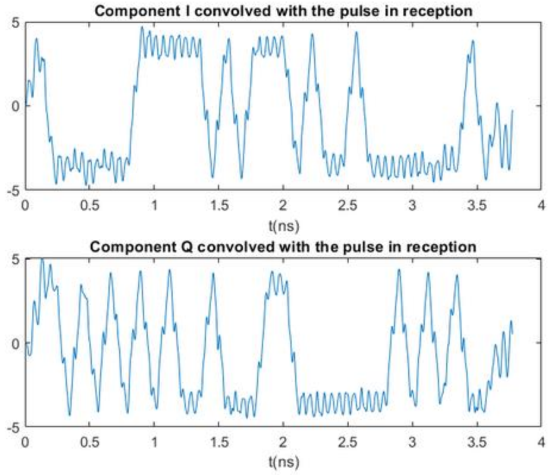



Figure 3.26: Spectrum of the components before and after filtering

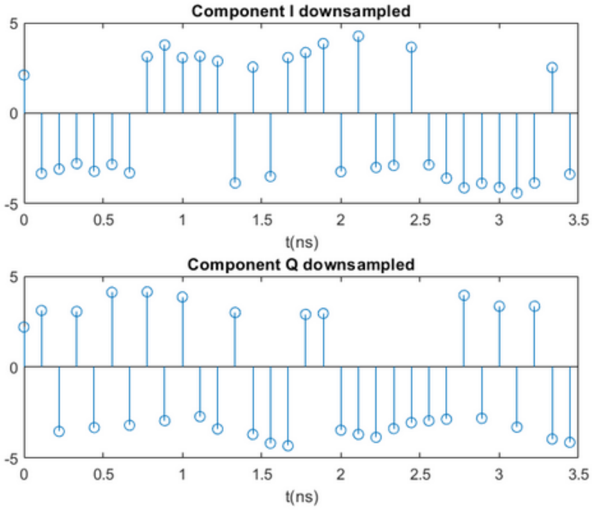



Figure 3.27: Components after convolution and down-sampling

At this point the user has to choose if the system has thermal noise.

Due to the filtering, the dispersion and the noise the sequences do not have a strong 1 or -1, to correct this there is a Decisor which using the 0 as a threshold decides is a sample is a 1 or -1. With the hard 1 and -1, the original sequence of bits can be obtained. (figure 3.28)

```
I_down=Thermal_Noise(I_...;
Q_down=Thermal_Noise(Q...;
```

```
I_des=Decisor(I_down);
Q_des=Decisor(Q_down);
PRBS_=IQdecode(I_des,Q_des);
PRBS_plot(PRBS,PRBS_);
```

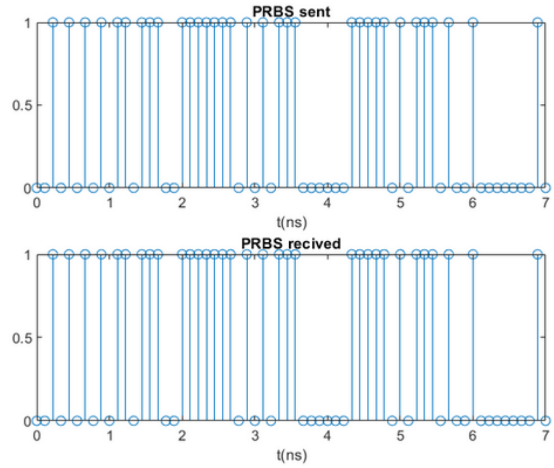



Figure 3.28: PRBS sent and received

#### 3.2.1.5. Data analysis

Since the signal has been modified through the transmission it could have some errors, so there is a BER value (figure 3.29) to indicate how the received sequence is changed from the original one.

```
BER=BerCalculator(PRBS,PRBS_)
BER = 0
```

Figure 3.29: BER

At the end of the code we can see a comparison table (figure 3.30) where the BER is shown for different combinations of the bias and amplification factors.

```
MZM="PushPull";
BER=BerComparator(MZM)
BER = 10×3 string array

"Bias"     "Amplification factor"    "BER"
"0.5"      "5e-07"                   "0.99632"
"0.5"      "6e-07"                   "0.99868"
"0.5"      "7e-07"                   "0.9995"
"1"        "5e-07"                   "0.50179"
"1"        "6e-07"                   "0.50153"
"1"        "7e-07"                   "0.50116"
"1.5"      "5e-07"                   "0.0036011"
"1.5"      "6e-07"                   "0.0013924"
"1.5"      "7e-07"                   "0.00051117"
```

Figure 3.30: BER Comparison table

The whole Live Script can be seen in the Annex II

### 3.2.2. Functions

RoF Design system has different groups of functions: basic functions where are some funtions to have a clearer Live Script. There are also functions dedicated to the transmission part, the data part, the BER and the noise.

For the common functions the user would be more interest in the Initialize. This function has the purpose to create all the variables for the code. Some of these variables, such as number of bits and the parameter of the filter, can be changed to create the configuration desired.

The transmission functions contain the functions of the MZM Transfer functions (Section 3.1) but instead of using a pure RF signal, they accept the signal created with the data.

The functions used for the data part are focused in creating the information to later modulate and demodulate it.

### 3.2.3. Results

This section is going to show how the change of the different parameters affects in the transmission and the different options the simulation tool gives.

The first parameter is Nbits (number of bits). This parameter decides the number of bits that are going to be used as information in the transmission. The number of bits to simulate depends on the range of the BER values from enough errors, basically around 100. That means that when looking for BERs on the order of $10^{-3}$ you need on the order of $10^{5}$ bits.

The f (frequency) decides the frequency of the bits.

The relation between frequency and number of bits decides the time window. From the figure 3.31 to 3.34 we can see 4 cases with the combination of number of bits and frequency with two values for each of them. Also comparing the figure 3.31 and figure 3.34 we can see that if we maintain the relation 133.78bits/GHz the time window will always have the same value.

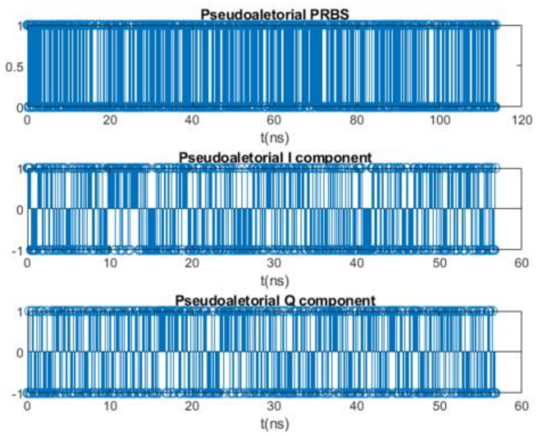



Figure 3.31: 1024 bits and 9GHz

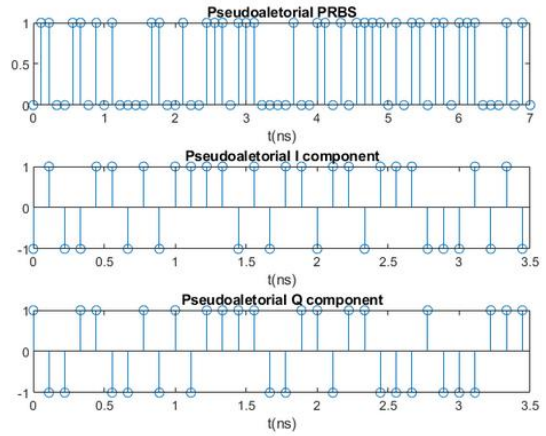



Figure 3.32: 64 bits and 9GHz

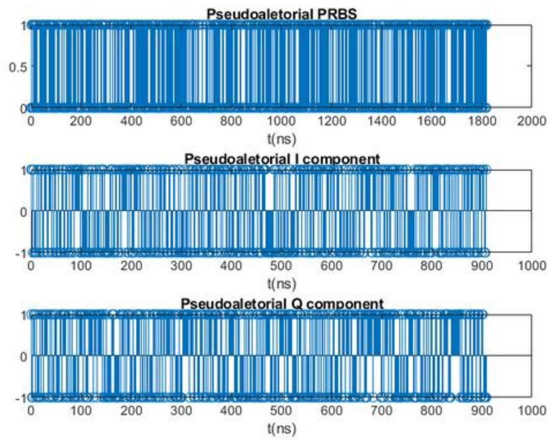



Figure 3.33: 1024 bits and 0.5625GHz

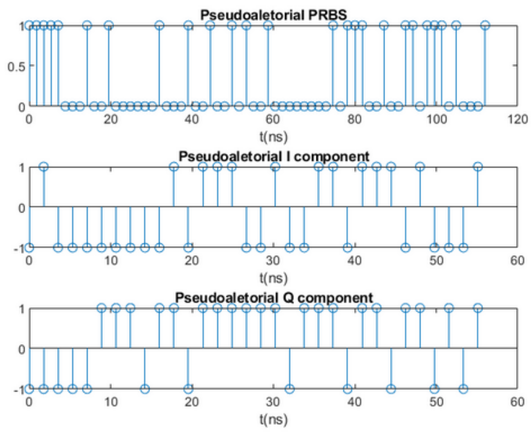



Figure 3.34: 64 bits and 0.5625GHz

The next parameter is sps. This value represents the number of zero samples that are going to be introduced between the bits. It also decides the number of samples in which the squared root raised cosine is going to be represented. It is recommended to not reduce the value of this parameter since it would mean a loss of information of the raised cosine. However, it can be done if the velocity of the result is more important that the quality. In the figure 3.35 and 3.36 we can see how the change of this parameters affects the raised cosine.

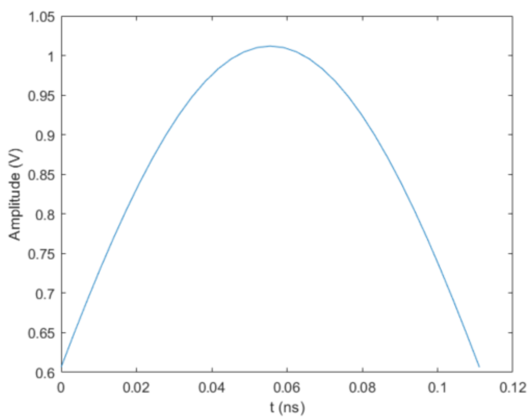



Figure 3.35: Square root raised cosine with sps=32

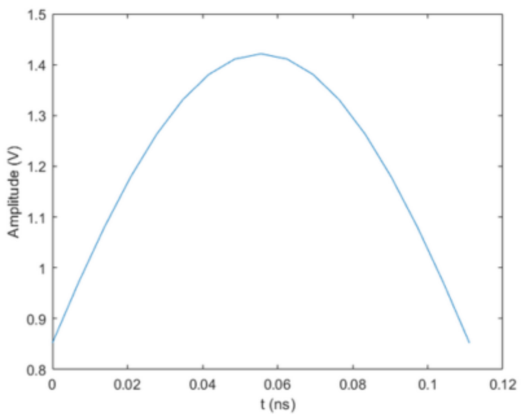



Figure 3.36: Square root raised cosine with sps=16

The two following variables are the ones that appeared in the MZM Transfer function, the vtheta_b and the ampli_factor, those variables save the values of the bias and the values that the signal would get at the end of the photodetector. For every combination of bias and amplification factor the code will give us a result. In the figure 3.37 and figure 3.38 we can see the results for two different configurations.

```
BER = 10×3 string array
    "Bias"    "Amplification factor"    "BER"
    "0.5"     "1"                       "1"
    "0.5"     "2"                       "1"
    "0.5"     "3"                       "1"
    "1"       "1"                       "0.49121"
    "1"       "2"                       "0.49121"
    "1"       "3"                       "0.49121"
    "1.5"     "1"                       "0"
    "1.5"     "2"                       "0"
    "1.5"     "3"                       "0"
```

Figure 3.37: Square root raised cosine with sps=32

```
BER = 7×3 string array
    "Bias"    "Amplification factor"    "BER"
    "0.5"     "5e-06"                   "1"
    "0.5"     "6e-06"                   "1"
    "0.5"     "7e-06"                   "1"
    "1"       "5e-06"                   "0.52344"
    "1"       "6e-06"                   "0.5293"
    "1"       "7e-06"                   "0.53027"
```

Figure 3.38: Square root raised cosine with sps=16

Finally, the last parameter that can be changed is in the Live Script. That is the noise that is going to be added to the system. The user in different points of the code can decide if he wants a specific type of noise (Shot and thermal noise) by choosing it with a drop-down window. From the figure 3.39 to figure 3.42 we are going to see 4 different scenarios where the noise affects in the code.

```
BER = 10×3 string array
    "Bias"    "Amplification factor"    "BER"
    "0.5"     "2e-07"                   "1"
    "0.5"     "3e-07"                   "1"
    "0.5"     "4e-07"                   "1"
    "1"       "2e-07"                   "0.50391"
    "1"       "3e-07"                   "0.50391"
    "1"       "4e-07"                   "0.50391"
    "1.5"     "2e-07"                   "0"
    "1.5"     "3e-07"                   "0"
    "1.5"     "4e-07"                   "0"
```

Figure 3.39: Without noise

```
BER = 10×3 string array
    "Bias"    "Amplification factor"    "BER"
    "0.5"     "2e-07"                   "0.98242"
    "0.5"     "3e-07"                   "0.99805"
    "0.5"     "4e-07"                   "1"
    "1"       "2e-07"                   "0.51563"
    "1"       "3e-07"                   "0.49316"
    "1"       "4e-07"                   "0.49707"
    "1.5"     "2e-07"                   "0.018555"
    "1.5"     "3e-07"                   "0.00097656"
    "1.5"     "4e-07"                   "0"
```

Figure 3.40: Shot &Thermal Noise

```
BER = 10×3 string array
    "Bias"    "Amplification factor"    "BER"
    "0.5"     "2e-07"                   "1"
    "0.5"     "3e-07"                   "1"
    "0.5"     "4e-07"                   "1"
    "1"       "2e-07"                   "0.50781"
    "1"       "3e-07"                   "0.5"
    "1"       "4e-07"                   "0.50195"
    "1.5"     "2e-07"                   "0"
    "1.5"     "3e-07"                   "0"
    "1.5"     "4e-07"                   "0"
```

Figure 3.41: Shot noise

```
BER = 10×3 string array
    "Bias"    "Amplification factor"    "BER"
    "0.5"     "2e-07"                   "0.98047"
    "0.5"     "3e-07"                   "1"
    "0.5"     "4e-07"                   "1"
    "1"       "2e-07"                   "0.51465"
    "1"       "3e-07"                   "0.49902"
    "1"       "4e-07"                   "0.48926"
    "1.5"     "2e-07"                   "0.024414"
    "1.5"     "3e-07"                   "0.0019531"
    "1.5"     "4e-07"                   "0"
```

Figure 3.42: Thermal Noise

# 4. Conclusions and Future Work

As a concise conclusion, since the main objectives of this thesis were to study the components of a RoF System and create a simulation tool (similar to VPIphotonics) to simulate those systems, it can be said that those objectives have been achieved.

The Chapter 2, the theoretical part, is focused in explaining (conceptually and mathematically) the components that are used to create the system in MZM Transfer function and RoF Design system in order to understand how these simulation tools work. Additionally, with eyes to future developments, we have provided an analysis of components which will be useful to include into the RoF simulation tool.

In Chapter 3 the main results of this thesis are presented. The MZM Transfer function is a software tool that analyses the transfer function of the three configurations of MZM (Push-Pull, Dual-Drive and Dual-Parallel). The purpose of this software tool is to represent the real system that would be used to obtain that MZM transfer function. The other software tool presented is the RoF Design System. This software simulates a link that creates, transmits and receives data with two purposes. The first purpose is to show the evolution of the different stages of the data through the system such as the sequence of bits created, the signal result of the modulation, the optical field given by the MZM and the signal received before and after the processing. The second purpose is to make a study of the BER for different values of bias and amplification factors chosen by the user. These result appear in a comparison table.

Comparing this simulation tool with the one used as a reference and the most commercial simulation tool used, VPIphotonics, two conclusions can be made:

- Our simulation tool is much faster than the VPI due to its simplicity. The VPI photonics' blocks can be configured with a lot of parameters. Since our code has a clear objective, the number of parameters are reduced and the unnecessary calculations are avoided.
- The results of our simulation tool in terms of BER and optical power are similar to the ones of VPI. For example, for a BER of $10^{-3}$ the VPI need an optical power of $10^{-6}$ W set by the EDFA. Our code can get this BER with an optical power of $6.25 \cdot 10^{-7}$W. Due the difference of the simulation tools it is normal to assume that the result will not be the same, but the magnitude of the results is accurate.

It goes without saying that this simulation tool can be evolved in different ways, it could introduce more parameters to change, it could be added more functions that works as a blocks and also modify the actual functions in order to get another interesting results. Another parameter interesting to add is the ASE noise, the one introduced by the EDFA, that would limit the capacity of amplification of the EDFA and give a more realistic solution.

# References

[1] Vega, S. Orientación de antenas mediante técnicas fotónicas en sistemas inalámbricos de radiofrecuencia. Degree Thesis, Universitat Politècnica de Catalunya, Spain, July 2018.

[2] Aabouch, K. Strategies for Efficient Fibre Wireless Networks (FiWiN)". Master Thesis, Universitat Politècnica de Catalunya, Spain, January 2021.

[3] Ucar, Y. Design and testing of microwave circuits for optimized R-o-F data transmission. Master Thesis, Universitat Politècnica de Catalunya, Spain, February 2019.

[4] Vergés, R. Sistema de generació i recepció per a la mesura de senyals de RF amb modulacions i codificacions digitals. Degree Thesis, Universitat Politècnica de Catalunya, Spain, June 2019.

[5] Rusch, L. *Simulation of Optical Communications Systems: Noise* [online]. Available at: https://n9.cl/gti7c

[6] Rusch, L. *Homework Set B - Matlab* [online]. Available at: https://n9.cl/unjqz

# 5. Annexes

## 5.1. Annex I: Live Script of MZM Transfer function

### MZM: Transfer function

#### Objective:

The objective of this program is to recreate the fuctioning of a Mach-Zehnder modulator with three different configurations (Push-Pull, Dual Drive and Dual Parallel) and see their transfer functions depending on the bias voltage.

#### Configuration:

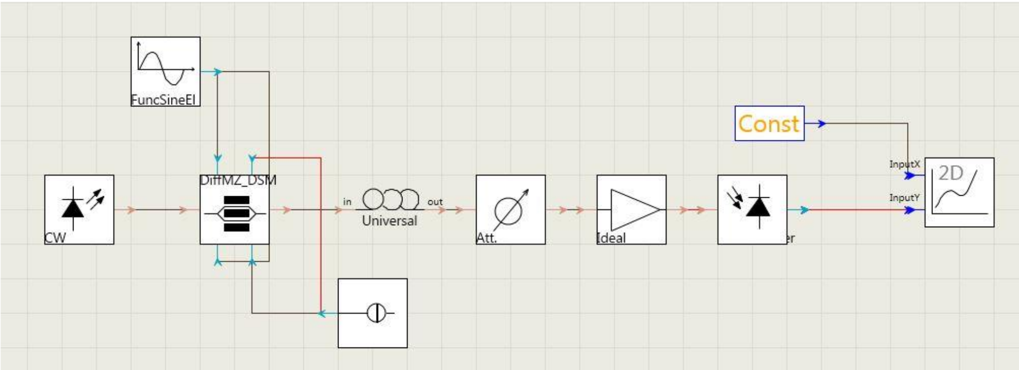
To fulfill the objective stated above, we need to introduce a pure tone as the input of the MZM, we are not using real data.

At the output we are connecting an optic fiber (which only effect considered is the dispersion), followed by an amplifier to control the level of power received by the photodetector.

Finally, we are plotting the signal received in function of the bias voltage of the MZM. Aditionally, this plot can be repeted with different levels of power at the photodetector.

#### Schematic:

In order to see graphically the configuration above and sum it up, we have created the following schematic using VPI:




#### Code:

We start of loading the constants needed and initialising the program. Any changes in the configuration of the program has to be done in this function:

```
clc;
Initialize();
```

Then, we choose which configuration of the MZM we want to be working with:

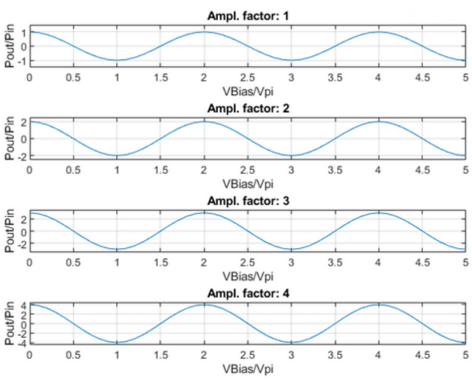
```
E= MZM_DP() ;
```

The next step is to introduce the fiber, the amplifier that controls the output at the receiver and the photodector:

```
F=Fiber(E);
R=Amplifier(F);
I=Photodetector(R);
```

Finally we launch the plot:

```
Output2D(I)
```



# 5.2. Annex II: Live Script of RoF Design System

## MZM: Full transmition system

### Objectives:

The objective of this program is to recreate the functioning of an entire radio-over-fiber transmition system using a Mach-Zehnder modulator with three different configurations (Push-Pull, Dual Drive and Dual Parallel). Additionally, it allows the user to calculate the bit error rate in function of the bias used, and the input power of the detector.

### Configuration:

**Data generation:**

With the objective of calculating the bit error rate, as stated above, we need to generate random bits and compare the originals with the ones received.

For this pourpose we are using a pseudoaletory bit generator which, once we have the vector of bits created, splits the bits in in-phase and quadrature in order to create a QPSK modulation.

**Digital modulation:**

Once we have our vectors of pseudoaletory bits in in-phase and quadrature created, we need to implement the data processing necessary in order to create a valid input for the MZM.

The first step is to do an up sampling of the bits, this is done to convert the bits vector into a temporal one. Then, we need to mulitiply in frequency (or convolve in time) the time vector with a rised consine filter in order to convert time deltas into pulses. Finally, the in-phase vector will be modulated with a cosine, the quadrature one with a sine and both will be added together. This will be the electric signal that we will use as an input for our MZM modulator.

**MZM and transmition:**

Once we have our vector created (an electric signal, actually), it will be introduced in the MZM modullator (with any of the aforementioned three configurations).

At the output we are connecting an optic fiber (the only considered effect of whch is the dispersion), followed by an amplifier to control the level of power received by the photodetector.

**Digital demodulation:**

After reciving our electric signal, we need to demodulate the input in the in-phase and quadrature component. In order to do so, we will multiply the signal by a sine and a cosine, both with the same frequency that the one we used in the modulation process. After the multiplication, we will use a low-pass filter to eliminate the parasite frequencies generated in the process. One we have done this process, we have the I and Q signal again.

After reaching this point, we will reverse the process we did in the modulation: firstly, we will convolve the signal with the conformator pulse; then, we will downsample the vector, in order to convert it from a time vector to a bit one.

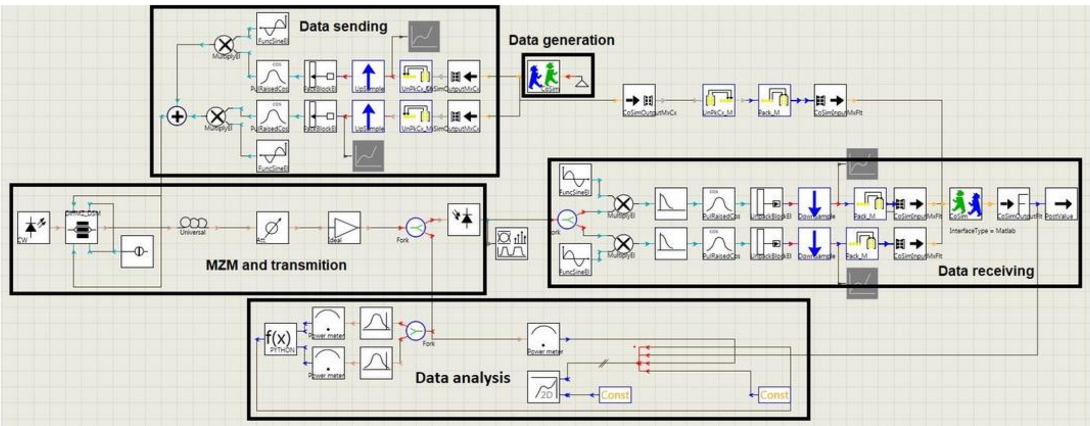
The last step is to use a decision to transform the "weak" 1's and 0's into "hard" ones, and -finally- we will reconstruct the PRBS signal sent with the I and Q vectors we have in reception.

**Data analysis:**

The final feature included in our system is to be able to compute and compare the BER using different bias voltage in our modulator and different photocurrents in the photodetector. This is done by itering the same program that the user can see in but in a particular function with the correspondet changes.

### Schemating:

In order to see graphically the configuration above and sum it up, we have created the following schematic using VPI:



## Code:

First of all, we need to comment a previous consideration: all the figures shown in this live script have been computed with the MZMs in quadrature point and an amplification of the optical signal of x1. To see the results using different bias voltage and photocurrent, go to the last section "Data analysis", where the process is repeated and calculated itering the parameters commented above.

Before starting our system description, we will call the initialise function that contains all the constants needed:

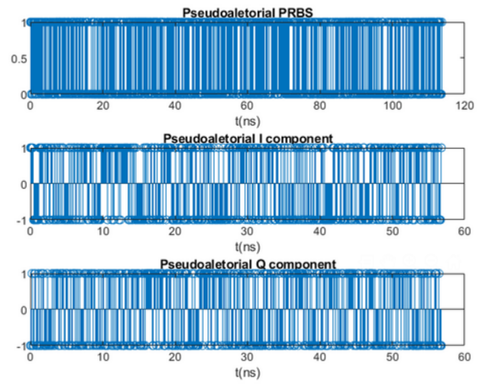
```
clear all;
Initialize();
load('variables.mat')
```

**Data generation:**

First of all, we use this function to generate the random bit sequence (PRBS) and split it in in-phase and quadrature.

This function has some internal arguments required: the number of bits, the bits per symbol and the amplitude of these bits. These can be changed in the initialize function.

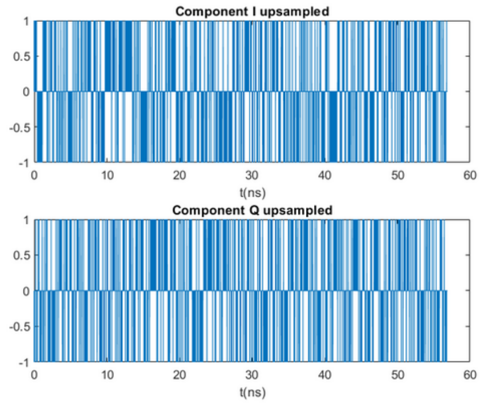
```
[I,Q,PRBS]=IQcode();
Generator_plot(I,Q,PRBS);
```




**Digital modulation:**

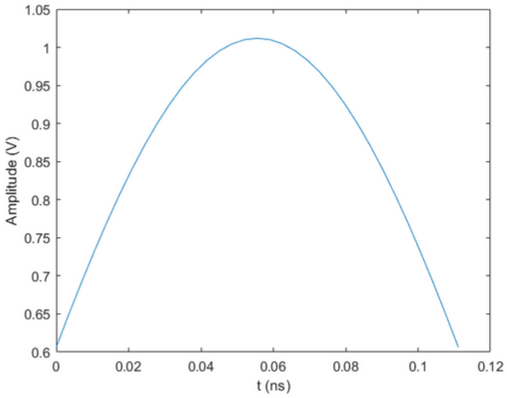
In order to upsample the I and Q vector generated we will execute the following Matlab native function (the argument required is the number of samples -zeros- between bits):

```
I_up=upsample(I,sps);
Q_up=upsample(Q,sps);
IQ_plot(I_up,Q_up,1);
```



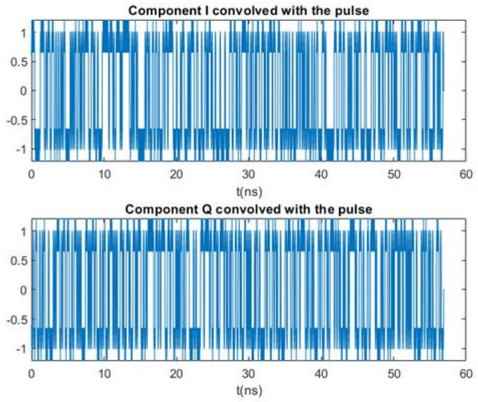

The following step is to create the conformator pulse. The arguments needed are: the roll-off, the number of symbols and the amount of samples per symbol.

```
h=5*rcosdesign(rolloff,span,sps);
figure;
plot((0:length(h)-1)*Tsampling*10^9, h); xlabel("t (ns)"); ylabel("Amplitude (V)")
```




Once it is created, we will convolve our I and Q vector with it :

```
I_conv=conv(I_up,h);
Q_conv=conv(Q_up,h);
IQ_plot(I_conv,Q_conv,2);
```




To end this part, we just need to modulate de in-phase component with a cosine, the quadrature with a sine and add them together in order to create a valid input for our MZM.

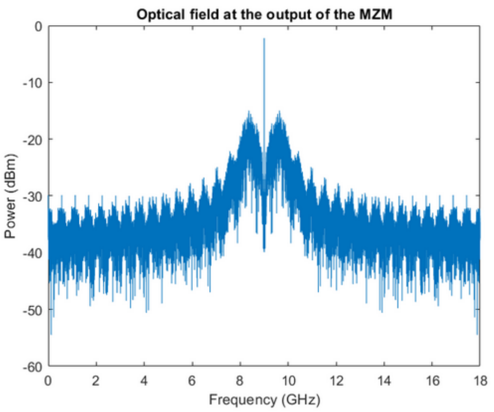
```
t_conv=(0:paso:tiempo_total-paso);

I_in=I_conv.*cos(2*pi*f_rf*t_conv);
Q_in=-Q_conv.*sin(2*pi*f_rf*t_conv);
S_in=I_in+Q_in;
```

**MZM and transmition:**

Firstly, we introduce the electric signal (S_in) in any of the three configurations of the MZM:

```
I_mzm=MZM_PP_(S_in);
MZM_plot(I_mzm,1);
```

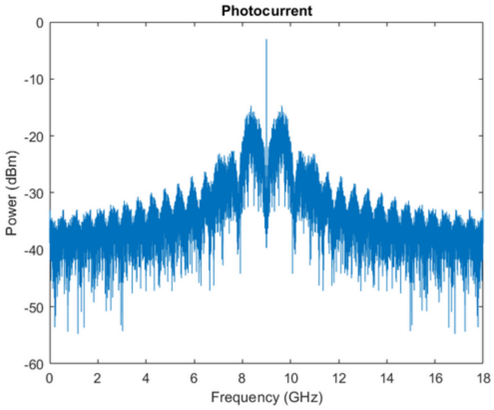



The optical output field will be sent through a fiber and amplified in reception so that the detector receives a normalised optical signal:

```
F=Fiber_(I_mzm);
A=Amplifier_(F);
```

Finally, the photodetector transforms the optical input field into an electrical one so that it can be demodulated. At this point the user all has to decide if it wants to introduce the shot noise of the photodetector:

```
S_out=Photodetector_(A);
S_out=Shot_Noise(S_out);
MZM_plot(S_out,2)
```



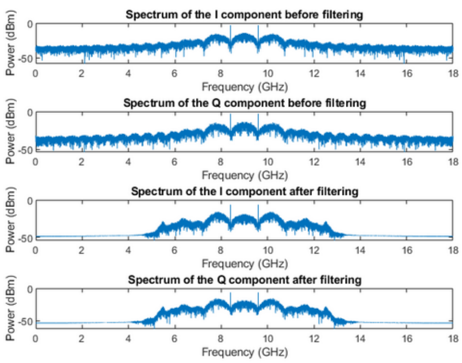

**Digital demodulation:**

First of all, we want to separete the signal in the in-phase and quadrature components. To do so, we need to multiply the signal with a cosine and a sine, both with the same frequency that the one we used in the sender to modulate them:

```
I_out=2.*S_out.*cos(2*pi*f_rf*t_conv);
Q_out=-2.*S_out.*sin(2*pi*f_rf*t_conv);
```

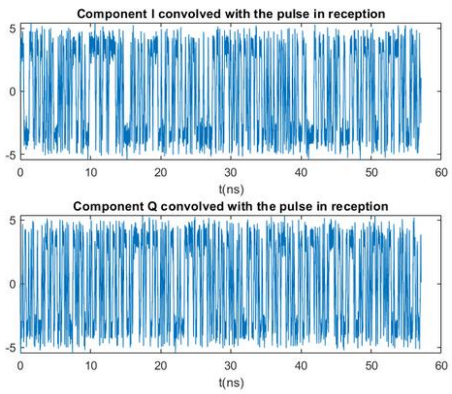
The second step is to use a low pass filter, to eliminate the parasite frequencies. The parameters needed to design the filter are: the order, the power in dBs admited in the filtered part and the border frequency of the pass band (normalized):

```
[b,a] = cheby1(6,10,pass_band);
I_fil = filter(b,a,I_out);
Q_fil = filter(b,a,Q_out);
Espect_plot(I_fil,Q_fil,I_out,Q_out);
```



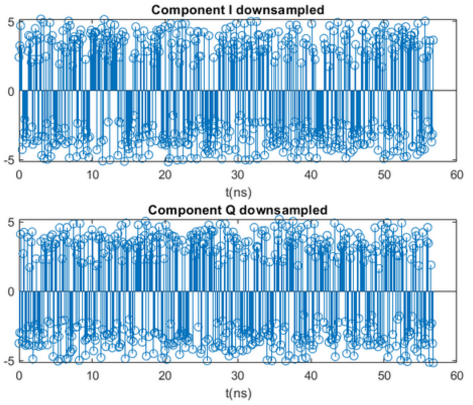

Once we have perfemored the low-pass filtering, we will convolve the vectors with the conformator pulse again:

```
I_des=conv(I_fil,h);
Q_des=conv(Q_fil,h);
IQ_plot(I_des,Q_des,3);
```



The following step is to downsample the signal again in order to get the original bits sent. The arguments needed in the downspampling are the same used in the upsampling:

```
I_down=downsample(I_des,sps);
I_down=I_down(2:1:length(I_down)-1);
Q_down=downsample(Q_des,sps);
Q_down=Q_down(2:1:length(Q_down)-1);
IQ_plot(I_down,Q_down,4);
```

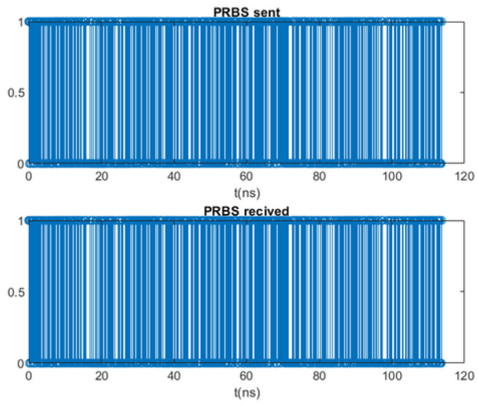



At this point, we can decide if we want to take into account the thermal noise introduced along all the system:

```
I_down=Thermal_Noise(I_...;
Q_down=Thermal_Noise(Q...;
```

Finally, we will introduce our components in a decisor bloc to obtain strongs '1' and '0' and recompose the PRBS sequency with the I and Q components received. For the ease of the user, we can also compare the PRBS vector generated in reception with the one pseudoaleatorally generated sent:

```
I_des=Decisor(I_down);
Q_des=Decisor(Q_down);
PRBS_=IQdecode(I_des,Q_des);
PRBS_plot(PRBS,PRBS_);
```




**Data analysis:**

```
BER=BerCalculator(PRBS,PRBS_)
```

```
BER = 0
```

```
MZM="PushPull";
BER=BerComparator(MZM)
```

```
BER = 10×3 string array
    "Bias"    "Amplification factor"    "BER"
    "0.5"     "5e-07"                   "0.99632"
    "0.5"     "6e-07"                   "0.99868"
    "0.5"     "7e-07"                   "0.9995"
    "1"       "5e-07"                   "0.50179"
    "1"       "6e-07"                   "0.50153"
    "1"       "7e-07"                   "0.50116"
    "1.5"     "5e-07"                   "0.0036011"
    "1.5"     "6e-07"                   "0.0013924"
    "1.5"     "7e-07"                   "0.00051117"
```